Création d'un plug-in Revit de gestion et mise à jour d'éléments 3D à destination de néophytes de la géomatique

Jordi Attencia

▶ **To cite this version:**

Jordi Attencia. Création d'un plug-in Revit de gestion et mise à jour d'éléments 3D à destination de néophytes de la géomatique. Sciences de l'ingénieur [physics]. 2015. dumas-01334139

HAL Id: dumas-01334139

https://dumas.ccsd.cnrs.fr/dumas-01334139

Submitted on 20 Jun 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0 International License

CONSERVATOIRE NATIONAL DES ARTS ET METIERS

ÉCOLE SUPÉRIEURE DES GÉOMÈTRES ET TOPOGRAPHES

MÉMOIRE

présenté en vue d'obtenir

le DIPLÔME D'INGÉNIEUR CNAM

Spécialité : Géomètre et Topographe

par

Jordi ATTENCIA

Création d'un plug-in Revit de gestion et mise à jour d'éléments 3D à destination de néophytes de la géomatique

Soutenu le 7 Juillet 2015

JURY

PRESIDENT : Monsieur Jean-Michel FOLLIN

MEMBRES : Monsieur Michel BARNOUD
Monsieur Vincent HABCHI, professeur référent
Monsieur Olivier POILPRÉ
Monsieur Gwénaël SAGNE, maître de stage
Madame Élisabeth SIMONETTO
Madame Nathalie THOMMERET

Remerciements

Je tiens tout d'abord à remercier Gwenaël Sagne, géomètre-expert, pour son encadrement tout au long de mon TFE. Ses conseils avisés et son souci du perfectionnement ont pu faire avancer ce projet dans la bonne direction.

Merci à l'ensemble du personnel du cabinet Géomat de Fougères pour leur excellent accueil et leur soutien tout au long de ce projet.

Merci à Jennifer Fardin, géomètre-expert à Géomat, pour les conseils qu'elle a su me prodiguer quant à la rédaction de ce mémoire.

Merci à Mustapha Sadoune de la société Leica Geosystems pour sa disponibilité et sa réactivité.

Un grand merci à Pauline Boucher, stagiaire de fin d'étude à l'ESGT pour nos échanges et ses conseils au cours de ce projet.

Merci à Vincent Habchi, mon professeur référent, pour ses conseils avisés tout au long de ce TFE.

Un grand merci à toute la communauté de Stack Overflow pour leur aide sur l'utilisation du langage C# totalement nouveau pour moi, et pour la réparation de nombreux bogues.

Un grand merci également à toute la communauté des forums Autodesk, particulièrement à la section Revit API, pour leurs conseils et leurs orientations sur les différentes fonctions mises à ma disposition. Un merci particulier à Aaron Lu et Arnošt Löbel pour leur réactivité et leur travail pour l'ensemble de la communauté.

Un immense merci à Jeremy Tammik, membre de l'Autodesk Developer Network, pour son travail colossal pour la communauté des développeurs Revit. Merci pour toutes les discussions que nous avons pu avoir et qui ont permis de faire de ce projet ce qu'il est aujourd'hui.

Merci à Christophe Proudhom pour son accessibilité et son écoute tout au long de ces cinq années.

Merci à toutes les personnes qui, parfois sans le savoir, ont eu un impact positif sur ce TFE.

Et pour terminer, merci à tous les membres de ma famille pour leurs soutiens et encouragements tout au long de mes études, particulièrement à mon père qui a su me donner le goût pour le métier de géomètre et m'a poussé à continuer même dans les moments les plus difficiles.

Liste des abréviations

API : Application Programming Interface

POO : Programmation Orientée Objet

BIM : Building Information Modeling

PPP : Projet Pré-Professionnel

ESGT : École Supérieure des Géomètres et Topographes

Table des matières

Introduction

L'industrie des télécommunications est un milieu très compétitif. Ses coûts techniques sont principalement liés à l'installation et l'entretien des réseaux. Il est donc primordial pour être compétitif de procéder à une maintenance efficace et à moindre coût.

La société Axione, filiale du groupe Bouygues Energies & Services, est un acteur important de l'industrie des infrastructures de télécommunications. Elle effectue de nombreuses poses d'antennes ainsi que les ajouts de modules à ces dernières et leurs maintenances. La problématique d'Axione s'articule autour de deux points :

-L'incapacité de disposer de plans à jour : lors de l'ajout d'un équipement, les équipes d'Axione doivent se déplacer préalablement pour vérifier que les dimensions du lieu de l'antenne permettent l'ajout du dit équipement.

-La tendance actuelle des opérateurs est de déléguer la gestion du parc. Il y a donc une nécessité à plus ou moins long terme d'en avoir la connaissance exhaustive et d'avoir des logiciels de gestion appropriés pour les projets d'installations nouvelles et leurs récolements.

L'acquisition de données 3D est une technologie maitrisée par quelques cabinets de Géomètres-Experts. Le protocole terrain est sensiblement le même que pour une station totale classique mais le traitement de données est bien plus lourd. Nous nous dirigeons lentement vers des modèles 3D et des marchés ou le BIM[1] est central. Malgré les nombreuses applications de ce type de technologie, les compétences qui lui sont associées ne sont pas encore accessibles à des néophytes de la géomatique. En effet, manipuler un tel volume de données de grandes précisions n'est pas chose aisée et le coût associé à ces manipulations n'est pas négligeable. La production de modèles 3D des sites et des équipements qu'Axione entretient permettrait de répondre à leur problématique, modèles 3D qu'Axione n'est actuellement pas en mesure de produire, faute de matériel et de compétences de leur personnel. En effet, des modèles 3D sont à privilégier par rapport aux plans 2D car le BIM est plus adapté à une gestion patrimoniale des équipements.

Le traitement de leur problématique nécessite donc des outils simples d'acquisition, de récolement et de traitement. C'est dans cette optique que GEOMAT a proposé à Axione de créer une base de données 3D de leurs antennes, équipements et lieux de poses. Ceci serait couplé à un système de photogrammétrie terrestre qui permet le lever 3D de manière simple : le système RAPH[2] développé par la société FIT ESIC. Ce système a été développé pour une utilisation simple par des personnes ne disposant pas de la compétence photogrammétrique. Un système de manipulation et mise à jour d'objets 3D sera nécessaire au bon fonctionnement du processus.

[1] Building Information Modeling
[2] Récolement Assisté par PHotogrammétrie

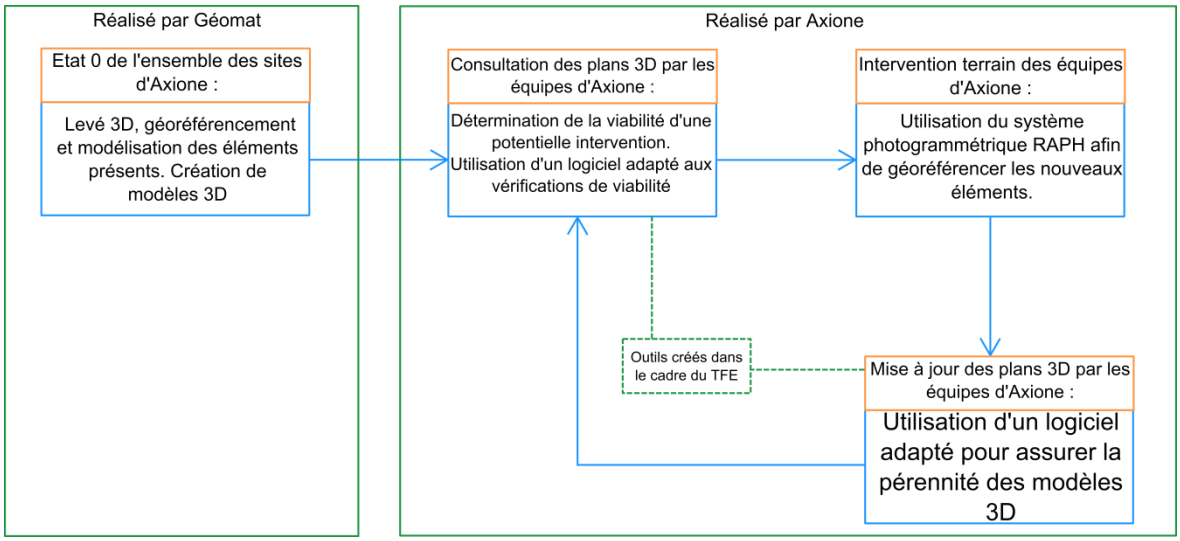


Figure 1 : Schéma du processus envisagé

L'utilisation du logiciel Revit, leader dans le domaine du BIM, semble la plus adaptée. Ce logiciel, étant relativement compliqué pour des personnes non familières avec l'environnement AutoDesk et des logiciels 3D, couplé au fait que l'éventail de fonctionnalités proposées est bien plus large que l'utilisation souhaitée, une simplification de son utilisation sera nécessaire.

C'est au sein de cette problématique qu'est apparue l'idée du TFE. J'ai intégré le cabinet de Géomètres-Experts GEOMAT de Fougères situé en Ille-et-Vilaine, siège social de la société comptant 15 agences réparties en Bretagne et Basse Normandie.

Le présent mémoire a pour objectif de détailler la mise en place d'un processus permettant de manipuler et mettre à jour des objets 3D, et d'adapter ce processus pour des utilisateurs possédant le minimum de compétences. Il sera traité ici uniquement de la partie **Consultation des plans 3D** et **Mise à jour des plans 3D**.

Dans un premier temps, une description plus poussée du contexte et du cahier des charges du projet sera présentée, puis, lors d'une seconde partie, nous aborderons l'utilisation logiciel Revit et quelques-unes de ses spécificités, la 3ème partie concernera la solution retenue pour simplifier ce processus et son développement. Enfin, la 4ème dernière partie sera une démonstration du plug-in créé.

I Contexte et cahier des charges

Le fonctionnement actuel des équipes d'Axione est le suivant :

-Réception d'une commande : un nouvel équipement doit être installé sur un site.

-N'ayant pas la bonne connaissance de tous leurs sites, une équipe se déplace sur site pour constater l'état des lieux et les équipements déjà installés. Des photos sont prises et des cotes sont relevées au ruban et à la chaîne.

-De retour au bureau, une vérification de la viabilité de l'installation est effectuée à l'aide de photos et cotes. L'équipement est ensuite commandé.

-Un retour sur site est alors nécessaire pour effectivement installer l'équipement.

La méthode est donc peu précise et nécessite des déplacements parfois longs. Nous pouvons estimer la précision d'un tel placement à 5cm lors de la phase de reconnaissance. Il faut donc mettre en place un processus permettant de gagner du temps en évitant cette phase de reconnaissance sur le terrain en la transformant en phase de reconnaissance au bureau via un logiciel, en l'occurrence Revit.

Ce processus doit apporter au moins la même précision que celle du processus actuellement utilisé par Axione. Il doit également être suffisamment simple pour être utilisé par des personnes totalement étrangères à des logiciels tels qu'Autocad ou Revit. De plus, fournir des informations permettant aux utilisateurs de contrôler la qualité de leurs manipulations est primordial. Il faut donc veiller à ce que ces informations soient suffisamment simples pour être interprétées par des utilisateurs n'ayant probablement pas de compétences en géomatique. Parler d'écart moyen quadratique ou de valeur du facteur unitaire de variance lors de la descente d'un test du Khi-deux portera moins de sens que de parler d'un écart absolu en distance aux yeux de l'utilisateur lambda.

Pour pouvoir placer précisément les objets 3D dans un modèle au sein de Revit, il est nécessaire de posséder un système de coordonnées permettant de situer les éléments les uns par rapport aux autres. Ces opérations pourront être réalisées par GEOMAT. Il est néanmoins important qu'une fois « l'état 0 » réalisé, c'est-à-dire l'ensemble des sites modélisé et muni d'un système de coordonnées, le total contrôle du processus demeure dans les mains d'Axione qui souhaite rester quasi autonome dans l'exercice de son travail. Axione doit donc être en mesure de rattacher les équipements qu'elle installe. C'est l'objectif du système RAPH.

RAPH permet à partir de cibles disposées sur le site, et de photos prises des équipements à rattacher et des cibles, de rattacher l'ensemble des éléments du site par un système de photogrammétrie terrestre. La précision du géoréférencement de ces photos est de l'ordre de 4mm. S'ajoute à cette erreur, l'erreur de pointé. En effet, l'utilisateur va venir pointer des points sur ces photos assemblées pour extraire les coordonnées de l'équipement qui l'intéresse, erreur estimée à 1cm. Il va ainsi obtenir les coordonnées nécessaires au récolement et permettre la pérennité du processus. L'imprécision globale obtenue est donc de l'ordre de 2cm, ce qui est acceptable au vu des méthodes actuelles d'Axione. Il faut donc que le traitement sous Revit ne dégrade pas ou très peu cette précision et rester ainsi dans les tolérances pour fournir à Axione un outil lui permettant de gagner du temps sans dégrader sa précision. Il faut garder à l'esprit, lors de la création de cet outil que, à l'image de RAPH, il doit rester très simple d'utilisation et suffisamment rapide pour effectivement gagner du temps.

II Utilisation de Revit

Avant de procéder aux opérations nécessaires à Axione, il convient de s'attarder sur le fonctionnement de Revit et sur son utilisation.

II.1 Modélisation

La première façon de modéliser des objets 3D dans Revit est de dimensionner ces objets manuellement. Les objets 3D dans Revit sont appelés Familles (extension .rfa) tandis que les fichiers dans lesquels il est possible de placer des nuages de points, plans, MNT, familles… sont appelés projets (extension .rvt). Ainsi pour créer un objet 3D, il est nécessaire d'ouvrir un modèle de fichier famille.

Il existe de nombreux modèles différents plus ou moins adaptés à différents objets tels que des portes, fenêtres, escaliers, balustrades, colonnes… ou simplement du mobilier. Chacun de ces modèles existe en deux versions différentes : l'une en système métrique et l'autre en système impérial. Pour modéliser les équipements j'ai initialement choisi le modèle « metric furniture » qui correspond à un modèle destiné au mobilier en système métrique.

L'interface de base repose sur celle des logiciels AutoDesk :

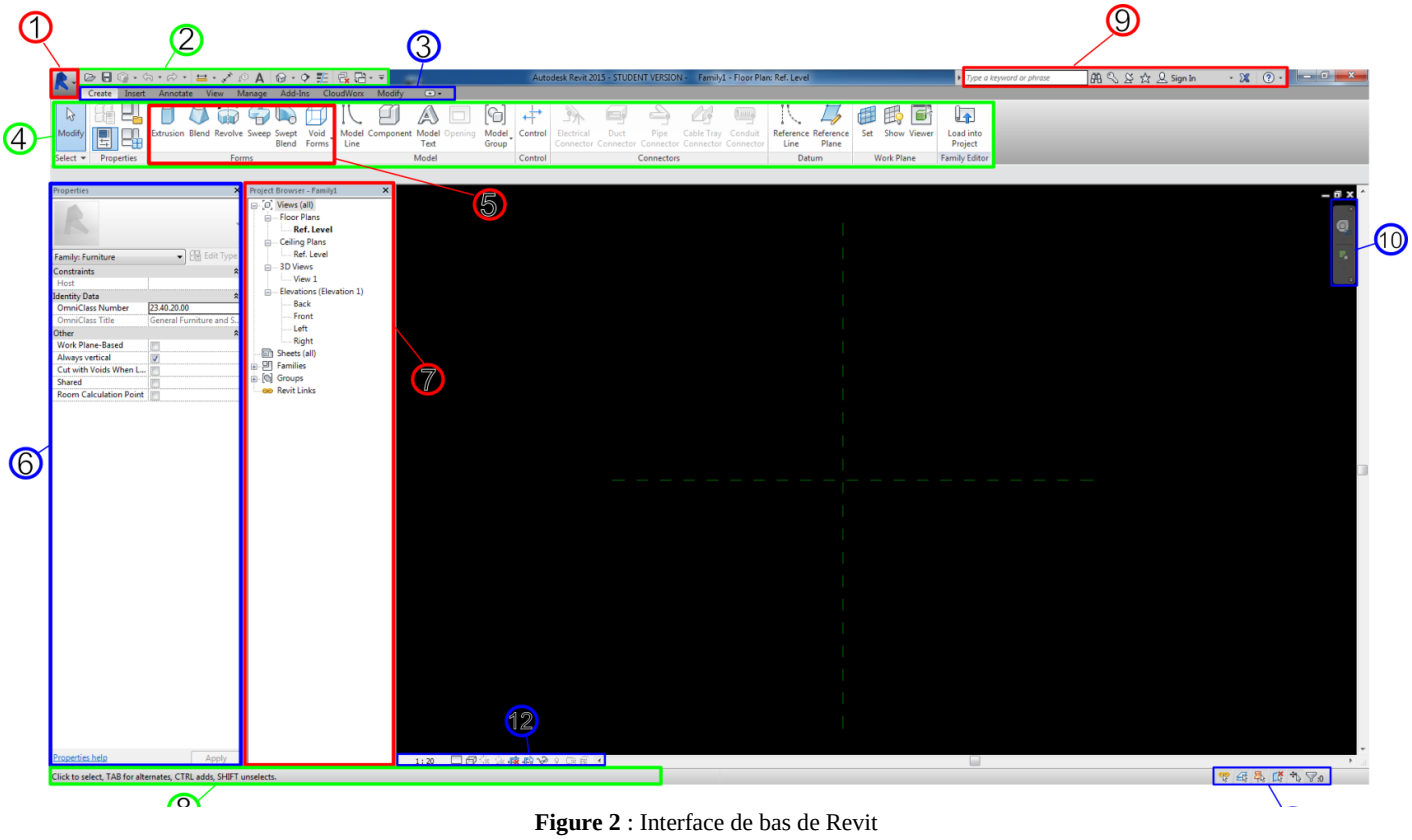


Figure 2 : Interface de bas de Revit

1 : Menu principal
2 : Barre d'accès rapide
3 : 1er ruban
4 : 2ème ruban
5 : 3ème ruban
6 : Panneau des propriétés

7 : Panneau des vues
8 : Barre d'indication contextuelle
9 : Barre d'aide et licences
10 : Raccourcis de navigation
11 : Raccourcis de sélection
12 : Raccourcis de visualisation

Pour modéliser un objet on utilise le ruban « Forme » issu du ruban Créer. Il existe six outils dans ce ruban que l'on peut combiner à volonté :

- Extrusion : L'on crée une forme de son choix dans un plan, puis une fois sorti de ce plan on peut simplement lui donner une profondeur. Exemple : un cylindre.
- Blend : Identique à Extrusion à la différence que les 2 formes aux extrémités ne sont pas nécessairement les mêmes, que ce soit en terme de dimensions ou de nature. Ainsi la forme du volume évolue d'une extrémité à l'autre. Exemple : une pyramide.
- Revolve : L'on crée un profil et choisit un axe autour duquel le profil va tourner pour ainsi créer un solide. On utilise donc revolve pour tout solide comportant un axe de symétrie centrale. Exemple : un vase plein.
- Sweep : On est à nouveau sur le même principe qu'Extrusion sauf qu'au lieu de donner une épaisseur l'on va simplement fournir un « chemin » sous la forme d'une ligne que la forme va suivre. Exemple : un tuyau comportant des coudes.
- Swept Blend : C'est simplement la combinaision de Sweep et Blend. Exemple : un tuyau changeant de diamètre et comportant des coudes.
- Void Forms : En cliquant sur Void Forms, l'on peut choisir entre les 5 outils précédents. Mais au lieu de créer un solide, l'on crée du vide. Si l'on crée un vase plein à l'aide de Revolve, on peut créer le même vase mais de dimensions plus petites avec Void Forms, et en combinant ces deux objets on parvient à créer un vase vide.

Revit nous fournit des outils très performants pour créer des objets. Ici la tâche est de représenter l'existant. Fort heureusement, les équipements de l'industrie de la télécommunication sont normalisés. Ainsi, les mêmes objets sont présents sur de nombreux sites. Néanmoins, la modélisation de ces objets à partir de cotes peut s'avérer très chronophage et fastidieuse. La modélisation à l'aide d'un nuage de points serait beaucoup plus performante temporellement, et à l'avantage de garantir une meilleure fidélité à la réalité. Cependant, pour des soucis de licences et de formation pour de telles opérations, tous les objets utilisés dans ce TFE ont été modélisé manuellement.

II.2 Système de coordonnées

Revit possède un système de coordonnées particulier, très différent de ce qui peut être retrouvé dans d'autres logiciels Autodesk tels que Autocad. Ce système est déterminé par deux points et la direction du Nord. Ces points et cette direction peuvent être placés de façon totalement arbitraire. Il est également possible d'affecter des coordonnées à ces points. Le premier point est le Survey Point. Il définit de façon relative les coordonnées Est et Nord de tous les éléments du projet. Le second point est le Project Base Point qui définit de façon relative toutes les altitudes du projet. Ces deux points peuvent évidemment être confondus. Ce système est peu adapté à un géomaticien mais plutôt à un architecte puisque ce dernier peut se contenter de travailler en coordonnées locales. Il est pourtant primordial de maîtriser ce système afin d'assurer un placement précis des éléments 3D.

II.3 Déplacer des objets

Il est possible de déplacer des familles de la même manière que dans Autocad. La commande **déplacer** fonctionne de manière totalement identique. Du fait que Revit soit un logiciel 3D il est plus compliqué de placer les familles aux positions voulues. Il faut utiliser les différentes vues, Nord, Sud, Est, Ouest, de dessus ou n'importe quelle vue créée, pour placer correctement la famille.

L'outil rotation fonctionne également de manière similaire. Il faut se placer dans une vue qui correspond au plan dans lequel la rotation va s'effectuer, lancer l'outil rotation, sélectionner un point de base puis une ligne de base et choisir l'angle désiré.

Ces deux outils seront suffisants pour permettre un placement précis, à moins de 5cm, des familles ce qui est suffisant pour déterminer si le site dispose de la place nécessaire à l'installation du nouvel équipement.

III Développement et utilisation du plug-in PlaceAndCheck

III.1 Choix de la méthode

Indépendamment de la méthode de modélisation choisie, il convient ici de définir précisément la manière selon laquelle l'utilisateur va procéder pour effectuer ses opérations, le but étant que cette manière soit la plus simple et la plus accessible possible. Revit dispose d'une API ou Application Programming Interface. Une API est un ensemble de types et de fonctions qui permet la communication entre un logiciel, en l'occurrence Revit, et un langage de programmation. Grâce à cette API, il est possible de créer un plug-in permettant le chargement de familles dans un projet, son placement dans celui-ci, ainsi que la vérification de certaines contraintes. Cela semble être une solution adaptée.

Plusieurs possibilités ont donc été envisagées :

- Intégration d'un nouveau raccourci dans Revit permettant le placement automatique d'une famille dans un projet et vérifiant ensuite les éventuelles collisions entre familles. Tout cela sans que l'utilisateur n'ait autre chose à faire que d'ouvrir un projet et sélectionner une famille.
- Intégration d'un nouveau raccourci dans Revit permettant le placement d'une famille dans un projet et vérifiant ensuite les éventuelles collisions. Ce raccourci guidera l'utilisateur et l'accompagnera à travers les différentes étapes du processus.

La première possibilité suppose la connaissance parfaite de chaque site et la définition de zones d'insertion possible pour chaque famille possible. Cela implique une connaissance exhaustive des équipements utilisés actuellement par Bouygues Telecom ainsi qu'une grande précision du point de vue de la modélisation. La programmation d'un tel raccourci est également très compliquée puisque l'on remplace « l'intelligence humaine », qui détermine où peut aller l'équipement, par un algorithme. L'avantage d'une telle méthode est qu'elle ne demande aucune compétence particulière à l'utilisateur. Cette méthode est donc considérée comme idéale du point de vue de la réponse à la problématique d'Axione.

La deuxième possibilité est, quant à elle, beaucoup plus simple à créer puisque l'utilisateur détermine lui-même ou placer l'équipement. En revanche, elle sollicite d'avantage les compétences de l'utilisateur que la première.

Dans le cadre du TFE, qui comporte un temps imparti, et compte tenu de mes compétences dans le domaine de la programmation, n'étant pas développeur de formation, la deuxième méthode a été privilégiée.

III.2 Langage et environnement

L'API de Revit est compatible avec 5 langages : C++, C#, VB, Ruby et Python. Parmi ces langages, le seul que j'avais un peu pratiqué est le langage Python. En revanche je ne l'avais que très rarement utilisé dans un cadre de POO (Programmation Orientée Objet). Je n'avais également jamais utilisé d'API auparavant. Lorsque j'ai effectué mes recherches pour apprendre à m'en servir, j'ai découvert un tutorial pour créer un plug-in sous Revit. Ce tutorial utilisait le langage C# car c'est le langage le plus répandu au sein de la communauté Revit. J'ai donc décidé de privilégier le C#. Ce langage a aussi comme avantage d'être supposément plus facile à aborder que les quatre autres.

Le tutorial utilisait comme environnement le logiciel Visual Studio Express 2013. N'ayant jamais utilisé le C# auparavant et étant satisfait de cet environnement après avoir suivi le tutorial, j'ai continué de l'utiliser pour mon développement.

III.3 Développement du plug-in

III.3.1 Prérequis

Par la suite nous allons utiliser des opérations et des éléments algorithmiques simples. Ils sont explicités ici pour simplifier la compréhension des différentes explications qui seront exposées.

-**Le type d'une variable** est sa nature. En C# il n'existe pas de variable sans type. Un type peut s'apparenter à une étiquette que l'on colle sur des variables. Le type conditionne beaucoup d'opérations. Il empêche notamment d'affecter une valeur à une variable si le type ne correspond pas. On ne peut pas affecter une chaîne de caractères à un entier.

-**Déclarer des variables** en C# signifie signaler au compilateur le nom et le type d'une variable pour que celui-ci lui alloue un emplacement mémoire et le stocke. Il est également possible de déclarer des fonctions.

-**Les arguments d'une fonction** sont les éléments nécessaires au traitement d'une fonction. Une fonction réalisant l'addition de deux variables prendra ces deux variables en arguments.

-**Un algorithme naïf** est une procédure réalisant les traitements les plus évident possibles, sans se soucier de rapidité ou d'adaptabilité. Á l'inverse, **un algorithme intelligent** effectue des traitements plus réfléchis, prend en compte la rapidité et s'adapte plus facilement à de nouvelles données.

III.3.2 Algorithme envisagé

Il est important d'avoir en tête une direction globale vers laquelle on veut avancer afin de savoir où l'on va, mais aussi pour pouvoir s'assurer un minimum que chaque étape est réalisable et que le processus est viable et a des chances de voir le jour.

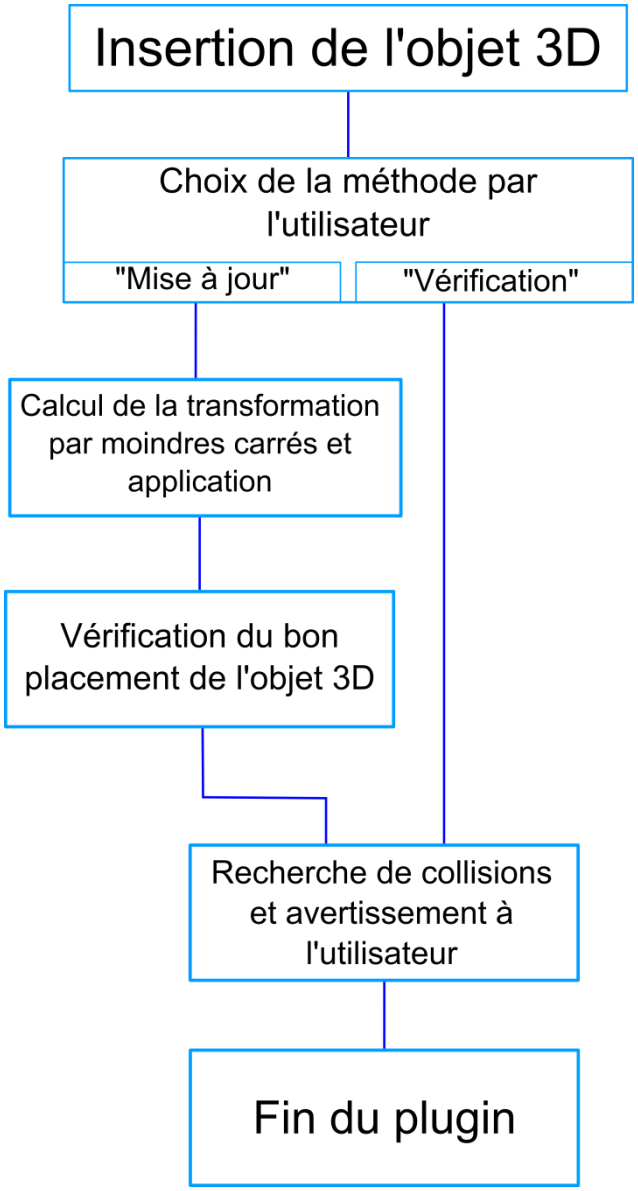


Figure 3 : Schéma de l'algorithme envisagé

Il faudra probablement compter sur des actions simples de la part de l'utilisateur telles que faire un choix entre 2 options ou encore sélectionner des éléments à l'écran.

III.3.3 Insertion de la famille

Il a été envisagé deux manières d'insérer une famille dans un projet :

- À partir d'un point ce qui permettrait d'insérer des familles où on le souhaite.

- À partir d'une face, où l'on forcerait la famille à se plaquer contre une autre par exemple dans le cadre de câblage.

En parcourant le manuel de l'API nous pouvons trouver plusieurs méthodes différentes pour insérer une famille. Selon la façon dont nous souhaitons insérer la famille, la méthode diffère. Dans tous les cas nous utilisons la fonction **NewFamilyInstance** qui prend des arguments différents selon la méthode. Il va donc falloir fournir les bons arguments à cette fonction pour permettre l'insertion de la famille dans le projet. Nous allons commencer par récupérer la famille que l'on souhaite insérer.

De façon assez étonnante, il n'existe pas de fonction dans l'API qui permet de faire l'équivalent de « Fichier » puis « Ouvrir » dans un logiciel. Il faudra donc créer cette fonction nous-même.

```
static string FileSelect(
  string folder,
  string title,
  string filter,
  ref string filename)
{
    OpenFileDialog dlg = new OpenFileDialog();
    dlg.Title = title;
    dlg.CheckFileExists = true;
    dlg.CheckPathExists = true;
    dlg.InitialDirectory = folder;
    dlg.FileName = filename;
    dlg.Filter = filter;
    bool rc = (DialogResult.OK == dlg.ShowDialog());
    filename = dlg.FileName;
    return filename;
}
```

Figure 4 : Fonction FileSelect

La fonction **FileSelect** prend en argument un dossier qui sera celui présenté par défaut à l'utilisateur dans la fenêtre ouverte, un titre qui sera le titre de la fenêtre, un filtre qui forcera la sélection d'un type de fichier particulier ainsi qu'une chaîne de caractère qui sera affichée par défaut dans la sélection. Ce sera cette chaîne de caractères qui sera renvoyée comme résultat par la fonction. Nous verrons plus tard que nous allons effectuer cette opération d'ouverture pour différents types de fichiers. Il parait alors judicieux de créer différentes fonctions pour différents types de fichiers. Ces fonctions appelleront **FileSelect**. Puisque les fichiers des familles sont des fichiers d'extension .rfa, nous allons créer une fonction **FileSelectRfa**.

```
static public string FileSelectRfa(
  string folder,
  ref string filename)
{
    return FileSelect(folder,
      "Selectionnez un fichier Revit family ou annulez pour quitter",
      "Fichier Revit Family RFA  (*.rfa)|*.rfa",
      ref filename);
}
```

Figure 5 : Fonction FileSelectRfa

Grâce à cette fonction on va récupérer une chaîne de caractères contenant le chemin complet de la famille c'est-à-dire quelque chose s'apparentant à « C:\Dossier1\Dossier2\...\MaFamille.rfa ».

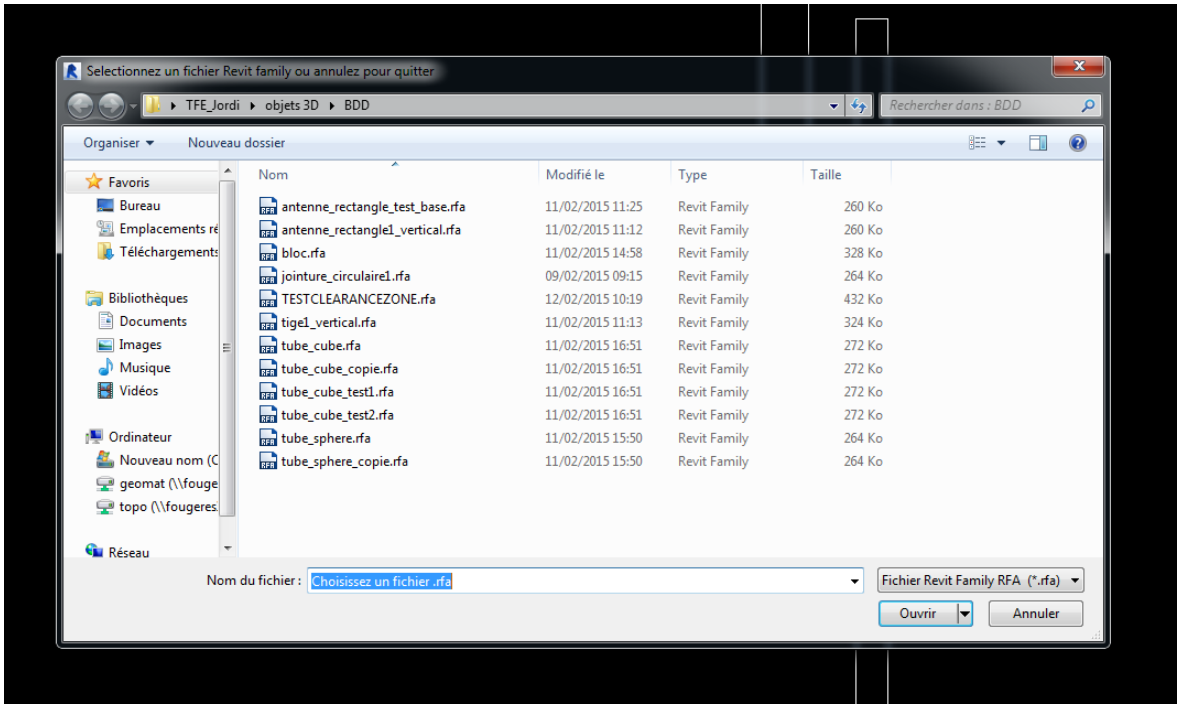


Figure 6 : Fenêtre obtenue avec FileSelectRfa

Avant d'insérer une famille dans un projet, il faut d'abord l'importer. Pour cela nous allons nous servir de **LoadFamily** qui prend en argument le nom de la famille et une variable de sortie pour stocker le résultat. La chaîne de caractères récupérée précédemment n'est donc pas adaptée. Il faut la diviser en 3, l'une contenant le chemin, l'autre le nom de la famille et la dernière l'extension. L'extension est la partie la plus simple puisque elle sera toujours .rfa. Le chemin et le nom peuvent quant à eux varier. Nous pouvons utiliser la fonction **GetFileNameWithoutExtension** qui prend en argument une chaîne de caractères décrivant le chemin complet, comme celle dont on dispose, et renvoie uniquement le nom du fichier. Pour trouver le chemin, il suffit de prendre le chemin complet et de trouver au sein de cette chaîne de caractères une chaîne correspondant au nom du fichier et son extension mis bout à bout. Cette sous chaîne est alors remplacée par du vide et il ne reste que le chemin.

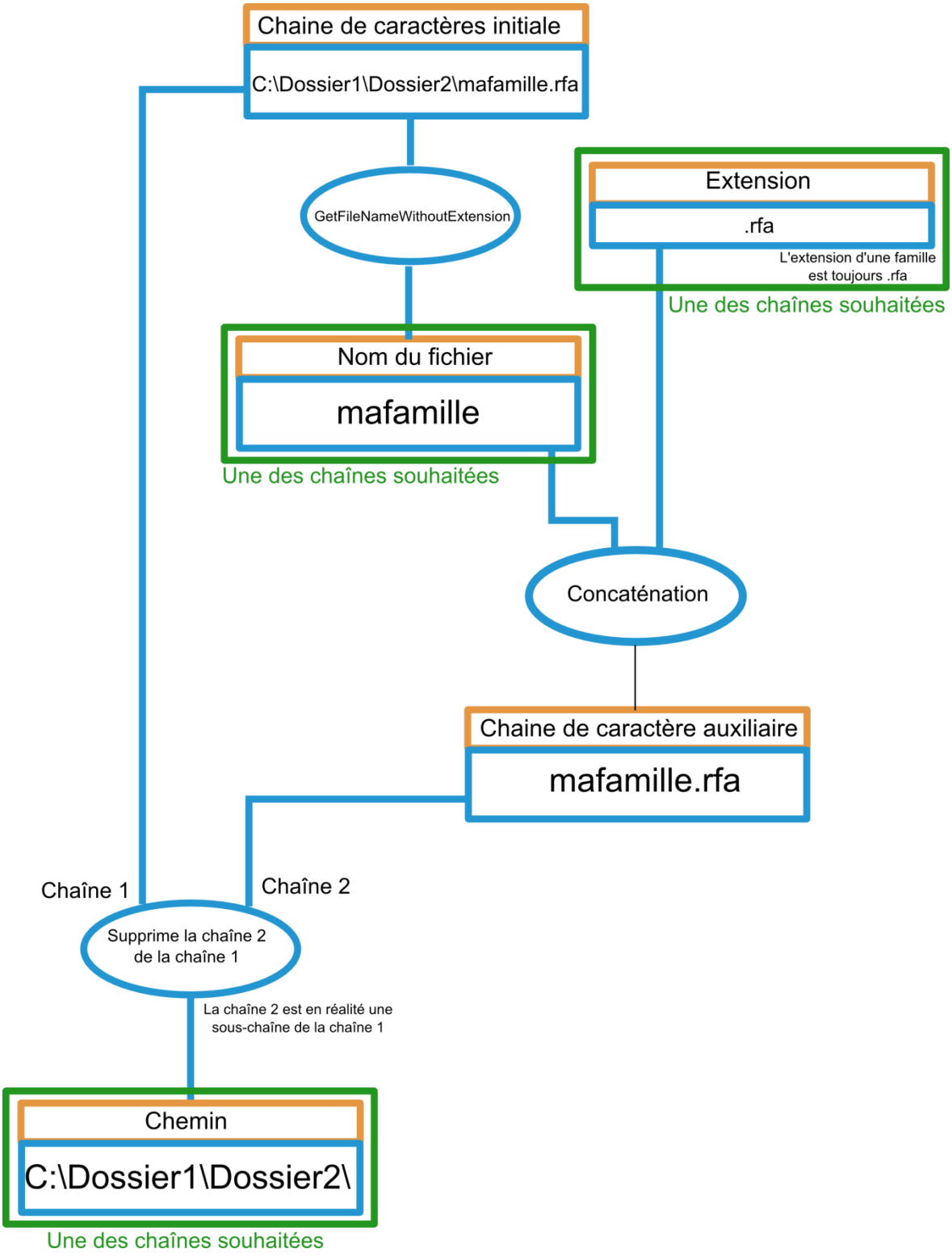


Figure 7 : Schéma de l'algorithme de récupération des chaînes de caractères

Nous pouvons alors utiliser la fonction **LoadFamily** et importer la famille. On pourrait penser qu'importer une famille déjà importée est une opération anodine et sans complications mais il n'en est rien. Si l'on tente d'effectuer une telle opération via l'API le plug-in s'arrête brutalement et le message suivant apparait :

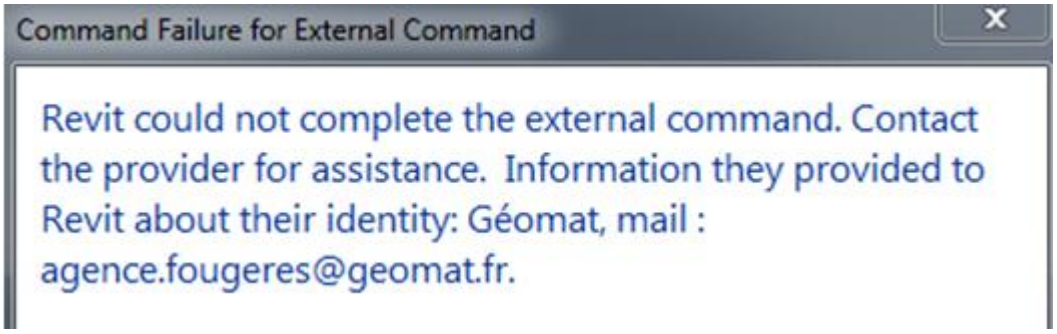


Figure 8 : Message d'erreur lors d'un arrêt brutal du plug-in
Traduction : Revit n'a pas pu compléter la commande externe. Contactez le fournisseur pour toute assistance. L'information qu'ils ont fourni à Revit à propos de leur identité : Géomat, mail :agence.fougères@geomat.fr

Il faut donc mettre en place des sécurités qui permettront d'éviter ce type d'erreur. Il faut donc vérifier si la famille est déjà importée. Si oui, nous la stockons simplement dans une variable que nous allons manipuler par la suite. Si non, nous l'importons. Nous allons donc passer en revue toutes les familles importées dans le projet, appliquer un filtre pour retenir uniquement celle dont le nom est le même que celle que l'utilisateur a sélectionné. Nous allons les compter et, si à l'issue du tri ce compte est 0, nous importons la famille et la stockons dans une variable, car elle est absente. Si ce compte n'est pas 0, nous prenons le 1er élément puisqu'il n'y en a qu'un et nous le stockons dans la variable que nous allons manipuler. De cette manière, on se prémunit d'une erreur fatale du plug-in. La famille est donc prête à être manipulée. Elle est stockée dans la variable family.

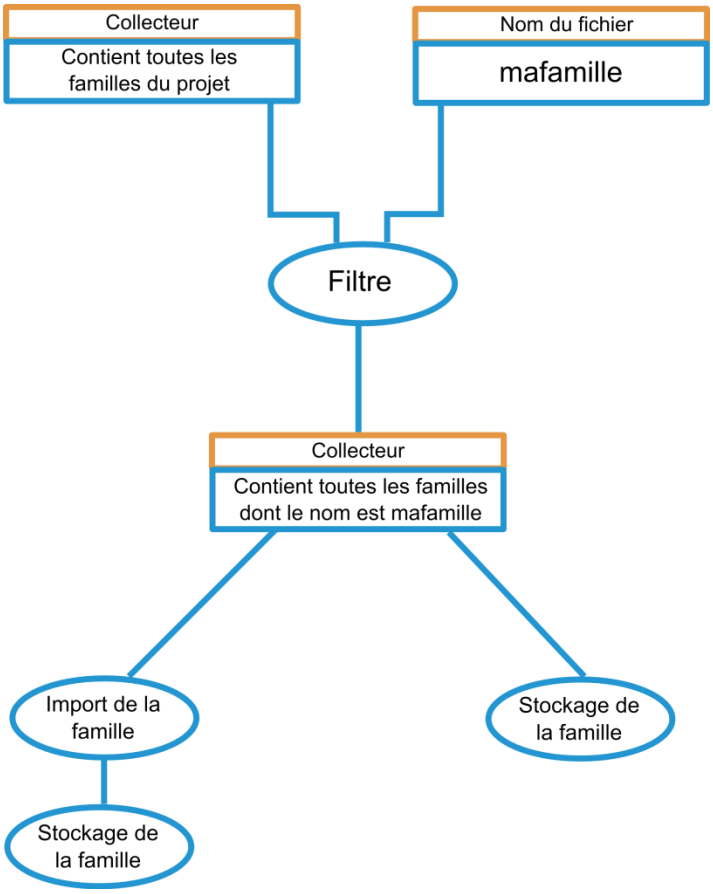


Figure 9 : Schéma de l'algorithme d'import

III.3.3.1 Cas de l'insertion sur un point

NewFamilyInstance prend dans ce cas comme argument un point d'insertion, un symbole et un type de structure. Un point est simplement un triplet de coordonnées. Nous allons donc pouvoir le demander à l'utilisateur grâce à la méthode **PickPoint** qui permet de cliquer un point à l'écran et récupérer ses coordonnées. Le symbole d'une famille est ce qui l'identifie de manière unique. La famille possède une géométrie, porte un nom, contient différents champs qui peuvent contenir de la metadonnée. Son symbole est ce qui est utilisé par Revit pour identifier cette famille. Le symbole est stocké dans la famille il faut donc le récupérer. Le type de structure est peu important pour nos besoins et nous pouvons le régler nous-mêmes sur **UnknownFraming** ce qui veut dire que nous ne le connaissons pas. Le symbole est situé dans la liste de symbole **family.Symbols**. Il suffit de prendre le premier de cette liste. Nous avons ainsi tous les éléments nécessaires à l'insertion de la famille sur un point.

III.3.3.2 Cas de l'insertion sur une face

L'insertion sur une face est plus délicate que sur un point et nécessite d'avantage d'arguments. Nous avons toujours besoin d'un point d'insertion mais nous devons nous assurer que celui-ci se trouve sur une face. Nous n'allons donc pas utiliser **PickPoint** car cette fonction ne permet pas d'accrocher des faces, seulement des points particuliers comme des angles d'objet. L'alternative consiste à sélectionner une face et ensuite appeler le point sur lequel la face a été sélectionnée. Ceci peut être accompli grâce à la fonction **PickObject** :

```
Reference r = uidoc.Selection.PickObject(ObjectType.Face,
  "Choisissez un point sur une face Pour l'insertion de l'objet");
```

Figure 10 : Utilisation de PickObject

En utilisant en premier argument **ObjectType.Face,** nous nous assurons que l'utilisateur ne pourra sélectionner que des faces. Pour récupérer les coordonnées du point, il suffit d'appeler l'attribut **GlobalPoint** de la référence.

Un symbole est toujours nécessaire, nous allons le récupérer de la même façon que précédemment. Le type de structure n'est lui plus nécessaire mais il laisse sa place à deux autres arguments. Le premier est une référence, référence que nous avons déjà récupérée pour notre besoin de point. Le second est une composante du vecteur qui définit le plan dans lequel la base de la famille se situera. Cela conditionne donc la direction de la famille. Ce vecteur, pour nos besoins, devra être tangent, au moins à une surface infinitésimale de la famille, sinon à la face entière sélectionnée. En fonction du type de face, la façon la plus simple de récupérer un tel vecteur peut varier. Par exemple pour une face cylindrique, il est possible d'obtenir la direction dans une variable v de la manière suivante :

```
XYZ v = face.axis.CrossProduct(XYZ.BasisZ);
```

Figure 11 : Récupération du vecteur dans le cas d'une face cylindrique

Tandis que pour une face plane nous utilisons :

```
XYZ v = face.normal.CrossProduct(XYZ.BasisZ);
```

Figure 12 : Récupération du vecteur dans le cas d'une face plane

Il existe aussi 4 autres types de face dans Revit qui sont « conical », « hermite » « revolved » et « ruled ». Si nous utilisons cette méthode de récupération du vecteur, il faut donc coder 6 cas différents en fonction du type de face. À ceci peut s'ajouter le fait que si les développeurs de Revit créent de nouveaux types de faces, notre script deviendrait en partie obsolète. Il faut donc récupérer ce vecteur d'une façon universelle, qui fonctionne pour les 6 types de faces différents déjà existant et les autres types de face n'existant pas encore. Pour parvenir à cela, nous allons utiliser la tangente à la face en un point qui peut être calculée grâce à la fonction **ComputeDerivatives**. Cette fonction prend en argument un **UVpoint** qui est un point dans l'espace 2D de la face. Il suffit d'appeler l'**UVpoint** de la référence sélectionnée précédemment. Cette tangente n'a en fait pas le type vecteur, il faut donc récupérer un de ses attributs qui peut être indifféremment **BasisX**, **BasisY** ou **BasisZ**. Nous disposons ainsi de tous les arguments nécessaires à l'insertion sur une face.

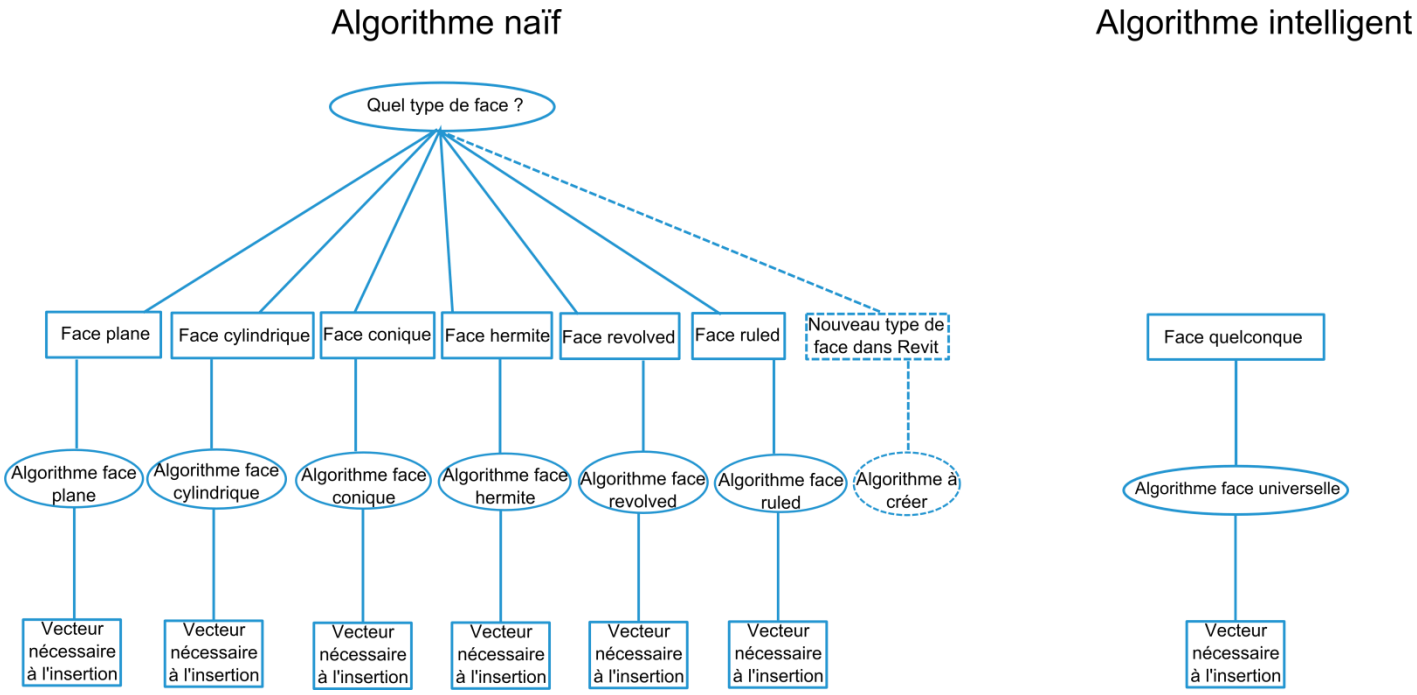


Figure 13 : Schéma de l'algorithme de calcul de vecteur

Une fois l'insertion effectuée, le choix entre la mise à jour et la vérification est laissé à l'utilisateur. Le premier choix donnera lieu à un traitement permettant le placement précis de la famille puis une recherche de collisions éventuelles. Le second débouche immédiatement sur la recherche de collisions. Voyons maintenant par quel moyen nous allons rendre le positionnement précis.

III.3.4 Mise à jour du projet

Un placement de la famille « à l'œil » n'est pas suffisant pour permettre une exploitation pérenne de ce plug-in. En effet de placement en placement, les imprécisions vont se cumuler et nous pourrons nous retrouver face à des situations où le plug-in affirmera qu'il n'y a pas la place pour l'équipement alors qu'en réalité cela est possible. Le

cas le plus défavorable étant le plug-in affirmant qu'il y a la place, une équipe se déplacera sur place pour installer l'équipement et se rendra compte que cela est en réalité impossible. Il faut donc garantir une certaine précision du placement dans le cadre d'une mise à jour. L'API ne doit pas ou très peu dégrader la précision des données qui est de l'ordre de 2cm.

Pour pouvoir garantir un placement dans le logiciel fidèle à la réalité, il faut disposer des coordonnées de l'équipement sur le site. Pour ce faire, un dispositif de mesures photogrammétriques très simple d'utilisation sera mis à disposition des équipes d'Axione. En plaçant des cibles spécifiques un peu partout sur le site et en prenant quelques photos, il est possible de déterminer les coordonnées de certains points. Ce système a été développé par la société FIT ESIC et porte le nom de système RAPH. Il est à l'intention de néophytes de la géomatique, ce qui en fait un outil parfaitement adapté à notre étude. A l'issue du traitement RAPH, nous allons obtenir un fichier texte contenant les coordonnées de toutes les cibles ainsi que des points de l'équipement qui ont été traités. Le fichier pouvant contenir parfois plusieurs points de plusieurs équipements différents, nous n'allons pas exploiter ce fichier directement. Il sera nécessaire à l'utilisateur, avant d'utiliser le plug-in, de créer un fichier texte contenant uniquement les coordonnées des points de l'équipement concerné.

```
1  996.407 4999.079 100.520
2  1000.196 4999.079 100.520
3  996.407 4999.079 99.900
4  1000.196 4999.079 99.9
5  1000.000 5000.000 100.0
```

Figure 14 : Fichier RAPH modifié

Il faudra alors stocker les coordonnées RAPH situées dans un fichier texte au sein de variables dans notre script. On va donc demander à l'utilisateur d'indiquer quel fichier texte est à traiter et lire ce fichier puis convertir les chaînes de caractères en nombres. Il faudra prendre soin de changer tous les points en virgule avant cette conversion sous peine de crash immédiat du plug-in dans le cas contraire.

Concernant les points de l'objet placé à l'œil, l'utilisateur devra cliquer sur les points correspondant dans le projet afin de pouvoir faire correspondre les coordonnées de la famille placée à l'œil avec les coordonnées déterminées via RAPH. La méthode **PickPoint** vue précédemment sera encore utilisée.

III.3.4.1 Récupération des coordonnées

À l'essai, **PickPoint** a fourni des résultats très étranges :

U	{double[4, 3]}
[0, 0]	0.04296244675158447
[0, 1]	0.50618586929505793
[0, 2]	0.50000000000007538
[1, 0]	-0.45703708106625174
[1, 1]	0.50667176111316337
[1, 2]	0.50000000000007538
[2, 0]	0.042962446751584504
[2, 1]	0.50618586929505793
[2, 2]	0.00000000000007539462387740 06
[3, 0]	-0.45703708106625174
[3, 1]	0.50667176111316337
[3, 2]	0.00000000000007539462387740 06

Figure 15 : Résultats de PickPoint

La matrice U contient des coordonnées de points cliqués à l'écran via **PickPoint**. U[0,0] contient la coordonnée Est du premier point cliqué, U[0,1] le Nord et U[0,2] l'Altitude. Les points se trouvent tous près du Survey Point et devraient donc avoir des coordonnées proches de (1000,5000,100). Il est primordial d'obtenir les coordonnées de ces points dans le même système que celui des résultats de RAPH sans quoi le traitement par moindres carrés est impossible. Ces coordonnées très petites font penser à un système local. Nous pouvons constater qu'en faisant des calculs de distances entre points selon chaque axe, on obtient les mêmes résultats qu'en utilisant les coordonnées du projet.

		E(m)	N(m)	Alti(m)	totale(m)	écart(m)	**< au mm ?**
dist 1-2	API	0,39962328	0,00036576	0	0,39962345	0,000376553	**OUI**
	Projet	0,4	0	0	0,4		
dist 1-3	API	0	0	0,39998904	0,39998904	1,096E-05	**OUI**
	Projet	0	0	0,4	0,4		
dist 1-4	API	0,39962328	0,00036576	0,39998904	0,56541147	0,000273954	**OUI**
	Projet	0,4	0	0,4	0,56568542		
dist 1-SP	API	1,68792144	0,00164592	0	1,68792224	7,89424E-05	**OUI**
	Projet	1,688	0,002	0	1,68800118		
dist 2-3	API	0,39962328	0,00036576	0,39998904	0,56541147	0,000273954	**OUI**
	Projet	0,4	0	0,4	0,56568542		
dist 2-4	API	0	0	0,39998904	0,39998904	1,096E-05	**OUI**
	Projet	0	0	0,4	0,4		
dist 2-SP	API	2,08754472	0,00201168	0	2,08754569	0,000455269	**OUI**
	Projet	2,088	0,002	0	2,08800096		
dist 3-4	API	0,39962328	0,00036576	0	0,39962345	0,000376553	**OUI**
	Projet	0,4	0	0	0,4		
dist 3-SP	API	1,68792144	0,00164592	0,39998904	1,7346679	7,93423E-05	**OUI**
	Projet	1,688	0,002	0,4	1,73474724		
dist 4-SP	API	2,08754472	0,00201168	0,39998904	2,1255207	0,000449198	**OUI**
	Projet	2,088	0,002	0,4	2,1259699		

Figure 16: Tableau d'analyse des coordonnées

Nous pouvons alors penser que ce système de coordonnées locales est une simple translation du système du projet. En demandant à l'utilisateur de cliquer également sur le Survey Point, nous pourrons récupérer ses coordonnées dans le système local et ainsi déterminer la translation à appliquer à nos coordonnées pour pouvoir les comparer aux coordonnées RAPH. Nous trouvons pourtant toujours des écarts allant jusqu'à plusieurs mètres avec les coordonnées attendues. Cette piste n'était donc pas la bonne.

En explorant la documentation sur la fonction **PickPoint**, nous pouvons constater que **PickPoint** ne renvoie pas les coordonnées du projet. Il renvoie une position dans un autre système. En réalité, **PickPoint** projette le point cliqué sur le WorkPlane qui possède son propre système de coordonnées 2D. Il couple ensuite la distance de projection à ce système pour renvoyer une position 3D.

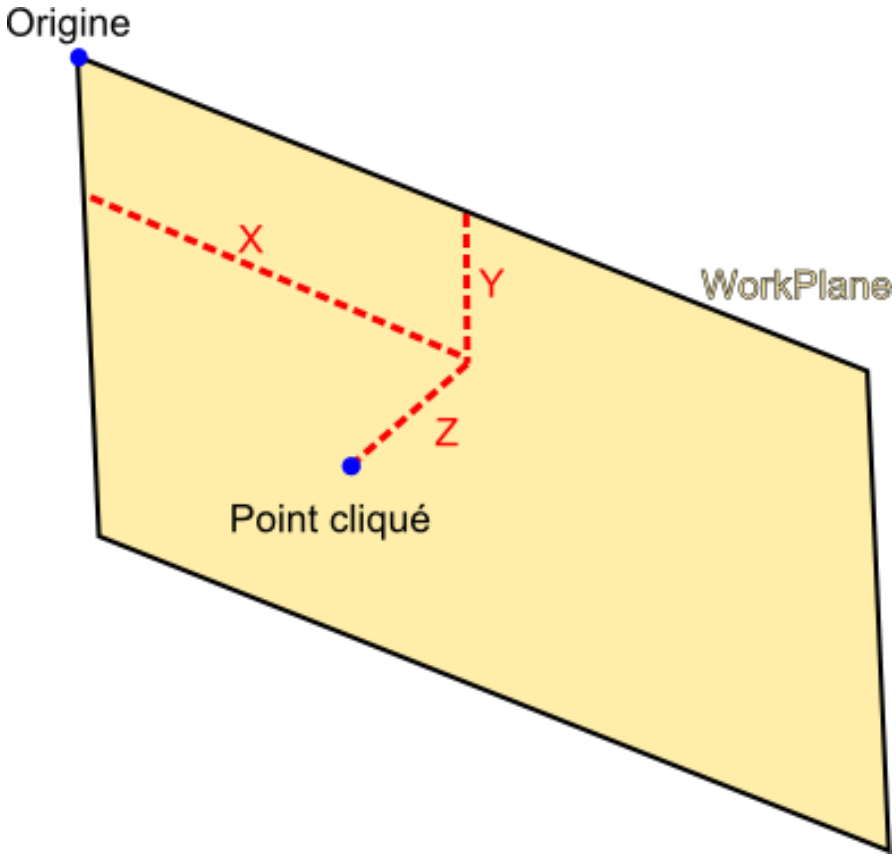


Figure 17 : Schéma du WorkPlane

Le WorkPlane dans Revit est en fait le plan dans lequel on dessine. Il peut être changé en le redéfinissant à l'aide d'une face plane par exemple. Si nous souhaitons utiliser **PickPoint**, il faut donc redéfinir le WorkPlane, et ce avant chaque sélection de chaque point. Il faudra également trouver le moyen de convertir ces coordonnées correctement. Pour simplifier ce traitement, nous allons utiliser une approche différente déjà utilisée lors de l'insertion sur une face. Nous allons donc sélectionner une face et récupérer le point sur laquelle cette face a été cliquée. Mais à nouveau, le système n'est pas celui recherché : les coordonnées sont dépendantes de la vue. Les choses qui peuvent paraître les plus simples et les plus intuitives dans Revit sont souvent les choses les plus compliquées. Pour obtenir les coordonnées qui nous intéressent il faut encore créer d'autres variables.

Nous allons créer les variables suivantes :

```
ProjectLocation projectLocation = uidoc.Document.ActiveProjectLocation;
ProjectPosition testco = null;
```

Figure 18 : Variables utilisée pour récupérer les coordonnées du projet

Utiliser la procédure suivante permet, à partir d'un point récupéré via **GlobalPoint**, d'obtenir les coordonnées dans le projet :

```
testco = projectLocation.get_ProjectPosition(coord);
```

Figure 19 : Fonction get_ProjectPosition

En observant les valeurs de ces coordonnées nous constatons qu'elles sont très loin de celles recherchées :

U	{double[4, 3]}
[0, 0]	3275.438850823517
[0, 1]	16404.369517581585
[0, 2]	326.09702134332207
[1, 0]	3273.8640892583048
[1, 1]	16404.369517581585
[1, 2]	326.07616648626936
[2, 0]	3275.4228732003221
[2, 1]	16404.369517581585
[2, 2]	324.53468872905711
[3, 0]	3273.856012657569
[3, 1]	16404.369517581585
[3, 2]	324.52214546826372

Figure 20 : Coordonnées des points après utilisation de get_ProjectPosition

Les coordonnées semblent avoir subi une multiplication d'un facteur légèrement supérieur à 3. Cela fait penser à une différence d'unité de mesure. En effet, peu importe les réglages du projet, l'API de Revit récupère des valeurs en pieds pour les distances, et des valeurs en radians pour les angles. Il faut donc effectuer une conversion afin d'obtenir les coordonnées. Une multiplication par 0.3048 sera suffisante compte tenu de la précision attendue. En effet les écarts entre les coordonnées obtenues et celles affichées dans Revit sont inférieurs au millimètre[3]. À l'issue de cette conversion nous obtenons finalement des coordonnées exploitables :

U	{double[4, 3]}
[0, 0]	998.36150205668548
[0, 1]	5000.0518577449393
[0, 2]	99.399363928948858
[1, 0]	997.86404067307944
[1, 1]	5000.0518577449393
[1, 2]	99.396698516328343
[2, 0]	998.36054421794483
[2, 1]	5000.0518577449393
[2, 2]	98.902078981171968
[3, 0]	997.86515605958084
[3, 1]	5000.0518577449393
[3, 2]	98.903614628428954

Figure 21 : Coordonnées des points après conversion en mètres

Nous disposons maintenant des coordonnées de départ et d'arrivée. Le moyen le plus simple pour parvenir à déplacer notre famille est de calculer la transformation puis de la lui appliquer. Nous allons donc déterminer une similitude 3D sans facteur d'échelle

[3] cf figure 16

puisque notre objet n'est pas déformable. Cela revient à calculer 3 rotations et 3 translations.

III.3.4.2 Calcul de la similitude 3D

Nous cherchons à déterminer 6 paramètres. La méthode la plus simple que nous pouvons utiliser est la méthode des moindres carrés. Pour cela il nous faut une redondance et donc plus de données que d'inconnues. Un point fournissant 3 données, il faudra un minimum de 3 points pour pouvoir traiter les équations. Il faut donc mettre en place un arrêt du plug-in avec un message d'erreur avertissant l'utilisateur si son fichier ne comporte pas suffisamment de points.

Pour pouvoir appliquer les moindres carrés, il convient de poser les équations correspondant aux relations entre points homologues.

Soit R la matrice colonne des coordonnées RAPH, U la matrice colonne des coordonnées de la famille placée à l'œil, T la matrice translation et Rot la matrice rotation, on a :

$$R = U + T + Rot \times U$$

U et R étant connues, on va utiliser l'équation suivante :

$$R - U = Rot \times U + T$$

La matrice rotation est le produit de 3 matrices correspondant aux rotations autour de chaque axe avec un angle différent :

$$Rot = Rotx(i) \times Roty(j) \times Rotz(k)$$

$$Rotx(i) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(i) & -\sin(i) \\ 0 & \sin(i) & \cos(i) \end{pmatrix}$$

$$Roty(j) = \begin{pmatrix} \cos(j) & 0 & \sin(j) \\ 0 & 1 & 0 \\ -\sin(j) & 0 & \cos(j) \end{pmatrix}$$

$$Rotz(k) = \begin{pmatrix} \cos(k) & -\sin(k) & 0 \\ \sin(k) & \cos(k) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

D'où :

$$Rot = \begin{pmatrix} \cos(j) \times \cos(k) & -\cos(j) \times \sin(k) & \sin(j) \\ \sin(i) \times \sin(j) \times \cos(k) + \cos(i) \times \sin(k) & -\sin(i) \times \sin(j) \times \sin(k) + \cos(i) \times \cos(k) & -\sin(i) \times \cos(j) \\ -\cos(i) \times \sin(j) \times \cos(k) + \sin(i) \times \sin(k) & \cos(i) \times \sin(j) \times \sin(k) + \sin(i) \times \cos(k) & \cos(i) \times \cos(j) \end{pmatrix}$$

Et T la matrice translation :

$$T = \begin{pmatrix} Tx \\ Ty \\ Tz \end{pmatrix}$$

Les paramètres que nous cherchons à déterminer sont donc Tx, Ty, Tz, i, j, k. Étant donné que la matrice Rot comporte des sinus et cosinus, le traitement est non linéaire. J'ai décidé de réaliser le traitement par moindres carrés en language Python plutôt qu'en language C# car j'avais déjà effectué des traitements similaires dans ce langage dans le cadre du PPP de l'ESGT. Pour pouvoir faire cela, il faut écrire toutes les données nécessaires au traitement dans des fichiers textes que le script Python lira. Après traitement, le script Python écrira les solutions dans un fichier texte que le script C# lira à son tour pour stocker les solutions dans des variables. Vu le faible nombre de données, le traitement sera très rapide. Le script Python sera lancé par le script C#, aucune fenêtre ne sera ouverte et l'utilisateur ne verra rien de particulier. Le traitement étant non linéaire, des paramètres approchés seront nécessaires. Ils seront tous initialisés à 1 pour éviter des erreurs étranges qui pourraient apparaître s'ils étaient approximés à 0. Une fois les solutions stockées dans le script en C#, il ne reste plus qu'à appliquer les transformations à notre famille.

III.3.4.3 Application des transformations

Pour appliquer des rotations nous allons utiliser la fonction **RotateElement** qui prend en argument un document, en l'occurrence celui ouvert, un identifiant d'élément, un axe et un angle.

L'angle est déjà connu puisque contenu dans les solutions des moindres carrés.

L'axe est facile à définir grâce à la fonction **CreateBound** qui prend en argument 2 points. Il suffira alors de créer 3 axes de cette manière où le 1er argument est systématiquement le point (0,0,0) et le 2ème sera un point de l'axe souhaité.

Le document est lui aussi très simple à récupérer : il est situé dans **commandData.Application.ActiveUIDocument.Document**.

L'identifiant d'élément est un attribut de l'élément. Il nous faut donc récupérer l'élément que nous souhaitons déplacer. Nous allons le demander à l'utilisateur à l'aide de **PickObject** mais cette fois le 1er argument sera **ObjectType.Element** et non **Face**. On dispose alors de tout ce dont on a besoin pour appliquer la rotation.

A l'application de cette rotation, nous obtenons le message d'erreur suivant :

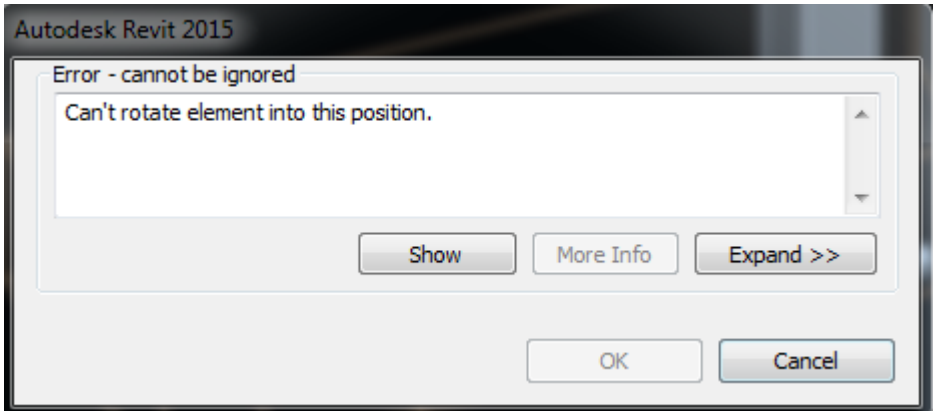


Figure 22 : Message d'erreur après rotation

En effet, Revit ne permet pas d'effectuer des rotations sur des familles facilement. Encore une fois, les choses qui paraissent les plus évidentes sont les plus compliquées. Pour pallier à cela, nous allons recourir à un subterfuge : Revit refuse d'effectuer différentes rotations sur des familles mais pas sur des groupes. Les groupes sont un ensemble d'éléments que l'on a groupé. Cela peut être utile pour créer la maquette 3D d'un hôtel où de nombreuses chambres sont identiques. On groupe tout le mobilier d'une chambre et on le copie dans les autres chambres. Dans notre cas nous allons simplement grouper notre famille, y appliquer les transformations et la dégrouper. Cela permet de se prémunir de nombreuses erreurs. Malgré cela, nous pouvons à nouveau rencontrer le même message d'erreur lors du dégroupage. En effet, Revit n'autorise les rotations des familles que lorsque celles-ci ne modifient pas l'inclinaison par rapport au plan à partir duquel elles ont été insérées. Dans le cas de l'insertion par un point, c'est le plan horizontal, dans le cas de l'insertion par une face, c'est la face où le plan tangent à la face.

Ce sont donc les arrondis sur les résultats des moindres carrés qui engendrent ces erreurs. Nous pourrions donc choisir de n'appliquer que la rotation de plus grand angle. En pratique cela peut engendrer des erreurs car une rotation autour de l'axe Z de 180° équivaut à une rotation autour de l'axe X de 180°, puis une rotation autour de l'axe Y de 180°, résultats que les moindres carrés produisent parfois. On va donc plutôt appliquer des rotations uniquement si les angles sont supérieurs à un seuil de 0.001 radian.

Maintenant que les rotations s'appliquent correctement, appliquons les translations. On définit une translation par un triplet de coordonnées. Nous l'appliquons ensuite grâce à la fonction **MoveElement** qui prend en argument un document (que nous avons déjà), un identifiant d'élément (que nous avons déjà également), et une translation. Après application de la translation, aucun message d'erreur ne s'affiche mais la famille est introuvable. Pour savoir ce qu'il s'est passé nous pouvons regarder le fichier texte contenant les solutions des moindres carrés :

```
1.9998244001641207 89e+03
9.9908683501896648 52e+03
4.9877417587629346 85e-01
4.4047939071453235 13e-05
-2.7897528034711622 68e-04
3.1415926069359683 13e+00
```

Figure 23 : Résultats des moindres carrés

On constate que les translations sont très grande par rapport à ce que l'on peut attendre. Notre objet a été déplacé à plus de 10Km de son emplacement initial.

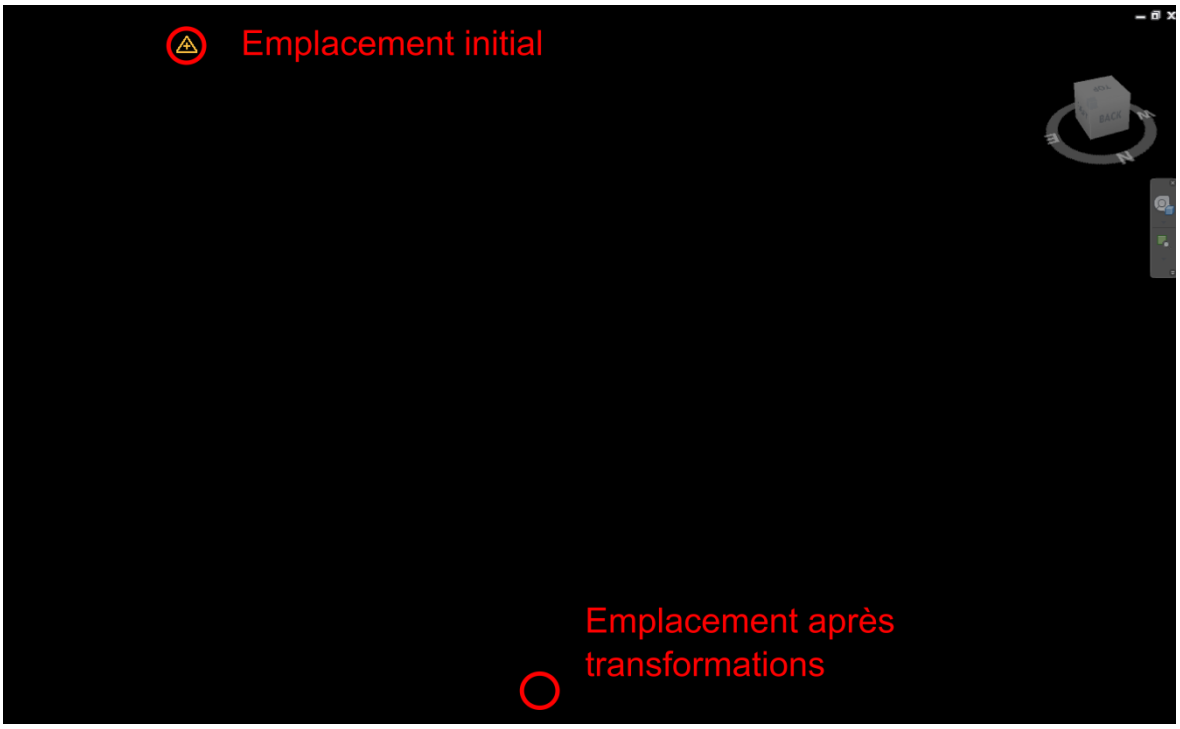


Figure 24 : Anomalie lors des transformations

Pourtant des translations aussi grandes sont normales : plus nous nous trouvons loin de l'origine, plus l'impact d'une rotation est fort sur la position. Pour compenser cela les rotations sont grandes. Nous pouvons alors supputer que la rotation ne s'applique non pas à l'origine mais depuis un point quelconque, apparemment proche de la position initiale. Ce point est peut-être le Survey Point.

Pour pallier à ça nous pouvons considérer que le Survey Point est l'origine et soustraire ses coordonnées à tous les points RAPH et les points de la famille placés à l'œil avant le traitement par moindres carrés. Pour obtenir les coordonnées du Survey Point, nous pouvons demander à l'utilisateur de rajouter une ligne au fichier texte de RAPH qui correspondrait aux coordonnées du Survey Point. Il pourra trouver ces coordonnées simplement en ouvrant son projet et en cliquant sur le Survey Point symbolisé par un triangle jaune. Ainsi, les nouvelles solutions sont d'un ordre de grandeur raisonnable et l'objet reste à l'écran.

```
1  -1.523500505465602783e+00
2  -6.912000282524788375e+00
3  -9.359371773566290134e-07
4  1.057714216621194806e-07
5  -1.687976139158507474e-08
6  3.141592568866166335e+00
```

Figure 25 : Résultats des moindres carrés après conversion en mètres

En revanche l'objet n'est toujours pas à l'emplacement attendu. Lorsque nous répétons les opérations du plug-in, nous nous rendons compte que la famille arrive toujours à des points différents mais toujours avec la même orientation : le point d'où s'appliquent les rotations doit être lié à la vue. Il suffit alors de replacer l'objet à l'aide d'une translation. Nous pouvons déterminer cette dernière par moindres carrés mais cette fois, les équations sont bien plus simples. On récupère les nouvelles coordonnées de l'objet ayant subi a première transformation dans U et on peut alors calculer une nouvelle transformation.

$$R - U = T$$

On se trouve face à une équation linéaire. On a alors :

$$B = R - U$$

$$X = T = \begin{pmatrix} Tx \\ Ty \\ Tz \end{pmatrix}$$

D'où :

$$B = A \times X$$

Avec :

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

On pose alors :

$$N = A^T \times A$$
$$C = A^T \times B$$

D'où :

$$X = N^{-1} \times C$$

On applique finalement les translations issues du second traitement et l'objet est placé à la position souhaitée.

Á l'application des rotations et translations, au n^{ème} lancement du plug-in, la famille semble subir les transformations du traitement n-1. Il est possible que l'action d'écrire et de lire soit tellement rapide que les modifications du fichier texte n'ont pas le temps d'être effectivement sauvegardées. Nous allons donc, après chaque écriture de fichier texte, c'est-à-dire avant et après l'exécution du script python, demander à l'ordinateur de stopper toute action grâce à la commande **System.Threading.Thread.Sleep(2000)**, 2000 étant le nombre de millisecondes d'arrêt. Il va donc y avoir un temps de chargement de 4 secondes au milieu du plug-in pour que l'écriture et la lecture des données des moindres carrées se déroulent dans de bonnes conditions.

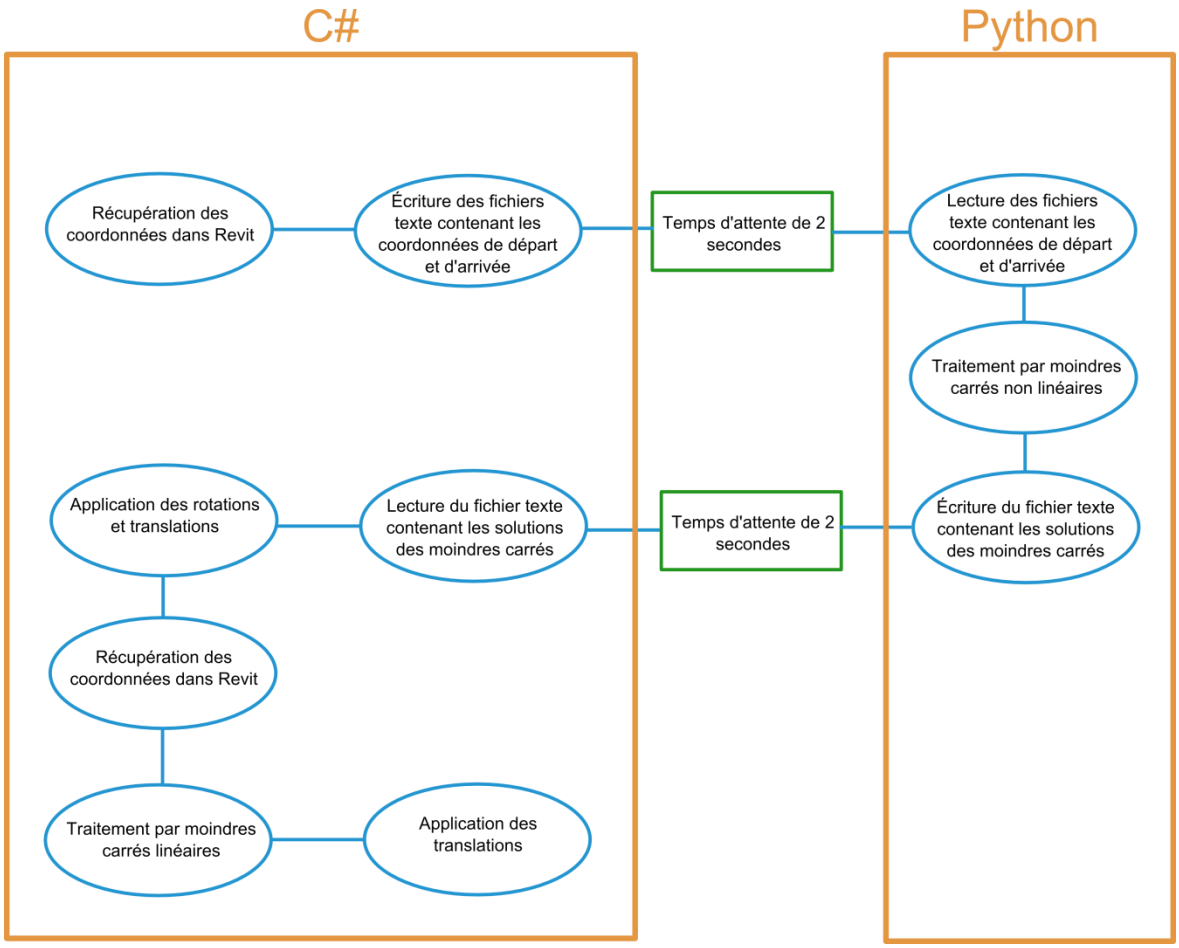


Figure 26 : Schéma de l'algorithme de calcul des transformations

III.3.5 Détermination et stockage de la qualité

La qualité du placement de la famille va dépendre de la qualité du traitement RAPH et de la précision avec laquelle l'utilisateur va cliquer sur les points nécessaires au traitement par moindres carrés. Il est important de catégoriser cette qualité afin que l'utilisateur sache si son opération est satisfaisante du point de vue de la précision. Pour cela il faudra veiller à créer un champ vide lors de la création des familles qui seront manipulées par les utilisateurs. Ce champ sera nommé « Quality » et sera de type texte. On le remplira de A, B, C ou D en fonction des écarts des coordonnées des points dans le projet avec les points dans le fichier RAPH.

Quality	Écart
A	écart<5mm
B	5mm≤écart<1cm
C	1cm≤écart<1.5cm
D	écart ≥1.5cm

Figure 27 : Tableau des catégories de précision

Nous allons également afficher un rapport indiquant si les écarts sont supérieurs à 2cm et la valeur du plus grand écart.

III.3.6 Recherche de collisions

Un test de collision dans Revit s'effectue grâce au menu « collaborate » du premier ruban, puis dans la partie « Coordinate » il faut utiliser le raccourci « Interference Check » puis « Run Interference Check ».

Cependant, ceci n'est pas possible via l'API. En effet, bien que Revit et son API sont développés en parallèle, l'API a toujours un train de retard sur le logiciel. Ainsi, toutes les fonctionnalités du logiciel ne sont pas présentes dans l'API.

La façon la plus simple de réaliser un tel test, via l'API, est d'utiliser des BoundingBox. Une BoundingBox est une boîte rectangulaire qui englobe notre objet. Cette BoundingBox dispose de propriétés qui nous permettent de trouver tout élément se trouvant pour tout ou en partie à l'intérieur de la BoundingBox. De cette manière, nous allons pouvoir lister ces éléments, en prenant soin d'exclure la famille en cours de test puisqu'elle est nécessairement à l'intérieur de la BoundingBox , et écrire un rapport à l'utilisateur.

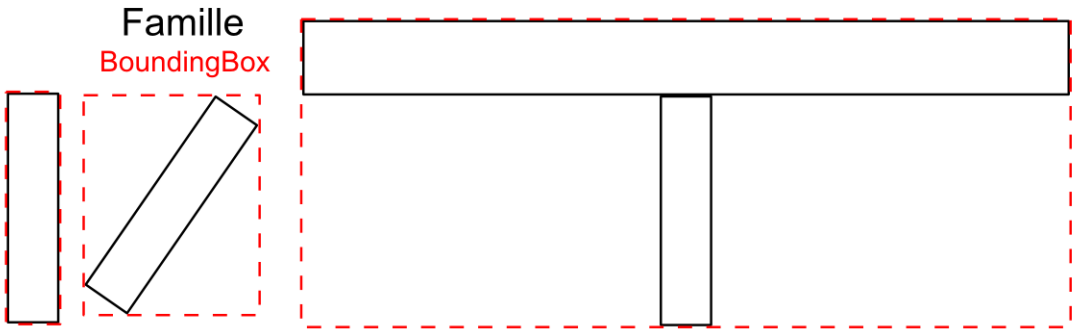


Figure 28 : Exemples de BoundingBox

Le problème d'une telle méthode est la précision. En effet nous détectons les éléments dont les BoundingBox intersectent celle de la famille. Ce n'est donc pas un test de collision famille à famille. En fonction de la géométrie de la famille et de son orientation, les résultats seront très différents.

Nous allons donc générer un rapport d'erreur pour des intersections qui n'existent pas en réalité. En revanche, les intersections existant en réalité seront bel et bien détectées. Il va alors être noté dans le rapport que ce sont des collisions « possibles » et conseiller à l'utilisateur de lancer son propre test de collision via Revit en lui indiquant les menus à suivre. Dans le cas où aucune intersection n'est détectée, nous signalerons simplement à l'utilisateur que sa famille ne présente aucune collision. Aucun traitement supplémentaire n'est nécessaire.

Un tel test est donc peu satisfaisant car nécessitant des traitements supplémentaires. En utilisant **ElementIntersectsElementFilter**, nous allons obtenir rigoureusement les éléments s'intersectant, et nous allons pouvoir afficher leurs nom et identifiant à l'écran. Si ces derniers ne sont pas suffisants pour déterminer quelles familles intersectent la famille placée, nous pouvons proposer à l'utilisateur de lancer **l'Interference Check** de Revit qui met en surbrillance les éléments s'intersectant et facilite ainsi leur identification.

III.3.7 Algorithme final

Du fait que le placement initial de l'objet ne peut être effectué à la main, lancer le test de collision dans une procédure de vérification n'a pas grand intérêt. Après le placement de l'objet en dehors du plug-in, l'utilisateur devra lancer le test de collision de Revit lui-même. L'algorithme final est donc quelque peu différent de celui envisagé.

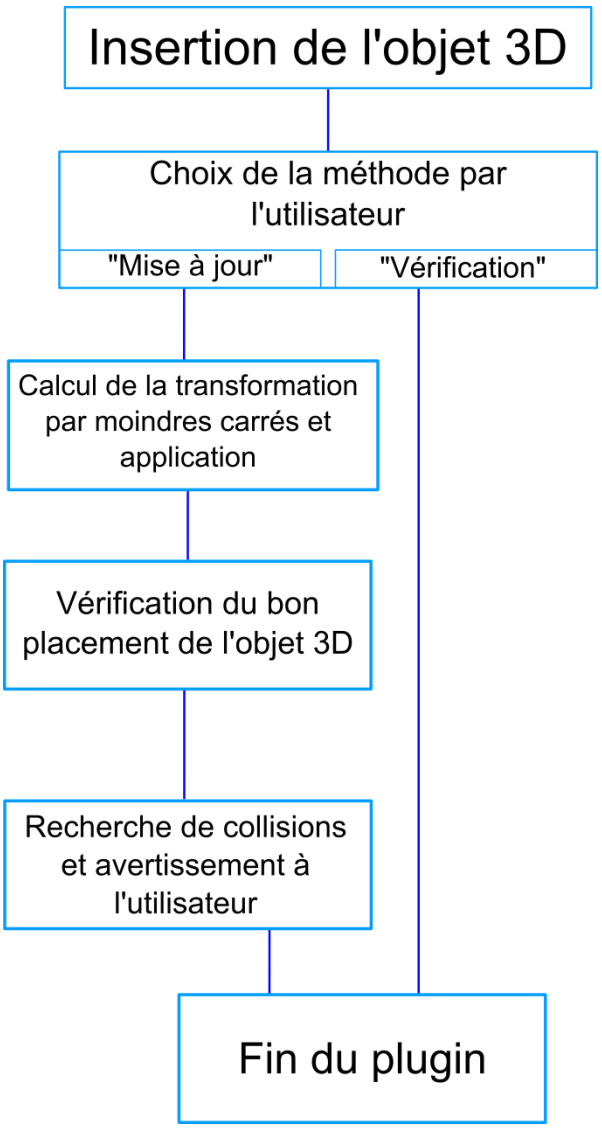


Figure 29 : Algorithme final

IV Démonstration

Le plug-in est lancé depuis le menu « Add-Ins » du premier ruban, en sélectionnant « External Tools » puis « Command PlaceAndCheck » :

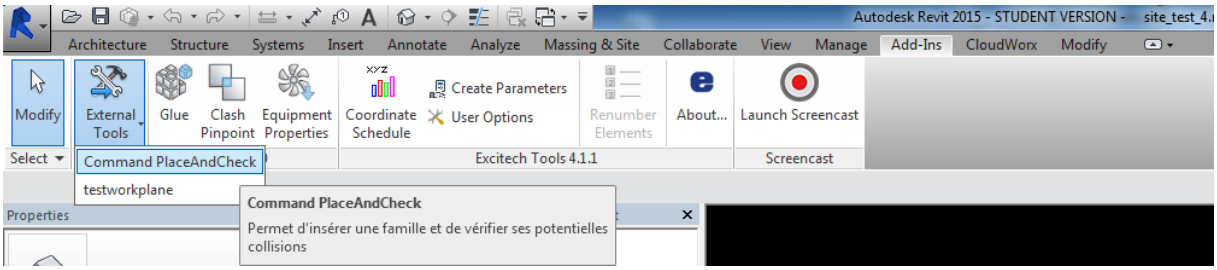


Figure 30 : Lancement du plug-in

Tout au long du plug-in, des instructions seront communiquées à l'utilisateur, soit par le biais de messages au milieu de l'écran, soit par des messages dans la barre d'indication contextuelle en bas à gauche de l'écran.

Ayant été développé pour une utilisation dans un document ouvert, et dans l'éventualité d'un lancement sans aucun document ouvert, un message d'erreur a été prévu pour informer l'utilisateur de sa mauvaise manipulation :

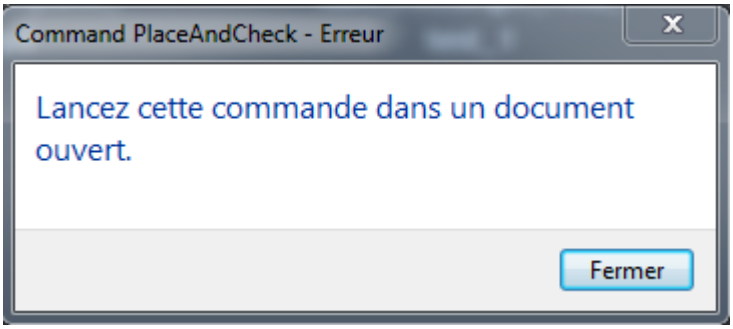


Figure 31 : Message d'erreur lors du lancement du plug-in sans document ouvert

Pour cette démonstration nous allons travailler avec le projet fictif suivant :

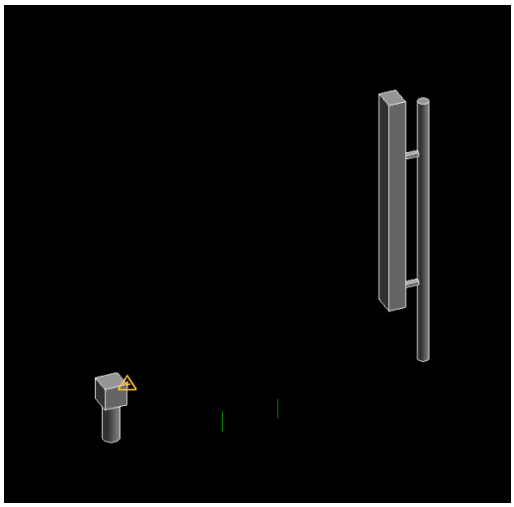


Figure 32 : Projet de base

Si le plug-in est correctement lancé, la fenêtre de choix de la famille à insérer apparait :

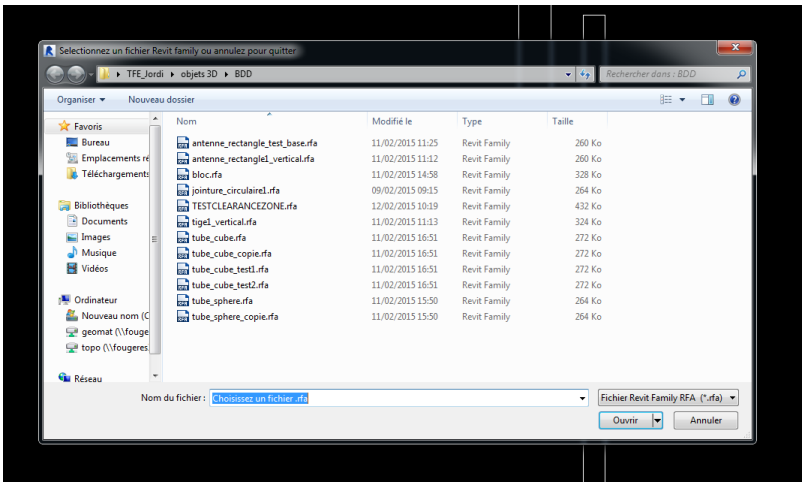


Figure 33 : Fenêtre de choix de la famille

Une fois la famille sélectionnée, un choix est proposé à l'utilisateur :

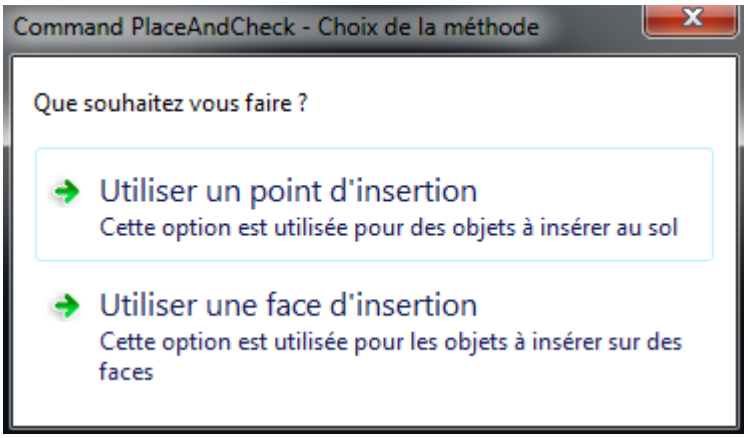


Figure 34 : Choix de la méthode

Un simple clic suffit alors à insérer l'objet dans le projet, que ce soit par la méthode du point d'insertion ou par celle de la face d'insertion :

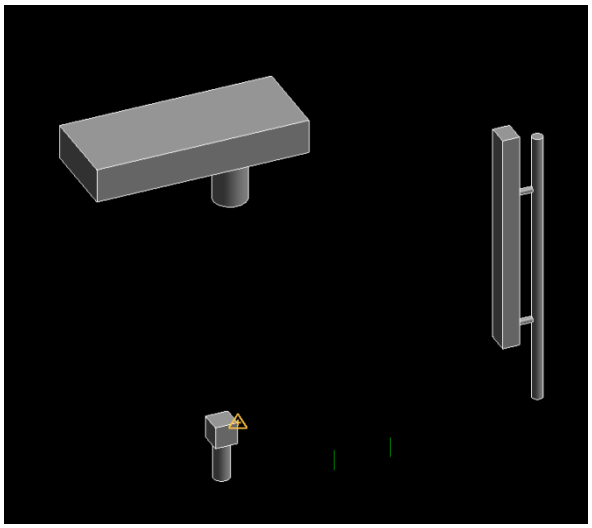


Figure 35 : Projet après le 1er placement de la famille

L'utilisateur est à nouveau confronté à un choix : la procédure mise à jour dans le cadre d'une opération post-intervention terrain, ou la procédure de vérification dans le cadre d'une opération pré-intervention terrain. Si cette dernière option est sélectionnée, le plug-in s'arrête en affichant quelques instructions à l'utilisateur pour lui permettre de bien placer l'objet avant de vérifier la présence d'éventuelles collisions :

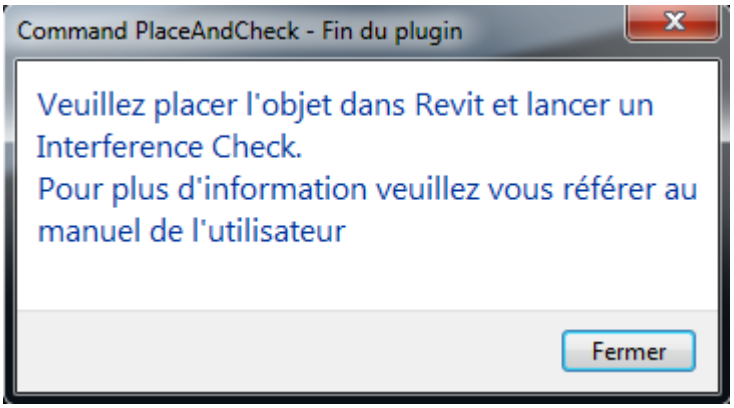


Figure 36 : 1er message de fin du plug-in dans le cas d'une vérification

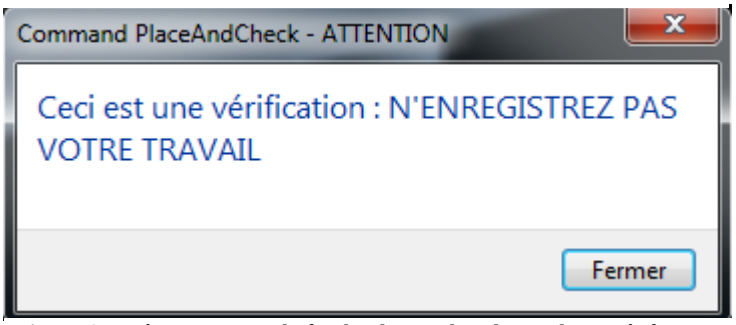


Figure 37 : 2ème message de fin du plug-in dans le cas d'une vérification

Si c'est l'option de mise à jour qui est sélectionnée, le plug-in continue et demande à l'utilisateur de sélectionner un fichier RAPH au format txt :

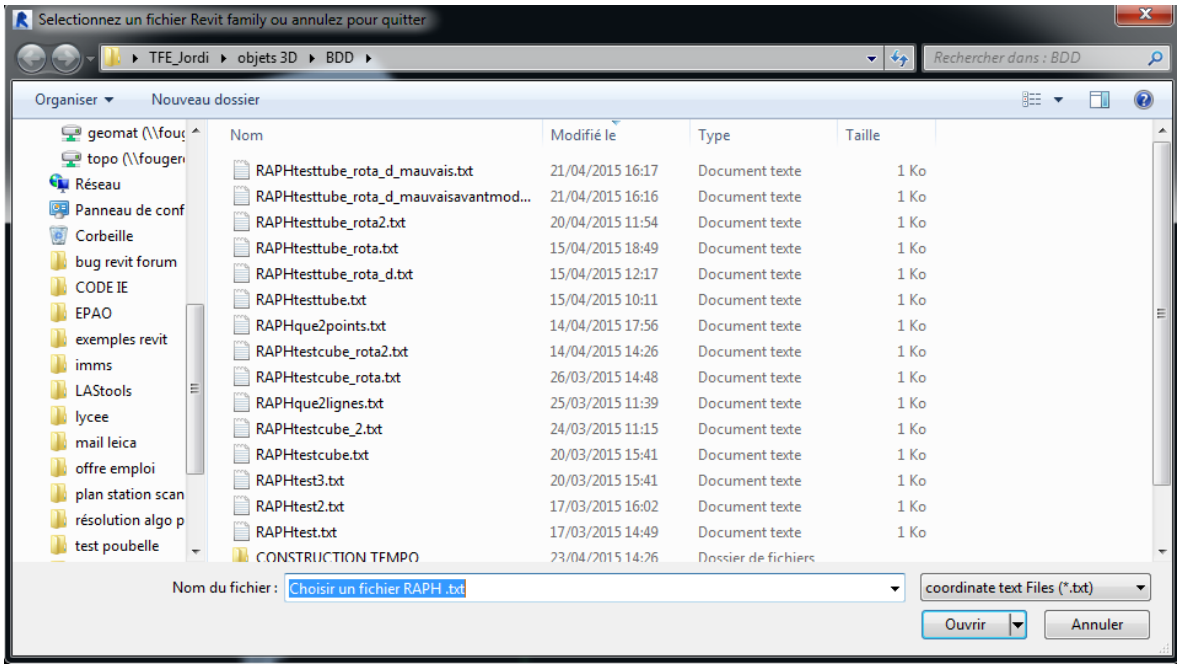


Figure 38 : Choix du fichier RAPH

Si le fichier RAPH sélectionné ne contient pas assez de points, c'est-à-dire 3 plus le Survey Point, le message suivant s'affiche et le plug-in s'arrête :

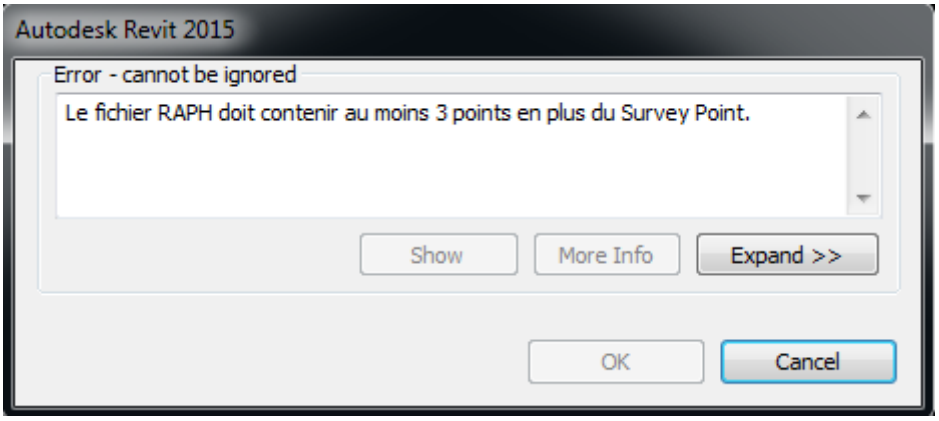


Figure 39 : Message d'erreur de contenu du fichier RAPH

En revanche, si le fichier RAPH est bien construit, l'utilisateur doit cliquer successivement sur les points correspondant à ceux du fichier RAPH. Va suivre un temps de chargement de 4 secondes, puis l'utilisateur devra sélectionner 2 fois de suite la famille à déplacer, la première fois pour la grouper et la deuxième fois pour lui appliquer les transformations et la dégrouper. On peut alors voir que la famille a changé de position et surtout d'orientation :

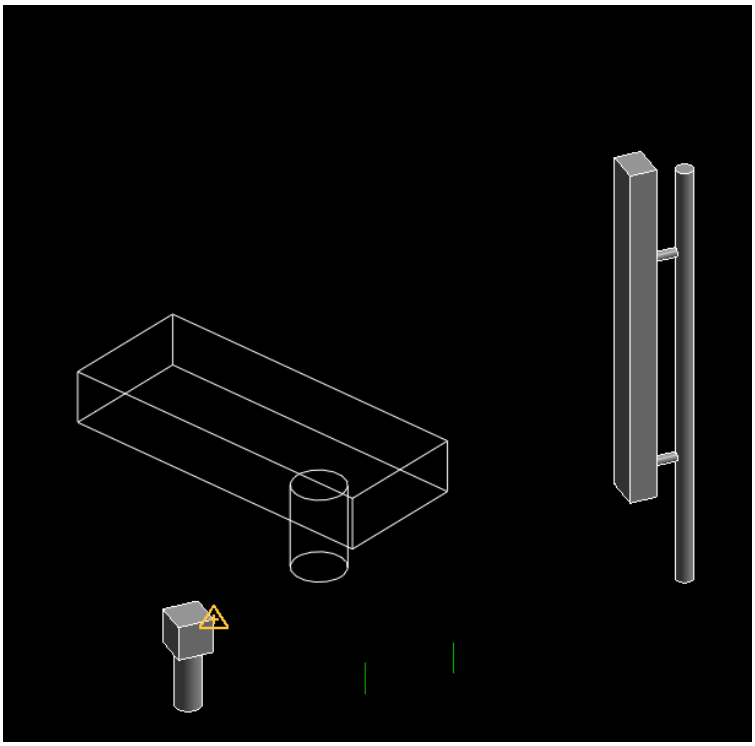


Figure 40 : État du projet après la première transformation

L'utilisateur doit alors cliquer à nouveau sur les points dans le même ordre afin d'effectuer la seconde transformation :

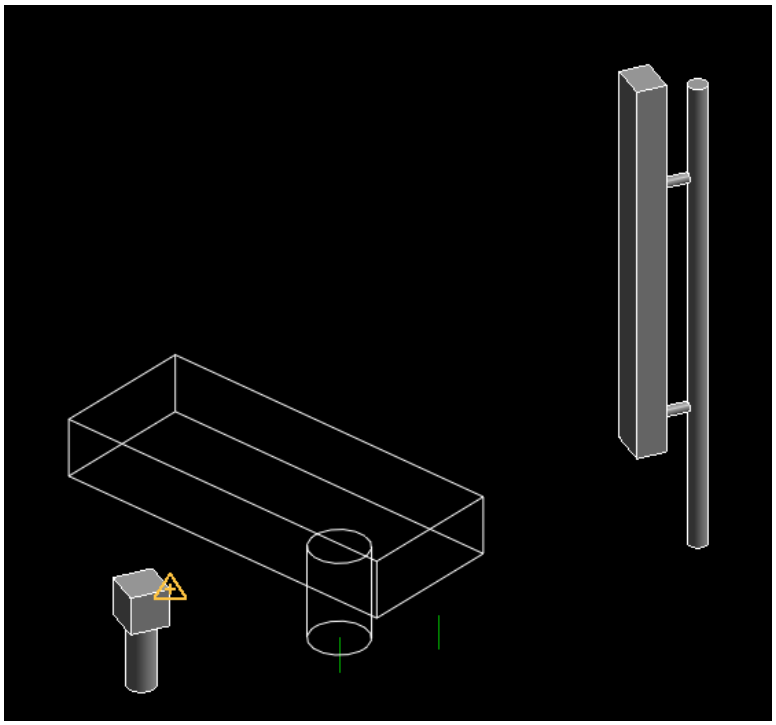


Figure 41 : État du projet après la seconde transformation

Il faut alors cliquer de manière précise les points toujours dans le même ordre afin de déterminer la qualité du traitement. Si le placement est juste, le message suivant apparait :

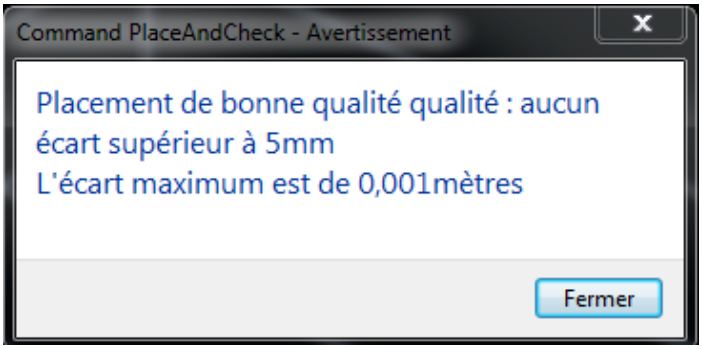


Figure 42 : Message relatif à la bonne qualité du placement

Si le placement laisse plus à désirer, un message de ce type apparait :

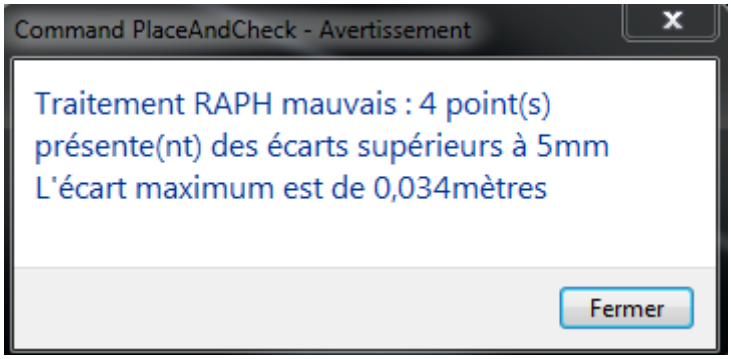


Figure 43 : Message d'écart maximum

L’utilisateur doit enfin sélectionner la famille pour laquelle on va vérifier la présence d’éventuelles collisions. Le message suivant s’affiche alors :

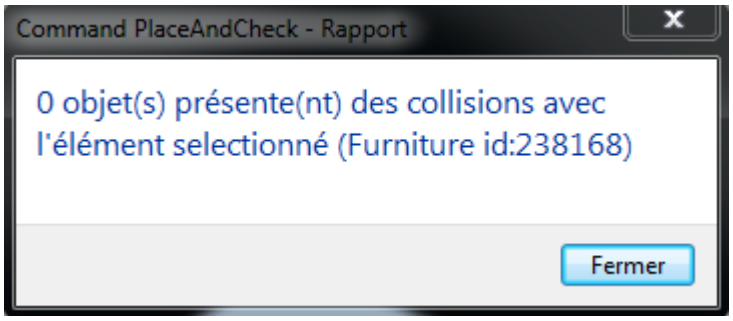


Figure 44 : Rapport de collisions

S’il y a présence de collisions, deux autres messages vont s’afficher

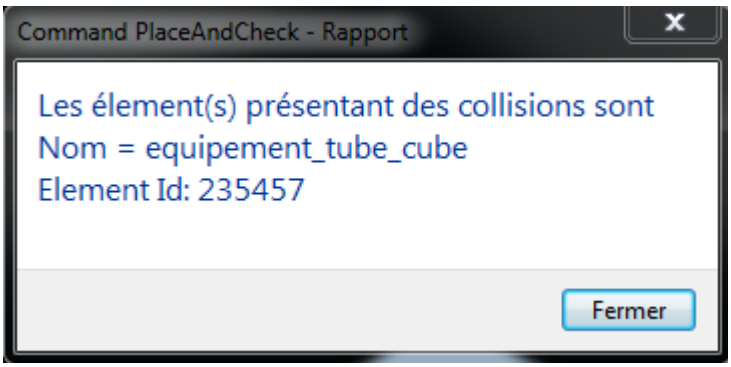


Figure 45 : Rapport de collisions

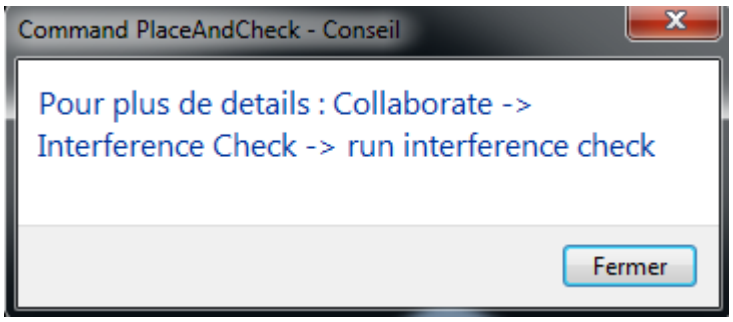


Figure 46 : Instruction de lancement de Interference Check

Le plug-in va ensuite se terminer par ce message :

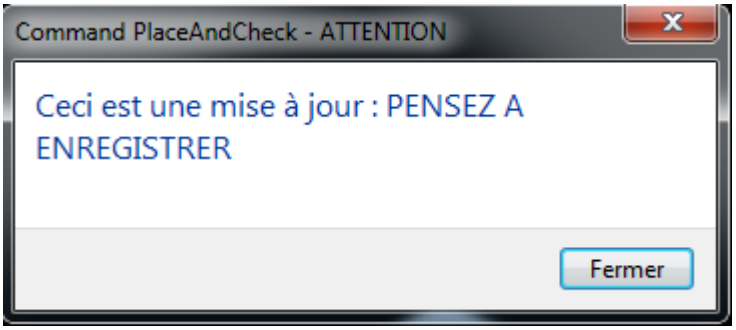


Figure 47 : Message d'enregistrement

On peut alors voir la famille correctement placée ainsi que le paramètre qualité renseigné :

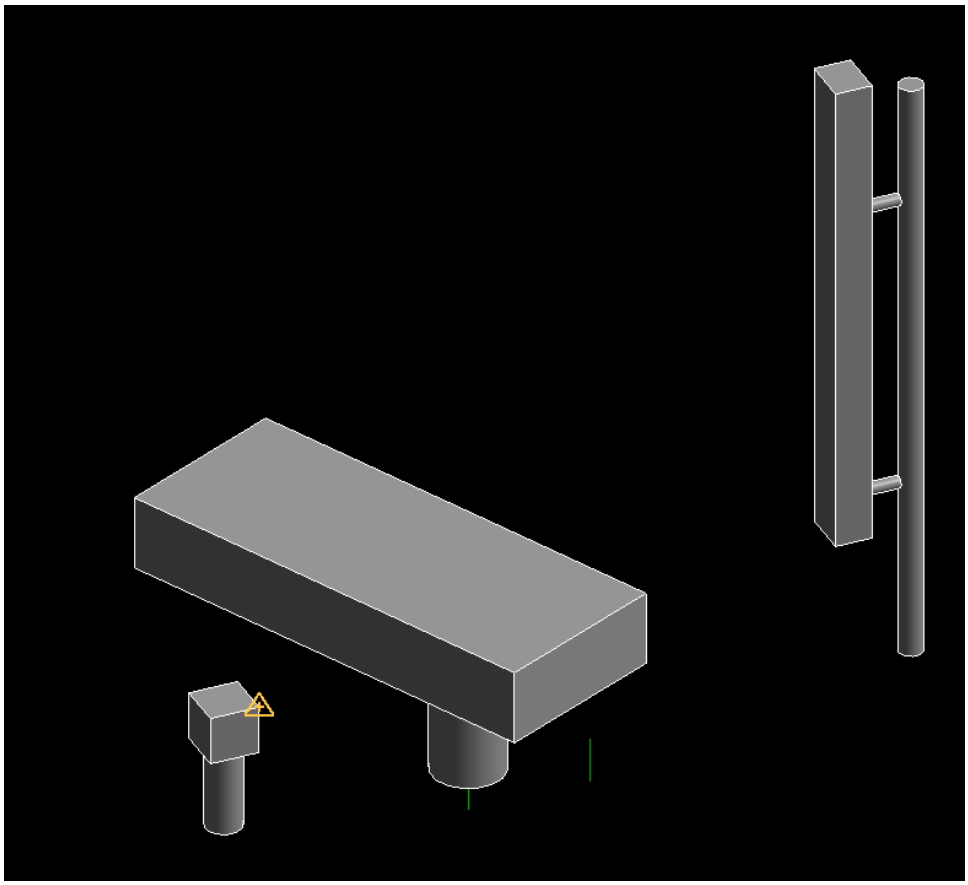


Figure 48 : État final du projet

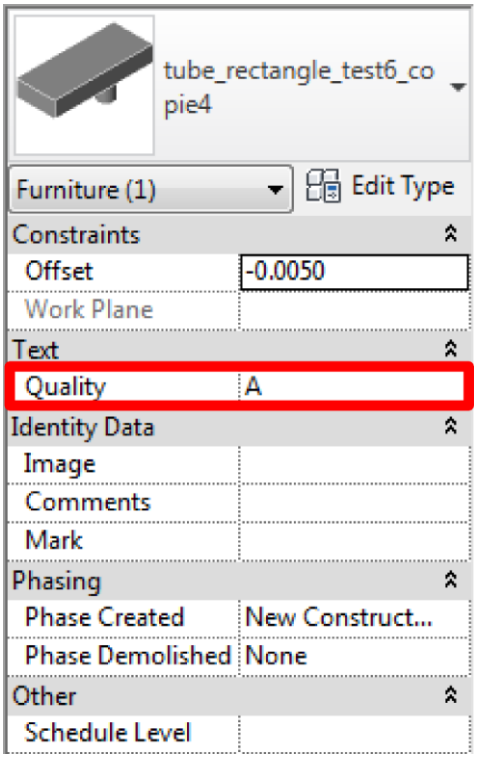


Figure 49 : Qualité renseignée dans le panneau de propriétés

Conclusion

Le plug-in crée PlaceAndCheck permettra à Axione de gagner du temps et donc de la productivité. Sa simplicité d'utilisation permet à tout utilisateur de manipuler des modèles 3D de manière précise. Le manuel de l'utilisateur fourni avec l'application détaille chaque étape et prodigue des conseils pour résoudre les éventuels problèmes qui peuvent survenir. Néanmoins, de nombreuses améliorations sont à prévoir. L'accrochage des points au clic pour réduire le risque d'erreurs et améliorer la précision est probablement la plus importante. Réduire le nombre de clics nécessaires à l'utilisateur pourra également réduire le risque d'erreur de manipulation.

Même avec le recul et l'expérience de la création de PlaceAndCheck, le plug-in entièrement automatisé envisagé paraît extrêmement difficile à coder. En effet Revit est extrêmement rigide aussi bien concernant les informations relatives aux géométries des objets que de manière générale. La création d'un tel outil sera impossible sur Revit tant que le logiciel imposera autant de contraintes à son utilisateur. Á l'heure actuelle, la seule solution envisageable est la création d'un logiciel consacré aux traitements de PlaceAndCheck, ce qui implique une interface graphique, des options de modélisation etc… Cela représente un travail énorme extrêmement chronophage.

En revanche, nous pouvons imaginer des fonctionnalités totalement nouvelles telles que l'export vers une base de données des objets et sites avec des alertes pour les maintenances des équipements devant être faîte dans le mois. Stocker l'information de la personne ayant effectué la pose d'équipement et celle ayant mise à jour le projet pourrait également être un atout pour la gestion du patrimoine. L'ajout de zone de sécurité, c'est-à-dire une marge autour de l'objet qui, lorsqu'un autre objet intersecte cette zone, génère un rapport d'alerte, est actuellement impossible dans Revit. Cependant la demande pour une telle fonctionnalité au sein de la communauté étant forte, notamment pour des besoins dans le domaine de la plomberie et du chauffage, Autodesk pourrait rajouter cet outil dans des versions futures. Il serait très intéressant de s'en servir afin de fournir toujours plus d'informations et d'avertissements à l'utilisateur.

Le plug-in PlaceAndCheck est très polyvalent puisqu'il peut être adapté à de nombreux usages autres que celui que peut en faire Axione. Un électricien venant réaliser un câblage dans un immeuble disposant d'un BIM pourra, à l'aide de RAPH et de PlaceAndCheck, rapidement mettre à jour ce modèle sans que des opérations terrain réalisées par des professionnels de la mesure tels que des géomètres ne soient nécessaires. Cela permet d'accélérer les échanges et les mises à jour dans le domaine de la construction, rend les professionnels de ce domaine plus autonome et réduit donc le nombre d'interventions des différents intervenants.

Avec l'avènement annoncé de l'ère BIM à travers le monde, le plug-in PlaceAndCheck, sous réserve de mises à jour, pourra trouver sa place au sein de nombreux processus de production, qu'ils soient des créations de BIM de bâtiment existant ou la maintenance de ces dits modèles.

Bibliographie

AUTODESK. Autodesk Community, en ligne. Disponible sur : http://forums.autodesk.com/t5/revit-api/. (consulté le 27/05/15)

AUTODESK. Easily find, preview and download high quality BIM models & DWG files, en ligne. Disponible sur http://seek.autodesk.com/. (consulté le 27/05/15)

JEREMY TAMMIK. The Building Coder, Blogging about the Revit API, en ligne. Disponible sur <http://thebuildingcoder.typepad.com/>. (consulté le 27/05/15)

AUTODESK. My First Plug-in Training, en ligne. Disponible sur http://usa.autodesk.com/adsk/servlet/index?siteID=123112&id=16777469. (consulté le 03/02/15)

LYNDA, Managing Location Coordinates with Revit, en ligne. Disponible sur http://tutgfx.com/lynda-managing-location-coordinates-with-revit/. (consulté le 07/03/15)

STACKOVERFLOW, Stack Overflow, en ligne. Disponible sur : http://stackoverflow.com/ . (consulté le 27/05/15)

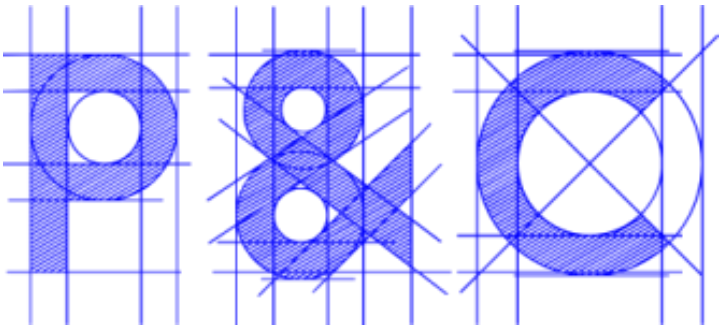
SCIPY.ORG, scipy.optimize.curve_fit, en ligne. Disponible sur https://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.optimize.curve_fit.html#scipy.optimize.curve_fit. (consulté le 27/05/15)

Table des annexes[4]

[4] Les annexes doivent être annoncées dans le texte principal en note de bas de page. On évitera alors de renvoyer à la page où se situe l'annexe mais on renverra plutôt au n° de l'annexe. On peut ici détailler ou illustrer des informations qui n'ont pas pu être développées dans le texte mais qui méritent de l'être. Les annexes sont numérotées et titrées. On évitera donc de faire figurer plusieurs annexes sur une même page. Pour enlever cette note de bas de page, supprimer l'appel de note ci-dessus.

Annexe 1
Manuel de l'utilisateur

Manuel de l'utilisateur



Plugin Revit PlaceAndCheck
v 1.0

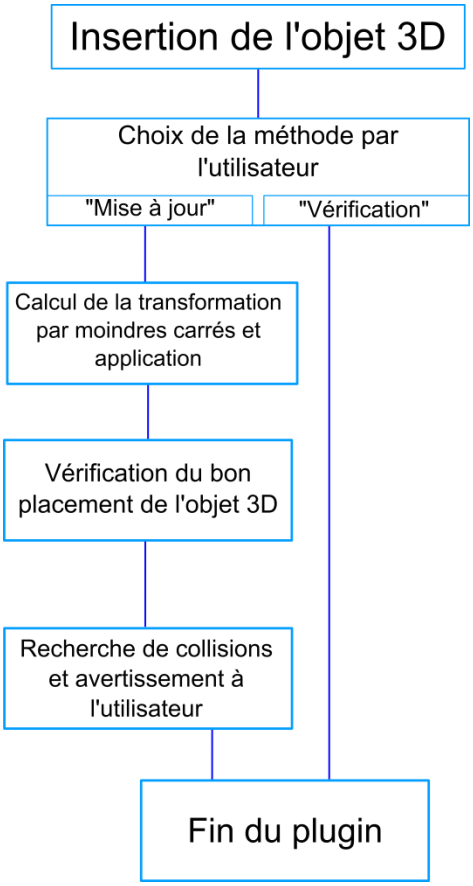
Compatible Windows 7 x64 et Revit 2015.

Juillet 2015 Jordi Attencia

SOMMAIRE

Introduction

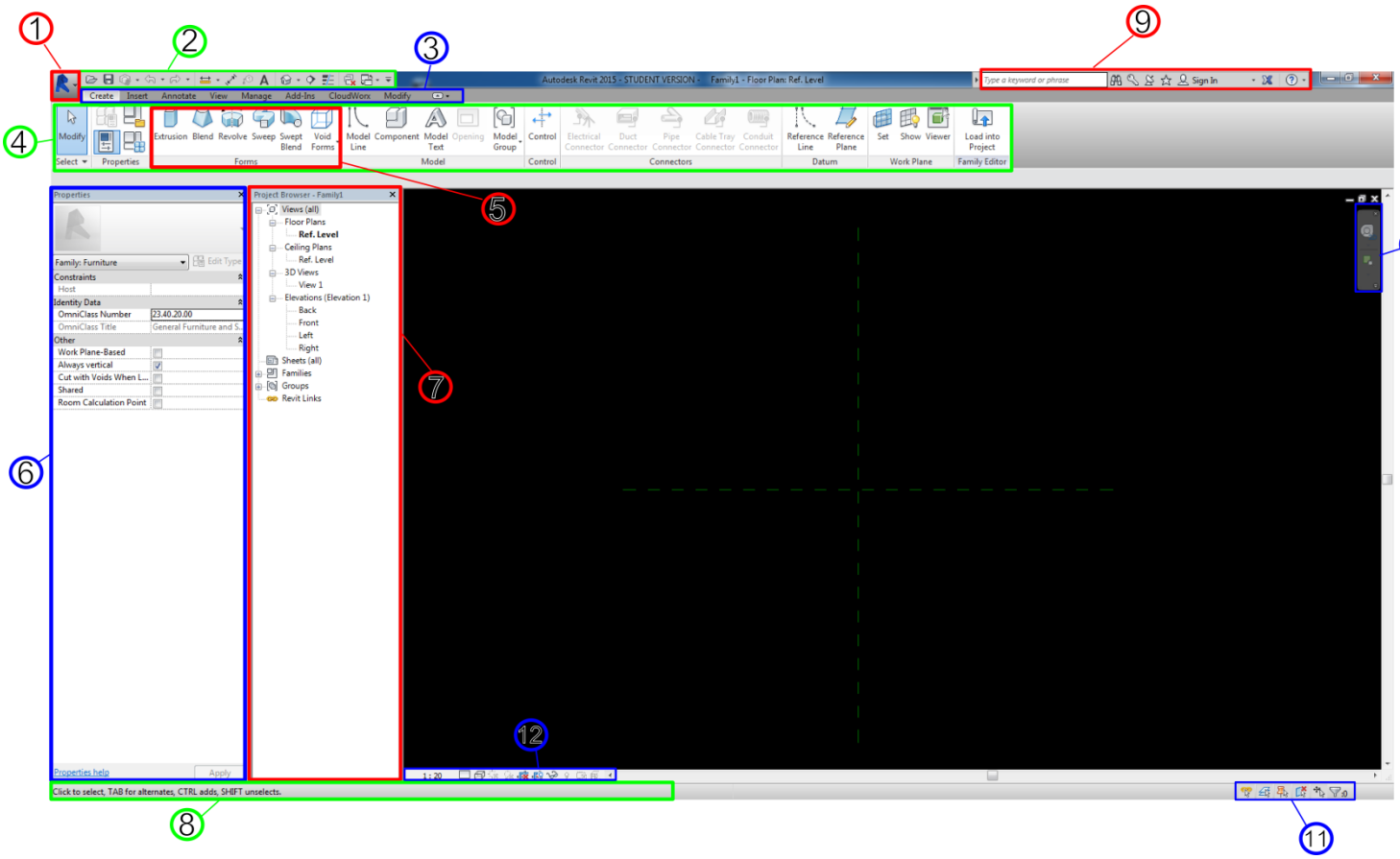
Le plugin P&C a pour but de faciliter l'insertion d'objets 3D de manière précise dans un plan 3D afin d'assurer la connaissance exhaustive des équipements présent sur différentes zones de travaux. Ce plugin est développé par Géomat et est utilisable via le logiciel Revit 2015. Son utilisation va de pair avec le système RAPH développé par FITESIC. P&C accompagne l'utilisateur tout au long du processus afin de rendre ce dernier simple, accessible et efficace. L'execution du plugin prend environ 3 minutes. Tout au long de l'exécution du plugin, des informations et instructions seront fournie à l'utilisateur via des fenêtres au centre de l'écran ou dans la barre d'indication contextuelle en bas à gauche de l'écran. Il est impératif de suivre ces instructions pour assurer le bon déroulement du programme.
P&C suit le schéma suivant :



P&C est un travail continu, n'hésitez pas à communiquer toutes suggestions à GEOMAT pour d'éventuelles améliorations du plugin.

I Interface Revit

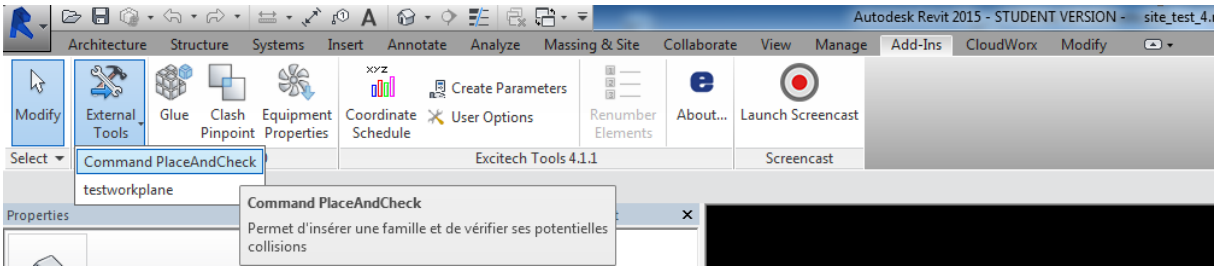
L'interface de Revit est basée sur les logiciels AutoDesk :



1 : Menu principal
2 : Barre d'accès rapide
3 : 1er ruban
4 : 2ème ruban
5 : 3ème ruban
6 : Panneau des propriétés
7 : Panneau des vues
8 : Barre d'indication contextuelle
9 : Barre d'aide et licences
10 : Raccourcis de navigation
11 : Raccourcis de sélection
12 : Raccourcis de visualisation

II Lancement du plugin

Le plugin est lancé depuis le menu « Add-Ins » du premier ruban, en sélectionnant « External Tools » puis « Command PlaceAndCheck » :



Cela doit impérativement fait depuis un document ouvert sous peine d'arrêt prématuré du plugin.

III Procédure pré-intervention

Les étapes à suivre sont :

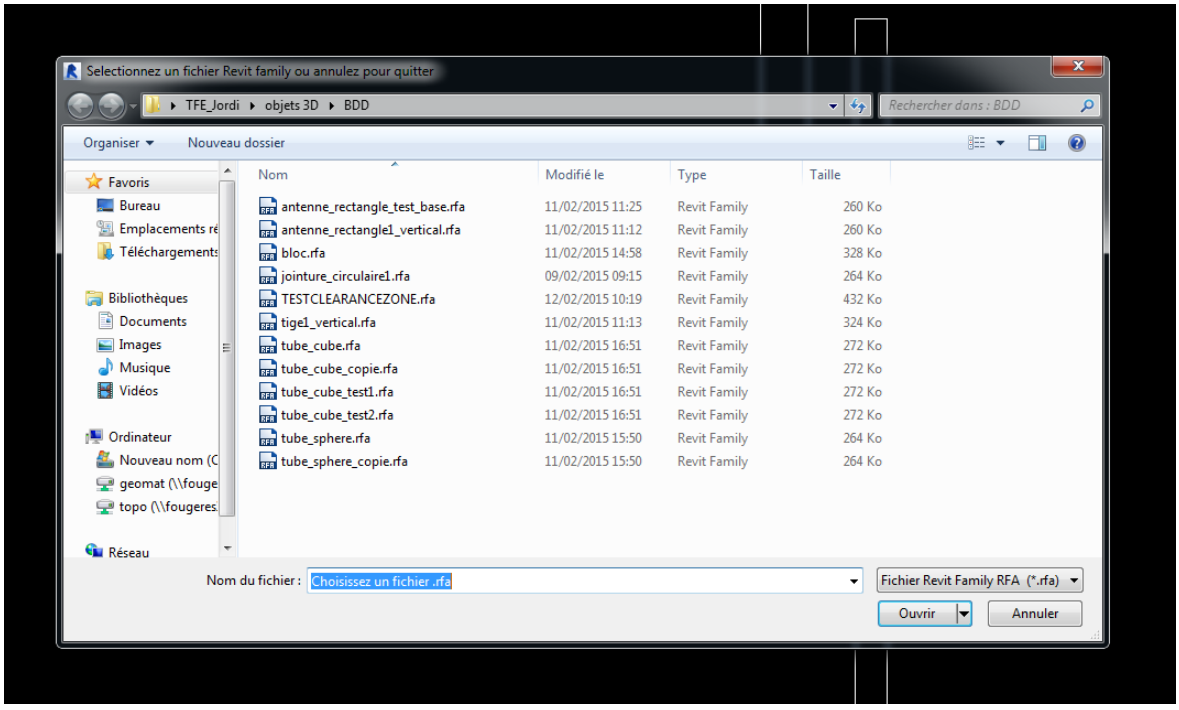
-Choix de l'objet/famille à insérer

-Choix de la méthode d'insertion

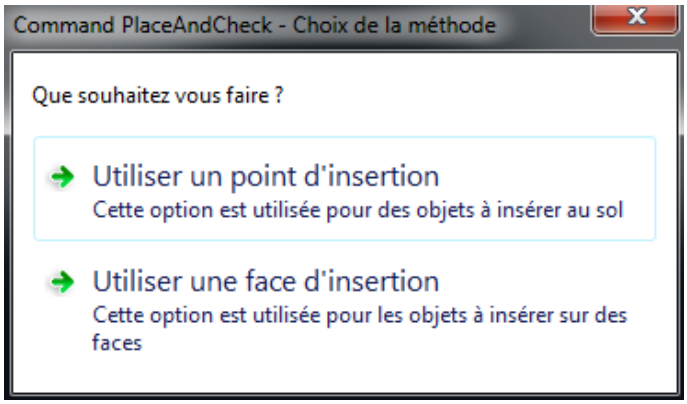
-Choix de la procédure de vérification

Insertion de l'objet :

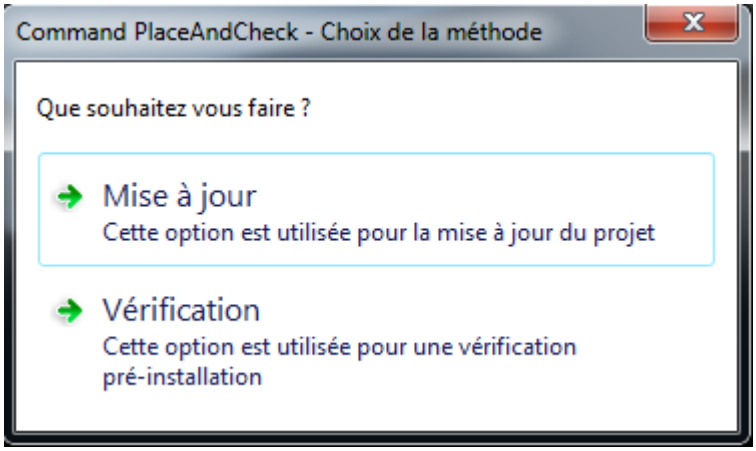
Dès le lancement de P&C, une fenêtre apparait :



Cette fenêtre permet de choisir l'objet à insérer. Une fois celui-ci choisi, cliquez sur ouvrir, ou double cliquez simplement sur le fichier. Choisissez ensuite la méthode d'insertion à l'aide de la fenêtre suivante :



Un simple clic suffit ensuite à insérer l'objet dans le dessin. Une nouvelle fenêtre d'option apparait alors :

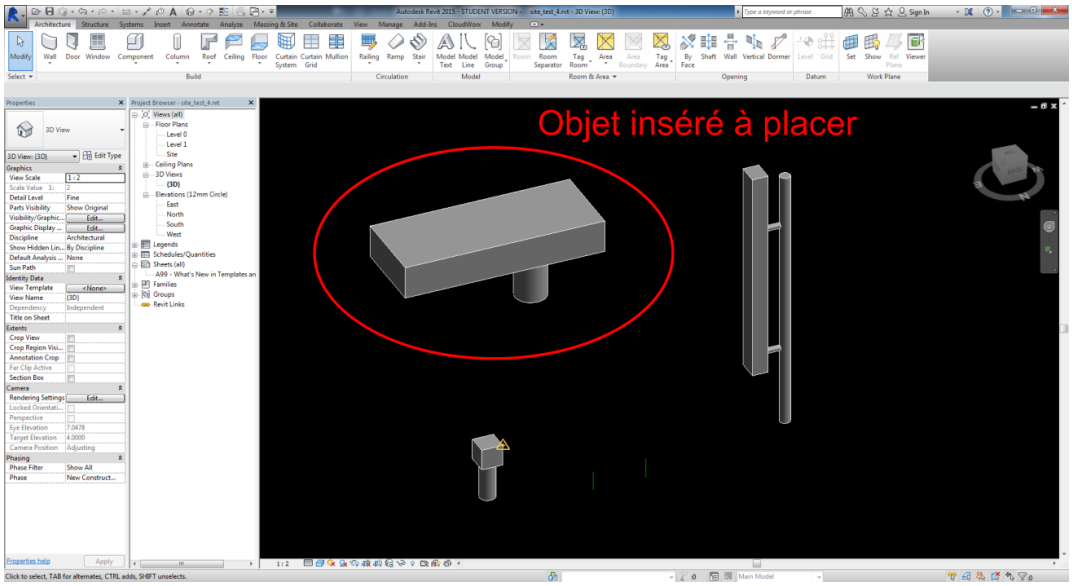


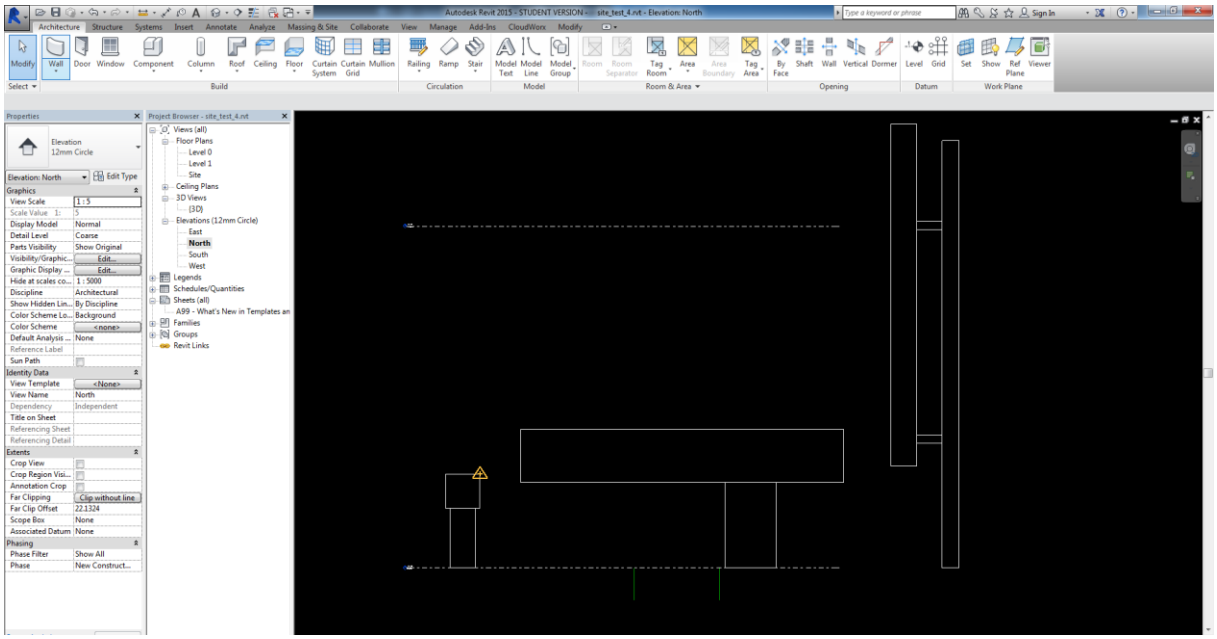
Dans le cadre d'une procédure pré-intervention on choisit l'option « Vérification ».

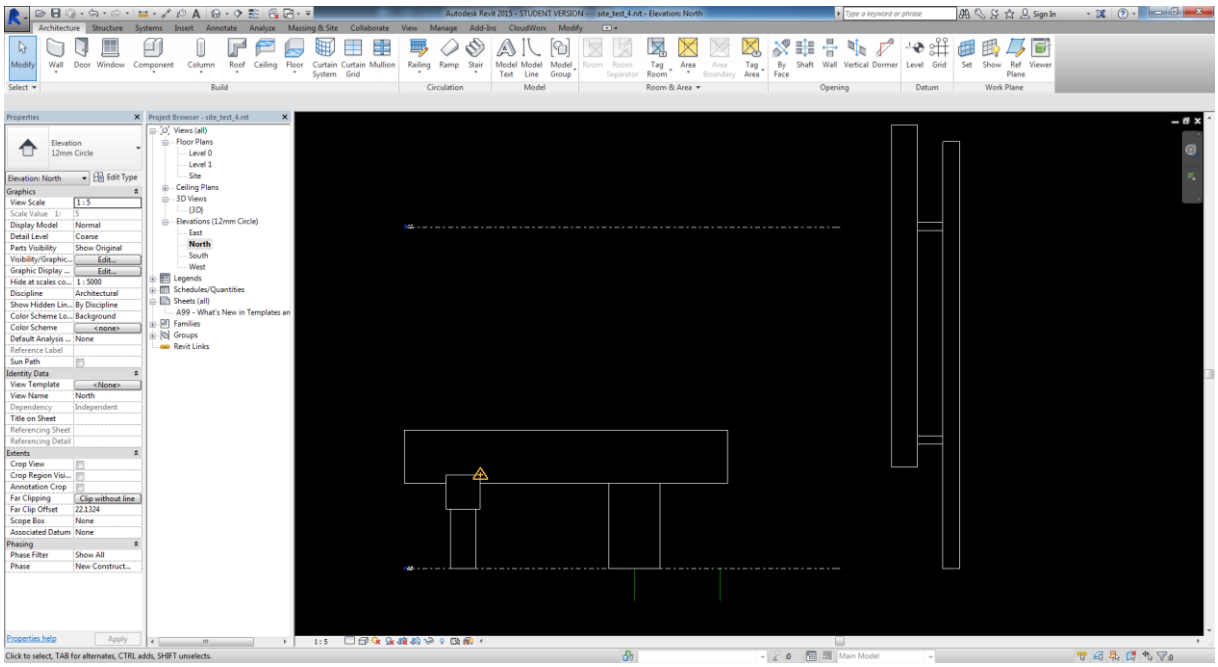
C'est alors l'arrêt du plugin. Pour placer l'objet plus précisément, il faut procéder à des opérations supplémentaires :

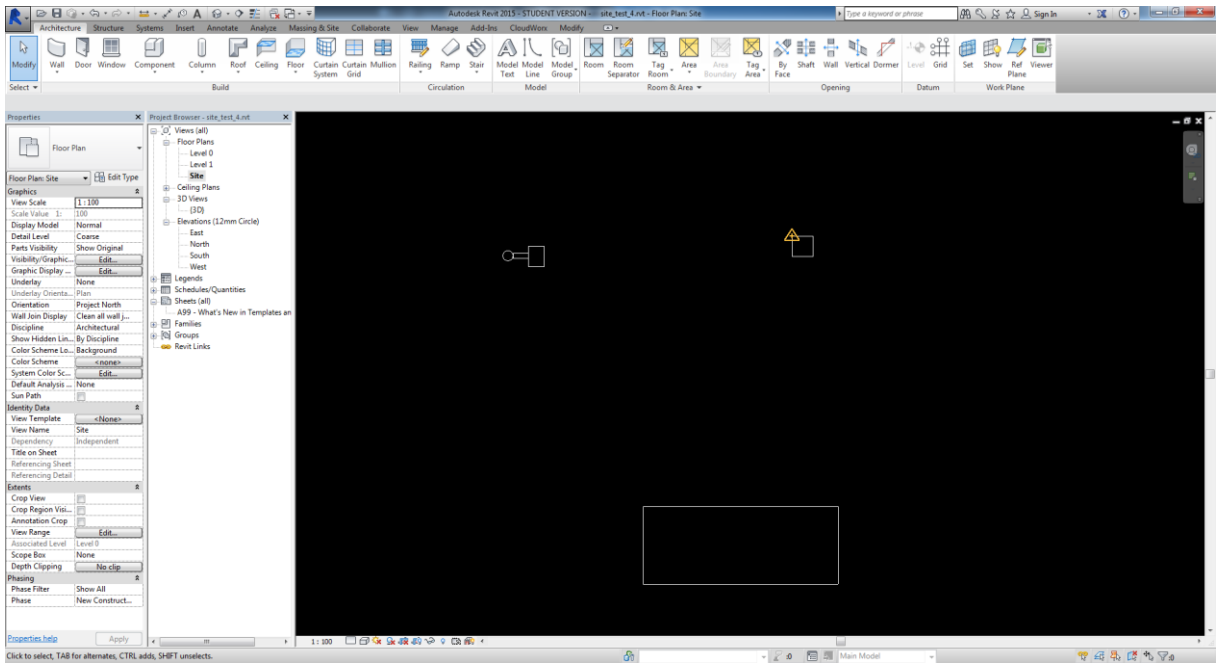
Déplacer l'objet :

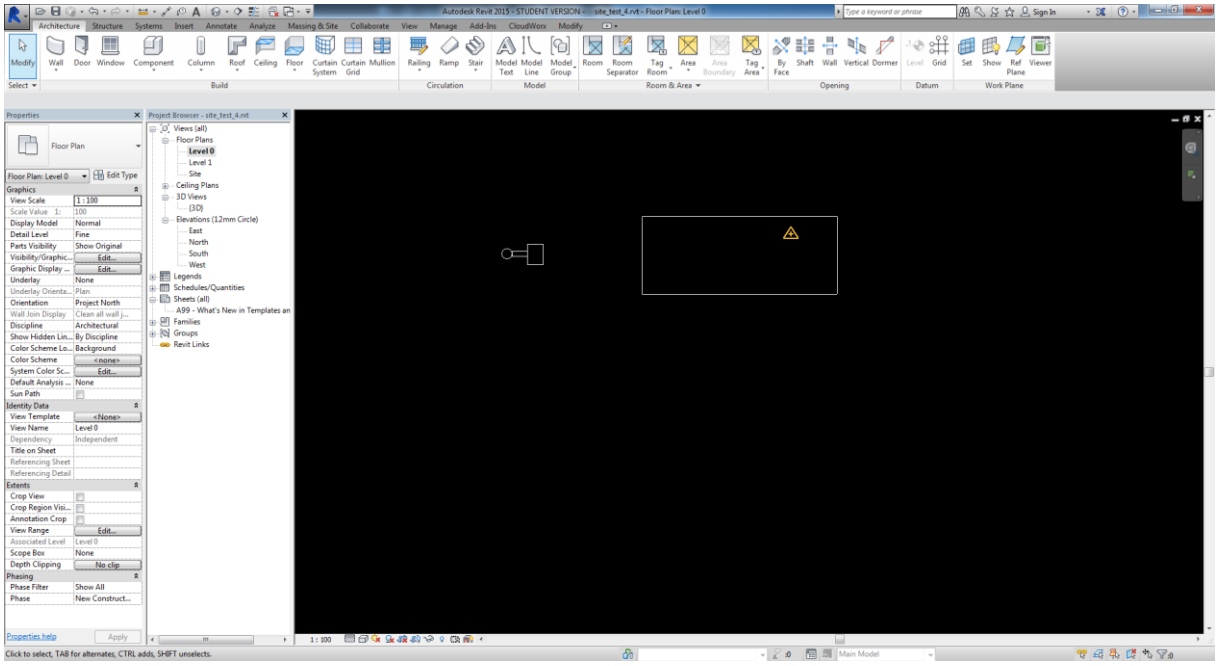
En sélectionnant l'objet insérer, on peut le déplacer en faisant un « cliquer-glisser ». Pour le placer plus précisément on peut changer plusieurs fois de vue en double cliquant sur ces dernières dans le panneau des vues (cf interface de base Revit).

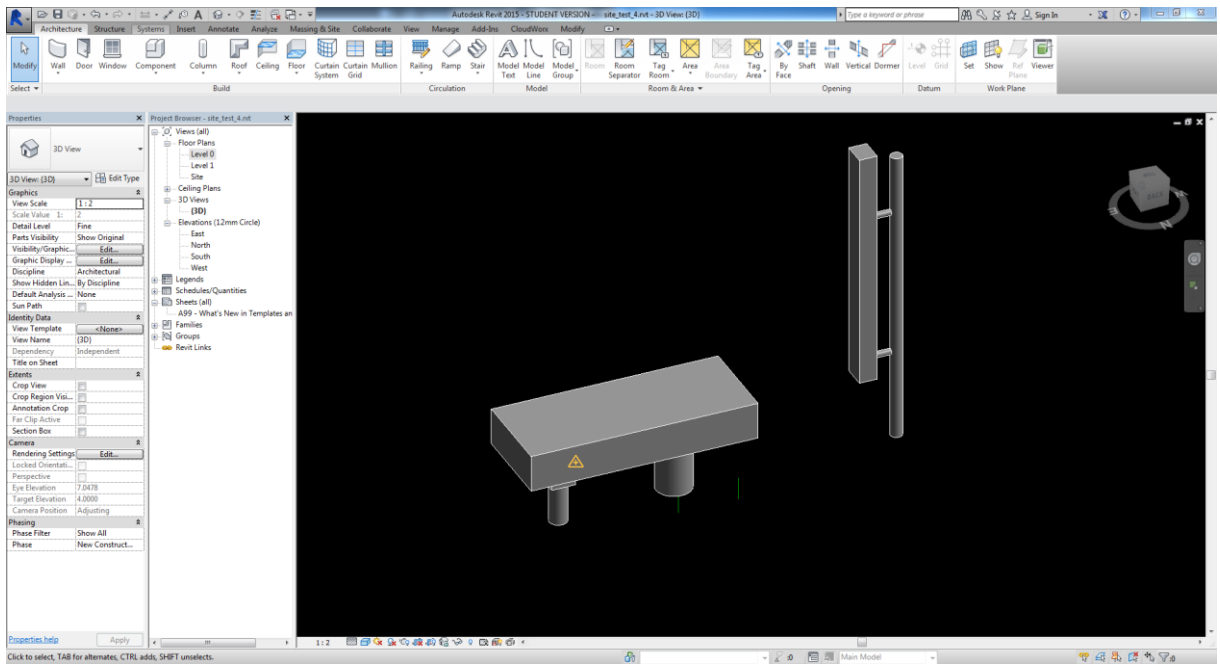

Objet inséré à placer





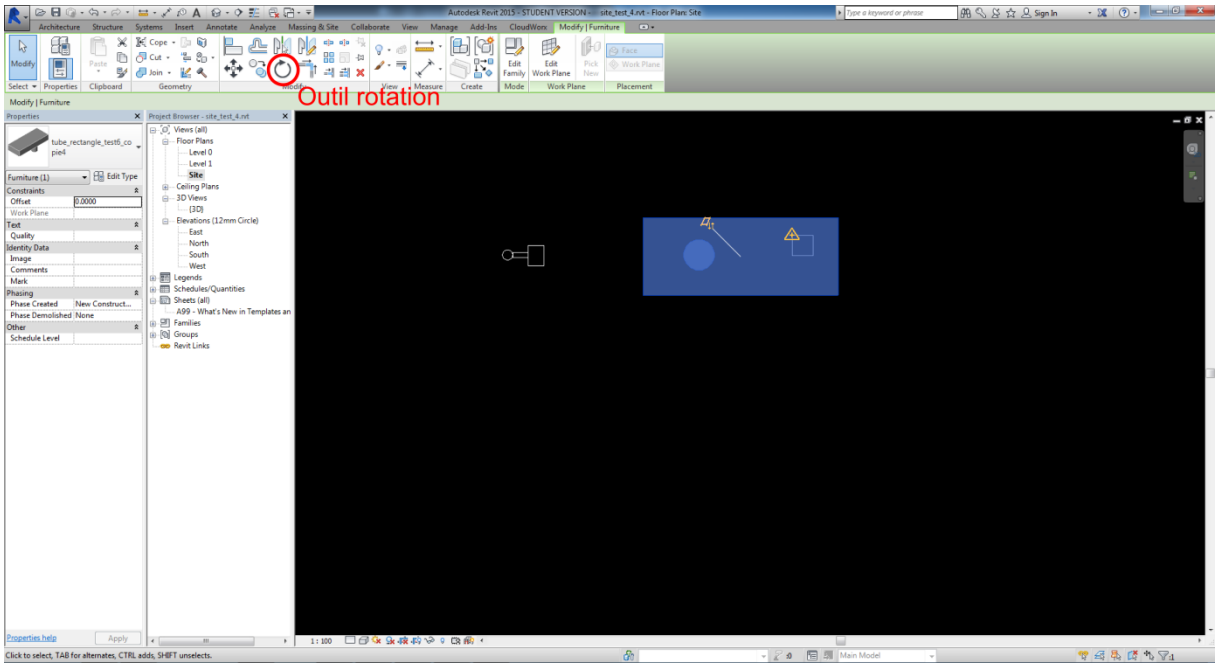




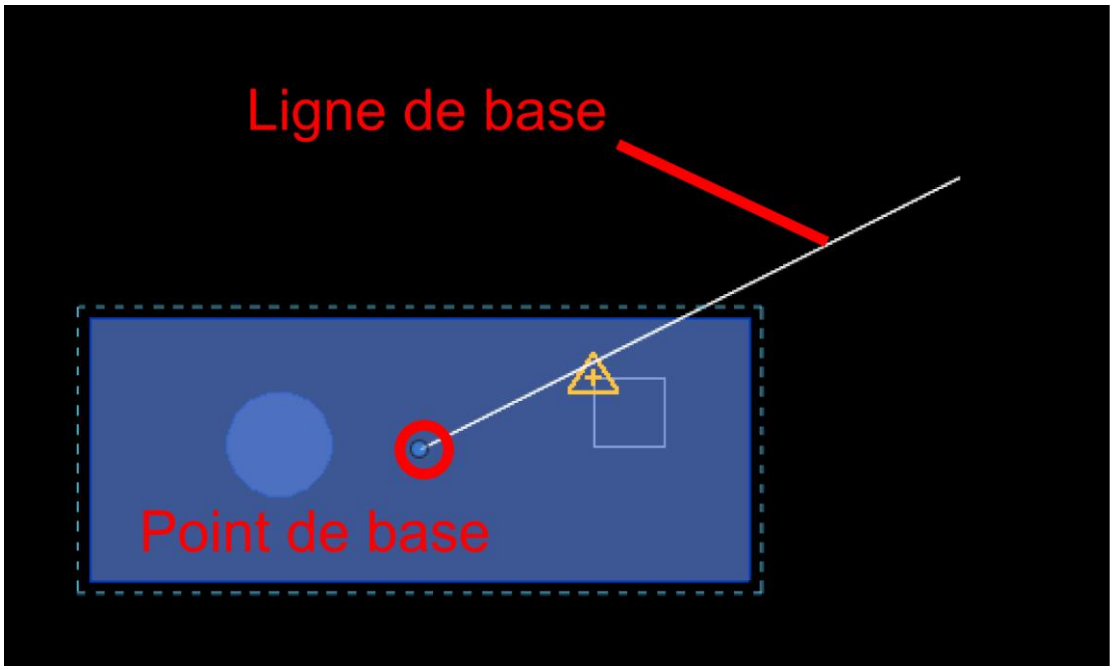


Faire tourner l'objet :

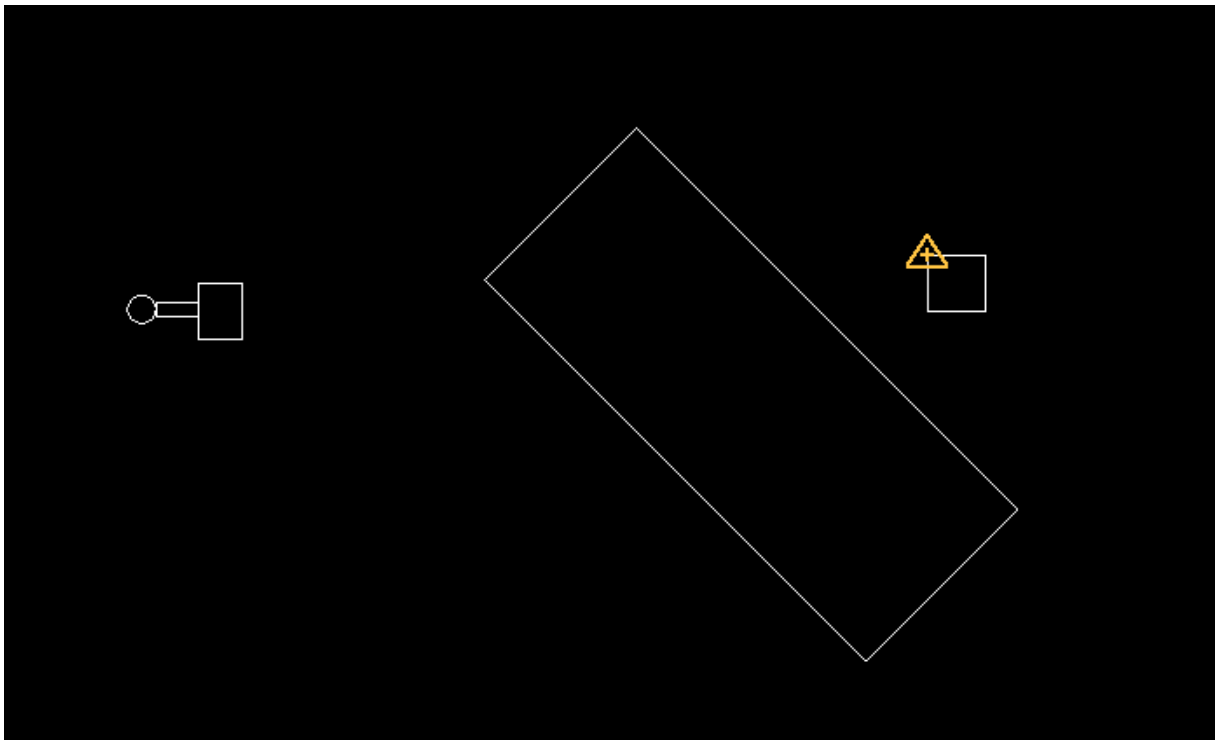
On peut ensuite appliquer des rotations à l'objet en cliquant sur l'objet puis en sélectionnant l'outil rotation dans le 2ème ruban.

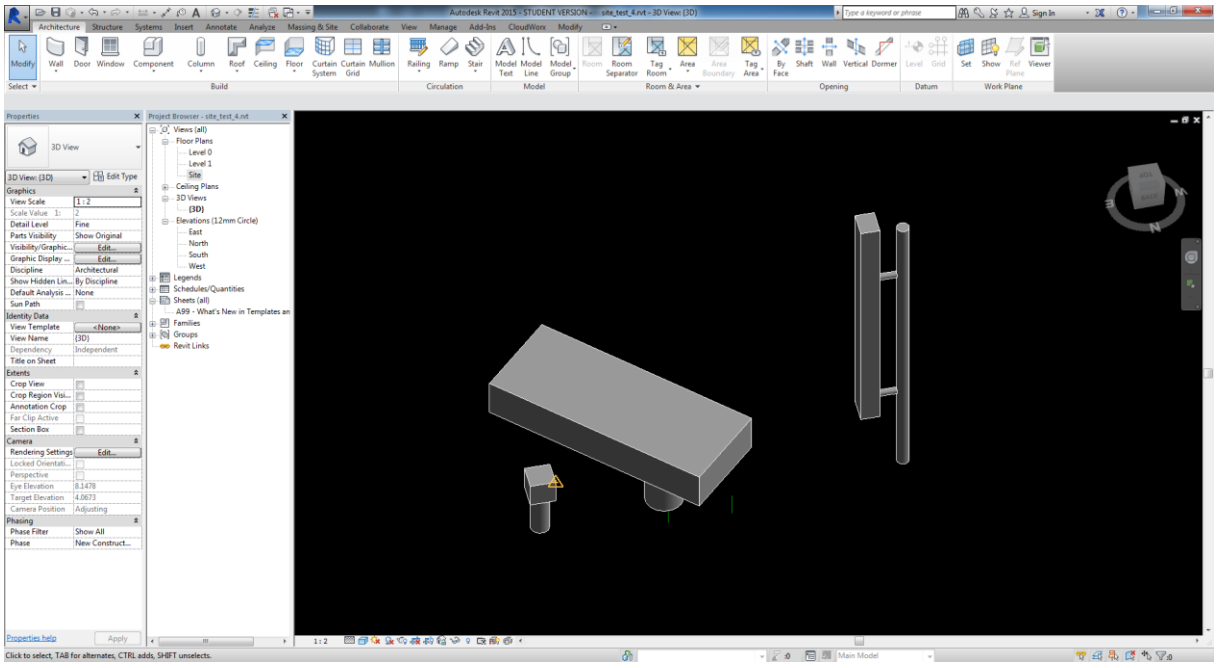


Apparait alors un point de base et une ligne de base pour la rotation :



Déplacez le point de base de la même manière que l'on déplace un objet, puis cliquez à l'endroit où vous souhaitez fixer la ligne de base. Déplacez ensuite la ligne de base pour observer une rotation de l'objet. Cliquez une fois la position voulue atteinte.

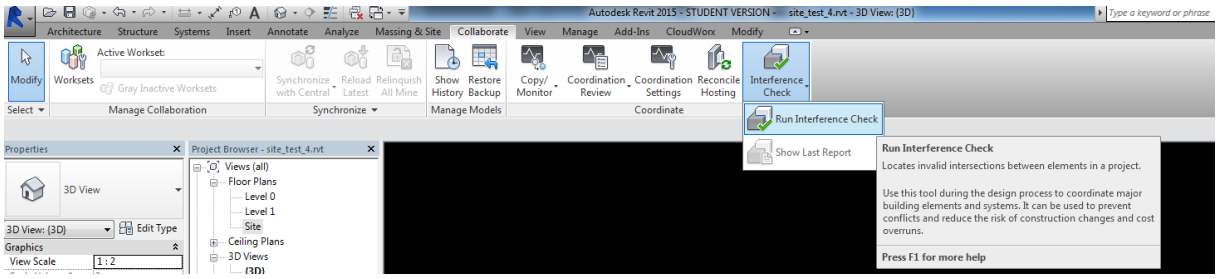




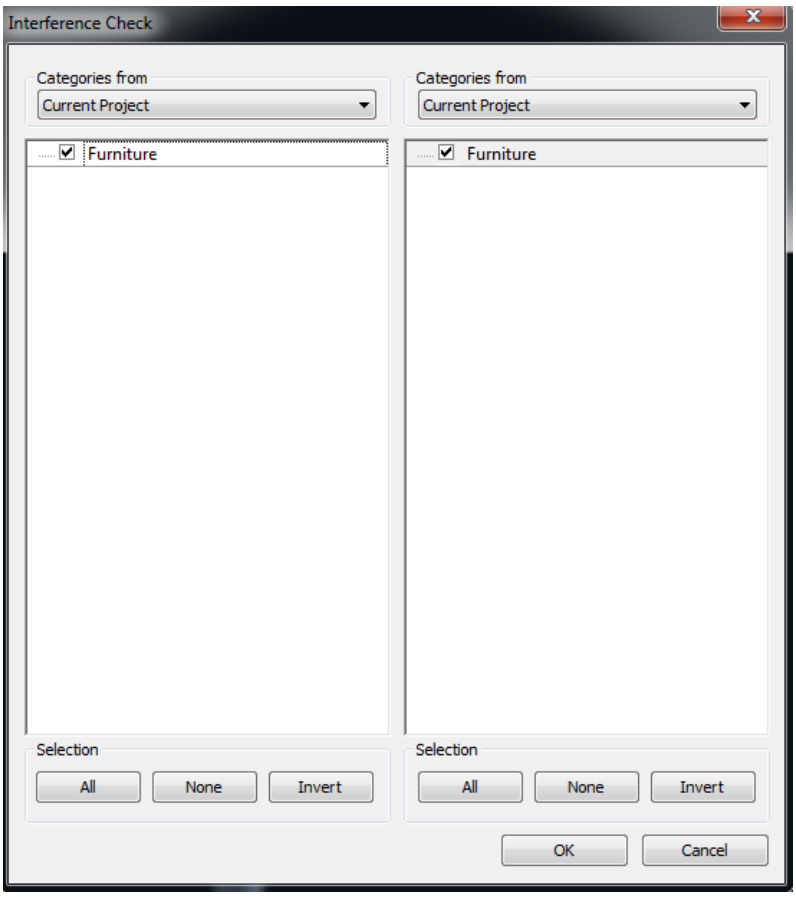
Si besoin on peut à nouveau appliquer rotations et déplacement pour placer l'objet précisément avant de procéder au test de collision.

Test de collision :

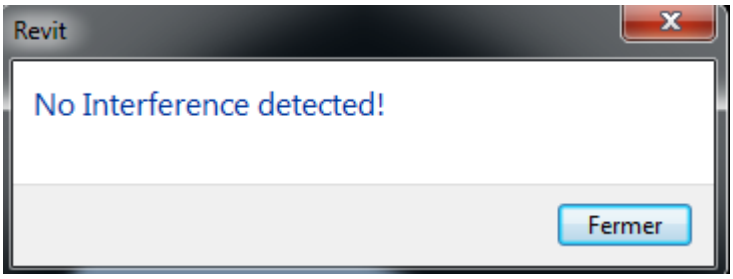
Pour lancer ce test il faut cliquer sur les menus suivant dans les rubans : « Collaborate », « Interference Check », « Run Interference Check ».



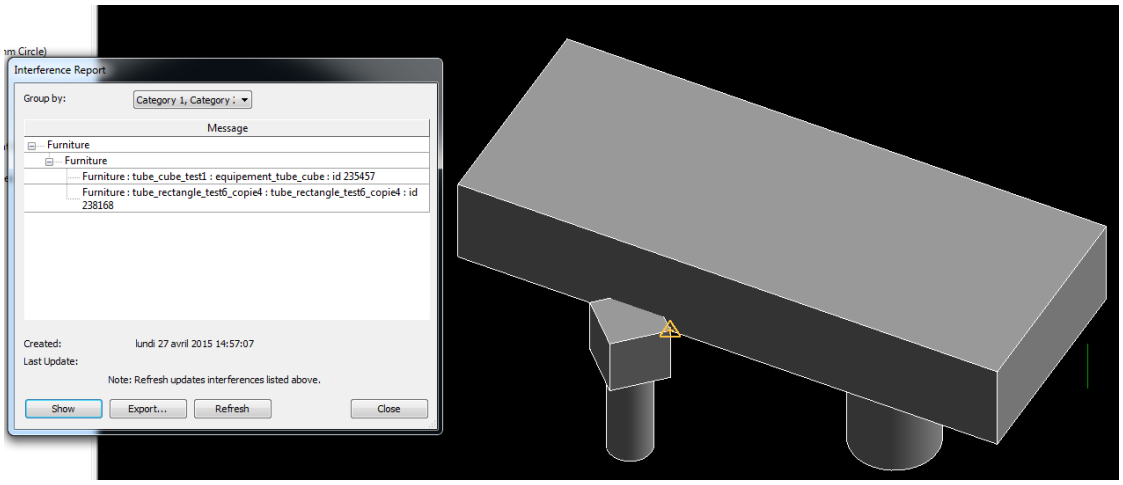
Il faut ensuite cocher toutes les cases et cliquer sur « OK »:



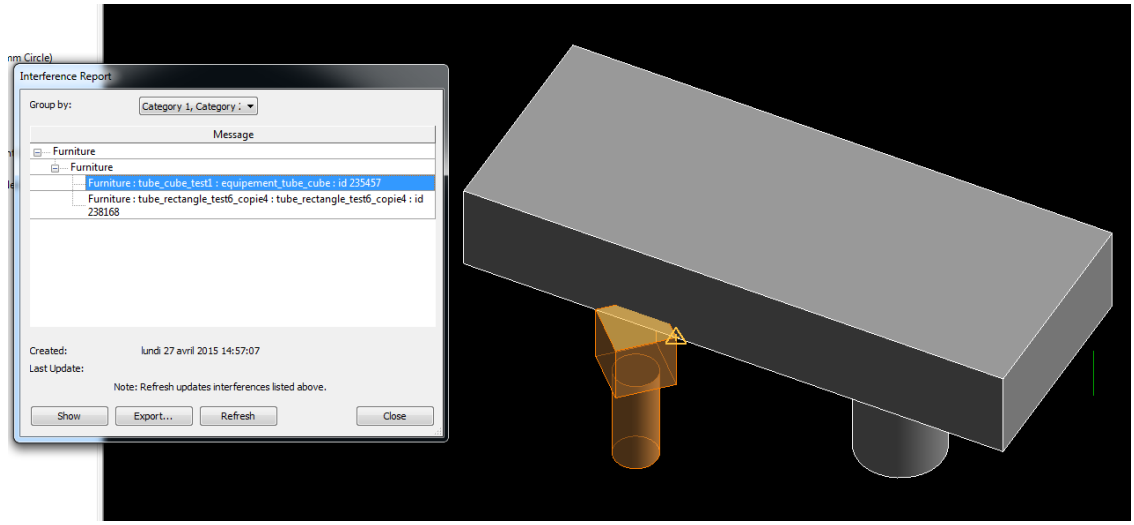
Dans le cas où aucune collision n'est détectée la fenêtre suivant apparaitra :

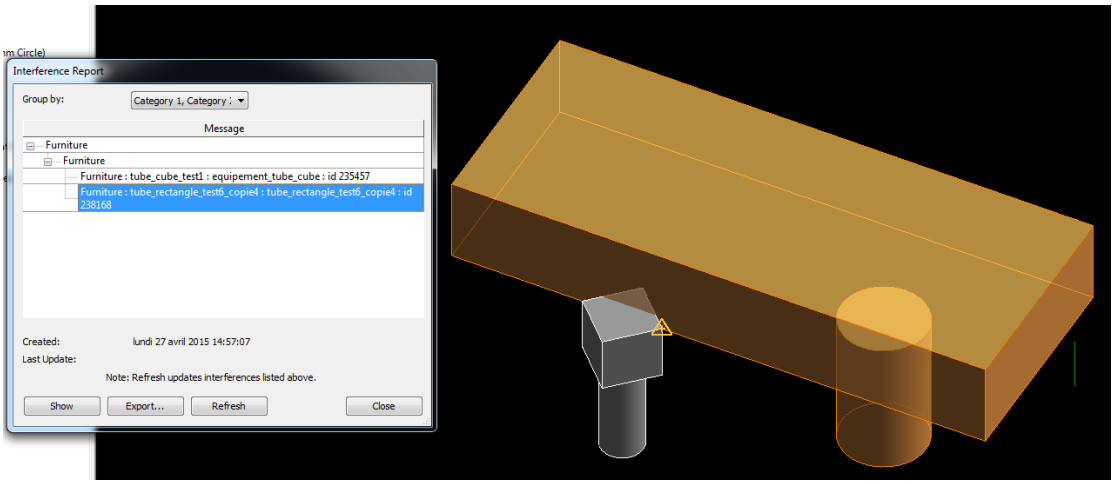


Dans le cas où des collisions sont détectées, le rapport suivant est affiché à l'écran :



En cliquant sur les éléments du rapport, on va mettre en surbrillance les objets concernés par les collisions :





Si l'Interference Check n'affiche aucune collision, on peut toujours prendre des cotes dans les vues 2D à l'aide du menu « Annotate » puis « Aligned » afin de se donner une marge de sécurité.

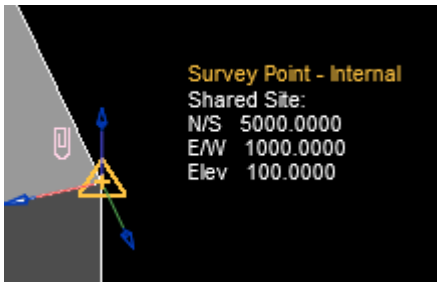
Ceci n'étant qu'une opération de vérification pré-intervention, il ne faut pas sauvegarder le travail afin que le plan 3D continue de représenter la réalité.

IV Procédure post-intervention

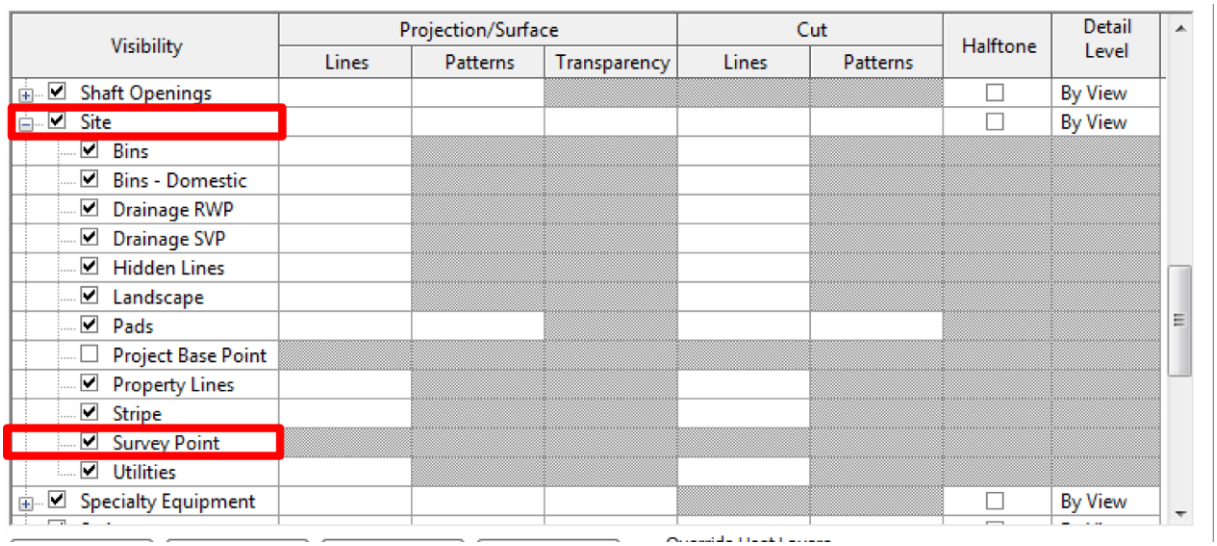
Il s'agit ici de placer dans le plan 3D l'objet installé sur le site et ce de manière très précise.

Opérations préliminaires :

Il faut ouvrir le projet correspondant au site et récupérer les coordonnées du « Survey point » :



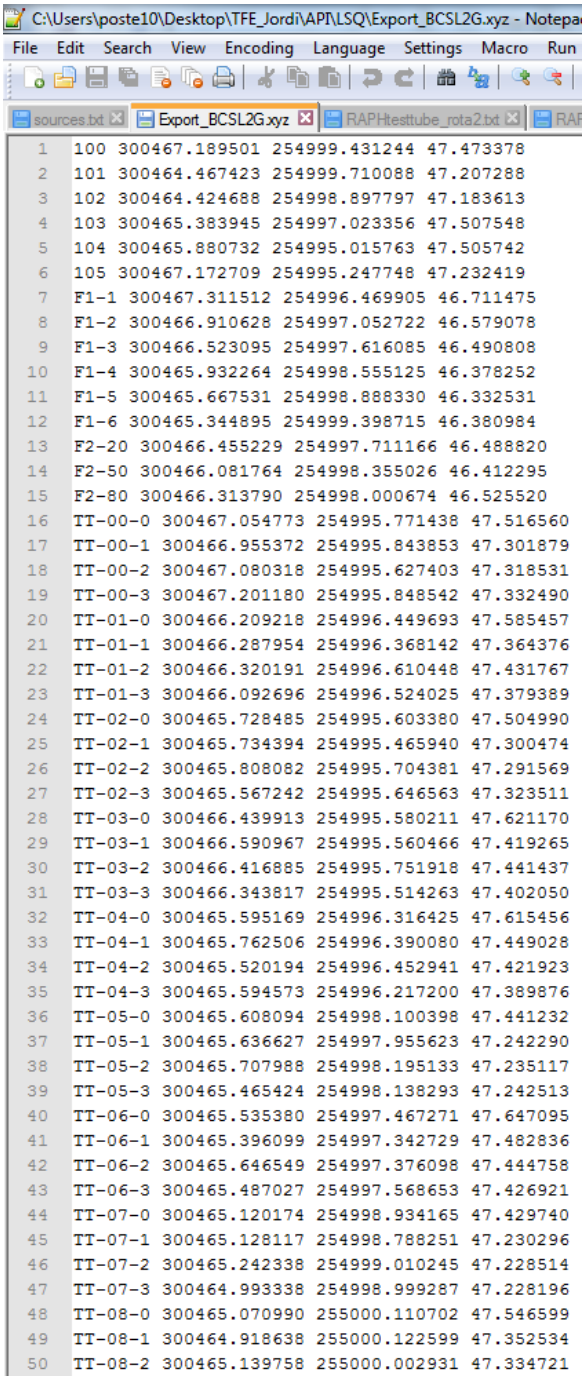
Si ce point n'apparait pas dans le dessin il faut taper au clavier « vg » ce qui va ouvrir la fenêtre d'option « Visibility/Graphic ». Déroulez la rubrique « site » de cette fenêtre et cochez la case « Survey Point ».



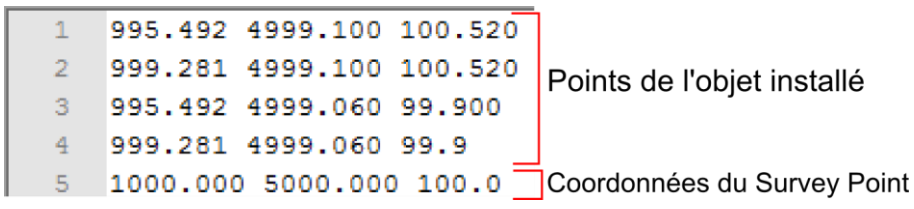
Cliquez sur OK et le Survey Point devrait alors s'afficher à l'écran. Notez les coordonnées de ce point. Elles seront à écrire dans un fichier texte.

NB : Revit étant un logiciel américain et encore peu utilisé par des francophones, les coordonnées sont en « Nord, Est, Altitude ». Le système Français étant en « Est, Nord, Altitude » veuillez les noter de cette façon.

Il faut maintenant traiter le fichier issu de RAPH. C'est un fichier d'extension .xyz et s'ouvre à l'aide du logiciel gratuit Notepad++. Le fichier s'apparente à cela :



On va seulement prendre les coordonnées des points appartenant à l'objet qui a été installé et placer ces coordonnées dans un autre fichier texte. Pour ce faire dans Notepad++, « fichier », « nouveau ». Coller les coordonnées dans ce fichier puis rajoutez à la fin les coordonnées du Survey Point. Le fichier devrait alors ressembler à ceci :



L'ordre des points est primordial puisqu'ils devront être cliqués dans le même ordre lors de l'exécution du plugin.

Enregistrez ce fichier puis fermez Notepad++. P&C est prêt à être lancé.

Insertion de l'objet :

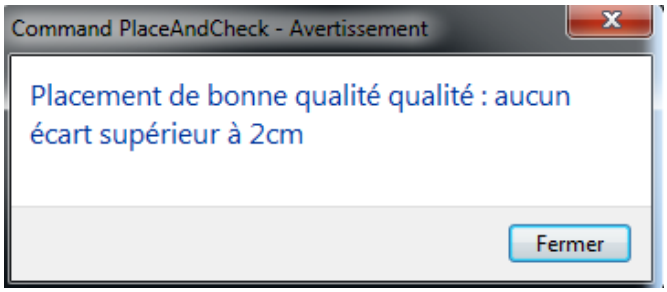
L'insertion s'effectue de la même manière que pour une procédure pré-installation. Cependant il faut cette fois sélectionner « Mise à jour » dans la Fenêtre post-insertion. Une nouvelle fenêtre de choix de fichier RAPH va apparaitre. Choisissez le fichier précédemment créé, cliquez sur « ouvrir » puis suivez les instructions en bas à gauche de l'écran.

Cliquez sur chaque point de l'objet correspondant aux points du fichier RAPH **en respectant l'ordre**. Les contours de la face de l'objet sur laquelle les points seront cliqués sont affichés en bleu. Veillez à cliquer sur la bonne face. Un petit temps de chargement suivra cette étape.

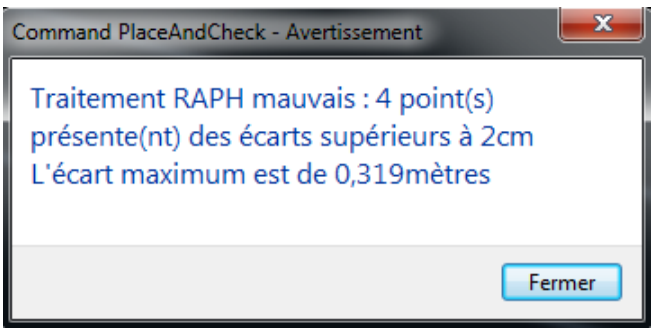
Cliquez ensuite à deux reprises sur l'objet à déplacer, il subira alors une série de rotations et translations. Il faut alors à nouveau cliquer sur les points dans le même ordre. L'objet ayant tourné, les faces concernées ne sont peut-être plus visibles. Pour pivoter autour de l'objet, dézoomez légèrement puis enfoncez la touche Shift/Maj et pivotez à l'aide du clic droit. Une fois les points cliqués, l'objet sera à nouveau déplacé.

Afin de vérifier le bon placement de l'objet, il faut à nouveau cliquer sur les points dans le même ordre, mais cette fois de façon précise. Pour cela zoomez sur les points à sélectionner afin d'améliorer la précision du pointé.

Un rapport de qualité va apparaitre à l'écran. Si la qualité est bonne, le rapport suivant s'affiche :



Dans le cas d'une qualité moindre, le rapport suivant s'affiche :

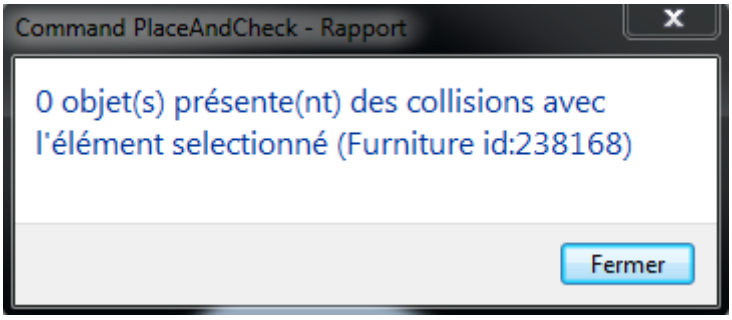


Une qualité moindre signifie que le traitement RAPH est de mauvaise qualité. Cela peut être dû au pointé lors du traitement RAPH. Si les écarts restent inférieurs à 3 centimètres cela reste acceptable.

Un paramètre « Quality » de l'objet est renseigné automatiquement.

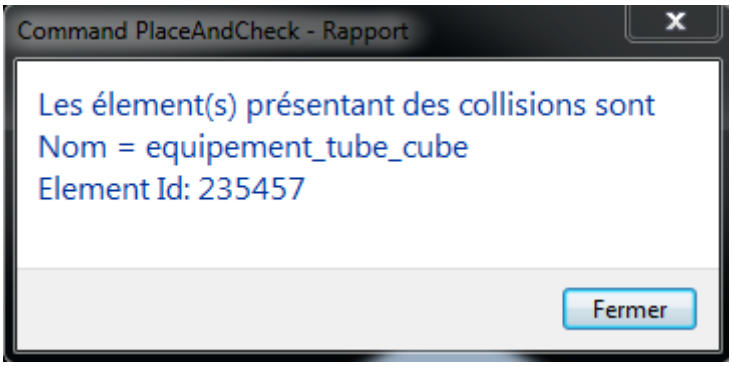
Quality	Écart
A	écart<2cm
B	2cm<écart<2.5cm
C	2.5cm<écart<3cm
D	écart >3cm

Une fois le rapport de qualité affiché, cliquez sur l'objet traité pour une recherche de collisions. Si aucune collision n'est détectée, le message suivant s'affiche :

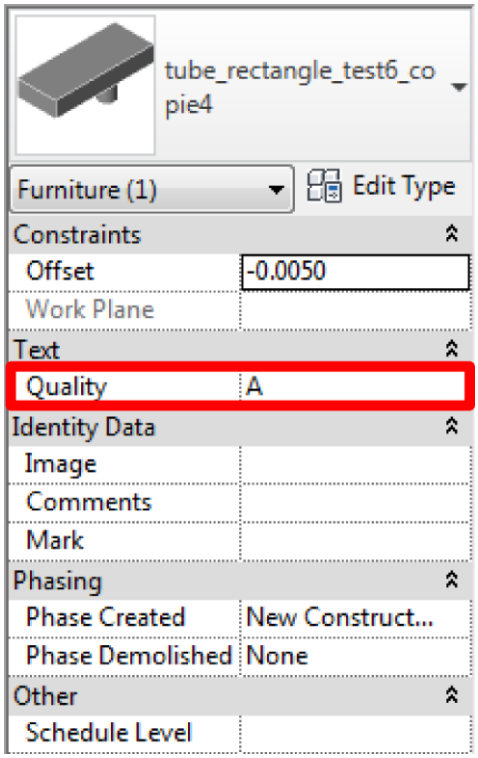


Dans ce cas, aucun traitement supplémentaire n'est nécessaire. Pensez a enregistrez votre travail pour sauvegarder la mise à jour.

Dans le cas de collisions détectées, la fenêtre suivante s'affiche :



Un champ qualité est rempli dans les propriétés de l'objet. Il est visible à gauche de l'écran simplement en cliquant sur l'objet.



V Procédures résumées

La procédure pré-intervention :

-Sélection de l'objet à insérer
-Insertion par clic
-Choix : « Vérification »
-Recalage manuel à l'aide de « déplacer glisser » et de l'outil rotation
-Test de collision: « Collaborate », « Interference Check », « Run Interference Check »
-Prises de cotes éventuelles
-Ne pas sauvegarder

La procédure post-intervention :

-Création du fichier RAPH
-Sélection de l'objet à insérer
-Insertion par clic
-Choix : « Mise à jour »
-Sélection du fichier RAPH
-Clic des points dans l'ordre
-Chargement de 4 secondes
-Clic 2 fois sur l'objet
-Clic des points dans l'ordre
-Clic des points dans l'ordre de façon précise
-Clic sur l'objet
-Sauvegarde du travail

VI Résolution de problèmes

Il peut arriver que la procédure post-intervention fournisse des résultats qui ne concorde pas avec les attentes (placement aberrant, collisions, qualité aberrante…). Voici les différentes vérifications à effectuer :

-Êtes-vous dans le bon projet ?

-Le fichier RAPH est-il bien construit ?

Les points sont-ils ceux de l'objet à insérer ? Les points sont-ils dans le bon ordre ? Les coordonnées du Survey Point sont-elles bien à la dernière ligne ? Ces coordonnées sont-elles bien formatées « Est, Nord, Altitude » ?

-Le fichier de l'objet sélectionné ou le fichier RAPH sélectionné sont-ils les bons ?

Si tout cela ne met pas en évidence de problèmes il faut alors recommencer la procédure en prenant bien soin de suivre les instructions et en cliquant un peu plus précisément.

Parmi les erreurs fréquentes, notamment sur des objets présentant des symétries : les points cliqués lors du deuxième calage sont du mauvais côté de l'objet. Pour pivoter, dézommez puis maintenez enfoncé la touche Shift/Maj et pivotez à l'aide du clic droit.

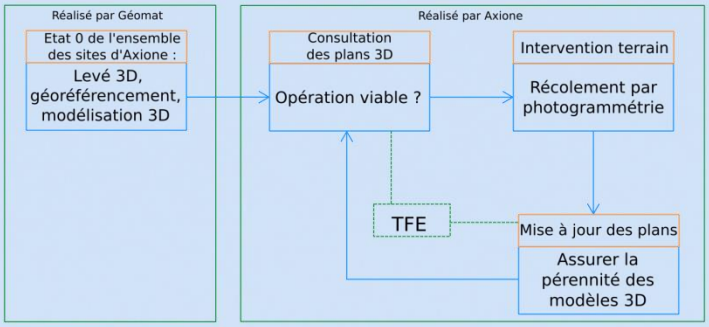
Dans le cas où le placement semble correct au regard de la qualité affichée mais que des collisions sont présente :

-Si le fichier RAPH est bien construit, cela peut provenir de la qualité des autres objets déjà présents dans le plan 3D. Cliquez sur les objets problématique et observez le contenu du champ qualité : si celui-ci est renseigné « D », il se peut que ce soit cet objet qui soit mal placé et non celui qui vient d'être traité.

Annexe 2
Poster

Création d'un plugin Revit de gestion et mise à jour d'éléments 3D à destination de néophytes de la géomatique

La société Axione est chargée de l'entretien des sites de pose d'antennes Bouygues Telecom. De leur activité nait le besoin d'outils simples d'acquisition, de récollement et de traitement qui permettraient la connaissance exhaustive de leurs sites ainsi que leurs gestions.
C'est dans ce contexte que Géomat à proposé à Axione une solution pour améliorer leur productivité.



Le présent travail s'intéresse uniquement à l'étude de viabilité et à la mise à jour des plans

Cahier des charges :

Simple

Cette solution est destinée à l'usage de néophytes de la géomatique. La simplicité d'utilisation est nécessaire pour permettre l'autonomie des utilisateurs.

Fiable

Les utilisations successive de la solution ne doivent pas dégrader la précision afin de fournir une solution pérenne.

Contrôlé

Toute anomalie détectée doit être signalée à l'utilisateur afin que ce dernier puisse assumer ses responsabilités.

Les outils mis à disposition :

RAPH

Système de Récollement Assisté par PHotogrammétrie terrestre dévellopé par FIT ESIC, il permet à des non professionnel de la photogrammétrie d'effectuer une opération de récolement à partir de photographies.

AUTODESK REVIT

Logiciel développé par Autodesk, incontournable pour tout acteur de l'industrie du BIM. Permet entre autre la modélisation, la gestion et la mise à jour d'élément 3D

P&C Plugin PlaceAndCheck

Dévellopé dans le cadre de ce TFE, PlaceAndCheck permet le placement précis d'objet 3D à l'aide des données RAPH. Si RAPH permet à des néophytes d'effectuer du récolement par photogrammétrie, Place&Check est son pendant pour la mise à jour d'élément 3D. Il ne nécessite pas de compétences particulières et est très simple d'utilisation. Il permet ainsi de boucler le processus et permet à Axione de conserver une autonomie complète.

Le plugin est découpé en deux parties distinctes

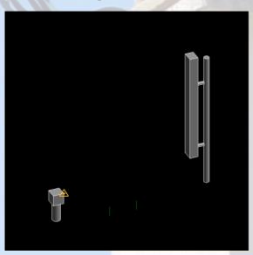
Simple vérification :

L'utilisateur place son objet 3D et est informé des éventuelles collisions entre objets. Cela permet aux équipes d'Axione de juger de la viabilité de leurs futures opérations.

Mise à jour :

L'utilisateur place son objet 3D à l'oeil et celui-ci est replacé à la position décrite par les données RAPH grâce à un calcul de similitude 3D. Cela permet aux équipes d'Axione de mettre leurs plans 3D à jour.

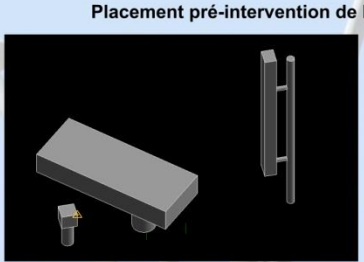
Projet initial :



L'utilisateur place le modèle 3D de l'objet qu'il souhaite installer sur le terrain.

Il peut alors estimer si son installation est effectivement possible ou non.

Placement pré-intervention de l'objet:



Une fois l'équipement installé sur le terrain, l'utilisateur effectue le récolement grâce au système RAPH.

De retour au bureau, l'utilisateur peut alors utiliser les données RAPH pour effectuer la mise à jour de ses modèles 3D.

Insertion grossière de l'objet :

En comparant les coordonnées initiales de l'objet 3D avec les données RAPH, P&C calcule une similitude 3D permettant de placer l'objet précisément.

On renseigne alors un champ de texte de l'objet 3D appelé "Quality" avec une lettre A, B, C ou D en fonction de la qualité du placement.

On signale également à l'utilisateur toute collision éventuelle avec d'autres objets.

L'utilisateur a alors mis à jour son modèle 3D en l'espace de moins de 2 minutes. La solution est simple, fiable et rapide. Le système RAPH produisant des coordonnées précises à 2cm, et le plug-in n'ajoutant pas où très peu d'imprécision (seulement quelques millimètres), la solution proposée à Axione présente des précisions très similaires à leur méthodes actuelles qui est la prise de photos et de cotes au ruban sur le terrain. Avec P&C, Axione économise ainsi un déplacement sur le terrain et améliore sa productivité

Annexe 3
Résumé

CONSERVATOIRE NATIONAL DES ARTS ET METIERS

ÉCOLE SUPÉRIEURE DES GÉOMÈTRES ET TOPOGRAPHES

RÉSUMÉ DU MÉMOIRE

présenté en vue d'obtenir

le DIPLÔME D'INGÉNIEUR CNAM

Spécialité : Géomètre et Topographe

par

Jordi ATTENCIA

Création d'un plug-in Revit de gestion et mise à jour d'éléments 3D à destination de néophytes de la géomatique

Soutenu le 7 juillet 2015

JURY

PRESIDENT : Monsieur Jean-Michel Follin

MEMBRES : Monsieur Michel BARNOUD
Monsieur Vincent HABCHI, professeur référent
Monsieur Olivier POILPRÉ
Monsieur Gwénaël SAGNE, maître de stage
Madame Élisabeth SIMONETTO
Madame Nathalie THOMMERET

Table des matières

Introduction

L'industrie des télécommunications est un milieu très compétitif. Ses coûts techniques sont principalement liés à l'installation et l'entretien des réseaux. Il est donc primordial pour être compétitif de procéder à une maintenance efficace et à moindre coût.

La société Axione, filiale du groupe Bouygues Energies & Services, est un acteur important de l'industrie des infrastructures de télécommunications. Elle effectue de nombreuses poses d'antennes ainsi que les ajouts de modules à ces dernières et leurs maintenances. La problématique d'Axione s'articule autour de deux points :

-L'incapacité de disposer de plans à jour : lors de l'ajout d'un équipement, les équipes d'Axione doivent se déplacer préalablement pour vérifier que les dimensions du lieu de l'antenne permettent l'ajout du dit équipement.

-La tendance actuelle des opérateurs est de déléguer la gestion de leur parc. Il y a donc une nécessité d'en avoir la connaissance exhaustive et des logiciels de gestion appropriés pour les projets d'installations nouvelles et leurs récolements.

L'acquisition de données 3D est une technologie maitrisée par quelques cabinets de Géomètres-Experts. Le traitement de ces données de grandes précisions n'est pas chose aisée particulièrement pour des non-professionnels de la géomatique. La production de modèles 3D des sites et des équipements qu'Axione entretient permettrait de répondre à leur problématique, modèles 3D qu'Axione n'est actuellement pas en mesure de produire, faute de matériel et de compétences de leur personnel.

Le traitement de leur problématique nécessite donc des outils simples d'acquisition, de récolement et de traitement. C'est dans cette optique que GEOMAT a proposé à Axione de créer une base de données 3D de leurs antennes, équipements et lieux de poses. Ceci serait couplé à un système de photogrammétrie terrestre qui permet le lever 3D de manière simple : le système RAPH[5] développé par la société FIT ESIC. Ce système a été développé pour une utilisation simple par des personnes ne disposant pas de la compétence photogrammétrique. Un système de manipulation et mise à jour d'objets 3D sera nécessaire au bon fonctionnement du processus.

Le présent résumé a pour objectif d'expliquer la mise en place d'un processus permettant de manipuler et mettre à jour des objets 3D, et adapter ce processus pour des utilisateurs possédant le minimum de compétences. Il sera traité ici uniquement de la partie **Consultation des plans 3D** et **Mise à jour des plans 3D**.

Dans un premier temps, une description plus poussée du contexte et du cahier des charges du projet sera présentée, puis nous nous attarderons sur la solution retenue. Enfin la 3ème et dernière partie concernera les résultats et les perspectives de ce TFE.

[5] Récolement Assisté par PHotogrammétrie

I Contexte et cahier des charges

Le fonctionnement actuel des équipes d'Axione est le suivant :

-Réception d'une commande : un nouvel équipement doit être installé sur un site.

-N'ayant pas la bonne connaissance de tous leurs sites, une équipe se déplace sur site pour constater l'état des lieux et les équipements déjà installés. Des photos sont prises et des cotes sont relevées au ruban et à la chaîne.

-De retour au bureau, une vérification de la viabilité de l'installation est effectuée à l'aide de photos et cotes. L'équipement est ensuite commandé.

-Un retour sur site est alors nécessaire pour effectivement installer l'équipement.

La méthode est donc peu précise et nécessite des déplacements parfois longs. Nous pouvons estimer la précision d'un tel placement à 5cm lors de la phase de reconnaissance. Il faut donc mettre en place un processus permettant de gagner du temps en évitant cette phase de reconnaissance sur le terrain en la transformant en phase de reconnaissance au bureau via un logiciel.

L'utilisation du logiciel Revit, leader dans le domaine du BIM[6], semble la plus adaptée. Ce logiciel, étant relativement compliqué pour des personnes non familières avec l'environnement AutoDesk et des logiciels 3D, couplé au fait que l'éventail de fonctionnalités proposées est bien plus large que l'utilisation souhaitée, une simplification de son utilisation sera nécessaire.

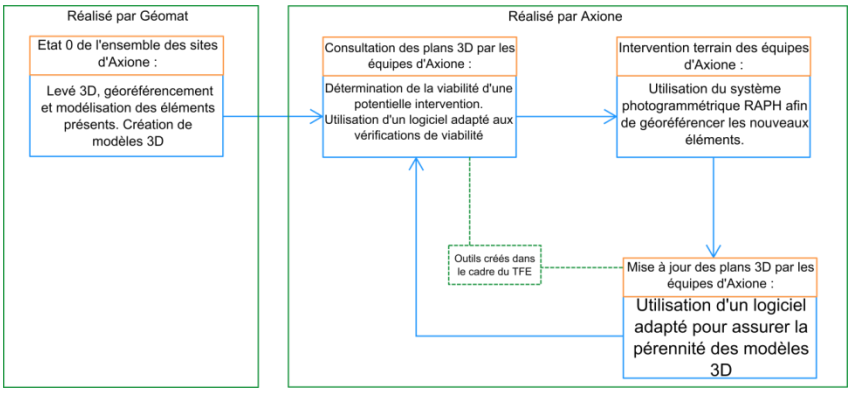


Figure 50 : Schéma du processus envisagé

Ce processus doit apporter au moins la même précision que celle du processus actuellement utilisé par Axione. Il doit également être suffisamment simple pour être utilisé par des personnes totalement étrangères à des logiciels tels qu'Autocad ou Revit. De plus, fournir des informations permettant aux utilisateurs de contrôler la qualité de leurs manipulations est primordial. Il faut donc veiller à ce que ces informations soient suffisamment simples pour être interprétées par les utilisateurs n'ayant probablement pas de compétences en géomatique. Parler d'écart moyen quadratique ou de valeur du facteur unitaire de variance lors de la descente d'un test du Khi-deux portera moins de sens que de parler d'un écart absolu en distance aux yeux de l'utilisateur lambda.

Pour pouvoir placer précisément les objets 3D dans un modèle au sein de Revit, il est nécessaire de posséder un système de coordonnées permettant de situer les éléments les uns par rapport aux autres. Ces opérations pourront être réalisées par GEOMAT. Il est néanmoins important qu'une fois « l'état 0 » réalisé, c'est-à-dire l'ensemble des sites

[6] Building Information Modeling

modélisé et muni d'un système de cordonnées, le total contrôle du processus demeure dans les mains d'Axione qui souhaite rester quasi autonome dans l'exercice de son travail. Axione doit donc être en mesure de rattacher les équipements qu'elle installe. C'est l'objectif du système RAPH.

RAPH permet à partir de cibles disposées sur le site, et de photos prises des équipements à rattacher et des cibles, de rattacher l'ensemble des éléments du site par un système de photogrammétrie terrestre. La précision du géoréférencement de ces photos est de l'ordre de 4mm. S'ajoute à cette erreur, l'erreur de pointé. En effet, l'utilisateur va venir pointer des points sur ces photos assemblées pour extraire les coordonnées de l'équipement qui l'intéresse, erreur estimée à 1cm. Il va ainsi obtenir les coordonnées nécessaires au récolement et permettre la pérennité du processus. L'imprécision globale obtenue est donc de l'ordre de 2cm, ce qui est acceptable au vu des méthodes actuelles d'Axione. Il faut donc que le traitement sous Revit ne dégrade pas ou très peu cette précision et rester ainsi dans les tolérances pour fournir à Axione un outil lui permettant de gagner du temps sans dégrader sa précision. Il faut garder à l'esprit, lors de la création de cet outil que, à l'image de RAPH, il doit rester très simple d'utilisation et suffisamment rapide pour effectivement gagner du temps.

II Développement du plug-in PlaceAndCheck

Indépendamment de la méthode de modélisation choisie, il convient ici de définir précisément la manière selon laquelle l'utilisateur va procéder pour effectuer ses opérations, le but étant que cette manière soit la plus simple et la plus accessible possible. Revit dispose d'une API ou Application Programming Interface. Une API est un ensemble de types et de fonctions qui permet la communication entre un logiciel, en l'occurrence Revit, et un langage de programmation. Grâce à cette API, il est possible de créer un plug-in permettant le chargement de familles dans un projet, son placement dans celui-ci, ainsi que la vérification de certaines contraintes. Cela semble être une solution adaptée.

La solution retenue est la suivante : intégration d'un nouveau raccourci dans Revit permettant le placement d'une famille dans un projet et vérifiant ensuite les éventuelles collisions. Ce raccourci guidera l'utilisateur et l'accompagnera à travers les différentes étapes du processus.

III Résultats et perspectives

L'algorithme développé se scinde en deux parties :

-La partie vérification qui correspond aux opérations pré-intervention terrain : on vérifie qu'il est possible d'installer l'équipement sur le terrain.

-La partie mise à jour qui correspond aux opérations post_intervention terrain : on met à jour les modèles 3D correspondant au site d'installation de l'équipement.

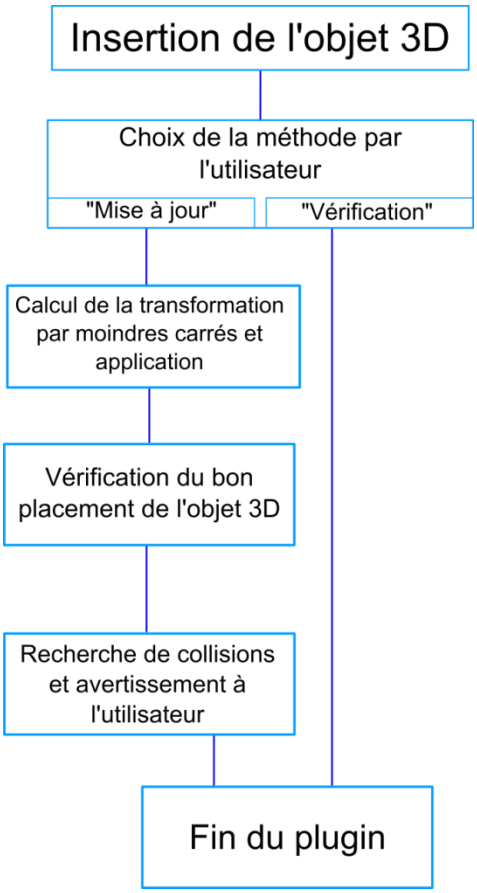


Figure 51 : Algorithme final

Le plug-in crée PlaceAndCheck permettra à Axione de gagner du temps et donc de la productivité. Sa simplicité d'utilisation permet à tout utilisateur de manipuler des modèles 3D de manière précise. Le manuel de l'utilisateur fourni avec l'application détaille chaque étape et prodigue des conseils pour résoudre les éventuels problèmes qui pourraient survenir.

PlaceAndCheck est très polyvalent puisqu'il peut être adapté à de nombreux usages autres que celui que peut en faire Axione. Un électricien venant réaliser un câblage dans un immeuble disposant d'un BIM pourra, à l'aide de RAPH et de PlaceAndCheck, rapidement mettre à jour ce modèle sans que des opérations terrain réalisées par des professionnel de la mesure ne soient nécessaires. Cela permet d'accélérer les échanges et les mises à jour dans le domaine de la construction, rend les professionnels de ce domaine plus autonome et réduit donc le nombre d'interventions des différents intervenants.

Avec l'avènement annoncé de l'ère BIM à travers le monde, le plug-in PlaceAndCheck, sous réserve de mises à jour, pourra trouver sa place au sein de nombreux processus de production, qu'ils soient des créations de BIM de bâtiments existants ou la maintenance de ces dits modèles.

Liste des figures

Création d'un plug-in Revit de gestion et mise à jour d'éléments 3D à destination de néophytes de la géomatique.

Mémoire d'Ingénieur ESGT, Le Mans 2015

RESUME

La création d'un plug-in Revit donnant la possibilité à des néophytes de la géomatique de manipuler des modèles 3D avec précision a permis d'optimiser de la gestion patrimoniale d'équipements de télécommunication.

Ce programme est adaptable a de nombreux usages pour toute gestion et suivi de divers équipement, par tout type de personnes et est implémentable dans des procédures BIM (Building Information Modeling).

Mots clés : Revit, plug-in, néophytes, modèles 3D, BIM.

SUMMARY

The creation of a Revit plug-in allowing neophytes in geomatics to manipulate 3D models with precision.led to an optimization of the gestion of telecommunication equipments.

This program is adaptable to numerous uses for any gestion or control of diverse equipment, by any type user and can be implemented in BIM processes.

Key words : Revit, plug-in, néophytes, 3D models, BIM.