Development of a Geo to BIM converter: CityJSON importer plugin for Autodesk Revit

Amir Hakim Jasper van der Vaart Ken Arroyo Ohori
Jantien Stoter

This is an author's version of the paper. The authoritative version is:

Development of a Geo to BIM converter: CityJSON importer plugin for Autodesk Revit. Amir Hakim, Jasper van der Vaart, Ken Arroyo Ohori and Jantien Stoter. In Thomas H. Kolbe, Andreas Donaubauer and Christof Beil (eds.), *Recent Advances in 3D Geoinformation Science: Proceedings of the 18th 3D GeoInfo Conference*, Lecture Notes in Geoinformation and Cartography, Springer Cham, Munich, Germany, February 2024, pp. 249–263. ISBN: 978-3-031-43699-4 (eBook), 978-3-031-43698-7 (Hardcover), 978-3-031-43701-4 (Softcover), ISSN: 1863-2246, eISSN: 1863-2351
DOI: 10.1007/978-3-031-43699-4_15

The integration of 3D city models and Building Information Models (BIM) in the context of GeoBIM has gained significant attention from both academia and industry. Harmonizing the distinct characteristics and goals of these models is crucial for successful integration. In this paper, we present the development of a plugin for Autodesk Revit, a popular BIM platform, which allows for the incorporation of 3D Geo-data encoded in CityJSON. The plugin, published as open source, enables the generation of individual geometries with associated city model attributes as parameters, facilitating analysing the impact of new or changed buildings (modelled in BIM) on the environment (captured in geo-data). Challenges addressed during development include georeferencing, data format import, handling different geometry approaches, hierarchy of attributes, code optimization, user-friendliness, and enhanced visualization. The plugin contributes to the seamless integration of geo- and BIM data, enhancing interoperability and supporting informed decision-making in the Architecture, Engineering, and Construction and urban domains.

1 Introduction

In recent times, the integration of 3D geoinformation, specifically 3D city models, with Building Information Models (BIM) has emerged as a significant subject of discussion to enhance urban applications. This convergence, commonly referred to as GeoBIM, has garnered substantial attention from diverse academic domains such as geoinformation, geomatics, construction, architecture, and urban planning. Furthermore, it has attracted interest from various non-academic entities including government-affiliated institutions, National Mapping and Cadastral Agencies, and private enterprises [1].

In the domain of GeoBIM, it is evident that 3D city models and BIM exhibit different goals and have therefore distinct characteristics (e.g. level of detail, designed versus measured geometries, and different semantic objects). This implies that in order to successfully integrate these models for any application, it is imperative to establish consistency and alignment in their features. Such harmonization usually entails ensuring that all data conform to the requirements of either the 3D city models or BIM. Alternatively, if a third-party use case is identified, they may adhere to a different set of specifications [1].

Many studies have focused on making BIM models at lower levels of detail available in geo-applications. With the growing interest to take the environment into account in the design and construction phase of a building or infrastructure project, there is also a growing interest to make geo-data, often available as open data in large datasets, available in BIM applications.

This paper presents our solution to make 3D geo-data, encoded in CityJSON, available in a BIM application. CityJSON [2], a JSON-based encoding (JavaScript Object Notation) of the CityGML data model (version 3.0.0) is an open, standardized format for exchanging and storing digital 3D models of cities and landscapes. CityJSON has been accepted by the Open Geospatial Consortium (OGC) as a community standard. CityJSON establishes methods for describing prevalent 3D features and objects encountered in urban environments, including buildings, roads, rivers, bridges, vegetation, and city furniture, along with their interrelationships. A CityJSON file encompasses both the geometric and semantic information of city features within a specified area. It also introduces standard levels of detail (LoDs) to enable the representation of objects at various resolutions, tailored to diverse applications and objectives [2].

The primary objective of CityJSON is to provide an alternative to the GML encoding of CityGML, which often proves verbose, complex, and challenging to work with. CityJSON strives for user-friendliness, facilitating the reading and creation of data sets. Its design prioritizes programmers, allowing for the rapid development of supporting tools and APIs. Additionally, CityJSON emphasizes compactness, typically achieving a 6x compression of publicly available CityGML files. Moreover, it aligns with the requirements of web and mobile development. A CityJSON object, representing a city, adopts a flattened structure, where the CityGML hierarchy is simplified, including only the essential city objects that are the lowest level of the hierarchy. This approach significantly streamlines the storage of city models without sacrificing information.

For this research, we developed a plugin for Autodesk Revit as the BIM platform for incorporating 3D geo-data encoded in CityJSON. This decision was made since Autodesk Revit is the main commercial software in the field of BIM. The plugin that we developed is published as open source [3] and freely available from the Autodesk app store [4].

There are already other solutions available to import geo-data in BIM software. Existing Geo to BIM solutions for Autodesk Revit generate city models as a single object represented by a shell in BIM environment. In contrast, our plugin generates each geometry individually and associates city model attributes as parameters to the resulting models. By creating individual geometries, the plugin allows for a more detailed representation of the city models

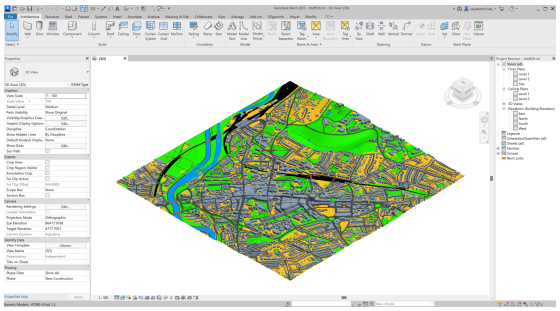

Figure 1: 3D City Model of Southampton, England created via CityJSON importer plugin

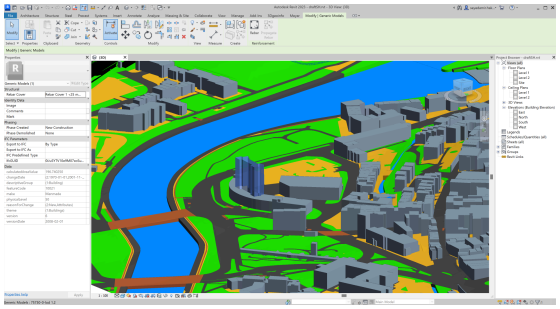

Figure 2: Visualization and selected geometry information from Southampton City Model in Autodesk Revit Software

within the BIM environment. This granularity enables more precise analysis and evaluation of how each specific element in the BIM model interacts with the surrounding environment. In addition, it allows for easier isolation and manipulation during analysis. Users can focus on specific elements, make adjustments, and explore various scenarios, fostering a more flexible and detailed examination of the BIM model's influence on the surrounding context.

The plugin's advancements in directly incorporating 3D Geo-data encoded in CityJSON into Autodesk Revit, rather than converting it to IFC for import offers a more streamlined workflow by eliminating the need for an intermediate format conversion. This direct integration saves time and reduces complexities associated with data transformation and data loss of workflows that can read IFC files but that cannot implement customised choices. In addition, the plugin offers a specialized solution tailored explicitly for Autodesk Revit, optimizing the integration process and ensuring compatibility with Revit's features and capabilities. This targeted approach provides a more efficient and effective solution for Revit users compared to the more generic IFC-based method.

Based on a CityJSON file's coordinate referencing system (CRS) and metadata, the plugin reprojects and translates imported data from CityJSON file for implementation within the Autodesk Revit environment.

This paper starts with a description of related work. Then the challenges that needed to be addressed for the development of the plugin are described. We finalise the paper with conclusions and future work.

2 Related Works

The integration of 3D geoinformation with Building Information Models (BIM) in the context of GeoBIM has been a subject of considerable research and development. Numerous studies have explored different approaches to achieve seamless interoperability between these distinct models. In this section, we review some of the relevant related work in the field of GeoBIM and data integration, upon which our plugin is built.

2.1 GeoBIM Integration Tools Evaluation

Noardo et al. [1] conducted a benchmark study on tools for GeoBIM integration, focusing on IFC (Industry Foundation Classes) georeferencing and conversions. The study evaluated various tools for importing 3D city models into BIM platforms, including methods for georeferencing and transforming spatial data. The authors highlighted the importance of harmonizing different data models and discussed the challenges and opportunities in achieving efficient GeoBIM integration.

2.2 Georeferencing Techniques in BIM

Jaud et al. [5] investigated georeferencing techniques in the context of building information modeling. The study explored different methods for linking BIM models to their corresponding real-world coordinates and discussed the advantages and limitations of each approach. The authors emphasized the significance of georeferencing in larger-scale projects involving infrastructure and the potential benefits for construction and urban planning applications. Another study by Jaud et al. [6] further examined the georeferencing of IFC models. The authors proposed an approach to meet the requirements of both infrastructure and building industries, enabling the placement of BIM models within a digital version of the site's encompassing environment. The study discussed the challenges in handling geospatial data within BIM platforms and provided recommendations for effective georeferencing.

2.3 GeoBIM Project Experience

Arroyo Ohori et al. [7] presented a practical experience of processing BIM and GIS models in a GeoBIM project in the Netherlands. The study discussed the challenges and opportunities for integrating geo-data with BIM models and showcased the development of a GeoBIM plugin for a popular BIM software. The authors demonstrated how the plugin allowed for efficient manipulation and utilization of imported geospatial information within the BIM environment.

Deng et al. [8] developed an instance-based method to map IFC from BIM and CityGML from GIS. They used the Semantic City Model, a reference ontology, to capture relevant information during mapping, ensuring accurate data exchange. The study also harmonized different levels of detail (LoDs) in CityGML for comprehensive mapping. The results demonstrated automatic data mapping between IFC and CityGML in various LoDs, enhancing interoperability.

2.4 Open-Source Plugin for CityJSON Support

Vitalis et al. [9] developed an open-source plugin to support CityJSON in QGIS, a popular open-source GIS software. The plugin enabled the import and visualization of 3D city models encoded in CityJSON format within the QGIS environment. The study highlighted the importance of open data formats for seamless data exchange between platforms as required for GeoBIM workflows.

Vroegindeweij [10] created an open-source Dynamo package which is a visual programming platform to facilitate the import of CityJSON objects into Autodesk Revit. His approach imports and visualizes geometries in each 3D city model file as a whole shell within the BIM environment instead of handling individual objects available in the 3D city model. In addition, basic knowledge of visual programming is needed for using the package.

The existing related works have emphasized the importance of seamless integration between 3D geoinformation and BIM models. Researchers have explored various techniques for georeferencing, data transformation, and data exchange between these models. Our work builds upon these studies by developing a plugin for Autodesk Revit that allows for the incorporation of 3D Geo-data encoded in CityJSON, enabling the analysis of the impact of the BIM model on the environment and enhancing interoperability between the two domains.

3 Challenges that needed to be addressed

This section describes the challenges that needed to be addressed in the development of the plugin to meaningfully import CityJSON data into Revit. These challenges are not limited to Revit and represent general

challenges for making geo-data available for BIM applications.

3.1 Georeferencing

In a general context, the process of georeferencing an object encompasses the transformation of its geometric coordinates in a manner that guarantees precise placement at the correct geographical location on a map, while also achieving congruence with other features existing within the map's framework [11].

The geometric context of a BIM model is generally regarded as a localized, three-dimensional Euclidean space, wherein objects situated on the construction site are represented utilizing a Cartesian coordinate system with an origin in the project site and axes usually aligned with the features in the model[5]. In light of recent research developments, there have been notable updates to open BIM that have enabled the representation of models using a projected coordinate referencing system (CRS). This advancement is particularly advantageous for larger-scale projects, such as those involving infrastructure [6].

Derived from Computer-Aided Design (CAD), BIM models often utilize local coordinate systems that are software-specific, leading to an inaccurate or entirely absent georeferencing information associated with these models when delivered. Properly georeferencing a BIM file makes it possible to link the (local) coordinates inside a BIM model with their corresponding real-world coordinates, and thus to place the model of a single building or construction within a digital version of the site's encompassing environment [7].

Autodesk Revit offers multiple methods for establishing the geographical context of a model, including the internal origin, project base point, and survey point. Users employ various approaches to link their BIM models using these points (see Fig. 3).

The plugin specifically addresses the internal origin within the BIM software, granting users the option to either update or retain the Revit site location (internal origin) based on the CityJSON file location. If the user decides to update the internal origin of the project, the plugin also provides the capability to preserve the project base point and survey point (see Fig. 4).

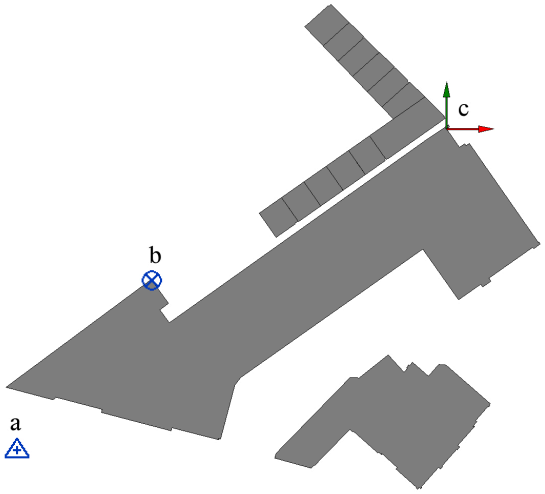

Figure 3: Different origin points in Autodesk Revit environment: Survey Point (a), Project Base Point (b), and Internal Origin Point (c).

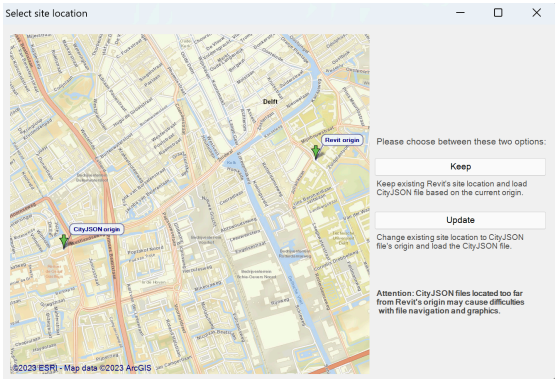

Figure 4: CityJSON importer plugin's keep/update site location feature

3.2 Data format

CityJSON's adoption of the JSON format offers advantages in terms of interoperability and data exchange. JSON is a lightweight, human-readable, and language-independent data interchange format, making it suitable for seamless communication between different systems and applications. Its textual representation enables easy comprehension and ensures flexibility in data handling.

The integration of CityJSON data with the Revit environment through .NET programming languages like C# and VB is an efficient approach that enables seamless manipulation and utilization of geo-data within architectural and construction workflows. These languages offer robust support for data manipulation, object-oriented programming, and efficient memory management, making them well-suited for handling complex geo-data and performing sophisticated calculations.

The integration process begins with reading and parsing the CityJSON data using established JSON libraries within C# (Newtonsoft.Json). This parsing allows for extracting and representing the relevant information from the CityJSON files, facilitating further analysis and processing.

Subsequently, the plugin applies various calculations and transformations to the processed geo-data. These operations includes geometric manipulations like translation, rotation, or scaling, ensuring precise alignment of the data with the coordinate systems and other location references in the Revit environment.

To accomplish the integration with Revit, the RevitAPI is harnessed through C# to access the software's extensive object model and features. The RevitAPI provides the interface to interact with Revit's model and allows for the creation, modification, and manipulation of architectural elements. This process adheres to the principles of computer science, such as Abstraction, Modularity, Encapsulation, Data Structures, Algorithms, Object-Oriented Programming, Software Design Patterns, and Error Handling. This ensures consistent data flow and maintains the integrity of the Revit model.

3.3 3D geometry representation

The geometry structure in Autodesk Revit and CityJSON varies. In CityJSON, points are represented as vertices within the file, while each surface is formed by a set of connected points. Multiple surfaces can be combined to create multi-surfaces and solids. In contrast, direct shapes in Revit possess distinct surface or volume properties, which are based on multiple surfaces generated by sets of XYZ points. These differences in geometry representation between Revit and CityJSON highlight the varying approaches and properties associated with the two formats originating from two different domains that needed to be bridged.

In regard to representing 3D geometry objects in a BIM environment, three main approaches are commonly utilized and widely employed:

3.3.1 3.3.1 Boundary Representation (B-Rep):

Boundary Representation, also known as B-Rep or solid modeling, is a widely adopted technique for representing 3D objects. B-Rep describes objects as collections of surfaces, edges, and vertices, providing a detailed depiction of their boundaries. This method enables precise definition and manipulation of the object's geometry. B-Rep has found extensive use in various applications, including computer-aided design (CAD) systems and geometric modeling [12].

3.3.2 3.3.2 Sweep Solid (SS):

Sweep Solid represents another approach to 3D geometry representation. In this method, a 2D profile or cross-section is swept along a predefined path, resulting in

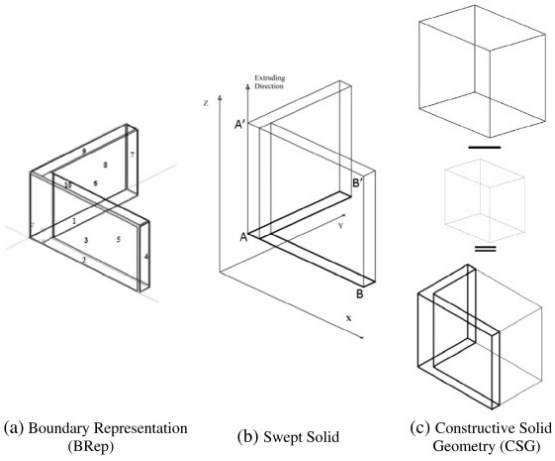

Figure 5: BRep, Swept Solid. and CSG [8].

the creation of a 3D solid. The resulting object maintains the shape and characteristics of the original cross-section as it moves along the specified path. SS proves particularly beneficial for generating objects with rotational or linear symmetries and finds applications in architectural modeling, industrial design, and computer graphics [13].

3.3.3 3.3.3 Constructive Solid Geometry (CSG):

Constructive Solid Geometry, or CSG, constitutes a modeling technique that involves combining simple geometric primitives through set operations such as union, intersection, and difference. Primitives like cubes, cylinders, and spheres serve as fundamental building blocks for creating complex objects through Boolean operations. CSG offers a concise representation of objects and facilitates the construction of intricate shapes by combining simpler elements. This approach has been widely employed in computer graphics, virtual reality, and solid modeling applications [14].

As CityJSON utilizes the Boundary Representation (BRep) strategy for representing 3D geometry objects, we have opted to utilize the BRep to BRep approach, specifically involving the conversion of boundary representation (BRep) models from CityJSON format to BRep models in Autodesk Revit. This chosen methodology allows for the effective transfer of geometric information and enhances interoperability between different software platforms for 3D modeling and building information management. Additionally, seamless representation of 3D objects is ensured during the conversion process, minimizing inconsistencies or inaccuracies.

3.4 Hierarchy of attributes

The challenge of dealing with the hierarchy of attributes and their appropriate assignment is a significant aspect in the integration of CityJSON data into the Revit environment. CityJSON's hierarchical data model, with attribute inheritance and propagation, aims to optimize storage and processing efficiency. However, Revit's lack of inherent support for such hierarchical inheritance necessitates a carefully designed solution to ensure accurate representation of the hierarchical structure within the BIM environment.

In the context of related studies, the mapping of CityJSON's hierarchical data model has been recognized as a crucial undertaking, emphasizing the importance of addressing this issue to achieve a successful integration [9]. Scientifically, the mapping process requires a systematic approach that considers the relationships and dependencies between attributes at different levels of the hierarchy.

To address the disparity between CityJSON's hierarchical data model and Revit's attribute assignment approach, a rigorous solution was developed. This solution entails individual assignment of relevant attributes to each geometry within the BIM environment. This process involves dissecting the hierarchical structure of CityJSON and mapping attributes to corresponding elements in the Revit model.

In the mapping process, advanced algorithms and data structures may be employed to efficiently handle the complexity

of attribute propagation within the CityJSON hierarchy. The analysis requires traversing the hierarchical data structure and identifying the relevant attributes to be assigned at each level. Techniques such as depth-first or breadth-first searches can be utilized to navigate the hierarchy and propagate the attributes accurately.

Additionally, the solution may require the development of custom data models or data mappings to reconcile the differences in attribute representations between CityJSON and Revit. By creating a specific mapping schema, scientists and engineers can systematically and uniformly link the attributes in CityJSON to the corresponding attributes in Revit, ensuring that no information is lost or misinterpreted during the integration process.

Furthermore, data validation and consistency checks are crucial steps to verify the accuracy and integrity of the attribute assignment. By conducting comprehensive validation tests, potential discrepancies or errors in the mapping can be detected and rectified, ensuring that the hierarchical structure is faithfully represented within the BIM environment.

The approach to handling attribute inheritance and assignment ensures that the integrated CityJSON data within the Revit environment retains its hierarchical structure and associated attributes accurately. This enables architects, engineers, and urban planners to leverage the benefits of both CityJSON's optimized data storage and Revit's powerful BIM functionalities. The developed solution fosters a seamless and efficient workflow, promoting collaboration and data exchange across different platforms and stakeholders in the architectural and construction domains.

3.5 Performance

During the software testing phase, the production of certain city models experienced delays for two primary reasons. Firstly, the transition between JSON information, C# libraries, and the Revit API posed challenges that required careful handling and integration. Secondly, some city models were significantly large in size, containing substantial amounts of data and information. To address these issues, the code underwent multiple reviews and optimizations to improve efficiency and streamline the processing of large city models.

Revit file sizes (for local, network, or cloud-based models), including linked Revit files, can become very large, up to 2 Gigabytes or more. When working with large-sized Revit files, several factors can significantly impact performance. These include file size, model complexity, hardware specifications, and network speed. These files with extensive model elements, such as high-detail families, numerous components, and intricate geometries, can lead to slow loading times, degraded system responsiveness, and an increased likelihood of system crashes. Creating a city model within Revit can lead to generation of large-sized files and entails performance implications that necessitate consideration.

Geometries situated far from the startup location can trigger various graphic rendering artifacts in Revit. As the distance increases, the precision of rendering and representation diminishes, leading to visual inconsistencies, aliasing, and texture distortion. The software's display algorithms may struggle to accurately portray intricate details of remote geometries, resulting in jagged edges, blurred textures, and misaligned elements, thereby compromising the visual quality and realism of the model (see Fig. 6).

To effectively address the graphic rendering artifacts experienced with geometries far from the startup location in Autodesk Revit, the plugin offers the "update site" option, as mentioned in the Georeferencing section. This adjustment significantly improves the precision of rendering and representation, thereby mitigating visual inconsistencies, aliasing, and texture distortion commonly associated with remote geometries.

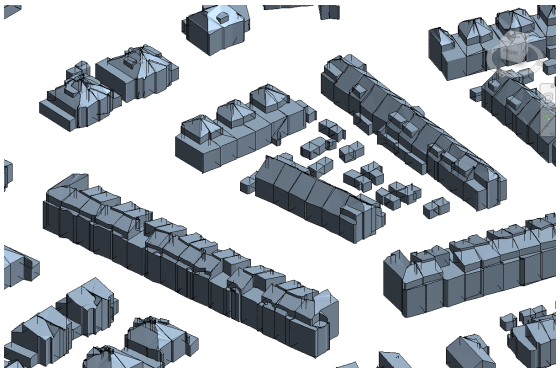

Figure 6: Graphic glitches and distortions in an Autodesk Revit project with geometries located far from the origin.

3.6 Friendly for BIM users

During the publication and deployment of the app on the Autodesk app store, the plugin was required to comply with all possible users' errors. The design of the plugin focused on handling various scenarios such as opening incompatible file formats, dealing with corrupted files, and seamlessly opening existing files. Moreover, the user interface was designed with the intention of being user-friendly and easily accessible for all BIM users. Additionally, the results obtained from the plugin automatically trigger the zooming of the 3D view for enhanced visualization and navigation through the project.

The plugin undergoes regular updates and redevelopment, driven by valuable feedback from BIM users.This commitment to continuous improvement ensures that the plugin stays up-to-date with the evolving needs and preferences of the BIM user community.

3.7 Enhanced visualisation

As a BIM software, Revit can establish connections with other software applications to facilitate collaboration. The development team has addressed this aspect during the development process. For instance, they have incorporated the capability to assign distinct materials to each class of CityJSON objects within the BIM environment, allowing for enhanced visualization purposes.

4 Conclusion

In this paper, we have presented the development of a Geo to BIM converter in the form of a CityJSON importer plugin for Autodesk Revit. The integration of 3D city models and BIM in the context of GeoBIM has garnered significant attention from academia and industry. Harmonizing the distinct characteristics and goals of these models is essential for their successful integration, and our plugin aims to bridge the gap between these two domains.

The plugin allows for the incorporation of 3D Geo-data encoded in CityJSON into Autodesk Revit, a popular BIM platform. By generating individual geometries with associated city model attributes as parameters, the plugin enables users to analyse the impact of the BIM model on the environment during the design phase. This approach fosters a seamless workflow, facilitating the handling of GeoBIM information within a unified environment and enhancing interoperability between geospatial and BIM data. The plugin's direct incorporation of CityJSON data into Autodesk Revit offers a streamlined workflow, preserves attributes, enhances analysis, improves GeoBIM interoperability, and provides a dedicated Revit solution, surpassing the method of converting CityJSON to IFC for importation into Revit.

During the development process, several challenges were addressed, including georeferencing, data format import, handling different geometry approaches, hierarchy of attributes, code optimization, user-friendliness, and enhanced visualization. Each challenge was tackled systematically, employing advanced algorithms, data structures, and software engineering principles to ensure the robustness and efficiency of the plugin.

5 Future Development

The focus of this application revolves around generating generic models within a BIM environment based on CityJSON geometries. For future advancements, it would be highly beneficial to generate geometries using their semantic attributes specific to Autodesk Revit. For example, the topography of a site could be represented as a Toposurface or Toposolid within Autodesk Revit. While Toposolid generates a solid 3D volume to portray the topography, Toposurface generates a 2D surface that faithfully captures the terrain's contours and shape. Both options can be generated by utilizing points or imported data. By ensuring semantically correct generation of topography within the Revit environment, designers and engineers are empowered to seamlessly incorporate and analyze site topography in their BIM models, conduct volume calculations, and make informed design decisions.

In the existing version, the app performs a reprojection of CityJSON points to the EPSG 4326 coordinate reference system (CRS) as the base reference. Although directly altering the coordinate system within Revit is not possible, it is feasible to extract the CRS information from an imported CAD file. As a further development, the application will incorporate the functionality to reproject CityJSON geometries to the acquired CRS from the imported CAD file.

The CityJSON importer plugin for Autodesk Revit contributes to the seamless integration of geospatial and BIM data, supporting informed decision-making in the Architecture, Engineering, and Construction (AEC) and urban domains. By addressing the challenges and requirements of GeoBIM integration in a rigorous manner, the plugin opens new avenues for collaboration, visualization, and analysis, fostering advancements in the field of 3D city modeling and building information management.

6 Acknowledgement

This project has received funding from the European Union's Horizon Europe programme under Grant Agreement No.101058559 (CHEK: Change toolkit for digital building permit).

References

[1] Noardo, F., Harrie, L., Arroyo Ohori, K., Biljecki, F., Ellul, C., Krijnen, T., ... & Stoter, J. (2020). Tools for BIM-GIS integration (IFC georeferencing and conversions): Results from the GeoBIM benchmark 2019. ISPRS international journal of geo-information, 9(9), 502.

[2] Ledoux, H., Arroyo Ohori, K., Kumar, K., Dukai, B., Labetski, A., & Vitalis, S. (2019). CityJSON: A compact and easy-to-use encoding of the CityGML data model. Open Geospatial Data, Software and Standards, 4(1), 1-12.

[3] Hakim, A. (2023). Github repository for cityjsonToRevit, Accessed: 20-06-2023.

[4] Hakim, A. (2023). CityJSON Importer for Autodesk Revit, Accessed: 25-07-2023.

[5] Jaud, Š., Donaubauer, A., Heunecke, O., & Borrmann, A. (2020). Georeferencing in the context of building information modelling. Automation in Construction, 118, 103211.

[6] Jaud, Š., Clemen, C., Muhič, S., & Borrmann, A. (2022). GEOREFERENCING IN IFC: MEETING THE REQUIREMENTS OF INFRASTRUCTURE AND BUILDING INDUSTRIES. ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences, 10.

[7] Arroyo Ohori, K., Diakité, A., Krijnen, T., Ledoux, H., & Stoter, J. (2018). Processing BIM and GIS models in practice: experiences and recommendations from a GeoBIM project in the Netherlands. ISPRS International Journal of Geo-Information, 7(8), 311.

[8] Deng, Y., Cheng, J. C., & Anumba, C. (2016). Mapping between BIM and 3D GIS in different levels of detail using schema mediation and instance comparison. Automation in Construction, 67, 1-21.

[9] Vitalis, S., Arroyo Ohori, K., & Stoter, J. (2020). CityJSON in QGIS: Development of an open-source plugin. Transactions in GIS, 24(5), 1147-1164.

[10] Vroegindeweij, M. (2021). GIS2BIM for Dynamo 2.x, Accessed: 10-02-2023.

[11] Snyder, J. P. (1987). Map projections–A working manual (Vol. 1395). US Government Printing Office.

[12] Hoffmann, C. M. (1993). Geometric and solid modeling: An introduction. Morgan Kaufmann.

[13] Vandenbrande, J., & Vos, R. (2001). Sweep solid generation: Algorithms for the non-manifold case. In Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (pp. 61-70). ACM Press.

[14] Requicha, A. A. G., & Rossignac, J. R. (1985). Constructive solid geometry: Representation and computation. ACM Computing Surveys (CSUR), 17(4), 335-375.