Create software deposit in HAL

Morane Gruenpeter, Jozefina Sadowska, Estelle Nivault, Alain Monteil

Create software deposit in HAL

User guide and best practices

Version 2.0 (March 2022)

Authors :

- Morane Gruenpeter
  morane@softwareheritage.org
- Jozefina Sadowska
  jozefina.sadowska@inria.fr
- Estelle Nivault
  estelle.nivault@inria.fr
- Alain Monteil
  alain.monteil@inria.fr

HAL open science

CCSD

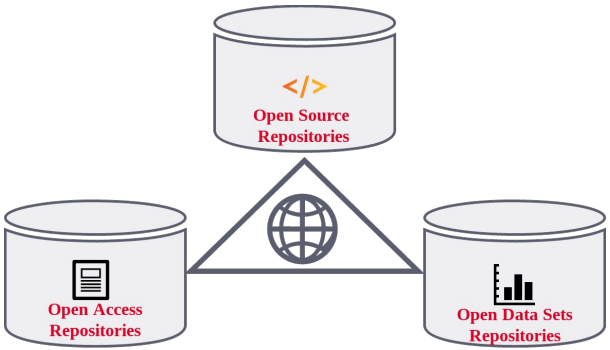
Inria

Software Heritage

Table of contents: Create software deposit in HAL

Introduction: why deposit on HAL?

- ★ Archive software in HAL and in SWH
  - Better visibility for software in open science context
  - Long term preservation HAL and SWH have a long term preservation service
- ★ Identify
  - the software artifacts with a **SWHID** (SoftWare Heritage Identifier)
  - the metadata record and citation with the **HAL-ID**
- ★ Describe source code with verified metadata
  - Moderation and control of the metadata by librarians and/or curators
- ★ Cite the software deposit with a complete citation
  - Several exports format make it easier for citation



The three pillars of Open Science, Software Heritage CC-By 4.0 2019

Introduction: What software object to deposit in HAL?

- ★ The source code of the software (not compatible for executables)
- ★ Software that was developed in academia for research purposes
- ★ Only the software creators/authors of the software or their representatives can deposit software in HAL

Use cases

I **develop** my software **locally**
And I share my code on my personnel website or my institution's website

Local method: deposit .zip /.tar.gz

I **develop** my software on a **collaborative plateforme** using a **version control system** (on GitHub, GitLab, Bitbucket...)

SWHID method: deposit SWHID with metadata

Introduction: The deposit steps

- ➔ Deposit on HAL
  - ◆ Local method or the SWHID method
- ➔ Validation of the form by the contributor
- ➔ Deposit in progress waiting for verification
  - ◆ Dialogue between contributor and moderator
- ➔ Validation of the deposit by the moderator
- ➔ Deposit published on HAL and transfer to SWH
- ➔ Export formats
  - ◆ Citation
  - ◆ BibTeX
  - ◆ codemeta.json
  - ◆ TEI

Checklist depending on use case

Local method: deposit .zip /.tar.gz

1. **Prepare your source code (locally)**
   - ❑ AUTHORS, LICENSE & README files
   - ❑ Compress documents into .zip / .tar.gz
2. **Deposit compressed archive**
3. **Complete metadata**
   - ❑ Choose deposit type
   - ❑ Add generic metadata
   - ❑ Add software specific metadata
   - ❑ Add authors
   - ❑ Validate deposit

SWHID method: deposit SWHID with metadata

1. **Prepare your source code (in code repository)**
   - ❑ AUTHORS, LICENSE & README files
   - ❑ Codemeta.json file
   - ❑ Save code now on Software Heritage
   - ❑ Choose SoftWare Heritage IDentifier (SWHID)
2. **Deposit SWHID in HAL interface**
3. **Complete metadata**
   - ❑ Add domain
   - ❑ Control entries from codemeta
   - ❑ Add authors
   - ❑ Validate deposit

Chapter 1:
Prepare the source code for archival

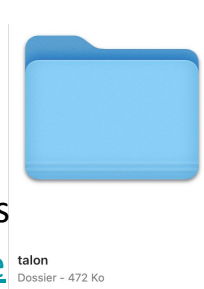
1.1 Prepare your code

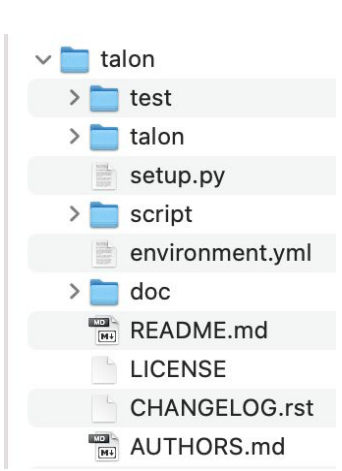
local method: deposit .zip /.tar.gz

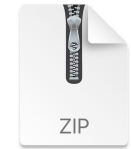
1.1.1 Add the following files :

these elements are verified by the moderators

- ❑ README
- ❑ AUTHORS
- ❑ LICENSE (Choose with the right holders of the software - Here you can find the SPDX reference list of licenses)







Updating content on the version control system

1.1.2 **Local method:** create a compressed archive

.zip ou .tar.gz

It is preferable to name the compressed file with the software name and version number

1.1.3 **SWHID method:** use your version control system

SWHID method: deposit SWHID with metadata

1.1.1 README file

- ★ Mandatory:
  - Software / project name
  - Description of the software
- ★ Recommended:
  - Website
  - Link to the documentation
  - Contact & support
  - List of functionalities
  - Development environment
    - build, installation, requirements
    - How to execute the code
- ★ Possible:
  - Usage - How to use the code
  - News about the project
  - Visuals

References: Software Release Practice HOWTO by Eric Steven Raymond and Make a README

```
This is Python version 3.8.0 alpha 0
====================================

.. image:: https://travis-ci.org/python/cpython.svg?branch=master
   :alt: CPython build status on Travis CI
   :target: https://travis-ci.org/python/cpython

.. image:: https://ci.appveyor.com/api/projects/status/4mew1a93xdkbf5ua/branch/master?svg=true
   :alt: CPython build status on Appveyor
   :target: https://ci.appveyor.com/project/python/cpython/branch/master

.. image:: https://dev.azure.com/python/cpython/_apis/build/status/Azure%20Pipelines%20CI?branchName=master
   :alt: CPython build status on Azure DevOps
   :target: https://dev.azure.com/python/cpython/_build/latest?definitionId=4&branchName=master

.. image:: https://codecov.io/gh/python/cpython/branch/master/graph/badge.svg
   :alt: CPython code coverage on Codecov
   :target: https://codecov.io/gh/python/cpython

.. image:: https://img.shields.io/badge/zulip-join_chat-brightgreen.svg
   :alt: Python Zulip chat
   :target: https://python.zulipchat.com


Copyright (c) 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011,
2012, 2013, 2014, 2015, 2016, 2017, 2018 Python Software Foundation.  All rights
reserved.

See the end of this file for further copyright and license information.

.. contents::

General Information
-------------------

- Website: https://www.python.org
- Source code: https://github.com/python/cpython
- Issue tracker: https://bugs.python.org
- Documentation: https://docs.python.org
- Developer's Guide: https://devguide.python.org/

Contributing to CPython
-----------------------

For more complete instructions on contributing to CPython development,
see the `Developer Guide`_.

.. _Developer Guide: https://devguide.python.org/

Using Python
------------

Installable Python kits, and information about using Python, are available at
`python.org`_.
```

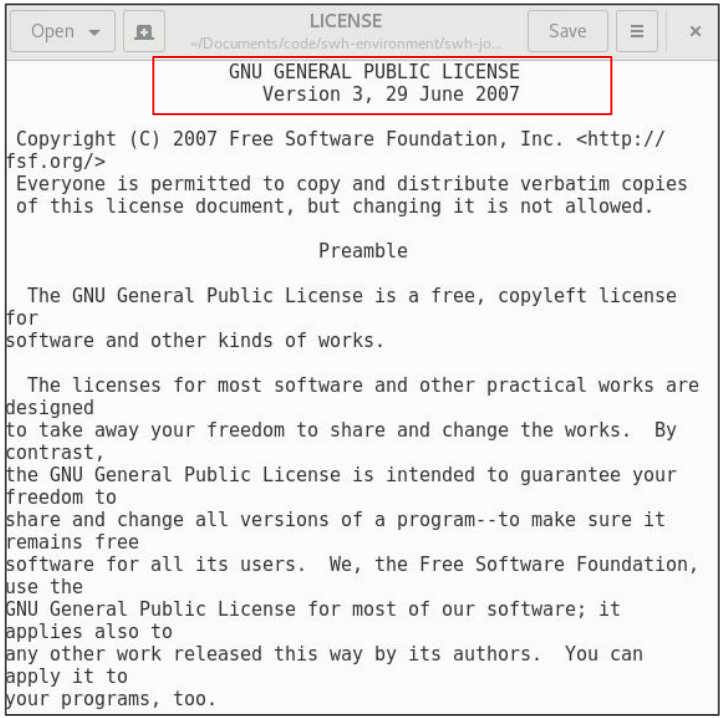
Readme from the CPython project archived on SWH : swh:1:cnt:970f62e6938908a9caaaa0a07fe425bd3976c101;origin=https://github.com/python/cpython/

1.1.2 AUTHORS files

- ★ Identify the people involved in the creation of the software who should be credited for the creative work
  - ○ Others contributors can be identified in a separate file or list, called “contributors”
- ★ Note all the authors in the file and on the form
- ★ The people that are noted on the HAL form and on the AUTHORS file, have authors right on the source code
- ★ You can use the following file names for this file: AUTHORS, AUTHORS.md, AUTHOR.rst, CONTRIBUTORS, CREDITS, CITATION, CITATION.cff, etc.

1.1.3 Fichier LICENSE

- ★ Before depositing the source code, you must choose a license with the appropriate rights owner
  - at INRIA, the CPPI is the best service to discuss the license question
  - Consult the appropriate service in your institution
- ★ If you have more than one license create a `LICENSES/` directory with all licenses
- ★ Note the same license/s in the HAL form's metadata
  - The compatibility between the license in the form and in the code is verified by the moderators
  - The contributor is responsible of the compatibility between licenses (between code and dependencies)



Resources to help review different licenses & practices:

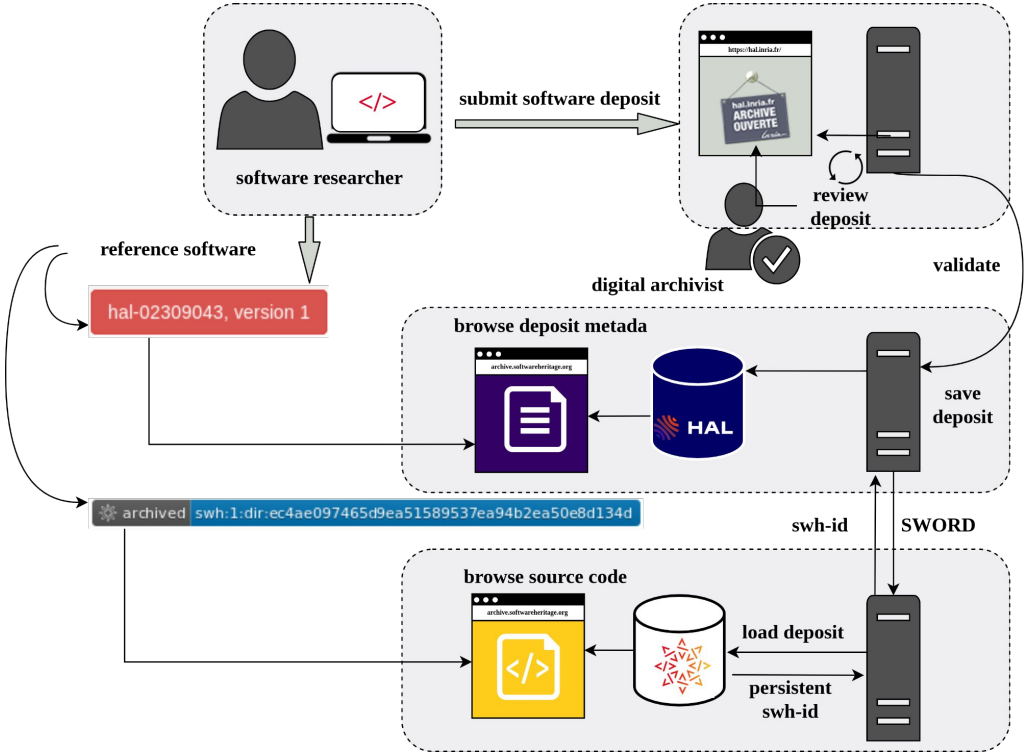
- https://choosealicense.com/
- https://reuse.software/

Chapter 2:
Deposit local source code

local method: deposit .zip /.tar.gz

The local method deposit

- ➔ One compressed **archive** containing source code (mostly text files)
- ➔ A collection of **metadata**
  - ◆ Generic metadata
  - ◆ Software specific metadata



2.1: Deposit the files

After logging in into HAL, use the “submit” tab , Drag and drop or click to upload compressed file.

Note that it is not possible for the software deposit to submit more than one compressed file.

The archive (.zip, .tar.gz) MUST contain all files and shouldn’t be decompressed before validation.

The archive (.zip, .tar.gz) shouldn’t contain another archive (.zip, .tar.gz).

2.2: Complete the metadata

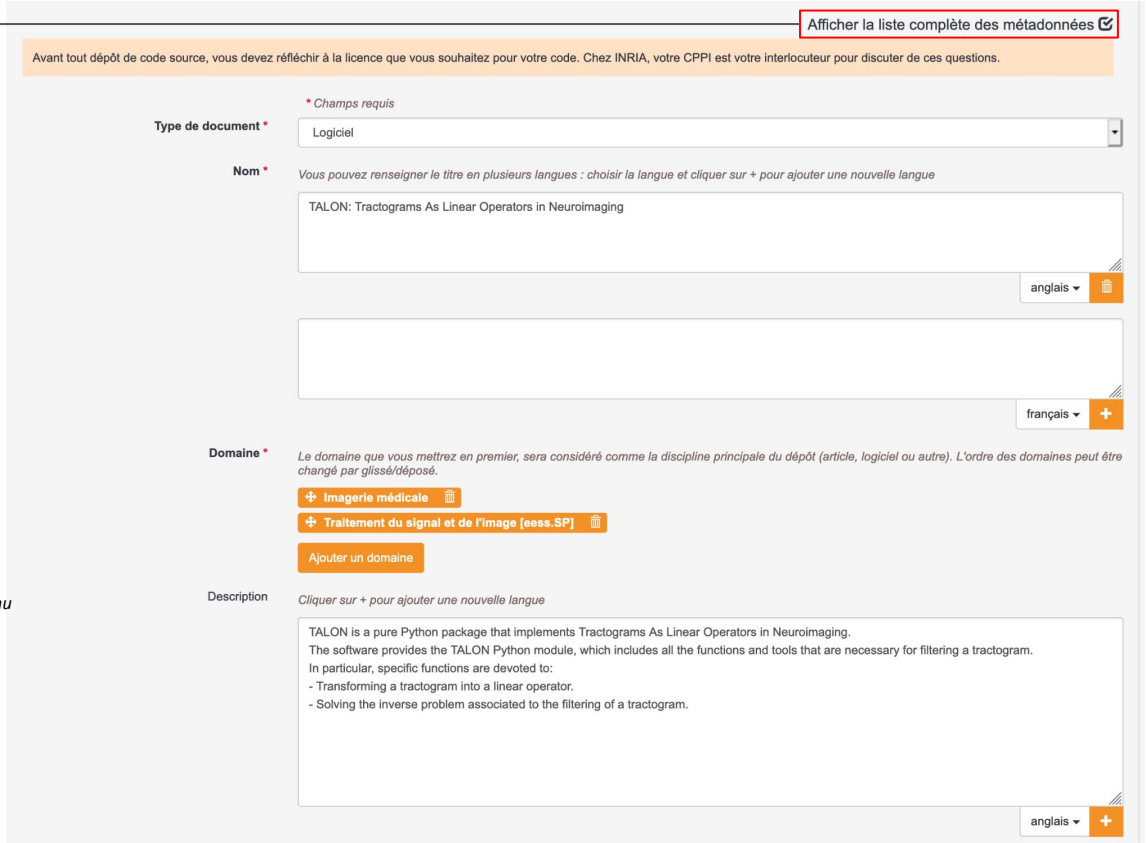
➔ 2.2.1 Choose the document type* :
Software

To see the full list of properties check the checkbox on the top right

Mandatory properties are identified with a star (*)

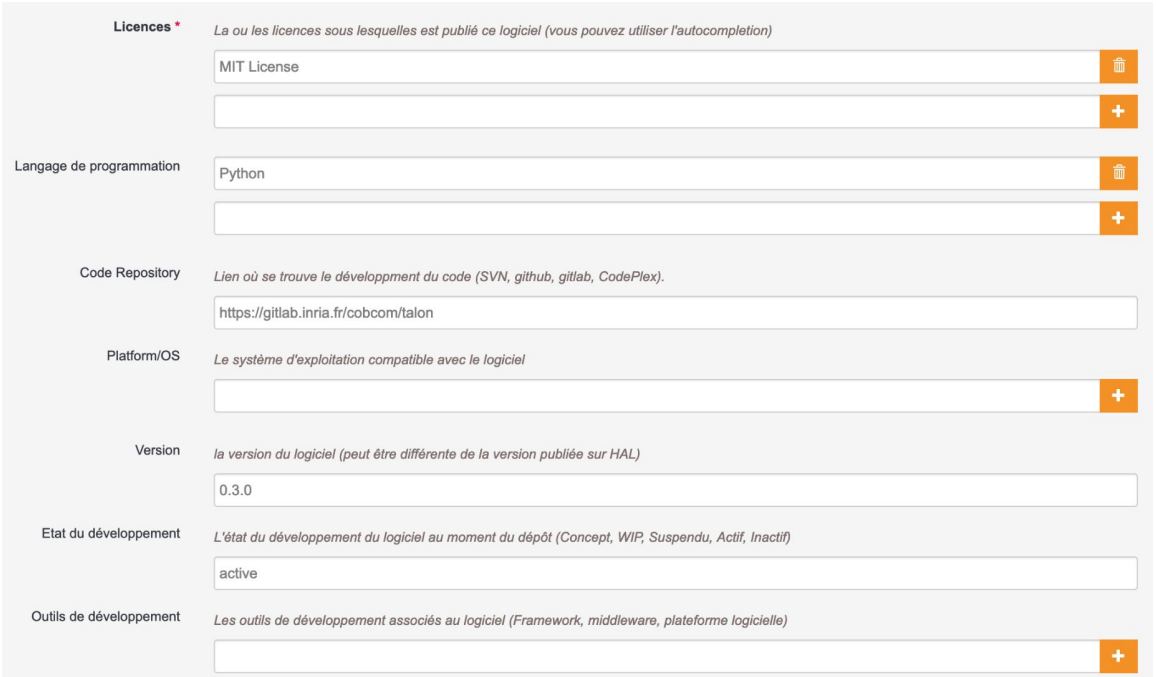
➔ 2.2.2 Add the **general properties** :

- Title*
- Domain*
- Description *(Il est recommandé d'avoir une description cohérente avec le contenu du fichier README)*
- Keywords
- Identifiers
- Related data
- Associated publications
- Production date
- Publisher
- Institution



➔ 2.2.3 Add **software specific metadata** :

- Licenses* (based on the SPDX reference list, it is also possible to enter a license that is not in this list)
- Programming language
- Code repository
- Platform/OS - environment
- Version
- Development status
- Runtime Platform



2.3 Complete information about author(s)

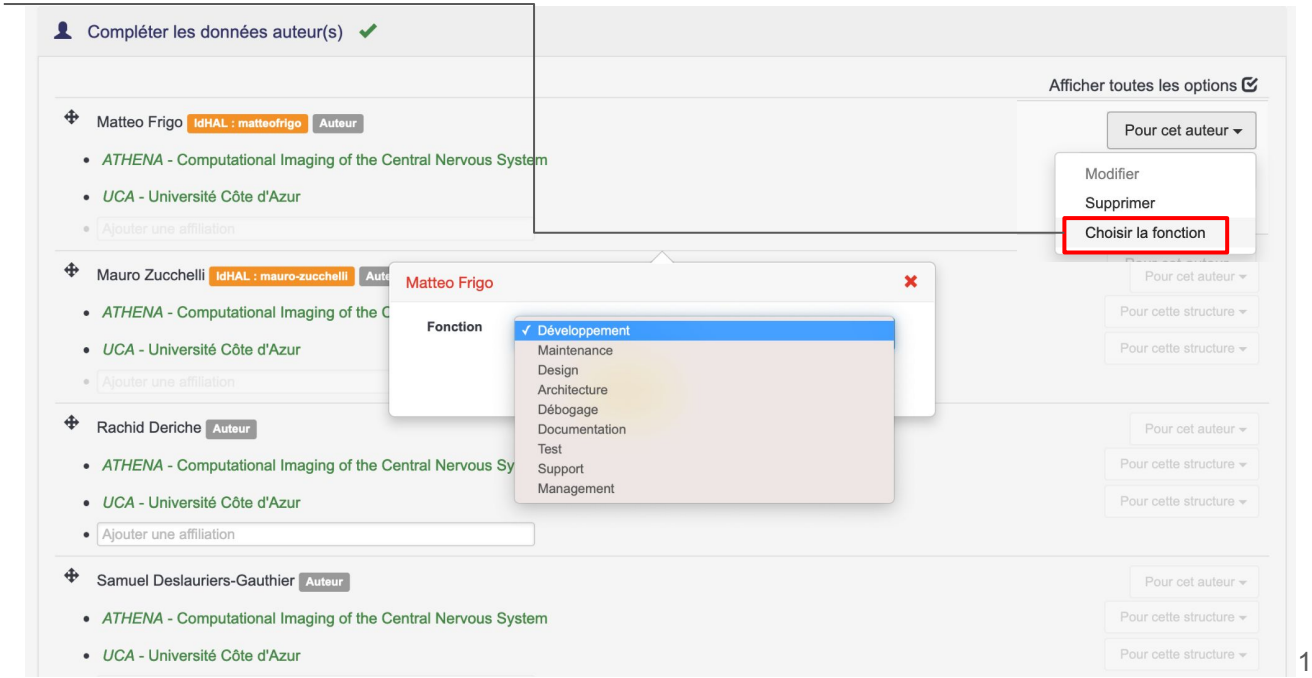
➔ Add the author(s)

➔ Add affiliation for each author *(at least one author must be affiliated)*

- It is possible to add different authors and add a role (development, maintenance, design, architecture, debugging, documentation, test, support, management).

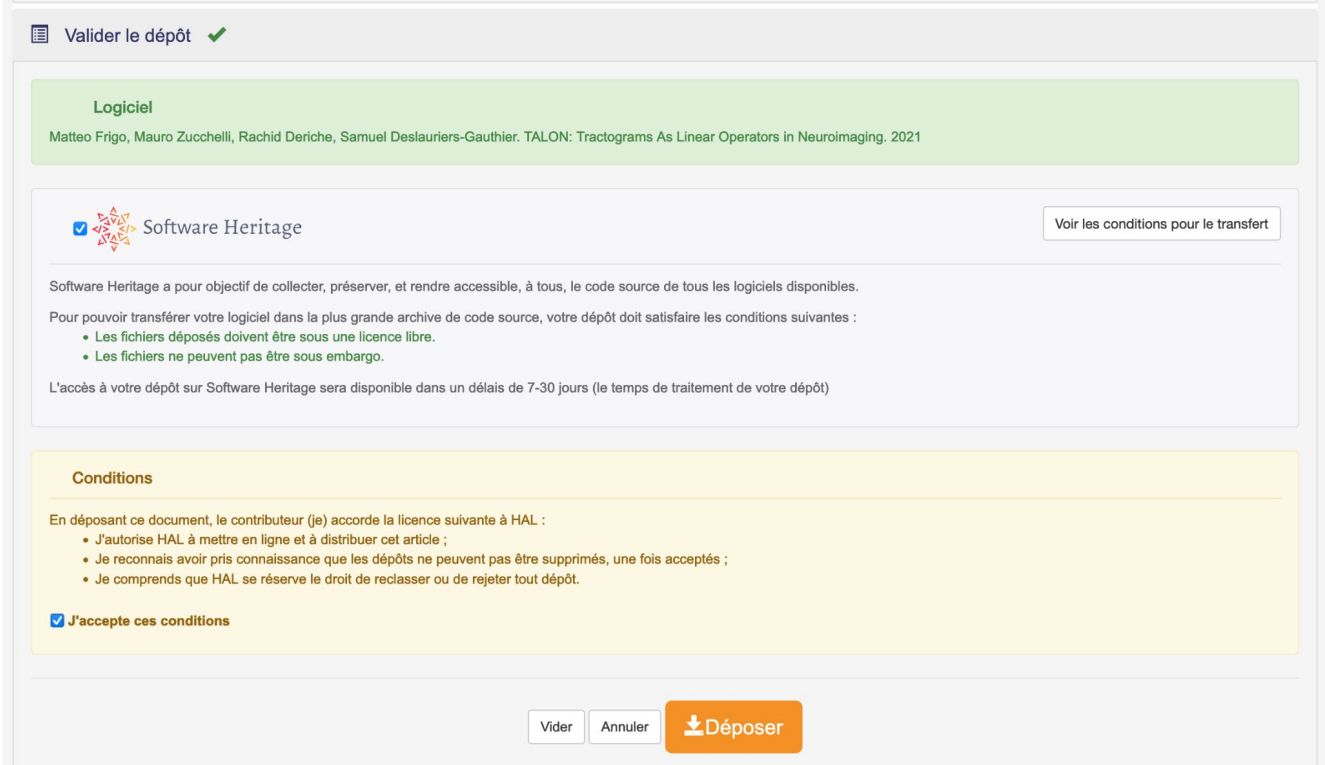
- It is necessary to add all authors from the AUTHORS file.

- You may add a CONTRIBUTORS file for people who participated in the creation of the software but are not considered authors.



2.4 Submit the deposit

- ➔ Accept transfer to Software Heritage and contribute to the largest software source code archive in the world.
- ➔ Accept HAL's conditions
- ➔ Click on "Upload" to submit



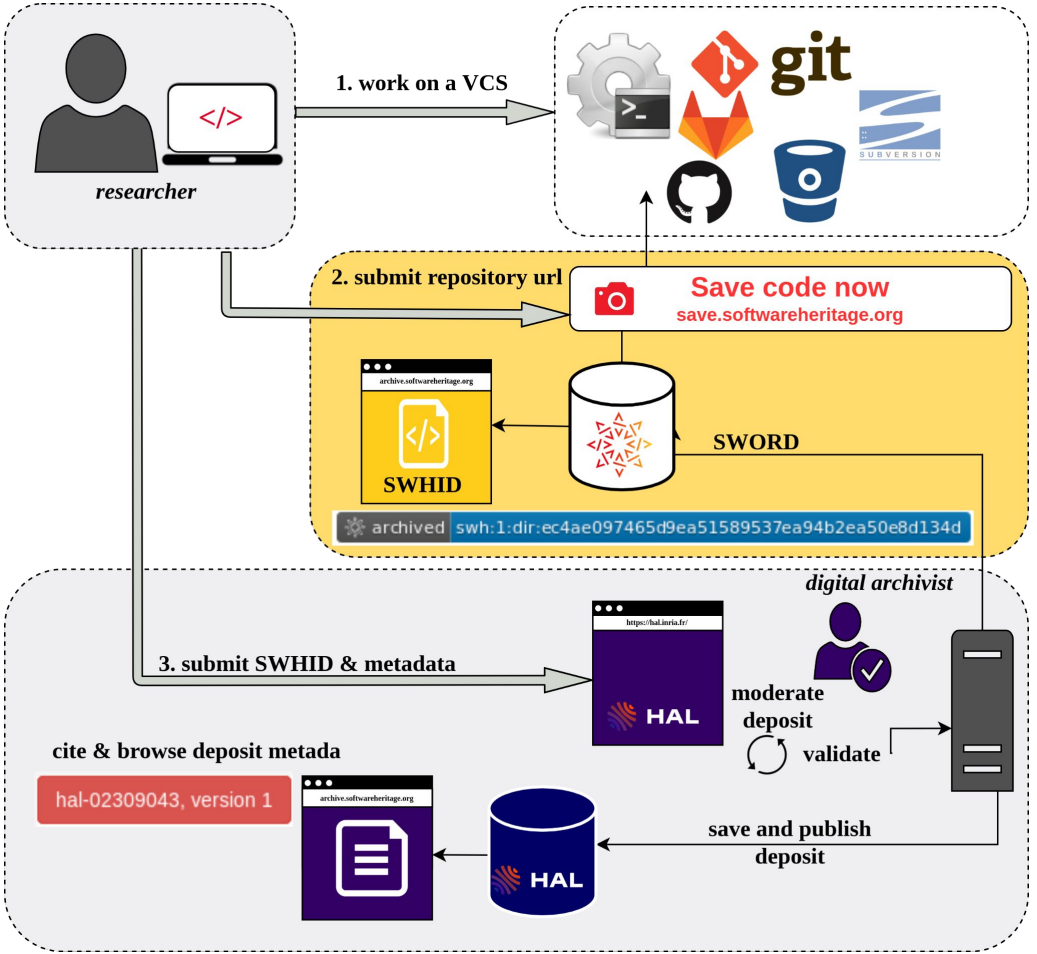
Chapter 3:
Deposit SoftWare Heritage ID (SWHID) with metadata

SWHID method: deposit SWHID with metadata

SWHID method: deposit SWHID with metadata

The SWHID deposit

- The SWHID deposit is composed by:
- A SoftWare Heritage identifier (**SWHID**) that can be found on the Software Heritage archive or calculated locally. The SWHID references the source code.
- A metadata collection
  - the metadata can be **pulled** into the form using the **SWHID**. The properties available on **codemeta.json** file in the root directory of your code can be extracted automatically on HAL's form.
  - **filling** the form with metadata properties to complete the form



*VCS = Version Control System

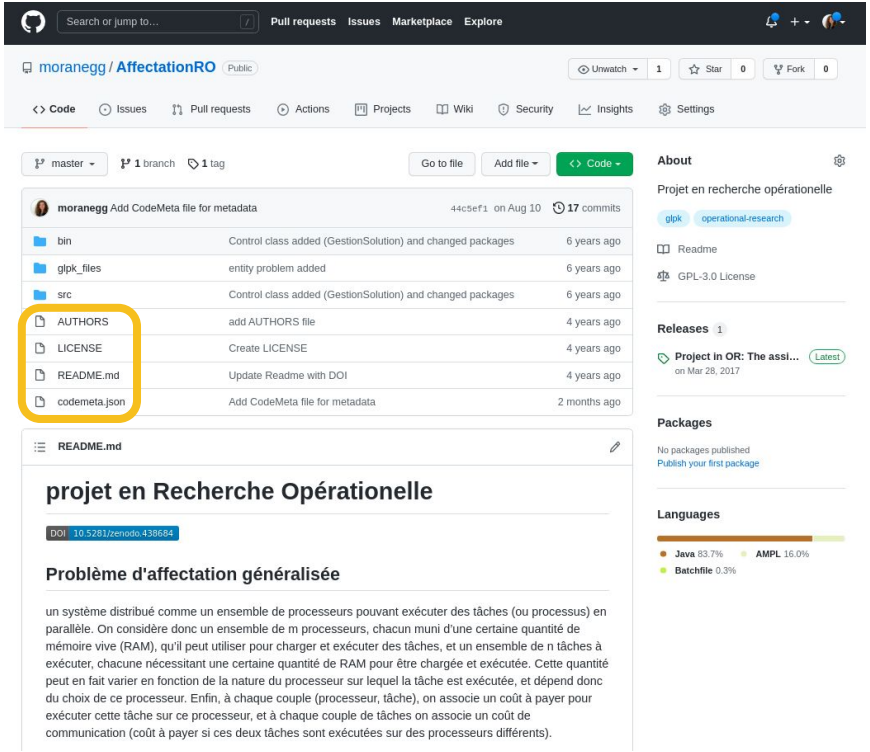
3.1 Prepare the source code

Before depositing on HAL you must prepare your source code on the collaborative development platform (GitHub, Gitlab, etc.).

The following elements are verified by a moderator

3.1.1 Add the following files to your source code and push to your `code repository`:

- ❏ README (see section 1.1.1 README file)
- ❏ AUTHORS (see section 1.1.2 AUTHORS files)
- ❏ LICENSE (see section 1.1.3 Fichier LICENSE)

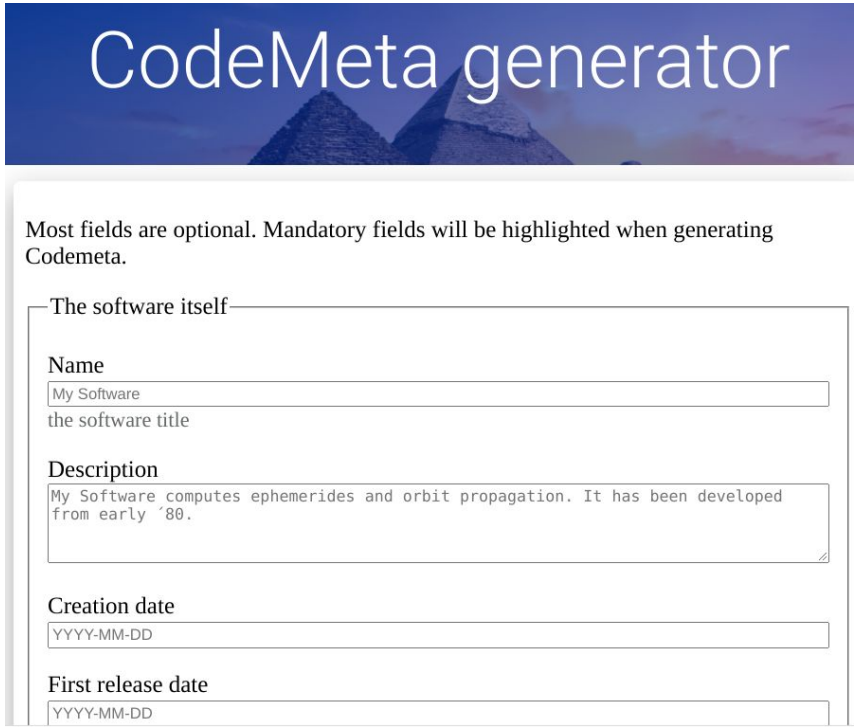


- ❏ codemeta.json - isn't mandatory but useful to complete the form especially when using the SWHID method.

3.1.2 Why CodeMeta ?

- A vocabulary extending schema.org
  - https://codemeta.github.io/terms/
- An academic community
- A crosswalk table enabling translations between different ontologies/vocabularies to CodeMeta

Tool to create a codemeta.json file



To create easily a codemeta.json file use the online tool

➢ You can contribute here

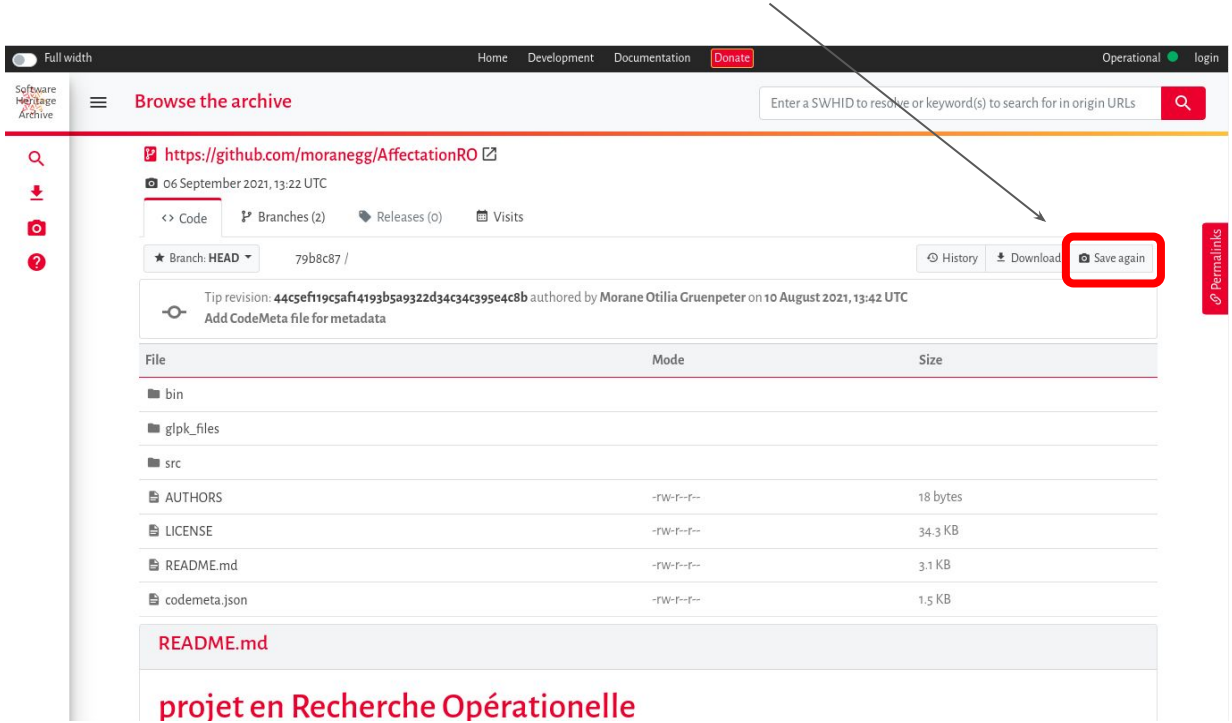
3.1.3 CodeMeta file: an example


```
{
    "@context": "https://doi.org/10.5063/schema/codemeta-2.0",
    "@type": "SoftwareSourceCode",
    "license": "https://spdx.org/licenses/AGPL-3.0",
    "codeRepository": "https://github.com/moranegg/AffectationRO",
    "dateCreated": "2014-01-01",
    "dateModified": "2019-06-26",
    "name": "AffectationRO- The assignment problem",
    "version": "1.0.0",
    "description": "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).\\r\\n 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.",
    "applicationCategory": "info",
    "releaseNotes": "First release with GLPK, in Beta testing",
    "developmentStatus": "concept",
    "keywords": [
        "distributed systems",
        "glpk",
        "optimisation",
        "OR"
    ],
    "programmingLanguage": [
        "Java"
    ],
    "author": [
        {
            "@type": "Person",
            "@id": "https://orcid.org/0000-0002-9777-5560",
            "givenName": "Morane",
            "familyName": "Gruenpeter",
            "email": "morane.gg@gmail.com",
            "affiliation": {
                "@type": "Organization",
                "name": "Software Heritage"
            }
        }
    ]
}
```

3.2 Save your code on Software Heritage

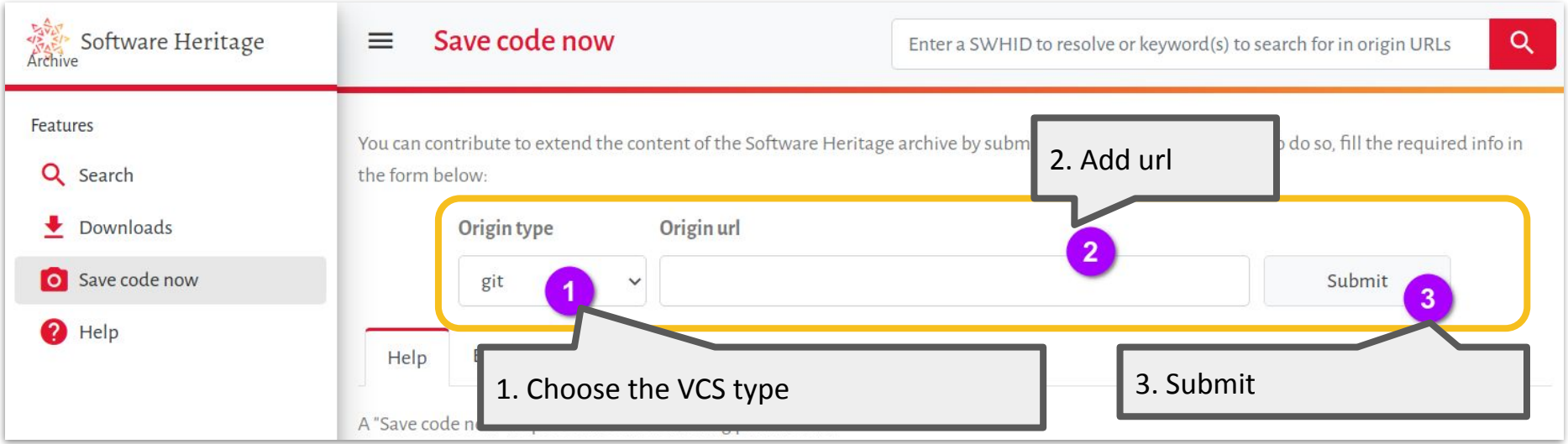
https://archive.softwareheritage.org/

3.2.1 Verify if your **code repository** and the **specific version** you want to submit are already in Software Heritage

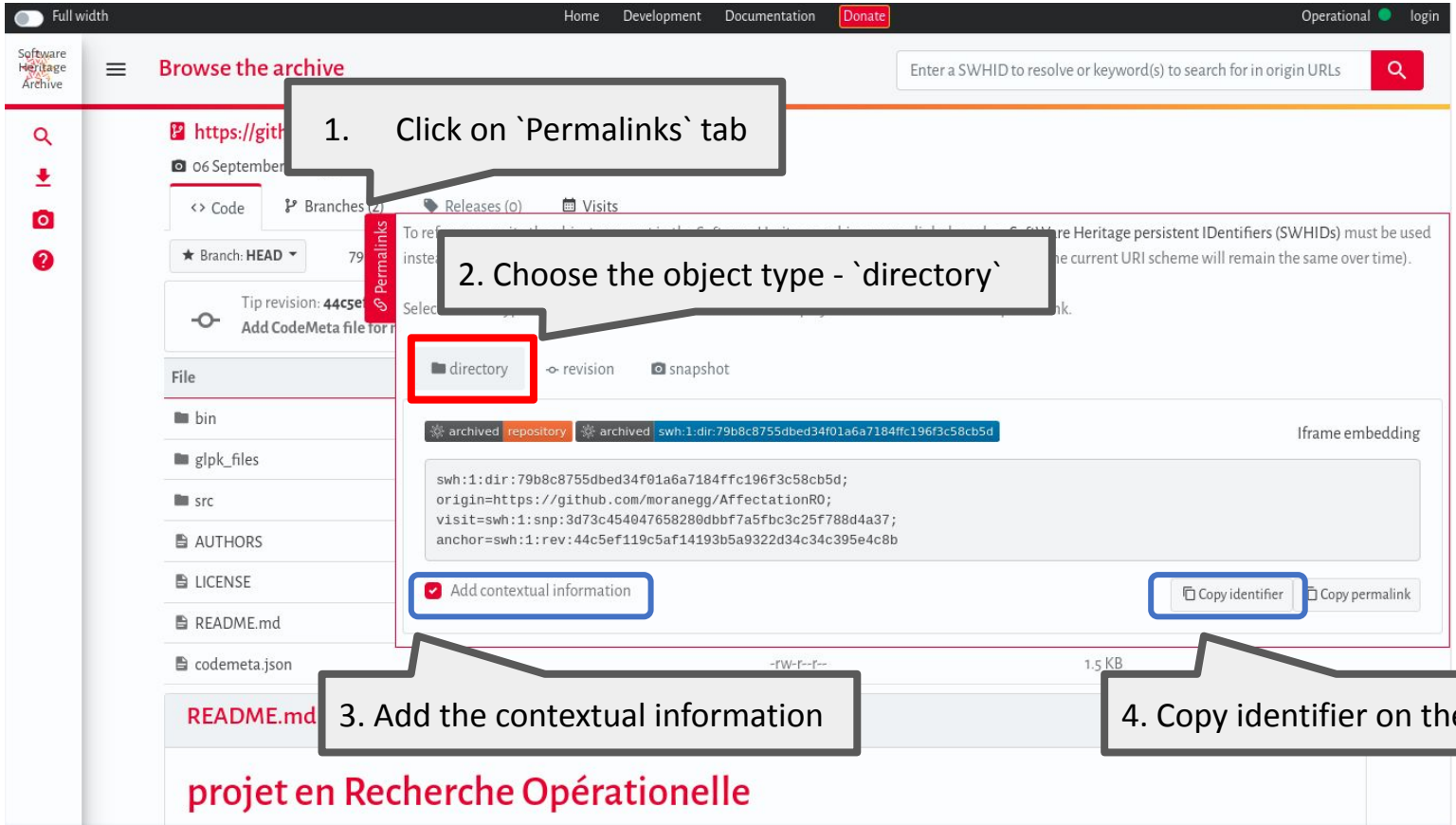


3.2.2 If your code isn't in Software Heritage- **Submit** the code repository url on the **Save Code Now:**

https://save.softwareheritage.org/



3.3 Choose a SWHID on the Software Heritage archive



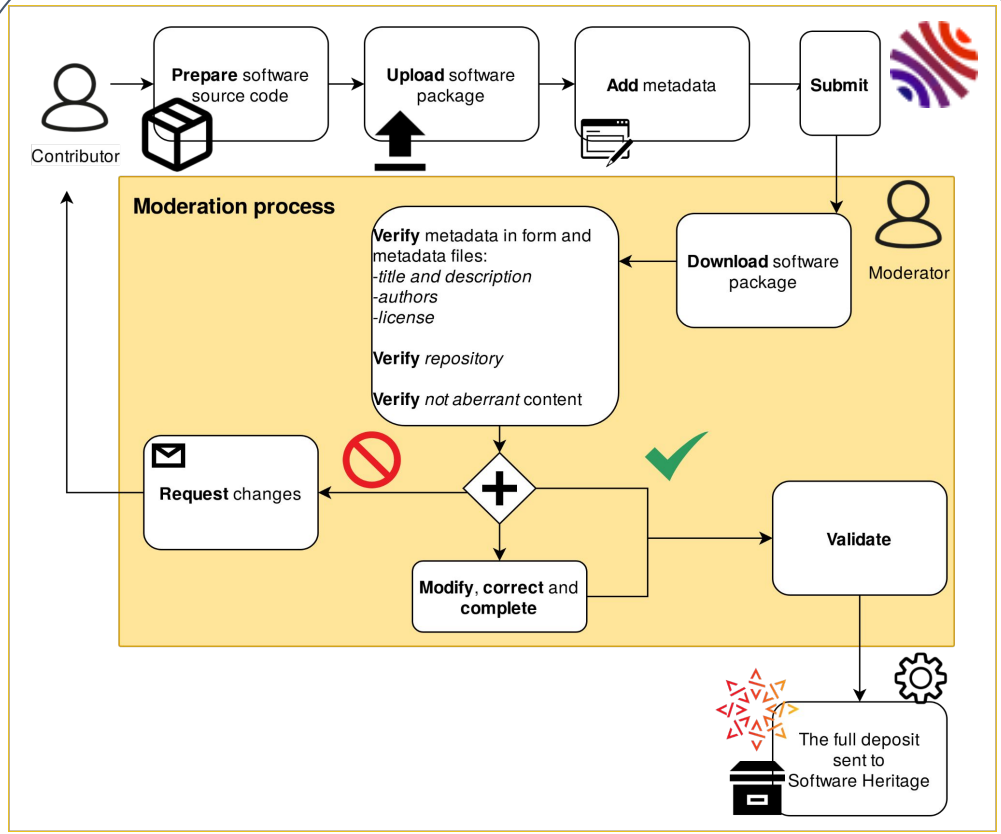
3.4 Deposit SWHID - the reference to the content

- ➔ Put the SWHID on the HAL form
  - ◆ Prefer a SWHID with contextual information (to keep the link with the contextual information)
  - ◆ If a codemeta.json is present in the root directory of the deposited SWHID - the HAL platform will pull the metadata automatically
- ➔ Verify and complete metadata
  - ◆ Verify inserted metadata
  - ◆ Complete missing metadata
  - ◆ Choose domain
  - ◆ Verify authors and add affiliations
- ➔ Validate deposit

Chapitre 4:
Behind the scenes- the moderation process

The moderation process

- ➔ Deposit in moderation
- ➔ Dialogue between contributor and moderator
- ➔ Deposit validation in moderation



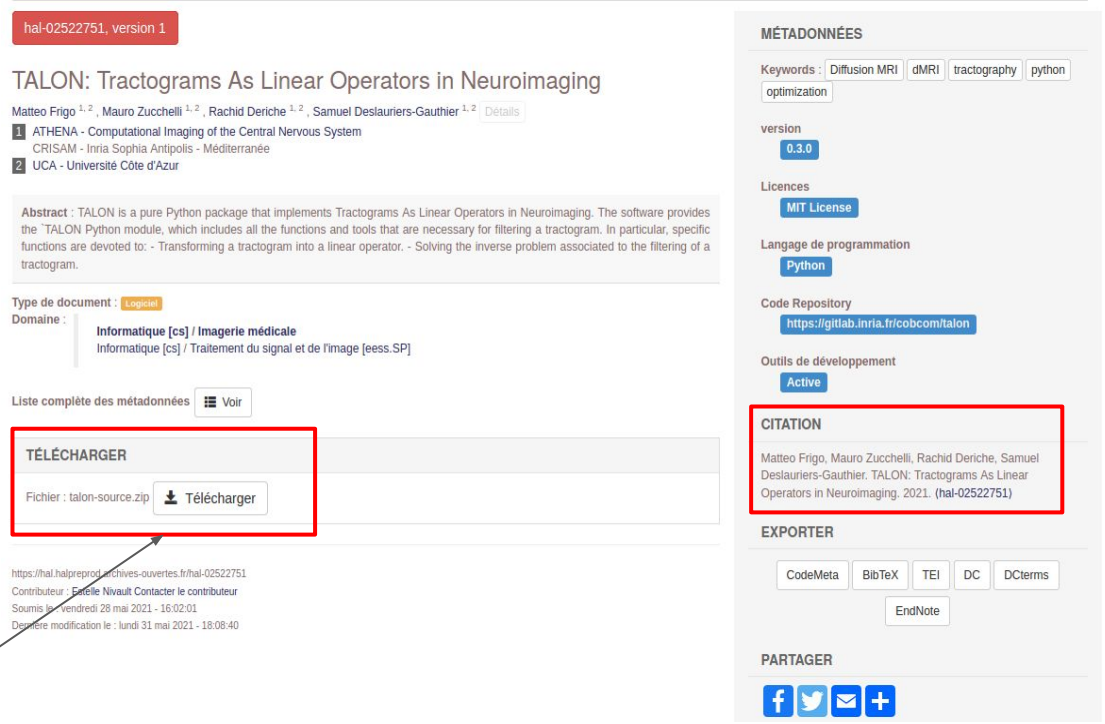
Moderation guide :
Morane Gruenpeter, Jozefina Sadowska. La modération d'un dépôt logiciel : Bonnes pratiques et guide pour le modérateur. [Rapport Technique] Inria; CCSD; Software Heritage. 2018. ⟨hal-01876705⟩

Chapitre 5:
The deposit publication and exports

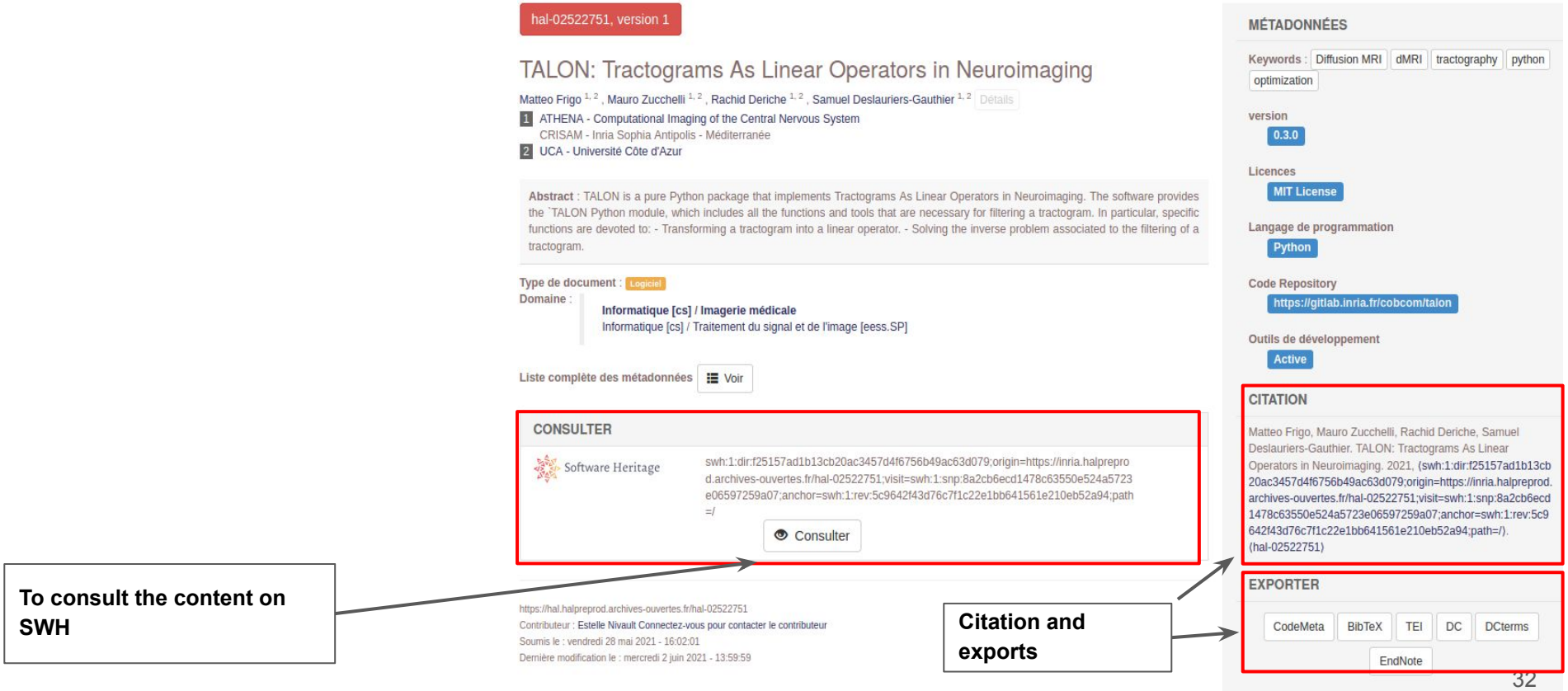
5.1 The deposit is transferred to SWH

- ➔ The local method deposit is transferred to SWH with:
  - ◆ Content
  - ◆ Metadata
- ➔ The SWHID method deposit is transferred to SWH with:
  - ◆ Metadata

When the local method deposit is transferred to SWH it will be visible on HAL without the link to SWH. When it is ingested by SWH, the link will appear on the deposit



5.2 The final deposit



5.3 The identifiers of the software deposit

5.3.1 Reference with the SWHID

(SoftWare Heritage Identifiers)

- ★ Identification of the software source code artifact
- ★ A digital fingerprint specific source code content

Needed to :

- **Identify - reproduce**
- **Archive**

5.3.2 Cite with the HAL-ID

- ★ Identification of the software record
- ★ Metadata of the deposit
- ★ Authors and contributors are verified in the moderation process

Needed to :

- **Give credit to the authors**
- **Index**

5.4 The citation and the BibTeX export

- ➔ The citation is accessible on the HAL record
- ➔ Export BibTeX using the format BibLaTeX **@software** or **@softwareversion** (if a version property was submitted)
- ➔ Export used in activity reports for scientific outputs at Inria since 2020.

HAL's citation format

Matteo Frigo, Mauro Zucchelli, Rachid Deriche, Samuel Deslauriers-Gauthier. TALON: Tractograms As Linear Operators in Neuroimaging. 2021, ⟨swh:1:dir:f25157ad1b13cb20ac3457d4f6756b49ac63d079;origin=https://hal.archives-ouvertes.fr/hal-03116143;visit=swh:1:snp:465d8995619657871 7f4cb5155e456c279aa6a22;anchor=swh:1:rev:10247a14640a280b9140a27ce003d382d70cccac;path=/⟩. ⟨hal-03116143⟩

Softwares

[38] [SW] M. Frigo, M. Zucchelli, R. Deriche and S. Deslauriers-Gauthier, *TALON: Tractograms As Linear Operators in Neuroimaging* version 0.3.0, 19th Jan. 2021. HAL: ⟨hal-03116143⟩, URL: https://hal.archives-ouvertes.fr/hal-03116143, VCS: https://gitlab.inria.fr/cobcom/talon, SWHID: ⟨swh:1:dir:f25157ad1b13cb20ac3457d4f6756b49ac63d079;origin=https://hal.archives-ouvertes.fr/hal-03116143;visit=swh:1:snp:465d89956196578717f4cb5155e456c279aa6a22;anchor=swh:1:rev:10247a14640a280b9140a27ce003d382d70cccac;path=/⟩.

```
@softwareversion{frigo:hal-03116143v1,
  TITLE = {{TALON: Tractograms As Linear Operators in
Neuroimaging}},
  AUTHOR = {Frigo, Matteo and Zucchelli, Mauro and
Deriche, Rachid and Deslauriers-Gauthier, Samuel},
  URL = {https://hal.archives-ouvertes.fr/hal-03116143},
  NOTE = {},
  YEAR = {2021},
  MONTH = Jan,
  SWHID =
{swh:1:dir:f25157ad1b13cb20ac3457d4f6756b49ac63d079;origin
=https://hal.archives-ouvertes.fr/hal-03116143;visit=swh:1
:snp:465d89956196578717f4cb5155e456c279aa6a22;anchor=swh:1
:rev:10247a14640a280b9140a27ce003d382d70cccac;path=/},
  VERSION = {0.3.0},
  REPOSITORY = {https://gitlab.inria.fr/cobcom/talon},
  LICENSE = {MIT License},
  KEYWORDS = {diffusion MRI ; dMRI ; tractography ; python
; optimization},
  FILE =
{https://hal.archives-ouvertes.fr/hal-03116143/file/talon-
source.zip},
  HAL_ID = {hal-03116143},
  HAL_VERSION = {v1},
}
```

References

- Y. Barborini, R. Di Cosmo, Antoine R. Dumont, M. Gruenpeter, B. Marmol, A. Monteil, J. Sadowska.. La création du nouveau type de dépôt scientifique - Le logiciel. *JSO 2018 - 7es journées Science Ouverte Couperin : 100 % open access : initiatives pour une transition réussie*, Jan 2018, Paris, France. 2018. ⟨hal-01688726⟩

- R. Di Cosmo, M. Gruenpeter, B. Marmol, A. Monteil, L. Romary, J. Sadowska. *Curated Archiving of Research Software Artifacts: lessons learned from the French open archive*. IJDC. 2020 (10.2218/ijdc.v15i1.698). (hal-02475835)

- R. Di Cosmo, M. Gruenpeter, S. Zacchiroli *Referencing Source Code Artifacts: a Separate Concern in Software Citation*, CiSE, IEEE, pp.1-9. 2020. (10.1109/MCSE.2019.2963148) (hal-02446202)

- P. Alliez, R. Di Cosmo, B. Guedj, A. Girault, M.-S. Hacid, et al.. Attributing and Referencing (Research) Software: Best Practices and Outlook from Inria. Computing in Science and Engineering, Institute of Electrical and Electronics Engineers, 2019, pp.1-14. ⟨10.1109/MCSE.2019.2949413⟩. (hal-02135891)

- A. Monteil, M. Gruenpeter, J. Sadowska, E. Nivault. Garantir la cohérence des données constitue le cœur de notre activité: entretien autour des enjeux descriptifs du code source. *Bulletin des bibliothèques de France*, Ecole Nationale Supérieure des Sciences de l'Information et des Bibliothèques (ENSSIB), 2021, Dossier BBF 2021-1 • Code source : libérer le patrimoine !. ⟨hal-03239502⟩