RAPPORT TECHNIQUE
PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
DANS LE CADRE DU COURS
GTI795 PROJET DE FIN D'ÉTUDES EN GÉNIE LOG/TI

UBUBI

Nicolas Bastien BASN23069204
Guillaume Forgues FORG19019100
André Koolen KOOA23039101
Edward Le Barbenchon LEBE08049208

DÉPARTEMENT DE GÉNIE LOGICIEL ET DES TI

Professeur-superviseur
Alain April

MONTRÉAL, 21 DÉCEMBRE 2017
AUTOMNE 2017

REMERCIEMENTS

Nous tenons à remercier Mathieu Dupuis et Alain April pour nous avoir accompagné tout au long du trimestre. Particulièrement Mathieu pour son temps, son calme et ses explications de l'application, car sans sa contribution il en aura été autrement.

RÉSUMÉ

Ce rapport présente l'évolution du trimestre d'automne, de l'année 2017, du projet UBUBI. Réalisé dans le cadre d'un projet de fin d'études à ÉTS et en collaboration avec l'université Polytechnique de Montréal. Ce projet vise à développer un logiciel qui permet de calculer l'impact environnemental d'un bâtiment à partir de données numériques d'un modèle en trois dimensions. Durant cette période de quatre mois, nous avons ajouté des fonctionnalités et améliorer l'expérience utilisateur de l'application. Ce qui nous a permis de travailler sur presque tous les aspects existants de l'application.

Tout le projet sera détaillé en plusieurs sections: une introduction au projet, la description des objectifs, une analyse de la méthodologie utilisée, la contribution détaillée des modifications et ajouts de fonctionnalités, les risque ainsi que les problèmes rencontrés. Il se terminera par un bilan du projet qui contiendra les recommandations de l'équipe et finalement une conclusion.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

HTML	HyperText Markup Language
CSS	Cascading Style Sheets
JS	Javascript
OLCA	Open Life Cycle Assessment
Git	Logiciel de gestion de version décentralisé
JSON	JavaScript Object Notation
SQL	Structured Query Language
Ajax	Asynchronous JavaScript And XML
BD	Base de données

INTRODUCTION

Le domaine de la construction a un impact très élevé sur l'environnement. De plus en plus de personne et d'entreprise accordent une grande importance à l'impact environnemental de leurs projets.

Présentement, lors de nouveaux travaux, il est très difficile d'observer et d'analyser l'impact réel que le bâtiment aura. Le but de ce projet est de rendre disponible aux gestionnaires de projet dans le domaine de la construction la possibilité de recenser tous les impacts environnementaux qu'auront leurs bâtiments.

UBUBI pourra analyser l'impact réel d'un nouveau bâtiment. Pour se faire, le programme devra être capable de : créer, sauvegarder, modifier et supprimer des modèles. Ces modèles représentent la structure qui sera étudiée pour développer une analyse d'impact environnemental. Pour faciliter la représentation graphique des matériaux, un arbre de processus est créé. L'utilisateur peut interagir avec l'arbre pour mieux comprendre la séparation des objets d'un bâtiment dans le plan, les processus, les codes uniformat, le type de matériel et les matériaux. Chaque processus contient également son l'impact environnemental propre ainsi que la somme de l'impact de tous ses enfants.

L'analyse de l'impact environnemental pourra aussi être effectuée après un changement à un modèle de bâtiment. De cette façon, les responsables pourront diminuer l'impact environnemental du bâtiment.

L'explication du travail accompli durant les derniers mois sera expliquée plus en profondeur et séparée par des chapitres pour une compréhension plus claire.

Chapitre 1 : Problématique et contexte

UBUBI est un logiciel d'analyse d'impact environnemental qui sera utilisé par des universités comme l'ÉTS et la Polytechnique ainsi que par des entreprises dans le domaine de la construction. Cette application a pour but d'offrir aux gestionnaires de projet la possibilité de facilement effectuer le suivi des impacts environnementaux d'un bâtiment et ce tout au long de son cycle de vie. Pour ce faire, l'application calcul, selon un ensemble de données, l'impact de chaque partie d'un bâtiment et agrège le tout pour l'exporter par d'autre logiciel de conception trois dimensions comme Revit. Le projet UBUBI est de grande envergure et dans le but de remplir notre mandat à temps, il a fallu restreinte la taille du projet à quelque nouvelle fonctionnalité portant sur des aspects avec lesquels les membres de l'équipe étaient familiers. Il est important de noter que UBUBI est déjà fonctionnel ce qui veut dire que la majorité du travail fait par l'équipe devra maintenir le bon fonctionnement existant.

Chapitre 2 : Objectifs du projet

L'objectif de la session d'Automne 2017 était de réaliser et d'ajouter des fonctionnalités tout en gardant l'intégrité de l'application. D'en apprendre plus sur les façons de faire, sur les technologies utilisées et expérimenter une situation d'entreprise lors d'un développement d'application.

UBUBI devra analyser l'impact environnemental d'un nouveau bâtiment. Pour se faire, le programme devra être capable de : créer, sauvegarder, modifier et supprimer des modèles. Les modèles représentent la structure qui sera étudiée pour développer une analyse d'impact environnemental. Pour aider la compréhension de l'utilisateur, il est utile d'avoir une représentation graphique des éléments, mais il est nécessaire de créer un arbre de modèle contenant tous les processus représentant les objets, les types d'objets, les matériaux utilisés, leurs quantités, leurs enfants, etc. Le tout trié dans l'arbre de processus. Toutes ces informations sont nécessaires pour pouvoir effectuer une analyse et un calcul complets. Chaque processus contiendra également l'impact environnemental du processus ainsi que l'impact de tous ses enfants.

L'analyse de l'impact environnemental pourra aussi être effectuée après la mise à jour d'un modèle de bâtiment. De cette façon, les personnes responsables pourront diminuer, en prenant de nouvelles décisions, l'impact environnemental du bâtiment.

De plus, l'application devra être facile d'utilisation c'est-à-dire que les interfaces de l'application devront prendre compte les utilisateurs, de leur habitude devant la technologie et doit également faciliter l'affichage d'une grande quantité d'information de façon simple et claire.

Chapitre 3 : Méthodologie

Pour atteindre ces objectifs, l'équipe se rencontre itérativement le jeudi à chaque semaine. Nous avons aussi créé un dossier de partage Google Drive pour toute l'information concernant l'application. Nous utilisons Slack, une application de communication, pour faciliter les interactions entre les membres de l'équipe et le client. De cette façon, chaque membre de l'équipe peut travailler et communiquer avec les autres au moment qui lui est le plus approprié.

Concernant la documentation des tâches, un tableau Trello a été créé pour faciliter le suivi et y documenter les tâches lors du développement. Ce qui permet plus facilement de comprendre et faire un suivi rapide du développement. Cela servira aussi pour le client pour y ajouter des nouvelles fonctionnalités pour les prochaines itérations.

Durant les réunions hebdomadaires, nous discutons des tâches complétées, des blocages et problèmes rencontrés au cours de la semaine. Puis, nous planifions les prochaines tâches pour la semaine suivante. Une fois qu'un développeur termine une tâche, le client valide le travail accompli et nous retourne ses commentaires au besoin. Nous devons auparavant avoir testé et documenté les tâches, car c'est de cette façon que le client sera en mesure de vérifier si cela correspondait aux besoins.

Chapitre 4 : Description des changements accomplis

4.1 Modification de la base de données

Comme décrit précédemment, mandat était de continuer le développement d'une application existante. Il existait déjà une structure de données contenu dans des tables. L'objectif de cette tâches est d'ajouter les trois tables *AL_Material*, *AL_ObjectType* et *LCADatabase*. Ces tables seraient utilisées pour avoir un concept de structure pour les liaisons automatiques. Pour plus de détails voir la figure 1 qui montre les liens entre les tables.

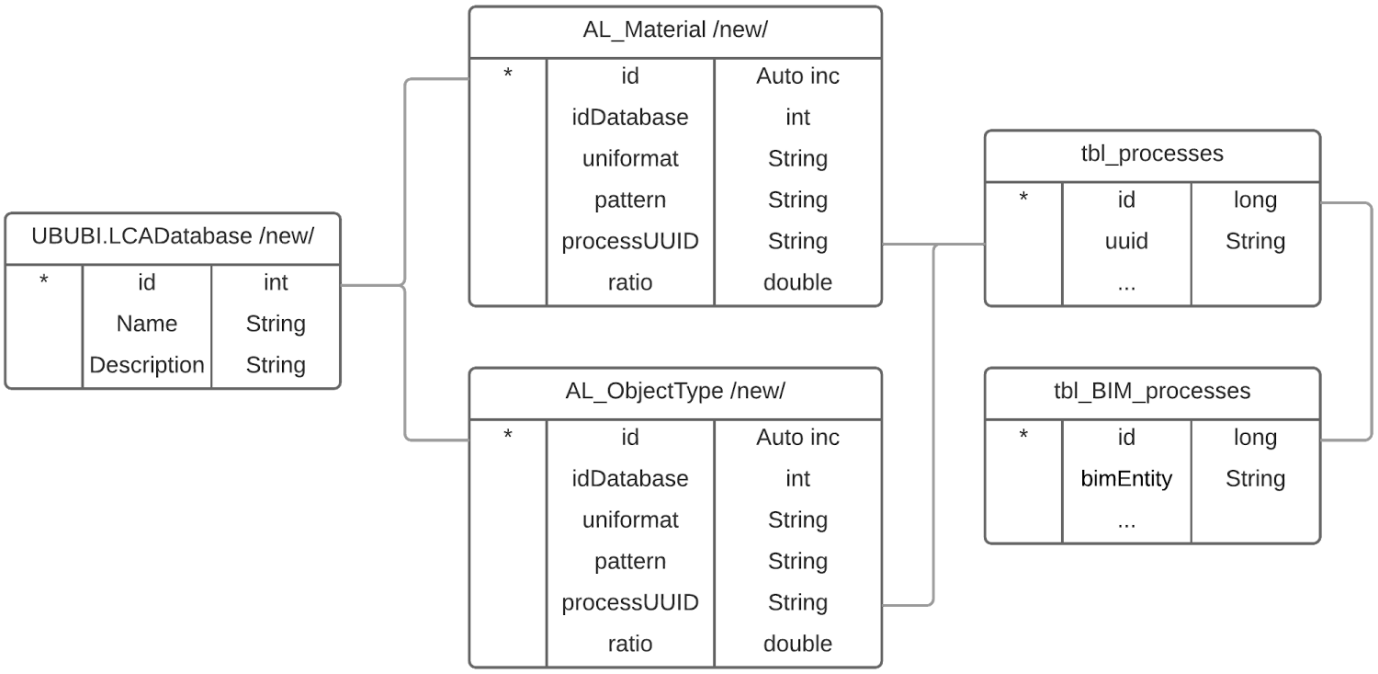

Figure 1: Tables ajoutées

Comme il est illustré à la figure 1 les tables sont liée par des clés étrangères tel que l'ID pour les tables *LCADatabase, AL_Material* et *AL_ObjectType*. On remarque aussi qu'il y a une liaison entre les tables *tbl_processes*, *AL_Material* et *AL_ObjectType* où il est question de la colonne UUID et la colonne processUUID des deux autres tables. Veuillez noter que la figure 1 contient seulement une partie de la structure des tables.

4.2 Création de Socket entre Revit et UBUBI

Premièrement, Revit est un logiciel technologique pour la modélisation des données de bâtiment. (BIM). Il est un outil pour faire de la conception architecturale. En quelques mots, cette application est capable de créer et charger des modèles de bâtiments en trois dimensions.

Cette tâche consiste à conceptualiser et à implémenter un protocole de communication avec le logiciel Revit et UBUBI. La technologie utilisée se base surtout sur le fonctionnement des sockets. L'objectif d'avoir cette technologie est d'être capable de facilement sélectionner un objet dans Revit directement à partir du site web de UBUBI. Puisque les deux logiciels sont indépendants, une méthode d'authentification à l'aide d'un jeton est utilisée pour créer le lien entre les deux applications. Voici le déroulement des étapes :

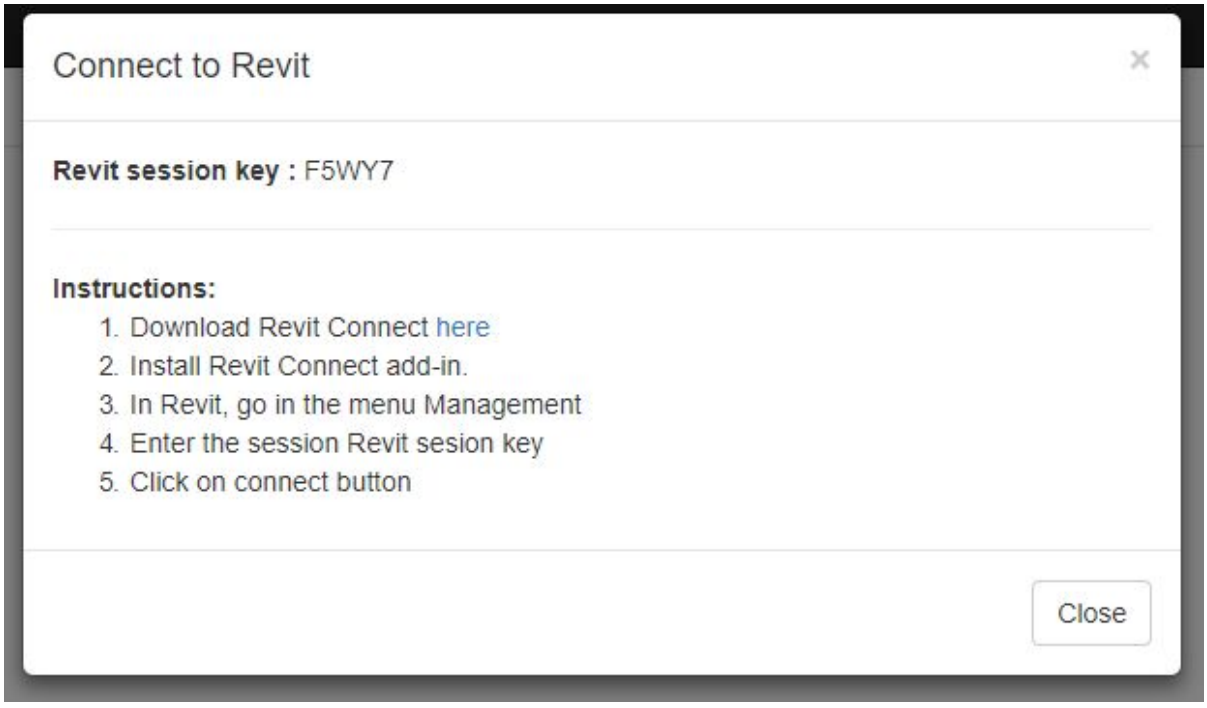

Figure 2: Module de connection Revit

4.3 Liaison automatique des données (matériaux et type objets)

Lors de la création, mise à jour d'un modèle, il est important de lier les objets contenus dans le modèle aux fournisseurs correspondants disponibles pour ce matériel/type d'objet. Cette liaison automatique permet à l'utilisateur de sauver du temps par rapport à une liaison manuelle qui peut devenir exponentielle selon la taille du modèle. Cette liaison se fait lorsqu'un modèle est soit mis à jour ou créé à neuf. Les tables AL_Material ainsi que AL_ObjectType créées dans la section 4.1 contiennent l'information nécessaire pour faire le lien entre les objets/matériaux à leurs fournisseurs respectifs. Si un matériel a un nom contenu dans une de ces tables, une vérification sera faite dans la classe IED2LCA et un échange sera créé automatiquement en les liant.

4.4 Mise à jour du modèle

Étant donné qu'un modèle change à multiples reprises dans un projet, il était avantageux de pouvoir le mettre à jour dans l'application sans avoir à en recréer un à chaque fois. Pour ce faire, une interface de mise à jour a été ajoutée en parallèle à celui du téléversement.

Ainsi, les fonctions de téléversement pouvaient être réutilisées pour convertir les fichiers.ied en données pouvant être storées dans la BD.

La première idée fut de tout reconstruire l'arbre à zéro et de comparer les éléments des fichiers aux éléments du modèle à modifier. Cependant, trouver les éléments manquants, ajoutés et modifiés pour toutes les tâches s'est avéré une tâche trop ardue.

Par contre, il s'est avéré par la suite que seulement certains éléments de l'arbre étaient à risque d'être modifiés. En effet, en gardant la racine de l'arbre, on pouvait y recréer tous les nouveaux processus du niveau *building* au niveau BIM *object type* (voir figure 3)

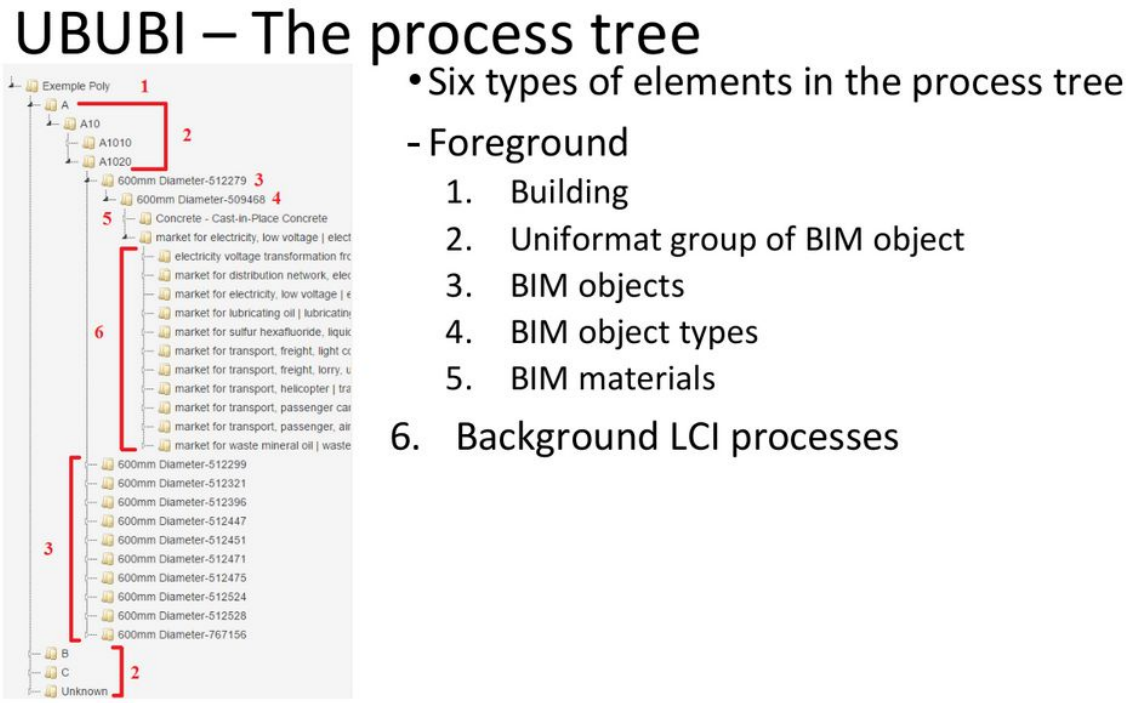

Figure 3: Arbre de processus

Il ne restait qu'à modifier la fonction LinkProcesses déjà créée pour prendre en compte la modification des échanges. Ainsi, si un échange existait déjà, sa quantité était modifiée avec la nouvelle valeur, si un échange n'existait pas, la méthode de création et liaison d'échange était utilisée comme lors d'un téléversement. Finalement, lorsque toutes les étapes de mise à jour étaient terminés il ne restait plus qu'à supprimer les anciens éléments du modèle mis à jour qui ne s'y trouvaient plus suite à la modification du modèle Revit.

4.5 Interface utilisateur

Durant l'analyse de l'application, nous avons remarqué que l'interface utilisateur manquait d'unité et de convivialité dans son ensemble. Il était difficile pour un utilisateur non habitué de comprendre l'application. De plus, l'application n'était pas à sa première itération et elle possédait déjà des contraintes d'utilisabilités. Alors, nous sommes concentrés plutôt sur l'aspect visuel, adaptation du contenu selon la résolution de l'utilisateur et l'interprétation des boutons.

4.5.1 Aspect visuel

Pour l'amélioration de l'aspect visuel de l'application, la première approche était de revoir la barre de navigation de l'application. Dans l'amélioration, nous avons ajouté le logo de UBUBI et enlever le texte des boutons pour les remplacer dans des icônes. Aussi, nous avons déplacé le formulaire de calcul qui était présent pour le mettre dans un modal. Cela a permis d'alléger la barre et ce formulaire sera présent que pour les personnes voulant faire le calcul de l'impact environnemental. En modifiant l'emplacement, cela a donné un clic de plus à l'utilisateur, mais cela à beaucoup aider pour l'adaptation du contenu selon la résolution.

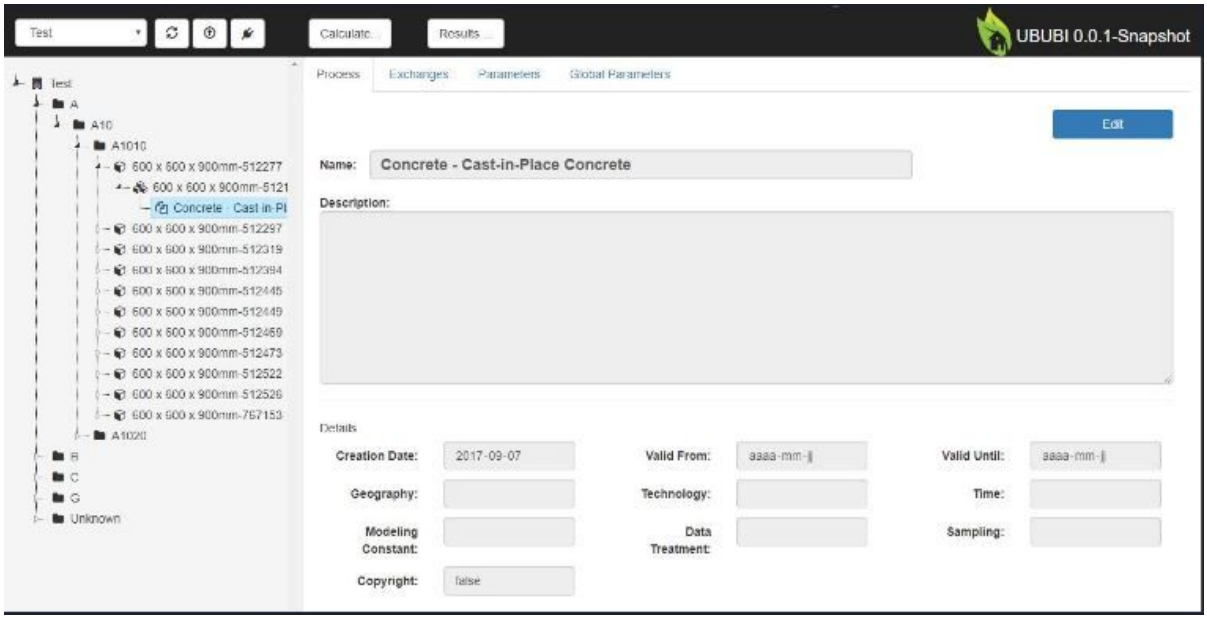

Figure 4: Interface de processus

Par la suite, ce fût au tour au reste du contenu en y ajoutant de la couleur au bouton pour différencier l'action principale de l'action d'annulation et d'avoir aux siens de l'application la même grosseur de bouton, car on pouvait selon la vue différente grosseur d'éléments. Puisque l'application utilise le blanc et le noir comme couleur principal, l'ajout d'un bleu pour les boutons principaux permet à l'utilisateur de mieux les voir.

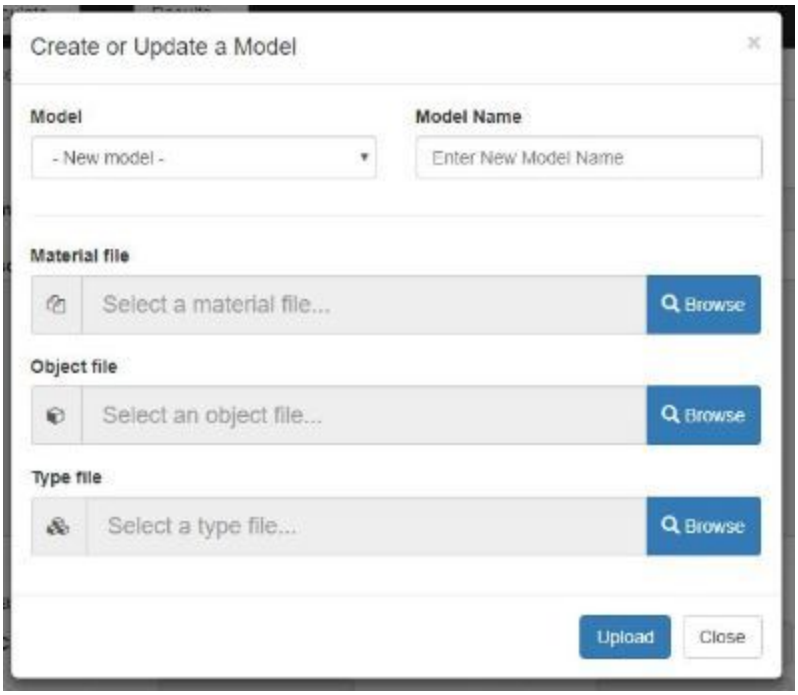

Figure 5: Interface de téléversement

4.5.2 Application Responsive

Pour améliorer l'aspect visuel, une restructuration des classes CSS et une analyse de l'importance de chacune ainsi de l'utilisation des ID aux siens des balises HTML a été nécessaire pour comprendre l'implémentation existante. Cela nous a permis d'avoir un plan et de comprendre comment mettre l'application adaptable selon la résolution sans pour autant reconstruire les vues. La préoccupation vient qu'il existe plusieurs sortes d'écran et de résolution et qu'il est fréquent pour une entreprise comme pour une université de ne pas pouvoir faire un changement global pour des écrans encore fonctionnels. L'ajout de l'adaptabilité du contenu permettra de rejoindre le plus d'utilisateurs. Nous avons utilisé la librairie CSS et JS de Bootstrap qui est une librairie open source et très utilisée dans l'industrie. Bootstrap était déjà utilisé dans l'application, mais elle n'était pas implémentée à son plein potentiel des itérations précédentes. Cette librairie vient avec une base d'un gabarit de style déjà opérationnel et un début du comportement des éléments selon le type d'appareil et la résolution. Par la suite, il restait à mettre en place la modification des classes CSS dans le HTML et ajouter des indicateurs aux besoins pour que l'application devienne adaptable. Pour nous aider dans la validation, nous avons utilisé l'émulateur d'appareil mobile de Google Chrome. Cela nous a permis d'avoir sous la main plusieurs types d'appareils et avec des résolutions différentes. L'émulateur vient par défaut en téléchargeant le navigateur et c'est gratuit.

4.5.3 Icônes

Nous avons par la suite concentré nos efforts sur la représentation des types d'objets de l'arbre des processus. Pour ce faire, nous avons eu besoin de plus d'icônes que la librairie Glyphicon possédait. Alors, nous avons trouvé la librairie fontAwesome qui permet plus de choix d'icône sans être obligée de générer des images. Cette librairie possède plusieurs types différents qui bonifient la librairie Glyphicon, car elle ne possédait pas ceux voulus ou du moins dans ceux qu'on pourrait y donner une signification. Pour l'utiliser, il a fallu l'intégrer à l'application et comprendre le fonctionnement de la génération de l'arbre qui a été conçu auparavant par d'autre développeur. Cette compréhension n'était pas facile, car ils ont utilisé une librairie qu'on peut retrouver plusieurs versions sur Internet qui possède les mêmes fonctionnalités, mais sans posséder la même structure de génération HTML et logique d'utilisation. Une recherche fût nécessaire pour trouver la bonne et beaucoup d'essais et d'erreur, mais le résultat est venu et les icônes ont été ajouté. Dans le futur, il sera plus facile de soit modifier la signification et d'en ajouter.

4.6 Formulaire de l'échange

Dans cette réalisation le but était de faciliter l'utilisateur dans le processus de création d'objet. Auparavant, il avait qu'un seul formulaire et l'utilisateur devaient cocher pour signifier si c'était à l'entrée et le laisser décocher pour la sortie du processus. Il avait certains problèmes pour l'utilisateur pour la modification de ce processus comme par exemple la compréhension du formulaire et les étapes nécessaires à la réalisation. Il avait auparavant deux formulaires en un et c'était qu'une case à cocher qui faisait la différence. Nous les avons séparés en ajoutant un bouton dans la section de la liste des sorties pour plus d'information voir la figure 6.

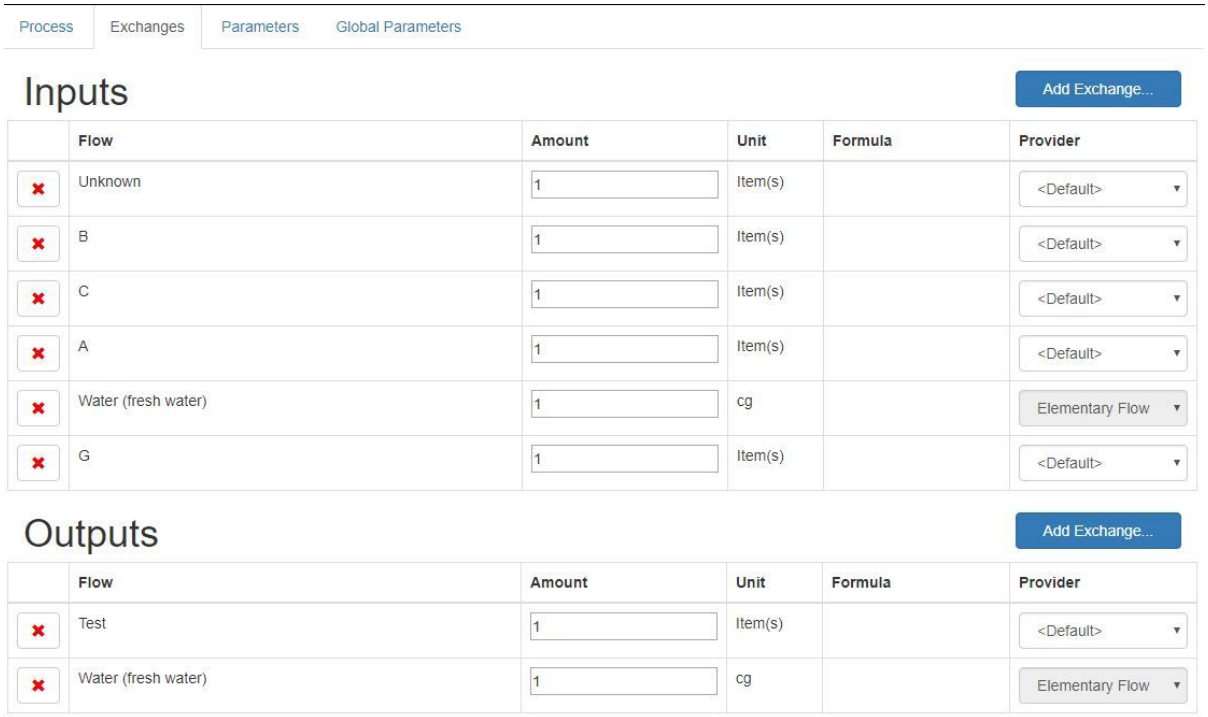

Figure 6: Interface de formulaire d'échange

Cette séparation permettra de mieux cibler l'action de l'utilisateur et facilitera l'ajout de processus dans l'arbre. Il reste encore certains points pour que l'expérience utilisateur soit à son maximum et que l'application puisse répondre à tous les besoins.

4.7 Ajouts et modifications de services webs

Afin de communiquer avec le serveur, l'application utilise des requêtes ajax pour obtenir l'information nécessaire à être affichée. Pour ce faire, des contrôleurs reçoivent le type de demande et ses paramètres et retourne les informations du modèle nécessaires. Les plus gros changements apportés ont été dans le but d'ajouter les groupes d'unités.

4.7.1 Liaison des données

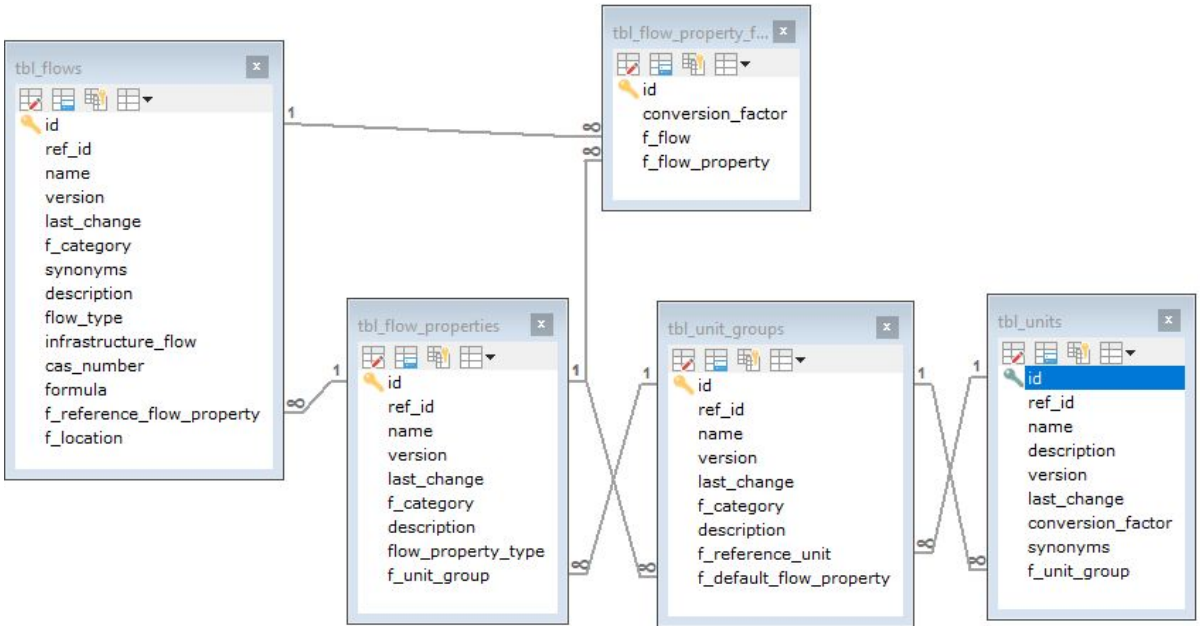

Figure 7: Tables de flux à unité

Tout d'abord, la première étape consistait à trouver les données selon les paramètres demandés. Pour ce faire, l'information était récupérée en utilisant les clés étrangères pour naviguer d'une table à l'autre. La difficulté de cette tâche variait grandement du nombre de tables et de liaisons nécessaires.

Par exemple, pour trouver dans quel groupe se trouvait une unité se trouvait, il suffisait d'utiliser la clé f_unit_group. Dans le sens contraire, nous pouvions récupérer toutes les unités d'un groupe en utilisant la clé de celui-ci.

D'un autre côté, pour obtenir l'unité par défaut d'un flux, il faut passer par la table flow_property_factors qui utilise 2 clés étrangères pour lier les flux à leurs propriétés. Puis, à partir de la table des propriétés, la clé étrangère f_unit_group donne encore la liaison avec sa table d'où la clé f_default_reference_unit permet d'obtenir l'unité par défaut de ce flux.

4.7.2 Modification des contrôleurs

Lorsque les liaisons sont trouvées, les fonctions appropriées peuvent être créées dans le contrôleur. Dans ce cas-ci, les objets de la librairie OLCA ne contenaient pas toutes les méthodes de liaison des classes nécessaires. Nous ne pouvions donc pas utiliser les fonctions d'hibernate pour automatiquement obtenir les objets et les lier entre eux. Le contrôleur a donc eu besoin de requêtes SQL pour faire les liaisons à la main et ainsi récupérer les données. La classe *JSONFormater* s'occupait finalement de prendre ces résultats et de les convertir en objet JSON pour être transmis à l'application.

Chapitre 5 : Problèmes rencontrés

Ce projet ne s'est pas déroulé sans accrochages. Heureusement du côté interpersonnel l'équipe n'avait aucun problème. La majorité des problèmes rencontrés sont dus à la complexité des tâches, manque de clarté des instructions et à l'absence de documentation de l'implémentation déjà existante de l'application.

Dans la création d'une interface, il est très difficile lors d'échange avec le client de bien comprendre ce qui était beau pour lui et de se mettre dans la peau d'un utilisateur sans connaître le domaine d'activité. Pour nous aider, il aurait eu nécessaire d'avoir fait un cours sur treize semaines au préalable sur les concepts et l'utilisation de l'application qu'elle aura à combler et un aperçu.

5.1 Problème avec la création de Socket entre Revit et UBUBI

Il y a plusieurs obstacles qui causent des problèmes avec cette fonctionnalité. Revit est un logiciel qui est faite avec la langue de programmation C#. Puisque UBUBI est programmé en Java, la création de socket entre deux langues de programmation différentes augmente la complexité de la tâche. Malgré cette obstacle, les sockets ont réussi à fonctionner correctement.

Finalement, il y a encore une difficulté importante à régler en lien avec les libraires de Revit. Le problème a surfacé lorsque le socket a été créer entre Revit et UBUBI. L'objectif du socket entre les deux applications est de trouver un moyen de garder les deux sockets toujours ouverts. Ceci permettra les deux logiciels de s'envoyer des messages de façon bidirectionnelle en tout temps. Par contre, pour avoir un socket toujours ouvert, il doit être exécuté sur un autre *thread*. Sinon Revit ne peut pas être utilisé en parallèle avec le socket. Malheureusement, l'utilisation d'un autre *thread* cause des problèmes avec les libraires de Revit. Les fonctions de Revit ne peuvent pas être exécutées dans un autre *thread*. Elle doit être lancée dans le *thread* principal qui cause des problématiques avec cette tâche. Si les fonctions de Revit sont utilisées dans un *thread* séparé, une exception est lancée. La figure au-dessous démontre l'exception

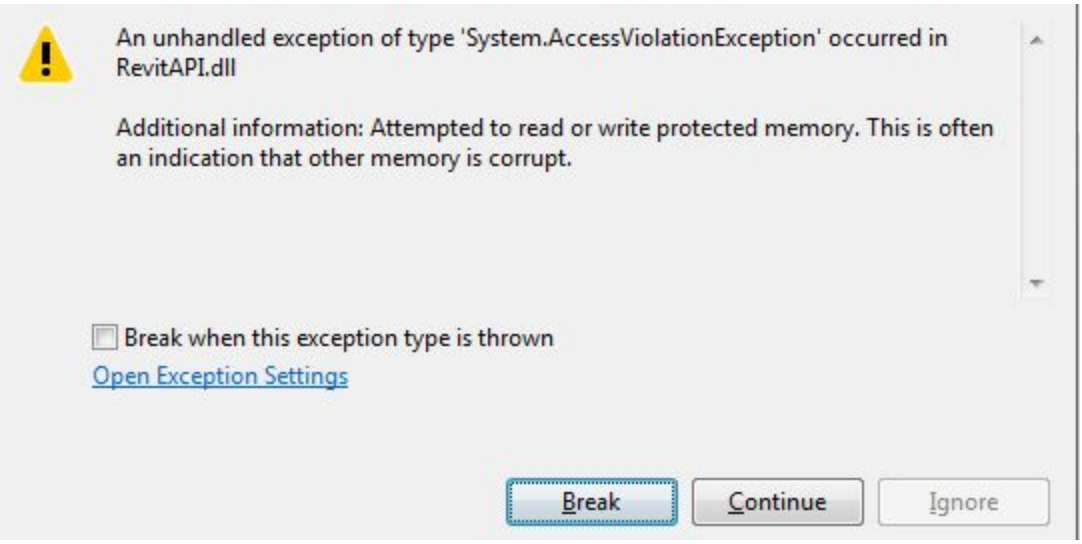

Figure 8: Exception Revit

Chapitre 6 : Risques

6.1 Risques prévus

Tableau 1: Risques prévus

Risque	Impact	Probabilité	Mitigation / atténuation
Retard sur le développement	4	80%	Rattraper le retard la semaine suivante
Manque d'expertise	2	100%	Poser des questions ou demander de l'aide au client ou à ses collègues.
Surcharge de tâches	3	10%	Bien communiquer ses limitations et ses compétences au chef de projet.
Manque de ressources	2	10%	Revoir les objectifs à la baisse.
Mauvaise évaluation ou compréhension des tâches	5	70%	Communiquer avec le chef de projet tout au long du développement d'une tâche pour s'assurer d'être sur la bonne voie.
Mauvaise gestion d'équipe	2	20%	Planifier une deuxième rencontre (possiblement à distance)
Manque de communication	2	20%	Ne pas hésiter à poser des questions avec les membres de l'équipe afin de bien distinguer les tâches. Communication entre les membres de l'équipe sur Slack.
Mauvaise relation interpersonnelle	1	5%	S'il y a un problème, de bien communiquer avec l'équipe pour favoriser la résolution des mauvaises relations.
Mauvais choix de conception/ d'implémentation	2	15%	Examen par les pairs à chaque changement effectué et révision du client avant ajout final.

Fonctionnalités non existantes sur la technologie choisie	3	15%	Revoir si la fonctionnalité peut être développé d'une autre façon et utilisation d'un service externe pour pallier aux fonctionnalités manquantes.
Disponibilité des membres en commun	2	10%	Prioriser la rencontre hebdomadaire dans son emploi du temps et avoir des disponibilités raisonnables.

6.2 Risques encourus

Dans la liste des risques prévus, ceux ayant la plus haute probabilité ont bien eu lieu. En effet, tout d'abord, il est arrivé à plusieurs reprises qu'un membre de l'équipe n'ait pas effectué la tâche prévue dans la semaine. Malheureusement, une bonne partie du travail a dû être complété à la dernière minute.

Ensuite, le manque d'expertise fut un grand obstacle à l'avancement du projet. D'une part, l'architecture du projet initial ainsi que de sa base de données a demandé un certain temps d'apprentissage. De l'autre côté, un manque flagrant de connaissances des éléments de construction a rendu la tâche difficile pour tous les membres de l'équipe au cours du projet. Par ailleurs, le fait que les membres de l'équipe n'ont pas toute la manière expérience avec les technologies utilisées de l'application n'a pas aidé dans la réalisation des tâches, mais cela nous a permis d'en apprendre davantage et de communiquer les trouvailles entre nous.

Finalement, le manque d'expertise a également contribué à une mauvaise évaluation des tâches. Notre stratégie de prendre des tâches étant évalué à facile ou peu de temps en début de parcours et nous permettre une introduction à l'application n'a pas porté le succès encouru. Cela nous a permis de voir qu'une tâche doit être évaluée selon l'implémentation et non seulement basée sur l'hypothèse de la réalisation. Cela nous a aussi permis de voir qu'il existe plusieurs façons de concevoir et de répondre à un problème.

Chapitre 7 : Post-mortem du projet

7.1 Objectifs non-atteint

Dû au manque de temps ou d'expérience des membres de l'équipe, certains objectifs n'ont pas été réalisés. Ces tâches sont quelque peu discutées dans les autres sections, mais en voici une courte liste.

La modification du code uniformat n'a pas été complétée. La taille de cette tâche semblait assez raisonnable à première vue, mais une fois commencée elle se trouvai beaucoup plus complexe et ardue que la description et les explications en donnaient l'impression. Plusieurs sections importantes de cette tâche ont été réalisées. Du côté "front-end" les modifications visuelles telles que l'ajout d'une liste de codes uniformat liée au code de l'objet ainsi que la modification de l'arbre ont été complétées. Pour ce qui est du "back-end" beaucoup plus a été fait: l'ajout des codes uniformat dans la base de données, la création d'une classe contrôleur, la liaison à la liste, ainsi que la logique de suppression de lien as été complétée. Malheureusement, l'étendue de cette tâche jumelée avec le fait qu'il s'agissait d'une première tâche a rendu sa réalisation complète d'autant plus difficile. Pour cette raison, le travail effectué a été déplacé dans une branche, sur git, séparé pour le futur.

7.2 Recommandations

Dans l'objectif d'avancement du projet, certaines recommandations sont suggérées. Premièrement, en ayant la base du module de connections à revit, celui-ci peut grandement être amélioré. Par exemple, ce module peut être utilisé pour directement envoyer des fichiers à l'application afin de créer un modèle ou d'en mettre un à jour.

Puis, lors des prochaines itérations du projet, nous croyons que l'emphase peut être mise sur le calcul et l'affichage des résultats. Une fois complétée, l'application pourrait être proche d'une phase beta qui pourrait être rendue disponible.

Puisque le projet est open source, le développement continu et la collaboration de tous sont des éléments essentiels au bon avancement du projet. Ainsi, les idées et suggestions de plusieurs collaborateurs seront la locomotive du projet.

CONCLUSION

Pour terminer, bien que quelques tâches aient été accomplies, la taille et la complexité du projet ont été une épreuve importante pour tous les membres de l'équipe. Une période de temps plus longue aurait grandement aidé à la compréhension de l'application ce qui aurait, à son tour, aidé. L'équipe a respecté les dates limites du projet. Plusieurs tâches ont été complétées à temps, la connections a revit, la modification de l'interface usager, la génération automatique des liens et la mise à jour du modèle. Malheureusement, plusieurs tâches n'ont pas pu être terminés tels que: la modification des codes uniformat et l'intégration de la nouvelle version du module de calcul Spark. Un groupe ayant plus de temps disponible ou ayant des ressources connaissant les technologies utilisées aurais probablement accomplir la majorité des tâches, mais dans le cadre d'un projet de fin d'études le projet a pris une envergure difficile à surmonter.

Bibliographie et références

OpenLCA, openLCA Schema
Internet, consulté le 12 Octobre 2017
http://greendelta.github.io/olca-schema/

JQuery, jQuery API
Internet, consulté le 22 septembre 2017.
https://jquery.com/

Wikipedia, Building Information Modeling
Internet, consulté le 12 septembre 2017.
https://en.wikipedia.org/wiki/Building_information_modeling

Youtube, Revit Demo
Internet, consulté le 2 décembre 2017.
https://www.youtube.com/watch?v=3ucMoBi7zFE

Revit, My First Plug-in Training
Internet, consulté le 28 novembre 2017.
http://usa.autodesk.com/adsk/servlet/index?siteID=123112&id=16777469

GitLab, The leading product for integrated software development.
Internet, consulté le 12 septembre 2017.
gitlab-ce.logti.etsmtl.ca

Google Drive, Document UBUBI
Internet, consulté le 8 septembre 2017.
https://drive.google.com/

StackOverflow, Where Developers Learn, Share, & Build Careers
Internet, consulté le 18 octobre 2017.
https://stackoverflow.com/

ANNEXE I

PLAN DE TRAVAIL

#	Commence	Termine	Effort (h)*	Tâches/Jalon	Livrables/Artéfacts	Responsable(s)
1	07-09-2017	07-09-2017	1	Première rencontre -superviseur et professeur	Planification	NB,ELB,GF, AK
1.1	récurrent	récurrent	10	Rencontre hebdomadaire -superviseur et professeur	Planification	NB,ELB,GF, AK
2	07-09-2017	07-09-2017	1	Remise de la documentation	Dossier “Google drive”	NB,ELB,GF, AK
2.1	07-09-2017	14-09-2017	10	Analyse et définition du projet	Analyse	NB,ELB,GF, AK
3	07-09-2017	28-09-2017	10	Faire le service backend pour la table BIMProcessus, se référer à Processus (dans le dossier controler)	Base de données	GF
4	28-09-2017	14-10-2017	10	Dans la fonction d'ajout d'échange, afficher seulement les bonnes unité pour le type de flux choisis.	Interface utilisateur	GF
5	07-09-2017	28-09-2017	12	Créer la structure de donnée pour les liaison automatique	Module d’import	ELB
6	26-10-2017	30-11-2017	20	Lier les données automatiquement (matériaux seulement)	Upload	ELB, NB
7	19-10-2017	----------	8	Refaire l'ajout d'échange pour faciliter		AK
8		09-11-2017	5	Permettre la modification de la quantité des échange	module d’import	ELB
9	07-09-2017	----------	30	Correction CSS	Interface utilisateur	AK
10	14-09-2017	----------	40	Permettre de changer le code uniformat d’un objet	module de résultats, Interface utilisateur	NB
11	2-11-2017	----------	5	Vérifier et mettre à jour la connexion au service de calcul	module de calcul	GF
12	2-11-2017	----------	12	Mise à jour du modèle	module d’import	GF
13	2-11-2017	9-11-2017	5	Ajouter la sélection d'un modèle pour l'update dans l'upload IED	Interface Utilisateur	ELB

14	30-11-2017	----------	8	Lier les données automatiquement (type objet seulement)	Upload	NB
15	07-12-2017	19-12-2017	20	Revit/UBUBI Sockets	Communications	ELB
18	23-11-2017	27-11-2017	8	Faire une fenêtre pour le lancement des calculs	Interface Utilisateur	AK
26	------	----------	60	Corrections des bugs en continue		AK, ELB, NB, GF
27			12	Rapport de proposition	Proposition de projet	
28			8	Rencontre – professeur superviseur		
29			24	Rapport d'étape	Rapport d'étape	
30			8	Rencontre – professeur superviseur		AK, ELB, NB, GF
31			3	Présentation	Présentation	AK, ELB, NB, GF
32			30	Remise du travail	Rapport	AK, ELB, NB, GF