Développement d'un formulaire machine-actionnable pour la gestion FAIR des logiciels de recherche dans les plans de gestion de projet

Maria Grazia Santangelo, Violaine Louvet, Sylvie Chachay, Jozefina Sadowska, Maud Medves, Hélène Tisserand, Michel de Moura, Régis Witz, Stéphanie Cheviron, Francis Ogereau, et al.

Développement d’un formulaire machine-actionnable pour la gestion FAIR des logiciels de recherche dans les plans de gestion de projet

Maria Grazia Santangelo (INRIA), Violaine Louvet (Laboratoire Jean Kuntzmann, CNRS), Sylvie Chachay (Université Grenoble Alpes), Jozefina Sadowska (INRIA), Maud Medves (INRIA), Hélène Tisserand (OSU-THETA, CNRS), Michel De Moura (Université de Versailles Saint-Quentin-en-Yvelines), Régis Witz (Université de Strasbourg), Stéphanie Cheviron (Université de Strasbourg), Francis Ogereau (Université Clermont Auvergne), Samrit Mainali (Université Paris-Saclay), Marie Christine Jacquemot (INIST), Valérie Warth (INIST), Julien Caugant (Aix Marseille Université), Yannick Biard (Cirad), Monica Michel Rodriguez (Université de Lille)

Résumé

Les logiciels constituent aujourd’hui l'un des piliers fondamentaux de la recherche, au même titre que les publications et les données. Pourtant, malgré leur rôle central, les logiciels de recherche restent difficiles à trouver, à citer et à référencer correctement. En raison de l'absence de mécanismes appropriés, les logiciels sont souvent négligés ou mal décrits dans les plans de gestion des données. Pour répondre à ce défi, un formulaire dédié aux codes sources et logiciels de recherche a été intégré dans la plateforme DMP OPIDoR. Il permet de décrire les logiciels et les données de recherche associées au sein d’un même plan. Conçu selon les principes FAIR et exploitable par machine, ce formulaire prend en compte les spécificités du cycle de vie des codes sources et logiciels. Il aborde des aspects clés tels que la description générale du logiciel, les outils de développement et environnements d’exécution, la préservation (référencement et archivage pérenne), les questions juridiques (auteurs, licences), ainsi que la valorisation scientifique des productions logicielles.

1 – Introduction

Les logiciels de recherche occupent aujourd’hui une place centrale dans les activités scientifiques, que ce soit par leur utilisation ou leur production. Ils constituent désormais un pilier fondamental de la recherche, au même titre que les publications et les données, tout en présentant des spécificités propres. En effet, ils se caractérisent par une nature dynamique, des dépendances techniques et un cadre juridique particulier. Devenus indispensables dans tous les domaines scientifiques (1), les logiciels interviennent à la fois comme outils de recherche, comme résultats de la recherche et comme objets de recherche à part entière.

Le logiciel est de plus en plus intégré aux feuilles de route science ouverte au niveau national (2) (3) et international (4). En France, cette reconnaissance se traduit par des dispositifs d'appui comme le Comité pour la science ouverte et son collège « codes et logiciels » (5). Au niveau plus opérationnel, le réseau des ateliers de la donnée de Recherche Data Gouv (6) s'est saisi dès sa création de la question d'accompagnement à la gestion des codes et logiciels. Les *Open Source Program Offices* académiques - OSPO (7) commencent à se mettre en place depuis fin 2025 dans les établissements de l'ESR pour définir et promouvoir la stratégie autour de l'ouverture des logiciels.

Malgré leur rôle central dans la recherche, les logiciels restent souvent difficiles à trouver, à citer et à référencer correctement, d'autant que les principes FAIR (Facile à trouver, Accessible, Interopérable et Réutilisable) appliqués aux logiciels de recherche (8) ne sont pas toujours adaptés ou consolidés. En raison de l'absence de mécanismes appropriés, les logiciels sont souvent négligés ou mal décrits dans les plans de gestion des données. Pourtant, ces plans restent un outil essentiel. Ils permettent de décrire de manière simple le logiciel, d'identifier et de planifier les actions de développement et de conservation, tout en tenant compte de ses spécificités et de sa diversité.

Pour initier un plan de gestion de logiciel, les scientifiques peuvent s'appuyer sur plusieurs modèles spécifiquement dédiés aux logiciels de recherche. Le modèle PRESOFT (*Preservation for REsearch SOFTware*) est disponible sur HAL (9), où il peut être téléchargé au format ODT, ainsi que sur la plateforme DMP OPIDoR. D'autres modèles existent également, notamment celui proposé par le *Software Sustainability Institute* (10), accessible dans plusieurs formats texte (odt, md, docx), ainsi que le modèle développé par ELIXIR, disponible via l'outil *Data Stewardship Wizard* (11).

Une autre option consiste à décrire le logiciel dans le plan de gestion du projet où il peut y être décrit comme une production scientifique à part entière, au même titre que les données. Les différents produits de recherche, données, logiciels, sont regroupés dans un même plan, chacun faisant l'objet de formulaires dédiés.

C'est dans ce contexte que s'inscrit notre travail : développer une nouvelle solution de gestion sous la forme d'un formulaire dédié aux logiciels, intégré à un dispositif plus large permettant également la gestion des données de recherche dans un plan de gestion de projet unifié. Nous avons élaboré un formulaire qui :

- prend en compte les spécificités du cycle de vie des logiciels

- permet la gestion de logiciel en tant que produit de recherche au sein d'un plan de gestion projet, soit comme un produit à part entière, soit en liens avec des jeux de données associés
- reste simple d'utilisation tout en étant adapté à la description de tout type de logiciel, du plus simple au plus complexe ;
- encourage l'adoption des principes FAIR adaptés aux logiciels (FAIR4RS, FAIR for research software (12));
- soit disponible dans des outils de rédaction de plan de gestion couramment utilisés par les scientifiques ;
- donne des repères pour réfléchir, adopter des bonnes pratiques et formaliser les choix effectués.

L'objectif principal de cette démarche est d'aider les scientifiques à anticiper les enjeux liés à la production logicielle. En posant les bonnes questions au plus tôt dans le processus de développement, le formulaire met en évidence les pratiques recommandées tout au long du cycle de développement. Il contribue également à clarifier les rôles et responsabilités au sein de l'équipe, à structurer le processus de développement et à guider dans le choix des infrastructures et outils nécessaires pour produire des logiciels de qualité, pérennes et bien documentés dès le début du projet.

Afin de garantir sa découvrabilité et son caractère machine-actionnable, le formulaire a été implémenté dans l'outil DMP OPIDoR[1], l'outil national d'aide à la rédaction des plans de gestion des produits de recherche. En pratique, lors de la création d'un nouveau plan de gestion avec le modèle « Science Europe – modèle structuré » (modèle proposé par défaut dans DMP OPIDoR), il est désormais possible de décrire plusieurs types de produits de recherche : des jeux de données grâce au questionnaire conforme aux pré-requis de Science Europe et des logiciels grâce au nouveau questionnaire dédié.

Il conduit ainsi à la génération d'un plan machine-actionnable de par l'utilisation d'identifiants, des vocabulaires contrôlés et des référentiels, et la génération du formulaire en format json (le format lisible par la machine). Celui-ci s'accompagne en outre de recommandations explicitant les attendus pour chaque question et les bonnes pratiques à adopter.

[1] DMP OPIDoR : https://dmp.opidor.fr/

Dans ce rapport, nous présentons la méthodologie employée pour concevoir le formulaire, le cadre de sa production, sa structure ainsi qu'une discussion à propos de la mise en œuvre des principes FAIR et leur apport pour l'ESR.

2 – Méthodologie

2.1 – Équipe de travail

Notre groupe de travail a réuni des membres des différents ateliers de la donnée de l'écosystème Recherche Data Gouv. La conception du questionnaire a été menée grâce à une collaboration étroite entre des profils très variés au sein du groupe de réflexion du réseau des ateliers de la donnée. L'équipe réunissait des documentalistes spécialisés en structuration de métadonnées et en rédaction des plans de gestion de données, des scientifiques et des informaticiens familiers des pratiques concrètes de développement logiciel, ainsi que des membres du CNRS - l'INIST[2] responsables du développement de l'outil DMP OPIDoR.

Cette diversité a permis de couvrir un large spectre de sujets : formalisation des informations pour les référentiels, prise en compte des réalités de terrain des développeurs, et cohérence avec les exigences institutionnelles ou nationales en matière de science ouverte.

La complémentarité des compétences et le travail apporté aux formulations ont joué un rôle clé dans la qualité du formulaire. Les documentalistes ont notamment assuré la clarté des libellés et la cohérence des recommandations éditoriales, tandis que les développeurs ont contribué par leur connaissance des pratiques informatiques courantes.

Le croisement de ces points de vue a permis de produire des formulations claires et partagées, compréhensibles par les développeurs, respectant les règles de structuration documentaire et utilisables avec les outils de saisie et d'export automatisés de la plateforme DMP OPIDoR.

2.2 – Construction du formulaire

Le travail s'est appuyé sur une méthodologie collaborative et itérative. Pour élaborer le formulaire dédié aux logiciels, nous sommes partis des sections qui structurent le modèle du Plan de Gestion de Données Science Europe tel qu'il est implémenté dans la plateforme DMP OPIDoR[3]. Ce modèle est organisé en plusieurs blocs correspondant aux grandes sections d'un

[2] CNRS-INIST : https://www.inist.fr/
[3] DMP OPIDoR data model for machine actionable DMPs : https://opidor.github.io/maDMP-OPIDoR_documentation/

plan de gestion de données. L'objectif était d'identifier où et comment intégrer la partie dédiée aux logiciels, tout en assurant l'interopérabilité et l'exploitation des informations. En particulier, le bloc « Produit de recherche » contient les sections et sous-sections permettant de décrire les objets tels que jeux de données, workflows et logiciels mais sans proposer des champs spécifiques dédiés à chaque produit. Nous avons vérifié lesquelles de ces sections pouvaient être réutilisées pour décrire les logiciels et évalué la nécessité d'en créer de nouvelles afin de mieux refléter leurs spécificités. En résultat, le modèle a été étendu pour inclure de nouvelles classes pour décrire le contenu d'un plan de gestion de logiciel.

Pour adapter les sections aux particularités du cycle de vie des codes sources et des logiciels, un effort particulier a été fait pour aligner un certain nombre de propriétés des sections sur celles contenues dans CodeMeta[4], qui fournit un standard de métadonnées largement reconnu pour les logiciels. Ce choix nous a permis de décrire les logiciels de manière structurée, en tenant compte de leurs caractéristiques et de leur cycle de vie complet.

Nous avons associé à ces sections des recommandations spécifiques visant à aider les scientifiques à anticiper les enjeux liés au développement logiciel et à mettre en lumière les bonnes pratiques à adopter tout au long du projet[5]. Ces recommandations ont été élaborées avec le soutien du Collège codes sources et logiciels du Comité pour la science ouverte (5).

3 – Structure du modèle

Dans ce paragraphe, nous présentons les différentes sections et sous-sections qui structurent le formulaire, ainsi que les recommandations associées (les recommandations ne concernent que certaines sous-sections).

3.1 – Description générale du logiciel

Cette première section vise à regrouper les informations descriptives permettant d'identifier, de caractériser et de référencer le logiciel. Il s'agit par exemple du nom, du domaine d'application, des mots-clés scientifiques et techniques, de la typologie du logiciel, ainsi que des liens éventuels avec d'autres produits de recherche.

[4] Code Meta Generator : https://codemeta.github.io/codemeta-generator/
[5] Recommandations Codes et Logiciels : https://dmp.opidor.fr/public_guidance_groups?page=1&search=Recherche+data+gouv

Description générale du logiciel		
Sous sections	**Description des sous sections**	**Recommandations**
Nom complet du logiciel	Dénomination officielle du logiciel tel qu'il est diffusé ou référencé.	
Description du logiciel	Présentation synthétique des principales fonctionnalités, objectifs et usages du logiciel.	Préciser quel est l'objectif du logiciel et indiquer à quel public potentiel (cible) le logiciel s'adresse.
URL du site de présentation du logiciel (ou du projet)	Lien vers la page officielle du logiciel ou du projet associé, permettant d'accéder à des informations détaillées.	
Date de début du développement	Date à laquelle le développement du logiciel a effectivement commencé.	
Date de première version	Date de mise à disposition de la première version fonctionnelle du logiciel.	
Principaux domaines d'application	Champs disciplinaires ou secteurs dans lesquels le logiciel est principalement utilisé.	Préciser la ou les disciplines au sein de laquelle le logiciel a été développé
Autres domaines d'application	Domaines secondaires ou usages complémentaires.	Préciser si votre logiciel peut s'appliquer à d'autres domaines
Mots clés contrôlés (vocabulaire, thésaurus, lexique...)	Termes issus de vocabulaires normalisés, thésaurus ou lexiques spécifiques permettant d'indexer le logiciel de manière rigoureuse.	Peuvent être liés au projet, à une méthode scientifique, au programme, aux données manipulées ou produites. Utiliser de préférence des mots-clés issus de thésaurus. On peut utiliser plusieurs thésaurus.
Numéro de version du logiciel	Version actuelle du logiciel.	Une bonne pratique pour les numéros de version est le « semantic versioning »[6].
Quelle est la (ou les) typologie(s) du logiciel ?	Nature et catégorie du logiciel. Par exemple, outil de simulation, bibliothèque, logiciel embarqué	
Votre logiciel est-il lié à d'autres produits de recherche ?	Indication d'une éventuelle articulation avec d'autres résultats de recherche.	Penser à faire le lien entre votre code et les autres produits de recherche du projet, par exemple les données traitées via le logiciel et déclarées dans le même plan

[6] Gestion sémantique de version : https://semver.org/lang/fr/

Le développement de votre logiciel implique-t-il des coûts ?	Indication de l'existence de ressources financières ou humaines spécifiques engagées dans le développement du logiciel.	Indiquer, par exemple, les coûts liés à des prestations externes ou à du stockage
Personne contact	Indication de la personne responsable du logiciel.	

3.2 – Outils et environnement d'exécution

Cette section regroupe les informations relatives à l'environnement de développement, aux outils de gestion du code, à la qualité logicielle ainsi qu'aux conditions d'exécution du logiciel. Elle permet de documenter les aspects techniques essentiels à sa maintenance, son évolution et son utilisation.

Outils et environnement d'exécution		
Sous sections	**Description des sous sections**	**Recommandations**
Environnement de développement		
Utilisez-vous une forge logicielle ?	Indication de l'éventuel hébergement du code source sur une forge logicielle (GitHub[7], GitLab[8], SourceForge[9]).	Une forge logicielle est un système de gestion de rédaction, de partage et de maintenance collaborative de texte. C'est un environnement web constitué d'un ensemble d'outils, issus en particulier du génie logiciel, afin de faciliter le développement de codes sources. L'utilisation d'une forge logicielle permet d'assurer la sauvegarde du code sur un serveur et est un outil indispensable dès qu'on développe à plusieurs. Consulter le rapport du COSO (13) pour avoir plus de détails sur les fonctionnalités des forges logicielles et consulter une liste non-exhaustive de forges logicielles publiques de l'ESR.

[7] GitHub : https://github.com/
[8] GitLab : https://about.gitlab.com/
[9] SourceForce : https://sourceforge.net/about

Indiquer, le cas échéant, le lien vers le dépôt du code ou l'URL d'accès	Adresse du dépôt public ou privé contenant le code source.	
L'accès au code source est-il ?	Nature de l'accès au code (ouvert, fermé ou propriétaire).	
Utilisez-vous un gestionnaire de version ?	Outil utilisé pour le suivi des versions et la gestion collaborative du code.	L'usage d'un gestionnaire de version est particulièrement utile pour tracer tous les développements et pouvoir revenir en arrière le cas échéant. C'est la bonne pratique lorsqu'on développe du code.
Quels types de tests utilisez-vous pour vérifier la qualité du code ?	Description des tests mis en œuvre pour assurer la qualité du code (tests unitaires, tests d'intégration, tests fonctionnels, etc.)	
Utilisez-vous un framework d'intégration continue ?	Indication de la mise en œuvre d'un outil d'intégration continue (CI) pour automatiser les processus de test et de déploiement.	Penser à faire au minimum des tests fonctionnels pour vous assurer que les évolutions du code n'impactent pas les résultats attendus. L'intégration continue permet en particulier de lancer ces tests automatiquement à chaque modification du code enregistrée par le gestionnaire de version (exemple : lors des commits via Git)
URL du framework d'intégration continue, s'il existe ?	Lien vers la plateforme d'intégration continue utilisée.	
Informations complémentaires	Tout élément utile pour mieux comprendre le contexte ou les choix techniques effectués lors du développement.	
Documentation		
Indiquer les liens vers la documentation utilisateur et/ou développeur si elle est disponible	URL vers la documentation disponible en ligne, destinée aux utilisateurs finaux et/ou aux développeurs contribuant au projet.	Penser à rédiger au minimum un fichier README avec les éléments permettant l'installation du code (dépendances, …) et la façon de l'utiliser. Idéalement, intégrer des exemples d'utilisation. Et si l'objectif est d'avoir des contributions externes, il est nécessaire de prévoir une documentation développeur

Environnement d'exécution		
Quels langages de programmation sont utilisés dans le logiciel ?	Liste des langages utilisés pour le développement du logiciel (ex. : Python, Java, C++).	Si c'est pertinent et applicable, penser à préciser quelle version du ou des langages de programmation vous avez utilisé (Python 2, Python 3, etc.)
Sur quels systèmes d'exploitation votre code s'exécute-t-il ?	Systèmes sur lesquels le logiciel peut être installé et exécuté (ex. : Linux, Windows, macOS).	Si vous souhaitez que votre logiciel soit utilisé à une échelle assez large, assurez-vous qu'il soit suffisamment portable sur les principaux systèmes d'exploitation. Dans tous les cas, précisez les systèmes sur lesquels le logiciel est censé fonctionner
Quelles sont les dépendances requises pour l'utilisation du logiciel ?	Bibliothèques, packages ou outils externes nécessaires à l'installation et à l'exécution du logiciel.	Par exemple, si c'est pertinent, préciser le compilateur, l'outil de construction de programme, les bibliothèques utilisées, l'environnement d'exécution (notebook par exemple). Si vous avez un fichier 'Requirements', ajoutez-le comme élément en précisant son lien URL ou listez les dépendances individuellement
Informations complémentaires	Précisions techniques ou recommandations liées à l'installation ou à l'utilisation du logiciel dans différents environnements.	

3.3 – Préservation du logiciel

Cette section présente les informations sur le référencement et l'archivage pérenne du logiciel.

Préservation du logiciel		
Sous sections	**Description des sous sections**	**Recommandations**
Référencement		
Indiquer le ou les catalogues dans lequel est signalé le logiciel ainsi que son identifiant pérenne (ou l'URL du dépôt)	Sélection ou ajout du nom de catalogue (ou d'entrepôt) dans lequel est signalé le logiciel. Indication de l'identifiant pérenne (DOI, HAL Id, …) ou l'URL du dépôt puis	Les métadonnées associées au signalement du logiciel dans une archive ouverte de confiance permettent une meilleure visibilité et la possibilité de faire le lien

	spécification du type d'identifiant.	avec les publications et les données. Pour faciliter le dépôt, il est possible d'utiliser l'outil Code Meta Generator qui génère un fichier codemeta.json interprété automatiquement par les entrepôts. Éviter les archives privées. Il est conseillé de faire le lien entre la notice dans HAL et l'archivage dans Software Heritage[10].
Informations complémentaires	Ajout de toute information complémentaire concernant le signalement de logiciel.	
Archivage pérenne		
Si votre logiciel est archivé dans l'archive Software Heritage, indiquer l'identifiant SWHID du dépôt	Indication de l'identifiant SWHID. Il se trouve sur la page du projet à partir de l'archive Software Heritage[11].	Software Heritage assure le moissonnage régulier des projets publics présents dans les principaux systèmes de forges et de référencement de paquets (GitHub, GitLab, R Archive Network, etc.)[12].

.4 – Questions juridiques

Cette section présente les informations sur les droits, licences et responsabilités.

Questions juridiques		
Sous sections	**Description des sous sections**	**Recommandations**
Lister les auteurs du logiciel	Possibilité d'ajout des informations concernant les auteurs : Nom, Prénom, Identifiant et son Type, Email, Affiliation et son Identifiant. L'interface propose des liens avec les référentiels : ORCID, ROR. Attribution du rôle par défaut : « Auteur ».	Un auteur est une personne ou une organisation qui a apporté une contribution significative et intellectuelle au développement du logiciel. Il est recommandé d'avoir un identifiant auteur ORCID[13]. Pensez à créer un fichier auteurs.
Lister les principaux contributeurs du	Possibilité d'ajout des informations concernant les	Les contributeurs non auteurs sont les personnes ou

[10] Déposer le code source d'un logiciel sur HAL : https://doc.hal.science/deposer/deposer-le-code-source/
[11] Software Heritage : https://archive.softwareheritage.org/
[12] Toutes les informations pour s'assurer de l'archivage dans Software Heritage sont en ligne sur https://www.softwareheritage.org/how-to-archive-reference-code/
[13] Créer un identifiant ORCID : https://orcid.org/register

logiciel puis leur attribuer un rôle	contributeurs : Nom, Prénom, Identifiant et son Type, Email, Affiliation et son Identifiant. L'interface propose des liens avec les référentiels : ORCID, ROR. Possibilité de choix du rôle spécifique à la gestion des logiciels pour les contributeurs dans une liste contrôlée : Architecture, Conception, Débogage, Développement, Documentation, Maintenance, Management, Support, Test ou Autre.	organisations qui ont contribué de façon non substantielle à la version du code publié. Il peut s'agir d'une ou plusieurs personnes, d'une équipe, d'un service.
Sous quelles licences est ou sera diffusé le logiciel ?	Choix de la ou des licences dans la liste contrôlée proposée basée sur SPDX.	La licence doit idéalement être mise en place avant le développement du logiciel, dans le respect des éventuels contrats et conventions et avec l'accord de la tutelle concernée et de l'ensemble des co-auteurs, et de toute façon impérativement avant sa diffusion. Elle doit aussi tenir compte des conditions de diffusion des briques logicielles préexistantes intégrées dans le code. Avant de choisir la licence contacter le service de valorisation de votre établissement. Dans le cadre de la science ouverte, préférer des licences libres. Penser à créer un fichier licence, et intégrer l'entête associé dans chaque fichier du code source. De préférence, indiquez la licence via son url sur le site SPDX[14].
Informations complémentaires	Ajout de toute information complémentaire en lien avec les aspects juridiques comme par exemple l'accord de consortium, les droits de propriété intellectuelle.	

[14] SPDX Licence List : https://spdx.org/licenses/

3.5 – Valorisation scientifique

Cette section précise si le logiciel a fait l'objet ou est lié à des publications scientifiques.

Valorisation scientifique		
Sous sections	**Description des sous sections**	**Recommandations**
Le logiciel a-t-il fait l'objet d'une publication scientifique dédiée ?	Indication du titre et de l'identifiant de la ressource dédiée au logiciel par exemple un *software paper*.	Indiquer la publication qui décrit votre logiciel. Consulter la ressource Doranum[15] qui définit ce qu'est un *software paper* et décrit quelques règles de rédaction. Vous pouvez également accéder à la liste des journaux qui permettent ce type de publications[16].
Le logiciel a-t-il été associé à des publications scientifiques ?	Indication du titre et de l'identifiant de la ressource où logiciel est cité.	Indiquer les principales publications liées aux logiciels.

4 – Discussion et Conclusions

Depuis juin 2025, DMP OPIDoR propose un questionnaire dédié aux codes sources et aux logiciels pour les plans de gestion de données élaborés avec le modèle Science Europe structuré. Il est désormais possible, au sein d'un même plan pour un projet ou pour une entité, de décrire séparément la gestion d'un logiciel de recherche et celle d'un jeu de données grâce aux formulaires dédiés, tout en tenant compte de la nature distincte des données et des logiciels.

Toutes les informations saisies via les formulaires sont structurées et standardisées. Elles deviennent ainsi interopérables et exploitables par les machines (*machine-actionable*), et donc facilement réutilisables par d'autres systèmes d'information. Il est notamment possible de télécharger le plan de gestion de données incluant des produits de recherche de type logiciel au format json DMP OPIDoR. L'une des étapes à venir sera de permettre l'export au format RDA DMP Common Standards incluant le nouvel objet logiciel. Nous finalisons également le mapping entre les champs des formulaires et le CodeMeta afin de proposer un import/export des données avec ce format. Ceci pour faciliter la réutilisation et l'échange des informations

[15] Le software papers : https://doranum.fr/data-paper-data-journal/les-software-papers_10_13143_hgwj-8s90/
[16] In which journals should I publish my software : https://www.software.ac.uk/top-tip/which-journals-should-i-publish-my-software

entre les différentes sources : forges logicielles, les entrepôts de données ou les archives ouvertes de publications.

Le travail que nous avons réalisé vise à favoriser la découvrabilité, la préservation à long terme et la pérennité des logiciels, répondant ainsi aux enjeux de reproductibilité et de réutilisation.

Notre formulaire a été conçu pour faciliter et encourager l'adoption des principes FAIR appliqués aux logiciels. La figure 1 illustre la correspondance entre les sections du formulaire et les principes FAIR, et montre comment un plan de gestion logiciel exploitable par machine contribue à l'application de FAIR4RS et à la valorisation des logiciels. Les différentes sections — qu'il s'agisse des informations sur le projet, sur le plan de gestion de données, de la définition du logiciel, des aspects juridiques, de la documentation, ou encore des informations sur l'outil et sur son environnement d'exécution — couvrent déjà un premier ensemble de principes FAIR. Elles permettent notamment de décrire le logiciel à l'aide de métadonnées riches. Elles garantissent aussi l'utilisation d'attributs précis et pertinents facilitant sa découverte. Enfin, elles incluent des références qualifiées vers d'autres logiciels et plus généralement d'autres objets de recherche.

Pour les autres principes FAIR, leur mise en œuvre repose sur les fonctionnalités offertes par différentes plateformes permettant le référencement (comme HAL ou Zenodo), la citation des logiciels, l'archivage du logiciel (comme Software Heritage) et la publication de *software papers*.

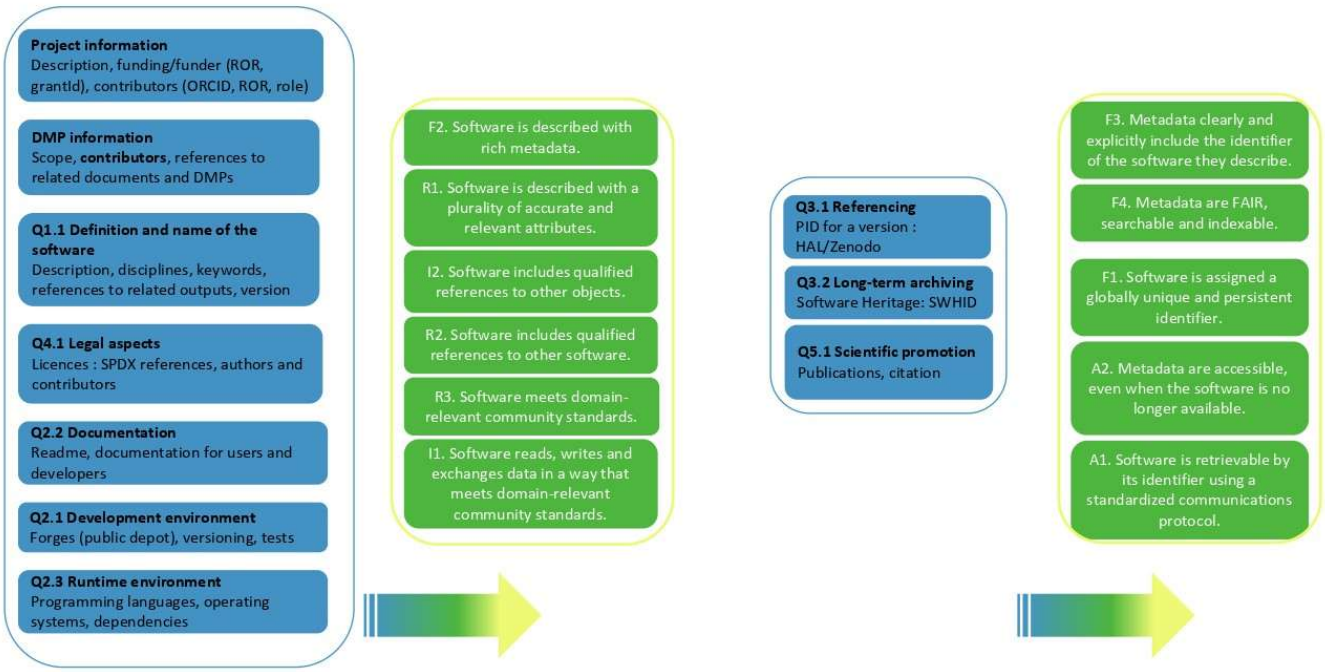

Figure 1 : Correspondance entre les différentes sections du formulaire (en bleu) et les principes FAIR pour les logiciels (en vert).

Notre questionnaire adresse la question de la maintenance du logiciel en encourageant l'utilisation de forges logicielles facilitant les collaborations et contributions. Ainsi, il permet de réfléchir à la création d'une communauté autour de celui-ci au-delà de la durée du projet financé. Cela est en cohérence avec la nature dynamique des logiciels, dont le cycle de vie peut se prolonger bien après la fin du projet de recherche ayant permis leur développement (1).

Enfin, il peut servir de base pour l'élaboration des autres modèles par des agences de financement qui souhaiteraient intégrer le logiciel dans leurs politiques, mais aussi dans l'élaboration des plans de gestion pour des établissements, unités de recherche, infrastructures ou plateformes produisant et développant des logiciels.

Bibliographie

1. Production et valorisation des logiciels issus de la recherche publique française : https://www.enseignementsup-recherche.gouv.fr/fr/production-et-valorisation-des-logiciels-issus-de-la-recherche-publique-francaise-98204

2. 2eme Plan national pour la science ouverte : https://www.enseignementsup-recherche.gouv.fr/sites/default/files/2021-09/2e-plan-national-pour-la-science-ouverte-12968.pdf

3. La feuille de route 2021-2024 du MESRI sur la politique des données, des algorithmes et des codes sources : https://www.enseignementsup-recherche.gouv.fr/fr/la-feuille-de-route-2021-2024-du-mesri-sur-la-politique-des-donnees-des-algorithmes-et-des-codes-50534

4 .Open source software strategy : https://commission.europa.eu/about/departments-and-executive-agencies/digital-services/open-source-software-strategy_en

5. Collège codes et logiciels du comité pour la science ouverte : https://www.ouvrirlascience.fr/college-codes-sources-et-logiciels/

6. Réseau des ateliers de la donnée de l'écosystème Recherche Data Gouv : https://recherche.data.gouv.fr/fr/page/reseau-des-ateliers-de-la-donnee

7. Réseau des Open Source Programme Office (OSPO) au sein du CURIOSS : https://curioss.org/news/academic_ospo_defn/

8. Barker, M., Chue Hong, N.P., Katz, D.S. et al. Introducing the FAIR Principles for research software. Sci Data 9, 622 (2022). https://doi.org/10.1038/s41597-022-01710-x

9 Teresa Gomez-Diaz, Genevieve Romier. Research Software Management Plan template, V3.2. 2018. (hal-01802565)

10. Checklist for a Software Management Plan : https://zenodo.org/records/2159713

11. ELIXIR Software Mannagement Plan : https://elixir-europe.org/sites/default/files/documents/software-management-plan.pdf

12. FAIR Principles for Research Software (FAIR4RS Principles) : https://zenodo.org/records/6623556

13. Forges de l'Enseignement supérieur et de la Recherche - Définition, usages, limitations rencontrées et analyse des besoins : hal-04098702