Faciliter l'interopérabilité à l'aide de flux de données directs, expérimentation d'une plateforme distribuée ouverte

Facilitate Interoperability with the help of direct data flows, experimentation of an open distributed platform

Ahmed Wael Ismail[1,2,*] et *Philippe* Marin[1]

[1]Laboratoire MHA, École Nationale Supérieure d'Architecture de Grenoble (ENSAG), Univ. Grenoble Alpes (UGA)
[2]Laboratoire CNPA, ENAC, École Polytechnique Fédérale de Lausanne (EPFL)

Résumé. Cette étude aborde la question de l'interopérabilité logicielle pour le secteur de la construction, en s'intéressant à l'écosystème Speckle, plateforme collaborative ouverte facilitant l'échange d'information. Nous cherchons à caractériser les modalités de conception qui s'appuient sur un flux informationnel direct (FID). Nous présentons ici le développement du connecteur Speckle.TopSolid qui permet un échange direct de données entre l'environnement CFAO TopSolid, et la plateforme Speckle. Après un rappel des notions d'écosystème et des modalités de collaboration, nous détaillons la description technique du connecteur. L'objectif général des travaux est d'offrir une alternative aux méthodes d'interopérabilité traditionnelles, basées sur les échanges de fichiers ou des solutions propriétaires dans des écosystèmes fermés, en suggérant un passage aux échanges directs entre logiciels, par le biais de flux informationnels directs, structurés et en temps réel. Nos développements s'inscrivent dans les dynamiques libres et ouvertes.

Mots clés. BIM, Flux d'information, interopérabilité, écosystème.

Abstract. This study addresses the issue of software interoperability in the AEC sector, it is interested in the ecosystem of Speckle, the open collaborative platform, facilitating data exchange. We seek to characterise to design modalities that rely on a direct information flow (DIF). We present here the development of the connector Speckle.TopSolid, allowing a direct data exchange between CAD-CAM TopSolid environment and the platform. After reviewing the notion of ecosystem and collaborative processes, we give a detailed technical description of the connector. The overall objective is to offer an alternative to traditional interoperability methods, based on file-exchange or proprietary solutions in closed ecosystems, by suggesting

* Corresponding author: ahmed.wael@grenoble.archi.fr

direct exchanges between software, through direct informational flows. Our development is inscribed in free and open dynamism.

Keywords. BIM, data flow, interoperability, ecosystem.

1 Introduction

Ce travail porte sur la caractérisation des modalités de conception collaborative qui s'appuient sur un flux informationnel direct (FID) pour déterminer les échanges d'informations entre les acteurs de la conception architecturale. La recherche passe par le développement informatique spécifique, avec l'objectif d'établir un lien entre un logiciel de CFAO, TopSolid, et les logiciels du secteur de la construction, via la plateforme de travail collaboratif Speckle.Systems. Le développement prend la forme d'un connecteur Speckle pour TopSolid. Son objectif est de proposer des fonctionnalités d'échanges en temps quasi-réel entre les différents environnements, pour faciliter les activités de conception concourante entre différents acteurs du bâtiment.

Après une présentation de l'objectif de la recherche et sa méthodologie, nous rappelons la notion d'écosystème logiciel et des caractéristiques de la collaboration en conception architecturale. Nous présentons ainsi la plateforme Speckle, comme modèle orienté-objet, tout en expliquant son architecture logicielle client/serveur. Ensuite nous décrivons en détail les développements réalisés, en commençant par le contexte général, puis une description technique, avant de présenter un cas d'usage, complété par une évaluation de l'outil développé.

2 Objectifs généraux et approche méthodologique

Cet article fait partie de la recherche menée dans le cadre d'une thèse doctorale. L'objectif est l'analyse des modalités de conception collaborative dans un contexte d'instrumentation, mettant en œuvre une interopérabilité directe pour accompagner les activités de conception.

La méthodologie s'appuie sur la réalisation prototypale d'outils numériques, dont le rôle est de faciliter la conception concourante et d'autoriser une interopérabilité directe par le biais de FID. La recherche s'intéresse au cas particulier de l'interopérabilité entre le secteur du bâtiment et l'industrie. Pour ceci, nous analysons les pratiques en matière de travail collaboratif et de conception concourante en architecture, et nous réalisons une comparaison avec les pratiques industrielles. La compréhension de ces modalités ainsi que l'établissement de son état de l'art, nous permettent de définir notre cadre théorique ainsi que l'orientation des réalisations informatiques. Cette démarche se concrétise par le développement de deux outils numériques : Le Connecteur *Speckle.TopSolid* et une solution *Rhino.Inside.Topsolid*, tous deux issus de projets *Open-Source*. L'enjeu est de privilégier des solutions libres, ouvertes et indépendantes.

La recherche part de l'hypothèse suivante : il existe un besoin d'interopérabilité directe et libre pour deux raisons : l'une associée aux limites techniques et fonctionnelles des processus traditionnels basés sur des échanges de fichiers, l'autre pour éviter l'enfermement dans des environnements propriétaires, constituant des « écosystèmes » logiciels fermés.

Le développement constitue une approche expérimentale, dont les produits sont des outils logiciels facilitant la collaboration, sans échanges de fichiers et au sein d'un écosystème ouvert. L'évaluation des outils se réalisera à l'occasion d'expérimentations en situation. Elle se basera sur les critères de la norme ISO/IEC 25010, adaptée à notre contexte. Cette norme définit un standard pour l'évaluation de la qualité des produits logiciels, le *Systems and Software Quality Requirement and Evaluation (SQuaRE)* [1]. Celui-ci identifie deux familles de critères : la qualité dans l'usage (*Quality in Use*) qui se base sur le rendement, l'efficacité,

la satisfaction, la liberté et l'adaptation au contexte, et la qualité du produit (*Product Quality*), qui porte sur la fonctionnalité, la performance, la compatibilité, l'utilisabilité, la sûreté ou la portabilité.

Dans cet article nous présentons la notion d'écosystème informatique ainsi que les principales caractéristiques associées à l'échange d'information dans le domaine de la conception architecturale et nous nous concentrons sur la description de l'outil Speckle.Topsolid.

Nos travaux visent à offrir une alternative aux processus habituels de conception collaborative. Les échanges d'informations se font à l'aide de FID, se basant sur les *Streams* de Speckle. Le développement est en cours, il s'agit d'analyser ici les premiers résultats, qui constituent une preuve de concept (*Proof of Concept, POC*), visant à confirmer l'adéquation de la solution au problème posé. Cette première évaluation a pour but de valider le travail de développement et d'orienter l'évolution et les améliorations de notre outil.

3 Écosystèmes logiciels et échanges d'informations

3.1 Les écosystèmes : définitions et caractéristiques

L'analyse de la notion d'écosystème établit un cadre technique général pour accompagner les activités de conception collaborative. Le concept d'écosystème est popularisé par la discipline de l'informatique, qui en fait un modèle d'innovation et de collaboration, aussi bien pour le domaine des hautes technologies que pour le secteur de la construction [2]. Ces processus d'innovation ont progressivement contribué à renouveler les pratiques de la conception architecturale en développant de nouveaux outils, de nouveaux processus, de nouvelles méthodologies de conduite de projet.

Un ensemble de définitions de la notion d'écosystème logiciel, *Software Ecosystem* (*SECO)*, est proposé par Manikas et Hansen [3]. Une première formulation [4] envisage ces écosystèmes sous un angle économique : *We define a software ecosystem as a set of businesses functioning as a unit […] under-pinned by a common technological platform, through the exchange of information, resources and artefacts.* Messerschmidt et Szyperski [5] fournissent une définition informatique, considérant un écosystème comme une collection de produits logiciels avec un degré de relation symbiotique. Bosch et al. [6] abordent l'aspect collaboratif d'un écosystème, en tant que solutions logicielles facilitant des tâches de collaboration entre différents acteurs. Campbell et Ahmed [7] distinguent les dimensions économiques, de l'architecture logicielle et des questions sociales. Une dernière caractéristique porte sur la distinction entre les écosystèmes propriétaires et les écosystèmes libres-gratuits, *Free or Open-Source Software (FOSS)*.

Ainsi, un écosystème constitue un système technique facilitant les échanges et la collaboration. Il facilite la mise en réseau d'environnements logiciels distincts et permet la collaboration entre différents acteurs de différentes cultures.

3.2 Collaboration et échanges d'informations en architecture

Le projet d'architecture mobilise par essence un grand nombre d'acteurs spécialisés, tout autant qu'une variété de cultures, d'approches et de méthodes. La collaboration autour du projet impose des échanges d'informations et de documents. Ces échanges sont conditionnés par le degré l'interopérabilité des différents outils logiciels. Bignon [8] identifie trois types de logiciel pour la collaboration : les bases de données, les gestionnaires de flux de tâches et les outils CAO à fonction augmentée de collaboration. Dans le bâtiment, ces trois types d'outils sont employés de manière complémentaire. La fluidité des échanges, tout autant que

la capacité à relier différents logiciels apparaissent fondamentaux. Par ailleurs, les pratiques avancées montrent l'intérêt des échanges d'informations qui s'opèrent en temps réel. Cette modalité marque le passage du modèle de collaboration séquentielle à un modèle coopératif simultané, autrement dénommé ingénierie concourante [9].

La plateforme Kroqi, récemment mise à la disposition du secteur et réalisé dans le cadre du Plan de transition numérique du bâtiment (PTNB) dans le cadre du programme Plan BIM 2022, propose une plateforme collaborative en rassemblant de manière structurée l'ensemble de la documentation d'un projet. Elle illustre les approches qui portent sur les flux de tâches.

Autodesk, l'éditeur logiciel dominant le marché, propose des fonctionnalités de collaboration au sein d'un d'écosystème fermé. Le groupe américain est devenu un acteur très influent dans le domaine de l'instrumentation des processus de conception et de fabrication [10]. En quatre décennies, l'écosystème Autodesk s'est formé par le biais d'une stratégie de fusion et d'acquisition : acquisition de nouvelles compétences, de concurrents, ou de collaborateur [10]. Le portfolio d'Autodesk s'étend sur 3 secteurs : le bâtiment (AutoCad, Revit, etc.), la conception et fabrication de produit (Fusion, Inventor, etc.) et les médias et loisirs (3ds Max, Maya, etc.) [11]. Les solutions Autodesk constituent en quelque sorte un écosystème d'écosystèmes. Les échanges d'informations sont favorisés à l'intérieur de cet ensemble, renforçant ainsi la notion d'écosystème logiciel propriétaire fermé. Cependant, il existe une possibilité d'extension avec la mise en place d'une *Application Programming Interface* (API) accessible, donnant lieu à des développements complémentaires. Ces développements peuvent ou pas faire partie de l'écosystème de base.

La plateforme Speckle se distingue en proposant un environnement logiciel ouvert, offrant un accès aux codes sources de la plateforme et s'appuyant sur les contributions de communautés de développeurs pour enrichir ses fonctionnalités. Elle est un moyen de partage des descriptions géométriques et des données du projet architectural entre différents environnements de conception. Nous décrirons plus en détail cet environnement dans les paragraphes qui suivent.

3.3 La standardisation des formats d'échange

Les pratiques de collaboration sont étroitement associées aux qualités et aux possibilités offertes par les outils logiciels et par la standardisation des formats d'échange.

Ainsi Eastman et al. [12] identifie 4 modalités principales d'échanges d'informations pour le secteur de la construction : 1) liens directs et propriétaires entre environnements logiciels ; 2) échanges de fichiers dans des formats propriétaires ; 3) échanges de fichiers dans des formats ouverts ; 4) échanges de fichiers dans un format XML. La première modalité facilite les échanges synchronisés, mais reste limitée aux produits d'un même éditeur ou de produits faisant l'objet d'un accord entre éditeurs. Les modalités 2 et 3 passent par l'échange de fichiers, elles présentent une forme de simplicité technique en se résumant à l'écriture et à la lecture de fichiers textes, mais elles impliquent une multiplication des étapes (sauvegarde, export, envoi, réception, import, conversion en natif) sans nécessairement intégrer des fonctions de gestion des versions, de traçage des historiques et de synchronisation en temps réel. Les modèles neutres d'objet, à l'image du standard IFC issu du langage EXPRESS, offrent une alternative aux méthodes propriétaires et permettent l'échange de données entre outils n'appartenant pas au même éditeur. Dans la pratique, la mise en place de protocoles collaboratifs basés sur les standards ouverts est matérialisée par l'échange de fichier .ifc, avec une étape de conversion et de traduction du modèle natif, souvent propriétaire, en un modèle IFC et inversement. Le langage XML, quant à lui, est dérivé de l'HTML, il est initialement envisagé pour le web et reste plus adapté pour les fichiers de petites tailles.

Les échanges directs entre applications, font l'objet aujourd'hui d'une attention particulière. Ces solutions dialoguent avec les API des logiciels pour envoyer et recevoir des

données et convertir les informations. Nous citons Conveyor réalisé par Proving Ground permettant le lien entre Revit et Rhino, ou encore BHoM (Building Habitat object Model), projet Open-Source développé par BuroHappold [13,14], basé sur un modèle d'objet intermédiaire et des liaisons avec la plupart des logiciels du bâtiment, ou encore Speckle, sur lequel nous appuyons nos développements et que nous détaillons dans les sections suivantes.

4 Speckle : un écosystème ouvert et *open-source*

4.1 Une approche orientée-objet

Speckle constitue un écosystème logiciel ouvert, conçu pour le secteur de la construction et permettant d'assurer une communication directe entre les différents outils informatiques.

Speckle est un projet de *Common Data Environment* (CDE) distribué, il se caractérise par les points suivants : il est complètement « Open-Source », il ne se base pas sur les échanges de fichiers et il n'offre pas de fonctionnalité de type *Information Container Data Drop (ICDD)*[†] [13,15]. Il prend la forme d'une plateforme ouverte d'interopérabilité accessible en ligne via un navigateur [16].

Son développement a été entamé en 2016 dans le cadre du projet de recherche Innochain, soutenu par le programme européen H2020 [13,16,17]. Il permet aux logiciels du secteur de la construction de communiquer entre eux, à l'aide de connecteurs, agissant comme clients pour établir une connexion avec un serveur. Les collaborateurs d'un projet peuvent ainsi échanger des informations entre eux en temps réel, sans usage de fichiers intermédiaires.

L'approche de Speckle est orientée « objet », par opposition avec les méthodes habituelles orientées « fichier ». L'avantage est de sélectionner les données échangées. La méthode impose d'anticiper l'utilité et l'intérêt de l'information transmise et facilite le suivi des échanges. Speckle se démarque par son système de communication directe entre logiciels, communications rendues possibles par des connecteurs accédant aux API des logiciels de conception et centralisant les données sur un serveur en ligne. La solution permet d'échanger de la géométrie 3D sémantique, d'organiser le flux de données ainsi que le traçage des interactions entre les collaborateurs.

4.2 Une architecture client/serveur

Speckle est basé sur protocole REST (*REpresentational State Transfer*), il permet la création de services web avec un schéma de fonctionnement client/serveur. La connexion au serveur se fait soit par le biais d'applications Web, soit par des connecteurs directement intégrés dans les logiciels de conception. Une interface graphique intégrée au logiciel de CAO permet la connexion au serveur et la réalisation des actions d'envoi et de réception. Les données sont stockées dans des *Streams* [17], eux-mêmes intègrent la notion de « calques » et ces derniers stockent des « objets » [15,17]. Le format de description des objets reprend le formalisme JSON. Il n'y a pas de modèle d'objet imposé à l'usager. Celui-ci peut choisir d'implémenter un modèle par défaut, fourni par Speckle, ou d'utiliser un modèle de son choix tels que BHoM ou l'IFC. Ainsi, Speckle est une solution décentralisée, sans fichier partagé, mais proposant plutôt une organisation des données à l'aide de *Streams* hébergées sur un serveur.

[†] « Conteneur d'informations pour la livraison de documents liés, un format de conteneur permettant d'échanger des fichiers de nature hétérogène afin de livrer, de stocker et d'archiver des documents qui décrivent un bien tout au long de son cycle de vie » (ISO 21597-1:2020)

Au niveau technique, il est important de noter que la première implémentation du projet Speckle a été réalisée en .NET, à l'instar de la plupart des logiciels et API du secteur. Les développements récents ont donné lieu à des SDK (*Software Development Kit*) en Python et bientôt en NodeJS, JS et C++ [18]. Ces évolutions permettent d'élargir la compatibilité de la solution avec un nombre plus important de logiciels. L'application web SpeckleViz permet de visualiser les différents éléments d'un projet (les *Streams*) ainsi que leurs liens et relations.

La gestion des versions se base sur le modèle Git, plébiscité dans le domaine du développement informatique [19,20]. Speckle intègre donc aussi bien un historique des révisions pour un *Stream* » qu'une chronologie de projet [13]. Cela se réalise grâce à une fonction similaire à celle dénommée *Git diff*. Elle confère aux usagers un outil puissant de collaboration, avec la possibilité de travailler en parallèle dans des branches, puis de réaliser des fusions (*Merge*) automatisées, tout en comparant le contenu des différents *Commits*.

5 Expérimentation : développement du connecteur Speckle. Topsolid

5.1 Contexte et objectifs du développement

Nos travaux portent sur le développement d'un connecteur Speckle pour TopSolid, dénommé Speckle.TopSolid. Il permet d'intégrer ce logiciel de CFAO à l'écosystème Speckle pour mettre en place une interopérabilité directe avec les autres logiciels de conception du secteur, déjà intégré à l'écosystème Speckle. Les développements bénéficient largement des connecteurs existants et dont les codes sources sont en libre accès sur la plateforme Github.

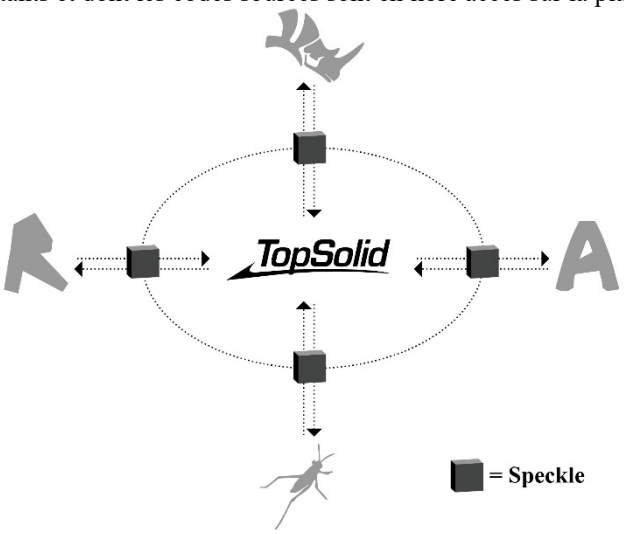

Fig. 1. L'intégration de TopSolid dans Speckle.

Le logiciel TopSolid intègre le noyau de modélisation géométrique exacte, dénommé Parasolid et développé par Siemens, ainsi qu'un système de *Product Data Management* (PDM) [21,22]. Le PDM permet d'assurer le lien entre les différents documents constituant une chaîne paramétrique et associative, avec un traçage de l'évolution et une gestion des versions. C'est un mode de travail caractéristique dans les domaines industriels, basé sur la modélisation paramétrique (de type *Feature-based*) et la logique de pièces/assemblages [23]. Cette dernière se caractérise par l'inclusion de pièces dans des assemblages. Les modules

TopSolid'Wood et *TopSolid'Steel* regroupent des fonctionnalités pertinentes pour les métiers du secteur du bâtiment. TopSolid intègre le standard IFC [24], ainsi que STEP ou d'autres formats géométriques standards du marché. Ainsi, le logiciel supporte les standards les plus employés en matière d'échanges de données. L'objectif du développement est de fournir une alternative aux pratiques collaboratives basées sur les échanges de fichiers.

5.2 Description technique

5.2.1 Environnement de développement

Notre environnement logiciel est constitué par les deux outils TopSolid et Speckle. L'accès aux fonctionnalités des deux environnements est réalisé à l'aide des API mises à disposition. Le développement informatique est réalisé en .NET, son développement est codé en C#.

L'API de TopSolid, dénommée Top'ADS, nous permet de développer une extension compatible avec le logiciel. Notre connecteur appelle les commandes des différentes classes d'objets de TopSolid, tels que la géométrie du noyau Parasolid, ou le PDM et sa base de données, permettant la gestion des informations appartenant aux différents projets.

Speckle fournit un SDK pour faciliter le développement des applications. Dans le vocabulaire Speckle, le SDK est nommé Core, et il est complété par la bibliothèque « Objects Kit », celle-ci contient la définition du modèle d'objet ainsi que les convertisseurs nécessaires. Le modèle d'objet contient la définition des objets géométriques et/ou sémantiques. C'est un modèle standard proposé par Speckle, dont l'usage est recommandé mais n'est pas obligatoire [25]. La définition de modèle personnalisé d'objet est possible. Les convertisseurs sont ceux qui permettent la conversion depuis et vers les modèles natifs. Il y a un convertisseur par connecteur, qui est double en offrant des fonctions se chargeant de la conversion depuis Speckle et vers Speckle. En tant que projet *Open-Source*, cette structure est accessible sur la zone de dépôt GitHub [26]. L'API de Speckle se base sur le même cadriciel .NET.

5.2.2 Phasage des développements

Nous avons réalisé un premier développement permettant d'intégrer notre extension à TopSolid pour assurer une connexion à Speckle. Le développement s'est fait sur plusieurs étapes et reste en cours.

La première étape a porté sur la construction de l'interface utilisateur. Nous avons fait le choix d'adopter l'interface usager par défaut proposée par Speckle et qui contient les boutons et commandes nécessaires. Cette interface de type WPF (*Windows Presentation Foundation*) permet : la connexion au serveur Speckle via un compte (authentification), le chargement ou la création d'un *Stream*, la sélection des objets à envoyer, l'envoi ou la réception. Une liaison (*Binding*) est réalisée entre le bouton et la commande de l'action du logiciel hôte. Le code de cette étape est donc de caractère purement fonctionnel, servant au bon chargement et lancement des différents modules ainsi qu'à la gestion des dépendances.

La seconde étape a porté sur l'encodage des objets et de la géométrie. Le codage des convertisseurs représente une étape importante du travail. Ceux-ci permettent de traduire les géométries de TopSolid en géométries Speckle, définies dans leur modèle *Objects*. Il s'agit, par exemple, des conversions géométriques des courbes ou surfaces BSpline de TopSolid, en NURBS de Speckle, ou encore des solides TopSolid, en BRep Speckle. Ces conversions sont laborieuses, elles demandent des connaissances en géométrie, en infographie et en computation. Parallèlement aux conversions géométriques, un travail a été réalisé pour accéder au PDM de TopSolid et permettre l'inscription des opérations de Speckle dans l'arbre chronologique et la structure TopSolid. Il convient de souligner l'avantage du travail en

Open-Source, il offre des possibilités d'inspiration et d'auto-vérification, ou permet les interactions avec la communauté.

5.3 Cas d'usage

Pour évaluer notre développement, un cas d'usage se base sur un scénario de collaboration entre deux architectes et le bureau d'étude technique d'un façadier (Fig. 2). Cela correspond à une situation professionnelle, avec une maquette principale existe sur un logiciel BIM. Des acteurs de l'industrie collaborent, des informations spécifiques nécessitent un partage, des envois et/ou réceptions, chaque collaborateur travaillant sur son logiciel, en fonction de ses habitudes et de ses pratiques. Cette phase de test a pour objectif la vérification des fonctionnements dans l'état actuel du développement, en cherchant à mettre à l'épreuve les interactions de notre extension avec les API TopSolid et Speckle dans une situation de conception collaborative.

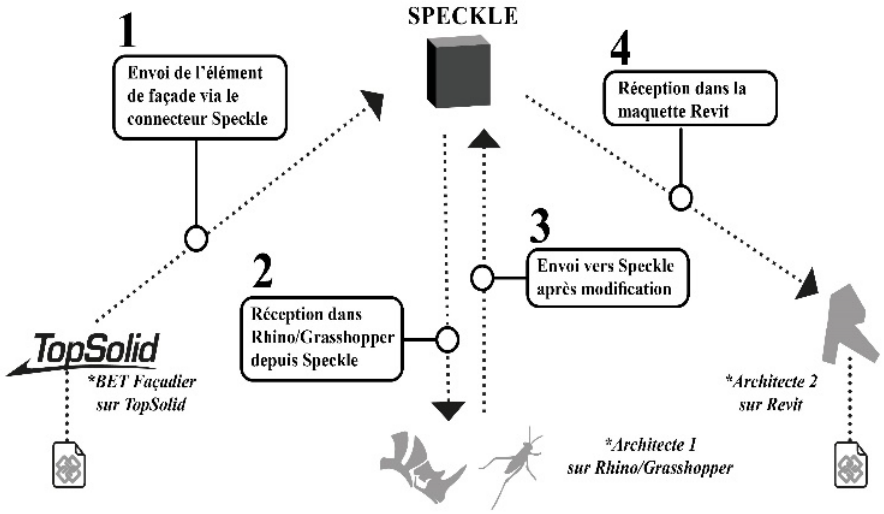

Fig. 2. Processus utilisé pour tester le connecteur Speckle.TopSolid.

Une maquette de la Villa Le Sextant de Le Corbusier (Fig. 3) a été prise comme cas d'étude. L'équipe doit collaborer à la mise au point d'un nouveau brise soleil, ce dispositif doit in fine être réceptionné par le bureau d'étude pour sa mise en fabrication. Le modèle IFC de la Villa a été importé dans TopSolid et dans Revit. Sur TopSolid, nous avons simulé le travail d'un façadier venant rajouter un élément modélisé paramétriquement sur TopSolid, dans l'objectif d'être usiné sur machine à commande numérique, pilotée par TopSolid. À l'aide de Speckle, cet élément a été envoyé et reçu dans Rhinoceros, puis il a subi des modifications, avant d'être renvoyé dans Speckle pour être finalement reçu dans Revit. À ce moment il trouve sa place dans la maquette numérique au bon endroit avec les bonnes dimensions.

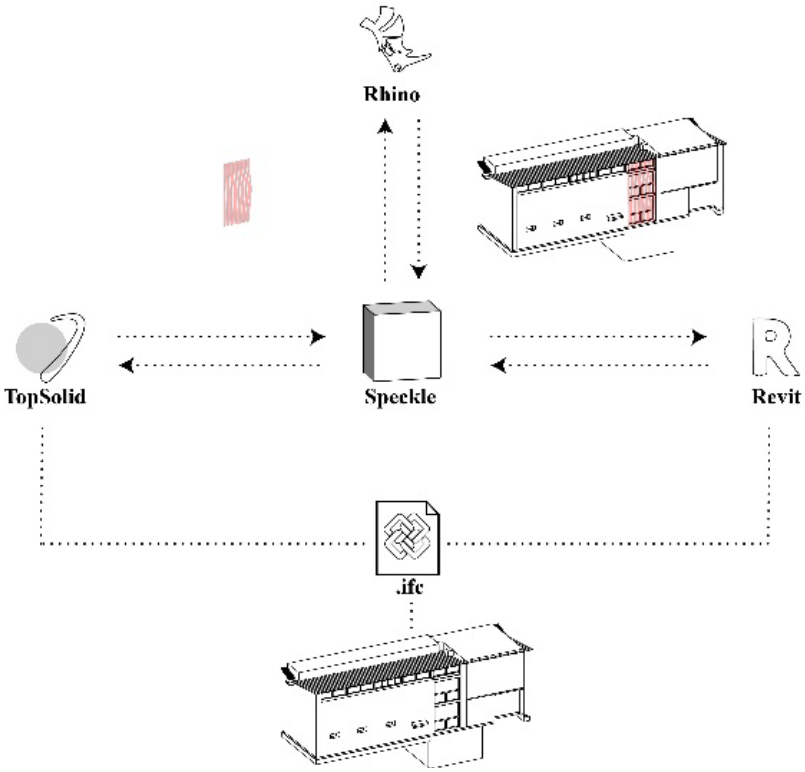

Fig. 3. Illustration des échanges dans notre cas d'usage.

5.4 Évolutions et prolongements

À ce stade du développement, nous avons testé le fonctionnement technique de notre outil. Le connecteur permet le transfert des géométries dans les deux sens. Il autorise des échanges de données entre différents logiciels, appartenant à différents écosystèmes sans constituer un environnement propriétaire et sans passer par des import/export de fichiers.

Des améliorations sont identifiées et devront être intégrées : 1) la prise en compte de la donnée sémantique, 2) l'intégration des objets au sein de la structure des données des environnements hôtes, avec le suivi des versions, 3) l'inscription des *Streams* dans le PDM de TopSolid, 4) la synchronisation des deux systèmes de gestion de versions.

Par ailleurs, une évaluation plus approfondie devra être menée à une échelle plus large pour envisager et tester d'autres cas d'usage et valider les développements futurs. Cette évaluation s'appuiera sur la méthode et les critères de la norme ISO mentionné plus haut. Elle mobilisera des utilisateurs professionnels et des étudiants en architecture pour examiner la conformité du processus proposé, pour vérifier qu'il répond aux exigences de la pratique et pour regarder comment il transforme les activités. Il s'agit d'examiner dans quelles mesures une solution libre basée sur les FID, résout le problème de l'interopérabilité logicielle, facilite la collaboration et la conception concourante.

6 Conclusion

Dans cet article, la notion d'écosystème numérique a été abordée et explicitée. Elle a permis d'identifier différentes dimensions techniques et socio-économiques. Celles-ci prennent de

l'importance dans les grands projets collaboratifs entre différents acteurs employant des outils différents. Le cas de Speckle est un exemple intéressant d'écosystème, offrant une alternative ouverte aux approches dites propriétaires. L'avantage de Speckle porte sur la suppression des échanges de fichier, il les remplace par des Flux Informationnels Directs (FID) entre logiciels ne faisant pas partie d'un même écosystème, tout en encadrant ces échanges par un système de traçage et de gestion de versions. Ceci offre une alternative aux écosystèmes logiciels propriétaires fermés, et facilitent l'usage et le libre choix d'une variété d'outils dans le but d'ouvrir la collaboration à l'hétérogénéité des pratiques et des instruments.

Nous avons présenté le développement d'un connecteur Speckle pour TopSolid, permettant d'établir une liaison entre les acteurs du bâtiment et l'industrie. Les verrous techniques ont été levés en faisant dialoguer deux environnements. La conversion des géométries natives a été réalisée pour permettre ces échanges. Un cas d'usage a permis de valider les fonctionnements. Ce premier résultat permet de confirmer les apports de l'outil développé, il facilite l'interopérabilité et offre une alternative aux pratiques courantes. Des développements complémentaires ainsi que des évaluations sur la base de nouveaux cas d'usage sont en cours.

De manière générale, les travaux permettent de révéler le potentiel des approches *Open-Source*, aussi bien dans la perspective de recherches et de développements que d'activités opérationnelles des acteurs de la construction. Les écosystèmes ouverts de type Speckle paraissent pertinents pour la communauté académique et le monde professionnel, pour assurer l'ouverture de la donnée et la liberté de l'usager. Les modalités d'échanges directs, associées aux activités concourantes de conception, impliquent une rigueur dans la définition des structures de données et dans le choix des informations partagées, elles nécessitent la compréhension de l'usage et des besoins en informations de chacun des acteurs.

Références

1 J. Estdale et E. Georgiadou, Applying the ISO/IEC 25010 quality models to software product, pp. 492–503 (2018).

2 L. Pulkka, M. Ristimäki, K. Rajakallio, and S. Junnila, Applicability and benefits of the ecosystem concept in the construction industry. Construction Management and Economics, **34**, no. 2, pp. 129–144 (2016) doi: 10.1080/01446193.2016.1179773.

3 K. Manikas and K. M. Hansen. Software ecosystems – A systematic literature review. Journal of Systems and Software, **86**, no. 5, pp. 1294–1306 (2013) doi: 10.1016/j.jss.2012.12.026.

4 S. Jansen, A. Finkelstein, and S. Brinkkemper. A sense of community: A research agenda for software ecosystems. in 2009 31st International Conference on Software Engineering - Companion Volume, Vancouver, BC, Canada, pp. 187–190 (2009) doi: 10.1109/ICSE-COMPANION.2009.5070978.

[5] C. Szyperski and D. G. Messerschmitt. Software Ecosystem: Understanding an Indispensable Technology and Industry. Cambridge; Ipswich: MIT Press ; Ebsco Publishing [distributor. Accessed: Mar. 22, 2022 http://ieeexplore.ieee.org/servlet/opac?bknumber=6267307

6 J. Bosch and P. Bosch-Sijtsema. From integration to composition: On the impact of software product lines, global development and ecosystems. Journal of Systems and Software, **83**, no. 1, pp. 67–76 (2010) doi: 10.1016/j.jss.2009.06.051.

7 P. R. J. Campbell and F. Ahmed. A three-dimensional view of software ecosystems. in Proceedings of the Fourth European Conference on Software Architecture Companion

Volume - ECSA '10, Copenhagen, Denmark, p. 81 (2010) doi: 10.1145/1842752.1842774.

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

9 J. C. Bignon, O. Malcurat, and G. Halin. Coopération et Conception, Vers une coopération assistée pour les acteurs du bâtiment. Saint Ferréol, p. 12 (1999)

10 J. K. C. Chen and H.-H. Ho. Transformation and Impact from the Software Ecosystem Perspective: Case Study of Autodesk Inc.'s Ecosystem Roadmap. p. 7.

11 Industry Collections | Autodesk https://www.autodesk.com/collections (2022).

12 C. M. Eastman, P. M. Teicholz, R. Sacks, and G. Lee, *BIM handbook: a guide to building information modeling for owners, managers, designers, engineers and contractors* (Third edition. Hoboken, New Jersey: Wiley, 2018)

13 P. Poinet, D. Stefanescu, and E. Papadonikolaki. Collaborative Workflows and Version Control Through Open-Source and Distributed Common Data Environment. in Proceedings of the 18th International Conference on Computing in Civil and Building Engineering, **98**, E. Toledo Santos and S. Scheer, Eds. Cham: Springer International Publishing, pp. 228–247 (2021) doi: 10.1007/978-3-030-51295-8_18.

14 P. Poinet and A. Fisher. Computational Extensibility and Mass Participation in Design. pp. 40–47 (2020) doi: 10.2307/j.ctv13xprf6.9.

15 P. Poinet, D. Stefanescu, and E. Papadonikolaki. SpeckleViz: A Web-based Interactive Activity Network Diagram for AEC. p. 8.

16 P. Poinet, D. Stefanescu, and E. Papadonikolaki. Web-based distributed design to fabrication workflows. p. 10.

17 P. Poinet, D. Stefanescu, G. Tsakiridis, A. De Boissieu, and E. Papadonikolaki. Supporting collaborative design and project management for AEC using Speckle's interactive data flow diagram. presented at the Design Computation Input/Output 2020, (2020) doi: 10.47330/DCIO.2020.VYBA4375.

18 Tools for Every Stack, Speckle (2021) https://speckle.systems/developers/sdks/

19 Speckle, Speckle (2022) https://speckle.systems/developers/sdks/

20 S. Chacon, Pro Git. Berkeley, CA : New York: Apress ; Distributed to the book trade worldwide by Springer-Verlag (2009)

21 M. Swink. Building Collaborative Innovation Capability. Research Technology Management, **49**, no. 2, pp. 37–47 (2006)

22 M. Eigner. *System Lifecycle Management: engineering digitalization (Engineering 4.0)* (Wiesbaden [Heidelberg]: Springer Vieweg, 2021)

23 P. Janssen and R. Stouffs, Types of Parametric Modelling. CUMINCAD, p. 10 (2015)

24 Certified Software, buildingSMART International https://www.buildingsmart.org/compliance/software-certification/certified-software/

25 Introduction | Speckle Docs. https://speckle.guide/dev/dotnet.html#getting-started

26 Speckle | Sharp. Speckle (2022) https://github.com/specklesystems/speckle-sharp