Sources ouvertes et architecture : vers une pratique numérique libre

Philippe Marin, Ahmed W. Ismail

Sources ouvertes et architecture : vers une pratique numérique libre

Ahmed W. Ismail[1,2], Philippe Marin[1]

[1] Univ. Grenoble-Alpes, ENSAG*, MHA, 38000 Grenoble, France
* School of Architecture Univ. Grenoble-Alpes
ahmed.ismail@grenoble.archi.fr
philippe.marin@grenoble.archi.fr

[2] Laboratoire CNPA, École polytechnique fédérale de Lausanne

Résumé : Cet article pose la question sur la place de l'*open source*[1] dans le projet architectural. Il s'y intéresse comme mouvement et en retrace les origines. Une attention particulière est portée aux concepts fondateurs du mouvement et à ses valeurs principales. L'*open source* est étudié en tant que modèle éthique et professionnel du développement logiciel. Le cas particulier de l'*open source* en architecture est analysé, dans le cadre de la transition numérique de la profession et du passage aux méthodes BIM. Finalement, son enjeu et son potentiel sont mis en évidence. L'objectif est de trouver dans l'*open source* une alternative aux pratiques courantes, pour une pratique numérique libre.

Mots-clés : Interopérabilité, *open source*, logiciel libre, BIM

Abstract : This article questions the place of Open Source in the architectural project. It is interested in the Open Source movement and traces its origins. A particular attention is paid to the founding concepts of the movement and its main values. The Open Source is studied as an ethical and professional model of software development. The special case of Open Source in architecture is analysed, as part of the digital transition of the profession and the transition to BIM methods. Finally, the Open Source issue and its potential are highlighted. The objective is to find in the Open Source an alternative to current practices, for a free (libre) digital practice.

Key-words : Interoperability, Open Source, Free Software, BIM

1 Introduction

Les outils numériques prennent une place toujours plus importante dans la pratique architecturale. Cela s'inscrit dans l'évolution des techniques numériques et dans la généralisation de l'usage du BIM, et constitue la transition numérique. Si ces techniques sont porteuses de promesses, elles s'accompagnent également de risques. Ainsi, nous souhaitons préciser dans cette contribution les modalités de la souveraineté numérique en considérant nos relations aux droits d'auteur, à la propriété des données et des outils que les architectes mobilisent. Les systèmes numériques propriétaires constituent aujourd'hui les environnements numériques dominants au sein des agences d'architecture. Ce caractère propriétaire concerne à la fois le logiciel et le format des fichiers. Il interroge la liberté de l'usager, la protection de la donnée et les possibilités d'interopérabilité. L'*open source* représente un modèle alternatif aux

[1] Nous utiliserons « *open source* » sans trait d'union pour parler du mouvement, et « *open-source* », avec un trait d'union, en tant qu'adjectif emprunté à l'anglais.

écosystèmes logiciels propriétaires fermés. L'expression « *open source* » regroupe à la fois le modèle de développement du logiciel et un type de licence (Meeker, 2020).

L'objectif de cet article est de préciser les caractéristiques des approches ouvertes et d'envisager plus spécifiquement ces possibilités dans le domaine de la conception architecturale.

Ainsi, nous commençons par la caractérisation de la démarche « *open-source* ». Cela passe par l'étude chronologique des origines du mouvement, ainsi que par la définition de ses composantes. Dans un second temps, nous nous attarderons sur le domaine particulier de l'architecture, en rappelant brièvement l'émergence des technologies numériques dans ce domaine, puis en examinant les caractéristiques de la structure des outils logiciels libres. Dans une dernière partie, nous prendrons du recul pour envisager les perspectives et les transformations des processus de conception à l'œuvre.

2 Code source ouvert *vs* logiciel libre

L'expression « *open source* » porte à confusion. D'une part, elle renvoie à la fois à un mode de développement et à un type de licence ; d'autre part, elle se confond avec le concept de logiciel libre. Ainsi, la distinction doit être faite entre l'« *open source* », littéralement « code source ouvert » et le logiciel libre communément appelé « *free software* ».

2.1 Perspectives historiques

L'un des premiers projets « *open source* » est le système d'exploitation UNIX développé par Bell Labs dans les années 1970. Pour des raisons juridiques liées à la concurrence et associées à l'existence d'un décret antitrust, AT&T, propriétaire de Bell Labs, se voit interdite la commercialisation de produits informatiques. La société décide alors de rendre son système accessible et ouvert au public. Ainsi, UNIX gagne en popularité et principalement au sein du milieu académique. Quelques années plus tard et après la levée en 1983 du décret précédent, AT&T revoit son modèle et envisage une approche propriétaire et fermée pour les versions suivantes. Le mouvement du logiciel libre naît en réaction à ce revirement (Meeker, 2020). Richard M. Stallman, à l'époque ingénieur au Massachusetts Institute of Technology (MIT), débute le projet GNU et lance le mouvement.

L'acronyme GNU se prononce « gnou » et signifie récursivement *GNU Not Unix* (GNU, 2022). Stallman est considéré comme l'un des pères fondateurs du mouvement (Eghbal, 2020). Sa contribution se poursuit avec la mise en place en 1985 de la « fondation du logiciel libre » (*Free Software Foundation*, FSF). Cette dernière se charge de soutenir et de défendre les logiciels libres et de financer le projet GNU. Le mouvement promeut la défense de quatre libertés fondamentales : l'accès au code source, le pouvoir de l'étudier, de le modifier et de le diffuser. L'enjeu du logiciel libre ne repose pas sur des questions de gratuité, mais plutôt de liberté.

En 1991, Linus Torvalds, à l'époque un jeune étudiant, développe le noyau LINUX, qui sera intégré au projet GNU pour donner naissance au système opérationnel GNU/Linux. Les travaux de Torvalds ont inspiré Eric Raymond, troisième figure clé du mouvement *open source*. Eric Raymond considère l'importance de la communauté des contributeurs. Il formalise ses idées dans son célèbre article intitulé « La Cathédrale et le Bazar » (Raymond, 1999). Il énumère les avantages de ce mode de développement dans lequel l'usager est associé au processus de développement. La perspective de Raymond est accueillie plus favorablement par les entreprises privées, qui la préfèrent aux idées de Stallman et du mouvement *Free Software*.

2.2 Visions et philosophie

L'*open source* n'est pas un nouveau paradigme, mais plutôt un retour au modèle initial. En effet, avant la standardisation des ordinateurs personnels et des systèmes d'exploitation, la livraison des logiciels se faisait par défaut en code source. La livraison de logiciels sous forme de fichiers binaires, compilés et prêts à être exécutés, est une pratique récente dans l'histoire de l'informatique.

L'accès au code source permet non seulement d'utiliser le logiciel, mais aussi de l'étudier, de le modifier et de le redistribuer. On retrouve ici les quatre libertés fondamentales défendues par le mouvement *Free Software* (GNU, 2022). Il convient de préciser que l'adjectif *free* signifie aussi bien « libre » que « gratuit ». Cette ambiguïté de la langue anglaise amène les adeptes du mouvement *Free Software* à insister sur le fait que *free* se réfère à *freedom* et non pas à *free beer*. La transposition au français serait de souligner la notion de liberté comme dans « liberté d'expression » et non pas comme dans « entrée libre ».

Alors que le mouvement du logiciel libre défend surtout les quatre libertés fondamentales, Raymond fait l'éloge du travail collaboratif spontané en développement logiciel, à l'image d'un bazar et contrairement à celle d'une cathédrale. Par exemple, il avance la loi Linus. Celle-ci repose sur l'implication des usagers dès le début du développement, par des publications précoces et fréquentes de nouvelles versions, sans se soucier des éventuels bugs. Ceux-ci vont être rapidement identifiés par la large communauté des usagers. L'« *open source* » se distingue alors du logiciel libre en mettant l'accent sur l'accessibilité du code source et l'avantage de ce mode de développement, tandis que *Free Software* défend des valeurs éthiques et philosophiques.

Il y a donc deux raisons principales qui ont fait la prévalence du terme « *open source* » sur celui de « logiciel libre » : (i) la confusion linguistique en anglais sur l'adjectif *free*, (ii) le pragmatisme du mode de développement en *open source*. Cette distinction entre les deux approches, philosophique et pragmatique, est intrinsèquement liée aux types de licences possibles pour les logiciels. Il nous faut distinguer les licences strictes, dites *Copyleft*, et les licences permissives.

2.3 L'*open source* comme licence

Les différents types de licences disponibles reflètent les disparités entre les approches. Le mouvement GNU, fondé par Stallman, propose la gamme de licences *General Public License* (GPL), qui sont des licences dites *copyleft*. À l'opposé, des licences permissives existent, telles que MIT ou BSD. Entre ces deux approches, il existe des licences moins permissives, souvent qualifiées de « *copyleft* faibles » (*Weak Copyleft*), à l'instar des licences Mozilla ou Eclipse.

Le terme *copyleft* semble s'opposer au terme *copyrights* (« droits d'auteur »). Mais c'est sur la base même de ces derniers que le *copyleft* fonde son droit à imposer des conditions restrictives (Dusollier, 2017). L'objectif de ce type de licence est de garantir que le code source transmis sous cette licence ne fasse pas partie d'un logiciel propriétaire fermé, et ainsi de permettre la diffusion du modèle du logiciel libre et sa survie. Les efforts du projet GNU et de la fondation FSF ont permis l'application des termes des licences *copyleft* dans le cas de disputes juridiques (Meeker, 2020). Techniquement, sur le plan juridique, les obligations dictées par une licence *copyleft* sont déclenchées par la distribution d'un produit dérivé.

En revanche, du fait des conditions contraignantes de ce type de licences, plusieurs entreprises ou entités privées hésitent à utiliser des solutions publiées en *copyleft* dans leurs produits commerciaux. C'est pourquoi l'approche de Raymond, renforcée par l'existence de licences plus permissives, a joué un rôle important dans l'adoption élargie de l'*open source* par le secteur privé. Les fondations Mozilla, soutenant le développement du *browser* Firefox ou encore du client de courriel Thunderbird, et Eclipse ont instauré les licences MPL et EPL. Elles sont reconnues comme étant des licences *copyleft* adaptées aux activités commerciales. De la même manière, la fondation Apache développe une licence plus permissive, de type *copyleft* faible. Les licences MIT et Berkeley (BSD) sont presque identiques, et sont les licences les plus permissives. Ces dernières ont très peu d'exigences, et leur exigence se limite

à la distribution de notices. Elles servent toutefois à protéger et à désengager le développeur du code de toute responsabilité.

Par ailleurs, malgré les restrictions imposées par les licences *copyleft*, leur intégration dans des produits propriétaires n'est pas impossible, tout en respectant les termes de la licence. Cela est le cas des licences LGPL, la lettre L au début pour *Limited*. Elle est souvent connue comme la licence à utiliser pour publier une librairie de code qui peut être dynamiquement liée et redistribuée dans un autre produit logiciel, sans être considérée pour autant comme produit dérivé. Sinon, même pour une licence restrictive comme la GPL classique, la mise sur serveur du produit dérivé d'un code GPL n'oblige pas à délivrer le code source, car il n'y a pas de distribution de code dérivé. C'est notamment le cas du modèle de logiciel en tant que service, connu comme *Software As A Service (SAAS)* en anglais. La licence AGPL, quant à elle, est encore plus stricte et exige toujours la publication du code sans exceptions.

Ainsi, il existe un large éventail de licences dans l'*open source*. La figure 1 organise les différentes licences selon leur degré de permissivité et l'octroi de brevet. La diversité des licences reflète les différentes approches idéologiques et pragmatiques des mouvements *open source*.

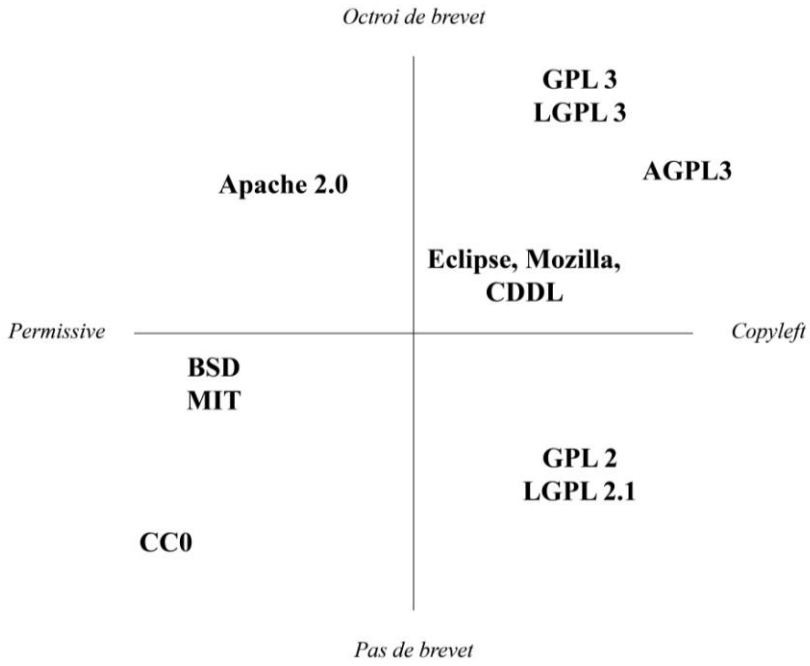

Figure 1 - Types de licences en open source *en fonction de la permissivité et de l'octroi de brevet (Meeker, 2020), traduction et adaptation de l'auteur.*

2.4 L'*open source* : une relation entre usagers et contributeurs

Avec le développement de l'internet et des plateformes de collaboration en ligne, l'*open source* prend un tournant social et communautaire. En 2005, Linus Torvalds contribue encore une fois à l'*open source* par un développement phare. Il produit GIT, le système de gestion de versions (*Versionning* en anglais) décentralisé, très populaire dans le monde de l'informatique pour sa simplicité et son efficacité (Chacon, 2009). Lui-même *open-source*, GIT s'impose comme outil de collaboration entre développeurs, facilitant le travail à distance et de manière asynchrone.

Le développement de GitHub commence en 2007 et il est lancé en 2008. Il connaît une expansion rapide grâce à son adoption du mode GIT et une interface fluide, ce qui lui permet de remplacer rapidement son prédécesseur SourceForge, et de devenir la plateforme

d'hébergement de codes la plus populaire (Eghbal, 2020). Ce succès se confirme malgré son acquisition en 2018 par Microsoft. Cette situation remet en cause sa crédibilité comme le lieu virtuel de collaboration autour de l'*open source*, n'étant lui-même ni libre ni ouvert.

C'est sa facilité d'usage et la présence d'outils techniques et sociaux pour collaborer qui contribuent à sa popularité. Il permet de publier son code, de faire des fiches de reports de bugs (*issues*), d'organiser le travail en équipe et de suivre l'évolution du code sur les différentes branches. Il facilite l'organisation des relations entre les développeurs d'un projet *open-source* et ses utilisateurs[2].

On parle de mainteneurs, de contributeurs et d'utilisateurs. Eghbal développe un modèle de classification des projets *open-source* en fonction du nombre de contributeurs par rapport au nombre d'utilisateurs. Selon ce modèle, il existe quatre types de projets : (i) les jouets, (ii) les clubs, (iii) les fédérations, (iv) les stades (Eghbal, 2020). Le premier type est celui d'un projet personnel de taille réduite. Les clubs ont un nombre important de contributeurs par rapport au nombre d'usagers. Inversement, les stades sont des projets à grand nombre d'usagers pour un nombre limité de contributeurs. Finalement, les fédérations représentent le cas de projets avec un grand nombre d'usagers et de contributeurs à la fois. La figure 2 illustre cette classification.

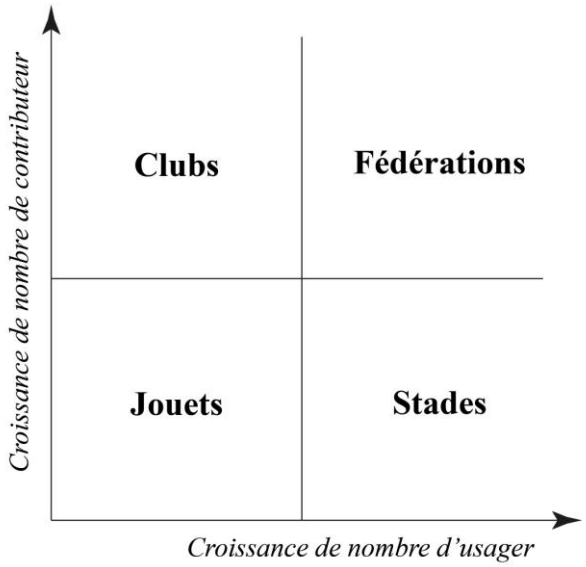

Figure 2 - Classification de projets open-source *en fonction du nombre d'usagers et de contributeurs (Eghbal, 2020), traduction et adaptation de l'auteur.*

3 L'*open source* des outils de conception architecturale

Les outils logiciels prennent une place centrale dans l'exercice du métier d'architecte et dans l'élaboration du projet, de la conception à la réalisation. La transition numérique se déroulent progressivement, mais la question de la liberté de l'usager et de son indépendance vis-à-vis des outils propriétaires émerge.

3.1 L'émergence du numérique pour la conception architecturale

La transition numérique trouve ses racines dans les traités d'architecture avec l'émergence de la position d'architecte qui devient le concepteur de l'ouvrage, qu'il décrit à l'aide de

[2] Utilisateurs de code, par opposition aux utilisateurs du produit fini connus en anglais comme *end users*.

dessins et de plans pour permettre sa réalisation (Carpo, 2011). Si la mesure et la numérisation accompagnent l'histoire des techniques de représentation, le XX^{e} siècle voit l'émergence des outils numériques. Ceux de conception assistée par ordinateur (CAO) et le dessin assisté par ordinateur (DAO) apparaissent. Dans un premier temps, le passage à la DAO a permis un gain de temps et d'efficacité. Puis, inspirés par les avancées dans les domaines de l'automobile et de l'aéronautique, les logiciels ont permis dans un second temps la manipulation des représentations mathématiques avec les courbes de Bézier, les Splines et les NURBS. Ils ont rendu possibles les activités de modélisation en trois dimensions ainsi que les relations à la fabrication (Cardoso Llach, 2015).

La puissance des outils numériques a permis aux architectes de concevoir de nouvelles formes et de les représenter à l'aide du dessin assisté par ordinateur. La conception paramétrique permet de représenter des objets géométriquement complexes, et aussi d'explorer l'espace des solutions. C'est l'invariant par variation formulé par Deleuze et Bernard Cache (Deleuze, 1988). Un premier virage numérique est lié à l'adoption de la CAO, de la DAO et de la FAO (Fabrication Assistée par Ordinateur). Le second virage numérique est la conséquence de la place centrale que prend la donnée (Carpo, 2017). Eastman développe le *Building Description System* (Eastman *et al.*, 1974), rebaptisé *Building Information Modelling* (BIM). Il constitue une réponse aux faiblesses de la DAO et de la CAO pour répondre aux besoins de la conception architecturale. Dans les années 1980, les logiciels appliquant des concepts BIM se développent, principalement chez les éditeurs du groupe Nemetschek (Bolpagni, 2022). En 1997, le format IFC est proposé par l'International Alliance for Interoperability (AIA), consortium qui deviendra BuildingSmart en 2005. Le groupe Autodesk entre dans le jeu du BIM avec l'achat de Revit en 2002. Le groupe acquiert une position monopolistique dans le secteur de la construction avec sa politique de croissance par fusion et acquisition de sociétés (Chen & Ho, 2021).

Les avancées technologiques facilitent la conception de formes de plus en plus libres et complexes, tout en manipulant des volumes importants de données. La conception paramétrique facilite l'exploration et la variabilité du modèle en maintenant l'associativité et les relations entre les composants de la géométrie. Les approches computationnelles et le BIM (de Boissieu, 2020) constituent les principales modalités d'instrumentation des processus de conception. Le BIM est la seule technologie conçue spécifiquement pour le secteur de la construction (Bernstein, 2018). Cependant, dans ce domaine, la majorité des outils utilisés sont propriétaires, et les principes et usages de l'*open source* restent marginaux.

3.2 Un regard sur d'autres secteurs d'application

Le secteur de la construction a majoritairement mobilisé les technologies inventées pour d'autres usages. En effet, il est considéré comme l'un des secteurs les plus en retard dans sa transition numérique (Poinet, 2020). Nous prendrons ici les exemples du secteur industriel, avec sa relation à la fabrication et la préfabrication, ainsi que le secteur de la géomatique avec la mise au point des systèmes d'information géographique (SIG), avec la gestion de grandes masses de données et la prolifération des pratiques ouvertes.

La transition numérique de l'industrie se caractérise par (i) un support à l'innovation et des investissements importants dans la recherche et le développement (R&D) des outils informatiques[3], (ii) une forte standardisation des formats de données, (iii) la gestion intégrée de l'ensemble de la donnée du produit, (iv) un processus associatif du suivi du produit de la conception à la fabrication.

Sur le premier point, l'industrie accorde de l'importance aux instrumentations numériques des processus, ce qui favorise des développements adaptés aux besoins spécifiques du secteur. De plus, l'industrie est familière de l'optimisation des tâches, et la taille importante des entreprises favorise des investissements massifs dans les outils numériques (Aram &

[3] Les courbes de Bézier sont le résultat des travaux de Paul de Casteljau et de Pierre de Bézier, travaillant pour Citroën et Renault respectivement. Ces travaux ont inspiré les développements des B-Spline dont les NURBS de nos jours découlent (Townsend, 2014).

Eastman, 2013). Le secteur a rapidement travaillé à la standardisation des formats de fichiers en proposant le format STEP (*Standard for Exchange of Product Data Model*), mis en place suite au développement du langage de modélisation EXPRESS au sein d'un comité ISO (Schenck & Wilson, 1994). Le format STEP a rencontré un large succès et s'est popularisé au sein de l'industrie. Il servira de base pour le développement de l'IFC.

Outre la modélisation géométrique, la gestion de la donnée joue un rôle important pour le secteur industriel. Les relations entre le produit et ses données sont gérées *via* le *Product Data Management* (PDM), fonctionnalité intégrée à la plupart des logiciels industriels et mécaniques. La gestion de l'évolution du produit et de ses versions, tout au long de son cycle de vie, est réalisée par le *Product Lifecycle Management* (PLM). L'ERP (*Entreprise Resource Planning*) complète le système. La gestion de la donnée est envisagée et formalisée au sein de ces différents modules (Swink, 2006). Les flux vont de la conception du produit aux programmes d'usinage sur machines à commande numérique. Les pratiques numériques de l'industrie intègrent la CAO et la FAO.

Par ailleurs, la place de la donnée est également centrale dans le domaine de la géomatique et des systèmes d'information géographique. Ici, l'exploitation des bases de données est nécessaire pour extraire, traiter, structurer et visualiser des volumes importants de données. L'un des tout premiers logiciels en SIG est GRASS GIS, développé dans les années 1980 par des ingénieurs de l'armée américaine. Son code est accessible, et il a été libéré en tant que bien public. Il constitue un bon exemple de logiciel libre et *open-source*. Si le laboratoire militaire s'est retiré du projet en 1996 sous la pression de développeurs privés, y voyant une ingérence de l'État, GRASS GIS a survécu en tant que projet *open-source*. Une communauté, principalement formée par des universitaires, a pris le relais (Casagrande, 2012). La démarche *open-source* peut alors garantir la pérennité d'un projet, en rendant possible la transmission du projet d'un mainteneur à son successeur, comme le précise Ramond (Raymond, 1999).

L'ampleur du mouvement *open source* dans le domaine de la géographie se prolonge avec la publication de QGIS (autrefois Quantum GIS) dès 2002, puis d'OpenStreetMaps en 2004. Ces outils sont forts de leurs années de développement et de leurs communautés engagées. QGIS se caractérise par sa modularité et la possibilité d'extension *via* les contributions de la communauté sous forme de *plug-ins*. Cela est le résultat non seulement de son caractère *open-source*, mais également de son infrastructure multiplateforme inclusive en Python pour Linux, Windows ou MacOS.

3.3 Les outils *open-source* pour le domaine de l'architecture

À l'inverse de la géomatique, la transition numérique du projet architectural a été largement dominée par des acteurs propriétaires, avec la constitution de monopoles. Cependant, les initiatives *open-source* ne sont pas inexistantes.

3.3.1 Anatomie du logiciel CAO

Une première classification d'un outil *open-source* est technique. Elle se base sur les différentes couches informatiques d'un logiciel de CAO. Il y a : (i) des librairies de codes, (ii) des logiciels indépendants, (iii) des extensions de logiciels existants. La figure 3 montre l'anatomie type d'un logiciel de CAO/DAO, et la figure 4 montre les trois couches informatiques.

Les librairies constituent des boîtes à outils exploitables par d'autres usagers. Une librairie peut être utilisée directement par d'autres usagers ou intégrée dans un logiciel ou une autre librairie. Un ensemble de librairies peut former un cadriciel (*framework* en anglais). Pour se servir d'une librairie de codes, l'utilisateur ou l'utilisatrice doit être en mesure d'écrire du code. L'usage des librairies exige donc un minimum de connaissances en informatique et en programmation. Open Cascade est une librairie *open source*. Elle est employée comme noyau géométrique par des logiciels de modélisation 3D comme FreeCAD. Des librairies spécialisées dans le traitement de l'IFC, dont IFCOpenShell, écrite en C++ et offrant une

interface en Python, sont disponibles et offrent des fonctions d'édition et d'extraction de données de l'IFC.

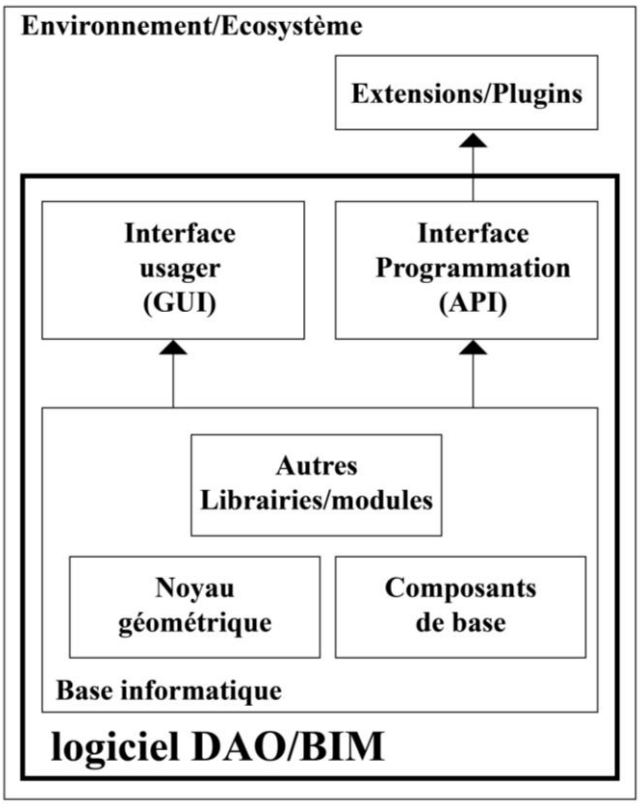

Figure 3 – Anatomie de l'environnement logiciel type en CAO/DAO.

Les librairies constituent les briques élémentaires d'un logiciel *Standalone*. À titre d'exemple, FreeCAD et Blender sont deux logiciels de modélisation 3D. Ils partagent des approches et des valeurs communes, comme la compatibilité inclusive avec les systèmes opérationnels usuels, tels que Linux, Windows et MacOS. Tous les deux s'adossent à une large communauté d'usagers et de contributeurs. La principale distinction est technique et repose sur les modalités de description des géométries. FreeCAD est un modeleur solide, basé sur le noyau Open Cascade. Il intègre les notions de modélisation exacte avec un solveur de contraintes et les représentations de courbes et surfaces NURBS. Blender est un modeleur polygonal, il travaille avec des maillages dans une logique de discrétisation des géométries. Pour modéliser des formes libres, souples et organiques, Blender se sert des algorithmes de subdivision de surfaces de Catmull-Clark.

Les caractéristiques de ces deux logiciels les rendent spécifiques. FreeCAD peut être utilisé pour le design industriel, mais également pour la conception architecturale, car il intègre aussi le standard IFC par le biais de la libraire IFCOpenShell et permet de faire des dessins en 2D et des quantitatifs. Blender se distingue par ses capacités de visualisation, de gestion de texture et des rendus graphiques sous forme d'images de synthèse ou d'animations. Blender est très populaire dans le monde des jeux vidéo et du dessin animé, son usage se répand également dans le domaine de l'architecture. Contrairement à Blender, FreeCAD peine à convaincre. Ce désintérêt pourrait s'expliquer par la culture artistique de la conception architecturale, qui se démarque de la culture mécanique qui domine FreeCAD. De plus, son noyau géométrique, Open Cascade, a ses limites en comparaison avec les autres noyaux. En effet, le marché des noyaux géométriques est dominé par le duopole propriétaires ACIS (Dassault Systèmes) et ParaSolid (Siemens). De même, l'interface graphique d'usager (GUI) de FreeCAD est difficile

d'accès et peu ergonomique, ce qui se traduit par un temps de modélisation plus important par rapport à d'autres outils (Junk & Kuen, 2016). Par ailleurs, Blender n'est pas nativement structuré pour gérer de l'IFC. Mais la récente extension BlenderBIM, basée sur la librairie IFCOpenShell, semble corriger ce défaut.

Les extensions constituent le troisième type de composants *open-source*. Leur développement est possible grâce à (i) la mise en place d'une API (interface de programmation, ou *Application Programming Interface*) ouverte et facilement accessible, (ii) une documentation claire et complète, (iii) une large communauté d'usagers développeurs. Ces « *AddOn* » viennent donc enrichir la version de base du logiciel. C'est le cas des deux logiciels libres précités, mais aussi de QGIS ou encore de logiciels propriétaires qui ont mis en place une API ouverte et ont réussi à développer une communauté de contributeurs. C'est le cas du logiciel Rhinoceros3D, que l'on appellera ici Rhino, qui a fait le choix de favoriser la contribution de sa communauté d'utilisateurs/développeurs par des outils dont un bon nombre sont *open-source*.

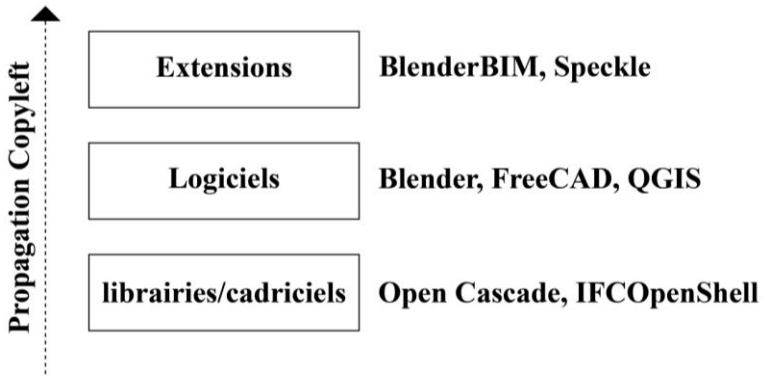

Figure 4 - Sens de propagation de licences copyleft*. L'usage de ces dernières au niveau informatique le plus bas force une continuité* copyleft *dans les niveaux supérieurs.*

Mentionnons également qu'une troisième classification peut être réalisée sur la base du type de licences. Nous observons que les logiciels *standalone* sont souvent distribués sous une licence *copyleft*, comme Blender dont la licence est la GPL, ou FreeCAD dont la licence est la LGPL. Ces logiciels sont non seulement *open-source*, mais également libres. La figure 4 résume les trois niveaux possibles et montre le sens de propagation des licences *copyleft* avec des exemples de logiciels du projet architectural.

3.3.2 Un regard sur l'échange d'informations en *open source*

Pour le traitement graphique, dominé par la suite Adobe, trois logiciels représentent une alternative. Ce type de logiciels est particulièrement important pour la production de documents graphiques en 2D. Sur le marché, il y a : GIMP pour l'édition d'images *raster*, InkScape pour le graphisme vectoriel ou encore Scribus pour la mise en pages. Tous les trois se basent sur l'usage de formats de fichiers ouverts comme le SVG ou le TIFF. Les trois logiciels sont également soumis à la licence *copyleft* GPL. Cela est aussi le cas de CloudCompare, logiciel *open-source* pour le traitement des nuages de points. Il présente également des fonctions de traitement de maillage et offre des outils puissants pour le relevé numérique. Quant à la bureautique, comme l'écriture de documents *text* ou encore les tableurs, il y a OpenOffice et LibreOffice.

D'autre part, Speckle se présente comme une solution au problème d'interopérabilité et de cloisonnement propriétaire (Poinet *et al.*, 2021 ; Stefanescu, 2020). C'est une plateforme *open-source* qui offre une nouvelle solution à la problématique de l'interopérabilité. Speckle propose une approche orientée objet contre une pratique orientée fichier, *via* la mise en place de flux informationnels directs entre les différents outils, par un protocole de communication

de type HTTP passant par un serveur web. Cette interopérabilité directe est possible grâce à l'ensemble des connecteurs développés pour un nombre important d'outils utilisés en architecture. Contrairement à Flux, une solution d'interopérabilité similaire qui l'a précédé et qui a été abandonné en 2018, Speckle est *open-source* et est publié sous la licence Apache, une licence *copyleft* faible assez permissive.

Finalement, l'outil Rhino.inside est également un projet *open-source* et vise à une meilleure interopérabilité entre Rhino et les autres logiciels du marché. Rhino.inside est une technologie qui permet au logiciel Rhino de se lancer dans un logiciel hôte, ce qui établit un dialogue direct entre les APIs du logiciel hôte et Rhino. Ce qui est à noter, c'est que c'est un projet *open-source* mis en place par un développeur privé et distribué sous la licence très permissive MIT. Cela a permis à cette solution de gagner en popularité. À l'instar de Speckle, c'est une solution d'interopérabilité libre et directe entre des outils différents, qui sont souvent propriétaires.

Si dans ce cas la liberté de l'usager n'est pas remplie comme le prévoit le mouvement Free Software, la donnée reste libre et échappe à l'encloisonnement des formats propriétaires.

3.3.3 Des modèles organisationnels différents

Dans la comparaison entre FreeCAD et Blender, ce qui est important à noter, c'est que la popularité du second dépasse largement celle du premier. Cela pourrait s'expliquer par la puissance de sa structure de support qu'est la fondation Blender, basée depuis 2002 aux Pays-Bas et fondée par Ton Rosendaal. FreeCAD, est un logiciel développé principalement par une communauté, ou *community-driven.* Ce modèle, même idéalisé par Eric Raymond, montre ses limites. Le succès que présente Blender aujourd'hui souligne l'importance de la présence d'une structure de soutien puissante. Ce postulat est confirmé par l'exemple précité de GRASS GIS, dont le développement a été possible grâce à l'implication d'un groupe de chercheurs de l'armée américaine, avant sa reprise par une communauté solide.

Un troisième schéma de développement possible est celui de l'implication d'entreprises privées développant des outils *open-source*. Ce cas est connu dans d'autres secteurs, à commencer par celui d'entreprises technologiques comme Sun ou IBM, mais aussi avec Google qui lance le projet Chromium en *open source*, qui donnera lieu au *browser* Chrome ainsi qu'une série de *browsers* basés sur Chromium. Dans le monde du bâtiment, McNeel développe des outils *open-source* comme OpenNURBS, Rhino.Compute ou Rhino.inside. Le lancement d'une entreprise privée dans l'*open source* a bien des avantages, en faisant de leurs produits des standards de marché *de facto*. En effet, la mise en *open source* d'un outil encourage son adoption sur une large échelle (Meeker, 2020). Sur la base de cette analyse, et à part des caractéristiques techniques, nous avançons dans la figure 5 l'hypothèse des facteurs de réussite de projets *open-source*, qui sont : (i) l'engagement de la communauté, (ii) le soutien financier public/privé, (iii) l'existence d'une structure de soutien institutionnelle.

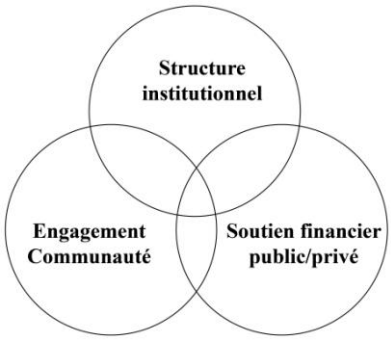

Figure 5 - Facteurs de réussite de logiciels open-source.

3.4 La quasi-absence de l'*open source* dans la pratique des agences d'architecture

Les logiciels intégrant les méthodes BIM sont les seuls conçus spécifiquement pour les professions du bâtiment. La démarche logicielle BIM est paramétrique et orientée objet. Elle permet de mettre en lien la géométrie et les données des différents objets, mais aussi les relations hiérarchiques et sémantiques entre les différents éléments. Les logiciels de ce type existant sur le marché sont majoritairement propriétaires. Revit domine la scène mondiale et constitue un monopole, surtout dans les pays anglo-saxons. ArchiCAD, quant à lui, est présent majoritairement dans une zone géographique limitée à l'Europe continentale, surtout en zone magyaro-germanique (Ulmanis, 2015). Le BIM en *open source* est donc limité aux librairies et aux extensions logicielles. Aucun logiciel autonome et *open-source*, spécifiquement conçu pour le bâtiment, n'émerge.

Cependant, quelques initiatives existent. La proposition de LendLease et Dion Moult présentée sous le titre *A modular toolkit for developing openBIM data pipelines* est une exception rare (buildingSMART, 2021). Elle a remporté le prix BuildingSmart en 2020, dans la catégorie de l'innovation. Elle représente une tentative de mise en place d'un écosystème complet en *open source*, basé principalement sur le standard IFC. C'est un flux de travail avec des échanges de donnée *via* des formats de fichier libres. Le flux emploie des solutions *open-source* existantes et développe de nouveaux outils pour compléter le *workflow*.

Si l'étude des outils *open-source* révèle leur étendue qui couvre une large panoplie de compétences, la réalité du terrain en architecture est contrastée. Dans une enquête menée dans le cadre d'une thèse de doctorat par Élodie Hochscheid sur la pratique numérique des agences d'architecture en France, presque aucun logiciel *open-source* ne figure parmi ceux utilisés. Une exception rare de LibreCAD figure dans 3 réponses uniquement sur 900 (Hochscheid, 2020). L'observation des résultats montre donc que le spectre des logiciels utilisés en architecture reste dominé par les outils propriétaires. L'absence d'outils *open-source* est couplée avec la réticence et le retard, constaté par l'étude, dans l'adoption du BIM par les agences d'architecture. Les obstacles à l'adoption du BIM, relayés par Hochscheid, sont valables également pour expliquer le manque de solutions *open-source* en architecture. Les pratiques actuelles, dominées par des outils CAO propriétaires, nécessiteraient un changement radical dans les processus. Cela est valable pour un changement de méthode de travail pour adopter le BIM ou, simplement, changer d'outils pour des alternatives *open-source* dans les processus en place. De même, le changement passerait par la remise en question de la culture de la profession. L'adoption de solutions *open-source* demanderait potentiellement l'usage du code et la programmation logicielle par les architectes, comme outils de travail. L'usage du code reste une exception dans les agences d'architecture, et son adoption contribuerait à celle de l'*open source* par les architectes.

Pour conclure sur la place des outils *open-source* dans la conception architecturale, le manque de solutions *open-source* dédiées à la conception architecturale peut s'expliquer par plusieurs raisons : (i) le retard généralisé et le faible degré de maturité numérique du secteur du bâtiment, (ii) le manque d'implication des acteurs du bâtiment dans le développement de leurs outils et leur dépendance vis-à-vis d'autres secteurs, (iii) le manque de standardisation et la faible adoption de l'IFC, (iv) la position dominante des éditeurs logiciels propriétaires, (v) l'absence d'instances publiques fortes.

4 Enjeux de l'*open source* pour l'exercice du métier d'architecte

Raymond se sert de l'exemple de la construction des cathédrales comme métaphore pour faire l'éloge de la spontanéité face à l'organisation stricte dans le développement logiciel. Bignon, lui, cite l'exemple des grands chantiers comme ceux des cathédrales et des pyramides pour souligner l'aspect collaboratif et peu codifié qui existait avant la révolution industrielle (Bignon, 2002). Le projet architectural, avec sa complexité et sa transdisciplinarité, demande une collaboration entre les acteurs. L'encloisonnement des écosystèmes logiciels propriétaires

et fermés constituerait un obstacle à l'interopérabilité et à une collaboration efficace. La seule solution réside dans les standards ouverts et les solutions logicielles libres.

4.1 Les enjeux de la standardisation et de l'interopérabilité

Comme avancé précédemment, le secteur de la construction est peu standardisé. Le processus de standardisation est courant dans d'autres secteurs. L'exemple le plus connu est celui du World Wide Web et des protocoles http et HTML. La mise en place de standards communs a donné lieu à une interopérabilité qui permet à tout utilisateur d'accéder au contenu web sans exigences particulières sur le type de *browser* ni même le système opérationnel. Les processus de standardisation ont deux volets : un volet technique et un volet institutionnel. Dans le cas du Web, le volet institutionnel est représenté par le consortium W3C.

La littérature scientifique sur le sujet de la standardisation distingue deux types d'entités pour la standardisation : les organisations de développement de standards (ODS) ou les consortiums industriels (Laakso & Kiviniemi, 2012). D'après les mêmes auteurs, les ODS sont considérées comme ouvertes, transparentes et inclusives, mais aussi lentes. Les consortiums industriels sont plus rapides et fonctionnels, mais exclusifs. En ce qui concerne le projet architectural, le processus de standardisation le plus développé est celui de l'IFC. Au départ, l'organisme IAI (rebaptisé BuildingSmart en 2005) envisageait l'IFC comme fermé et limité au consortium de douze entreprises. Ce n'est qu'en 1999 lors d'un sommet à Munich que l'idée d'un standard ouvert et accessible est mentionnée pour la première fois.

Dans les processus de standardisation liés à l'informatique, il y a deux scénarios quant à la temporalité de l'évolution : (i) une standardisation précoce, qui va influencer les produits, (ii) une standardisation tardive, qui ne fera que s'adapter à l'existant (Cargill, 1989). Paradoxalement, même si la standardisation de l'IFC a été assez précoce, son influence sur le développement des outils logiciels a été marginale. L'IFC n'a pas réussi à s'imposer comme standard à implémenter efficacement par les développeurs de logiciels. Il serait important de noter que le groupe Autodesk, un des douze membres fondateurs de l'IAI en 1994, acquiert Revit en 2002, ce qui expliquerait le changement de paradigme au sein du groupe vers un BIM fermé. En revanche, si le format DWG est devenu un standard *de facto* pour la DAO (Björk & Laakso, 2010), le format RVT n'a pas réussi à s'imposer comme standard.

Parmi les outils *open-source* examinés précédemment, l'IFC est l'un des vecteurs de leur développement. Cela est dû à la transparence et à l'ouverture qui ont accompagné le processus de standardisation. La standardisation sans cloisonnement est une garantie d'interopérabilité et un moteur de développement de l'*open source*.

4.2 *Open source* et propriété intellectuelle dans la création artistique

Le terme *copyleft* a été employé pour la première fois par Li-Chen Wang en 1976 (Rauskolb, 1976) pour s'opposer à l'abus d'usage du copyright dans une citation devenue célèbre : *@Copyleft – All wrongs reserved*. Malgré cette opposition apparente entre *copyright* et *copyleft*, ce dernier et tous les mouvements qui en découlent prennent sens par l'application même des règles de la propriété intellectuelle, établies suite à une longue procédure de législation et de jurisprudence inscrite dans le droit américain (Bert-Erboul, 2015). Dans le cas de l'*open source*, un auteur décide de céder ses droits à d'autres sous certaines conditions régies par une licence. L'application des termes de ces licences est garantie par le droit de propriété intellectuelle. La FSF se charge de se pourvoir en justice contre les entités qui violent les exigences des licences *copyleft* (Meeker, 2020).

Le mouvement *open source* a eu des répercussions sur la création originale artistique et littéraire. Le mouvement Free Art, né en France, s'est largement inspiré du mouvement Copyleft et partage les valeurs des licences GPL (Dusollier, 2017). Il part du principe que la création est une œuvre collective de l'humanité. Concrètement, à l'instar de la production de code source ouvert, réutilisable par d'autres usagers dans d'autres créations logicielles, les créateurs de contenu créatif rendent leur contenu libre d'accès et de réutilisation. Les licences

Creative Commons (CC) sont inspirées du mouvement Copyleft. Elles ont toutefois plus de flexibilité et offrent différents types de licences avec divers niveaux de restrictions sur la création de travaux dérivés. Les CC prennent leur essor avec l'arrivée d'Internet et la prolifération de contenus médiatiques, et sont à mi-chemin entre le domaine public et le mouvement Copyleft (Bert-Erboul, 2015).

En termes de production architecturale et computationnelle, l'*open source* libère le concepteur à deux niveaux : il est libre, au sens voulu par le mouvement du logiciel libre, de choisir, d'étudier et d'adapter l'outil informatique dont il se sert ; mais aussi d'avoir la liberté sur le contenu créé et de continuer à avoir accès à son contenu indépendamment de l'outil logiciel. L'émergence de pratiques computationnelles ajoute un type intermédiaire de création originale : les algorithmes. Ceux-ci se positionnent comme un type hybride entre la création artistique et la production algorithmique sous forme de programmation visuelle ou de code source. Une communauté de concepteurs computationnels émerge, où l'open source et le partage sont des pratiques de plus en plus observées. Cela contribue à l'évolution de la culture du métier d'architecte, où la propriété intellectuelle change de signification.

4.3 Un changement de processus et des pratiques nouvelles

En matière d'usage, l'adoption de l'*open source* par les architectes, et les autres acteurs du bâtiment, est motivée par des raisons pratiques liées aux besoins du métier dans son exercice. Les outils *open-source* s'invitent pour résoudre l'épineux problème d'interopérabilité en évitant l'encloisonnement propriétaire. L'argument économique justifie une adoption accélérée et une prise de conscience croissante de l'enjeu de l'*open source*. La pratique des éditeurs de logiciels propriétaires rend les licences de plus en plus chères. Cela s'ajoute au passage progressif à un modèle d'abonnement, où le logiciel est loué. Ainsi, le risque est de ne plus avoir accès à ses propres fichiers (en formats propriétaires) en absence d'abonnement.

Cela rejoint la question précédente de la propriété intellectuelle sur ses propres productions originales. La durabilité de la donnée est donc une question fondamentale. L'usage de formats de fichiers neutres et libres assure une liberté de la donnée. Celui de logiciels libres garantit la liberté de l'usager, résumée dans les quatre libertés fondamentales du mouvement du logiciel libre. Les principaux apports de l'*open source* sont donc : (i) un gain économique en se passant des licences, (ii) une interopérabilité accrue et améliorée, grâce aux formats et standards libres, (iii) une durabilité et une liberté de la donnée et de l'usager.

Les formats neutres et les logiciels libres sont indissociables. Les outils *open-source* manipulent la donnée, *a fortiori* dans des formats libres. Dans le processus de la production architecturale, seuls le standard IFC et le format de fichier qui en résulte sont spécifiquement conçus pour le bâtiment et traitent à la fois d'une géométrie 3D en plus de sa donnée dans une logique BIM orientée objet. Or son usage a été longtemps limité, au profit des formats propriétaires. Certes, les logiciels propriétaires permettent l'export ou l'import de donnée IFC, au moyen d'une traduction du modèle natif (Eastman *et al*., 2018). Les librairies *open-source* de type IFCOpenShell ou IFC.js proposent de travailler avec l'IFC en natif. Cela exige une maîtrise du standard et un minimum de connaissances en programmation.

Finalement, l'hypothèse que nous formulons est que le passage à l'*open source* constituera en soi un troisième virage numérique et un véritable changement de paradigme. L'adoption réussie de l'*open source* jouerait un rôle de catalyseur dans la transition numérique du secteur de la construction, en invitant les architectes à adopter le code comme outil de travail, dans un changement culturel fondamental. Sur le plan technique, le passage aux outils *open-source* et l'implication des architectes dans la configuration des outils apporteront un regard nouveau sur le métier de l'architecte. En matière de transition numérique, l'architecte n'est plus un récepteur passif d'outils, mais un acteur actif dans la confection des outils de son métier. Sur le plan philosophique et éthique, le changement concerne la notion de paternité (*authorship* en anglais) et de propriété intellectuelle, dont les racines remontent à la pratique albertienne (Dusollier, 2017). Des pratiques similaires à celles du mouvement *Free Software* peuvent être

transposées dans la conception architecturale sous la forme de bibliothèques d'objets ou de dessins en libre accès, sous des licences comme la Creative Commons.

5 Conclusion

Les logiciels libres sont le résultat d'un mouvement idéologique. Ce mouvement est propulsé par des exigences à la fois techniques et éthiques. Si le terme *open source* est celui qui est retenu par le grand public, il englobe différents types de produits logiciels. Une première classification montre que les logiciels *open-source* existent sous différentes formes. La distinction par rapport au type de la licence est d'une grande importance, car le type de licence révèle la philosophie sous-jacente. Les licences *copyleft* diffèrent des licences permissives en insistant sur le devoir de publier le code source à chaque distribution de travaux dérivés. La vision permissive défend un modèle où le point fort de l'*open source* reste son côté peu réglementé et basé sur la présence de communautés d'usagers et de contributeurs, qui offrent un mode de fonctionnement alternatif aux entreprises privées, et qui reste très efficace.

Dans la pratique numérique du projet architectural, le retard de la transition numérique est suivi d'un retard dans l'adoption d'outils *open-source* pour la conception et la fabrication. Si les outils *open-source* ont fait leur entrée dans la gamme des logiciels utilisés, ils restent peu nombreux à être conçus spécifiquement pour le bâtiment. Cela peut s'expliquer par le manque de standardisation dans le secteur du bâtiment, notamment pour les processus numériques. De même, s'ajoute à cela le manque d'implication des architectes, en tant qu'acteurs principaux en charge du projet architectural de la conception à la réalisation, dans le développement de leurs outils. La montée des outils BIM et la conception computationnelle créent une nouvelle génération d'architectes : les *superusers* (Deutsch, 2019). Ces derniers s'investissent dans la mise en place de solutions optimisant les flux de travail ou encore apportant une qualité de conception.

Le futur de l'*open source* en architecture se jouera *via* les architectes. Ils et elles devront prendre position pour la liberté de la profession vis-à-vis des outils. Cette liberté concernera les outils, se matérialisant dans les quatre libertés fondamentales prônées par le mouvement *Free Software*, mais aussi dans la liberté de la donnée. Elle est possible grâce aux standards et modèles de données ouverts. Les architectes ont la responsabilité morale d'opter pour des formats et outils ouverts. Or, pour cela, le soutien des autorités publiques est nécessaire. Cela passe par l'exigence de standards ouverts dans les projets publics.

L'étude montre ainsi que pour atteindre les objectifs de liberté, mais aussi d'efficacité et d'interopérabilité, les formats libres sont à privilégier. Pour une transition numérique réussie vers les méthodes BIM, les standards libres comme l'IFC sont la solution. Si les outils propriétaires continuent à renforcer l'encloisonnement logiciel en favorisant les formats propriétaires, les outils *open-source* existent. Leur adoption demandera une évolution de la culture des architectes. Le passage à une vision de l'architecte maîtrisant les outils numériques et le code comme outil de travail accélérera le passage à l'*open source* et au BIM dans l'exercice du métier. L'argument économique renforcera ce passage aux logiciels libres, sans oublier la durabilité de l'accès à ses propres créations originales.

Enfin, si cet article adopte une position favorable à une migration progressive vers l'*open source* et une adoption de sa philosophie, l'objectif n'est pas de remettre en cause le modèle des logiciels propriétaires. Ceux-ci continueront à avoir leur place dans le paysage professionnel. Par contre, les standards ouverts, les API ouvertes et les formats ouverts garantiront une liberté de la donnée, une meilleure interopérabilité et une garantie d'indépendance dans l'exercice du métier.

Références

Aram S. & Eastman C. (2013), *Integration of PLM Solutions and BIM Systems for the AEC Industry*, 30th International Symposium on Automation and Robotics in Construction and Mining, held in conjunction with the 23rd World Mining Congress, Montréal, Canada. https://doi.org/10.22260/ISARC2013/0115;

Bernstein P. (2018), *Architecture, design, data: Practice competency in the era of computation*, Birkhäuser Verlag.

Bert-Erboul C. (2015), « Les Creative Commons. Une troisième voie entre domaine public et communauté ? », *Recherches sociologiques et anthropologiques*, 46 (2), 43‑65. https://doi.org/10.4000/rsa.1514.

Bignon J.-C. (2002), *Modélisation, simulation et assistance à la conception-construction en architecture*, Habilitation à diriger des recherches, Université Henri Poincaré-Nancy I. https://tel.archives-ouvertes.fr/tel-00145570.

Björk B.-C. & Laakso M. (2010), « CAD standardisation in the construction industry. A process view », *Automation in Construction*, 19 (4), 398‑406. https://doi.org/10.1016/j.autcon.2009.11.010.

Bolpagni M. (2022), « Building Information Modelling and Information Management », *in* M. Bolpagni, R. Gavina & D. Ribeiro (dir.), *Industry 4.0 for the Built Environment : Methodologies, Technologies and Skills* (p. 29‑54), Springer International Publishing. https://doi.org/10.1007/978-3-030-82430-3_2.

Boissieu A. de (2020), « Super-utilisateurs ou super-spécialistes ? Cartographie des catalyseurs de la transformation numérique en agence d'architecture », *Cahiers de la recherche architecturale, urbaine et paysagère*, *9|10*. https://doi.org/10.4000/craup.5551.

buildingSMART (2021), *Lendlease Modular Toolkit Case Study*. BuildingSMART International. https://publications.buildingsmart.org/a-modular-tookit-for-developing-openbim-data-pipelines.html.

Cardoso Llach D. (2015), *Builders of the vision. Software and the imagination of design*, Routledge.

Cargill C.F. (1989), *Information technology standardization. Theory, process, and organizations*, Digital Press.

Carpo M. (2011), *The alphabet and the algorithm*, MIT Press.

Carpo M. (2017), *The second digital turn. Design beyond intelligence*, The MIT Press.

Casagrande C. (2012), *GIS open source*, D. Flaccovio.

Chacon S. (2009), *Pro Git*, Apress ; distributed to the book trade worldwide by Springer-Verlag.

Chen J. K. & Ho H.H. (2021), « Transformation and Impact from the Software Ecosystem Perspective. Case Study of Autodesk Inc.'s Ecosystem Roadmap », *2021 IEEE International Conference on Social Sciences and Intelligent Management (SSIM)*, 1‑7.

Deleuze G. (1988), *Le Pli. Leibniz et le baroque*, Éditions de Minuit.

Deutsch R. (2019), *Superusers. Design technology specialists and the future of practice*, Routledge, Taylor & Francis Group.

Dusollier S. (2017), « Open source and copyleft. Authorship reconsidered ? », in *Intellectual Property* (p. 563‑578), Routledge.

Eastman C., Fisher D., Lafue G., Lividini J., Stoker D., Yessios C. *et al.* (1974), « An outline of the building description system », *Research Rep.*, 50, 1974.

Eastman C., Teicholz P.M., Sacks R. & Lee G. (2018), *BIM handbook. A guide to building information modelling for owners, managers, designers, engineers and contractors* (third edition), Wiley.

Eghbal N. (2020), *Working in public. The making and maintenance of open source software* (first Edition), Stripe Press.

GNU (2022), *What is Free Software ? - GNU Project—Free Software Foundation*. https://www.gnu.org/philosophy/free-sw.en.html.

Hochscheid E. (2020).,*Diffusion, adoption et implémentation du BIM dans les agences d'architecture en France*, Université de Lorraine.
Junk S. & Kuen C. (2016), « Review of Open Source and Freeware CAD Systems for Use with 3D-Printing », *Procedia CIRP*, 50, 430‑ 435. https://doi.org/10.1016/j.procir.2016.04.174.
Laakso M. & Kiviniemi A. (2012), « The IFC standard. A review of history, development, and standardization, information technology », *ITcon*, 17 (9).
Meeker H. (2020), *Open source for business . A practical guide to open source software licensing* (third edition), CreateSpace Independent Publishing Platform.
Poinet P. (2020), *Enhancing Collaborative Practices in Architecture, Engineering and Construction through Multi-Scalar Modelling Methodologies*. https://doi.org/10.13140/RG.2.2.25478.73280.
Poinet P., Stefanescu D. & Papadonikolaki E. (2021), « Collaborative Workflows and Version Control Through Open-Source and Distributed Common Data Environment », *in* E. Toledo Santos & S. Scheer (dir.), *Proceedings of the 18th International Conference on Computing in Civil and Building Engineering* (Vol. 98, p. 228‑ 247), Springer International Publishing. https://doi.org/10.1007/978-3-030-51295-8_18.
Rauskolb R. (1976), « Dr. Wangs' Palo Alto TINY BASIC », *Interface Age*, 2 (1), 92‑ 108.
Raymond E. (1999), « The cathedral and the bazaar », *Knowledge, Technology & Policy*, 12 (3), 23‑ 49. https://doi.org/10.1007/s12130-999-1026-0.
Schenck D. & Wilson P.R. (1994), *Information modeling. The EXPRESS way*. Oxford University Press.
Stefanescu D. (2020), *Alternate Means of Digital Design Communication*.
Swink M. (2006), « Building Collaborative Innovation Capability », *Research Technology Management*, 49 (2), 37‑ 47. JSTOR.
Townsend A. (2014), « On the Spline. A Brief History of the Computational Curve », *International journal of interior architecture + spatial design*, Issue 3.
Ulmanis J. (2015), « CASE 6 : Graphisoft. The Architecture of International Growth », *in* J. Prats, M. Sosna & S. Sysko-Romańczuk, *Entrepreneurial Icebreakers* (p. 193‑ 209), Palgrave Macmillan UK. https://doi.org/10.1057/9781137446329_12.