Logiciels libres et flux de travail

https://www.fosteropenscience.eu/node/2329

La taxonomie FOSTER définit la science ouverte comme le mouvement visant à rendre la recherche, les données et la diffusion scientifiques accessibles à tous les niveaux d'une société en quête d'informations.

Cela semble une bonne chose, mais que signifie la science ouverte (SO) dans un sens pratique ? Les dix cours de science ouverte de FOSTER répondent à certaines des questions les plus courantes que vous pourriez vous poser sur la mise en pratique de la science ouverte. Chaque cours dure environ 1 à 2 heures et vous recevrez un certificat à la fin. Les cours comprennent des conseils pratiques pour se lancer dans la SO ainsi que des informations sur les outils et les ressources spécifiques à la discipline que vous pouvez utiliser. Il n'y a pas d'ordre précis dans les cours - il suffit d'explorer les sujets que vous souhaitez approfondir à votre propre rythme.

Logiciels libres et flux de travail

https://www.fosteropenscience.eu/node/2329

Cette leçon présente la gestion et le flux de travail des logiciels libres (open source software ou OSS) comme une composante émergente mais essentielle de la science ouverte.

La leçon explique comment le partage et la pérennité des logiciels permettent la reproductibilité, la confiance et la longévité.

Elle ouvre sur différentes perspectives autour du partage et de la réutilisation du code source et des méthodes de calcul par :
- le producteur de logiciels,
- le réutilisateur du code/des portions de code,
- le non-codeur intéressé à reproduire des résultats de recherche ou à poursuivre des processus expérimentaux.

Vous découvrirez des ressources et des outils utiles pour partager et exposer votre code et vos flux de travail.

À l'issue de cette leçon, vous pourrez :
- comprendre les rôles que les logiciels libres et les flux de travail ouverts jouent dans le soutien à la science ouverte
- savoir comment la science ouverte peut favoriser la reproductibilité
- être conscient des questions à prendre en compte aux différents stades du cycle de vie de la recherche
- connaître les outils et ressources utiles pour vous aider à utiliser les logiciels libres et les flux de travail ouverts.

Sommaire

1. Que signifie « open source » ?

Selon Wikipedia, le logiciel libre (OSS) est un type de logiciel informatique dont le code source est publié sous une licence par laquelle le titulaire des droits d'auteur accorde aux utilisateurs le droit d'étudier, de modifier et de distribuer le logiciel.
L'Open source Initiative explique que la licence doit permettre aux utilisateurs d'effectuer des modifications et des travaux dérivés et doit permettre aux travaux dérivés d'être distribués dans les mêmes conditions que celles imposées par la licence du logiciel original.

Cette vidéo produite par Socialsquare vous aide à vous familiariser avec le concept d'open source (en LEGO !).

2. Explication des concepts fondamentaux

2.1. Code logiciel

Le *code logiciel* présente une série d'instructions qui indiquent à un ordinateur d'effectuer une ou plusieurs tâches ou de définir un objet. Le code peut être soit autonome, soit modulaire, par exemple dans le cas de méthodes qui contribuent à une tâche plus importante.

Un programme très simple, souvent la première tâche effectuée lors de l'apprentissage d'un nouveau langage de programmation, est la tâche "Hello World".
En BASIC, ce serait :
10 PRINT "HELLO WORLD"
20 GOTO 10
RUN

Le texte "HELLO WORLD" s'affiche alors à l'écran en continu jusqu'à ce que l'utilisateur appuie sur la touche "Escape" (ou toute autre touche sélectionnée pour arrêter le programme).

Afin de reproduire véritablement les résultats, en particulier ceux des études plus anciennes, il est souvent nécessaire de recréer l'environnement informatique d'origine. Les outils d'émulation et de virtualisation tels que Docker peuvent y contribuer.

2.2. Flux de travail

Un flux de travail scientifique est une série d'étapes répétables qui transforme l'information ou traite des données ou d'autres informations d'une manière ou d'une autre. Ces étapes peuvent être simples et linéaires ou complexes, impliquant des opérations logiques pour décider des étapes suivantes en fonction des données saisies. Les flux de travail sont essentiels pour permettre de valider des résultats publiés.

Un exemple de flux de travail simple pourrait être :
Titre : "Making Toast v1.0"
1. Mettre le pain dans le grille-pain
2. Allumer le grille-pain
3. Attendre que le pain soit suffisamment grillé
4. Retirer soigneusement les toasts et les mettre dans une assiette

Bien sûr, cela suppose qu'il y ait du pain dans la maison. Cela pourrait être amélioré par des étapes expliquant ce qu'il faut faire s'il n'y a pas de pain (par exemple, obtenir de l'argent, aller faire des courses, acheter du pain, rentrer chez soi), et idéalement, cela serait archivé comme une autre version du même flux de travail ("Making Toast v2.0") avec des métadonnées expliquant les changements entre la v1.0 et la v2.0.

Un système de gestion des flux de travail (WfMS) est un système logiciel permettant de mettre en place, d'exécuter et de contrôler une séquence définie de processus et de tâches, avec pour objectifs généraux d'augmenter la productivité, de réduire les coûts, de devenir plus agile et d'améliorer l'échange d'informations au sein d'une organisation. Ces systèmes peuvent être centrés sur les processus ou les données, et ils peuvent représenter le flux de travail sous forme de cartes graphiques. Le système de gestion des flux de travail peut également inclure une interface extensible afin que des applications logicielles externes puissent être intégrées et fournir un support pour des flux de travail à grande échelle qui permettent des temps de réponse plus rapides et une meilleure productivité.

2.3. Reproductibilité et réplicabilité

Il y a une différence entre *reproductibilité* et *réplicabilité*. L'une est plus contraignante que l'autre.

Une étude est ***reproductible*** s'il existe un ensemble spécifique de fonctions/analyses de calcul (généralement spécifiées en termes de code) permettant de reproduire exactement tous les chiffres

d'un article publié à partir de données brutes. Il est maintenant admis que le fait de pouvoir reproduire des analyses de données est un élément essentiel du processus scientifique.
Ce point a été mis en avant en particulier pour les applications de la médecine personnalisée, où des résultats non reproductibles peuvent entraîner des retards dans l'évaluation de nouvelles procédures qui affectent la santé des patients.

Mais ce n'est pas parce qu'une étude est ***reproductible*** qu'elle est ***réplicable***. La *réplicabilité* est plus contraignante que la *reproductibilité*. Une étude n'est ***réplicable*** que si vous pouvez effectuer exactement la même expérience (au moins) deux fois, recueillir des données de la même manière les deux fois, effectuer la même analyse de données et arriver aux mêmes conclusions. La différence avec la ***reproductibilité*** est que pour obtenir la ***réplicabilité***, vous devez à nouveau effectuer l'expérience et collecter les données. Cela introduit bien sûr toutes sortes de nouvelles sources d'erreurs potentielles dans votre expérience (nouvelles personnes, nouveaux matériaux, nouveau laboratoire, nouvelle façon de penser, réglages différents des machines, etc.).
Simply Statistics. Replication, psychology, and Big Science. 18 avril 2012.
https://simplystatistics.org/posts/2012-04-18-replication-psychology-and-big-science/

Le partage et la préservation du code et des flux de travail répondent à ces deux objectifs.

2.4. Normes communes en matière de codage

Si vous travaillez dans un domaine où la reproductibilité est pertinente, préserver et permettre l'accès au code et aux flux de travail est de plus en plus vital pour la crédibilité de votre recherche. Les développeurs professionnels ou expérimentés de logiciels seront certainement conscients des avantages de l'utilisation d'un référentiel de versions et du cycle de vie des logiciels (Développer > Partager > Préserver) ; cependant, de nombreux chercheurs sont des programmeurs autodidactes, et connaissent moins bien les bonnes pratiques et la nécessité d'une documentation lisible pour une réutilisation ultérieure.

Le Software Sustainability Institute et le Software Preservation Network sont deux organisations qui cherchent à améliorer les normes en matière de programmation de la recherche et à fournir des ressources utiles pour vous aider dans votre démarche.

2.5. Comment effectuer une analyse de son flux de travail

Cette vidéo produite par Alicia Hofelich Mohr, responsable des données de recherche à l'Université du Minnesota, Liberal Arts Technologies and Innovation Services (LATIS), offre un aperçu utile des flux de travail et de la manière de les utiliser efficacement dans ses recherches.

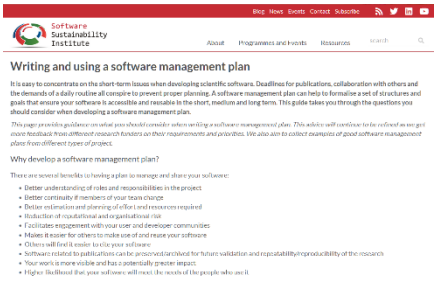
3. Gérer par étapes

3.1. Au tout début du projet (ou avant qu'il ne commence !)

C'est important de créer un plan de gestion du logiciel et de le tenir à jour tout au long du cycle de vie du projet. Certains détails ne seront peut-être connus que plus tard, mais il peut être utile de prendre en compte des questions telles que les formats, les licences et les identifiants pérennes dès le début du projet.

Si vous travaillez avec d'autres personnes, vous devez vous mettre d'accord et adopter des pratiques de codage et de gestion des flux de travail partagés (et adaptés), en accompagnant le code et les flux de travail d'une documentation, de métadonnées riches et de commentaires en ligne.

Le Software Sustainability Institute du Royaume-Uni recommande de rédiger des plans de gestion des logiciels et fournit des conseils sur la manière de le faire.


Blog News Events Contact Subscribe

Software Sustainability Institute

About Programmes and Events Resources

Writing and using a software management plan

It is easy to concentrate on the short-term issues when developing scientific software. Deadlines for publications, collaboration with others and the demands of a daily routine all conspire to prevent proper planning. A software management plan can help to formalise a set of structures and goals that ensure your software is accessible and reusable in the short, medium and long term. This guide takes you through the questions you should consider when developing a software management plan.

This page provides guidance on what you should consider when writing a software management plan. This advice will continue to be refined as we get more feedback from different research funders on their requirements and priorities. We also aim to collect examples of good software management plans from different types of project.

Why develop a software management plan?

There are several benefits to having a plan to manage and share your software:

- Better understanding of roles and responsibilities in the project
- Better continuity if members of your team change
- Better estimation and planning of effort and resources required
- Reduction of reputational and organisational risk
- Facilitates engagement with your user and developer communities
- Makes it easier for others to make use of and reuse your software
- Others will find it easier to cite your software
- Software related to publications can be preserved/archived for future validation and repeatability/reproducibility of the research
- Your work is more visible and has a potentially greater impact.
- Higher likelihood that your software will meet the needs of the people who use it

Dans DMP OPIDoR, vous trouverez un exemple de modèle de plan de gestion spécifique aux logiciels, celui du projet PRESOFT.

3.2. Pendant le projet

Il est habituel de commencer à mettre en place un entrepôt pour permettre le co-développement et la publication des différentes versions de son code. Dans l'idéal, cet entrepôt sera indexé et consultable, afin de respecter les principes FAIR. Ceux-ci ont été créés à l'origine pour les données, mais s'appliquent aussi au code.

Le code logiciel est le plus souvent partagé via GitHub, Bitbucket ou SourceForge. Ceux-ci permettent de conserver et de préserver les anciennes versions des logiciels. Cela protège la robustesse du dossier scientifique et permet de remanier le code lors de futurs travaux, afin de faciliter sa réutilisation.

Attention cependant à la pérennité de ces entrepôts.
Il est conseillé de déposer également les codes sources dans Software Heritage, une archive universelle destinée à préserver sur le long terme tous les codes sources des logiciels (avec attribution d'un identifiant SWHID).

Il est désormais possible de déposer les codes sources dans l'archive ouverte française HAL, avec modération des métadonnées et archivage pérenne dans Software Heritage. Pour être transféré à Software Heritage, le fichier déposé doit être sous licence libre et ne peut pas être sous embargo. La fonctionnalité de dépôt actuelle permet de déposer seulement une archive aux formats zip ou tar.gz. On ne peut donc pas archiver tout un historique de développement, mais seulement un répertoire contenant des fichiers sources à un moment donné.

A étapes régulières, il est judicieux d'archiver les versions majeures afin que chacune ait son propre identifiant pérenne tel qu'un DOI. Cela favorise la reproductibilité en gardant possible l'accès et la référence aux anciennes versions des logiciels, de sorte que les expériences puissent être relancées comme elles l'étaient à l'époque, tout en permettant la poursuite du développement du logiciel.

Les mêmes principes s'appliquent aux flux de travail, la gestion des versions et l'archivage des modifications majeures étant effectués de manière standard.

3.3. Vers la fin du projet

Vers la fin du projet, on peut déposer le code/les flux de travail dans un entrepôt numérique avec des liens vers les publications et/ou les données, ainsi qu'une copie du plan de gestion du/des logiciel(s) (Software Management Plan, SMP) et du/des fichier(s) Readme. Les métadonnées doivent rester accessibles même lorsque le logiciel n'est plus disponible.

Il est souhaitable d'encourager la citation des logiciels dans les documents et chercher à établir des liens permanents entre les données, le code, les publications et les personnes. Ceci est facilité si le logiciel est déposé dans HAL.

Les principes FAIR, développés à l'origine pour les données de recherche, peuvent également être appliqués aux logiciels, par exemple :

- Faciles à trouver : attribution d'un identifiant unique ; description avec des métadonnées riches ; indexation dans une ressource consultable

- Accessibles : logiciel récupérable à l'aide de protocoles standards, ouverts et libres ; métadonnées accessibles même lorsque le logiciel n'est plus disponible
- Interopérables : langage formel, accessible et partagé pour la représentation des connaissances ; références qualifiées à d'autres métadonnées
- Réutilisables : licence claire et publique ; détails sur la provenance ; conformité aux normes de la communauté disciplinaire.

Cette présentation synthétique montre l'importance et l'utilité

- de sauvegarder les codes sources de façon pérenne pour pouvoir les réutiliser
- de savoir comment les déposer et les rechercher sur Software Heritage et sur HAL.

4. Procédure de diffusion des logiciels

Cette procédure s'adapte à chaque situation :

- **Choisir un nom** : éviter les noms déjà utilisés, les marques.
- **Établir la liste des auteurs** en indiquant le pourcentage de participation, leurs affiliations. (*)
- **Établir la liste des fonctionnalités** principales. (*)
- **Établir la liste des briques logicielles**, avec leurs licences. (*)
- **Choisir une licence**, avec l'accord des auteurs et des propriétaires des droits. Si c'est possible, préférer un accord signé. Attention à la compatibilité et à l'héritage des licences.
- **Choisir un site web, une forge ou un entrepôt** pour la diffusion : indiquer les licences et les conditions d'utilisation, les possibilités de copie, comment citer l'œuvre, le PIDs...
- **Créer et indiquer une adresse courriel** de contact.
- **Archiver en .tar.gz régulièrement** car la **traçabilité** est importante. (*)
- **Informer la direction des laboratoires et les tutelles** (si ça n'a pas été fait au point licence).
- **Diffuser le logiciel** (associé aux données).
- **Informer la communauté cible** : considérer la possibilité de rédiger un data/software paper.

(*) A revoir à chaque nouvelle version du logiciel.

Teresa Gomez-Diaz. Logiciels libres/open source dans l'ESR, Science ouverte, Évaluation de la recherche. 9-10 novembre 2021. http://igm.univ-mlv.fr/~teresa/logicielsLIGM/documents/Internacional/2021novOSExp_TGD_ScienceOuverte.pdf

5. Etudes de cas

5.1. Travailler avec des données non structurées dans le domaine des sciences humaines

D'énormes quantités de données sont collectées et utilisées par les chercheurs de toutes les disciplines, y compris les sciences humaines. Avant de pouvoir analyser ces grands ensembles de données pour en dégager des tendances, les chercheurs doivent être capables de donner un sens à ces données.

OpenRefine

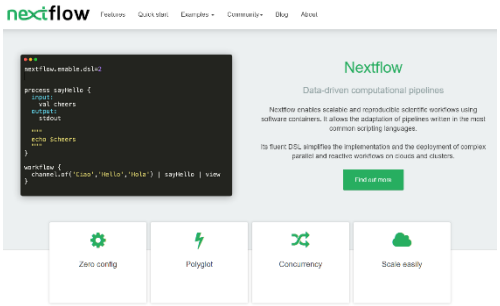
OpenRefine (anciennement Google Refine) est entièrement soutenu par des bénévoles depuis 2012. OpenRefine est un outil gratuit et open source pour travailler avec des données non structurées. Les données peuvent être nettoyées, transformées en différents formats, enrichies avec d'autres outils ou liées à des bases de données.

OpenRefine aide les chercheurs à nettoyer et à traiter de grands ensembles de données, ce qui est essentiel pour gérer les données et les rendre réutilisables.

Pour plus d'informations, suivez cette présentation de Stefan Gaget sur OpenRefine.

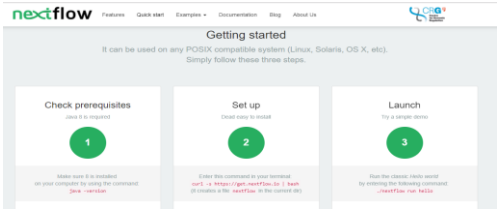
5.2. Rationalisation du développement et de l'exécution des flux de travail informatiques dans les Sciences de la vie

Nextflow



Nextflow est un outil open source qui simplifie le développement et l'exécution de flux de travail d'analyse de données à grande échelle. Il permet aux scientifiques de créer des applications parallèles et distribuées en utilisant un langage spécifique au domaine (Domain-Specific Language, DSL), conçu pour simplifier les tâches et les opérations récurrentes du flux de travail.
Le support intégré de technologies open source telles que celles proposées par Docker, ainsi que l'intégration native de l'outil Git et de plateformes de partage de code comme GitHub, permettent de

- prototyper précisément des flux de calcul autonomes,
- garder une trace de toutes les évolutions au fil du temps
- reproduire rapidement toute configuration antérieure qu'il faudrait réutiliser.



L'analyse de grands ensembles de données de manière performante et reproductible est une question de plus en plus pressante dans de nombreux domaines scientifiques, y compris - et plus particulièrement - dans les disciplines des sciences de la vie.
Ce problème a été alimenté par une double dépendance à des méthodes d'analyse de données de plus en plus complexes et à une croissance exponentielle des ensembles de données biologiques.
Lorsqu'on prend en compte l'installation, le déploiement et la maintenance des pipelines bio-informatiques (groupe de logiciels exécutés en série de telle façon que la sortie d'un logiciel sert d'entrée pour le suivant), une situation encore plus complexe apparaît en raison de l'absence de normes communément admises. En outre, l'effet sur la reproductibilité de cette absence de normes est amplifié par la très grande diversité des plateformes et des configurations de calcul sur lesquelles ces applications sont censées être appliquées.
La technologie open source Nextflow garantit des résultats cohérents dans le temps et permet aux scientifiques de reproduire facilement l'exécution d'analyses complexes de données volumineuses sur différentes plateformes de calcul.

Nextflow : interopérabilité des flux de travail

Kevin Sayers présente « Workflows interoperability with Nextflow and Common WL » dans cette courte vidéo filmée lors de la conférence sur la bio-informatique open source (BOSC) en 2017 à Prague.

5.3. Harmonisation des données dans les Sciences sociales

L'harmonisation des données est une partie importante de l'analyse statistique. Elle exige beaucoup de temps et d'efforts de la part du chercheur pour examiner la documentation de diverses études internationales et pour comprendre la syntaxe utilisée. Pour alléger ce fardeau, plusieurs initiatives européennes s'efforcent de proposer des solutions pour l'harmonisation des données.

5.3.1. SERISS

Synergies for Europe's Research Infrastructures in the Social Sciences (SERISS) a pour objectif de doter les infrastructures de données en sciences sociales européennes d'un rôle majeur dans la résolution des principaux défis sociétaux auxquels l'Europe est confrontée aujourd'hui et à garantir que l'élaboration des politiques nationales et européennes repose sur une base solide de données socio-économiques de la plus haute qualité. Ils développent une plateforme d'harmonisation des données.

En rendant la plateforme d'harmonisation des données librement accessible à tout chercheur, SERISS contribue à améliorer la transparence du processus d'analyse des données.

Cette courte vidéo de Brian Kleiner, FORS, Swiss Centre of Expertise in the Social Sciences, explique le travail de SERISS dans l'amélioration des outils actuels.

5.3.2. CharmStats

CharmStats est un logiciel libre pour l'harmonisation des données. Il a été développé par l'Institut GESIS Leibniz pour les sciences sociales et est disponible en deux versions (compatibles) :
- QuickCharmStats - outil gratuit pour les chercheurs indépendants
- CharmStatsPro - soutien au travail collaboratif des groupes de recherche. Il permet aux chercheurs d'harmoniser les données, de documenter le processus d'harmonisation, ainsi que de publier ladite harmonisation.

En documentant l'ensemble du processus d'harmonisation des données, l'outil logiciel permet aux chercheurs de rendre leurs concepts d'harmonisation réutilisables, par exemple via un serveur de réplication.

6. Testez vos connaissances

Consigne : cochez la bonne réponse

1/2. Parmi les objectifs suivants, quel est celui de la gestion et du partage des logiciels et des flux de travail ?

☐ Reproductibilité
☐ Réplicabilité
☒ Ces deux éléments

Une étude est reproductible s'il existe un ensemble spécifique de fonctions/analyses informatiques (généralement spécifiées en termes de code) qui reproduit exactement tous les chiffres d'un article publié à partir de données brutes. Une étude est réplicable si vous réalisez (au moins) deux fois la même expérience, collectez les données de la même manière, effectuez la même analyse des données et arrivez aux mêmes conclusions.
Le partage et la préservation du code et des flux de travail servent ces deux objectifs.

2/2. Je dois déposer des versions de mon code source ouvert dans un entrepôt approprié à différents stades du cycle de vie du projet

☒ Vrai
☐ Faux

À intervalles réguliers, il est judicieux d'archiver les principales versions afin que chacune d'entre elles possède son propre identifiant pérenne, tel qu'un DOI. Cela favorise la reproductibilité en maintenant

l'accès et la référence aux anciennes versions du logiciel, de sorte que les expériences peuvent être refaites telles qu'elles étaient à l'époque, alors que le logiciel lui-même peut continuer à évoluer.

Vous êtes maintenant prêt à mettre en œuvre les bonnes pratiques en matière de développement de logiciels libres et de flux de travail. Vous pouvez réclamer votre certificat pour avoir suivi ce cours avec succès en utilisant le lien en bas de cette page.
N'oubliez pas de :

- rendre votre code et vos flux de travail accessibles pour favoriser la reproductibilité (et la science ouverte !)
- réfléchir aux besoins des différentes parties prenantes en matière de logiciels et de flux de travail
- utiliser les outils et ressources disponibles gratuitement qui peuvent vous aider à conserver l'accès à votre code et à vos flux de travail

Vous voulez en savoir plus ? Veuillez consulter les ressources supplémentaires ci-dessous. Vous voulez apprendre autre chose ? Alors veuillez sélectionner votre prochain cours dans notre menu principal.

Ressources supplémentaires

- DoRANum. Les codes sources : définition, enjeux et préservation. https://doranum.fr/stockage-archivage/les-codes-sources-definition-enjeux-et-preservation/
- DoRANum. Zoom sur ZWHID. https://doranum.fr/zoom-swhid/
- HAL Documentation. Déposer le code source d'un logiciel. https://doc.archives-ouvertes.fr/deposer/deposer-le-code-source/
- SSI. Software Management Plans. https://www.software.ac.uk/software-management-plans
- Software Carpentry. https://software-carpentry.org/
- Data Carpentry. https://datacarpentry.org/
- Standards des communautés scientifiques :
    - ESIP (Earth Sciences). https://esipfed.github.io/Software-Assessment-Guidelines/
    - CLARIAH (Arts and Humanities). https://github.com/CLARIAH/software-quality-guidelines
    - IPOL (Image Processing). https://tools.ipol.im/wiki/ref/software_guidelines/
    - ELIXIR (Life Sciences). https://elixir-europe.org/communities
- Innovating with Open Knowledge: Open source software with Scott Wilson. https://openinnovation.is.ed.ac.uk/data-software/open-source-software-with-scott-wilson/
- Wikipedia. Technical debt. https://en.wikipedia.org/wiki/Technical_debt
- Code For Science & Society. https://codeforscience.org/
- Code is Science. http://www.codeisscience.com/
- Global Open Science Hardware Roadmap. http://openhardware.science/global-open-science-hardware-roadmap/
- Simply Statistics. Replication, psychology, and Big Science. 18 avril 2012. https://simplystatistics.org/posts/2012-04-18-replication-psychology-and-big-science/
- SSI. Making Code Citable with Zenodo and GitHub. 28 juillet 2015. https://www.software.ac.uk/blog/2016-09-26-making-code-citable-zenodo-and-github
- Carl Boettiger. CodeMeta: a Rosetta Stone for Software Metadata. 30 janvier 2017. https://figshare.com/articles/Codemeta_A_Rosetta_Stone_for_Software_Metadata/4490588
- Greg Wilson et al. Good enough practices in scientific computing. 22 juin 2017. https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005510
- Daniel S. Katz. Using citation to provide credit for software contributions. Août 2017. http://os.helmholtz.de/fileadmin/user_upload/os.helmholtz.de/Workshops/helmholtz_oswebinar42_katz.pdf
- Nicky Phillips. Online software spots genetic errors in cancer papers. 1er novembre 2017. https://www.nature.com/news/online-software-spots-genetic-errors-in-cancer-papers-1.23003
- Roger C. Schonfeld. What is Researcher Workflow? 13 décembre 2017. https://sr.ithaka.org/blog/what-is-researcher-workflow/

- British Ecological Society. A Guide to Reproducible Code in Ecology and Evolution. 2017. https://www.britishecologicalsociety.org/wp-content/uploads/2017/12/guide-to-reproducible-code.pdf
- Best Practices for a Future Open Code Policy for NASA Space Science. 2018. https://www.nationalacademies.org/our-work/best-practices-for-a-future-open-code-policy-for-nasa-space-science
- Teresa Gomez-Diaz. Les logiciels de la recherche et leurs licences : trois visions sur un objet. 10 décembre 2019. https://hal.archives-ouvertes.fr/hal-02434287/document
- Numéro spécial du Bulletin des bibliothèques de France (BBF). 2021-1 : Code source : liberer le patrimoine ! https://bbf.enssib.fr/sommaire/2021/1
- Teresa Gomez-Diaz. Logiciels libres/open source dans l'ESR, Science ouverte, Évaluation de la recherche. 9-10 novembre 2021. http://igm.univ-mlv.fr/~teresa/logicielsLIGM/documents/Internacional/2021novOSExp_TGD_ScienceOuverte.pdf

Remerciements

Nous devons ce cours à la présentation de Neil Chue Hong lors de la conférence Dealing With Data 2017, dont les diapositives et la vidéo sont disponibles sur la page web de la conférence. Il a été mis à jour en 2021.

Ces cours ont été développés en réutilisant des contenus disponibles librement produits par une série de fournisseurs de contenus, notamment DataOne, Research Data Netherlands, Open Data Institute, European Data Portal, Digital Curation Centre, UK Data Service, CESSDA ERIC, DARIAH, ELIXIR, Software Sustainability Institute, FOSTER et bien d'autres qui développent activement des ressources éducatives libres liées à la science ouverte. Ils ont été mis à jour en 2021.

Ils ont été traduits et adaptés à la France par l'Inist-CNRS en 2022.

Les cours sont présentés dans un style similaire à celui utilisé par l'Open Data Institute (ODI) et l'European Data Portal, dans l'espoir que cela permettra à notre contenu d'accroître le corpus de ressources liées à la science ouverte déjà produites et de rendre leur réutilisation collective plus transparente. À cette fin, nous avons aussi fait usage de l'outil de création Adapt, également utilisé par l'ODI et l'European Data Portal.

Nous avons utilisé une variante de l'approche des études de cas développée par l'Open Science Monitor de la Commission européenne pour aider à illustrer les outils et les initiatives utiles du point de vue des disciplines.