# About the Geo-referencing of BIM models

Abdoulaye Diakité

3D geoinformation, Delft University of Technology

**Abstract**

This report is willing to give an overview of the situation regarding the geo-referencing of BIM models in GIS. The focus is made on identifying how the geographic coordinates of a construction project shall be integrated in its 3D model such that the information is kept in the context of a GIS application. We study the case of BIMs designed using the Autodesk Revit software and exported as IFC. Furthermore, we provide a web service interface which allows to visualize the geographic position associated with an IFC file and correct its Latitude/Longitude by picking new location. The tool is based on the Cesium project allowing to visualize the process on a 3D globe.

## 1 Introduction

The proper geo-referencing of BIM models plays a critical role in the BIM-GIS integration problem. While GIS can provide to BIM a deeper context (e.g. for accounting the surrounding environment of a construction project), this can only work if the 3D model of the project can be properly located on a map of that environment. As we are mainly interested in open formats, by the term "BIM" we mainly refer to the Industry Foundation Classes (IFC) format, which is the most renown open standard associated with the BIM paradigm. Properly geo-referencing an IFC file makes it possible to link the coordinates inside it with its real-world coordinates and thus to place the model of a single building or construction within the virtual environment.

The IFC standard provides adequate classes that allow to describe the required information for its geo-referencing. However, practices have shown that those classes are often poorly or not filled at all in the available project files. Since IFC is not a native file format to any existing BIM software, it results from their export schemes which allows interoperability between them. For this reason, if there is a missing information in the IFC file, mainly regarding the geo-referencing, it either means that it was not provided by the designer at the time of the export or that the software does not support its export. In order to bring further understanding of the issue for both BIM and GIS sides, we study the case of a widely used BIM software (Autodesk Revit 2018) and investigate the tools that it offers to allow proper geo-referencing of its exported IFC files. We also analyse the types of GIS files that it can natively deal with in order to explore possibilities of integration tracks. Furthermore, as a piece of the bridge between BIM and GIS, we provide a tool that allows to visualize the location of an IFC file on a 3D globe of the Earth based on Cesium and a Javascript-based

web interface. The tools also allows to modify and save a newly picked location for the IFC file, as a beginning of a correction tool.

# 2 Classes to Support Geo-Referencing in IFC (optional section taken from the final geobim report)

The IFC standard takes into account the possibility to describe precisely the geographic coordinates of a project. For this, it is possible to use the IFC entity **IfcSite**, which defines an area where construction works are undertaken, and optionally allows for storage of the real-world location of a project using the *RefLatitude*, *RefLongitude* and *RefElevation* attributes.

The latitude and longitude are defined as angles with degrees, minutes, seconds, and optionally millionths of seconds with respect to the world geodetic system WGS84 (EPSG:4326). Positive values represent locations north of the equator, west of the geodetic zero meridian (nominally the Greenwich prime meridian) in IFC2x3, or east of the zero meridian IFC4. Negative values represent locations south of the equator, east of the zero meridian in IFC2x3, or west of the zero meridian in IFC4. These angles are expressed according to the type **IfcCompoundPlaneAngleMeasure** and all components (i.e. degrees, minutes, seconds and millionth-seconds of arc) should have the same sign. According to the IFC standard, the geographic reference given might be the exact location of the origin of the local placement of the IfcSite or it might be an approximate position for informational purposes only. The elevation is defined according to the datum elevation relative to sea level.

In addition, **IfcSite** contains a couple of other attributes that allow for an approximation of the real-world location to be given. The *LandTitleNumber* can store the designation of the site within a regional system (e.g. a cadastral record ID), and the *SiteAddress* can store the postal address of the site.

The IFC entity **IfcGeometricRepresentationContext** is used to define the coordinate space of an IFC model in 3D and optionally the 2D plan of such a model. This entity can be used to offset the project coordinate system from the global point of origin using the *WorldCoordinateSystem* attribute, it defines the *Precision* under which two given points are still assumed to be identical, and it defines the direction of the *TrueNorth* relative to the underlying coordinate system. The latter attribute defaults to the positive direction of the y-axis of the *WorldCoordinateSystem*.

In theory, the use of the latitude, longitude and altitude values in the **IfcSite** with an optional offset and true North direction given by the **IfcGeometricRepresentationContext**, should make it possible to precisely georeference an IFC model. In fact, in our experience most IFC files do fill in the requisite values in the IfcSite. However, these values are almost always set to zero, to a completely wrong location or to a very rough approximation of the real location (e.g. a point in the same city). This is unfortunately compounded by the mismatched definitions of the positive direction for the longitude in IFC2x3 and IFC4.

We therefore recommend that IFC files set their precise real-world location using the latitude, longitude and altitude values in the **IfcSite** taking into

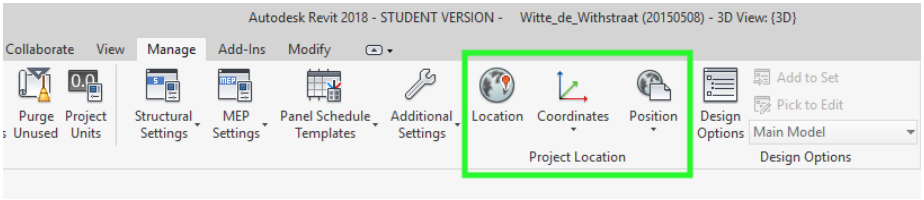



Figure 1: Definition of project location in Revit 2018.

account the offset given by the *WorldCoordinateSystem* of the **IfcGeometricRepresentationContext** for the 3D model and the 2D plan (if used). Due to practical difficulties, it cannot be expected that these values match the reality with the *Precision* given in the **IfcGeometricRepresentationContext**, but the values should be easy to set to within a few meters of the real location. In addition, if the y-axis of the *WorldCoordinateSystem* in the **IfcGeometricRepresentationContext** does not match the true North direction, the *TrueNorth* attribute should be set as well.

# 3 Feeding IFC Classes with Geo-Referencing data using Revit 2018

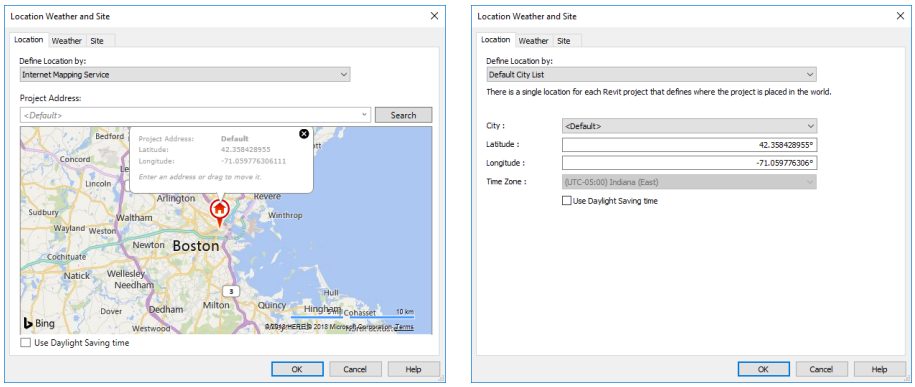
The data that can be found in IFC files are originating from the modeling software used to export them. Here we will study the case of the Autodesk Revit software (v2018) and we will focus on the definition of the latitude, longitude, elevation and true north.

Contrary to the general thought (that the BIM tools are not well adapted for processes such as geo-referencing), there is surprisingly a set of relevant functions made available for designers to incorporate such information appropriately in their models. In the case of Revit, there is a set of buttons that are meant to define the project's location under the *Manage* tab, as illustrated in Fig. 1.

## 3.1 Latitude and Longitude

The information related to the latitude, longitude and elevation of the project are dedicated to the **IfcSite** class. While the *Coordinates* and *Position* buttons (see Fig. 1) are more related to the local coordinate reference system of Revit, the *Location* button allows to make the link to the real world by dealing with those geographic coordinates. Figure 2 shows the options offered to specify such location for a given project. The location can be defined using an *Internet Mapping Service*, which allows to precisely pick a place on a map window or by directly using the address of the project (Fig. 2(a)). On the other hand, the city-based location offers a list of cities to locate the project, which is a less precise geo-referencing (Fig. 2(b)). Alternatively, it is also possible to manually specify the Latitude and Longitude of the project in order to bring more precision.

By default, the project's location is set to a place in Boston (USA), with a latitude value of $42.3584^o$ and a longitude of $-71.0597^o$. For this reason, any IFC model exported from Revit, in which no project location has been specified,




(a) (b)

Figure 2: Map-based (a) and City-based (b) selection of the project's location.

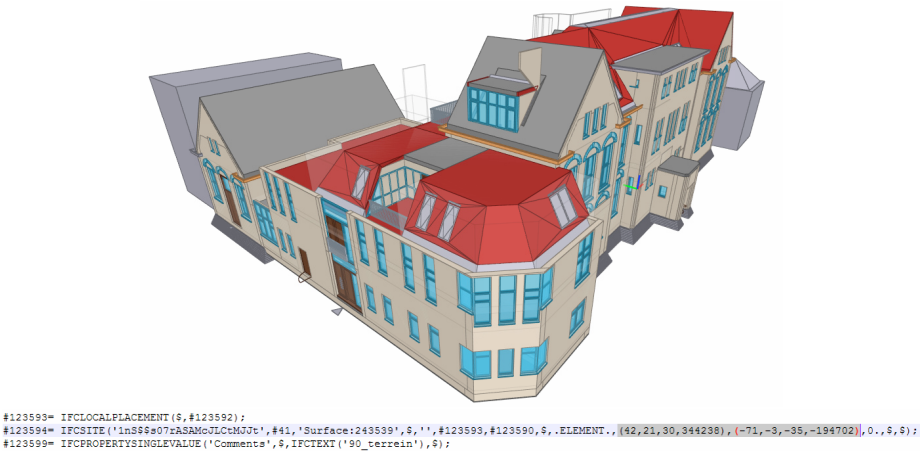



Figure 3: Wrong goe-location of the *Witte_de_Withstraat.ifc* model.

will be located to that same place. It is the case of the *Witte_de_Withstraat.ifc* model, which is supposed to be located in Den Haag (The Netherlands), but which contains the default location coordinates of Revit, as illustrated in Fig. 3 (where the latitude and longitude are expressed in hexagesimal - Degree, Minute, Second and optionally Millionth of second).

## 3.2 Project North vs. True North and Elevation

Another very important parameter for the geo-referencing of a BIM model is the definition of the True North. The latter specifies the real orientation of the geographic North at the difference of the Project North, which defines simply the local North of Revit's coordinate system and corresponds to the Y axis, which is the vector $(0, 1, 0)$ in 3D or $(0, 1)$ in 2D. By knowing the position of the True North and therefore the angle between it and the Project North, the model can be rotated on the XY plane such that it can be oriented properly when placed on a map. However, most of the time, the True North parameter

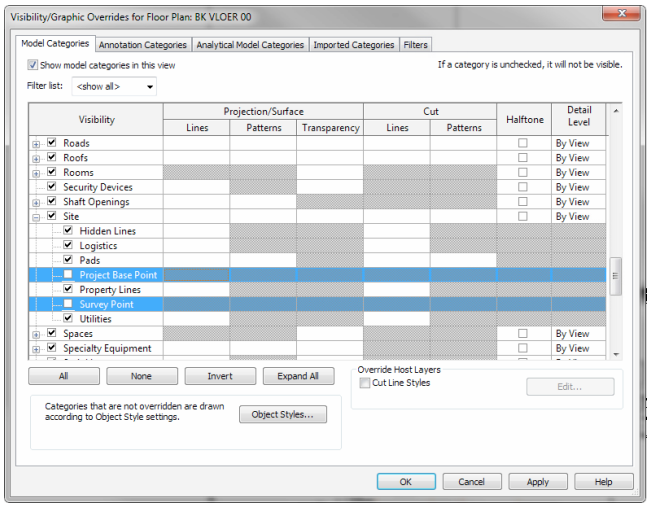
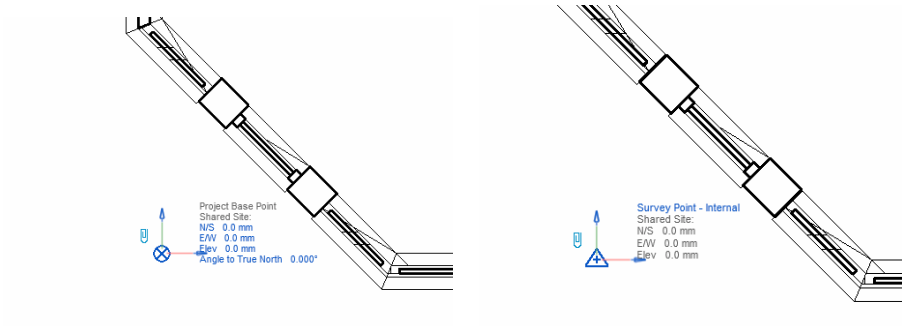



Figure 4: Making the base and survey points visible in Revit.

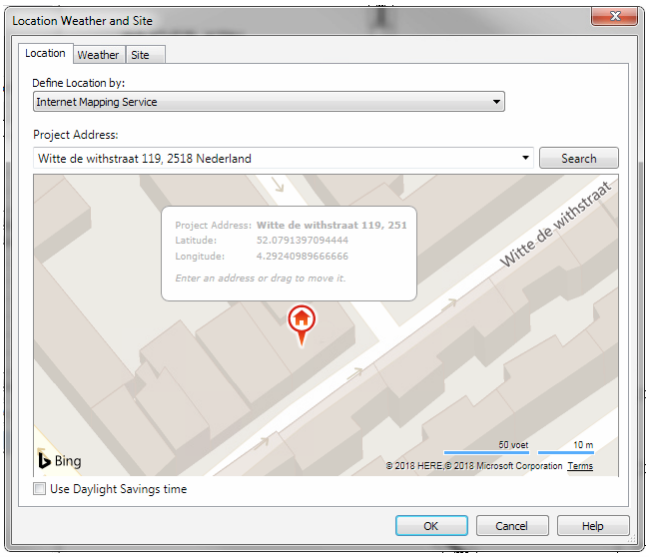



(a) (b)

Figure 5: (a) Project Base Point and (b) Survey Point of the *Witte_de_Withstraat* model.

is neglected, ending up with the default value of the Project North. In other words, unless clearly specified, the *TrueNorth* attribute of the **IfcGeometricRepresentationContext** class of the exported IFC will remain to the value $(0, 1)$, saying that the models are somehow always aligned to the geographic North.

Revit offers here as well the possibility to set the True North properly. This can be done under the "floor plan view", as the coordinates that need to be specified are in 2D (although the IFC standard allows to specify it in 3D as well). Nevertheless, similarly to the elevation, that information is assumed to be provided by the project's surveyor team. To access to them in a given project, one has to enable their visibility under the *Properties* panel of the *Floor Plan* view (usually at the left side of the Revit window), at the *Visibility/Graphics Overrides* option. Figure 4 shows the two features involved, under the *Site* tab, which are the *Project Base Point* and the *Survey Point*. They are both illustrated in Fig. 5. The project base point (Fig. 5(a)) represents the reference




```
#1432827= IFCLOCALPLACEMENT($,#1432826);
#1432828= IFCSITE('1nS$$s07rASAMoJLCtMJJt',#41,'Surface:317636',$,'',#1432827,#1432822,$,.ELEMENT.,(52,4,44,902954),(4,17,32,675628),0.,$,$);
#1432833= IFCPROPERTYSINGLEVALUE('Material',$,IFCLABEL('90_zand'),$);
```

Figure 6: Corrected goe-location of the *Witte_de_Withstraat.ifc* model.

point of all the geometry of the model. It is the origin of the representation, thus the $(0, 0, 0)$ of the project. It is therefore the reference point of the **IfcSite** as well. On the other hand, the survey point (Fig. 5(b)), as its name says, represents the geodetic data related to the project. It is actually the most critical information for all the geo-referencing issues. However, it is assumed to be imported from other sources provided by the surveyors of the project (generally some shared CAD files of the site of construction). When such source files are missing, all the values are set to 0 by default, including the angle to True North, thus assuming that it coincides with the Y Axis of the project.

Now the question is: ***how can we overcome the issue of the missing geodetic information from the survey point?*** In theory, the easiest option would be to import GIS data of the site (if known) directly in Revit and add the missing data, before exporting a well located and oriented IFC model. But in practice, this is not as smooth as this. In the following section we will explore the possibilities offered by Revit to correct the geo-referencing of an IFC file containing wrong location (such as the *Witte_de_Withstraat* model).

# 4 Correcting the geo-location of an IFC file using Revit

## 4.1 Importing GIS data

Revit does not deal with common GIS file types. It can import natively common CAD files only, such as DWG, DXF, gbXML, etc. The exchange with the surveyor is mostly done using DXF files. But other types of files such as KML, CityGML, GeoJSON, etc., which can deal with proper geo-location are not supported. Although some solutions seem to exist, as extension of Revit (e.g.

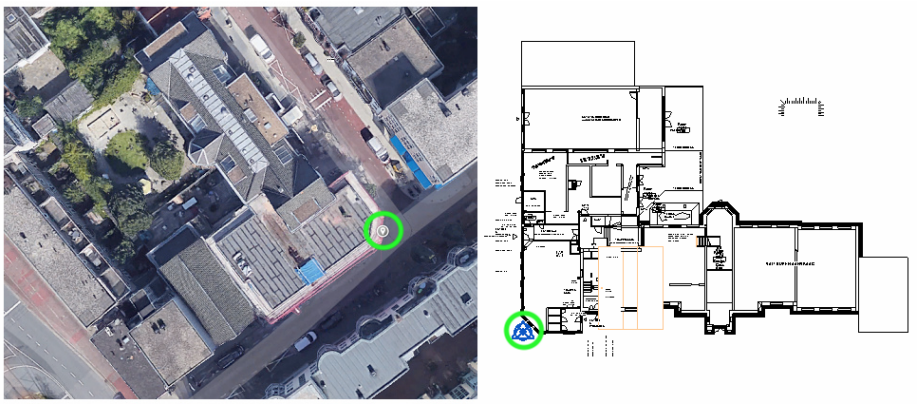


Figure 7: Screen shot of Google Maps to identify the orientation of the Building against the True North. The green circles indicate the project base and survey points (right) and their approximated equivalent point on the image (left).

Dynamo [1]), we will not explore such track in this report. Instead, we will explore the remaining ways that we can rely on to reach our goal.

## 4.2 Correcting the Latitude and Longitude of IFC files

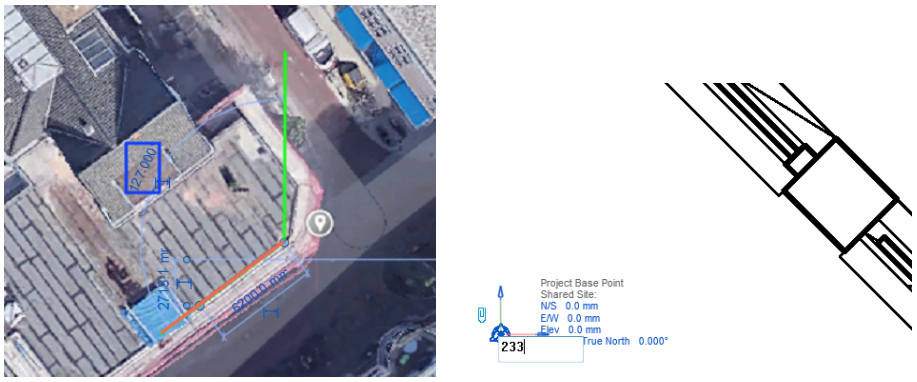
As discussed in the previous section, the latitude and longitude can be specified quite easily with the *Location* tool. To test it, we used the address of the project to set its new location, as illustrated in Figure 6. The figure shows also that the change is considered in the newly exported IFC file, as we can identify the new latitude and longitude inserted in the **IfcSite** attributes. The elevation is kept at 0.0, which is a reasonable assumption in the Netherlands. More precision could have been brought by manually placing the reference point closer to the project base point. But since the latter is not placed at an easily identifiable point of the building (e.g. a corner or an edge), this would still remain an approximation.

## 4.3 Correcting the True North

Correcting the True North is a little bit more challenging than the latitude and longitude. The information we need here is simply the angle difference between the Project North and the True North. This is where importing GIS data could have been useful, but since we do not have such option here, we will have to do some manual work. As it is possible to import images in Revit, we will rely on a screen-shot of the satellite image of the real location, with the building properly oriented with respect to the True North. Figure 7 shows the floor plan of the model next to a screen-shot extracted from Google Maps, which reveals the real orientation of the buildings in the IFC model compared to the geographic North (towards the upper side of the image).

By spotting on the image the point that nearly corresponds to the project base point as well as the survey point in the case of our test model, it is pos-

[1] https://knowledge.autodesk.com/support/revit-products/learn-explore/caas/screencast/Main/Details/5040f6cc-2cfb-4513-9a59-30fd78edeca7.html




(a) (b)

Figure 8: (a) Detection of the angle between the true north (green line) and the project north (orange). (b) Correction of the Project Base Point's True North attribute.

sible to visually recognize the corresponding line of the project north. Then the problem turns down to determining the angle between that line and the geographic north one. This can be done by drawing two lines corresponding to those directions, as illustrated in Fig. 8(a), where the orange lines have been drawn following the left-most wall on the floor plan in Fig. 7. The measuring tools of Revit allow then to simply compute the angle, which is $127^o$ in this case. That value being the angle from the green line to the orange line in the counterclockwise direction, the correct value of the angle to the true north from the project north is equal to $360 - 127 = 233^o$, the new value entered in the *Angle to True North* attribute of the *Project Base Point* (8(b)).

Once this parameter is set, one can switch from the project north to the true north, and vice versa in the Revit interface, as shown in Fig. 9(a) and (b). Furthermore, all the changes are reflected in a new IFC file when exported. The newly generated IFC classes integrate the correct information as shown in Fig. 9(c) and the model can be directly oriented after being loaded. In Revit, we can then see that the new Survey Point gives the right direction (Fig. 9(d)).

With the above operations, we have shown that it is possible for a BIM model to be properly geo-referenced in Revit, but it is also possible to correct its wrong location information after it has been already exported to IFC. A corrected IFC could then be generated afterward. However, several geometric and semantic issues occur in the process of importing and exporting IFC files. Thus, in the scope of our experiences, we are not able to guarantee the integrity of the newly generated file in comparison to its original one (e.g. several pre-processes of Revit imply geometric operations and removal of items in the latter). Furthermore, as active supporter of open-source tools, we are willing to provide a freely accessible tool which is able to provide similar corrections in a less invasive way, meaning that only classes related to geo-location would be handled. We will introduce such tool in the following section.


Figure 9: Superposed floor plan and image oriented with respect to the (a) true north and (b) project north. (c) New **IfcDirection** of the true north. (d) New *Survey Point* of the project.

# 5 IfcLocator: an Open-source Web Service for Geo-Referencing IFC models

We introduce the ***IfcLocator***, a supporting tool for BIM and GIS practitioners, which allows to simply visualize the actual location of an IFC file and eventually bring correction to that information when necessary. The tool is based on Cesium[2], which is an open source Javascript library for 3D mapping, including an intuitive 3D viewer of the globe. The choice of a web-based Javascript open source code was motivated by the high flexibility offered by such approach (highly cross-platform). Furthermore, the users can perform all the operation locally, which means they do not have to send their model to a remote server. Thus no privacy issues as well.

## 5.1 Download and Installation

The IfcLocator tool can be downloaded freely on Github[3]. Its installation requires few steps, which can be summarized as follow:

- installing Cesium;
- copying the IfcLocator project in the Cesium folder;

[2]https://cesium.com/
[3]https://github.com/tudelft3d/IfcLocator

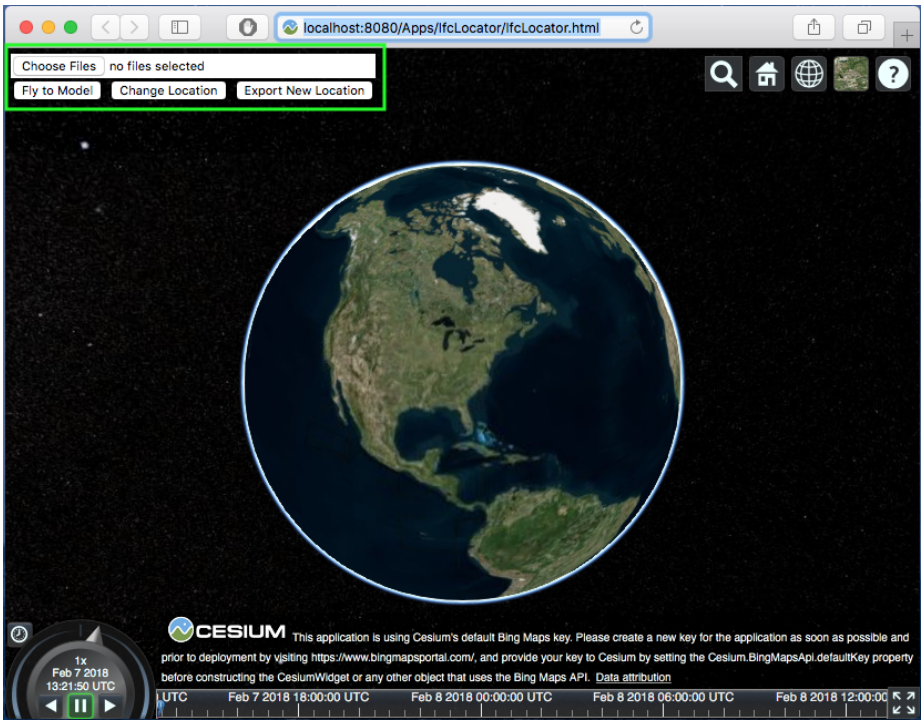



Figure 10: IfcLocator's interface and all its added functions (in the green box).

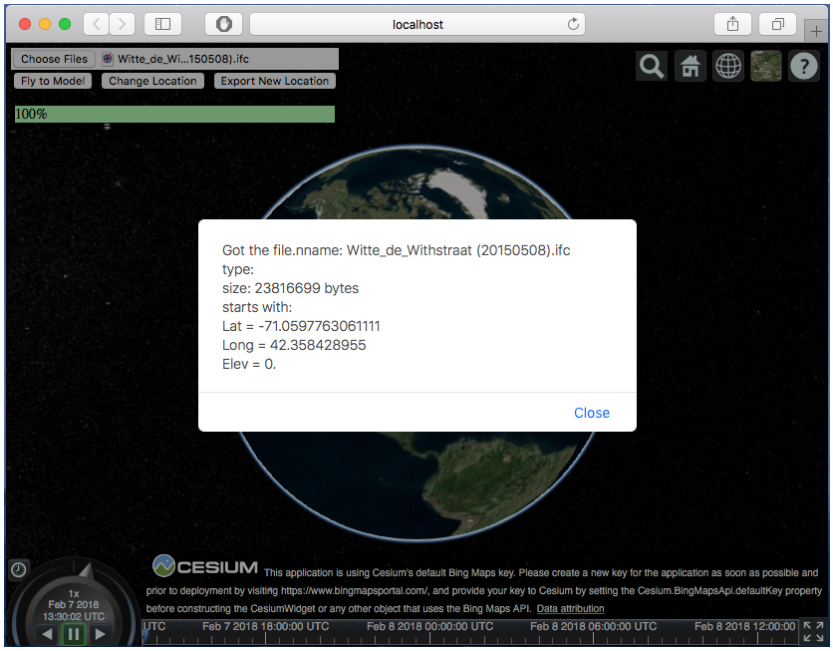
- running the tool on a web browser.

The detailed steps can be found at the download page of the project. The necessity for the user to download Cesium is the downside of performing all the operations locally, without the need for the user to send his files to a remote server. This is not a tricky step anyway, as its process is quite easy and straightforward, but in the future, other solutions could be considered to avoid that step.

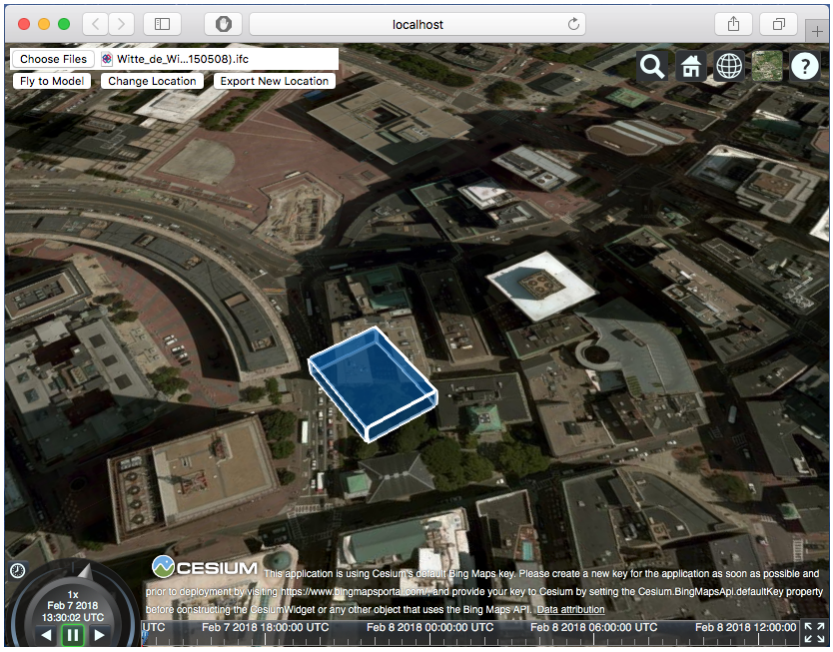
## 5.2 Overview of the main functionalities

The interface of the tool is simple. It is a web page including the Cesium viewer as it can be seen in Fig. 10. The functions that we have added to the viewer can be seen at the top left side of the image. The first button allows to import locally an IFC file. Once the latter is selected and loaded, the camera of the viewer flies automatically to the detected location of the IFC model. That location is characterized by the latitude, longitude and elevation that can be found under the attributes of the **IfcSite** class of the file (see Fig. 11(a)). For now, a simple 3D box is used to represent the geometry of the model (Fig. 11(b)).

Once the model is located, the user can determine if the location is correct or not. In the latter case, the user can find the proper location either by manually browsing the globe or by entering the address on the research bar (the magnifying glass button). By default, the Cesium's viewer integrates several types of map imagery, including satellite and topographic images. Once the lo-




(a)




(b)

Figure 11: (a) Log of the information after opening a file. (b) 3D box created at the detected location of the file.

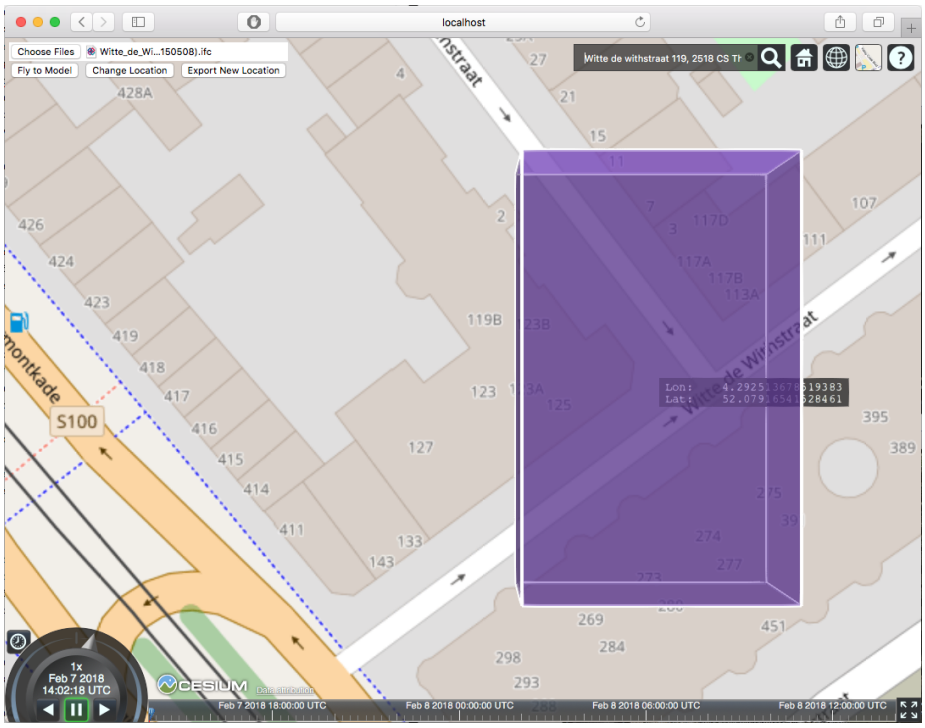



Figure 12: Setting up a new location for the IFC file. The center of the box lays at the location picked by the user.

cation is identified, a small box showing the latitude and longitude of the cursor of the user can be activated my continuously pressing the **Shift** button of the keyboard. When the user clicks at the desired point, which is equivalent to the survey point mentioned earlier for Revit, the new location (illustrated by the small grey box near the cursor) can be attributed to the model by clicking on the *Change Location* button. Figure 12 illustrates the result of the process. It can be seen that the map layout has been switched from satellite imagery to the topographic map (OpenStreet Map in this case), where the footprint of our model offers a better visibility.

The 3D box that is drawn to represent the building uses as center point (of its footprint) the newly picked location by the user. Because the box is not really representative of the shape of the real building, the user cannot really expect a higher precision with the current version of IfcLocator, unless the Project Base point is known. However, the newly selected location can be exported in a new IFC file. IfcLocator will simply copy the original IFC file as it is and bring changes where needed (here, in the **IfcSite** class). Later on, when the new file is loaded, it will directly appear where the user has specified.

## 5.3 Limitations and Future improvements

The ambition of IfcLocator is to provide a complete tool to support the geo-referencing process. Because of the short time dedicated to this first version of the project, only limited functionalities have been implemented, which leaves

room for several possible improvements.

One of them is related to the appearance of the building in the interface. As previously explained, only a simple 3D box is used so far to represent the building. It would be way more intuitive to provide a better approximation of the geometry of the model so that the user can better spot and correct the actual location of the building on the map. While the ultimate (but not achieved yet) solution to this problem would have been the extraction of the exterior shell of the building, a simpler solution could be the usage of a coarser global geometry of the building (e.g. convex hull). But in order to achieve this, one needs to handle properly the geometric classes of the IFC file, which leads us to the next point of possible enhancement.

The actual method used to deal with the information in the IFC file is a simple parsing approach. The file is parsed as a basic *ascii* text file and the needed classes are detected based on keyword search, using basic functions of Javascript. While it has the advantage to be straightforward and very portable, it is very limited when it comes to deeper processes, as it gets less optimal (in scripting as well in running time). An alternative to this could be to used a specialized IFC library for Javascript and as such, BIM Server's Javascript API [4] seems to be one of the rare options (although its usage may imply the usage of a remote server for the BIM model).

Another point of improvement is the orientation of the model with respect to the true north. This issue is also related to the two previous ones: to the first one because any visualization/correction of that attribute would require a good perception of the original model; to the second one because integrating the the true north's reading and writing on en IFC file would imply tricky steps with the actual parsing approach, without even necessarily guaranteeing a generic work-flow. Therefore, improving the parsing of IFC classes seems to be the key basis to any further improvement related to the IfcLocator tool.

# 6 Conclusion

While geo-referencing is thought to be exclusively a GIS matter, it plays a key role in the BIM-GIS integration as it is one of the strongest ways to provide a contextual view to any construction project. This report has shown that in fact, there are several tools made available in BIM software products (e.g. Revit), allowing to support the proper definition of the geo-location information for a BIM model and export them as IFC. Without claiming that we have explored all the ways possible for the targeted process, a quick exploration of Revit has allowed us to perform the task successfully. We pushed further and introduced the IfcLocator tool, which stand as an open-source alternative to the commercial software products. Although at its very early stage, the tool is already able to provide an intuitive point of view of the actual location stored in an IFC file and perform simple change of location. With further investigations and developments, this can definitely help the practitioners to address the issue easier and become a milestone in the whole process of BIM/GIS integration.

[4] https://github.com/opensourceBIM/BIMserver-JavaScript-API