Quelles stratégies de diffusion et pérennité pour les logiciels de recherche dans les établissements publics français ?

Sylvie Tonda-Goldstein, Mélanie Clément-Fontaine, Nicolas Jullien, Jeanne Robineau, Francois Sabot

Ouvrir la science !

Quelles stratégies de diffusion et pérennité pour les logiciels de recherche dans les établissements publics français ?

Sylvie TONDA-GOLDSTEIN – Pilotage
Mélanie CLEMENT-FONTAINE
Nicolas JULLIEN
Jeanne ROBINEAU
François SABOT

Avril 2025

Quelles stratégies de diffusion et pérennité pour les logiciels de recherche dans les établissements publics français ?

Groupe de travail 3 - Collège Codes Sources et Logiciels / *Valorisation et durabilité*

Sylvie TONDA-GOLDSTEIN – Pilotage
Inria
https://www.inria.fr

Mélanie CLÉMENT-FONTAINE
Université Paris-Saclay Versailles Saint Quentin en Yvelines
https://www.uvsq.fr/

Nicolas JULLIEN
Institut Mines Télécom Atlantique
https://www.imt-atlantique.fr/

Jeanne ROBINEAU, François SABOT
Institut de Recherche pour le Développement
https://www.ird.fr/

Avril 2025

DOI : 10.52949/82

Conception graphique : opixido

Résumé

Le présent document propose une analyse de la diffusion et de la pérennité des travaux scientifiques de type logiciel produits par les établissements de recherche français. Partant du constat qu'il n'existe pas de définition partagée du concept de valorisation de la recherche et que le Haut conseil d'évaluation de la recherche et de l'enseignement supérieur (Hcéres) considère que c'est un sujet particulier qui a besoin d'une définition rigoureuse et partagée, la définition proposée en 2020 par l'Hcéres ne couvre pas la valorisation académique des productions de la recherche telles que les publications scientifiques ; pourtant, l'enquête du MESR de 2023-2024 met en exergue que les scientifiques ayant répondu à l'enquête citent largement cette valorisation. Nous décidons tout au long de ce rapport de ne pas employer le terme « valorisation de la recherche », et utiliserons les termes « diffusion » et « pérennité », dynamique d'innovation, transfert, impact. Par cet ensemble de termes, nous souhaitons couvrir tous les axes : socio-économique, économique, scientifique (citations des logiciels ou des publications, participation à des projets de recherche), notoriété.

Cette analyse prend appui sur un travail d'enquête mené sur une dizaine de projets logiciels, dont six projets étudiés en détail, avec 13 entretiens réalisés. Après avoir présenté l'affinité pour les logiciels ouverts dans la partie 1, et un état des lieux de l'écosystème actuel des logiciels de recherche dans la partie 2, notamment les types de logiciels produits, nous faisons une synthèse, informée par les résultats de l'enquête, des perspectives de diffusion et promotion des logiciels de recherche et des investissements nécessaires pour ce faire (partie 3).

Les logiciels sont des productions intellectuelles codées en langage informatique pour effectuer des tâches spécifiques. Ils peuvent évoluer, être diffusés et avoir une existence autonome. Faire d'un logiciel un bien échangeable nécessite, comme pour les autres « biens publics »[1] et, parmi eux, les biens produits intellectuellement, de créer les conditions techniques, en particulier juridiques, par le biais des droits de la propriété intellectuelle (DPI), permettant de déterminer la titularité des droits, les modes d'exploitation et de protection. Une fois la titularité des DPI déterminée, les titulaires des droits peuvent contrôler l'accès à la production du logiciel et en fixer les contreparties, selon les éléments de diffusion et pérennité recherchés, *via* un contrat, qui sera le plus souvent appelé une « licence d'utilisation de logiciel ». Ces contrats peuvent être à titre onéreux ou pas, les conditions d'utilisation peuvent encadrer les possibilités d'améliorer le logiciel ou encore la manière de citer les contributeurs (notamment au sein de la communauté scientifique).

En permettant à un acteur de contrôler l'accès à un code source, le régime des DPI permet au titulaire des droits de contrôler aussi les évolutions futures de ce logiciel ainsi que la dynamique d'innovation qui peut se développer autour de lui (le « développement continu du logiciel », Fitzgerald et Stol, 2017). Le logiciel libre est une stratégie qui consiste à renoncer à une partie du contrôle sur le patrimoine logiciel existant afin de dynamiser le développement continu et l'innovation. On peut comprendre pourquoi la recherche, dont l'un des objets est de favoriser l'innovation, est sensible à cette stratégie de diffusion et pérennité. C'est cependant une stratégie parmi d'autres, qui doit être évaluée (examen coût-avantage), par rapport aux bénéfices attendus, au flux d'innovation et aux contributions logicielles générés.

[1] Au sens économique du terme.

Notre étude a consisté à examiner les coûts et les avantages d'un tel choix grâce à une analyse approfondie des cycles de vie de logiciels de recherche, en se basant sur des approches telles que celles de l'European Open Science Cloud (EOSC) et le concept de développement continu.

Elle conclut d'abord que le cycle de vie d'un projet logiciel de recherche se divise en 4 étapes, qui peuvent ne pas concerner tous les projets. Ces étapes incluent la preuve de concept, l'initiation d'une collaboration scientifique, le développement continu en collaboration, et la production de logiciels métiers.

- Preuve de concept : Un logiciel naît souvent d'un projet de recherche pour répondre à une question spécifique. Le logiciel est destiné à une utilisation interne par les chercheurs pour valider des hypothèses scientifiques. S'il est nécessaire pour la publication des résultats, il peut être diffusé, mais sans forcément de volonté de le développer au-delà. Cette diffusion ou ce premier travail peut initier de nouveaux développements et collaborations.
- Initiation de collaborations : Si le projet se révèle prometteur, il peut continuer d'évoluer au-delà de la première publication. Cette phase implique souvent des collaborations internes et/ou externes, où les chercheurs à l'initiative du logiciel coordonnent son développement. Cela peut impliquer une gestion plus formelle, notamment si des contributions externes (comme des rapports de bogues ou des demandes d'évolution) sont reçues. Cela se fait souvent via des plateformes collaboratives éventuellement hébergées par l'institution, et demande une implication de l'équipe de recherche. Elle en retire des bénéfices en termes de publication scientifique et de participation à des projets de recherche.
- Développement continu en collaboration : Si le projet continue de croître, il devient obligatoire de structurer les contributions et d'organiser le développement de manière plus formelle, en mettant en place des règles pour les contributions, les tests, et la publication des versions. C'est un investissement important en temps de la part des porteurs du projet, qui peut être compliqué à maintenir dans la durée, même si cela assure de la visibilité à l'équipe (et à son institution), et des retours (projets de recherche, thèses industrielles, etc.).
- Production continue de « logiciels métiers »[2] : Certains logiciels évoluent pour répondre aux besoins spécifiques d'industries ou d'institutions. Ils se transforment alors en logiciels applicatifs utilisés dans des contextes professionnels. À ce stade, le projet peut être géré par lesdits acteurs socio-économiques, en se séparant de l'équipe de recherche initiale du projet logiciel car il est devenu un produit métier avec des objectifs plus commerciaux que scientifiques.

Notre étude a identifié quatre « idéaux-types » de logiciels, chacun avec ses propres enjeux et perspectives de soutenabilité dans le temps.

1. **Les logiciels « en sommeil » :** Ces logiciels, développés dans le cadre d'un projet, n'ont plus de valeur stratégique pour l'équipe de recherche (et donc l'institution). La publication sous licence libre, par exemple sur une plateforme comme Software Heritage, est encouragée, car rendre le logiciel public peut générer des collaborations non anticipées ou renforcer la visibilité des recherches internes à l'institution.

2. **Les logiciels sur commande** : Ces logiciels, souvent produits pour des partenaires externes, sont régis par des contrats de recherche partenariale ou de prestation. L'institution s'assure du transfert auprès du partenaire privé, en définissant clairement les DPI et en déposant et archivant les versions du logiciel. La licence peut être libre ou privative selon le partenaire.

3. **Les logiciels de collaboration** : Ces logiciels, développés par l'institution de recherche (*via* une équipe ou un laboratoire) en collaboration avec un ou des partenaires publics ou privés, nécessitent un contrôle strict des DPI par l'institution. La diffusion sous licence libre, notamment *copyleft*, permet de garantir l'accès aux contributions tout en suscitant des retours et des améliorations. Une approche de double licence (libre et propriétaire) est souvent envisagée pour optimiser les opportunités de dynamique d'innovation, à condition que l'institution ait sécurisé les droits sur l'intégralité du code, notamment par le biais de contrats de cession avec les contributeurs externes.

4. **Les logiciels stratégiques** : Ces logiciels jouent un rôle central dans l'activité scientifique de l'institution, constituant des indicateurs clés de cette dernière. Ils contribuent à renforcer l'image de l'institution et favorisent les collaborations scientifiques, voire industrielles. Il est essentiel de maintenir un contrôle total des DPI et de choisir des licences permettant d'élargir la communauté d'utilisateurs de

[2] Voir lexique

favoriser l’externalisation partielle de la R&D et d’attirer de nouveaux partenaires scientifiques ou industriels. La maintenance et l'évolution de ces logiciels peuvent être assurées en interne ou par une startup issue du laboratoire.

La diffusion et la pérennité des logiciels de recherche repose sur des fondements juridiques et techniques impliquant des choix de licences adaptés à chaque logiciel et à son évolution. L'utilisation de modèles de licences libres, en particulier *copyleft*, parfois en complément d’autres stratégies de diffusion et pérennité, apparaît comme un levier important pour optimiser la diffusion et la valorisation scientifique, mais aussi industrielle. Ces stratégies reposent toutes sur une analyse des objectifs et des actions concrètes pour s’assurer un contrôle des DPI du projet logiciel.

Finalement, notre enquête souligne que les stratégies de diffusion et de pérennité reposent aussi sur des ressources humaines et des outils nécessaires à la production et la gestion des logiciels de recherche.

La maintenance des logiciels, en particulier ceux diffusés sous une licence libre, requiert un investissement important en temps et en ressources humaines. Cette tâche devient d'autant plus lourde lorsque le logiciel attire de nombreuses contributions externes, parfois au point d'exiger un personnel dédié à temps plein. Idéalement, il semble que la maintenance doive être assurée par le laboratoire à l'origine du logiciel, même si on peut imaginer faire appel à des personnels spécialisés en développement logiciel partagés entre plusieurs laboratoires (lorsque le langage utilisé est maîtrisé largement). Ces spécialistes garantissent la stabilité et la pérennité des logiciels et jouent un rôle crucial dans la sensibilisation des scientifiques à la diffusion et à la pérennité des développements logiciels.

Pour optimiser cette maintenance, il semble nécessaire de mutualiser les outils et les ressources au niveau national, facilitant ainsi la pérennité des logiciels. Des plateformes génériques de développement logiciel existent déjà dans plusieurs institutions françaises, mais de nouvelles fonctionnalités semblent attendues, telles que des outils de publication web facilitant la communication avec la communauté scientifique mondiale ou encore l’intégration continue pour les contributions aux logiciels. L’harmonisation des outils pourrait aussi faciliter l’identification des nouvelles créations au sein des laboratoires, notamment en intégrant des déclarations de création/logicielle (analogues aux déclarations d’invention) dès qu'un projet de développement est initié et en déployant les développements sur les forges nationales.

Le logiciel constitue une des productions scientifiques. Pour une diffusion et une pérennité efficaces, notre enquête révèle la nécessité d'intégrer tous les types de productions, publications, données, y compris les supports de formation, la documentation, etc., dans un système unique et accessible. Cela nécessite une complémentarité des outils utilisés par les contributeurs et utilisateurs de ce système, afin de favoriser le travail collaboratif.

Actuellement, il n’existe pas de cadre précis pour le choix des licences, qu’elles soient libres ou propriétaires. La variété des licences libres disponibles peut complexifier l’exploitation des logiciels, soulignant l’importance de définir une liste restreinte de licences compatibles avec les enjeux de la recherche.

Dès le début du cycle de vie d’un logiciel, il apparaît essentiel de planifier les stratégies d’exploitation du code pour anticiper ses évolutions, ses modes de diffusion, et les différentes perspectives de promotion afin d'assurer sa soutenabilité dans le temps.

En synthèse, les retours d'expérience suggèrent de travailler sur trois axes principaux : définir des outils de diffusion et pérennité (méthodes, techniques, plateformes, etc.) dans les pratiques existantes, en mutualisant les développements autant que faire se peut, former à l’usage de ces outils (propriété intellectuelle et modèles de projets pérennes de toute nature), et appliquer ces outils dans les politiques de propriété intellectuelle des institutions. Il a été aussi souligné que cela doit passer par la reconnaissance dans les carrières des scientifiques du travail de gestion et d’animation des projets logiciels.

Sommaire

Quelles stratégies de diffusion et pérennité pour les logiciels de recherche dans les établissements publics français ?

1 | Contexte

Dans les milieux scientifiques et académiques, le mouvement de l'accès ouvert aux publications scientifiques, puis de science ouverte s'est largement déployé. Il représente aujourd'hui « différents mouvements et pratiques » (UNESCO, 2021) mis en œuvre à l'échelle nationale (Plan National pour la Science Ouverte – PNSO), européenne (programmes Horizon 2020/Europe) et internationale (Recommandations sur la science ouverte, UNESCO) visant à « [la diffusion] sans entrave des résultats, des méthodes et des produits de la recherche scientifique »[2].

La place et le rôle des codes sources et des logiciels en science ouverte ont pendant longtemps été peu visibles. Néanmoins, depuis 2018, plusieurs initiatives autour de leur diffusion et de leur maintenance se sont structurées. Elles ont d'abord souligné l'importance du code source et du logiciel en tant qu'outils de la recherche, c'est-à-dire des ressources essentielles de la reproductibilité des travaux scientifiques et infrastructures de recherche durable (Bilder, Lin et Neylon, 2015). En France, la note d'opportunité du groupe d'expertise « logiciels libres et ouverts » a élargi cette perspective, en rappelant que ces logiciels sont également des résultats et des objets de recherche. Elle a également pointé l'importance de définir des stratégies, des outils et des procédures adaptés aux différents enjeux soulevés par les logiciels de la recherche et leur valorisation (Clément-Fontaine *et al.*, 2019). Dès 2021, l'axe 3 du deuxième PNSO promeut d'« ouvrir et promouvoir les codes sources produits par la recherche ». Pour cela, le collège « codes sources et logiciels », a été créé en 2022 au sein du comité pour la science ouverte (CoSO), pour accompagner cette dynamique d'ouverture et de promotion des codes sources et logiciels.

L'affinité pour les logiciels ouverts au sein des milieux académiques est consubstantielle de la naissance de ce mouvement – dont les pionniers provenaient des milieux de la recherche. On y retrouve les mêmes besoins de répondre aux limites technico-économiques de la production privée et fermée de logiciels, à savoir vérifier ce qui est produit et s'en inspirer, pour faciliter le travail collaboratif, renforcer l'amélioration par la revue par les pairs, mais aussi permettre des stratégies variées de valorisation (Jullien et Viseur, 2021). Une enquête entre 2023-2024 auprès des laboratoires français de recherche publique (MESR, 2023-2024) révèle que 69 % des logiciels issus de la recherche sont diffusés sous licence libre, 10 % sous une licence privative, 3 % sans licence et le reste des répondants n'a pas précisé ou ne connait pas la licence utilisée. Notre rapport, tout comme l'analyse du MESR (*ibid.*), conclut que la diffusion sous licence libre est compatible avec les différentes formes de valorisation économique. D'autre part, l'analyse du MESR précise que l'impact de la production logicielle de la recherche française est visible dans le milieu académique français et international, mais aussi dans la sphère socio-économique en étant notamment créatrice d'emplois. Elle souligne également « la diversité des stratégies de valorisation, qui permettent d'envisager des modèles économiques alternatifs. Par exemple, les modèles hybrides allient licences libres et propriétaires pour répondre à des besoins variés tout en permettant le transfert vers des entreprises. »

[22] Extrait de la définition de la science ouverte sur le site du Comité pour la science ouverte. (Pour une analyse des différentes définitions de la science ouverte et des courants associés, voir Gruson-Daniel 2023).

Néanmoins, plusieurs facteurs – comme le manque de gouvernance explicite ou de financement spécifique – limitent le potentiel de diffusion et de pérennité. On note parfois des freins culturels en fonction des disciplines, un déficit de compétences et de ressources, etc. (Gomez-Diaz et Recio, 2024). Le manque d'incitation des politiques institutionnelles, et de reconnaissance dans les carrières et les évaluations des professionnels de la recherche (Gruson Daniel et Jean, 2021), et une faible visibilité des politiques de propriété intellectuelle des établissements, voire une absence de politique de propriété intellectuelle dédiée aux logiciels (MESR, 2023-2024 ; Clément-Fontaine, 2024) semblent freiner les pratiques.

Les établissements publics peuvent avoir différents objectifs de pérennité des logiciels, selon leurs statuts et les missions qui leur sont assignées :

- social (développement des usages ou de certains usages, contribution au patrimoine logiciel mondial) ;
- production logicielle en appui aux politiques publiques ;
- financier (contrats d'exploitation – licence ou cession contre rémunération) ;
- économique (création d'entreprises ou soutien aux entreprises françaises) ;
- scientifique (publications, colloques…) ;
- notoriété (connaissance de l'institution parmi nos publics-cibles : étudiants, chercheurs, entreprises, prescripteurs).

Ces objectifs, bien que communs à tous, ont des poids qui varient selon les établissements, les missions, et aussi selon les types de logiciels produits.

Souvent, les choix de diffusion (et les licences l'encadrant) se font directement dans les équipes de production sans demande de validation / expertise auprès des services de relation partenariale et valorisation de l'établissement, et ce, notamment pour pouvoir publier rapidement un article scientifique et le code logiciel associé. Il existe plusieurs raisons à cet état de fait :

- la peur de se voir bloqué ou ralenti dans son projet ;
- la méconnaissance des différentes options de diffusion et promotion (autres que par le brevet) ;
- la méconnaissance de l'importance de la protection des résultats.

Pour les services de relation partenariale et valorisation, la diffusion d'un logiciel sous une licence libre nécessite une analyse préalable permettant d'anticiper d'autres modalités de pérennité pour les versions ultérieures (autorisation d'accès contre rémunération, protection d'un résultat de recherche associé par brevet d'invention pour les inventions techniques qui contiennent des fonctionnalités du logiciel, transfert de savoir-faire, etc.). La science ouverte représente une stratégie « audacieuse »[3] pour les personnels des services de relation partenariale et valorisation, qui n'ont pas encore tous les outils pour la gérer ; ce besoin de montée en compétences a été pointé par le deuxième PNSO. Plusieurs services de relation partenariale et valorisation, comme récemment ceux d'INRAE (Gadet et Brunner, 2023) ou du CNRS (programme Open lancé en 2023 par CNRS Innovation) mènent des réflexions et expérimentations de valorisation de logiciels libres. Loin de nous l'idée que ce sont les seuls. D'autres institutions pionnières dans cette démarche ont développé des réflexions, voire des recommandations en ce sens. Nous citerons, et sans exhaustivité, ces établissements qui ont déployé des politiques de propriété intellectuelle :

Inria a une politique de propriété intellectuelle qui propose une modalité de diffusion adaptée et évolutive, de façon à ne pas compromettre l'impact économique et en retardant les choix de licence sur lesquels il est difficile de revenir en arrière ;

L'IRD préconise dans sa charte de propriété intellectuelle une approche pragmatique pour le logiciel de recherche, « aussi ouvert que possible, aussi fermé que nécessaire », en utilisant notamment les possibilités de doubles licences pour une valorisation vers les partenaires privés pour les logiciels ayant un possible impact financier ;

L'UGA recommande dans sa charte et son schéma directeur pour la science ouverte, que les codes soient diffusés en libre accès et préconise l'utilisation de forges logicielles pour le développement. Elle prescrit également l'archivage des codes de recherche dans Software Heritage en lien avec une notice HAL. Elle s'engage pour cela à soutenir les équipes d'accompagnement (cellule Data Grenoble Alpes, atelier de la donnée de l'université) et les infrastructures nécessaires.

[3] Lors de l'école d'hiver de la valorisation 2024 du réseau C.U.R.I.E., la science ouverte a été abordée. La question posée était notamment : « La science ouverte fait-elle partie de la révolution du métier ? »

Pour contribuer à ces réflexions, le présent document propose une analyse de la diffusion et de la pérennité des travaux scientifiques de type logiciel produits par les établissements de recherche publique en France. Cette analyse prend appui sur un travail d'enquête (6 projets étudiés en détail, avec 13 entretiens réalisés). Par « production d'un logiciel », nous entendons le processus conduisant à la création d'un objet qui peut aller au-delà d'un code à usage unique qui appuie un travail scientifique, mais n'avait pas *a priori* vocation à être réutilisé dans une communauté plus large, et intègre au fil du temps une accumulation d'ajouts externes qui doivent être coordonnés dans son architecture. Dans les parties suivantes du rapport, après avoir présenté un état des lieux de l'écosystème actuel des logiciels de recherche dans la partie 2, notamment les types de logiciels produits, nous faisons une synthèse, informée par les résultats de l'enquête, des perspectives de diffusion et promotion des logiciels de recherche et des investissements nécessaires pour ce faire (partie 3). Des exemples des projets logiciels qui nous ont permis de construire ces analyses sont proposés en annexe.

2 | Le logiciel de recherche : état des lieux

Les logiciels produits par les établissements n'ont pas tous la même capacité/vocation à exister dans le temps : on parlera de durabilité des logiciels ou bien de leur soutenabilité dans le temps. Pour analyser la pérennité et la durabilité des logiciels de recherche, intéressons-nous dans un premier temps aux catégories de production logicielle.

Cela dit, qu'elles aient été « commandées », créées par effet de bord ou pour mener une activité de recherche courante, il convient de s'interroger sur la façon dont ces créations particulières doivent être gérées, et de proposer des stratégies adaptées aux métiers de nos établissements, aux types de logiciels développés, etc., tout comme aux ressources mobilisables. Il faut inscrire ces stratégies dans les méthodes standards de travail des agents publics, scientifiques, personnels chargés de partenariat, transfert, innovation et valorisation, responsables des carrières, afin qu'elles soient mises en œuvre de façon appropriée au cours des cycles de vie des logiciels développés et en intégrant les objectifs de chaque partie prenante.

Logiciel, de quoi parle-t-on ?

Un logiciel est le fruit d'une production intellectuelle qui formalise, dans un ou plusieurs langages informatiques, un ensemble de règles décrivant les tâches de traitement de l'information que l'on souhaite faire réaliser par un ordinateur. Selon le *Livret codes et logiciels, de la collection « passeport pour la science ouverte »* (Pellegrini *et al.*, 2022), « on parle de logiciel lorsque l'ensemble a suffisamment d'utilité par lui-même pour être considéré comme ayant une existence propre en termes de préservation, d'évolution et de diffusion », incluant le transfert. « Dans ce cas, le logiciel peut exister en plusieurs versions, se succédant dans le temps ou coexistant en parallèle, au gré des adaptations ».

Les logiciels de recherche ont été définis par le collège « Codes sources et logiciels » du comité pour la science ouverte du ministère de l'enseignement supérieur et de la recherche comme suit : « Les logiciels de recherche sont développés pour répondre à des besoins spécifiques de la science. Ils sont conçus, maintenus, et utilisés par des scientifiques (chercheurs et ingénieurs) et institutions de recherche, éventuellement dans une dimension internationale. Ils peuvent découler de travaux de recherche comme ils peuvent les favoriser, notamment par des publications avant/sur/autour/avec le logiciel. Ceux-ci peuvent se formaliser de différentes façons (une plateforme, un intergiciel, un *workflow* ou une bibliothèque, module ou greffon d'un autre logiciel) et être ainsi en interaction dans un écosystème ou au contraire plus autonomes. »[4]

Le logiciel a les caractéristiques d'un « bien public » (au sens économique du terme), c'est-à-dire d'un bien qui, une fois produit, est difficilement rival. Il peut être utilisé par plusieurs personnes, en même temps, sans que l'utilité de son utilisation ne soit diminuée. De plus, sans certains ajouts techniques ou barrières légales, il est non exclusif, puisque tout fragment de code sous forme numérique peut être copié et transféré à un coût presque nul. C'est un parangon de l'économie de la connaissance (Foray, 2004), où les éléments immatériels produits intellectuellement sont à la base de la création de valeur.

La production d'un bien public rencontre un problème de financement, qui est appelé, en économie, le problème du « passager clandestin » : si tous les acteurs peuvent bénéficier du bien sans participer au coût de production, alors, il semble qu'aucun ne soit incité à le produire, chacun s'attendant à ce que d'autres s'en chargent.

En théorie, il existe trois possibilités pour produire ce type de bien :

[4] Voir l'analyse de cette définition dans Pellegrini (2024). F. Pellegrini, « Qu'est-ce qu'un logiciel de recherche ? », Revue Lamy Droit de l'immatériel, (2024).

Il peut être produit par un seul ou plusieurs personnels, pour répondre à un besoin propre à l'équipe ; c'est certes un bien public, mais qui est développé indépendamment de son utilité en dehors de celle de l'équipe et non mis à disposition. Ce type de production est récurrent dans l'histoire de l'informatique. Aujourd'hui encore, ce phénomène se rencontre comme lorsque des plateformes en ligne conçoivent des services dédiés, protégés par des bases de données et des algorithmes secrets pour gérer leurs utilisateurs. C'est aussi souvent l'origine des logiciels scientifiques, produits par un ou des scientifiques pour tester une hypothèse, ou proposer une solution à une question scientifique, informatique ou non (nous y reviendrons). Cependant, si les producteurs le gardent pour eux seuls, ils doivent seuls subir les coûts de production, et ne sont que rarement reconnus pour avoir produit ce logiciel, surtout lorsqu'un concurrent plus ouvert finit par entrer sur le marché et par acquérir de la notoriété. Des scientifiques peuvent préférer cette solution de garder des développements en interne au laboratoire. Cela minimise leurs coûts de maintenance et ne remet pas en cause leurs habitudes de développement. Cependant, cela ne participe pas à l'effort de science ouverte et, du point de vue du laboratoire, plus le temps passe, plus la dette technique va augmenter et donc le coût du changement si à un moment une solution concurrente s'impose[5] ;

Il peut également être produit par un établissement public et, à la différence du point 1, mis à la disposition de tous, comme d'autres biens publics. C'était essentiellement le cas au début de l'histoire de l'informatique, lorsque la production de logiciels était une activité risquée, orientée vers la recherche, ainsi qu'une activité de conception de prototypes (programme développé pour un seul utilisateur). Même si cela existe encore aujourd'hui, le soutien de l'État à l'ensemble des développements de logiciels peut difficilement être maintenu une fois l'informatique devenue une industrie qui s'appuie sur la production de logiciels plus ou moins standards, et ce, pour des raisons évidentes de coût et de capacité de l'État à comprendre l'ensemble des besoins et à y répondre.

Il peut être produit par un acteur, public ou privé, qui supporte le coût de production en anticipant le fait que des tiers seront prêts à payer pour accéder à ces développements (et donc les co-financer). Cela n'est possible que si l'acteur privé a la garantie de pouvoir vendre sa production sans affronter la concurrence de tiers à qui l'usage a été concédé et qui en auraient fait une copie. C'est aussi vrai pour les logiciels financés par l'État. Par exemple, les logiciels de recherche répondent à ses besoins propres (celui des institutions de recherche ou de ses personnels, notamment chercheurs), mais ils peuvent aussi répondre à des besoins d'acteurs privés à qui l'État peut vouloir demander une participation à la production, à l'exploitation ou à la maintenance du logiciel. De manière générale, un acteur qui développe un logiciel peut avoir besoin de contrôler les usages et les modifications qui en sont faits, pour s'assurer qu'il correspond toujours à ses besoins.

Faire d'un logiciel un bien échangeable nécessite, comme pour les autres biens publics et, parmi eux, les biens produits intellectuellement, de créer les conditions techniques, en particulier juridiques, par le biais des DPI, permettant de déterminer la titularité des droits, les modes d'exploitation et de protection[6].

Une fois la titularité des DPI déterminée, les titulaires des droits peuvent contrôler l'accès à la production du logiciel et en fixer les contreparties, selon la stratégie de pérennité identifiée, en définissant les conditions qui permettent de rémunérer la production. Ces conditions peuvent être financières ou de toute autre nature, telles que des échanges d'informations ou des citations (notamment au sein de la communauté scientifique). Le titulaire des DPI aura recours à un contrat, qui sera le plus souvent appelé une « licence d'utilisation de logiciel ».

Si cette solution a permis le développement de la production logicielle, elle soulève un certain nombre d'enjeux.

Les enjeux en droit de la propriété intellectuelle appliqué aux logiciels

Le droit appliqué au logiciel est une adaptation du droit d'auteur, que nous ne détaillerons pas ici (nous proposons en annexe des liens vers des sites de référence). L'application de ces DPI, même adaptés, soulève quelques difficultés.

[5] Les auteurs du rapport remercient Célya Gruson Daniel pour avoir souligné ce cas de figure. -> à mettre en remerciements généraux

[6] Voir cette question à propos des systèmes d'intelligence artificielle en accès libre : Clément-Fontaine (2024)

Tout d'abord, le droit d'auteur ne protège pas une idée, mais l'expression particulière de cette idée. En d'autres termes, les fonctionnalités du logiciel ne sont pas protégées. De plus, les principes sous-jacents sur lesquels le code est basé ne sont pas protégés de manière sûre (il est difficile de prouver qu'un autre producteur n'a pas repris une partie du code pour ses développements propres). En conséquence, les producteurs privés de logiciels commercialisent le plus souvent leurs produits uniquement sous forme de programmes exécutables, c'est-à-dire de « code objet », en cachant le « code source » des programmes, c'est-à-dire l'expression explicite de leur architecture, de leurs procédures et de leurs algorithmes. Ce n'est évidemment pas satisfaisant pour la recherche, puisque cela nuit à la reproductibilité et à la transparence des résultats, et empêche de s'appuyer sur le travail précédemment réalisé pour développer de nouvelles recherches.

Ensuite, les producteurs d'une ressource logicielle doivent faire face à des objectifs contradictoires (Cusumano, 2004 ; Horn, 2004) : le logiciel doit être aussi adaptable et « personnalisable » que possible pour répondre à différents besoins à court terme, mais aussi résilient que possible, pour répondre aux besoins sur le long terme. Cela signifie que la solution doit évoluer en permanence pour faire face à l'évolution des technologies (matériels, logiciels tiers, formats de fichiers) avec lesquelles elle interagit, tout en restant stable sur le long terme pour permettre un fonctionnement non altéré. Or, les titulaires des DPI peuvent conditionner l'accès à la ressource à une durée limitée (conditions qui sont décrites dans le « contrat » de licence), et peuvent empêcher les utilisateurs de faire évoluer la ressource (interdiction juridique, ou bien barrière technique lorsque seul le code objet est diffusé). L'évolution du logiciel à travers des versions successives reste entièrement dépendante des capacités et des souhaits du titulaire des droits, ce qui signifie que toute intervention sur le produit de la part de l'utilisateur ou d'un autre développeur est empêchée soit par le droit soit par les clauses contractuelles.

Au-delà de l'impact sur l'innovation, cette pratique a une conséquence spécifique en raison de l'« interrelation » ou l'« adhérence » technologique (un logiciel fonctionne rarement seul, mais en interaction avec d'autres composants, bibliothèques, systèmes d'exploitation, etc.), car sans comprendre comment un programme permet à d'autres programmes de s'interfacer avec lui, il est très difficile de construire un système d'information efficace.

En d'autres termes, en permettant à un acteur de contrôler l'accès à un code source qu'il a développé (ce que nous appellerons le « stock du logiciel »), le régime des DPI lui a aussi donné le contrôle sur les évolutions futures de ce logiciel ainsi que sur la dynamique d'innovation qui peut se développer autour de lui (ce que nous appellerons le « développement continu du logiciel », pour reprendre le terme de Fitzgerald et Stol, 2017).

Le mouvement des logiciels libres est apparu au début des années 1980 en réponse aux restrictions présentées ci-dessus. Ce mouvement, structuré en 1984, a permis la naissance de stratégies de pérennité par la diffusion sous une licence libre d'un logiciel, notamment dans la recherche (on pourra notamment lire le manifeste du projet GNU[7], de Richard Stallman). Pour protéger l'ouverture du code et le droit de modification, R. Stallman et E. Moglen ont créé une licence spécifique, la GNU GPL (General Public Licence), qui accorde des droits, en contrepartie de certaines obligations. En particulier, la redistribution d'un programme diffusé sous licence GPL, avec ou sans modifications apportées, doit se faire selon les mêmes termes et conditions que la licence initiale. L'objectif est de pérenniser le libre accès au logiciel en empêchant toute réservation exclusive, comme cela peut être le cas pour les programmes distribués sous une licence permissive telle que les licences BSD ou MIT. Pour disposer des droits de diffuser sous les conditions de la GPL ses productions, R. Stallman a démissionné du MIT et créé une fondation – la Free Software Foundation, ou FSF – avec pour objectif de développer un système d'exploitation libre. En effet, s'il était resté employé du MIT, ce dernier aurait été l'ayant droit de sa production.

Ainsi le mouvement du logiciel libre repose depuis son origine sur une connaissance fine des droits de propriété intellectuelle. La licence GPL, comme toute autre licence libre, ne représente pas un déni de la propriété intellectuelle, mais au contraire une nouvelle façon d'exercer les droits de propriété intellectuelle, puisqu'elle tire sa force légale de l'existence du droit d'auteur. Les ayants droit ne renoncent pas à leurs droits de propriété intellectuelle mais, bien au contraire, les exercent afin de distribuer et de garder ouverte leur production intellectuelle (Clément-Fontaine, 2006). En revanche, ils renoncent à la rente de monopole sur les frais d'accès ainsi qu'au contrôle strict de l'évolution du logiciel. Cette manière originale de gérer la propriété intellectuelle a été appelée par ses initiateurs l'attitude « *copyleft* » (en référence au « copyright », et traduit parfois par « gauche d'auteur »).

[7] http://www.gnu.org/gnu/thegnuproject.html

Autrement dit, la publication sous licence libre d'un logiciel est un choix, parmi d'autres, un outil au service de la stratégie de diffusion et pérennité du titulaire des droits de propriété intellectuelle sur le logiciel. C'est une stratégie qui consiste à renoncer à une partie du contrôle sur le patrimoine **logiciel existant afin de dynamiser le développement continu et l'innovation.**

On peut comprendre pourquoi la recherche, dont l'un des objets est de favoriser l'innovation, peut être sensible à cette stratégie de diffusion et pérennité du flux d'innovation et des contributions logicielles ainsi générés. L'analyse de plusieurs projets de développement de logiciels de recherche nous a permis de mieux cerner différentes étapes du cycle de vie d'un projet logiciel – qui varient en fonction des objectifs de promotion spécifiques – et de déterminer les ressources qui doivent être mobilisées pour atteindre ces objectifs de promotion.

Dans la partie suivante, nous présentons le cycle de vie du logiciel de recherche en nous appuyant sur deux types d'approche :

- l'approche de la TaskForce on infrastructures for quality research software de l'European Open Science Cloud (EOSC) ;
- l'approche de développement continu de logiciel (Fitzgerald et Stol, 2017).

Le cycle de vie d'un projet logiciel de recherche

Les projets logiciels issus de l'activité de recherche scientifique peuvent s'articuler en quatre catégories qui se suivent dans le temps, mais qui ne concernent pas tous les projets : (i) la preuve de concept ; (ii) l'initiation d'une collaboration scientifique basée sur le logiciel ; (iii) la collaboration large sur la gestion d'un projet de développement continu de logiciel ; et (iv) la gestion de la production d'un développement continu de logiciel métier.

Le logiciel comme preuve de concept

Les logiciels naissent d'une activité de recherche spécifique portée par un scientifique ou une équipe – par exemple dans le cadre d'un projet de recherche financé sur appel à projet (ANR, Europe, etc.)-, d'un programme scientifique, ou en collaboration/prestation de recherche avec un tiers. C'est le cas de tous les projets que nous avons étudiés. On pourra, pour s'en convaincre, relire l'origine des projets CGAL et CMCDOT.

À l'origine, cette catégorie de logiciels n'est *a priori* pas conçue pour une diffusion ou un transfert ultérieur en tant que tel, mais plutôt comme un outil pour mener l'activité de recherche du scientifique ou de l'équipe de recherche, ou comme preuve d'existence d'une réponse à une question précise, conduisant à l'obtention de résultats scientifiques. L'évaluation des codes sources, la réutilisabilité du logiciel, ou son évolution future ne sont pas prioritaires. Toutefois, les contraintes de publication des résultats scientifiques peuvent nécessiter la diffusion du logiciel (codes sources joints à la publication, souvent à la demande de l'éditeur de l'article, notamment pour confirmer la validité des résultats présentés).

Si les résultats sont de qualité suffisante, le logiciel lui-même peut être publié dans une revue scientifique, sous une forme étendue (les concepts sont approfondis, développés dans des preuves de concept supplémentaires) et consolidée (la technologie est améliorée, le logiciel est rendu plus robuste pour être réellement utilisé). Ces « *software papers* » sont une nouvelle forme de publication qui vise à décrire un logiciel qui a été développé pour la recherche, afin de le présenter à la communauté scientifique. On peut citer par exemple le logiciel Scikit-Learn qui fait l'objet de *software papers* et la revue Journal of Machine Learning Research qui publie ce type d'articles, à condition que les logiciels soient publiés sous une licence libre[8]. Cette première phase de développement va s'enrichir dans le temps, notamment grâce aux retours des pairs et aux extensions d'usage ou de recherche qui se dessinent en perspective. Le projet logiciel continue, et il est alors possible que son développement s'ouvre, par le biais d'une collaboration avec d'autres acteurs que les développeurs initiaux.

On aura alors parcouru ce que Courbaisse *et al.* (2023) ont appelé un « cycle » de logiciel de recherche (voir Figure 1).

[8] voir les conditions de publication ici : https://www.jmlr.org/mloss/mloss-info.html

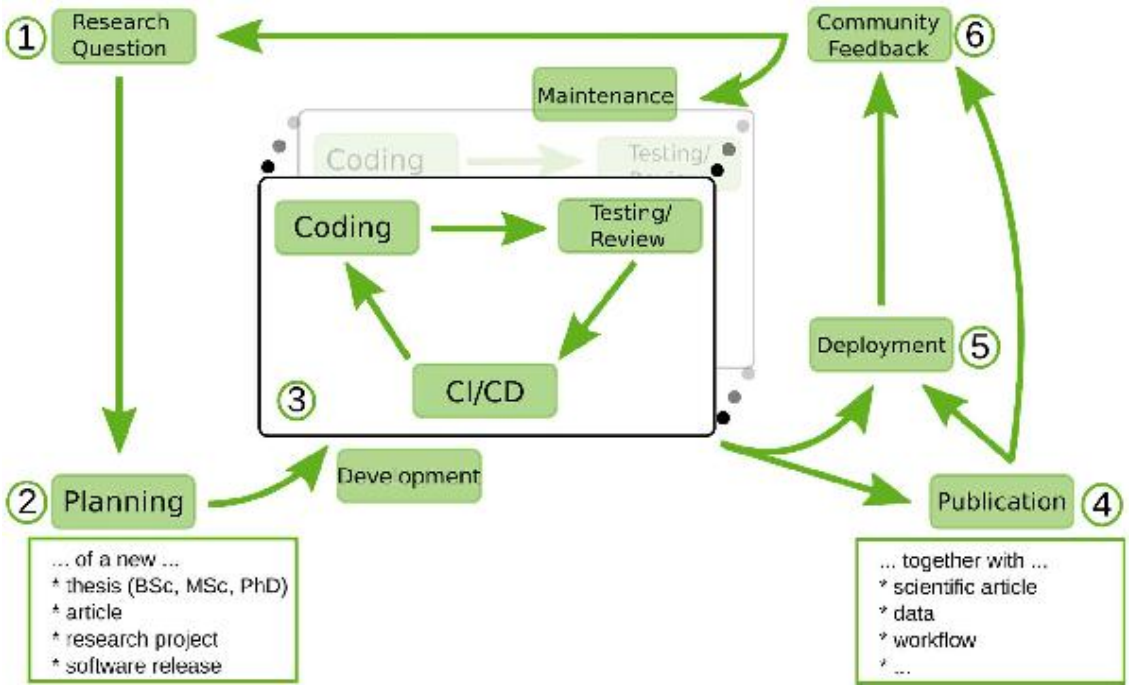

Figure 1: Cycle d'un logiciel de recherche, Figure reprise de (Courbaisse et al., 2023)

Si les résultats ne sont pas intéressants ou ne nécessitent pas d'aller plus loin, les développements logiciels vont être abandonnés. C'est le cas lorsque le projet de recherche s'arrête : la publication a été effectuée, le contrat honoré, ou aucune perspective ne s'ouvre pour l'équipe de recherche. Le code produit est alors en sommeil, en ce sens qu'il peut sembler abandonné par ses créateurs, qui ne souhaitent pas s'investir dans le maintien et l'aide à l'évolution du logiciel existant. À ce stade, la promotion du logiciel sera essentiellement une promotion en terme d'image (cf. le paragraphe **« logiciel en sommeil », partie 3.a) 1**).

Initiation d'une collaboration

Si les contributions restent internes à l'équipe, voire à l'institution, la gestion des échanges peut rester informelle. Cependant, on sort de la simple preuve de concept, car le projet fédère une équipe, et peut même devenir central pour le laboratoire. Il s'agit d'un **projet de « collaboration » interne, ou développement « *inner-source* »** (Stol et Fitzgerald, 2014). On prendra soin de le développer sur une forge reconnue (Le Berre *et al*., 2023) et de référencer les codes sources et archiver les versions pour le suivi du cycle de vie du logiciel et de ses DPI.

Si les contributions, et surtout les demandes, sont issues de partenaires dans le cadre d'un projet contractuel, et qu'elles dépassent les demandes initiales, on peut se poser la question du financement de leur développement. Éventuellement, d'autres contrats partenariaux pourront être négociés par les services de valorisation et transfert, pour aller au-delà de la preuve de concept : il s'agit de projets de développement de **logiciels de « commande »** (*v.* le paragraphe consacré, **partie 3.a) 1**), si le ou les partenaires sont des partenaires privés qui sont « clients » du projet. Pour autant, dans tous les cas, il s'agit de refaire un autre cycle de développement, qui reste maîtrisé par l'équipe de recherche et qui ressemble toujours à celui décrit par la figure 1, le projet GPAC en est un exemple.

Certains logiciels diffusés peuvent être repris par d'autres équipes ou individus, qui peuvent effectuer des contributions mineures (de type *« rapport de bogue »*), ou proposer des évolutions fonctionnelles (spécifications ou apports en code). Si c'est le cas, l'équipe initiale doit s'organiser pour gérer ces demandes et faire des retours, si elle le souhaite, à ces utilisateurs et contributeurs. Le logiciel peut aussi être soumis à une plateforme de diffusion spécifique (par exemple, une bibliothèque proposée au téléchargement dans un dépôt de type CRAN ou bioconductor pour R, ou Pypi pour Python), pour être intégré à un projet logiciel plus vaste. Dans les deux cas, **il s'agit alors d'un projet logiciel de « collaboration »** (partie 3) pour lequel il convient d'évaluer les coûts de gestion au regard des potentiels

gains de l'inclusion de collaborateurs externes (citation du travail, contribution en code ou en idée de fonctionnalité qui renforcent l'intérêt du projet, etc.). Par rapport aux cycles initiaux, il s'agit d'intégrer des retours externes, y compris en code directement dans la partie développement (Figure 1, étape 3).

L'enjeu est celui rencontré par les promoteurs des projets libres, à savoir : de stimuler l'innovation et la production de créations nouvelles. Le principe de l'utilisateur-innovateur (Lakhani et von Hippel, 2003 ; von Hippel et von Krogh, 2003), qui est aussi celui de la recherche (Cohen et Livental, 1990), est le fondement du mouvement de la science ouverte. Ainsi le ou la productrice de la recherche (ici formalisée au sein du logiciel) est incitée à la produire, en tant que première bénéficiaire directe. Elle est aussi incitée à publier cette recherche pour obtenir des retours d'information sur sa proposition, voire des innovations complémentaires et cumulatives. Ce faisant, elle bénéficie également d'effets d'apprentissage individuel et renforce, par cette expérience, son propre niveau de compétence (Foray, Thoron et Zimmermann, 2007) : elle accélère ainsi le développement de sa propre recherche.

Plus le projet est important en termes de volume et de complexité du code, ou du nombre de participants, plus les contraintes techniques et organisationnelles sont fortes. Si les contributions restent sporadiques, une gestion informelle par l'équipe initiale peut perdurer. Sinon, il s'agit de construire et de gérer une « architecture de participation » (MacCormack *et al.*, 2006) dans une organisation de « développement continu du logiciel » (Fitzgerald et Stol, 2017), où toutes les tâches du cycle de vie sont entremêlées continuellement. Plusieurs équipes de différentes institutions peuvent, en parallèle et parfois en concurrence, organiser un cycle de développement (Figure 1), et proposer leur contribution en code au projet, avec toutes les contraintes que cela implique en termes de gestion de projet logiciel (arbitrage ou au minimum articulation entre les différentes propositions), comme expliqué dans le paragraphe suivant.

Production continue d'un logiciel de recherche

La multiplicité des contributeurs impose que les développements initiaux soient réorganisés vers la construction d'un projet logiciel plus structuré permettant une plus grande indépendance des travaux. Citons le logiciel Coq (renommé récemment The Rocq Prover), qui est un exemple de durabilité structurée depuis la fin des années 1990. The Rocq Prover a intégré régulièrement dans ses versions distribuées des améliorations technologiques renforçant ses performances. Il est ainsi devenu robuste pour servir de langage à des outils en preuve de programmes (voir sa fiche en annexe). Pour favoriser l'intégration de nouveaux contributeurs (Bessen, 2005) et optimiser la gestion des projets sur le plan du génie logiciel et de l'organisation (Baldwin et Clark, 2003 ; MacCormack *et al.*, 2006), il est essentiel d'organiser les projets en composants modulaires bien définis. En effet, les équipes réparties géographiquement ou comptant plus de cinq développeurs rencontrent souvent des difficultés à travailler efficacement (Mockus *et al.*, 2002). La formalisation et la structuration de règles de contribution, des outils d'évaluation et de tests du code, de décision, deviennent nécessaires conduisant à une forme de « bureaucratisation » de l'organisation (O'Mahony et Ferraro, 2007). Il s'agit en fait de mettre en place l'**édition du logiciel :** gestion technologique, gestion des demandes, validations techniques des modifications, publication des versions, etc. C'est un métier spécifique de l'informatique, qui n'est pas *a priori* celui des scientifiques à l'origine du code.

Les demandes des personnes utilisatrices peuvent diverger entre elles, mais aussi s'éloigner des intérêts de l'équipe initiale. Notamment, lors de la restructuration du code, le développement d'un logiciel plus générique peut l'éloigner des objectifs initiaux de recherche. Cette catégorie peut être nommée « logiciel générique de recherche ». Comme la précédente, elle s'observe dans le cadre de partenariats industriels, européens, académiques. Elle génère des contrats de prestation, mais aussi facilite l'accès à de nouveaux projets de recherche (logiciel vitrine d'un savoir-faire, plateforme technologique permettant de développer plus efficacement d'autres recherches et innovations).

Certains de ces projets logiciels sous licence libre s'organisent autour de consortia. Un exemple international de ce type est celui du Genone Analysis Toolkit (GATK), porté par le Broad Institute[9], diffusé sous licence libre Apache v2, et auquel tout un chacun peut apporter des contributions.

La gestion formelle du projet de développement, le travail d'édition et de documentation demandent des ressources humaines importantes. Si le projet reste stratégique pour l'institution, celle-ci devra mettre en place des moyens de financer tous ces aspects. Il s'agit d'une promotion dite d'un **projet logiciel « stratégique »**. Sinon, la séparation de l'équipe initiale (et de l'institution ayant droit du logiciel) se

[9] https://gatk.broadinstitute.org/hc/

produit souvent à ce moment-là. La gestion du projet stratégique est alors confiée à un tiers, chargé de poursuivre le travail d'édition du logiciel : soit une fondation, soit un éditeur privé (qui pourrait être une startup issue de l'équipe initiale créée dans pour cela), qui monétise les demandes d'évolution et/ou d'expertise pour financer les développeurs cœurs du projet et l'animation de la communauté. C'est le cas dans notre étude du projet CGAL (v. en annexe la fiche de synthèse).

Enfin, le projet scientifique initial peut devenir secondaire dans le projet logiciel : d'un projet de recherche, on passe à un projet de R&D, dans lequel domine la partie développement. Les logiciels génériques de recherche sont devenus des logiciels utiles à des applications métiers, évoluant en logiciels applicatifs spécifiques. Il s'agit alors d'organiser un projet de production continue de logiciel métier.

Production continue d'un logiciel métier

Cette dernière catégorie concerne des cas d'usage mobilisant des partenaires industriels ou assimilés (hôpitaux, par exemple), puis de développements réalisés par des intermédiaires (des entreprises, le plus souvent), qui fournissent des adaptations à des clients qui ne sont pas en mesure de les développer eux-mêmes. Les relations entre ces besoins marchands et l'équipe de recherche doivent être claires, notamment si les intermédiaires demandent ou proposent de nombreuses modifications, ou de nombreux conseils. Si la séparation du projet d'avec l'équipe de recherche n'a pas encore eu lieu, elle se produit à ce moment-là (v. paragraphe précédent sur la création d'un éditeur du projet). Les démarches sont formalisées par les services de relation partenariale et valorisation des établissements.

Finalement, un projet logiciel peut s'arrêter pour deux raisons principales :

- Il y a de moins en moins d'utilisateurs, car d'autres technologies sont apparues et d'autres outils ont été adoptés par la communauté scientifique ;
- Le logiciel est toujours utilisé, mais il ne bénéficie plus du développement de nouvelles fonctionnalités, ce qui réduit les besoins d'organiser les contributions. Les besoins économiques peuvent cependant persister, mais ce n'est plus un projet en développement continu : la gestion des contributions n'est plus centrale. Le rôle de l'éditeur diminue, tout comme les investissements liés à ce rôle. Le rôle de diffuseur peut cependant continuer à exister.

Ces différentes étapes peuvent se succéder dans le temps. C'est pourquoi il est nécessaire, dès le début de la vie d'un logiciel de recherche, de prévoir les différentes stratégies de protection, modalités de diffusion, et perspectives de pérennité.

Nous allons maintenant détailler ces perspectives, en continuant à nous appuyer sur les cas que nous avons étudiés.

3 |Les logiciels de recherche : perspectives de pérennité

À l'issue des entretiens menés auprès de plusieurs porteurs de projets logiciels et services de relation partenariale et valorisation (voir la liste des projets logiciels en annexe), nous proposons de distinguer quatre « idéaux types » de logiciels de recherche, allant des logiciels qui n'intéressent plus le laboratoire ou l'institution jusqu'aux logiciels les plus stratégiques pour eux. Pour chaque cas, nous décrivons des stratégies de diffusion et pérennité, de distribution et donc de contrats ou licences. Il apparaît clairement qu'à côté des licences classiques, les licences libres sont des outils importants pour défendre au mieux et à moindre coût les intérêts patrimoniaux de nos institutions.

C'est pourquoi, dans ce rapport, sont détaillés d'une part les ressources humaines et les outils nécessaires à la production et à la pérennité de logiciels de la recherche et, d'autre part, le cadre ainsi que la stratégie juridique à mettre en place.

Enfin, la dernière sous-partie détaille les actions qui doivent être développées *a minima* au niveau de l'établissement (par le gestionnaire ou établissement de tutelle des laboratoires, ou « GET ») dans l'intérêt des projets concernés, mais aussi dans l'optique d'aider nos institutions à construire une démarche volontariste et méthodique pour la diffusion et la pérennité des logiciels.

Les idéaux types de promotions des logiciels de recherche

Notre étude a identifié quatre idéaux types de logiciel, qui peuvent se rencontrer à différents moments du cycle de vie d'un logiciel. Ce dernier peut ainsi changer de catégorie au cours de son existence.

Présentation des quatre situations types de projet logiciel :

Les projets logiciels « en sommeil »

Ces logiciels ont été produits à un moment donné, dans le cadre d'un projet, d'une expérience, mais ne font pas partie des axes stratégiques de l'institution, du laboratoire ou de l'équipe qui les a produits. On n'identifie pas non plus de partenaire susceptible d'être intéressé par cette production.

Si un tel logiciel est basé sur la production d'un tiers, deux solutions sont possibles :

- Cette production est disponible sous licence libre : l'institution peut publier la contribution en respectant les conditions de la licence (copyleft ou non) ;
- Cette production est disponible sous une licence privative : il faudra obtenir une autorisation spéciale pour publier la contribution, ce qui peut être long et coûteux au regard de l'intérêt du logiciel.

Si l'institution de recherche est titulaire de l'ensemble des DPI sur le logiciel, ou que les autres composants sont déjà placés sous licence libre, il est intéressant de rendre accessibles ces logiciels, c'est-à-dire de les publier, sous une licence libre, et ce pour plusieurs raisons :

- Cela montre l'étendue de la production de l'institution ;
- Cela fait partie des publications des scientifiques impliqués, et des moyens pour les faire (re)connaître par leur communauté scientifique.

Le logiciel peut intéresser un utilisateur qui n'avait pas été identifié en amont, et permettre des collaborations futures.

Le logiciel peut alors être publié sur un site spécifique de l'institution, ou un site national identifiant l'institution. Notamment, un archivage sur Software Heritage est normalement fait si le logiciel a servi dans un article scientifique. Cependant, il est important de bien stipuler qu'il s'agit d'un logiciel actuellement non maintenu par l'institution, que les personnes sont libres d'utiliser, mais sans soutien ni engagement de responsabilité de l'institution (Le Berre *et al.*, 2023). La licence pourrait être de type *copyleft* (telle que la GPL) dans un premier temps, afin de se ménager la possibilité d'un éventuel contrat industriel portant sur une licence plus permissive dans un second temps (telle qu'une licence BSD), ou inversement une licence fermée.

On peut aussi imaginer concéder des licences d'utilisation, sans diffusion du code source. Cette modalité de distribution permet de gagner du temps pour établir une stratégie si des personnes ou des organisations se manifestent et expriment le désir de modifier le logiciel. Cela suppose cependant de disposer des ressources permettant de créer une version binaire compatible avec le matériel des utilisateurs, ce qui peut être très coûteux si les versions ou matériels se multiplient[10].

Les licences selon lesquelles seront diffusés les composants logiciels créés par l'institution, ainsi que de la licence couvrant l'œuvre dérivée, doivent être juridiquement compatibles avec les licences des composants préexistants incorporés (ce qui peut représenter un travail d'analyse important).

Les projets logiciels de commande

Ces logiciels sont produits dans le cadre d'un projet de recherche, mais ne font pas partie des axes scientifiques stratégiques du producteur ou de l'équipe. Cependant, il existe au moins un partenaire susceptible d'être intéressé par cette production (ce partenaire ayant éventuellement passé commande du logiciel).

Dans ce cas, on est dans le cadre d'une négociation contractuelle de sous-traitance « classique », qui va porter soit sur le service de développement d'un logiciel, avec sur une cession totale ou partielle des DPI attachés au logiciel incluant son code source, et possiblement des engagements d'assistance. Si le logiciel est construit sur une base existante (éventuellement des logiciels diffusés sous licence libre), il s'agira de valoriser la contribution de l'institution à l'amélioration de cette base et l'assistance qu'elle peut fournir par la suite.

L'enjeu premier est financier, et il s'agit de défendre au mieux les intérêts patrimoniaux et pécuniaires de l'institution. Afin d'identifier les apports de chacun, il sera important à titre de conservation de preuve d'effectuer un dépôt des versions de référence du logiciel avant et après l'obtention des nouveaux résultats logiciels, auprès d'un tiers séquestre tel que l'Agence de Protection des Programmes (APP) ; les dépôts dans Software Heritage, dans le cadre des publications scientifiques ou non, sont aussi des éléments de preuve.

Les projets logiciels de collaboration

Ces logiciels ont été produits dans le cadre d'un projet de recherche et ont été identifiés comme devant être maintenus. Deux cas poussent à rechercher une collaboration externe pour maintenir et poursuivre le développement de ces logiciels :

Les logiciels considérés comme moyennement stratégiques, mais qui pourraient le devenir : l'ayant-droit veut alors se réserver l'ensemble des possibilités de promotion, et susciter des possibilités de collaboration à travers leur publication ;

Les logiciels identifiés d'emblée comme importants pour l'institution, mais celle-ci ne dispose pas en interne des compétences pour assurer leur développement, et/ou des partenaires se sont manifestés pour l'utiliser.

Dans tous les cas, il est important de conserver le contrôle de l'évolution du logiciel. Les contrats porteront sur des transferts non-exclusifs, ou sur la concession d'exemplaires du logiciel et des codes sources, mais n'entraîneront pas de transfert de DPI. Il est impératif, à ce stade, **que l'ayant droit** (c'est-à-dire en règle générale l'institution qui emploie l'équipe) **conserve les DPI sur l'ensemble du code du logiciel,** pour pouvoir choisir la ou les licences de diffusion. Il est conseillé de mettre en place, dès les premiers contributeurs externes en code, des contrats de cession des droits sur la contribution au profit de l'institution ayant droit du logiciel, afin qu'elle conserve le contrôle sur la stratégie de diffusion et le choix des licences d'exploitation (contrat de type *Contributor License Agreement*).

Le logiciel peut aussi être soumis à une plateforme, un projet logiciel plus vaste (par exemple une bibliothèque développée dans l'environnement R ou PyTorch). Cela peut s'accompagner d'une demande de cession des droits de la part de l'ayant droit du projet. Voir, par exemple, les explications du Pytorch Contributor License Agreement11 sur ces points spécifiques.

Si le logiciel a été construit à partir d'un logiciel détenu par un tiers, plusieurs cas se présentent :

- Le tiers a publié le logiciel sous une licence libre de type copyleft, ou il existe une communauté de développement active autour du logiciel. L'institution devra s'efforcer de voir ses modifications

[10] Les auteurs de ce rapport remercient François Pellegrini pour rappeler ce point.

[11] https://pytorch.org/blog/a-contributor-license-agreement-for-pytorch/

intégrées dans la version de référence du logiciel (le « tronc »), afin que les évolutions ultérieures du projet s'appliquent également au code apporté ;

- Le tiers a publié le logiciel sous une licence permissive (par exemple BSD), sans qu'une communauté de développement active existe. L'institution peut alors considérer qu'il s'agit d'un logiciel « en sommeil » et elle n'est pas limitée dans ses choix stratégiques par la licence ;
- Le tiers a publié le logiciel sous une licence privative permettant à l'institution de faire des développements à partir de cette base. Il conviendrait d'obtenir de ce partenaire, soit une cession des droits, soit une publication du logiciel sous une licence libre (plutôt une licence copyleft, par exemple LGPL, afin de garantir que les contributions de l'institution resteront ouvertes). C'est le principe du consortium garanti par la licence, comme dans les cas d'OW2 ou d'Eclipse.

Si l'institution est ayant droit du logiciel développé, en pratique, une publication sous une licence de type *copyleft*, couplée à un contrat de cession des droits pour les contributeurs externes, est la plus indiquée. Cette publication oblige un partenaire qui souhaite utiliser le logiciel dans une production fermée (« privative ») à s'adresser à l'institution (à son service de relation partenariale et valorisation) qui pourra, moyennant rétribution, lui accorder une licence autorisant cette pratique (système de double licence, v. les cas de MySQL ou de CGAL, fiche en annexe).

La publication sous une licence libre de type *copyleft* permet d'avoir des retours, des améliorations sur le logiciel, ainsi que d'augmenter la base des testeurs et donc l'évaluation de sa robustesse. Il s'agit ainsi d'« externaliser » une partie de la R&D. Signalons que pour que ce système soit efficace, il faut exercer une veille sur les utilisateurs du logiciel, et assurer le respect des conditions de la licence choisie, notamment si elle est de type *copyleft*.

Si une communauté de contributeurs se développe, la question de la gestion des contributions se posera rapidement. S'il est difficile d'évaluer avec précision le coût de cette gestion, on peut estimer qu'elle est au minimum d'un mois-ingénieur par an quand la personne a participé au projet dès son origine. Plus les contributions sont importantes, plus cette charge de travail sera élevée. Si une communauté d'utilisateurs se développe sans être apte à contribuer techniquement mais en continuant à émettre des besoins nouveaux ou améliorations, alors la capacité à arbitrer, à faire évoluer et à faire savoir les choix pour conserver une communauté active est aussi une gageure humaine.

Si cette gestion n'est pas possible en interne, il est souvent intéressant de créer une jeune pousse (par exemple, une *spin-off* du laboratoire) destinée à assurer cette maintenance ou de confier ce travail à un tiers. L'institution pourra être un des clients de ladite jeune pousse (maintenance, développement spécifique) pour assurer son démarrage. La concession à la jeune pousse des droits sur le code peut s'effectuer contre une prise de participation dans l'entreprise, comme c'est le cas à Inria, afin de ne pas lui retirer les moyens financiers dont elle a besoin à son démarrage, ainsi que pour conserver une certaine autorité sur les actions entreprises au sein de celle-ci.

Les projets logiciels stratégiques

Au cœur de la recherche scientifique d'un laboratoire ou du métier d'un service (formation, par exemple), ces logiciels sont identifiés comme stratégiques pour l'institution, qui souhaite s'assurer le contrôle de leur évolution. Comme pour les logiciels « de collaboration », il faut que l'institution soit sûre de pouvoir contrôler l'évolution du logiciel.

Si l'institution n'est pas titulaire des DPI sur l'ensemble du logiciel, il faut alors soit qu'elle obtienne une cession des droits de la part du tiers titulaire, soit qu'elle se garantisse contractuellement l'accès à long terme des codes sources (c'est le cas si le logiciel est le produit d'un projet libre associé à une communauté dynamique).

Dans le cas où l'institution est titulaire exclusive des DPI sur le logiciel (ou peut le devenir), une publication sous une licence de type *copyleft* paraît plus indiquée, car elle permettra de diffuser le logiciel, et donc d'étendre la base d'utilisateurs et d'externaliser une partie de la R&D du logiciel, en ayant accès aux contributions extérieures.

Cela nécessite de continuer à travailler sur le logiciel, pour être à même d'intégrer ces contributions, d'avoir réfléchi au statut de ces contributions, de sorte que l'institution conserve la maîtrise de leur exploitation future (cession des droits en faveur de l'institution contre garantie d'une publication sous licence de type *copyleft*, double licence libre/privative, passage à un statut d'œuvre de collaboration).

Dans tous les cas, l'investissement pour assurer la maintenance du logiciel peut être important. Il est réservé à quelques logiciels stratégiques en termes de recherche et développement, de ressources financières, ou de rétribution en termes d'image pour l'institution. Comme dans le cas des logiciels de collaboration, on peut envisager d'externaliser la maintenance et l'évolution du logiciel protégé par une licence de type *copyleft* à une *jeune pousse*, qui pourrait développer des services (adaptation du logiciel, formation, développement de modules complémentaires).

Exemples :

Société : Probabl. : Fondée en septembre 2023, issue d'un projet confié à Inria par le secrétariat général pour l'investissement dans le cadre de France 2030, la jeune pousse Probabl propose des solutions et des services basés sur la bibliothèque *open-source* d'apprentissage statistique en Python « Scikit-Learn », l'une des plus utilisées dans le monde avec PyTorch et TensorFlow, **«** *afin de pérenniser son existence et d'assurer son rayonnement* ».[12]

Société GeometryFactory : Fondée en 2003, issue du projet logiciel CGAL (voir exemple en annexe).

Le passage d'un logiciel d'un type à un autre.

La promotion d'un logiciel n'est pas une activité statique, réalisée une bonne fois pour toutes. Au cours de sa vie, un logiciel peut évoluer, et son intérêt, son importance stratégique, changer (tant pour l'institution et l'équipe de recherche que pour des utilisateurs éventuels).

Nous avons noté que le choix initial de diffusion et pérennité, notamment, le choix de la licence, doit intégrer cette évolution dès le départ. Pour faciliter la réflexion stratégique d'un gestionnaire, établissement tutelle (GET), nous proposons un schéma illustrant les passages possibles d'un type à un autre (Figure 2), et les actions pour chaque type de pérennité :

- Centralisation des droits pour le cas d'un logiciel stratégique, diffusion et pérennité dans une communauté ou auprès d'un tiers licencié ;
- Publication en ligne avec des perspectives de transfert à une entreprise ou de collaboration avec un tiers ; promotion par une communauté d'utilisateurs.

Si les demandes concernant un projet logiciel (de corrections, d'améliorations, de contributions à intégrer, etc.) sont issues de partenaires dans le cadre d'un projet contractuel, et qu'elles dépassent les demandes initiales, peut se poser la question du financement de leur traitement. Éventuellement, d'autres contrats partenariaux pourront être négociés par les services de relation partenariale et valorisation, pour aller au-delà de la preuve de concept : il s'agit de projets de développement de **logiciels de « commande »** (v. le paragraphe consacré, partie 3.a) 1), si le ou les partenaires sont des partenaires privés « clients » du projet. Cependant, dans tous les cas, il s'agit d'initier un nouveau cycle de développement, qui restera maîtrisé par l'équipe de recherche et ressemblera toujours à celui décrit en Figure 1.

[12] https://www.actuia.com/actualite/probabl-vers-une-science-des-donnees-souveraine-et-open-source/

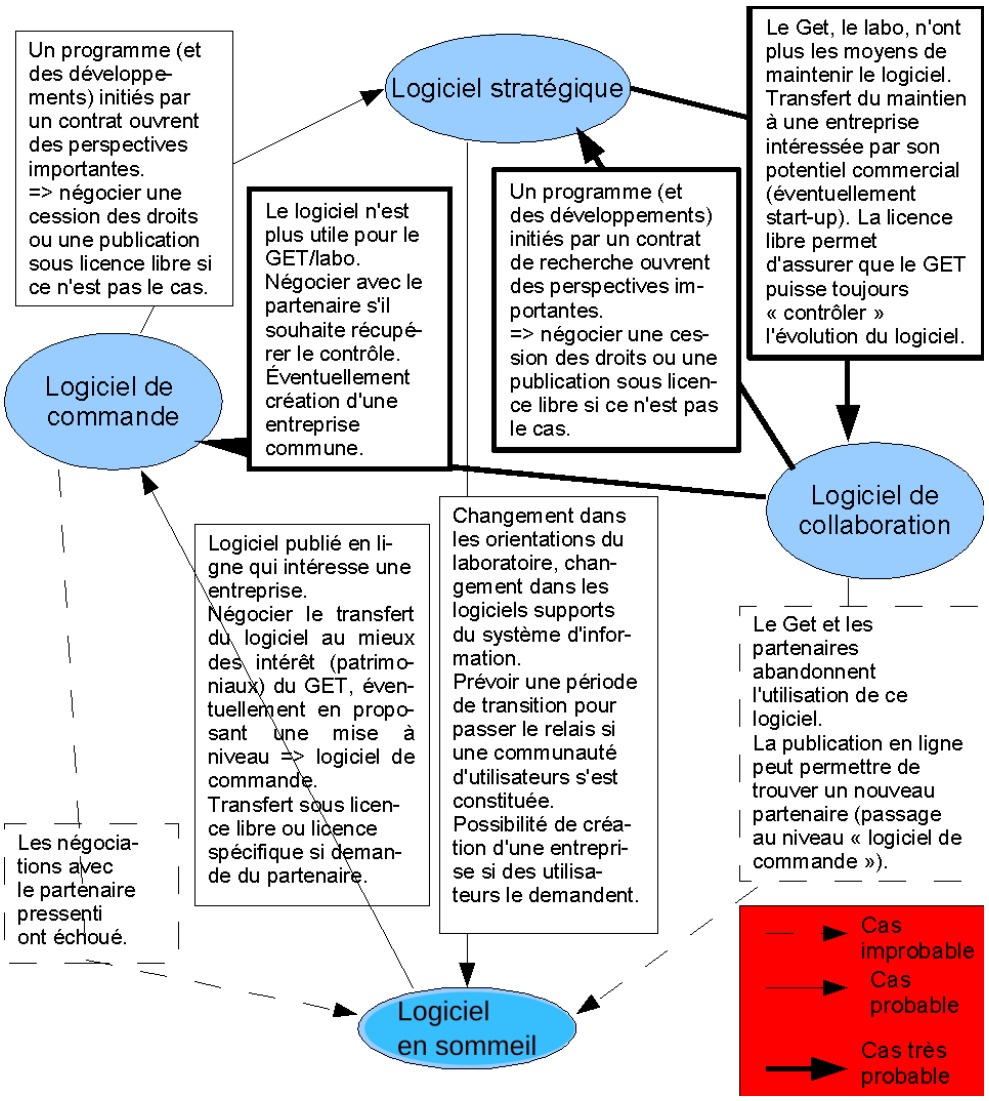

Figure 2 : Stratégies de promotion et évolution de ces stratégies pour les logiciels de recherche. Nb : GET (ou Get) pour « gestionnaire, établissement tutelle »

Les ressources humaines et les outils nécessaires à la production et à la pérennité de logiciels de la recherche

L'enquête permet aussi de souligner que le travail de pérennité est un effort qui n'est pas limité à l'équipe qui a initié le logiciel, mais implique l'ensemble de l'institution, qui doit investir dans la maintenance des logiciels (temps humain), dans des plateformes pour gérer les projets logiciels.

La maintenance de logiciels

Maintenir un logiciel divulgué sous une licence libre est coûteux en temps-personne, surtout si ce logiciel attire de nombreuses contributions (cela peut parfois aller jusqu'à un temps plein). Outre qu'il s'agit de logiciels stratégiques pour l'institution, ces succès contribuent à l'image de l'institution, aussi bien auprès des centres de recherche, que des entreprises ou des étudiants (un logiciel est une réalisation que l'on peut parfois « montrer »).

Il semble préférable que le logiciel développé soit maintenu par le laboratoire qui l'a initié. Ceci veut dire qu'il existe dans les laboratoires des personnels dédiés au développement logiciel (éventuellement partagés

entre plusieurs laboratoires) qui assistent les scientifiques sur le développement des logiciels et pourraient assurer leur maintenance.

Un autre avantage souligné, en sus d'assurer la stabilité dans le suivi des logiciels, est que ces personnes peuvent être des relais dans les laboratoires pour sensibiliser les autres à la diffusion et à la pérennité des développements logiciel.

Néanmoins, nos entretiens montrent que des outils au service de ces scientifiques et laboratoires peuvent être mutualisés, jusqu'au niveau national, au service des besoins et processus généraux de la soutenabilité dans le temps des logiciels :

- Le développement d'un logiciel nécessite toujours la mise en œuvre technique d'un environnement informatique approprié, composé d'un certain nombre d'outils permettant, entre autres, la conception, le développement, les tests, la distribution sous forme de paquetage, le suivi technique, la maintenance, le respect de normes de qualité, etc.
- Un certain nombre de démarches d'ordre « administratif » sont nécessaires pour s'inscrire au plus près du cadre méthodologique recommandé dans les stratégies de pérennité étudiées dans le présent document. Elles doivent être effectuées au bon moment, en parallèle de la mise en œuvre technique évoquée ci-dessus, et bénéficieraient du développement d'un système d'un système unique et accessible ;
- Des activités spécifiques et nécessaires à certains modes de diffusion et pérennité s'ajouteront aux activités purement liées au développement du logiciel (suivi de l'impact en diffusion, contractualisation, promotion et publication, animation de communautés, organisation de tutoriels, etc.). Là encore, le partage des coûts de développement de ces outils de suivi semble attendu par nos interlocuteurs.

L'ensemble de ces trois processus sera d'autant plus efficace qu'il s'appuiera sur des applications informatiques utiles, adaptées, mutualisées, intégrées dans les processus existants, mais aussi accompagnées d'actions de formation de l'ensemble des acteurs de la promotion logicielle.

Plateformes informatiques utiles à la pérennité des travaux de recherche

Des plateformes génériques destinées au support du processus de développement logiciel, ou forges, sont déjà partiellement déployées dans plusieurs institutions françaises (Le Berre *et al.*, 2023). Ces plateformes étant utilisées quotidiennement par les scientifiques, et parfois relativement en amont du processus de développement, il semble intéressant d'envisager l'intégration, au sein de plateformes génériques, de nouvelles fonctionnalités spécialisées ayant trait au support de la démarche de diffusion et pérennité. Certaines fonctionnalités existantes dans ces plateformes de support au développement seront naturellement mieux adaptées à certaines stratégies de diffusion et pérennité.

Ainsi, les plateformes offrant des outils de publication web sont bien évidemment au cœur de dispositifs visant à la diffusion et à la pérennité par des licences logicielles libres. Elles offrent une capacité à communiquer naturellement auprès de l'ensemble de la communauté scientifique mondiale, et à organiser l'interaction avec une communauté d'utilisateurs ou de contributeurs.

Une des difficultés rencontrées par les personnels chargés de partenariat et valorisation est l'identification des nouvelles créations au sein des laboratoires. L'harmonisation des outils utilisés et leur mutualisation pourraient beaucoup aider dans cette tâche. Par exemple, les déclarations de logiciel (analogues aux déclarations d'inventions) peuvent être systématisées dès lors qu'une nouvelle demande d'hébergement d'un projet de développement sur la plateforme mutualisée est émise par un ou une scientifique. Il est ainsi possible d'intégrer la démarche de diffusion et pérennité à l'activité habituelle des personnes créatrices, ce qui nous semble constituer un atout déterminant quant à son acceptation. Ainsi, Inria a mis en place la BIL (Base d'Information des Logiciels), grâce à laquelle tout personnel peut consulter les logiciels sur lesquels travaillent les équipes de recherche Inria, mettre à jour les informations, générer automatiquement dans les sites web des équipes le contenu lié aux logiciels d'équipe, instruire un dépôt APP avec le service STIP (transfert, innovation et partenariat), générer la partie logicielle du rapport d'activité des équipes, créer des catalogues, etc. Nous avons noté d'autres actions, comme le dépôt automatique dans Software Heritage

des logiciels liés à un dépôt HAL[13] ou encore la collecte automatique des citations de logiciel dans les articles mise en place par Inria.

Dans les cas de stratégies de diffusion et pérennité spécifiques, il conviendrait de mettre à disposition de l'ensemble des personnels intéressés des outils plus spécialisés qui ne s'intégreraient pas naturellement dans les plateformes de support du développement existantes. Ce serait le cas, par exemple, d'un portail de publication/téléchargement qui peut intégrer, par exemple, le suivi des citations des logiciels, ou de paiement et/ou d'acceptation de modalités contractuelles spécifiques.

Nous ne détaillons pas ici l'ensemble des fonctionnalités attendues d'un tel dispositif, qui constitue un projet à part entière, et qui fait l'objet d'un groupe de travail spécifique dans le collège Codes sources et logiciels. Quelles que soient les fonctionnalités mises en œuvre, un point essentiel souligné dans nos entretiens est l'identification d'un ensemble limité d'informations de référence, voire de fiches pratiques pour les personnels chargés de partenariat et valorisation souhaitant coordonner ou consolider l'ensemble du dispositif de pérennité. Un nombre limité de plateformes mutualisées bien identifiées, gérant de manière adéquate les contributions et la PI, permettant une pérennité et une souveraineté des informations, et hébergeant une majorité de projets (centralisées dans les institutions) semble être un facteur clé de la mise à disposition de l'ensemble des informations utiles relatives aux projets logiciels en cours.

Intégrer les outils dans les processus existants

La problématique d'informatisation coordonnée du dispositif organisationnel de développement, de diffusion et de pérennité des logiciels au sein des institutions de recherche n'échappe pas à nos interlocuteurs, qui pointent un certain nombre de considérations classiques : adéquation aux besoins, adoption par les utilisateurs, coût des investissements, rentabilité finale du dispositif, etc.

Un travail en ce sens est initié au sein du collège Codes sources et logiciels. Rappelons quelques-unes de ces considérations.

De façon générale, la large mise à disposition d'un ensemble d'outils de support, à la fois des processus de développement, ou des opérations nécessaires dans certaines stratégies de promotion, nécessite une centralisation de moyens conséquents dans des plateformes informatiques sophistiquées. Deux points apparaissent importants à l'issue de notre étude :

- Porter un effort particulier sur l'utilisabilité, l'ergonomie des outils, ainsi que leur intégration dans les dispositifs existants, pour favoriser leur adoption par les scientifiques ;
- Ne pas mettre à disposition un nouvel outil supplémentaire difficile à comprendre, d'autant qu'il pourrait être perçu comme une contrainte administrative supplémentaire. La formation à ces outils sera un pan important du dispositif de formation générale.

Le logiciel n'est qu'un des éléments produits par les scientifiques parmi d'autres tels que les publications, les supports de formations, etc. La plupart de ces artefacts sont aujourd'hui de nature similaire, fortement immatérielle, produits avec des outils informatiques.

La problématique d'un système d'information nécessaire à une amélioration des processus de pérennité pour ce qui concerne le logiciel peut être généralisée à d'autres types de productions. Les outils utilisés par les acteurs, à la fois pour leurs productions ou leurs pérennités, sont probablement similaires et nécessiteront une intégration coordonnée au sein d'un même système d'information global. Tout comme pour les logiciels, un certain nombre des activités de production ou de pérennité de ces travaux met en œuvre des aspects de travail collaboratif avec d'autres collègues, des partenaires, des étudiants, à l'intérieur des institutions de recherche, comme à l'extérieur de ces institutions.

Les aspects évoqués ci-dessus peuvent donc être intégrés au sein d'une démarche globale du système d'information des établissements d'enseignement supérieur et de recherche, intégrant simultanément ces deux grands types d'activités.

Une analyse des plateformes de publication existantes dans l'espace français de la recherche a été produite par le Collège Codes sources et logiciels. D'un point de vue budgétaire, il semble qu'un tel investissement, substantiel, peut néanmoins s'avérer porteurs d'économies vis-à-vis d'une situation où des budgets importants sont mobilisés, de façon « invisible » mais répétée, dans chacun des environnements informatiques mis en œuvre localement au service des projets logiciels.

[13] La procédure de dépôt logiciel est expliquée ici : https://doc.hal.science/deposer/deposer-le-code-source/. Pour ce qui concerne le transfert vers Software Heritage, il est mentionné que « Pour être transféré à Software Heritage, le fichier déposé doit être sous licence libre et ne peut pas être sous embargo ».

Le cadre contractuel

Ainsi que cela a été présenté dans le rapport MESR 2023-2024, et dans ce rapport, plusieurs options de diffusion et pérennité sont ouvertes et chacune d'elles nécessite de choisir les licences appliquées aux logiciels de recherche. Le choix des licences propriétaires ou fermées est bien connu. En revanche, le choix d'une licence ouverte est plus délicat et nécessite de bien être compris. On trouvera dans un premier temps des cadrages pour effectuer le choix d'une licence ouverte.

Dans un second temps, afin de protéger l'avenir des projets logiciels, l'administration peut être amenée à conclure des accords de contribution qui ainsi déterminent les termes de la participation de nouveaux contributeurs dans le projet logiciel.

Le choix d'une licence ouverte

Actuellement, il n'existe pas de règles particulières quant au choix de la licence pour diffuser un logiciel conformément aux principes de la science ouverte. Toutefois, ainsi que cela a été souligné dans ce rapport, la science ouverte rejoint la diffusion sous licences libres, et c'est pourquoi il sera naturel de puiser parmi les licences libres pour diffuser les logiciels en science ouverte.

Diffuser un code source ou un logiciel selon les principes de la science ouverte suppose de choisir une licence dont les conditions permettent au moins l'accès au code source, son usage, sa copie, sa diffusion et, selon les cas, sa modification. Dans le cadre d'une stratégie de pérennité, la possibilité de modifier le logiciel est essentielle.

Il existe deux types de licences libres : les unes qui permettent l'usage, la copie, la diffusion et la modification sans restriction, excepté parfois l'usage à des fins promotionnelles de l'institution au sein de laquelle le code source ou le logiciel a été produit (les licences dites de domaine public, p. ex. BSD, licence X) ; les autres qui ajoutent des conditions pour rediffuser le code ou le logiciel modifié, de sorte qu'il demeure accessible aux mêmes conditions (les licences dites *copyleft*, p. ex. GPL, CeCILL, EUPL).

Enfin, certaines licences dites « ouvertes » n'accordent pas les mêmes droits aux utilisateurs successifs, notamment en interdisant un usage commercial (les licences dites « asymétriques[14] »).

La diversité des licences peut être contre-productive dans la mesure où ces licences ne sont pas nécessairement compatibles entre elles, de sorte qu'il ne sera pas permis de combiner des logiciels les uns avec les autres pour en créer de nouveaux. Or, l'intérêt de rendre un logiciel accessible réside dans le développement de solutions informatiques inédites à partir de l'existant. Si les licences ne permettent pas la combinaison de composants logiciels du fait de conditions incompatibles, l'exploitation de ces derniers s'en trouve fortement diminuée. Par ailleurs, une trop grande diversité de licences réduit la sécurité juridique ou, du moins, complique la tâche des développeurs qui sont contraints de procéder à un audit des licences applicables aux composants logiciels qu'ils voudraient utiliser pour leurs travaux ultérieurs. C'est pourquoi il est préférable de limiter le choix à une liste prédéfinie qui peut varier selon le type de stratégie retenue.

Par exemple, en matière de logiciels publics, c'est-à-dire les logiciels développés par l'administration publique et non issus de travaux de recherche scientifique, le choix de leur licence est celui du régime juridique de l'article L. 300-2 du Code des relations entre le public et l'administration. Trois hypothèses sont prévues à l'article L. 323-2 du CRPA :

1.Un principe : la liste des licences applicables est prédéfinie lorsque la réutilisation à titre gratuit donne lieu à l'établissement d'une licence, cette licence est choisie parmi celles figurant sur une liste fixée par décret (actuellement : décret n° 2017-638 du 27 avril 2017 relatif aux licences de réutilisation à titre gratuit informations publiques et aux modalités de leurs homologations – article D. 323-2-1 du CRPA) ;

2.Une limite au principe : possibilité d'obtenir l'autorisation de recourir à une autre licence que celles prévues par décret lorsqu'une administration souhaite recourir à une licence ne figurant pas sur la liste, celle-ci doit être préalablement homologuée par l'État dans des conditions fixées par décret ;

3.Une dérogation au principe : le respect du choix préétabli lors de la participation à un projet développé et diffusé par des tiers, l'administration devant respecter les termes de la licence applicable au projet.

[14] Voir les licences appliquées aux LLM (Large Language Model) Mélanie Clément-Fontaine (2024)

À la différence des logiciels publics, le choix de pérennité des logiciels de recherche peut amener à exploiter ces derniers sous des licences propriétaires. On souligne ici que « les licences propriétaires sont choisies pour protéger des intérêts commerciaux, assurer une exclusivité sur la technologie ou encore pour générer des revenus. […] Ce choix [celui d'une licence propriétaire] peut aussi s'accompagner d'une double licence libre et propriétaire afin de conserver des possibilités de choix ultérieurs » (MESR, 2023-2024).

Qui peut décider du choix de la licence ? Pour répondre à cette question, il faut distinguer deux hypothèses :

- **- Première hypothèse, le logiciel est développé au sein d'une seule institution publique :** en ce cas, l'institution publique dispose des DPI sur le logiciel et par conséquent peut choisir librement la licence aux conditions de laquelle il sera diffusé. Tel est le cas, en particulier, lorsque le code source ou le logiciel est créé par un agent public (article L. 113-9, alinéa 3 du CPI) ou « des personnes qui sont accueillies dans le cadre d'une convention par une personne morale de droit privé ou de droit public réalisant de la recherche créent des logiciels dans l'exercice de leurs missions ou d'après les instructions de la structure d'accueil si elles se trouvent à l'égard de cette structure dans une situation où elles perçoivent une contrepartie et où elles sont placées sous l'autorité d'un responsable de ladite structure » (L. 113-9-1 du CPI), comme par exemple un stagiaire.

- **- Seconde hypothèse, le logiciel est issu de logiciels ou d'un projet initié par des tiers** : dans ce cas, l'institution n'a pas d'autre choix que de se conformer à la licence appliquée aux logiciels utilisés ou au projet. C'est la raison pour laquelle, dans cette hypothèse, il n'est pas possible d'imposer en amont une liste limitative de licences utilisables pour les projets en science ouverte (voir la limite au principe prévu à l'article L. 323-2 du CRPA).

Les accords de contribution

Dans le cas où des personnes extérieures, c'est-à-dire différentes de celles à l'origine du projet et appartenant à des institutions différentes, souhaitent contribuer au projet ainsi ouvert, il convient de mettre en place un accord de licence de contributeur, ou Contributor License agreement (CLA) en anglais. Ce document juridique vise à protéger l'avenir du projet logiciel en faisant signer à chaque contributeur externe une autorisation d'usage de sa contribution. Ce système permet de laisser aux personnels créateurs du code original, et à l'institution titulaire des DPI, le choix sur le devenir du code. Ceci est important en particulier dans le cas où les titulaires des DPI souhaitent diffuser le code sous une nouvelle licence (changement de licence ou diffusion sous licences multiples aussi appelé « double licence »).

Ces accords de contribution, au niveau individuel comme institutionnel, permettent de rester maître de la pérennité d'un code, en particulier pour les codes définis ci-avant comme « stratégiques » tout en autorisant des contributions extérieures.

4 |Conclusion

Aujourd'hui, de nombreux personnels sous-estiment l'importance de protéger les logiciels pour pouvoir les promouvoir au mieux et les soutenir dans le temps, c'est-à-dire pour pouvoir choisir la licence (éventuellement libre) la plus appropriée à la situation. La raison en est souvent que les scientifiques, à travers leur mission de diffusion de la connaissance, ont des difficultés à concevoir que les termes de « protection » et de « publication » ne sont pas forcément antagonistes.

Qui plus est, les personnels ignorent parfois le fait que les choix techniques ou juridiques/administratifs initiaux ont des conséquences sur l'avenir de la diffusion/pérennité du logiciel.

Ceci est d'autant plus préjudiciable à la diffusion et à la pérennité qu'il faudrait agir au plus tôt du processus de création, et notamment avant l'intervention de tiers extérieurs aux institutions (collaboration, diffusion), qui complexifie souvent les situations juridiques.

Plus généralement, le personnel n′est pas systématiquement (in)formé (des)aux aspects de la propriété intellectuelle et, pendant longtemps, les établissements n'ont pas toujours reconnu la production de logiciel comme un élément de production scientifique au même titre que les articles scientifiques ou les brevets. En conséquence, participer à un projet d'édition logicielle a souvent nui à la carrière des scientifiques. Depuis juillet 2022, un accord pour la réforme de l'évaluation de la recherche, auquel adhèrent plus de 40 pays, a défini quatre principaux engagements, notamment la reconnaissance et valorisation des rôles techniques qui peuvent inclure ceux de gestion de projet logiciel et d'animation de communauté.

Ce rapport a proposé une synthèse des stratégies de diffusion et pérennité logicielle possibles, et les actions à mener pour les réaliser.

Finalement, ce qui ressort de notre travail d'enquête est le fait que la pérennité tant des logiciels que de l'organisation qui les crée, est une démarche, au même titre que la qualité ou la sécurité. Il ne s'agit donc pas d'un investissement ponctuel mais d'un processus qui nécessite des moyens pérennes et une implication constante des salariés ainsi qu'une reconnaissance par le Gestionnaire ou Établissement de Tutelle (GET) des laboratoires dans la gestion des carrières.

Les retours d'expérience pratiques disponibles en matière de développement logiciel, provenant des scientifiques et des chargés de partenariat et valorisation participant à ce groupe, incitent à travailler simultanément sur trois axes :

1. Consolider ou développer des stratégies de pérennité, notamment telles que décrites dans la partie 3. Le choix des licences sera évidemment un point clef de la réussite de ces stratégies, mais il doit intervenir en conclusion d'une analyse stratégique, pas en préambule, et la stratégie doit être construire au niveau de l'établissement, en lien avec les objectifs fixés par l'établissement sur les productions logicielles, notamment en matière de science ouverte ;

2. Intégrer, autant que possible, ces dimensions de diffusion et de pérennité dans les pratiques actuelles des personnels, voire proposer des outils mutualisés au niveau national qui leur facilitent le travail (notamment dans le développement des logiciels et la gestion des co-développements). Bien construits, ces supports permettront de faciliter la détection de la création, d'identifier les auteurs des logiciels, de garantir la protection de la création et la gestion de la diffusion de celle-ci ;

3. Acculturer tous les personnels, dans les laboratoires, mais aussi dans les services de relation partenariale et valorisation, à la propriété intellectuelle et à la pérennité du logiciel, formation qui déclinera chaque étape du processus de diffusion et pérennité de la production, à savoir :

- principes généraux de la propriété intellectuelle ;
- déclaration du logiciel qui permet d'identifier le ou les auteurs du logiciel, le contexte de sa création et la titularité des droits sur celui-ci et d'indiquer, le cas échéant, les opportunités de diffusion et pérennité du logiciel ;
- diffusion et pérennité possibles ou souhaitées par l'entité (financière, industrielle, en termes de publication scientifique, de diffusion du logiciel, d'apport pour des usages sociaux, etc.) dans le cadre des objectifs stratégiques de l'établissement ;
- outils juridiques existants (contrats, tout spécialement les licences, moyens de protection tels que le dépôt APP, les mentions de droit d'auteur, etc.) ;
- dispositifs opérationnels et expertises disponibles dans l'entité employeuse, et notamment formalisation du rôle des juristes et des chargés de partenariat et valorisation comme aide, conseil pour s'approprier ces points ;
- éléments permettant de mesurer l'impact du logiciel (publications dérivées, citations, contrats obtenus) et donc d'évaluer, au même titre que les publications classiques, l'activité scientifique des personnes impliquées dans un projet logiciel.

Nous espérons que ce rapport contribuera à l'instauration d'un tel dialogue.

Annexes

Lexique

API (Application Programming Interface) : Interface de programmation applicative (IPA).

APP : Agence pour la protection des programmes.

Bibliothèque (logiciel) : Ensemble de fonctions utilitaires, regroupées et mises à disposition afin de pouvoir être utilisées par un tiers sans avoir à les réécrire. Les fonctions sont regroupées par leur appartenance à un même domaine conceptuel (mathématique, graphique, tris, etc.). Les bibliothèques logicielles ne représentent pas une application : elles ne sont pas complètes, et notamment ne possèdent pas les caractéristiques d'un programme, telles qu'une fonction principale.

Bien public : bien qui, une fois produit, est difficilement rival. Il peut être utilisé par plusieurs personnes, en même temps, sans que l'utilité de son utilisation ne soit diminuée. De plus, sans certains ajouts techniques ou barrières légales, il est non exclusif, puisque tout fragment de code sous forme numérique peut être copié et transféré à un coût presque nul.

Code source : Ensemble des instructions et des lignes de programme du logiciel écrites par un humain, auquel l'accès est nécessaire en vue de modifier le logiciel.

Code objet : fichiers issus de la compilation de code source. Lorsqu'ils sont codés en langage machine, ils peuvent être exécutables sur ordinateur[15].

Démonstrateur (aussi **preuve de concept**) : Logiciel applicatif modèle qui doit être adapté pour être employé dans un contexte et un environnement spécifique. À ce titre, un démonstrateur n'est pas un logiciel applicatif destiné à un emploi opérationnel dans l'institution.

Diffusion : partage auprès d'un public différent de celui qui a développé à l'origine. Par définition, cela signifie qu'une licence définissant les conditions de ce partage a été choisie par les ayants droit et est indiquée de manière explicite.

Éditeur de logiciel : entité qui organise la production du logiciel, notamment la feuille de route, la construction de l'architecture et l'intégration des contributions, les différentes versions. Attention : le diffuseur (« *publisher* » en anglais) est celui qui diffuse, éventuellement contre rémunération, des versions du logiciel.

Impact : se dit de la phase dans laquelle le logiciel est utilisé par d'autres acteurs que le laboratoire éditeur du logiciel.

Inner source : L'« *inner source* », ou « *innersource* » en un seul mot, est l'adoption des pratiques de développement logiciel « *open source* » et l'établissement d'une culture de type open source à l'intérieur d'une organisation ». Cela veut dire, notamment, que les codes sources développés dans une organisation sont visibles de l'ensemble des membres (développeurs) de l'organisation, et que chacun peut proposer des modifications. Les équipes de développement doivent être prêtes à accepter des demandes/critiques/contributions d'autres membres de l'organisation, mais aussi avoir du temps pour gérer un projet logiciel sous forme ouverte.

Logiciel métier : un logiciel qui répond à une application d'un secteur thématique donné

Licence : une licence est contrat par lequel le titulaire des droits de propriété intellectuelle détermine les conditions d'accès de l'œuvre auquel il s'applique.

---

[15] L'opération de compilation du code source peut également avoir pour but de le minifier ou de l'offusquer.

https://fr.wikipedia.org/wiki/Developpement_de_logiciel : pratiques de développement logiciel

https://fr.wikipedia.org/wiki/Open_source : open source

https://fr.wikipedia.org/wiki/Inner_source : inner source

http://www.gnu.org/licenses/licenses.fr.html#GPL

European Union Public Licence (EUPL) : rédigée par la Commission européenne, cette licence a pour objectif principal de permettre la jouissance de l'œuvre et de sa création tout en garantissant que ces libertés soient maintenues. Elle remplit les critères des licences *Copyleft*.

General Public License (GNU GPL) : licence de diffusion de logiciels de type *copyleft* obligeant la rediffusion des codes modifiés sous une licence compatible. L'objectif général de la GPL est de mettre à disposition le logiciel de sorte que quiconque puisse à toute fin exécuter le programme, l'étudier, l'adapter, le copier et le diffuser. La licence est élaborée à partir d'un concept appelé le *copyleft*. Le nom est un jeu de mots élaboré en réponse au *copyright*. Selon le concept du *copyleft*, chacun peut participer pour alimenter un fonds commun, mais ne peut rien en soustraire *(art. 2)*. Concrètement, lorsqu'une personne contribue à l'œuvre, elle ne peut pas se réserver l'exclusivité de cette contribution dès lors qu'elle la divulgue. Au contraire, elle doit accorder les mêmes libertés que celles dont elle a bénéficié pour jouir de l'œuvre.

Lesser General Public License, anciennement appelée **Library General Public License (LGPL)** : version modifiée de la GPL pour être moins contraignante quant à l'utilisation dans un contexte de cohabitation avec les logiciels propriétaires.

Licence BSD Licence Berkeley Software Distribution : L'objectif principal de la licence est de permettre toute utilisation de logiciel sans imposer le maintien de ce libre usage aux utilisateurs ultérieurs. En effet, la licence n'impose pas au licencié de rediffuser le logiciel modifié ou tel quel sous la même licence. Cela permet aux utilisateurs de changer les conditions d'utilisation du logiciel. À ce titre il ne s'agit pas d'une licence *copyleft*.

Licences *copyleft* : ces licences organisent une liberté pérenne en ce sens que l'autorisation de copier, diffuser et modifier n'est accordée qu'à condition de conférer les mêmes libertés à autrui sur ses propres contributions (par exemple : la licence publique générale GNU, la licence CeCILL, l'*European Union Public Licence*, la licence *Creative Commons* Paternité - Partage des conditions initiales à l'identique, et la Licence Art Libre ci-après présentées). Concrètement, la personne qui décide d'utiliser l'œuvre soumise à ce type de licence ne doit se réserver ni l'usage de l'œuvre telle quelle ni l'usage de l'œuvre modifiée dès lors que ces modifications sont divulguées. L'objectif est de créer un fonds commun auquel tout le monde peut contribuer, mais duquel personne ne peut retrancher. Ainsi l'auteur qui soumet son œuvre à une licence libre peut espérer bénéficier de la jouissance des contributions des autres. Ces licences offrent la garantie que l'œuvre évoluera au gré des contributions spontanées de tout à chacun.

Licences libres : « peuvent être définies comme des licences par lesquelles l'auteur autorise l'usage, la copie sans restriction, la modification et la diffusion de l'œuvre modifiée ou non, de façon non exclusive, sans transférer les droits d'auteur qui y sont attachés et sans que l'utilisateur ne puisse réduire ces libertés tant à l'égard de l'œuvre originelle que de ses dérivés. Les œuvres ainsi diffusées sont qualifiées de libres » (Clément-Fontaine, 2014).

Licences de libre copie : Licences se distinguant des licences libres dans la mesure où elles n'autorisent pas la modification de l'œuvre, de sorte que l'œuvre n'est ni évolutive ni à pluralité d'auteurs puisque nul autre que l'auteur initial peut apporter des modifications. Il ne s'agit donc pas d'œuvres libres.

Licences de type domaine public (dites licences permissives) : Licences ne garantissant pas une liberté pérenne (par exemple la licence *Berkeley Software Distribution ci-avant présentée)*. Les licences de type domaine public sont appelées ainsi, car elles sont censées rendre l'œuvre aussi librement utilisable qu'une œuvre parvenue dans le domaine public. Elles confèrent les libertés de copier, diffuser et modifier l'œuvre sans qu'il y ait l'obligation d'accorder les mêmes libertés lors d'une rediffusion de l'œuvre modifiée ou non. Aussi, toute personne peut, si elle le souhaite, jouir de l'œuvre sans pour autant soumettre l'œuvre modifiée aux conditions d'une licence libre. En ce cas, la nouvelle version de l'œuvre ne sera plus libre. Cette particularité atténue de manière significative leur intérêt pour la création d'une œuvre libre dont l'évaluation peut à tout moment être empêchée.

Licence permissive : voir Licences de type domaine public

Logiciel : œuvre de l'esprit utilisant l'outil informatique pour effectuer des traitements spécifiques (bibliothèque, applicatif, service Web, API, logiciel système etc.). Sont inclus les codes sources et codes objets ainsi que les données et la documentation nécessaires au fonctionnement de l'outil.

Logiciel applicatif : Type de logiciel conçu pour automatiser l'activité de l'utilisateur final d'un système informatique. Cette caractéristique le distingue des autres types de programmes informatiques, tels que les systèmes d'exploitation (qui sont ceux qui font fonctionner l'ordinateur), les langages de programmation (qui permettent de créer les programmes informatiques en général) et les logiciels utilitaires (qui accomplissent des tâches de manutention ou d'usage général).

Logiciel dérivé : Œuvre de l'esprit réalisée en utilisant tout ou partie d'un logiciel initial pour lui apporter des modifications (logiciel utilisant une bibliothèque, personnalisation d'un applicatif, etc.).

Maturation : Accompagnement du transfert de l'édition du logiciel.

Prématuration : Financement visant à évaluer l'intérêt d'un logiciel pour une communauté d'utilisateurs.

Production d'un logiciel : Processus conduisant à la création d'un objet qui peut aller au-delà d'un code à usage unique qui appuie un travail scientifique, mais n'ayant pas *a priori* vocation à être réutilisé dans une communauté plus large, et intègre au fil du temps une accumulation d'ajouts externes qui doivent être coordonnés dans son architecture.

Transfert : délégation de l'édition du logiciel, du laboratoire créateur à un organisme privé. Cela peut prendre la forme de la création d'une entreprise jeune pousse ou la concession de licences à des entreprises existantes.

Valorisation : La valorisation des résultats de la recherche porte sur toute activité qui, sans nécessairement augmenter la connaissance, trouve un développement 1) dans le monde socio-économique (transfert de technologie et/ou de connaissance, support à l'industrie et aux services, start-ups, brevets, licences, etc.), 2) dans la société pour son organisation et son évolution et en appui des politiques publiques locales et nationales » (HCERES, 2020). Cette définition ne couvre pas la valorisation académique des productions de la recherche telles que les publications scientifiques, pourtant l'enquête du MESR de 2023-2024 met en exergue que les scientifiques ayant répondu à l'enquête citent largement cette valorisation. Nous lui préférons les termes diffusion et pérennité, nous couvrons ainsi tous les axes : socio-économique, économique, sociétal, scientifique (citations des logiciels ou des publications, participation à des projets de recherche), notoriété...

Synthèse des entretiens réalisés

Tableau 1 : synthèse des études de cas réalisées

Nom du logiciel et lien vers le projet	**Institution porteuse principale (ou « GET », gestionnaire – établissement tutelle du projet)**	**Autres institutions porteuses**	**Autres institutions participantes**	**Début du projet**	**Licence actuelle**	**Nombre d'entretiens**
Coq https://coq.inria.fr/)	Inria		Univ. Paris Saclay, ENS Lyon, Université Paris Cité	1984. Diffusé sous licence LGPL à partir de 1999	LGPL	3
GPac https://gpac.wp.imt.fr/	Telecom Paris	Société créée en 2012 : Motion Spell		premiers codes développés en 1999, projet libre initié en 2003 (en licence GPL)	LGPL (depuis 2005)	3
Pl@ntnet https://plantnet.org/	CIRAD		Inria, INRAE, IRD	2008	MIT principalement, et GNU GPL V3.0 et BSD-2-clauses	1
Gama https://gama-platform.org/	IRD		Université Thuyloi (Hanoi), Université Can Tho, Université Nationale du Vietnam (Hanoi), INRAE, CNRS, Université Paris-Saclay.	2009	GNU GPL V3.0 pour la plateforme, MIT pour les modèles	2
CMCDOT	Inria	IRT NanoElec	CNRS, CEA	2014	Licence propriétaire et des modules *open source*	2
CGAL	Inria	Spin-off créée en 2003 GeometryFactory		1996	Double licence GPL3+ propriétaire (GeometryFactory)	2

Analyse de quelques exemples de projets de logiciels scientifiques

Coq

Objet du logiciel

Assistant de preuve de théorème, ou de programme. D'après la page Wikipédia du projet, Coq permet de « manipuler des assertions du calcul ; vérifier mécaniquement des preuves de ces assertions ; aider à la recherche de preuves formelles ; synthétiser des programmes certifiés à partir de preuves constructives de leurs spécifications. » Site Web du projet : https://coq.inria.fr/

Plateforme de développement et nombre de contributeurs (projet principal) : https://github.com/coq/coq, 227 (quelques dizaines de contributeurs au plus ne sont pas décomptés par GitHub, car ayant contribué uniquement avant le passage à cette plateforme).

Création de valeur

De nombreuses preuves de théorème réalisées grâce à Coq, notamment le théorème des « 4 couleurs ».

De nombreuses utilisations en production de programmes sûrs (test et application, notamment chez Dassault Systems, Bull, France Télécom, une entreprise espagnole et une étasunienne membre du consortium, cf. Infra, mais aussi Google, Amazon, Apple, etc.), mais pas de liste réellement tenue. Création de nombreuses communautés thématiques (autour de thèmes scientifiques, mathématiques)

Utilisation dans de nombreuses universités (en Europe, mais aussi aux É-U : MIT, Penn State, etc.).

Conséquences « indirectes » de Coq, expertise des laboratoires sur la preuve de programme : Frama-C et Altergo qui est un SMT solver. Dans la suite, développement de CompCert (compilateur, toujours par l'Inria) qui a largement contribué au succès de Coq dans le domaine des langages de programmation, et qui a reçu à son tour le prix ACM en 2022.

Captation de valeur

Coq est utilisé à partir des années 1990 comme plateforme dans plusieurs projets de recherche contractuels (crédits des fonds ministériels puis ANR à partir de sa création en 2005) et projets européens. Il a donné lieu à plusieurs thèses CIFRE avec les entreprises françaises mentionnées.

Deux prix ACM en 2013 (notoriété internationale). L'utilisation dans les universités étasuniennes précède le prix ACM et l'obtention de ce prix a été poussée par les chercheurs étasuniens.

Très forte visibilité et notoriété apportée par Coq.

Moyens, ressources

Inria a financé du personnel pour maintenir le projet. Sur les aspects technologiques, les chercheurs restent très en première ligne car souvent les ingénieurs sont trop peu expérimentés (jeunes ingénieurs), Le financement de thèse est efficace pour des sujets de recherche (prospectifs) et pour le transfert de connaissance avec les industriels. Aujourd'hui, Inria a recruté des ingénieurs pour travailler spécifiquement sur The Rocq Prover et qui avaient fait une thèse voire un post-doc sur Coq. Le service d'assistance aux équipes d'ingénierie de l'Inria peut également être sollicité. Cela semble avoir stabilisé le projet.

La participation au développement du logiciel est perçue comme un frein à la carrière classique du chercheur pour les porteurs les plus impliqués, et le prix ACM comme une reconnaissance de ce travail.

GPAC (GPAC Project on Advanced Content)

Objet du logiciel

Plateforme implémentant les différents éléments de la norme vidéo MPEG4. D'après la page Wikipédia présentant le projet (en anglais, traduit par nous) : GPAC fournit trois ensembles d'outils basés sur une bibliothèque centrale appelée libgpac : Un lecteur multimédia, MP4, qui est un client basé sur une ligne de commande multiplateforme ou avec une interface graphique Osmo4 ; un packager multimédia, MP4Box ; quelques outils serveur, autour du multiplexage et du streaming (en cours de développement). Site Web du projet : https://gpac.wp.imt.fr/ et page Wikipédia du projet : https://en.wikipedia.org/wiki/GPAC_Project_on_Advanced_Content

Plateforme de développement et nombre de contributeurs (projet principal) : https://github.com/gpac/gpac, 62

Création de valeur

En interne, cela permet d'avoir une plateforme fiable, stable pour :

- tester de nouvelles idées, des nouveaux algorithmes (notamment produits dans le cadre de travaux de doctorant) ;
- diffuser rapidement ces résultats de la recherche.

Une très grande utilisation par les centres de recherche publics et privés pour tester/implémenter des algorithmes.

Dès le début de la plateforme (fin 2004, 2005), elle intéresse les industriels, mais certains souhaitent avoir une licence classique (leur permettant de ne pas forcément rediffuser le logiciel quand intégré dans leurs produits).

Favorise la position de l'équipe et du département dans les instances de normalisation (notamment MPEG)

Captation de valeur

- plusieurs contrats de licence (deux entreprises, européenne et étasunienne, plus tard asiatique) et maintenance (notamment entreprises internationales asiatiques) et d'autres contrats ;
- de nombreux projets de recherche contractuels (Europe, ANR, etc.), grâce à la plateforme, qui permet de tester des choses. Cela assure :
    - une autonomie très large à l'équipe (financements, embauche d'ingénieurs pour le développement, doctorant) ;
    - une position reconnue dans les instances de normalisation (capacité à expertiser et à valider la faisabilité technique d'une proposition) ;
    - des collaborations avec de nombreux centres de recherche européens, sur la partie recherche, normalisation, même s'ils ne participent pas directement en code.

Plateforme récompensée (au travers du groupe MPEG) en 2021 et 2022 par un Emmy Award pour sa contribution technique à la diffusion des vidéos (difficile de trouver des informations là dessus à part sur le site de GPAC, peu de communication institutionnelle).

Création d'une entreprise, MotionSpell (2012) qui développe les aspects licensing et support pour lesquels l'école et l'équipe de recherche/développement n'avait pas les ressources, ni en terme contractuel (gestion des contrats, réactivité commerciale, équipe valorisation), ni en termes de support (réponse aux demandes d'assistance d'entreprise, réactivité et support 365). La valorisation commerciale (chiffre d'affaires) se fait principalement aux États-Unis (plus de 80 % du CA de l'entreprise).

Moyens, ressources

Service valorisation : soutien important sur les aspects juridiques (propriété intellectuelle) au moment du rachat de la PI (2004-2007), pas d'aide sur la partie stratégique (type de valorisation, pourquoi). L'argument était que c'étaient aux chercheurs de décider de leur stratégie de valorisation. **La valorisation économique n'est pas forcément très soutenue par l'école (2 ans pour mettre en place l'accord, pas de suivi des reversements générés).**

Les contrats permettent d'embaucher des ingénieurs qui participent aux activités de l'équipe (recherche, développement logiciel de la plateforme).

Maintenir une telle plateforme est très coûteux en temps pour les permanents. Ce n'est pas reconnu dans l'évolution de la carrière, notamment parce que ça se fait au détriment de la publication scientifique classique (articles, citations). Cette valorisation par la normalisation, par la diffusion d'une plateforme logicielle, freine le développement de l'équipe (recrutement de permanents, notamment quand il faut remplacer un départ) et la promotion de ses membres (sachant que la publication scientifique classique n'était pas forcément leur priorité).

Le mouvement d'open-science, de reproductibilité, de valorisation de la publication de jeux de données est perçu comme favorisant ce genre de plateforme.

Pl@ntNet

Objet du logiciel

Pl@ntNet est une plateforme de science citoyenne qui s'appuie sur l'intelligence artificielle (IA) pour faciliter l'identification et l'inventaire des espèces végétales. Il s'agit de l'un des plus grands observatoires de la biodiversité au monde, avec plusieurs millions de contributeurs dans plus de 200 pays.

L'application Pl@ntNet, disponible en version web et sur smartphone (android, iOS), vous permet d'identifier des dizaines de milliers d'espèces de plantes simplement en les prenant en photo.

Pl@ntNet est basé sur un principe d'apprentissage coopératif. Les utilisateurs ayant créé un compte peuvent partager leurs observations et celles-ci peuvent être révisées par la communauté et utilisées par l'IA pour lui apprendre à reconnaitre les plantes. Il est par exemple possible de confirmer le nom d'une espèce ou bien de suggérer une autre détermination si l'on s'y connait un peu en botanique. Seules les observations qui atteignent un degré de confiance suffisant sont ensuite ajoutées à la base de données publique et utilisées pour l'entraînement de l'IA.

Création de valeur

1.Adaptation du consortium pour financer un poste d'ingénieur d'études (IE) pour maintenir la plateforme. L'IE initial a quitté le projet en 2017. Le consortium finance aujourd'hui le personnel.

2.2013 = discussion sur modèle économique. L'équipe a brainstormé jusqu'à 2018. Ont profité des opportunités de valorisation qui se présentaient. InriaSoft pour le soutien de la structure en 2019.

3.Aujourd'hui, le consortium développe pour des acteurs privés, des nouvelles fonctionnalités, etc. Mais pas d'accès direct au code.

4.Fort impact médiatique, dès le salon de l'agriculture notamment avec Quechua.

5.Tela Botanica pour le public test, dès les premières années du projet.

Captation de valeur

6.Projet Etendard, AGROPOLIS Fondation.

7.Puis PIA FLORISTIC en 2014 avec ANRU pour passage à l'échelle pour des usages à vocation éducative ou promotion de la culture scientifique.

8.2019, financement projet européen.

Moyens, ressources

Au départ : interaction en interne de l'équipe, problématique de pérennisations car pas les mêmes visions en fonction des personnels de valorisation des tutelles. Chacun voyait son intérêt.

Aujourd'hui : dynamique fluide

Contractualisation avec les services de transfert et valorisation pour modèles types

GAMA Plateform

Objet du logiciel

Gama est une plateforme pour construire et explorer des modèles informatiques à base d'agents. Le logiciel permet de représenter des phénomènes sociaux et naturels complexes en proposant un environnement de développement complet et accessible, y compris pour des non informaticiens.

La plateforme permet de représenter toutes les problématiques impliquant la complexité des interactions d'acteurs sociaux. Les usages les plus avancés se sont fait dans le domaine de la ville durable : sur des problématiques urbaines, comme la réponse et résilience aux catastrophes, les enjeux de mobilité présents et futures, la pollution urbaine, la gestion territoriale mais également un axe fort pédagogique pour former à la modélisation et avec les modèles.

Gama est un outil accessible à la fois pour diffuser les modèles scientifiques et former à la modélisation informatique. Il est utilisé au Nord et au Sud pour initier les scientifiques et acteurs sociaux à la construction de modèles. Gama en tant qu'outil pédagogique permet de mieux comprendre la complexité des enjeux socio-écologiques et la multiplicité des réponses et de leurs impacts. La plateforme est également un instrument versatile au service d'une recherche résolument transdisciplinaire. En effet, les modèles à base d'agents sont particulièrement adaptés pour interfacer des modèles informatiques de diverses disciplines tout en intégrant explicitement les facteurs humains, sociaux et comportementaux. Du terrain au laboratoire, Gama permet de soutenir une recherche action pour la durabilité de nos modes de vie.

Gama répond au besoin essentiel de transdisciplinarité appelé par les sciences de la durabilité.

Création de valeur

La valorisation économique a débuté, mais est en cours de précision. Des start-ups se créent à partir de modèles développés à l'aide de la plateforme, comme par exemple MaeLab, start-up autour du modèle Maelia qui est basé sur GAMA.

La problématique rencontrée est que, dans le cas de la création de jeune pousse, aucune référence à Gama n'est réalisée. L'équipe souhaite alors revenir sur cet aspect en définissant une stratégie de valorisation qui permettrait plus de reconnaissance de leur travail.

La plateforme permet de nombreux partenariats publics et privés depuis sa création, dans des domaines variés.

Le MIT MediaLab est un partenaire et s'appuie largement sur Gama dans le cadre de son meta-projet CityScope, notamment à travers l'activité de recherche d'Arnaud Grignard (site web).

l'IMT Lille-Douai a sollicité l'équipe Gama pour produire une solution de symbiose-industriel (dépôt de projet Horizon Europe en avril 2023)

GaneshAid au Viet-Nam a souhaité être formé et accompagné en 2021 dans le développement d'une solution logicielle pour étudier les futures migrations climatiques au regard de la structure démographique de l'Asie du Sud-Est.

La Banque mondiale s'est mise en relation avec l'équipe en 2020 pour construire des modèles de la propagation de l'épidémie de Covid-19 dans les camps de réfugiés.

Un partenariat existe depuis près de 20 ans avec EDF R&D dans le cadre du projet SMACH, qui a permis de nombreuses contributions réciproques (usage de Gama pour la simulation de l'activité quotidienne des ménages, apport d'EDF pour la génération de populations synthétiques, etc.). Il s'est formalisé en 2020 dans le cadre du Plan de Relance, avec la mise en place d'un post-doc entre EDF et l'IRD.

Captation de valeur

La première source de financements : ANR. De 2008 à 2022, plusieurs projets ont ainsi été obtenus grâce à son existence et ont permis, en retour, de développer la plate-forme.

3Worlds (2008-2011), en partenariat avec l'ENS Paris, sur la modélisation des éco-systèmes végétaux, a permis de recruter les post-docs qui ont posé les bases de l'architecture actuelle;

GenStar (2013-2017), en partenariat avec l'IRIT et EDF, a permis de recruter d'autres contributeurs toujours actifs, de financer plusieurs thèses et de fournir de nombreux stages à des étudiants de Master dans le cadre de la plate-forme;

ESCAPE (2016-2021), en partenariat avec IDEES et le LITIS et le bureau d'études BRLI, a joué le même rôle, tout en permettant de tester les capacités de la plate-forme à s'insérer dans une démarche d'aide à la décision (sur la gestion des évacuations en cas d'inondations).

D'autres projets ANR, dont l'IRD ne faisait pas directement partie, ont aussi directement contribué à GAMA (Acteurs, porté par IDEES, SWITCH, porté par l'IRIT, MIRO3 porté par Geographie-Cité, etc.). En comptant tous les soutiens financiers, il est ainsi possible d'estimer que l'apport financier de l'ANR a été d'environ 150k€ par an depuis la création de la plate-forme.

RTRA - STAE sur le projet MAELIA (en 2010),

ANRS sur le projet COMOKIT (en 2020),

Agence Française de Développement sur le projet GEMMES (en 2018),

Belmont-Forum *via* l'ANR et AllEnvi sur le projet PREMISS (en 2021), etc.

Niveaux inégalés en 2023 grâce à l'obtention de trois projets de grande ampleur :

STAR FARM, financé par le programme DeSIRA de l'UE et porté par la FAO, comporte une contribution de 1M€ à l'IRD pour construire un simulateur basé sur GAMA et des outils de concertation permettant d'évaluer l'impact sur la biodiversité et l'adaptation au changement climatique du passage à l'échelle de certaines pratique agro-écologiques dans le delta du Mékong vietnamien.

SIMPLE, signé avec la délégation de l'UE auprés de l'ASEAN et qui est porté par l'IRD dans le cadre des *EU-ASEAN Green Partnerships*, porte sur une aide de 2,5M€ pour construire des environnements de réalité virtuelle couplés à GAMA afin de sensibiliser les jeunes de l'ASEAN aux problématiques environnementales.

Dans le cadre du PEPR FairCarbon et porté par l'INRAE, va financer le développement de la version 2.0 de la plate-forme sur les deux prochaines années, au travers de deux attributions, l'une de 250k€ à l'IRD, l'autre de 150k€ à l'IRIT. Le projet a comme objectif, *via* une approche de modélisation et d'évaluation intégrées, de construire et de comparer des scénarios de bioéconomie circulaire verte visant à l'atteinte de la neutralité carbone en 2050.

Moyens, ressources

Le service de valorisation a rapidement été un appui.

En 2014, le CVT SUD avait repris le dossier afin de l'instruire et d'identifier un modèle de valorisation. Cette première réflexion a mené au schéma envisagé aujourd'hui de distinguer la valorisation de la plateforme de celle de ses modèles.

En 2015, le service a procédé au dépôt de la marque GAMA PLATEFORM.

En 2019, une prestation avec la société INNO3 a été menée par le service afin d'identifier la ou les stratégies envisageables pour l'outil.

En 2023, le service appui l'outil pour l'obtention d'un financement interne à l'IRD (Action d'Amorçage Innovation) afin de reprendre les réflexions sur la pérennisation de la plateforme. La stratégie de marque et logiciel doit être précisée et réellement mise en œuvre à l'aide d'un cadre, et une entité tierce doit être identifiée pour reprendre l'animation de la communauté.

CMCDOT

Objet du logiciel

CMCDOT : Conditional Monte Carlo Dense Occupancy Tracker

Système de filtrage Bayésien de grilles d'occupation dynamiques, permettant d'estimer parallèlement au niveau de chaque cellule d'une grille les probabilités d'occupation, d'inférer les vitesses, prédire les risques de collision et associer les cellules appartenant à un même objet dynamique.

Le système est composé d'un paquet ROS, pour gérer la connectique des entrées/sorties, qui encapsule le cœur de l'application embarquée et optimisée sur GPU Nvidia (code Cuda), permettant une analyse temps réel de l'environnement direct sur cartes embarquées (Tegra X1, X2).
ROS (Robot Operating System) correspond à un ensemble d'outils informatiques open source permettant de développer des logiciels pour la robotique. Développées dans un cadre automobile, ces techniques peuvent être exploitées dans tous les domaines de la robotique mobile, et sont particulièrement adaptées à la gestion d'environnement hautement dynamiques et incertains (exemple : scenario urbain, avec piétons, cyclistes, voitures, bus, etc.)

Création de valeur

1.l'évolution du projet, qui a commencé par intégrer la perception embarquée sur des cartes de calcul, puis a développé toute la chaîne de navigation et de contrôle du véhicule, en utilisant différentes plateformes expérimentales et en intégrant de nouveaux capteurs et algorithmes ;

2.les enjeux de propriété intellectuelle liés au projet, qui a généré plusieurs brevets et licences logicielles, ainsi que les modalités de gestion de la PI dans les contrats de collaboration de recherche ;

3.les perspectives d'avenir du projet, qui incluent la possibilité de créer une jeune pousse.

Moyens, ressources

4.Les ressources humaines employées pour le projet logiciel sont principalement des ingénieurs de recherche, dont deux permanents et trois contractuels

5.Le service support est assuré par le CEA, qui est le porteur administratif de l'IRT, et par le service de partenariat et innovation d'Inria

6.Les ressources techniques comprennent la base logicielle de perception embarquée, qui est composée de plusieurs briques logicielles, ainsi que les plateformes expérimentales, qui sont des véhicules ou des robots équipés de capteurs et de cartes de calcul.

Captation de valeur

Les sources de financement du projet logiciel sont principalement l'IRT NanoElec, qui est un consortium entre industriels et académiques sur la région de Grenoble, et qui finance des ingénieurs de recherche, des plateformes expérimentales, et des licences de brevets et de logiciels.

De plus, le projet logiciel bénéficie aussi de contrats associés avec des industriels qui ne sont pas membres fondateurs de l'IRT, mais qui sont intéressés par la technologie développée. Ces contrats permettent de valoriser la base logicielle auprès de partenaires industriels.

CGAL

Objet du logiciel

CGAL : Computational General Algorithms Library

Bibliothèque logicielle de calcul géométrique écrite en C++ visant à fournir une bibliothèque de composants modulaires et fiables de bas niveau pour le calcul géométrique

Ces composants sont destinés à être intégrés dans des logiciels industriels ou des plateformes de recherche, dans des domaines variés comme la robotique, la CAO, la simulation, la modélisation urbaine, l'astronomie, la modélisation médicale, etc.

CGAL est un projet *open source*, qui existe depuis 1996 et qui mobilise un collectif de scientifiques issus d'Inria et d'autres institutions de recherche, ainsi que la *spin-off* d'Inria GeometryFactory, qui se charge depuis 2003 de la commercialisation, du support technique et de l'amélioration des composants.

CGAL est sous double licence (libre et propriétaire).

Création de valeur

En interne,

le projet CGAL a permis d'obtenir des financements de sources variées, comme des ERC, des projets ANR, des projets européens, des contrats avec des partenaires industriels, etc.

- CGAL a aussi permis de développer une expertise reconnue dans le domaine de la géométrie algorithmique et du traitement numérique de la géométrie, avec des publications de haut niveau et des collaborations avec des chercheurs du monde entier, (académiques et industriels).
- L'équipe de recherche a reçu le prix qui récompense un article scientifique marquant lors du symposium international SoGG 2023.
- CGAL a également permis de former des doctorants, des post-docs, des ingénieurs et des chercheurs qui ont contribué à la maintenance, à l'évolution et à la qualité de la bibliothèque.
- Des redevances venant de l'exploitation par la *spin-off* GeometryFactory.

En externe :

- le projet CGAL a permis la création de valeur économique avec la spin-off d'Inria GeometryFactory, qui a pris en charge le release management, le support technique, la commercialisation et l'amélioration des composants.
- La spin-off prospère et emploie aujourd'hui 8 personnes, a été récompensée au SoGG 2023
- GeometryFactory a aussi participé à des projets de transfert technologique avec des grands groupes comme Google, Apple, BMW, etc.

Moyens, ressources

- Des ressources humaines : il y a une vingtaine de développeurs, principalement à Inria, qui contribuent à la maintenance, à l'évolution et à la qualité de la bibliothèque. Il y a aussi des doctorants, des post-docs, des chercheurs et des ingénieurs qui travaillent sur des projets de recherche liés à CGAL, en collaboration avec des partenaires académiques et industriels.
- Des ressources de service support : il y a une *spin-off* d'Inria, GeometryFactory, qui a pris en charge le support technique, la commercialisation et l'amélioration des composants.
- La *spin-off* intervient sur la gestion de projet, sur la formation et le conseil pour les clients qui utilisent CGAL.
- Des ressources techniques : une plateforme dédiée permet de tester les composants sur 45 plateformes différentes chaque nuit.

Il y a aussi des dépôts publics de formes 3D qui servent de corpus pour évaluer les performances des algorithmes.

Captation de valeur

Le projet CGAL a été financé par différentes sources, selon les périodes et les objectifs :

- Au départ, CGAL a été financé par un projet européen, suite à un White Paper de la communauté mondiale de géométrie algorithmique, qui appelait à rendre les algorithmes géométriques disponibles pour l'industrie.
- Ensuite, CGAL a été soutenu en interne par Inria, ce qui a permis de recruter des développeurs, des doctorants, des post-docs, des chercheurs et des ingénieurs qui ont contribué à la maintenance, à l'évolution et à la qualité de la bibliothèque.
- Plusieurs sources de financements variés : européens, nationaux, industriels, bourse ERC
- En 2003, valorisation économique par la *spin-off* d'Inria, Geometry Factory

Depuis, CGAL bénéficie aussi des revenus générés par la vente des licences des composants à des clients industriels

Bibliographie

Rapports :

- MESR 2023-2024. « Production et valorisation des logiciels issus de la recherche publique française », (nov. 2024) : https://www.enseignementsup-recherche.gouv.fr/sites/default/files/2024-12/rapport-sur-la-production-et-la-valorisation-des-logiciels-issus-de-la-recherche-publique-fran-aise-35553.pdf
- D. Le Berre, J.-Y Jeannas, R. Di Cosmo, F. Pellegrini. « Forges de l'ESR – Définition, usages, limitations rencontrées et analyse des besoins », Rapport du collège codes sources et logiciels, (mai 2023) : https://www.ouvrirlascience.fr/forges-de-lesr-definition-usages-limitations-rencontrees-et-analyse-des-besoins/
- UNESCO, « Recommandations de l'UNESCO sur une science ouverte », (2021) : https://doi.org/10.54677/LTRF8541
- T. Gomez-Diaz, T. Recio. « Articles, software, data: An Open Science ethological study », 3 (4), (2024) : https://mapletransactions.org/index.php/maple/article/view/17132
- O. Gadet. et A. Brunner. « Modèles de valorisation socio-économique des productions numériques issues de la recherche publique dans un contexte de Science Ouverte », INRAE Transfert, (2023) : https://hal.inrae.fr/hal-04279988v1

Droit des logiciels et des données :

- F. Pellegrini, S. Canevet, M. Rocard. Droit des logiciels : Logiciels privatifs et logiciels libres, Édition PUF, 2014.
- M. Clément-Fontaine. L'œuvre libre, Larcier, 2014.
- M. Clément-Fontaine, « L'accès aux connaissances logicielles : enjeux et perspectives pour l'Intelligence artificielle et la science ouverte », Propriété industrielle, 2024, les revues LexiNexis, 4 (4), pp.9-13 : https://hal.science/hal-04884965/document

Notion de logiciel :

- F. Pellegrini, R. Di Cosmo, L. Romary, S. Granger, S. Hodencq, J. Janik, R. Coutanson, M. Géroudet. « Livret codes et logiciels », Passeport pour la science ouverte, (2022) : https://www.ouvrirlascience.fr/wp-content/uploads/2022/10/Passeport_Codes-et-logiciels_WEB.pdf

Notion de logiciel de recherche et de développement logiciel :

- M. Clément-Fontaine, R. Di Cosmo, B. Guerry, P. Moreau, F. Pellegrini. « Note d'opportunité sur la valorisation des logiciels issus de la recherche ». Rapport de recherche, Comité pour la science ouverte, (2019) : https://hal-lara.archives-ouvertes.fr/hal-03606374
- G. Courbebaisse, B. Flemisch, K. Graf, U. Konrad, J. Maassen, & Raphael Ritz. "Research software lifecycle" (2023) : https://zenodo.org/records/8324828
- B. Fitzgerald et K.-J Stol. "Continuous software engineering: A roadmap and agenda." Journal of Systems and Software, volume 123, (2017).
- F. Pellegrini, « Qu'est-ce qu'un logiciel de recherche ? », Revue Lamy Droit de l'immatériel, (2024).
- M. Clément-Fontaine. « La Science ouverte (Partie 1) : les logiciels et codes sources de recherche », Revue Lamy Droit de l'Immatériel, nº 210, (2024).

Notion de licences libres :

- M. Clément-Fontaine, « L'œuvre libre », Jurisclasseur PLA fascicule 1972, 2024.
- J. Nicolas, R. Viseur. « Les stratégies open-sources selon le paradigme des modèles économiques. » Systèmes d'information et management, 26.3 (2021): 67-103.
- K.-J Stol, B. Fitzgerald. "Inner source--adopting open source development practices in organizations: a tutorial », IEEE Software, 32.4 (2014): 60-67.

- K.-R Lakhani, E. Von Hippel. « How open source software works:“free” user-to-user assistance », Gabler Verlag, (2004).
- G. Von Krogh, E. Von Hippel. « Special issue on open source software development », Research Policy 32.7, 1149-1157 (2003).
- D. Foray, S. Thoron, J-B Zimmermann. « Open software: Knowledge openness and cooperation in cyberspace », (2007).
- A. MacCormack, J. Rusnak, and C. Y. Baldwin. « Exploring the structure of complex software designs: An empirical study of open source and proprietary code », Management Science 52.7 (2006):1015-1030.
- A. Mockus, R. T. Fielding, J. D. Herbsleb. « Two case studies of open source software development: Apache and Mozilla », ACM Transactions on Software Engineering and Methodology (TOSEM) 11.3, (2002):309-346.

Pour aller plus loin :

- G. Bilder, J. Lin, C. Neylon. « Principles for Open Scholarly Infrastructures », (2015).
- C. Gruson-Daniel. « Science ouverte: Comment exploration des différentes définitions de la science ouverte nous plonge dans les coulisses de la fabrique de la science à l’ère numérique ? », Petit Dico Critique du Big Data, sous la direction de A. Théviot, FYP, (2023).
- D. Foray, « The Economics of Knowledge » https://doi.org/10.7551/mitpress/2613.001.0001 ISBN (electronic) 9780262272926, Publisher: The MIT Press, Published: 2004
- M.-A Cusumano, « The business of software: What every manager, programmer, and entrepreneur must know to thrive and survive in good times and bad », Simon and Schuster, (2004).
- G. Courbebaisse, B. Flemisch, K. Graf, U. Konrad, J. Maassen, & R. Ritz. « Research software lifecycle », (2023):https://zenodo.org/records/8324828
- W.-M Cohen, D.-A. Levinthal. « Absorptive capacity: A new perspective on learning and innovation », Administrative science quarterly 35.1, (1990):128-152.
- C. Y. Baldwin, K. B. Clark, Managing in an age of modularity. Managing in the modular age: Architectures, networks, and organizations (2003) ,149, 84-93.
- S. O'Mahony, F. Ferraro. "The emergence of governance in an open source community.", Academy of Management Journal 50.5 (2007) : 1079-1106.

Remerciements

- La rédaction de l'introduction (Contexte) a bénéficié du travail réalisé dans le cadre d'une réponse à l'appel « science ouverte » 2024 de l'ANR (C. Gruson-Daniel *et al.*) : Grusson-Daniel *et al.* (2024). Projet RECOVAS: REcherche sur la production du COde et sa Valorisation dans le cadre de la Science ouverte. Réponse à l'appel ANR RESO édition 2024.
- Nous remercions nos relecteurs pour leurs remarques avisées et leur aide dans la construction de ce rapport.
- Nous remercions tous les porteurs de projet logiciel qui ont accepté d'être interviewés, y compris ceux dont le projet n'a pas été intégré dans l'annexe de ce rapport. Les échanges avec ces porteurs ont grandement contribué à l'analyse faite dans ce rapport.