« GARANTIR LA COHÉRENCE DES DONNÉES CONSTITUE LE CŒUR DE NOTRE ACTIVITÉ »

Entretien autour des enjeux descriptifs du code source

Alain Monteil

Service Information et édition scientifique (IES) – Inria

Morane Gruenpeter

Software Heritage – Inria

Jozéfina Sadowska

Service IES – Inria

Estelle Nivault

Service IES – Inria

Imaginez votre réaction en découvrant cette référence bibliographique : « Notre-Dame de Paris, *aux alentours de je ne sais quoi, écrit peut-être par un certain Victor, dont le patronyme me reviendra plus tard ; c'est l'édition avec la couverture bleue.* » Tel peut être le degré d'exactitude auquel les chercheurs sont confrontés lorsqu'il s'agit de désigner des logiciels. Certes, le trait est grossi, mais la question de la description des logiciels de recherche est restée entière pendant de longues années. Essentiels à l'activité scientifique, les logiciels sont les mal-aimés des pratiques de référencement et de citation. Afin d'œuvrer à une meilleure reconnaissance (dans tous les sens du terme) des logiciels conçus par les chercheurs, Roberto Di Cosmo a sollicité, en 2017, le service Information et édition scientifique (IES) de l'Inria pour impulser une collaboration entre Software Heritage et HAL. Une collaboration inter-métiers s'est ainsi nouée entre Morane Gruenpeter (Software Heritage) et Jozéfina Sadowska et Alain Monteil (IES), rejoints par Estelle Nivault (IES).

*

BBF : Quelles sont les pratiques descriptives en vigueur pour rendre un logiciel citable, identifiable ?

A. Monteil, M. Gruenpeter, J. Sadowska et E. Nivault : Le rapport EOSC[1] définit quatre piliers qui garantissent les bonnes pratiques en matière de science ouverte, plus particulièrement pour les logiciels :

- archiver ;
- référencer ;
- décrire ;
- créditer.

Software Heritage assure les deux premiers objectifs, en proposant l'archivage pérenne et surtout, en calculant un identifiant unique et intrinsèque : le *SoftWare Heritage persistent IDentifier* (SWHID). Cet identifiant permet de référencer sans équivoque le logiciel mais aussi les lignes de code elles-mêmes.

La description et le fait de créditer un code source à son concepteur sont des aspects quant à eux assurés par HAL. En d'autres termes, l'archive ouverte facilite la description du logiciel à l'aide de métadonnées appropriées et crédite les créateurs. Les différentes fonctions associées aux contributeurs du logiciel décrites dans l'article du groupe de travail chercheurs (Alliez *et al.*, 2019) ont ainsi été mises en œuvre.

Quelles sont les informations clés pour s'assurer de la réutilisation d'un logiciel sur le long terme ?

La première réflexion doit porter sur ce qu'on entend par « réutilisation » ? Quelle définition en donne-t-on ?

L'objet de notre action est de rendre accessible et disponible le code logiciel. Nous ne testons pas le bon fonctionnement de ces codes. Les modérateurs de HAL ne peuvent préjuger de la motivation des personnes à disposer de ce code. Par exemple, récupérer le code de la mission Apollo 11 est une démarche très intéressante de relecture et de réutilisation à des fins historiques, même sans pouvoir faire de la compilation-exécution du code.

Pour en assurer une réutilisation sur le long terme, l'archivage pérenne du code constitue la base : il faut un identifiant unique et pérenne. Le logiciel doit être accompagné des éléments descriptifs suivants, pour assurer son utilisabilité et ré-utilisabilité :

- la licence, qui décrit les termes d'usages du code ; celle-ci est obligatoire et vérifiée par les modérateurs ;
- le fichier README, avec une documentation minimale pour comprendre le code ; seule l'existence du fichier est vérifiée, sans contrôle qualité des informations qui ont été apportées par les créateurs ;
- l'environnement et les dépendances avec d'autres logiciels, pour pouvoir relancer

1 Scholarly infrastructures for research software. *Report from the EOSC Executive Board Working Group (WG) Architecture Task Force (TF) SIRS* : https://colibris.link/u7BaN

le logiciel sur d'autres machines ; ces éléments ne sont pas vérifiés par les modérateurs HAL.

Existe-t-il une norme descriptive pour le code source, comme c'est le cas d'une monographie par exemple ?

Des normes descriptives existent dans différentes communautés de développement qui ne sont pas nécessairement académiques. Or, les besoins inhérents aux pratiques scientifiques sont très spécifiques et ne sont pas pris en compte de manière satisfaisante pour des chercheurs.

Pour combler ce manque, le travail s'appuie sur CodeMeta[2], projet qui a vu le jour en 2016 et dont l'objectif est de proposer une norme descriptive pour les logiciels de recherche. CodeMeta s'appuie sur les termes[3] du vocabulaire schema.org[4], afin d'adapter et d'élargir la liste des propriétés des logiciels aux spécificités des productions scientifiques. CodeMeta a pour autre avantage de proposer un système de tables de correspondances, qui permet de traduire une ontologie/vocabulaire vers CodeMeta. Ce système permet ainsi de savoir que le code P277 renvoie au type de langage de programmation utilisé par Wikidata, alors que cette même information apparaît sous le libellé « Format » pour DataCite, ou encore « languages_url. » quand on travaille sous GitHub.

La bonne pratique est d'ajouter un fichier de métadonnées appelé codemeta.json à la racine du code source. Voici un exemple d'un lot de données de base que l'on peut ajouter à la racine d'un entrepôt GitHub :

```
    "@context": ["http://schema.org", {"author": {"@id": "schema:author",
"@container": "@list"} } ] ,
    "name": "Generate CodeMeta Metadata for R Packages",
    "description": "Codemeta defines a 'JSON-LD' format for describing
software metadata. This package provides utilities to generate, parse, and
modify codemeta.jsonld files automatically for R packages."
```

Mais cette pratique est très peu utilisée[5]. Le format CodeMeta est lisible par les humains comme en témoignent ces exemples : downloadUrl, operatingSystem, softwareRequirements. Mais ce format est aussi interprétable par les machines, pour automatiser des processus d'indexation.

Par ailleurs, les pratiques en matière de citation du logiciel sont très variables et lacunaires. Elles apparaissent souvent en note de bas de page avec un lien vers une

2 https://codemeta.github.io/

3 https://codemeta.github.io/terms/

4 *« Schema.org est un schéma de micro-données utilisé sur le Web. Les micro-données permettent aux robots d'indexation de saisir plus précisément le sens des pages indexées. »* Pour en savoir plus : https://fr.wikipedia.org/wiki/Schema.org

5 Les raisons de cette sous-utilisation restent à étudier, mais on peut avancer d'ores et déjà le manque de temps. Le dépôt avec SWHID est précisément conçu pour aider les développeurs à traiter ce point plus facilement.

forge comme GitHub et au mieux comme référence BibTeX de type *@misc* dans la liste des références.

Roberto Di Cosmo[6] a traduit en BibLaTeX[7] le fruit de nos travaux pour proposer une solution de description du logiciel au format BibTeX avec les types *@software*, *@softwaremodule*, *@softwareversion* et *@codefragment*. Cette solution est à présent intégrée dans l'environnement LaTeX de TexLive et disponible sur le site CTAN. Désormais sur HAL, l'export BibTeX est disponible avec le type *@softwareversion*. Nous espérons que ce modèle BibTeX deviendra un standard largement adopté par les chercheurs.

Certaines revues[8] de la plateforme Episciences l'ont déjà intégré dans leurs instructions aux auteurs. Cette nouvelle recommandation a également été mise en pratique dès l'année 2020 lors de la production du rapport d'activité scientifique de l'Inria[9].

En parallèle, nous avons démarré un chantier de transposition des éléments CodeMeta dans l'univers TEI. Cela participe aussi à une forme de standardisation de la description des logiciels.

De plus, des réflexions sont en cours pour formaliser la citation des logiciels dans le texte intégral pour le standard JATS[10], largement utilisé chez les éditeurs, en particulier dans le domaine des sciences du vivant.

Quelles sont les compétences clés selon vous pour assurer un référencement de qualité ?

Levons immédiatement tous freins : il n'est pas nécessaire d'être développeur ou d'avoir des connaissances informatiques pour accompagner le référencement du logiciel. Tout au plus, connaître un peu les pratiques de fonctionnement des développeurs du monde de l'open source et des forges pour repérer plus facilement la présence des fichiers nécessaires au référencement du code.

Nous sommes dans l'environnement du libre accès dans lequel c'est l'auteur/développeur qui initie le signalement du logiciel. Autrement dit, l'auteur du logiciel est donc responsable de sa conception, mais il joue aussi un rôle majeur dans la description du code qu'il dépose. Le dépôt comprend aussi des éléments tels que les fichiers README, AUTHORS, LICENSE, codemeta.json, etc. Pour guider les chercheurs, nous avons élaboré un guide de dépôt de logiciel, selon les bonnes pratiques qui proviennent du monde du logiciel libre[11].

6 Roberto DI COSMO. « Announcing biblatex-software : software citation made easy », *ACM SIGSOFT Software Engineering Notes*. 12 octobre 2020, vol. 45, n° 4. p. 22-23. En ligne : https://doi.org/10.1145/3417564.3417570 (Green open access : https://hal.inria.fr/hal-02977711v1)

7 https://ctan.org/pkg/biblatex-software

8 *Revue Africaine de la Recherche en Informatique et Mathématiques Appliquées* : https://arima.episciences.org/ ; *Journal of Theoretical, Computational and Applied Mechanics* : https://jtcam.episciences.org/

9 https://www.inria.fr/fr

10 https://en.wikipedia.org/wiki/Journal_Article_Tag_Suite

11 https://colibris.link/Ex3wJ

La qualité du code et celle du logiciel ne sont pas évaluées par les professionnels de l'information lors du dépôt dans HAL. Ce contrôle ne fait pas partie du champ de compétences des documentalistes. En revanche, les informations descriptives telles que les métadonnées, la présence des fichiers README, AUTHORS, LICENCE, sont systématiquement vérifiées.

Garantir la cohérence des données renseignées constitue le cœur de notre activité. Nous avons élaboré un guide de modération pour faciliter ce travail de validation aux modérateurs qui vérifient l'exactitude et la cohérence des métadonnées, avec le code source déposé. Grâce à ce guide, les professionnels de l'information qui modèrent le code source n'ont pas besoin de connaissances informatiques approfondies. Toutes les informations nécessaires à la vérification qualitative de la description de logiciel sont rassemblées dans la documentation utilisateurs.

Nous travaillons bien dans un environnement d'auto-archivage modéré, propre à HAL, que l'on ne retrouve pas forcément dans d'autres archives comme Zenodo par exemple.

Les pratiques de référencement sont-elles différentes selon qu'on cherche à décrire quelques lignes de code ou un logiciel ?

Sur le plan de la théorie, la réponse est oui, les pratiques de référencement sont différentes, car l'identification des différents niveaux ne se fait pas de la même manière ou avec les mêmes identifiants. C'est aussi la raison pour laquelle nous utilisons les deux identifiants, l'identifiant HAL et l'identifiant SWHID, sur une citation d'un dépôt logiciel, car l'un identifie la notice du projet et l'autre identifiant est calculé à partir de l'objet, il l'identifie comme une empreinte digitale.

Sur le plan de la pratique, pour le moment, nous proposons sur HAL le dépôt d'un dossier code-source qui peut contenir plusieurs fichiers. Le formulaire de dépôt avec les métadonnées descriptives des objets logiciels est unique. Nous aimerions à terme proposer une typologie plus fine sur HAL pour permettre d'encore mieux citer le code logiciel.

Le type Software Module, par exemple, renseigne sur un élément particulier d'un logiciel, voire d'un ensemble comme peut l'être CGAL[12] ou MATLAB. La question de la citation d'un fragment de code lui-même sera aussi vu comme un « sous-type » de logiciel : le fragment. En ce qui concerne la citation, Roberto Di Cosmo a implémenté les types suivant en BibLaTeX :

- *@software* pour le projet logiciel ;
- *@softwaremodule* ;
- *@softwareversion* ;
- *@codefragment*.

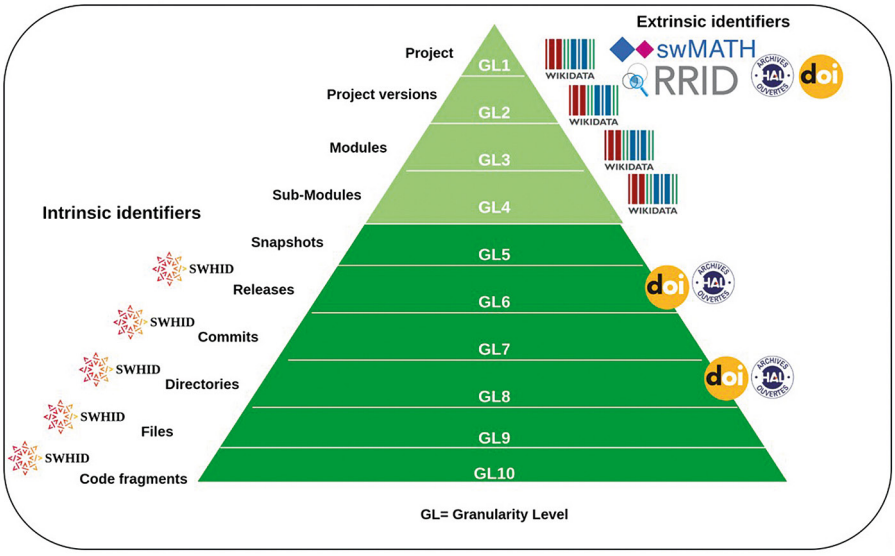
Chaque élément hérite des informations de l'élément supérieur, ce qui offre la possibilité de référencer les lignes de codes en les contextualisant. En revanche, les quatre

12 https://www.cgal.org/

types ci-dessus ne représentent pas la totalité des niveaux de granularité que vous pouvez trouver dans un projet logiciel complexe.

Sur la figure 1, sont représentés 10 niveaux différents de granularité, du projet, noté GL1, à l'algorithme (code fragment), noté GL10. Entre ces extrémités, il y a des niveaux qui peuvent être identifiés avec un identifiant extrinsèque et d'autres niveaux qui peuvent être identifiés uniquement avec un identifiant intrinsèque, comme le SWHID – l'empreinte digitale du code source.

Figure 1 : Niveaux de granularité d'un projet logiciel
(cette figure est une extension de la figure proposée sur : Research Data Alliance/ FORCE11 Software Source Code Identification WG, Alice ALLEN, Anita BANDROWSKI, Peter CHAN, Roberto DI COSMO, *et al.* « Software Source Code Identification Use cases and identifier schemes for persistent software source code identification (Version 1.1) ». 6 octobre 2020. En ligne : http://doi.org/10.15497/RDA00053)



Quelles ont été les grandes étapes pour mettre en place l'interfaçage entre Software Heritage et HAL ?

Pour faciliter le dialogue, le service Information et édition scientifique de l'Inria a joué le rôle d'intermédiaire entre Software Heritage et le CCSD lors de leur première rencontre. D'un point de vue méthodologique, nous avons pris le parti de commencer uniquement sur l'environnement de HAL-Inria et de nous appuyer sur l'expertise d'un groupe de chercheurs pour effectuer les premiers tests et suggestions d'améliorations. Nous avons donc bénéficié d'un public averti qui connaissait le fonctionnement de HAL.

Par ailleurs, une interconnexion entre Hal et Software Heritage a été très rapidement mise en place par les développeurs afin de rendre opérationnel le transfert des

données entre ces deux plateformes. Nous avons débuté par une phase de test pour finaliser les interfaces et les métadonnées.

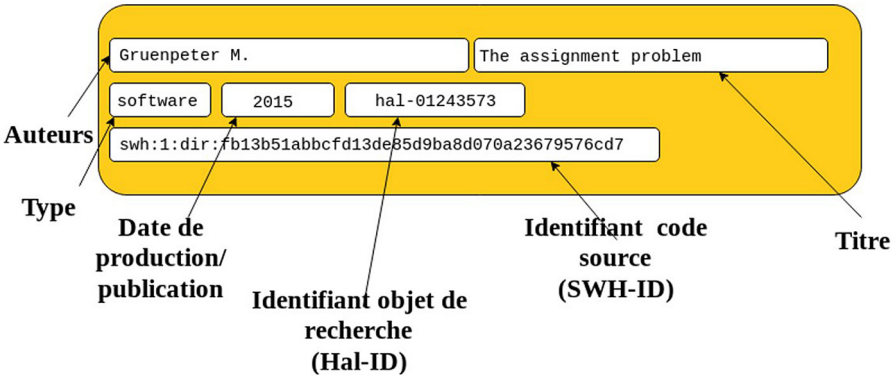
En parallèle, nous avons préparé des guides[13] du déposant et du modérateur. Une fois les tests terminés, nous sommes passés en production pour HAL-Inria dans un premier temps puis sur HAL générique et enfin sur l'ensemble des portails de HAL.

Nous communiquons de plus en plus sur ce service de valorisation du logiciel au travers de HAL et Software Heritage. Par exemple, la campagne annuelle du rapport d'activité des équipes de recherches Inria 2021 a vu l'intégration des logiciels issus de HAL comme éléments de production scientifique au même titre que les publications.

Lors de la mise en place de l'intégration HAL et SWH, les priorités étaient de comprendre l'objet logiciel afin d'ajouter ce nouveau type de document dans HAL. Nous avons ainsi fait évoluer le formulaire de dépôt en reprenant les métadonnées classiques de HAL, en les adaptant au contexte du logiciel et en ajoutant celles manquantes.

Et afin de favoriser la citation ainsi que la reconnaissance des auteurs (au sens de leur reconnaissance dans le milieu de la recherche, la valorisation de leurs travaux), nous avons travaillé sur l'affichage de la citation logiciel sur la notice (figure 2).

Figure 2 : proposition de citation d'un logiciel sur le poster « 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*, janvier 2018, Paris, France. En ligne : https://hal.inria.fr/hal-01688726



Existe-t-il des procédures de contrôle de qualité des métadonnées descriptives dans Software Heritage ?

Software Heritage est une archive universelle qui cherche à collecter, préserver et partager le patrimoine logiciel sans se positionner en tant qu'autorité de métadonnées. C'est pourquoi toutes les informations qui concernent la description ou la curation des

13 Morane GRUENPETER et Jozéfina SADOWSKA. *Create software deposit : User guide and best practices*. [Technical report] Inria ; CCSD ; Software Heritage. 2018. En ligne : https://hal.inria.fr/hal-01872189. Morane GRUENPETER et Jozéfina SADOWSKA. *La modération d'un dépôt logiciel : bonnes pratiques et guide pour le modérateur*. [Rapport technique] Inria ; CCSD ; Software Heritage. 2018. En ligne : https://hal.inria.fr/hal-01876705

logiciels dans l'archive sont factuelles et ne sont pas contrôlées côté SWH. Cela dit, les dépôts logiciels qui utilisent le service *swh-deposit*[14] comme HAL, nécessitent quelques métadonnées minimales pour pouvoir les ajouter dans l'archive. Ces métadonnées sont vérifiées de manière automatique avec le protocole SWORD[15].

Côté HAL, chaque dépôt, que ce soit avec le code source ou en lien avec Software Heritage, est modéré par les professionnels de la documentation.

Quelles sont les principales étapes du workflow ?

Sur la figure 3, vous pouvez visualiser le déroulement complet du processus de vérification et de publication du dépôt logiciel sur HAL jusqu'à son transfert vers SWH. Le contributeur interagit avec la plateforme HAL pour déposer le logiciel et compléter le dépôt.

Le modérateur vérifie les métadonnées, la cohérence entre l'objet déposé et les métadonnées et s'assure que les informations sont correctes et complètes. Dans le cas contraire, le modérateur contacte le contributeur. Enfin, le modérateur valide le dépôt qui sera mis en ligne côté HAL et envoyé chez SWH pour y être archivé. Aucune intervention humaine n'est faite côté SWH.

Les contrôles portent sur le contenu des fichiers d'une part et sur les métadonnées d'autre part. Rappelons que ce sont les chercheurs qui effectuent les dépôts. Un certain nombre de champs sont obligatoires pour assurer un signalement et une citation optimaux. La règle de l'affiliation d'au moins un des développeurs s'applique comme pour tout dépôt en texte intégral sur HAL.

Le contrôle s'applique donc sur la saisie des champs et la cohérence avec le code déposé et les fichiers associés comme le README ou la licence. Nous utilisons la même interface de modération que pour les autres types de dépôt de HAL. Les professionnels de l'information peuvent intervenir directement et corriger le dépôt. Le cas le plus fréquent concerne des erreurs d'affiliations.

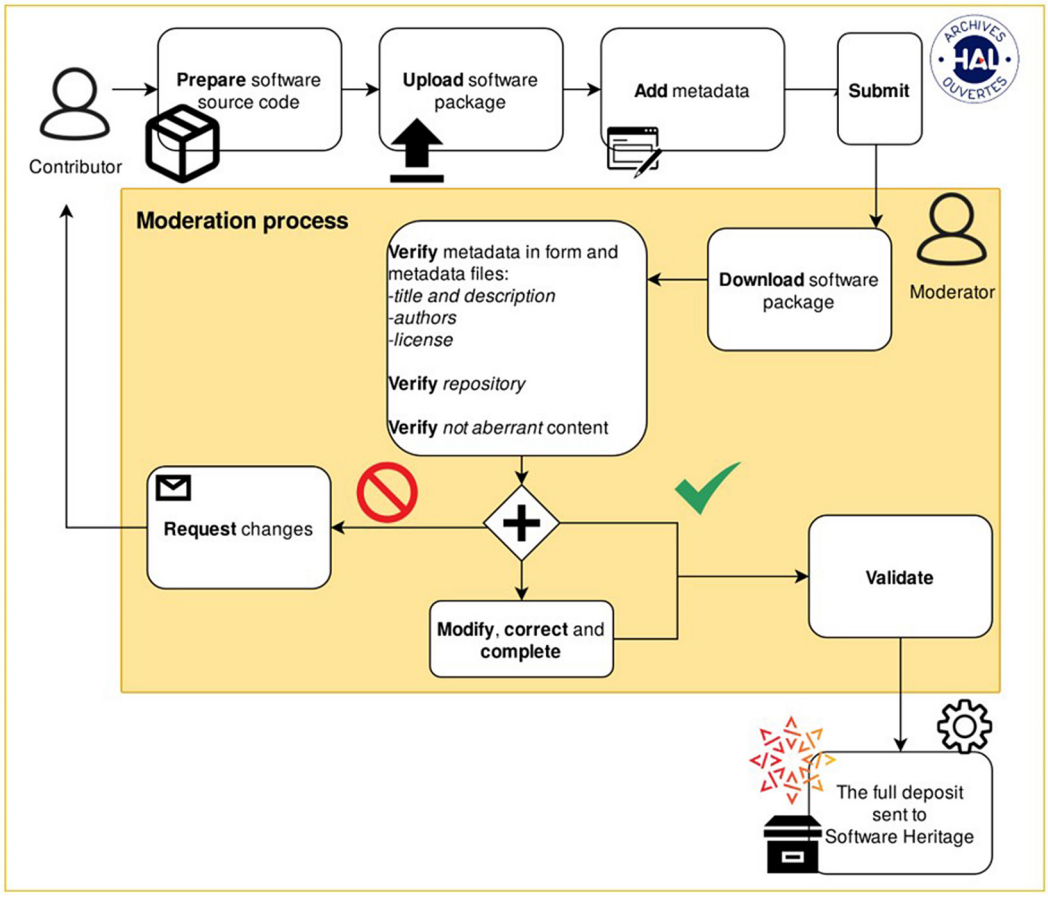
En cas de modification plus importante comme l'absence du fichier README, le modérateur demande des modifications au déposant qui corrige son dépôt conformément aux recommandations du modérateur. Une fois les conditions requises remplies, les modérateurs mettent en ligne le dépôt.

Ainsi le dépôt logiciel bénéficie de : l'environnement et des services de HAL, en particulier pour les référentiels des auteurs ; la très bonne indexation de l'archive et des connexions avec d'autres plateformes comme OpenAIRE. •

14 https://colibris.link/pCMHN

15 https://colibris.link/7u9kt

Figure 3 : circuit de dépôt d'un logiciel
(cette figure apparaît sur : Roberto DI COSMO, Morane GRUENPETER, Bruno MARMOL, *et al.* « Curated Archiving of Research Software Artifacts : Lessons Learned from the French Open Archive (HAL) », *IJDC*. 2020, vol. 15, n° 1.
En ligne : https://doi.org/10.2218/ijdc.v15i1.698)



RESSOURCES POUR LES PROFESSIONNELS DE L'INFORMATION QUI SOUHAITERAIENT SE FORMER : LA SÉLECTION DES AUTEURS

- « Politique de contribution aux logiciels libres de l'État ». Document de la Direction interministérielle du numérique et du système d'information et de communication de l'État (DINSIC) : https://www.numerique.gouv.fr/publications/politique-logiciel-libre/pratique/
- ALLIEZ, Pierre, Roberto DI COSMO, Benjamin GUEDJ, Alain GIRAULT, Mohand-Said 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, p. 1-14. En ligne : https://dx.doi.org/10.1109/MCSE.2019.2949413 et https://hal.archives-ouvertes.fr/hal-02135891v2

- BARBORINI, Yannick, Roberto DI COSMO, Antoine R. DUMONT, Morane GRUENPETER, Bruno MARMOL, *et al.* « 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*, janvier 2018, Paris, France. En ligne : https://hal.inria.fr/hal-01688726
- DI COSMO, Roberto, Morane GRUENPETER, Bruno MARMOL, Alain MONTEIL, Laurent ROMARY, *et al.* « Curated Archiving of Research Software Artifacts : lessons learned from the French open archive (HAL) ». *International Digital Curation Conference (IDCC)*, février 2020, Dublin, Ireland. En ligne : https://hal.inria.fr/hal-02475835
- DI COSMO, Roberto, Morane GRUENPETER, et Stefano ZACCHIROLI. « Identifiers for Digital Objects : the Case of Software Source Code Preservation ». *iPRES 2018 – 15th International Conference on Digital Preservation*. Boston, United States. 2018. En ligne : https://hal.archives-ouvertes.fr/hal-01865790
- DI COSMO, Roberto. « Announcing biblatex-software : software citation made easy », *ACM SIGSOFT Software Engineering Notes*. 12 octobre 2020, vol. 45, no 4. p. 22-23. En ligne : https://doi.org/10.1145/3417564.3417570 (Green open access : https://hal.inria.fr/hal-02977711v1)
- DI COSMO, Roberto. biblatex-software – BibLaTeX stylefiles for software produits. En ligne : https://ctan.org/pkg/biblatex-software
- Documentation HAL dédiée au dépôt logiciel : https://doc.archives-ouvertes.fr/deposer/deposer-le-code-source/
- GOMEZ-DIAZ, Teresa, et Genevieve ROMIER. *Research Software Management Plan template, V3.2*. Bilingual document (FR/EN). 2018. En ligne : https://hal.archives-ouvertes.fr/hal-01802565
- GOMEZ-DIAZ, Teresa. « Article vs. Logiciel : questions juridiques et de politique scientifique dans la production de logiciels », *1024 : Bulletin de la Société Informatique de France*. mars 2015 no 5. En ligne : http://www.societe-informatique-de-france.fr/bulletin/1024-numero-5/
- GRUENPETER, Morane, et Jozéfina SADOWSKA. *Create software deposit : User guide and best practices*. [Technical report] Inria ; CCSD ; Software Heritage. 2018. En ligne : https://hal.inria.fr/hal-01872189
- GRUENPETER, Morane, et Jozéfina SADOWSKA. *La modération d'un dépôt logiciel : Bonnes pratiques et guide pour le modérateur*. [Rapport technique] Inria ; CCSD ; Software Heritage. 2018. En ligne : https://hal.inria.fr/hal-01876705
- JACKSON, Michael. « Software Deposit : Guidance For Researchers ». 7 août 2018. En ligne : https://zenodo.org/record/1327310
- Les documents de référence PLUME pour mieux gérer les développements logiciels, les diffuser et les valoriser dans un laboratoire constituent une mine d'informations : https://www.projet-plume.org/fr/ressource/referentiel-doc-dev. À lire plus précisément les recommandations juridiques et administratives pour diffuser un logiciel de laboratoire : https://www.projet-plume.org/ressource/diffuser-logiciel-recomm-juridiques-admin

- PUBLICATIONS OFFICE OF THE EUROPEAN UNION. Scholarly infrastructures for research software : report from the EOSC Executive Board Working Group (WG) Architecture Task Force (TF) SIRS. 2020. En ligne : https://colibris.link/kQuom
- Software Heritage : https://www.softwareheritage.org/?lang=fr
- TASK FORCE ON BEST PRACTICES FOR SOFTWARE REGISTRIES, Alain MONTEIL, Alejandra GONZALEZ-BELTRAN, Alexandros IOANNIDIS, Alice ALLEN, *et al. Nine Best Practices for Research Software Registries and Repositories : A Concise Guide*. 2021. En ligne : https://hal.inria.fr/hal-03093837
- The CodeMeta Project : https://codemeta.github.io/

PRÉSENTATION DES AUTEURS

Alain Monteil : Coordinateur du pôle Archives Ouvertes au sein du service IES de la DGD-S de l'Inria. Cette mission consiste avant tout à développer et accompagner l'offre de service associée à l'ambition institutionnelle du libre accès aux résultats scientifiques de nos communautés. Il s'agit notamment de favoriser le dépôt du texte intégral dans l'archive mutualisée HAL, de participer aux évolutions de son environnement fonctionnel et technique, tout en améliorant la qualité des données et en favorisant l'accueil de nouvelles collections à forte visibilité pour les sciences du numérique. (CV HAL : https://cv.archives-ouvertes.fr/alain-monteil).

Morane Gruenpeter : Après avoir été harpiste pendant plusieurs années, Morane Gruenpeter a trouvé une nouvelle voie dans l'ingénierie logicielle. Elle a rejoint l'équipe Software Heritage en 2017, à la fin de sa maîtrise en informatique à l'université Pierre-et-Marie-Curie. En 2018-2019, elle continue son travail de recherche en collaboration avec le projet européen EU2020 CROSSMINER pour construire le Web sémantique des projets FOSS par la compilation des métadonnées existantes. Elle est une membre active dans plusieurs groupes de travail dans les domaines de la citation du logiciel, de la préservation digitale et du Web sémantique, y compris : FAIRsFAIR for FAIR Software, Software Citation Implementation WG (FORCE11), Software Identification working WG (RDA & FORCE11) et Wikidata for digital preservation (WikiDigi). (CV HAL : https://cv.archives-ouvertes.fr/morane-gruenpeter).

Józéfina Sadowska : Elle est titulaire d'un DESS en information scientifique et technique et intelligence économique. Après une première expérience en tant que consultante en processus métier dans le domaine des technologies de l'information au Luxembourg, elle rejoint en 2006 le service IES de l'Inria. Chargée de la valorisation de la production scientifique, elle accompagne les chercheurs dans leur cheminement vers la science ouverte via des actions de sensibilisation au libre accès aux publications et aux données. Elle contribue aux évolutions technologiques des outils en lien avec HAL (Haltools, Bib2Hal…) ainsi qu'aux projets d'alimentation automatique de la plateforme (pilote IFIP-DL, imports automatiques de bases de données de laboratoires…).

Depuis 2016, Jozéfina Sadowska participe à l'ajout des logiciels comme nouveau type de production scientifique dans HAL et à la définition d'un lien fiable avec l'archive Software Heritage. (CV HAL : https://cv.archives-ouvertes.fr/jozefina-sadowska).

Estelle Nivault : Après une maîtrise en sciences de l'information et de la communication, Estelle Nivault intègre le CNRS comme chargée de ressources documentaires. Elle développe une expertise dans le domaine de la valorisation et de la diffusion des publications en libre accès. En 2015, elle rejoint l'Inria et pilote, durant trois ans, la migration de la bibliothèque numérique de l'IFIP (International Federation for Information Processing), dans l'environnement HAL. À l'issue de ce projet, elle assure des activités éditoriales en accompagnant un portefeuille de revues en informatique et mathématiques appliquées, hébergées sur la plateforme Episciences.org. Cette mission lui permet notamment de promouvoir l'utilisation du modèle BibTeX pour la citation des logiciels. Aujourd'hui, Estelle Nivault a rejoint le pôle Archives Ouvertes du service IES de l'Inria et consacre son investissement à la visibilité, l'archivage et la qualité du référencement des travaux de recherche en libre accès, dont les logiciels Open Source. (CV HAL : https://cv.archives-ouvertes.fr/enivault).

Résumé

Les logiciels, extension du domaine du signalement ? Assurément. Morane Gruenpeter (Software Heritage, Inria), Alain Monteil, Jozéfina Sadowska et Estelle Nivault (service « Information et édition scientifique », Inria) analysent les défis en matière de métadonnées. Regards croisés d'une développeuse et de trois documentalistes.