2021 European Conference on Computing in Construction
Online eConference
July 26-28, 2021

A CLOUD IFC-BASED BIM PLATFORM FOR BUILDING ENERGY PERFORMANCE SIMULATION

Kyriakos Katsigarakis, Georgios N. Lilis, Dimitrios Rovas
University College London, London, UK

ABSTRACT

The BIM Management Platform (BIM-MP) is a digital Building Information Model processing tool, responsible for handling BIM data that conform to the IFC standard. It provides an integrated data management solution for storing, versioning, updating and checking IFC data, which are created and modified by AEC practitioners. The embedded API allows the data exchange between BIM-MP and other online tools which forward IFC files into the BIM-MP repository to perform various operations using different BIM-MP functional modules. Some of the modules create visual and textual reports regarding data quality issues in terms of data consistency, completeness and correctness, while others enrich the IFC data with new IFC objects with the necessary semantic links. All these modules are deployed as standalone containerized applications using the Service-Oriented Architecture (SOA) design principle.

INTRODUCTION

The proliferation of BIM usage within Architectural Engineering and Construction domains, paved the path to transparent project collaboration, which led to the development of multiple open BIM handling software platforms such BIMserver (Beetz et al. 2010) , xBIM (Lockley et al. 2017), to name a few. On the other hand, the latest efforts to reduce the new and existing buildings' carbon footprint, by assessing design alternatives and retrofitting scenarios, led to an increased use of Building Energy Performance Simulation (BEPS) programs.

To support these work pathways, a cloud based framework of BIM processing software tools, called BIM Management Platform (BIM-MP), is introduced. The basic philosophy and goal of BIM-MP, which differentiates it from other openBIM platforms, is the support of automatic BEPS model generation from open-BIM data conforming to the IFC ISO standard (ISO et al. 2018). To this direction, BIM-MP offers data quality checking and data enrichment services to AEC practitioners, in a unified cloud-based service-oriented architecture. BIM-MP platform is used to support H2020 project BIMERR operational workflows.

The rest of the paper is organized as follows: Initially, the platform's architecture is presented followed by the description of its functional and supporting components. BIM-MP's revision control management scheme is also analyzed in a separate section. The paper concludes with the description of operation, assumptions and restrictions based on the current state of development. Application results on a pre-validation building, from a H2020 project called BIMERR, are presented in respective subsections.

PLATFORM ARCHITECTURE

BIM-MP is responsible for providing functionalities in relation to the storing, checking, updating, and querying of BIM models. These models conform to the openBIM International Foundation Classes (IFC) standard (ISO 16739:2018) which contains a rich set of classes designed to provide a robust interoperability solution for data exchange between different built environments and software applications. This section describes the back-end system architecture of a cloud-BIM management solution that enables the interoperability between functional modules and external clients.

High-level Architecture

BIM-MP includes a core and a set of reusable modules to support synchronous and asynchronous requests in a unified manner. Some modules can respond to requests immediately while others require more time depending on the complexity of the BIM models. All of them include a set of low- and high-level libraries to perform their business logic operations. Figure 1 displays a generalized view of BIM-MP's architecture with its core and reusable modules.

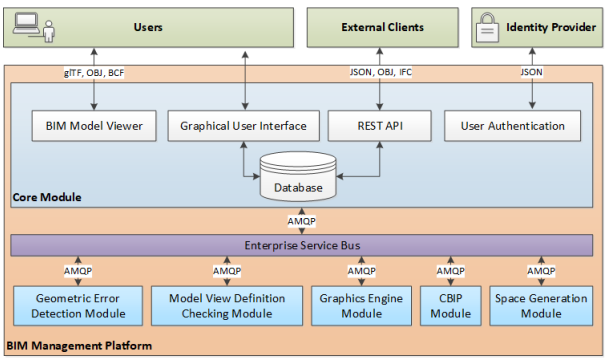

Figure 1: BIM-MP's high-level architecture

The high-level libraries are responsible for handling and querying BIM models and for extracting the required information for the execution of the low-level operations. The low-level libraries support multithread processing and exchange information with the high-level libraries through the JNI programming framework. The modules are deployed in the cloud infrastructure as a standalone Spring Boot Application following the Software-Oriented Architecture (SOA) design principle.

For the communication of the different modules, BIM-MP uses the Red Hat Fuse Enterprise Service Bus (ESB) which is in charge of providing the asynchronous communication using queues as well as for handling the routing of the messages. The ESB acts as a message broker providing the required queues and facilitates interoperability. The ESB includes Extract-Transform-Load (ETL) logic and custom mappers to transform the payload of the messages in a proper form according to the requirements of the receivers. In the proposed architecture, BIM-MP modules are standalone applications based on the Spring Boot Framework. Each of them contains high-level and low-level libraries to perform its business logic operations. We defined the following modules as a basis for the prototype implementation of the platform:

1. Core Module contains the GUI implementation and REST API of BIM-MP. It uses high-level libraries to parse BIM models and extract the necessary information used by the functional modules.
2. Geometric Error Detection (GED) Module integrates geometric model-checking functionalities into the platform helps BIM designers to create an error-free model in terms of geometry in the scope of building energy performance simulation analysis. The GED module reports detected errors to the designer in a visual form using OBJ and BCF data.
3. Model View Definition Checking Module helps BIM designers to validate BIM models in terms of data completeness based on predefined rules following the mvdXML specification.
4. Graphics Engine Module uses the geometric information of BIM models to generate B-Rep solids of the structural and non-structural building elements. It stores the B-Rep solids in BIF using OBJ files. Additionally, the embedded WebGL viewer of the BIM-MP uses the B-Rep solids for a visual representation of the model through the GUI.
5. Common Boundary Intersection Projection (CBIP) Module uses geometric operations to enrich the BIM model with 2nd-level space boundaries and shading surfaces.
6. Automatic Space Generation Module uses the geometric information to generate the geometry of the spaces and enriches the BIM model.

Each of these modules exchange information using data structures which conform to open standards. The Core Module itself controls the binding between the BIM-MP front-end system and the functional modules.

Module Orchestration

As mentioned earlier, BIM-MP implementation follows the SOA design approach. The distributed nature of the platform requires orchestration techniques and a messaging framework that provides a loose coupling between components, to achieve performance and reliability. The use of asynchronous messaging offers many benefits, but also brings challenges such as the delivery sequence of messages and the concurrency between the different BIM-MP modules.

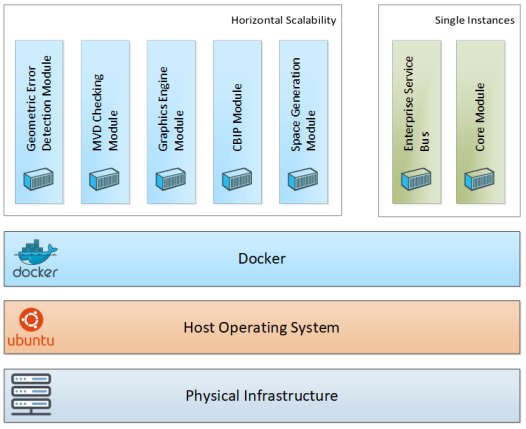

Figure 2: BIM-MP module orcherstration

A secure and private containerized environment hosted on dedicated server infrastructure is used to provide the BIM-MP functionality. The container images facilitate the portability and distribution of workloads in a standardized manner, and allow developers to package all software components and dependencies into reusable units.

CORE MODULE

The Core Module is deployed as a cloud application and provides the web-based interface and the API of BIM-MP. It uses a set of software components and libraries to handle the BIM models and to extract the information required by the functional modules.

Web-based User Interface

The Core Module is developed in Java EE, adopting the micro-service design pattern. It includes the GUI (see screenshot of Figure 3) of the platform and provides authorized access to users and tools through the Identity Provider.

The Core Module is based on the Spring Boot Framework and uses its embedded web-server to support the Model-View-Control (MVC) design pattern.

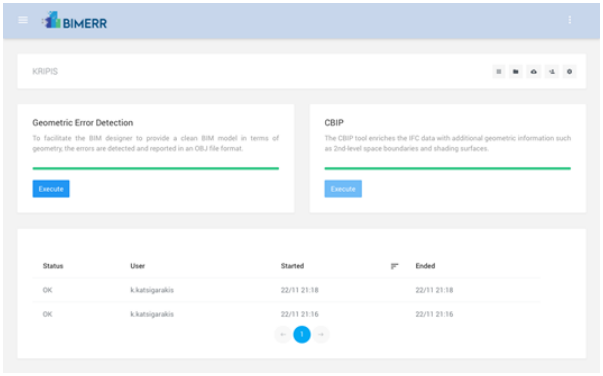

Figure 3: BIM-MP's modules control panel

Moreover, the Core Module uses the Spring Security Framework and the Spring Keycloak Adapter to handle access policies for the BIM-MP. The Identity Provider grants access to BIM-MP for accessing information such as user data, user roles and groups. Apart from the requirement to support modern web technologies, the Core Module includes a set of components to configure a database for storing its data. Moreover, it includes AMQP adapters to initialize asynchronous communication interfaces with the functional modules through the Enterprise Service Bus.

Data Model

The Core Module of BIM-MP includes entities, attributes, and relationships between entities, to represent the logical data model which stores the data needed to perform the BIM-MP operations (illustrated in Figure 4).

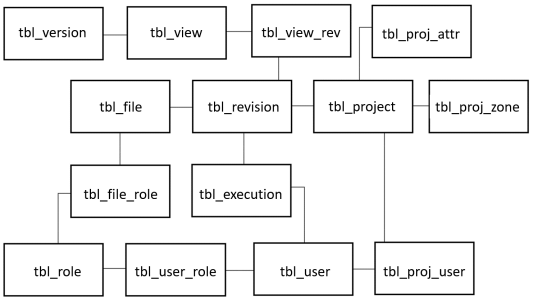

Figure 4: BIM-MP's data model

Using the Spring Framework and Hibernate2, the automatic deployment of the database is driven based on Java Persistence API (JPA) annotations. In this context, Java classes represent the tables, while some fields inside the classes represent the relations between different tables. The framework supports all types of relations such as one-to-one, many-to-one, one-to-many and many-to-many. When using this approach, the relational database can be transparently managed from Java, increasing the abstraction level of the persistence layer. The Core Module requires a connection to a MySQL Server. MySQL Server is a Relational Database Management System (RDBMS) that support multi-tenancy. The Hibernate framework utilizes the MySQL dialect to access the BIM-MP's database to perform transactional operations and queries.

SUPPORTING COMPONENTS

BIM-MP uses software components and libraries to handle the complexity of the functional modules and to increase the automation and reliability of the business logic operations. It contains five primary software components described below:

EXPRESS Schema Compiler

The EXPRESS Schema Compiler is a standalone application that uses the Java EE Code Model framework to generate the IFC Java classes directly from EXPRESS data. It can parse successfully the most frequently used IFC releases, from IFC2X3 to IFC4X1. Initially, the application transforms the EXPRESS data into in-memory objects using an internal data model representation. Then, a set of predefined transformation rules is applied on the objects to instantiate the representation of the Code Model. Finally, the Code Model generates the IFC classes and organizes them into Java packages based on the release of the IFC schema. This IFC class generation and Java class organization process from an input IFC EXPRESS schema, is illustrated in Figure 5.

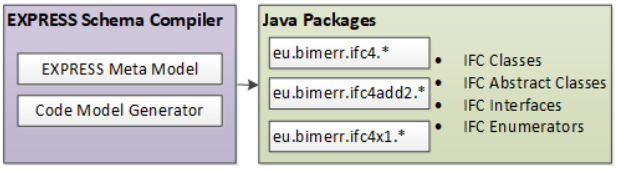

Figure 5: IFC Classes generation using the EXPRESS Schema Compiler

IFC Java Library

The IFC Java Library uses the generated IFC Java classes to parse efficiently the STEP data and to instantiate a representation of the BIM model. The current version of the IFC library can handle the most frequently used IFC releases, from IFC2X3 to IFC4X1. The IFC Java Library provides an API that includes a set of useful functionalities for manipulating the loaded objects. Moreover, it supports the initialization of reverse relations. Reverse relation is the addition of the instance of an object to the corresponding collection of the inverse connected instances of another object. The creation of the IFC components from an input IFC model by the IFC Library, is demonstrated in Figure 6.

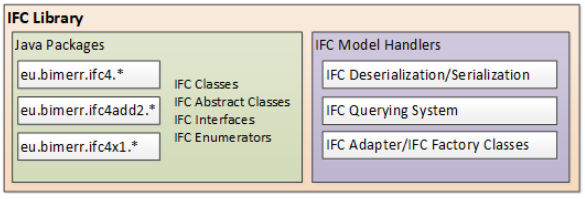

Figure 6: IFC Java Library Components

IFC Geometry Exporter

The IFC Geometry Exporter extracts the geometric content and the respective semantics of an input IFC model, to provide the necessary geometric data to low-level geometric algorithms, implemented in C++. The binding between the Geometry exporter, which is developed using the IFC Java library, and the low-level C++ geometric routines, is implemented via JNI programming interface. To extract the geometric content, the exporter serializes the input IFC file and populates respective XML classes organized in a tree-like structure, following the building hierarchy. This XML file population from an input IFC model, is displayed in the block diagram of Figure 7.

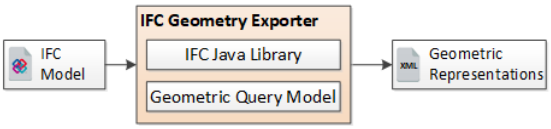

Figure 7: IFC geometry exportation process

IFC Optimizer

The IFC Optimizer performs lossless compression of large IFC files, to speed up the loading process and the execution of some ETL tools such as the IFC Geometry Exporter. The IFC exporters of BIM authoring tools often generate multiple instances of the same object, increasing IFC verbosity. The IFC Optimizer uses the IFC Java Library to compare the hash values of the IFC objects and to merge them, in case they are equal. Furthermore, it updates the express identifiers of the deleted objects to maintain the original connections. This optimization operation is applied on an input IFC_0 model to produce an output IFC_1 model, as illustrated in the block diagram of Figure 8.

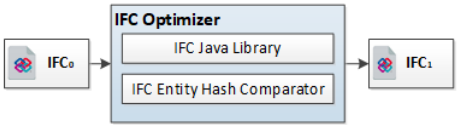

Figure 8: IFC file size optimization process

IFC Unit Converter

In general, the tools and algorithms require the input BIM model to conform to the defined MVD that ensures the semantics and completeness of the used IFC data. The units of values in an IFC file depend on the IFC Exporters of the BIM authoring tools. The IFC Unit Converter uses the methods of the IFC Java Library to convert the values of basic and derived units into a new unit system as illustrated in the diagram of Figure 9.

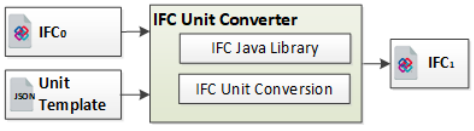

Figure 9: IFC unit converter process

REVISION CONTROL

BIM-MP can handle the geometry and the semantic information of a BIM model that is provided in IFC. In the IFC schema specification, IFC objects may reflect a final as well as a transient state. For instance, during a renovation project, external BIM tools may continuously revise IFC objects. In case multiple tools are making changes to the same objects, IFC schema supports local copies of the modified IFC objects. This revision scheme identifies changes, per object, instead of identifying changes in text. An IFC object is considered modified when: a) any of its direct attributes change; b) any of its referenced resources change; and c) items are added or removed from any collection. Within this revision scheme, each IFC object is marked with a change action indicating if the IFC object was ADDED, MODIFIED, DELETED or not changed.

Object-based Tracking Changes

BIM-MP can efficiently determine which objects have been added and modified in a series of IFC files modified by different external software tools. Finally, BIM-MP can either merge or reject these changes, as determined by the user. For this tracking service to work, when an external tool updates the IFC file, should set the ChangeAction attribute of the IfcOwnerHistory of the new added IFC objects to ADDED, of the modified objects to MODIFIED and of the deleted objects to DELETED. If no change is applied, the ChangeAction attribute of the IfcOwnerHistory should be set to NOCHANGE. Finally, the OwningApplication should be updated with the application characteristics of the updating tool.

FUNCTIONAL COMPONENTS

Overview

In this section, the function of individual BIM-MP components is analyzed. BIM-MP components can be grouped into four categories: (1) Data quality checking, (2) Semantic enrichment, (3) Geometry engine and (4) Revision control components. These categories and their respective BIM-MP functional components are described in the following subsections.

Data Quality Checking

The quality of IFC data should be checked regularly, as there multiple software applications which alter the IFC data content. These data quality checking operations can be classified into three categories: a) schema compliance where the consistency of the IFC data is checked based on the IFC data schema, b) model view definition (MVD) checking where the completeness of the IFC data files are checked, and c) Geometric Error Detection operations where the geometric correctness checks are performed to IFC data files. These operations are analyzed in the following subsections.

Schema Compliance

One of the advantages of the openBIM standards, such as the ISO IFC4 ADD2 TC1 (ISO et al. 2018) data exchange format, is the transparent data exchange among different tools within the AEC domain. In this standard, BIM data are coded in a STEP file, the structure of which is defined according to an EXPRESS schema conforming to the EXPRESS data modelling language. Taking this into account, the IFC Schema Compliance Checker of BIM-MP validates the STEP data of the input IFC files against the corresponding EXPRESS schema defined in the standard. This functionality is integrated into the IFC Library of BIM-MP and includes validation of datatypes, class names, the range of numerical variables and the sizes of the data collections.

MVD Checking

The IFC specification provides a multi-domain information model for capturing building data such as geometry, materials, components, properties and more. To support specific data exchange requirements between different tools and processes only a subset of the IFC specification is required in terms of used entities and properties. The Model View Definition (MVD) specification (buildingSMART 2020) allows to define reusable Concept Templates and Rules for describing precisely the data exchange requirements. Along with the maintenance of the IFC specification, buildingSMART has published two general-purpose Model View Definitions described below:

IFC Reference View is mainly used by tools and processes that do not require modifications of the geometry. The geometric representation is optimized for analysis and display purposes but excludes parametric geometry definitions.

IFC Design Transfer View includes instances with support for editing the geometric representations of building elements and spaces. It is the preferred MVD in BIMEER, because it enables the enrichment of the BIM model with new geometric information and property sets.

BIM-MP provides an MVD Checking Module to help users to automatically validate the completeness of the IFC data against specific rules defined using the mvdXML format. Three steps are needed to achieve automatic MVD validation of a BIM model:

1. The creation of the mvdXML file using the IfcDoc tool
2. The application of the rules on the IFC objects using the IFC Library
3. The generation of the validation report based on openBIM standards such as the BCF

IfcDoc has been developed by buildingSMART International, to improve the computer-interpretable implementation of the specification. The user can create custom a Model View and assign new Concepts. Each Concept contains the applicable IFC Entity connected to a Concept Template and the applied Rules that include the logical operations, e.g. entities that are checked by type or properties that are checked by value.

The MVD Checking Module takes as an input the IFC data and the mvdXML data. BIM-MP's IFC Library is then used for the STEP data serialization and for the initialization of the inverse relations of the IFC instances. The output of the tool is a detailed report of the validation process for each individual Concept. This process is summarized in Figure 10. MVD checking will examine whether the data requirements of a BIMEER tool are satisfied. These requirements include material thermal properties, property sets, operating schedules etc.

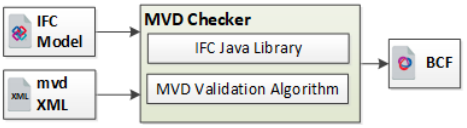

Figure 10: BIM-MP's MVD checking process

Geometric Error Detection

Geometric errors contained in IFC files affect significantly Building Energy Performance Simulations (BEPS) since they alter the geometric content of BEPS models i.e. the second level space boundary topology (Bazjanac 2010), which is derived from the geometric content of BIM (IFC) files using well documented and tested algorithms (Lilis et al. 2017). BIM-MP contains all the necessary software components to perform the necessary checks to detect geometric errors which affect BEPS model generation process. These checking procedures are executed during the Geometric Error Detection (GED) process (Katsigarakis et al. 2019), the block diagram of which is presented in Figure 11.

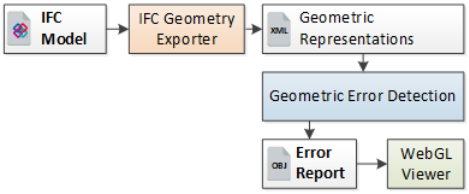

Figure 11: BIM-MP's GED checking process

According to the process diagram of Figure 11, the GED process receives the geometric content and semantics of the IFC model, extracted by the IFC Geometry Exporter, performes the detection process described next and outputs an error report in OBJ format (Wavefront, 1992). This error report is forwarded to the WebGL viewer, for user display.

Detection process If at least one internal building space volume description is present in the IFC file, then a BEPS-dedicated geometric detection process is executed by BIM-MP which, according to the geometric error classification of (Lilis et al. 2015), includes the following three stages:

1. Surface error detection In this first stage, the surface integrity of the boundary representations of all architectural elements, including the building spaces, is checked. Missing surfaces or surfaces with inverted surface normal vectors (inverted surfaces) are reported. The geometric solids passing these checks pass to the next stage.
2. BEPS-clash error detection During this stage only clashes with the clash surfaces attached to neighbour building spaces are detected (neighbour space condition). This kind of clash errors affects the BEPS model generation. If the clash involves a building space volume, then it is reported in this stage without taking into account the neighbour space condition.
3. Space error detection Finally, the BEPS geometric error detection process concludes with the space error detection process, described in the previous section, applied to all building space volume geometric descriptions contained in the IFC file.

Data quality is of paramount importance for the BIM to BEPS automated model generation processes to properly function. Since error-free data seems like a utopic objective, the quality improvement is addressed through a top-bottom perspective (guidelines for preparing the BIM models) with a bottom-up perspective (explicit checking at the BIM-MP level). It is possible that model-checkers (e.g. Solibri) can also be used to ensure modelling errors (geometric or otherwise) are detected. While the developments here focus primarily on using the data for BEPS, they also prove the concept that model checking can mostly be performed on the cloud. The development and integration of a generic rule-engine is considered out of scope, at the moment.

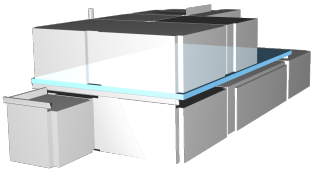

Figure 12: Application of BIM-MP's GED algorithm on KRIPIS building

Semantic enrichment

BIM-MP offers IFC data enrichment services, where certain data classes of input IFC files are populated with data which are obtained from existing IFC data classes. These enrichment services are offered to ensure that all IFC BIMs have the necessary information for the generation of BEPS models required for the assessment of building retrofitting projects. Two data enrichment services are offered by BIM-MP: Automatic Space Generation (ASG) and Common Boundary Intersection Projection (CBIP) services, both analyzed in the following sections.

Automatic Space Generation (ASG)

Frequently, the geometric representations of the internal building space volumes are missing or are defined incorrectly in the respective IFC BIM data classes. This happens because the design on an inner building space with BIM authoring tools is a tedious task involving filling all the space cavities (gaps), between the internal building space volume and its surrounding building architectural elements (walls, slabs, ...). These cavities can be too complicated, rendering the design of the inner building space impossible, even with the best BIM design software suites.

To overcome such difficulties, BIM-MP offers the Automatic Space Generation (ASG) service, which enriches an input IFC file with the geometric data of all building inner shells defined by the inner building space volumes.

According to Figure 13, ASG process receives as input the geometric content and its semantics exported in XML format by the IFC Geometry Exporter, of an initial IFC model IFC_0. Then, the building's space geometric representations in XML format produced by ASG algorithm, together with the initial model IFC_0 are used as input to the BIM-MP enrichment service to produce the final enriched IFC model IFC_1. ASG implements this IFC data enrichment by populating IfcSpace data classes which contain the boundary representations of the inner shells of the building spaces translated to the local coordinate systems of the respective spaces' level. ASG leaves no space gaps between the generated building space volumes and the surrounding building architectural elements.

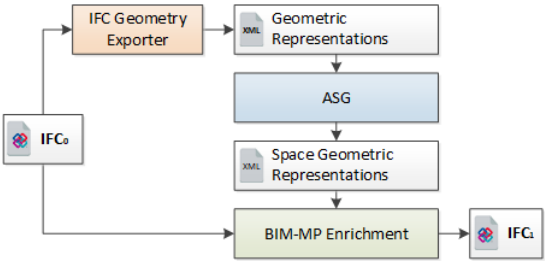

Figure 13: BIM-MP's ASG enrichment process

Initially, the ASG algorithm extracts the joint boundary surfaces among all the possible pairs of boundary representations of architectural elements of the building of interest, illustrated with blue color in part I Figure 14. Then, these joint boundary surfaces are subtracted from the respective boundary representations of the elements they belong to, yielding the remaining set of surfaces illustrated with green colors in part II of Figure 14.

The remaining surfaces of all architectural elements of a building form inner and outer surface shells. The obtained inner shells define the desired inner building space volumes. BIM-MP ASG service is applied on KRIPIS building as displayed in Figure 15.

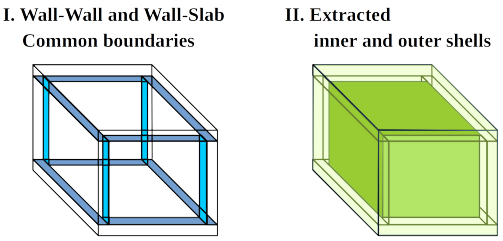

Figure 14: ASG algorithm demonstration

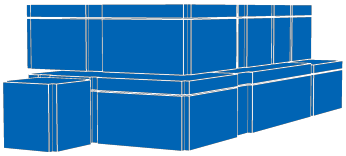

Figure 15: Application of BIM-MP's ASG algorithm on KRIPIS building

Common Boundary Intersection Projection (CBIP)

When the BIM files are generated using BIM authoring tools (Revit, ArchiCAD), necessary geometric content for the BEPS model generation (the building's second-level space boundary topology (Bazjanac 2010) and the external shading surfaces) may be either missing or incorrectly exported due to flaws in the IFC exporter of the BIM authoring tool. In this case, the generation of the second-level space boundary surface topology of the building and the population of the respective IFC data classes (IfcRelSpaceBoundary2ndLevel), are performed by BIM-MP's CBIP tool (Lilis et al. 2017).

According to Figure 16, CBIP process receives as input the geometric content and its semantics, of an initial IFC model IFC_0, exported in XML format by the IFC Geometry Exporter. Then, CBIP process, outputs: the building's boundary surface topology (second-level space boundaries and external shading surfaces) in XML format. CBIP's output together with the initial model IFC_0, are used as input to BIM-MP enrichment service to produce the final enriched IFC model IFC_1.

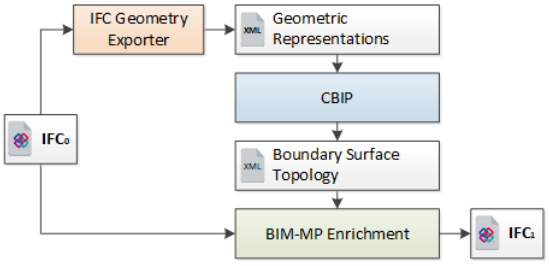

Figure 16: BIM-MP's CBIP service process

CBIP algorithm is also applied on BIMERR's KRIPIS pre-validation building and obtained second-level space boundary surfaces from this building are displayed in Figure 17.

Geometry Engine

The geometric data content of IFC files is not in a graphics compatible representation format to avoid verbosity and be as short as possible without losing critical information. To convert the IFC geometric

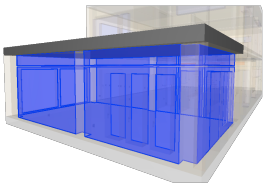

Figure 17: Application of BIM-MP's CBIP algorithm on KRIPIS building

data into a graphics compatible format, BIM-MP has a dedicated geometry engine which transforms all geometric representations of the architectural elements into boundary representations first using a B-rep generation process and finally into a graphics compatible format (OBJ, glTF) using a model viewer. These low-level geometric operations are performed using a dedicated C++ geometric library which based on clipper - one of the fastest and robust open-source freeware libraries for clipping and offsetting lines and polygons in two dimensions. The B-rep generation and the model viewer components are described in the following sections.

B-rep Generation

The B-rep generation (BRG) module of BIM-MP is responsible for transforming the various geometric representations of building elements existing in IFC data into correctly oriented boundary representations. This process is illustrated in Figure 18: BRG receives as input the geometric content and its semantics of an IFC model, exported in XML file by the IFC Geometry Exporter, and returns the B-reps of this content, in OBJ format.

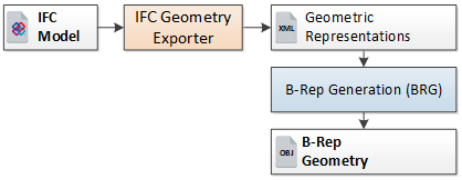

Figure 18: B-rep generation process

The BRG module outputs correctly oriented B-reps, which conform to the right-hand rule. BRG transforms to B-reps, multiple solid geometric representation types defined by the IfcProducDefinitionShape and its subclasses. Essentially, the BRG module performs the necessary complex geometric transformation operations, depending on the input solid representation, to produce the three-dimensional points of the surfaces of the boundary representations contained in the output OBJ file.

Model Viewer

The Model Viewer is a BIM model visualization component based on the open-source project xeogl (Xeolabs 2018). It uses WebGL for rendering 3D graphics natively within any compatible web browser. This component loads the geometry of a BIM model from glTF data (Robinet, 2014), data, allowing the BIM-MP user to navigate with the camera and to explore

the structural and non-structural elements of a building and their properties. The Model Viewer is written in JavaScript and deployed as part of the Core Module. It has access to the project repository to load and visualize the 3D geometric representation of the BIM model. The process is the following: The BIM-MP triggers the execution of the B-Rep Module automatically when the IFC data are available in the project repository. Next, the B-Rep Module generates and stores the OBJ data into the project repository. Finally, BIM-MP uses an open-source glTF converter (CesiumGS) to generate the glTF data. The operation of BIM-MP's model viewer is summarized in Figure 19.

Figure 19: Process diagram of BIM-MP's model viewer

ASSUMPTIONS AND RESTRICTIONS

BIM-MP operation has the following assumptions:

- BIM-MP's Core Module contains a 3D viewer with limited functionalities i.e. colour and texture support. The model navigation panel, as well as textures and colours of the building elements will be included in future versions.
- BIM-MP's REST API supports queries of IFC objects by type or GUID. Additional REST API endpoints such as updating, editing of IFC objects will be included in a future version of the BIM-MP.
- BIM-MP is fully compatible with IFC4, and supports only, non-curved IFC geometric representations. Some IFC curved geometric representations are approximated using segmentation processes.

All BIM-MP software components were developed by the authors using JAVA (BIM Library) and C++ (Geometric library). The only external dependence was the clipper program (Johnson 2016), for low level C++ geometric polygon operations in 2D.

CONCLUSIONS

An integrated cloud-based platform called BIM Management Platform (BIM-MP) is introduced and described in detail. BIM-MP is developed to handle OpenBIM files conforming to the IFC4 ISO standard and to prepare them for automatic generation of building energy performance simulation models. BIM-MP follows a SOA design approach, in which micro-services offering IFC data quality checking and data enrichment operations are orchestrated and executed in an asynchronous manner to ensure fast performance and reliability. Apart from its core module which performs GUI and data storing operations, BIM-MP offers five additional services grouped into three sub-categories: (a) two data quality services: an MVD-based data completeness checking service and a geometric error detection service, (b) two data enrichment services: the automatic space generation (ASG service) and the second-level space boundary topology generation service (CBIP service) and (c) a graphics engine for displaying IFC components. These operations are demonstrated on a simple office building.

ACKNOWLEDGEMENTS

The research leading to these results has been partially funded by the European Commission H2020 project "BIM-based holistic tools for Energy-driven Renovation of existing Residences" under contract #820621 (BIMERR).

References

Bazjanac, V. (2010), Space boundary requirements for modeling of building geometry for energy and other performance simulation, in 'CIB W78: 27th International Conference'.

Beetz, J., van Berlo, L., de Laat, R. & van den Helm, P. (2010), BIMserver. org–An open source IFC model server, in 'CIP W78 conference', p. 8.

buildingSMART (2020), Model View Definition (MVD) - An Introduction, `https://technical.buildingsmart.org/standards/ifc/mvd/`.

ISO, TC-59 & SC-13 (2018), ISO 16739-1: Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries — Part 1: Data schema, ISO.

Johnson, A. (2016), 'Clipper - an open source freeware library for clipping and offsetting lines and polygons', `http://www.angusj.com/delphi/clipper.php`.

Katsigarakis, K., Lilis, G. N., Giannakis, G. & Rovas, D. (2019), An IFC data preparation Workflow for building Energy Performance Simulation, in 'European Conference on Computing in Construction', EC3.

Lilis, G. N., Giannakis, G. & Rovas, D. (2015), Detection and semi-automatic correction of geometric inaccuracies in IFC files, in 'IBPSA Building Simulation Conference 2015', IBPSA, pp. 2182–2189.

Lilis, G. N., Giannakis, G. & Rovas, D. (2017), 'Automatic generation of second-level space boundary topology from ifc geometry inputs', Automation in Construction **76**, 108–124.

Lockley, S., Benghi, C. & Cerny, M. (2017), 'Xbim. essentials: a library for interoperable building information applications', The Journal of Open Source Software **2**(20), 473.

Xeolabs (2018), xeogl, `http://xeogl.org/`.