Le génie pour l'industrie

RAPPORT TECHNIQUE
PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
DANS LE CADRE DU COURS LOG792

Calculateur ACV

Interface simplifiée pour l'évaluation du cycle de vie (LCA) d'un bâtiment en construction

MATHIEU BINETTE
BINM27029100

Département de génie logiciel et des TI

Professeur-superviseur
ALAIN APRIL

MONTRÉAL, 9 AOÛT 2015
ÉTÉ 2015

Calculateur ACV

Interface simplifiée pour l'évaluation du cycle de vie (LCA) d'un bâtiment en construction

MATHIEU BINETTE

BINM27029100

RÉSUMÉ

Ce rapport porte sur le projet d'interface simplifiée pour l'évaluation du cycle de vie (LCA) d'un bâtiment en construction effectué dans le cadre du Projet de fin d'études en génie logiciel (LOG792).

L'objectif de ce projet est le développement d'une interface web simplifiée qui permet de faire l'évaluation du cycle vie à partir d'un modèle de bâtiment (BIM) en format IFC, basé sur la norme ISO 16739. Cette interface devra permettre de visualiser le modèle 3D, ainsi que de voir différentes statistiques sur l'évaluation des nombreuses parties de ce bâtiment. L'interface permettra de naviguer dans les statistiques fournies afin de visualiser les données à différents niveaux de granularité (Drill-Down).

Ce projet se limite au développement de l'interface web, soit la visualisation du modèle 3D du bâtiment ainsi que la visualisation des données sous différentes formes et à différents niveau de granularités.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

ACV ou LCA	Analyse du Cycle de Vie ou Life-Cycle Assessment
API	Application Programming Interface
BIM	Building Information Modeling : Un modèle BIM est un modèle de données 3D intelligent.
CPU	Central Processing Unit : Un processeur
IFC	Industry Foundation Classes : Standard pour représenter un modèle BIM.
ISO	International Organization for Standardization
NPAPI	Netscape Plugin Application Programming Interface : Une architecture datant de 1995 pour le support de plugins variés sur les navigateurs.
LEED	Leadership in Energy and Environmental Design
SGBD	Système de Gestion de Base de Données

LISTE DES SYMBOLES ET UNITÉS DE MESURE

Mo	Mégaoctet

INTRODUCTION

La multiplication des outils informatiques lors des dernières décennies a permis aux architectes de modéliser des bâtiments de plus en plus rapidement et de modifier ceux-ci avec beaucoup plus d'aisance que dans le passé.

Avec la création d'incitatifs corporatifs à réduire leur empreinte écologique, ces mêmes intervenants sont de plus en plus intéressés aux conséquences environnementales de ces décisions. Certaines mesures comme les bourses du carbone les incitent à réduire leurs impacts environnementaux au minimum.

Présentement, la modélisation et l'évaluation du cycle de vie d'un bâtiment se fait à l'aide de plusieurs outils différents, avec des fichiers de formats différents et par des intervenants différents.

Le projet *Calculateur ACV* a pour but de centraliser toutes ses opérations au sein de la même application, en utilisant un format de fichier unique, soit le format IFC tel que défini dans la norme ISO 16739[1].

Le projet "IfcViewer" a pour but de créer l'interface graphique de *Calculateur ACV* et se limite à la visualisation du modèle 3D d'un bâtiment ainsi que la visualisation de différentes statistiques dérivées de l'évaluation des nombreuses parties de ce bâtiment. L'interface permettra de naviguer dans ces statistiques afin de visualiser les données à différents niveaux de granularité.

Les résultats de ce projet seront par la suite intégrés à *Calculateur ACV*.

CHAPITRE 1

Mise en contexte

1.1 Analyse du cycle de vie (ACV/LCA)

L'analyse du cycle de vie (ACV, ou LCA en anglais) est une méthode qui permet d'évaluer les impacts écologiques d'un produit ou un service sur différentes catégories environnementales. Pour se faire, on effectue d'abord une liste des matériaux utilisées pour le produit ou le service, puis on étudie les différentes phases de construction de ce produit afin de déterminer les impacts de l'utilisation de ces matériaux à différents moments. Cette étude doit prendre en compte tous les impacts d'un matériel, incluant son acquisition, ses transformations, son transport ainsi que les étapes de fin de vie de ce matériel (e.g. recyclage, récupération de matériel dangereux, etc).

Cette analyse nous permet par la suite de déterminer où on peut réduire les impacts environnementaux du produit ou service étudié. Il est très utilisé dans la rédaction d'études d'impacts et devient de plus en plus intéressant dans les entreprises en raison des nombreux incitatifs financiers qui visent à réduire l'impact environnemental des procédés utilisés au sein des entreprises (e.g. bourse du carbone, législations nationales, etc).

1.2 ACV d'un bâtiment entier

L'analyse du cycle de vie d'un bâtiment entier se penche sur l'évaluation de toutes les composants (matériaux et services) nécessaires de la conception à la destruction du bâtiment. Cela inclut donc la construction, mais également la maintenance de celui-ci en prenant en compte la durée de vie attendue du bâtiment.

Les ACV de bâtiments entier sont d'une grande valeur pour les entreprises de conception de bâtiment, autant pour les raisons mentionnées précédemment que pour atteindre certaines certifications comme la certification LEED.

Des outils informatiques et des bases de données sont déjà disponibles pour faire l'évaluation d'un bâtiment entier, mais ceux-ci sont souvent liés spécifiquement à un outil de modélisation BIM particulier, ou n'offrent qu'une évaluation partielle ou peu détaillée du bâtiment. Plusieurs outils sont souvent utilisés conjointement afin d'effectuer une ACV complète d'un bâtiment.

1.3 Calculateur ACV

Calculateur ACV[2] est un logiciel qui a pour but de permettre aux architectes, aux concepteurs et aux responsables du développement durable d'une entreprise de minimiser l'impact écologique de leurs bâtiments en visualisant ses impacts écologiques lors de la conception de celui-ci. Ces impacts sont calculés à l'aide d'un modèle de bâtiment construit à l'aide des outils de modélisation BIM disponibles actuellement sur le marché (e.g. Autodesk Revit), mais n'est pas lié à aucun d'entre eux spécifiquement.

Cet outil permettra de centraliser toutes les opérations liées à l'analyse du cycle de vie d'un bâtiment au sein d'une même plateforme, accessible via le web et de voir en temps réel l'impact de certaines décisions de conception sur différentes catégories d'impacts écologiques.

1.4 Problématique

Toutes ces données sont difficiles à représenter de manière simple autant pour les architectes, les concepteurs ainsi que pour les responsables du développement durable d'une entreprise. Les résultats d'une analyse du cycle de vie d'un bâtiment entier peuvent comporter plusieurs dizaines de catégories différentes d'impacts environnementaux, chacune étant à son tour divisée en plus de sous-catégories.

De plus, il est difficile, avec les outils qui existent aujourd'hui, de voir visuellement quelle partie du bâtiment a le plus gros impact sur les différentes catégories écologiques dont il a été question plus tôt.

1.5 Objectifs

L'objectif de ce projet est de développer une interface web simple qui permettra de présenter les données extraites d'un fichier IFC provenant d'un logiciel de modélisation BIM. Cette interface devra être disponible entièrement sur le web et il devra être possible d'utiliser cette application avec la plupart des navigateurs récents.

Ce projet se limite au développement de la visualisation du modèle 3D du bâtiment ainsi qu'à la visualisation des données de l'ACV calculées à partir des données du fichier IFC. Ces données devront ensuite être présentées à l'utilisateur de deux façons: à l'intérieur d'un tableau (qui pourra être exporté sous format Excel) ainsi qu'à l'aide d'un graphique à barres qui pourra être observé à différents niveaux de granularités (Drill-Down).

Éventuellement, les utilisateurs pourraient utiliser l'interface afin de faire des changements au modèle du bâtiment et voir en temps réel leur impact sur l'ACV, mais cela est hors portée de ce projet. Le calcul des données de l'ACV, le système d'authentification ainsi que le *back-end* général de l'application est également hors de la portée de ce projet.

1.6 Objectifs révisés

En raison de changements technologiques en milieu de projet, la visualisation des données de l'ACV a été exclue du projet et remplacée par la création d'un lecteur IFC qui aura pour but d'extraire l'information relié à la géométrie du bâtiment en plus de l'information nécessaire à l'ACV du bâtiment.

CHAPITRE 2

Visualisation du modèle 3D

2.1 Unity

Puisque nous souhaitions permettre aux utilisateurs de visualiser leur modèle 3D à même notre application web, nous avons choisis d'utiliser Unity puisqu'il permet de compiler notre projet en version en un fichier de format .unity3d, qui peut être lu par le lecteur Unity web.

Cela suppose que les utilisateurs auront installé le lecteur web de Unity, qui est disponible sur le site de Unity[3].

Il est à noter qu'au moment du développement de l'application, le lecteur web de Unity utilisait encore NPAPI [4], qui n'est plus supportée dans certains navigateurs comme les dernières versions de Google Chrome[5][6]. Il a été décidé de continuer dans cette voie puisque les travaux de Unity avec WebGL sont prometteurs et que le même projet (avec sans doute des modifications mineures) pourra être utilisé avec ceux-ci. Cependant, tel qu'expliqué dans la section 2.4 et 2.5, nous avons finalement décidé de laisser tomber Unity et d'utiliser WebGL.

2.2 Asset Unity: Esoteric Mesh Importer

Esoteric Mesh Importer[7] est un asset disponible sur le Asset Store de Unity. Il permet d'importer une vingtaine de formats de modèles 3D en plus de ceux supportés nativement par Unity. Il permet entre autre d'importer un modèle IFC/STEP dans un projet Unity. Pour se faire, l'asset utilise entre autres des librairies disponibles seulement au sein de l'éditeur Unity (pour l'importation de modèles) et pour ces raisons, l'asset ne fonctionne qu'à l'intérieur de l'éditeur Unity.

2.3 Le script d'automation (Éditeur Unity)

Tel qu'expliqué dans la section précédente, l'asset utilisé ne permettait que d'importer via l'éditeur de Unity et ne permet pas de faire l'importation dynamique d'un modèle téléversé par un utilisateur. Cependant, Unity permet de lancer l'éditeur en mode batch, c'est-à-dire, sans interface graphique[8].

Un script pour l'éditeur a donc été développé pour permettre l'importation d'un modèle IFC, son ajout dans une nouvelle scène ainsi que la compilation de cette nouvelle scène dans un format adapté au lecteur web[9].

Cependant, puisque Unity ne permet pas d'ouvrir le même projet dans deux instances différentes de l'éditeur, et que le script d'éditeur développé ne peut qu'être lancé dans le contexte d'un projet Unity, nous avons développé un script Python[10] qui fait essentiellement une copie d'un projet type et lance le script de l'éditeur Unity sur cette copie.

Bien que l'asset ne permette pas de paralléliser l'importation d'un même modèle IFC, cette façon de faire nous permet néanmoins de distribuer les différentes importations de modèles sur différents CPUs et donc de traiter plusieurs importations en même temps. De plus, elle nous assure que toute faute lors de l'importation d'un fichier IFC sera contenue dans cette copie du projet et n'aura pas d'effet sur les importations futures.

Cette méthode crée un fichier .unity3d par modèle IFC et ne permet pas de gérer de manière intelligente plusieurs versions différentes d'un même bâtiment (chaque version doit être réimportée à 100%).

2.4 Lecteur IFC/STEP: IfcReader

Tel qu'expliqué dans le chapitre 4 ("Problèmes rencontrés"), *Esoteric Mesh Importer* peut prendre près d'une minute pour un fichier de 500 Ko et plus de 3 minutes pour l'importation d'un fichier IFC de 50 Mo. Considérant qu'un fichier IFC typique dans les scénarios étudiés

peut faire jusqu'à 500 Mo, d'autres alternatives ont étés considérées d'autres alternatives à cet asset.

Pour ces raisons, une application externe à l'interface web a donc été développée pour la lecture d'un fichier IFC ainsi que la création des fichiers de géométrie rattachés à celui-ci. Cette application est le sujet du prochain chapitre, mais il est important de noter que suite au développement de ce lecteur, il a finalement été décidé de laisser tomber Unity pour la visualisation du modèle 3D et d'utiliser WebGL, puisque les performances avec Unity et le nouveau lecteur n'étaient toujours pas satisfaisante (résultats en section 4.1). En effet, l'importation du modèle était aussi long que la lecture du fichier et son importation avec *Esoteric Mesh Importer*. Il aurait probablement été possible d'optimiser le temps d'importation du modèle au sein de Unity, mais nous avons décidé d'explorer d'autres avenues avant d'investir plus de temps dans cette voie.

2.5 WebGL

WebGL[11] est une API JavaScript pour l'affichage 2D et 3D qui a fait ses débuts en 2011, et qui est aujourd'hui supporté sur la plupart des navigateurs modernes. Après les tests de performance infructueux de Unity avec le nouveau lecteur IFC/STEP, nous avons décidé d'étudier d'autres avenues pour la visualisation du modèle 3D dans le navigateur. WebGL est la solution la plus supportée et la plus rapide que nous avons trouvé, la génération du modèle atteignant parfois 10x la vitesse de la génération d'un modèle équivalent avec Unity.

Il est à noter que ce changement de cap a eu lieu un mois à peine avant la remise de ce rapport. Beaucoup de travail effectué avec Unity a donc dû être refait et cela a sans aucun doute eu un impact sur le reste du projet.

2.6 Restauration du travail fait avec Unity

Tel qu'expliqué plus tôt, le changement de technologie de Unity à WebGL a signifié la perte de tout le travail effectué jusqu'au changement. Il a donc été décidé de laisser tomber la partie sur la visualisation des données de l'ACV et de se concentrer sur la visualisation du modèle 3D ainsi que le lecteur IFC.

Bien que les résultats expérimentaux faits sur les contrôles avec Unity soient toujours valides, leurs implémentation est beaucoup plus ardue puisque nous n'avons pas accès à un environnement de programmation aussi intéressant avec un canvas/WebGL qu'avec Unity. Bien que la logique puisse facilement être traduite en Javascript, les opérations/transformations sur la caméra doivent être converties en opérations matricielles sur les matrices de vue et le manque de temps a eu comme résultat que ces fonctionnalités n'ont pas été réimplémentées.

2.7 Schéma des classes logicielles

La figure 2.7.1 illustre les modules principaux de notre application web. Certaines méthodes et paramètres ont étés omis pour des raisons de clarté.

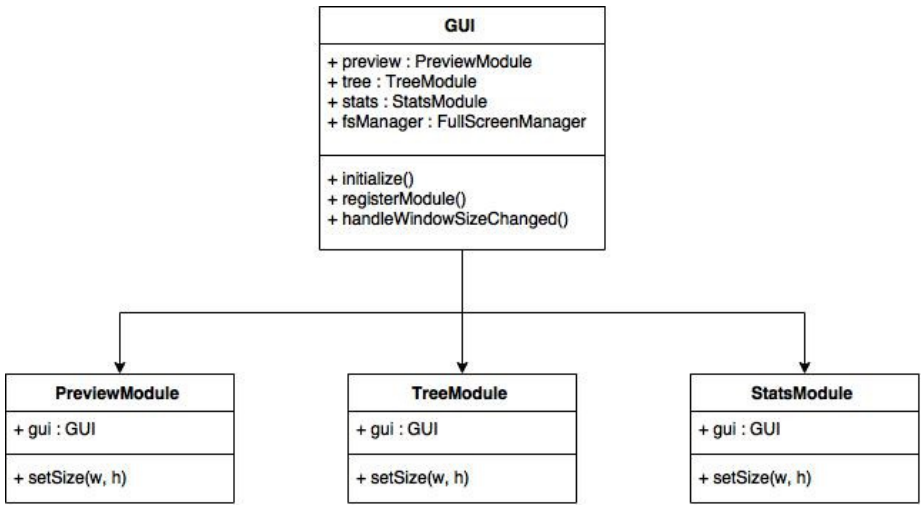

Figure 3.5.1 Diagramme de classes partiel des modules de l'application web

La figure 3.5.2 illustre les classes et les méthodes principales du visualisateur WebGL, soit le module PreviewModule. Encore une fois, certaines méthodes et paramètres ont étés omis pour des raisons de clarté.

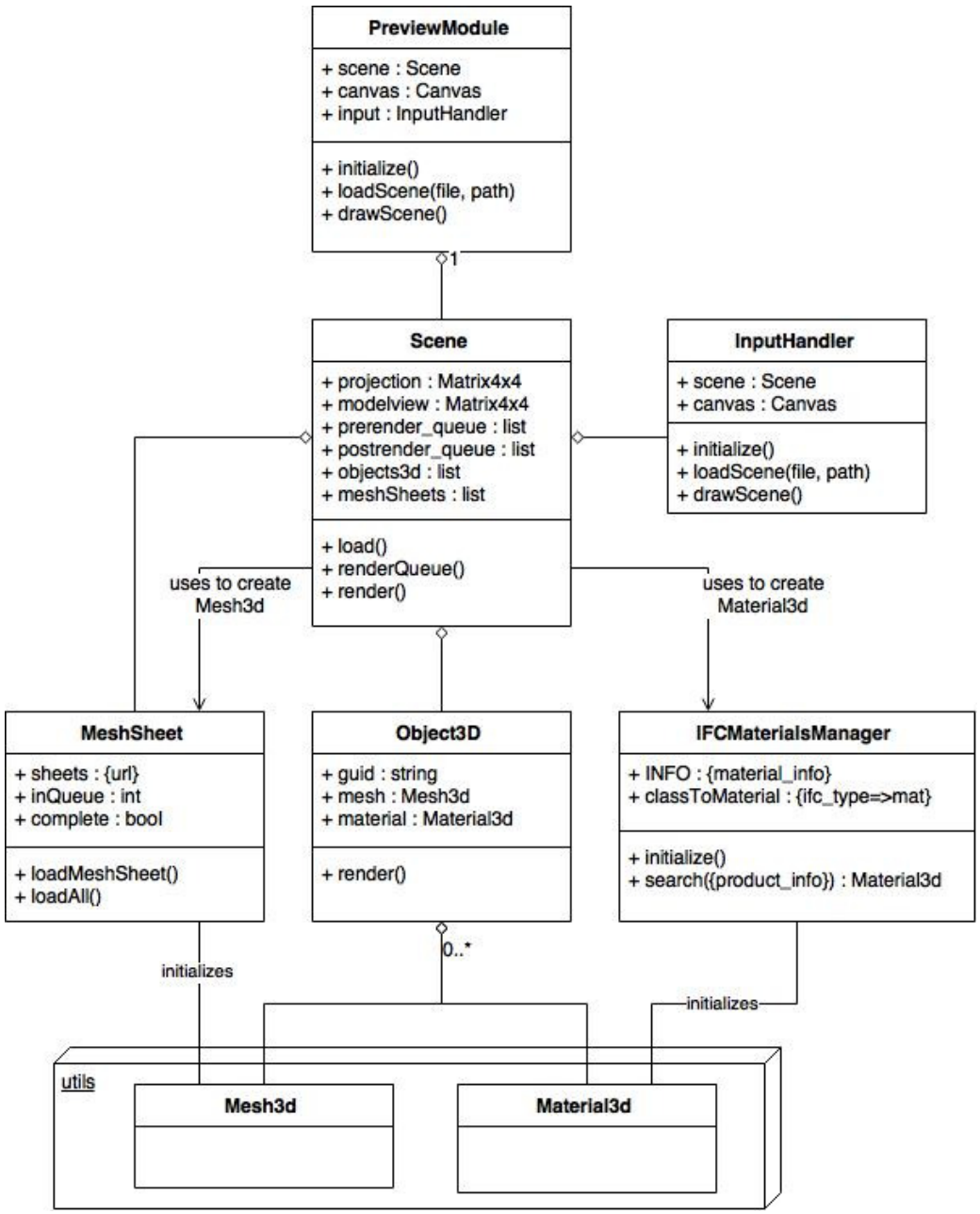

Figure 3.5.2 Diagramme de classes partiel du visualisateur WebGL (PreviewModule)

PreviewModule est la façade au module de visualisation de WebGL et s'occupe surtout de l'initialisation du contexte WebGL.

Scene a comme responsabilité de charger les fichiers de géométrie créés par IfcReader via les MeshSheets, qui sont une abstraction des différents fichiers (partitionnement des produits) qui représentent la géométrie du bâtiment IFC. Les MeshSheets permettent de gérer un nombre concurrent de requêtes au différents fichiers, mais s'assure également de ne pas faire crouler le navigateur sous des milliers de requêtes consécutives.

Scene s'occupe également de déterminer les différents paramètres (matériaux, éclairage, etc) des produits (parfois au travers d'autres classes, comme avec IFCMaterialManager). Cela aurait pu être fait lors de la lecture du fichier IFC, mais il a été décidé de séparer ces tâches. Cela permet entre autre de ne pas avoir à relire tous les fichiers IFC déjà traité dans le cas d'un changement de règle d'affaire au niveau de l'assignation des textures.

IFCMaterialManager doit, à partir de toute l'information d'un produit, de retourner le matériel approprié pour l'affichage. Pour l'instant, on ne se fit qu'au IfcType.

Finalement, bien que la Scene s'occupe de l'affichage des objets et des matériaux via les Object3D, elle n'instancie jamais ces composantes. En effet, ces instanciations sont déléguées respectivement à MeshSheet et à IFCMaterialManager.

CHAPITRE 3

Lecteur IFC/STEP

3.1 Modification du devis initial

Tel que mentionné dans le chapitre précédent, le temps d'exécution beaucoup trop long de l'Asset Unity pour l'importation de la géométrie d'un fichier IFC dans Unity nous a forcé à évaluer d'autres alternatives afin de créer le modèle 3D. Une des solutions envisagée fut la création de notre propre lecteur de fichier IFC. Bien que cette tâche ne fût pas dans le devis initial, nous avons décidé de modifier ce dernier puisque ce lecteur serait également utile pour d'autres projets futurs ainsi que le projet principal *Calculateur ACV*.

3.2 Projets existants

Jusqu'à maintenant, le seul travail qui avait été fait sur ce front au sein du projet de *Calculateur ACV* était un lecteur IFC format XML afin de faire le nettoyage d'un fichier IFC pour y éliminer la géométrie. Nous avons décidé de ne pas utiliser ce travail et de se concentrer sur le développement d'un lecteur de fichier IFC format STEP puisque ce format était celui visé pour le projet final.

Du côté des projets open-source disponibles en ligne, plusieurs lecteurs de fichiers IFC ont étés étudiés, soit STEPcode[12], IfcOpenShell[13] et IfcPlusPlus[14]. Nous avons finalement choisi d'utiliser IFCPlusPlus, en grande partie en raison de son lecteur performant qui tire avantage du parallélisme que pourra offrir nos serveurs, en plus de son modèle IFC bien défini.

3.3 Développement du lecteur IFC/STEP: IfcReader

Un lecteur IFC/STEP, IfcReader, a donc été développé afin de faire la lecture d'un fichier IFC/STEP, en extraire l'information sur la géométrie ainsi que la liste des produits

composant le modèle. Un minimum d'information a été extrait du fichier IFC (soit la liste des produits, leurs noms et leurs GUID, ainsi que leur géométrie), mais le lecteur a été conçu de façon à ce que plus d'information additionnelle puisse facilement être extrait dans le futur en fonction des besoins du projet maître. Le modèle IFC qui est extrait du fichier comporte plus de 1000 types d'entités qui permettent de facilement naviguer au sein de ce modèle pour trouver l'information que l'on recherche.

Tel qu'illustré dans le tableau des résultats de la section 5.1, les performances d'IfcReader sont plus que satisfaisantes et peuvent être encore plus impressionnante si l'on augmente le nombre d'unité de calculs disponibles lors de la lecture puisque cette lecture est effectuée de façon parallèle à l'aide d'OpenMP[15], une API multiplateforme en C++ pour le traitement parallèle à mémoire partagé.

3.4 Collecte de données

Tel qu'expliqué dans le dernier chapitre, il a été décidé dans les dernières semaines de ce projet de se concentrer sur la restauration des fonctionnalités présentes avec le lecteur Unity ainsi que sur le lecteur IFC. IfcReader a donc été modifié à nouveau afin de récolter et enregistrer les informations relatives aux propriétés de chaque élément.

Ces informations sont présentement sauvegardées dans une base de données SQLite dans des tables peu normalisées, mais il serait trivial de changer le schéma de tables et le SGBD utilisé dans les versions ultérieures de l'application.

Parmi les informations sauvegardées, on compte l'information sur les matériaux (les *IfcMaterial* et les *IfcMaterialLayer*) ainsi que les propriétés définies (les *IfcSingleProperty* et les *IfcPropertySet*). Ces informations sont des chaînes de caractères localisées dans le fichier IFC, il reste donc beaucoup de travail à faire pour classifier les différents matériaux dont sont construits les différents produits.

Il était parfois difficile de trouver quelle instance d'une information sauvegarder puisque certaines informations sont dupliquées par les logiciels BIM au sein du fichier IFC. La figure 3.4.1 présente un exemple de schéma partiel type de quelques éléments tels que représentés dans un fichier IFC créé par Autodesk Revit. Bien que "Leichtbeton 300", le matériel dont est conçu les trois produits en bleu, se retrouve plusieurs fois dans le diagramme, chaque information désigne une propriété un peu différente. Selon les spécifications IFC[16], il faudrait utiliser le type dans IfcWallType, et non celle dans IfcMaterialLayerSet. [Note de l'éditeur: Dans le produit tel que remis avec ce rapport, le type enregistré est celui du IfcMaterialLayerSet et non celui de IfcWallType.]

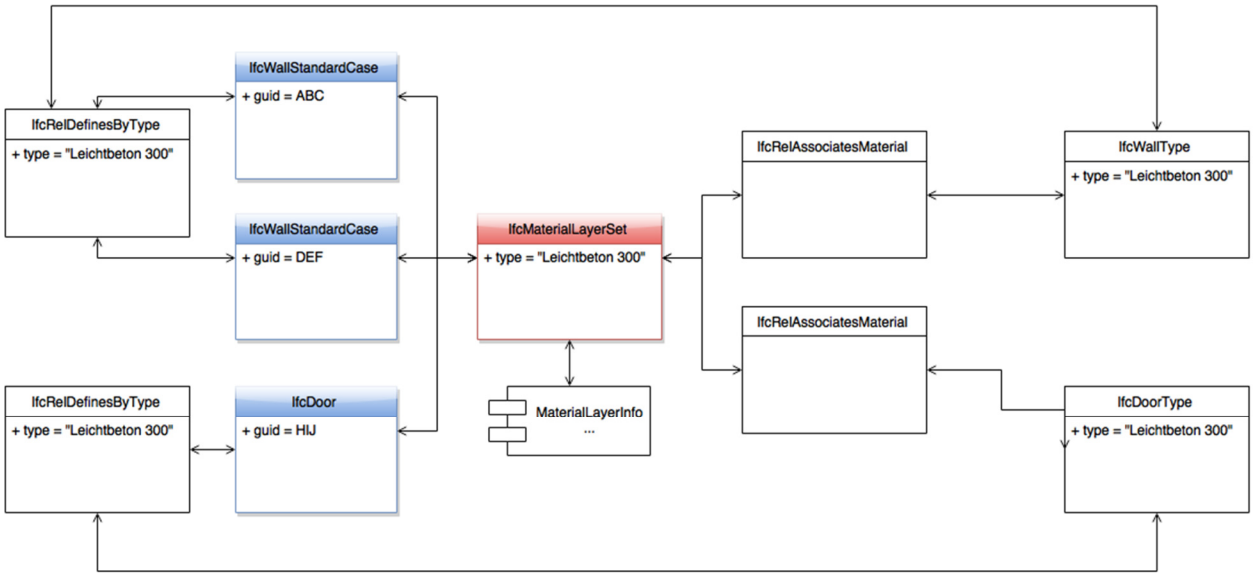

Figure 3.4.1 Exemple de schéma partiel exporté par Autodesk Revit

3.5 Schéma des classes logicielles

La figure 3.5.1 présente un schéma partiel des éléments principaux des différentes classes présentes dans l'application IfcReader. Certaines méthodes et/ou paramètres ont été omis du diagramme pour des raisons de clarté.

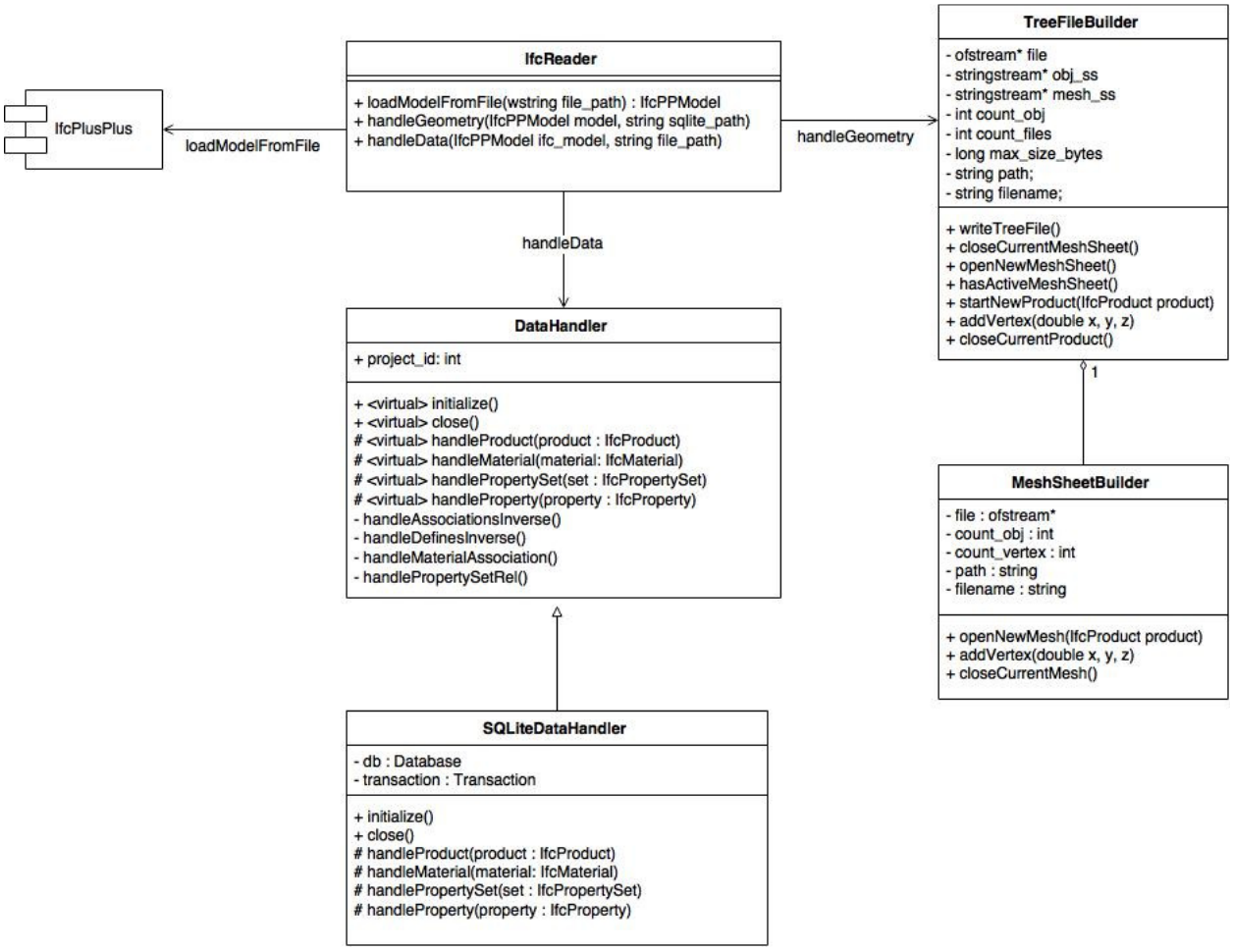

Figure 3.5.1 Diagramme de classes partiel de IfcReader

La classe IfcReader se charge de déléguer les tâches de chargement, de sauvegarde de géométrie et sauvegarde de propriétés à différents modules. Chaque module se trouve dans un dossier différent, soit le projet IfcPlusPlus pour le chargement du fichier IFC, le dossier /geometry pour les tâches reliés à la création des fichiers de géométrie, et le dossier /data pour la recherche et la sauvegarde d'information sur les propriétés des objets définis dans le fichier IFC.

Tel qu'expliqué plus tôt, SQLite peut être changé pour un autre SGBD en créant une autre sous-classe de DataHandler, en définissant les méthodes virtuelles de celle-ci et en instanciant cette nouvelle classe à la place de SQLiteDataHandler dans la méthode IfcReader::handleData présente data/Data.cpp. C'est dans DataHandler que se trouvent les boucles principales de recherche d'information dans le modèle et c'est dans cette classe que devrait être ajouté tout code qui sert à extraire de l'information du fichier IFC (autre que de la géométrie).

Les classes TreeFileBuilder et MeshFileBuilder se chargent de construire les fichiers de géométrie nécessaire à l'application WebGL pour afficher le modèle 3D et ces fichiers ne devraient pas avoir à être vraiment modifiés.

3.6 Diagramme de séquence

La figure 3.6.1 présente la séquence typique de l'exécution de la méthode handleData, qui est celle qui le plus de chance d'être modifiée d'ici la fin du projet de Calculateur AVC.

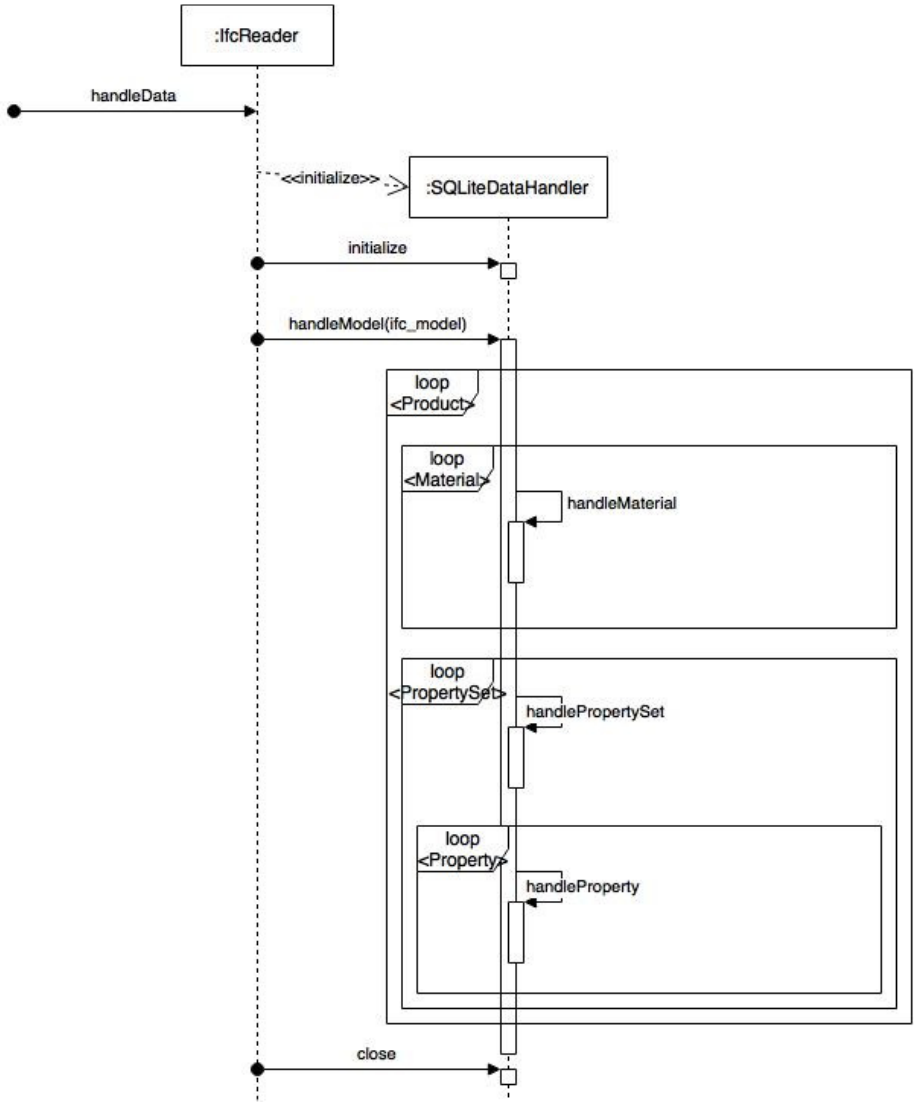

Figure 3.6.1 Diagramme de séquence de la méthode IfcReader::handleData

CHAPITRE 4

Problèmes rencontrés

4.1 Importation du modèle 3D (Performances)

Lors de la construction du premier prototype de la visualisation de modèle 3D avec Unity avec *Esoteric Mesh Importer* (Section 2.3), un des points à améliorer était le temps d'importation du modèle 3D. En effet, sur mon ordinateur personnel muni d'un Intel Core i7-3520M (2.90GHz), l'importation d'un modèle de quelques Mo prenait près de 20 secondes alors que celle d'un modèle de 46 Mo prenait plus de 3 minutes.

Considérant que le modèle IFC typique à nos scénarios d'utilisation pouvait atteindre les 500 Mo, ces performances sont loin d'être satisfaisantes. De plus, *Esoteric Mesh Importer*, ne supporte pas le travail parallèle et peut difficilement être adapté à nos besoins, son code source n'étant pas disponible et ses options de configurations étant très limitées. Il a donc été décidé d'étudier la possibilité de construire nous-même notre importateur IFC.

Tel qu'expliqué dans la section 2.4, cette application maison nous a permis d'augmenter la vitesse à laquelle un fichier IFC est lu, mais n'a pas eu d'impact significatif sur nos performances globales avec Unity. Cependant, l'utilisation de WebGL (section 2.5) nous a permis d'augmenter la performance de la génération et l'importation du modèle 3D par près de 1 000 %, tel que l'on constate dans le tableau 5.1.1.

Grandeur du fichier IFC	***Esoteric Mesh Importer* & Unity**	**IfcReader & Unity**	**IfcReader & WebGL**	**%**
4 Mo AC14-FZK-Haus.ifc	18 s	18 s	~2 s	900 %
46 Mo rme_advanced_sample_project.ifc	200 s	223 s	21 s	950 %
54 Mo Master.ifc	N / A	200 s	18 s	1 111 %

Tableau 5.1.1 Temps d'importation d'un modèle 3D sur un Intel Core i7-3520M (2.90GHz)

CHAPITRE 5

Travaux futurs

5.1 Le visualisateur 3D avec WebGL

Il reste encore du travail à faire pour revenir au point où le visualisateur 3D avec Unity était. Il faudrait entre autre retravailler les contrôles afin qu'ils soient un peu plus naturels, comme avec Unity. Pour se faire, il est possible de réutiliser la même logique que celle défini dans le dernier commit avec Unity, mais les transformations de caméra doivent être converties en opérations matricielles.

De plus, la fonctionnalité de focus sur les éléments sélectionnés, qui avait été implémentée avec Unity, n'a également pas été réimplémentée. Encore une fois, l'algorithme utilisé avec Unity peut être traduit en Javascript et les animations de caméras doivent être converties en manipulations matricielles.

5.2 IfcReader

Tel qu'expliqué plus tôt, le module de géométrie est complet. Cependant, il y aura probablement des modifications à effectuer lors de l'extraction d'information du fichier IFC. Ces modifications peuvent être effectuées facilement en consultant la section 3.5 et 3.6 de ce rapport.

5.3 Visualisation des données

Tel qu'expliqué dans ce rapport, le module de visualisation des données a été laissée de côté afin de se concentrer sur le lecteur IFC. Dans de travaux futurs, une fois les résultats de l'ACV disponible, il faudrait trouver un moyen de représenter ces données de façon claire et intuitive pour un utilisateur type (e.g. architecte ou responsable du développement durable).

Cette interface devra permettre de naviguer dans les résultats de l'ACV afin de visualiser ces données à différents niveaux de granularité (Drill-Down).

5.4 Classification des données

Les données extraites d'un fichier IFC sont très localisées, c'est-à-dire qu'un mur en béton léger pourrait être référé comme étant du "Béton léger", du "Leichtbeton" ou bien du "Lightweight concrete". De plus, la même propriété peut être définie de plusieurs façon, et même au sein du même fichier IFC. Par exemple, la grandeur d'un élément peut être une propriété complexe définie par 3 nombres différents (largeur, hauteur, longueur), ou bien une propriété simple définie par "L x l x h".

Bref, il a beaucoup de travail à faire afin de classifier et normaliser toutes ces données.

CONCLUSIONS

Ce projet a permis le développement d'une interface web qui permet de visualiser le modèle 3D d'un fichier IFC ainsi que le développement d'un lecteur IFC très performant. Le lecteur IFC permet d'extraire toute la géométrie du fichier en plus de toute les propriétés des différents éléments qui constituent le bâtiment défini dans le fichier, et ce, près de 10x plus rapidement que l'Asset Unity qu'il était prévu d'utiliser au départ. De plus, le visualisateur 3D avec WebGL est moderne et rapide et ne risque pas d'être déprécié si tôt (contrairement au lecteur Unity (NPAPI) qui est déjà déprécié sur les dernières versions de Google Chrome).

Cependant, certains points restent à améliorer sur le visualisateur 3D, entre autre les contrôles et certaines transformations (bien que la plupart des fonctions de transformations soient définies).

RECOMMANDATIONS

Les deux modules développés dans ce projet peuvent être intégrés à Calculateur ACV dès maintenant. Le lecteur IFC doit être déployé sur une infrastructure avec plusieurs unités de calculs à mémoire partagée pour profiter le plus possible de l'implémentation avec OpenMP. Le visualisateur 3D avec WebGL est à la fois moderne et performant. Outre les modifications aux contrôles, le visualisateur 3D est également prêt à être intégré à Calculateur ACV. Finalement, les informations qui se trouvent dans la base de données doivent être classifiés, puisque tel qu'expliqué plus tôt, elles ne sont pas normalisées et sont localisées (e.g. Leichtbeton et le béton léger sont deux matériaux différents pour IFC).

LISTE DE RÉFÉRENCES

[1] **ISO 16739:2013**, Industry Foundation Classes (IFC)

Consulté le 10 mai 2015

http://www.iso.org/iso/catalogue_detail.htm?csnumber=51622

[2] **Calculateur LCA** Version : 0.0

Consulté le 10 mai 2015

https://drive.google.com/file/d/0ByNBuxDzH224M0ZEUXRZZDhQbFk/view?usp=sharing

[3] **Unity 3D**, Unity Web Player

Consulté le 10 mai 2015

https://unity3d.com/webplayer

[4] **Wikipedia**, NPAPI

Consulté le 10 mai 2015

http://en.wikipedia.org/wiki/NPAPI

[5] **The Chromium Project**, NPAPI deprecation: developer guide

Consulté le 10 mai 2015

https://www.chromium.org/developers/npapi-deprecation

[6] **Chrome Help**, NPAPI plugins don't work on Chrome version 42 and higher

Consulté le 10 mai 2015

https://support.google.com/chrome/answer/6213033?hl=en

[7] **Unity 3D**, Esoteric Mesh Importer

Consulté le 15 mai 2015

https://www.assetstore.unity3d.com/en/#!/content/15428

[8] **Unity 3D**, Esoteric Mesh Importer
Consulté le 25 mai 2015
http://docs.unity3d.com/Manual/CommandLineArguments.html

[9] **GitHub**, ConstructionLCA
Consulté le 3 août 2015
https://github.com/mbinette91/ConstructionLCA/blob/5f11ba3be25b425d8b0be780ee971f088852258d/IFCViewer/Assets/AutomatedScript.cs

[10] **GitHub**, ConstructionLCA
Consulté le 3 août 2015
https://github.com/mbinette91/ConstructionLCA/blob/5f11ba3be25b425d8b0be780ee971f088852258d/UnityBuilder.py

[11] **Wikipedia**, WebGL
Consulté le 27 juin 2015
https://en.wikipedia.org/wiki/WebGL

[12] **STEPcode**, Home
Consulté le 8 juin 2015
http://stepcode.org/

[13] **IfcOpenShell**, Home
Consulté le 15 juin 2015
http://www.ifcopenshell.org/

[14] **IfcPlusPlus**, Home
Consulté le 15 juin 2015
http://www.ifcplusplus.com/

[15] **OpenMP**, Home
Consulté le 18 juin 2015
http://openmp.org/wp/

[16] **Building Smart**, IfcWallType
Consulté le 2 août 2015
http://www.buildingsmart-tech.org/ifc/IFC2x3/TC1/html/ifcsharedbldgelements/lexical/ifcwalltype.htm

AUTRES RÉFÉRENCES CONSULTÉES

Wikipedia, Life-Cycle Assessment

Consulté le 3 août 2015

https://en.wikipedia.org/wiki/Life-cycle_assessment

Wikipedia, IFC

Consulté le 3 août 2015

https://en.wikipedia.org/wiki/IFC

Wikipedia, BIM

Consulté le 3 août 2015

https://en.wikipedia.org/wiki/Building_information_modeling

Wikipedia, WebGL

Consulté le 27 juin 2015

https://en.wikipedia.org/wiki/WebGL

Wikipedia, OpenMP

Consulté le 22 juin 2015

https://en.wikipedia.org/wiki/OpenMP

ANNEXE I

Comptes rendus hebdomadaires

Semaine du 6 mai 2015

Ce qui a été fait

- Squelette d'interface web avec tabulations, un Unity Web player (WebGL) avec communication deux sens et un mode "plein écran".

Élément(s) bloquant(s)

- Google Chrome ne supporte plus Unity WebPlayer avec la configuration par défaut (http://answers.unity3d.com/questions/833168/why-wont-unity-web-player-work-in-chrome.html)
    - Solution: chrome://flags/#enable-npapi (+relaunch)
    - Problème potentiel si cette tendance se propage?
- Unity décharge la scène courante lorsqu'elle est cachée dans le navigateur, ce qui rend impossible la communication avec le player lorsqu'on est sur le tab "Statistiques" par exemple.
    - http://forum.unity3d.com/threads/unity-flash-div-issue-in-browser.22270/
    - Solutions possibles:
        - Utiliser `visibility` à la place
        - Redimensionner à 0x0 (ou 1x1) - la scène n'est pas relancée lors d'une redimension
        - Utiliser un iframe
- Ne peut pas superposer d'éléments par-dessus le visualisateur.
    - Est-ce que c'est aussi le cas avec un iframe (ça ne devrait pas…)?
    - Est-ce que c'est vraiment un problème ?

Plan d'action

- 1- Régler le problème du Unity Webplayer qui se décharge lorsqu'il est caché
- 2- Charger un fichier IFC avec Unity
- 3- Regarder les textures IFC avec Unity, sinon juste le C# DLL. Textures Revit → Bitmap (Semaine 1 et 2)
    - Checker des modeles IFC sur internet avec textures parce que ceux de Mathieu ont pas de textures

Semaine du 11 mai 2015

Ce qui a été fait

- 1- Le problème de Unity Webplayer qui se décharge lorsque caché est réglé
    - Jouer avec la visibilité au lieu du display marche très bien, et combiné avec un petit hack de size, la scène reste chargée lorsque cachée, et en plus on peut continuer d'interagir avec.
        - Est-ce vraiment une bonne chose? Si le modèle est très lourd et que l'utilisateur ne veut pas utiliser le preview mais veut seulement consulter les statistiques, on utilise beaucoup de mémoire pour rien.
- 2- Chargement d'un fichier IFC avec Unity
    - Le asset de probesys (Esoteric Mesh Importer) nécessite Unity Pro[1].
        - En attendant une solution légitime pour la v5, j'ai la version pro de v4.5.
    - On peut bouger la caméra sur le preview (à améliorer..!).
- 3- Investigation de l'importation des textures dans Unity (fichiers IFC)
    - Les fichiers IFC supportent les textures[2], mais...
        - *"Right now, **IFC import/export implementation for ALL vendors doesn't support colors or textures**, as there has not been clear guidelines or implementers agreements to support this exchange requirement. **IFC can technically support it**, though it isn't easy. Today, **IFC compatible application set their own color schemes**, if any, for IFC files on import."* [3]
            - 2011, mais la situation semble encore la même aujourd'hui
    - Revit (la version de Mathieu) n'exporte pas les textures dans un fichier IFC.
        - Revit utilise le même IFC Loader que Naviswork, qui devrait maintenant supporter l'importation des couleurs (depuis 2014), mais l'exportation?[4].
            - Quelle version Mathieu utilise?
    - L'asset (Unity) utilisé supporte l'importation de texture[5].
    - Bref, l'asset supporte les textures, mais pas le logiciel qui exporte!
        - Regarder ailleurs que Revit ou on attend après eux?

Élément(s) bloquant(s)

- Version pro (légitime) de Unity pour le asset d'importation IFC.

Plan d'action

- 1- Intégration de base de d3js (Semaine 2)
- 2- Afficher les composantes IFC dans le tree (Semaine 2)

---

[1] http://forum.unity3d.com/threads/released-esoteric-mesh-importer.232290/#post-1545098

[2] http://www.buildingsmart-tech.org/ifc/IFC4/final/html/schema/ifcpresentationappearanceresource/lexical/ifcsurfacetexture.htm

[3] https://techboard.vectorworks.net/ubbthreads.php?ubb=showflat&Number=162486

[4] http://autode.sk/1A0lCkN

[5] http://forum.unity3d.com/threads/released-esoteric-mesh-importer.232290/#post-1634300

Semaine du 18 mai 2015

Ce qui a été fait

- 1- Intégration de base de d3js avec un fichier de données bidon
    - Un peu de travail sur la redimension automatique, mais encore TRÈS buggy
- 2- Afficher les composantes IFC dans l'arbre
    - Affichage d'un arbre contenant toutes les composantes de l'objet Unity
        - Il ne contient par contre pas les autres informations du fichier IFC.
            - Il va falloir un autre service qui fait l'association nom => autres information IFC
    - Intégration avec jsTree[6] (avec icônes de statut)
    - Intégration des boîtes à cocher avec Unity (cacher/montrer certains éléments)
- Travail sur la navigation au sein du modèle 3D
    - Plus intuitive maintenant, mais pas encore satisfaisant...

Élément(s) bloquant(s)

- * Importation dynamique des IFC possible ?
    - Pas selon l'auteur du plugin… [7] On peut peut-être hack quelque chose…
- * Est-ce que le module d'import garde la même arborescence (et mêmes noms) que le module IFC?
    - Noms: pas tout à fait, mais rien d'insurmontable - Arborescence: À vérifier.
- Problèmes de navigation/contrôles
    - On veut comme Revit ou une navigation plus automatique en cliquant sur des composantes et en limitant les contrôles à rotations et zooms ?
    - Désactiver le clic droit sur le lecteur web Unity
- Problèmes de redimension avec D3JS (& mode fullscreen)
    - Chaîne de redimension → window, layout, mon app, d3js.

Plan d'action

- 1- Améliorer le visualisateur Unity
    - 1.1 - Lors d'un clic sur un élément, a) ne pas le décocher, b) zoomer sur cet élément et rendre le reste du modèle semi-transparent.
    - Rendre la navigation un peu plus comme Revit
    - Double-cliquer sur un élément du modèle = sélectionner dans l'arbre et bouger avec la même vue que 1.1
- 2- Importation avec le script pour l'éditeur Unity
    - Créer une scène server-side et script de traitement post-importation
    - Regarder la structure IFC: http://www.solibri.com/products/solibri-model-viewer/
        - Importer la hiérarchie? Les IDs?

---

[6] http://www.jstree.com/

[7] http://forum.unity3d.com/threads/released-esoteric-mesh-importer.232290/#post-1634300

Semaine du 25 mai 2015

Ce qui a été fait

- 1- Améliorer le visualisateur Unity
    - 1.1 - Lors d'un clic sur un élément, a) ne pas le décocher, b) zoomer sur cet élément et rendre le reste du modèle semi-transparent.
    - Double-cliquer sur un élément du modèle = sélectionner dans le tree et bouger avec la même vue que 1.1
- 2- Importation avec le script pour l'éditeur Unity
    - Créer une scène server-side et script de traitement post-importation
        - Pas grand chose qui se fait post-importation pour l'instant autre l'ajout de colliders et changement de shader pour le Preview.
        - Création d'un nouveau projet temporaire
            - Avantages et inconvénients discutés dans le rapport
- Commencement du rapport de mi-session

Élément(s) bloquant(s)

- Support de build.py uniquement sur Windows.
    - Le plugin d'importation utilisé ne supporte que Windows de toute façon..[8]
- Problèmes de navigation/contrôles
    - Un peu plus comme Revit
    - Désactiver le clic droit sur le lecteur web Unity
    - Le positionnement de la caméra est étrange en mode OnFocus
- Problèmes de redimension avec D3JS (& mode fullscreen)
    - Chaîne de redimension → window, layout, mon app, d3js.

Plan d'action

- 1- Améliorer le visualisateur Unity
    - Améliorer le OnFocus mode: Center point of object, heuristiques pour déterminer le "sens" de l'objet (largeur vs profondeur, etc).
    - Rendre la navigation Revit-like
- 2- Importation avec le script pour unity editor
    - Créer une scène server-side et script de traitement post-importation
        - Ajouter un comportement custom (avec center point pour le positionnement OnFocus)
    - Regarder la structure IFC: http://www.solibri.com/products/solibri-model-viewer/
        - Importer la hierarchie? Les IDs?
- 3- Avancer le rapport pour la remise de mi-session

[8] https://www.assetstore.unity3d.com/en/#!/content/15428

Semaine du 1 juin 2015

Ce qui a été fait (commit: 5f11ba3be25b425d8b0be780ee971f088852258d)

- Intégration du système de build automatique avec le reste de l'application web
    - Formulaire de création de "projets" et (re)set-up de l'environnement de développement Unity intégré avec le nouveau système de build automatique.
- Refactoring webapp et Unity (UnityScript => C#)
- 1- Améliorer le preview Unity
    - Améliorer le OnFocus mode ("sens" de l'objet, distance)...
        - Vraiment nice heuristiques (double projection + ajustement zoom), mais pas parfait (petites pièces VS grosses pièces, donuts, devant-derrière)
    - Fix des bugs de sélection
    - Revit-ish navigation
- 2- Importation avec le script pour Unity Editor
    - Ajout d'un comportement custom pour chaque composante
        - Centroide, grandeur, etc… (pour OnFocus)
    - Hierarchie comme dans Solibri
- 3- Avancer le rapport pour la remise de mi-session

Élément(s) bloquant(s)

- Backend
    - Django? Rails? Autre? - MVC5
- Système de contrôle d'accès
    - On fait quoi avec ça ? - Hors scope.
- Modèle de projet
    - 1 ou plusieurs IFC par projet? C'est quoi exactement un projet? Versioning?
        - Pour l'instant, 1 IFC = 1 projet
        - Hors portée
- Génération du Preview Unity
    - Environ 3 mins pour un modèle IFC de 45mb.
    - Marche pas pour Master.ifc (???) - à investiguer. seulement 50mb
    - Distribution difficile
- Le navigateur gèle quand on charge des gros previews (Load de la scène? GetTree?)
- Autobuild seulement sur Windows (le plugin d'importation supporte que Windows[9])
- Problèmes de redimension avec D3JS (& mode fullscreen) (voir semaine passée)
- Bug OnFocus avec Revit controls (enlever le precomputing (?))

Plan d'action

- 1- Faire une application externe pour le parsing de fichier IFC/Step
- 2- Améliorer les performances d'importation de modèle 3D pour Unity
- x- Vérifier le modèle IFC sur le drive qui manque un mur dans Unity (???)

[9] https://www.assetstore.unity3d.com/en/#!/content/15428

Semaine du 8 juin 2015

Ce qui a été fait (commit:)

- 1- Faire une application externe pour le parsing de fichier IFC/Step
    - Avec STEPcode
    - Work in progress
- 3- Avancer le rapport pour la remise de mi-session
- x- Vérifier le modèle IFC sur le drive qui manque un mur dans Unity (???)
    - Le mur est "là", mais il manque des points (Pourquoi?)

Élément(s) bloquant(s)

- Génération du Unity Preview lente
- Browser hang quand on charge des gros previews (Load de la scène? GetTree?)
- Autobuild seulement sur Windows (le plugin d'importation supporte que Windows[10])
- Problèmes de redimension avec D3JS (& mode fullscreen) (voir semaine passée)
- Bug OnFocus avec Revit controls (enlever le precomputing (?))

Plan d'action

- 1- Faire une application externe pour le parsing de fichier IFC/Step
- 2- Améliorer les performances d'importation de modèle 3D pour Unity
- Rechercher des options open source pour la lecture performante de fichiers IFC/STEP
    - Bonus si pas trop primitive

Semaine du 15 juin 2015

Ce qui a été fait (commit:)

- Recherche d'alternatives à STEPcode qui n'est pas assez performant à mon goût et très primitif.
    - Choix de IFCPlusPlus en raison de son parallélisme et de sa performance
- Configuration de IFCPlusPlus

Élément(s) bloquant(s)

- Génération du Unity Preview lente
- Le navigateur gèle quand on charge des gros previews (Load de la scène? GetTree?)
- Autobuild seulement sur Windows (le plugin d'importation supporte que Windows[11])
- Problèmes de redimension avec D3JS (& mode fullscreen) (voir semaine passée)
- Bug OnFocus avec Revit controls (enlever le precomputing (?))

Plan d'action

- 1- Développement d'un lecteur IFC de base performant

[10] https://www.assetstore.unity3d.com/en/#!/content/15428

[11] https://www.assetstore.unity3d.com/en/#!/content/15428

Semaine du 22 juin 2015

Ce qui a été fait (commit:)

- 1- Développement d'un lecteur Ifc/STEP de base à partir de IfcPlusPlus
    - Clean up des dépendances non-pertinentes pour notre projet (e.g. Qt, OpenSceneGraph, ...)
        - Reste: Boost (et support de OpenMP par le compilateur)
- Setup d'une mini base de données SQLite (temporaire) pour
    - GUID, Nom_Produit
- Préparation du code pour le stockage de plus d'information (matériaux, …) au besoin
- Parsing 3-10sec pour un fichier de 50MB (pas de création de modèle 3D).
    - C'est un bon pas en avant pour l'instant.

Élément(s) bloquant(s)

- Génération du Unity Preview lente
- Le navigateur gèle quand on charge des gros previews (Load de la scène? GetTree?)
- Autobuild seulement sur Windows (le plugin d'importation supporte que Windows[12])
- Problèmes de redimension avec D3JS (& mode fullscreen) (voir semaine passée)
- Bug OnFocus avec Revit controls (enlever le precomputing (?))

Plan d'action

- 1 - Génération de la géométrie/modèle 3D à partir du nouveau parseur Ifc

[12] https://www.assetstore.unity3d.com/en/#!/content/15428

Semaine du 29 juin 2015

Ce qui a été fait (commit: cccfc3662f8034a17ec95da1f9ca8538c949e1df)

- 1 - Génération du modèle 3D à partir du nouveau parseur IFC/STEP.
    - Création d'un fichier JSON de géométrie
        - Pas le format idéal, pas compressé, …
    - Chargement de ce fichier dans Unity: Pas vraiment plus concluant que le EMI.
- x - Switch à WebGL
    - Reverse engineering du JS de FinalMesh[13] (logiciel payant)
        - Pas fait à l'origine pour gérer des milliers de petites pièces
            - Ajout de fonctionnalités pour gérer ça
    - Il n'y a plus de *hanging* lors du chargement du modèle non plus.
    - L'AutoBuild est maintenant (techniquement) multiplateforme (mais pas testé…)
    - Les nouvelles performances beaucoup plus satisfaisantes.

Grandeur du fichier IFC	***Esoteric Mesh Importer* & Unity**	**Lecteur IFC/STEP & Unity**	**Lecteur IFC/STEP & WebGL**	**%**
4 Mo AC14-FZK-Haus.ifc	18 s	18 s	~2 s	900 %
46 Mo rme_advanced_sample_project.ifc	200 s	223 s	21 s	950 %
54 Mo Master.ifc	N / A	200 s	18 s	1 111 %

Élément(s) bloquant(s)

- Lecteur IFC/STEP - IfcReader
    - Il manque certains éléments (Pourquoi?) Ils sont parsés, j'ai juste pas de géométrie.
- Perte de tout le travail sur Unity
    - Textures, Contrôles, Arborescence et Sélection d'éléments
- Problèmes de redimension avec D3JS (& mode fullscreen)
    - Possiblement le même problème avec WebGL (à vérifier)

Plan d'action

- 1- Réécrire le script JS
- 2- Lecteur IFC/STEP - IfcReader
    - Il manque certains éléments (Pourquoi?) Ils sont parsés, j'ai juste pas de géométrie.
- 3- Perte de tout le travail sur Unity
    - Textures, Contrôles, Arborescence et Sélection d'éléments

[13] http://www.finalmesh.com/webglmore.htm

Semaine du 6 juillet 2015
Semaine du 13 juillet 2015

Ce qui a été fait (commit:fd289fbd47ccd85c8b4962aa972441aa89072219)

- 1 - Réécrire le script JS
    - Refactoring des classes utils (vec3, mat3 et mat4) pour WebGL dans /js/utils.webgl.js
    - Refactoring de la façon dont les textures sont chargées/déterminées.
    - Work in progress
- 2- Investigation
    - Ils semblent être définis pareils.
        - Entités à vérifier:
            - IfcShellBasedSurfaceModel depuis IFC 2.x
            - IfcOpenShell/ClosedShell
        - Tous les SurfaceModel sont des entités manquantes:
            - shape_data->m_representation_type == SurfaceModel
            - + 2 MappedRepresentation qui sont habituellement OK.
- 3- Refaire le travail fait sur Unity
    - Refait le calcul des normales, textures encore mieux qu'avec Unity.
    - Affichage de la liste des composantes (arbre à un niveau)
    - Work in progress

Élément(s) bloquant(s)

- Lecteur IFC/STEP - IfcReader
    - Il manque certains éléments (Pourquoi?) Ils sont parsés, j'ai juste pas de géométrie.
- Perte de tout le travail sur Unity
    - Contrôles, Arborescence à plusieurs niveaux et Sélection d'éléments

Plan d'action

- Réécrire le script JS
- Lecteur IFC/STEP - IfcReader
    - Il manque certains éléments (Pourquoi?) Ils sont parsés, j'ai juste pas de géométrie.
- Perte de tout le travail sur Unity
    - Contrôles, Arborescence à plusieurs niveaux et Sélection d'éléments

Semaine du 20 juillet 2015

Ce qui a été fait (commit:0d787c6..7aa3a1d)

- 1 - Réécrire le script JS
    - Work in progress
- 2- Il manque certains éléments de géométrie
    - Il peut y avoir de la géométrie sur des IfcAnnotation[14].
        - Pas dans notre cas.
    - Les éléments sont tous là, mais la géométrie n'est pas définie pour certains.
        - Définition OK dans le fichier IFC.
        - Problème avec IfcShellBasedSurfaceModel[15]. Les seuls 16 éléments manquants sont reliés aux 16 instances de cette instruction dans mon fichier de test.
            - IfcOpenShell au lieu de IfcClosedShell pour la géométrie qui ne fonctionne pas (?)
                - Bingo, sont dans les meshset_open.
- 3- Refaire le travail fait sur Unity
    - Sélection d'élément en cliquant et via l'arborescence
        - Dans les deux sens.
        - Fenêtre pour display l'information sur l'élément sélectionné
- x- Ajout des matériaux/layersets de matériaux dans la BD (dénormalisé)
    - Extraction dans IfcReader et affichage dans ConstructionLCA

Élément(s) bloquant(s)

- Perte de tout le travail sur Unity

Plan d'action

- Réécrire le script JS
- Amélioration de l'apparence de la page de création de projet
- Perte de tout le travail sur Unity
    - Contrôles
    - Arborescence à plusieurs niveaux
        - Par classe et nom du type

[14] http://www.buildingsmart-tech.org/ifc/IFC2x4/beta3/html/ifcproductextension/lexical/ifcannotation.htm

[15] http://www.buildingsmart-tech.org/ifc/IFC2x3/TC1/html/ifcgeometricmodelresource/lexical/ifcshellbasedsurfacemodel.htm

Semaine du 27 juillet 2015

Ce qui a été fait (commit:0d787c6..7aa3a1d)

- 1 - Réécrire le script JS
    - Work in progress
- 2 - Amélioration de l'apparence de la page de création de projet
- 3 - Refaire le travail fait sur Unity
    - Arborescence à plusieurs niveaux
        - Par type et type matériel
        - Sélection d'un niveau au complet
    - Sélection multiple
        - Affichage de l'information commune (si possible)
    - Contrôles
        - Incomplet
- IfcParser
    - Sauvegarde de l'information des IFCPROPERTYSINGLEVALUE
        - Affichage de cette information sur l'interface

Élément(s) bloquant(s)

-

Plan d'action

- Préparer la présentation orale de ce projet.

ANNEXE II

Comment compiler IfcReader

Pour Windows seulement (avec Visual Studio 2013).

Prérequis:

- Boost
- OpenSceneGraph (testé avec 3.2)

Instructions:

1. Configurer les chemins dans set_devenv64-VS2013.cmd (ou set_denv32-VS2013.cmd) et exécuter ce script pour ouvrir Visual Studio.
2. Ouvrir la solution `IfcPlusPlus`.
3. S'assurer que le mode de compilation est en mode `Release`.
4. Compiler le projet `IfcReader` (ça peut être très long, particulièrement la première fois).

* Ne pas oublier de copier le fichier .exe résultant (dans /IfcReader/bin[32]/IfcReader.exe) dans le dossier `bin` du projet web.

ANNEXE III

Comment exécuter le serveur Web

Prérequis:

- Python 2.7

Instructions:

1. Compiler IfcReader (voir Annexe 2)
    a. Ne pas oublier de copier le fichier .exe dans le dossier `bin` du projet web.
2. Copier les DLL suivantes de OpenSceneGraph dans le dossier `bin` du projet web
    a. osg100-osg.dll
    b. osg-osgDB.dll
    c. osg-osgText.dll
    d. osg-osgUtil.dll
    e. ot20-OpenThreads.dll
3. Exécuter webapp.py