Technical University of Munich

Department of Civil, Geo and Environmental Engineering

Chair of Computational Modelling and Simulation

Versioning of Geometry Representation in BIM Models

Master Thesis

For the Master of Science Program Civil Engineering

Author: Ahmed Mohamed

Matriculation number: [redacted]

Professor: Prof. Dr.-Ing. André Borrmann

Supervisor University: Sebastian Esser (TUM)

Supervisor Company: Paulina Sala (ZM-I)

Date of issue: 01. April 2022

Date of submission: 27. September 2022

Abstract

BIM models are always subjected to different types of changes and modifications across project development phases. Starting from the early design until the handover. Normally, the elements of a BIM model change continuously due to several factors such as applying changes in the structural design, changes in the conceptual design, demand for higher LOD during various stages of the project, and many other factors making it difficult for the project coordinators to document and identify those changes.

The lack of version control between different versions of the BIM model can create a huge gap between different parties involved in the project regarding coordination and collaboration which leads to huge delays and extra costs in the end.

The aim of this research is to evaluate different approaches for the comparison of geometry stored in BIM models independent of the way the geometry is represented to apply the concept of version control so that, any modification regarding geometry between model versions can be reported and documented to enhance coordination tasks. Two approaches for geometry comparison will be discussed in this research from the concept and the implementation until the evaluation of results and limitations.

- First approach is based on assembling bounding boxes for elements and comparing them to identify the geometric changes encountered.
- Second approach is based on generating a triangulated mesh for elements to retrieve the list of vertices that describes the geometry of each element and then comparing them to specify the changes.

In the end, experiments were held on test models provided by Zilch + Müller Ingenieure GmbH to evaluate the accuracy of those approaches in terms of results and to identify the limitation of each approach.

Table of Content

List of Figures

List of Tables

List of Algorithms

List of Abbreviation

AEC	Architecture, Engineering, and Construction
BIM	Building Information Modeling
BREP	Boundary Representation
CAD	Computer Aided Design
CDE	Common Data Environment
CSG	Constructive Solid Geometry
DTM	Digital Terrain Model
IFC	Industry Foundation Classes
IGES	Initial Graphics Exchange Specification
ISO	International Organization for Standardization
LOD	Level of Development/Detail
STEP	Standard for The Exchange of Product Model Data
VCS	Version Control System
WPF	Windows Presentation Foundation

1 Introduction

1.1 Introduction

As the demand for applying Building Information Modeling (BIM) technology in the AEC industry increases, the need for effective coordination and exchange of data between engineers, architects, MEP specialists, and contractors increases. Efficient collaboration and communication can serve as good indicators to state whether a project was successful or not.

Currently, the AEC industry is evolving in the field of digitalization leading to an increase in the complexity and size of BIM models as well as the amount of information stored within each object in a model. Thus, it requires an accurate versioning concept to avoid the loss of data and information.

The lack of version control creates uncertainties in understanding the changes and modifications especially when users from different sectors work together on one project considering the fact that each project is unique. It is challenging to quantify the difference between two large-scale models manually as they might be composed of thousands of elements, this will only lead to huge delays and would be redundant as the models will be in continuous development. So, the implementation of version control mechanisms must be one of the necessary requirements for each project to improve collaboration and coordination.

Version control systems can observe different aspects of data sets. Perhaps, not all changes might be relevant for every stakeholder. Hence, changes regarding different aspects need to be transmitted to the responsible engineer. Currently, the thesis limits its scope to geometrical features stored in a BIM model. Additionally, changes in other aspects like changes in semantic information stored in such a model, however, can be observed and controlled by other existing techniques and software products.

Furthermore, geometry versioning is not a simple task as objects can be described by simple points and lines, up to complex three-dimensional shapes. So, it is challenging to find a proper method to be used as a general method in geometry comparison of objects that can achieve reliable results independent of the representation of geometry.

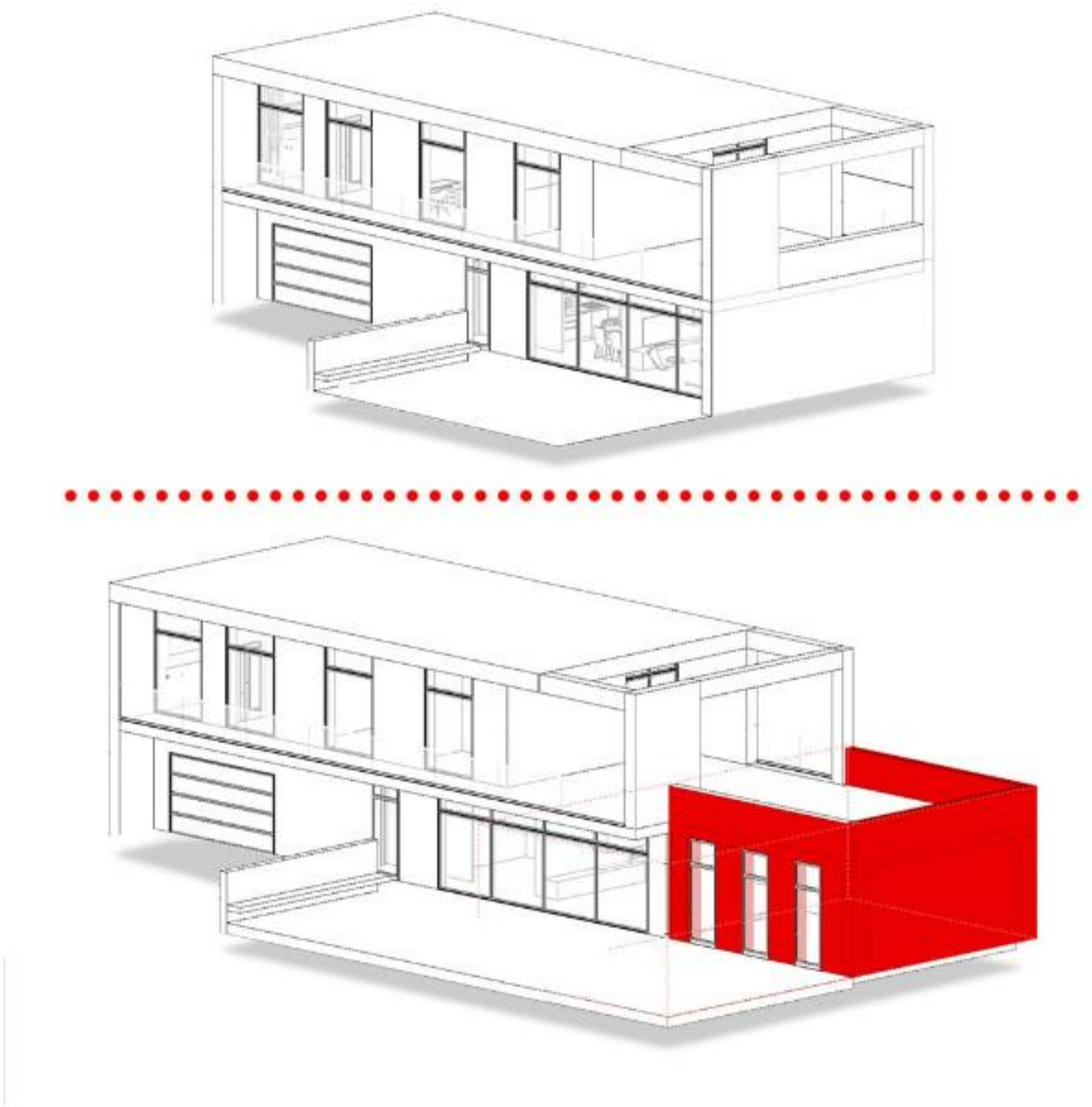


Figure 1 - Two versions of a model with changes (usBIM 2022).

Figure 1 sheds the light on the main issues that users are expecting to face while the model they are working on keeps developing across each phase without proper version control systems. Some users tend to compare two model versions visually, however, it is not a solution as, minor changes in geometry or properties of elements might not be clear and cannot be foreseen, a small change in the properties of elements can affect the quantity take off which is a sensitive process when considering budgeting a project.

1.2 Motivation

BIM models undergo many modifications during every stage of development, which raises the problem of monitoring and tracking changes within elements.

The concept of version control for geometric changes should evaluate the applied changes against a set of criteria to reason about their impact on other domain models or components. In this way, loss of information about the geometric development of elements will be avoided and coordination works can be performed without uncertainties.

The purpose of this research is to come up with classification criteria that is applicable for comparison of geometry between elements among versions of the BIM model so that any kind of geometric modification between instances of the model can be identified and documented.

1.3 Structure

Chapter 2 starts with a brief introduction to Building Information Modeling in addition to several aspects of BIM development. Then, the term IFC or Industry Foundation Classes is explained, defining the structure of IFC, and discussing its role in the collaboration phase of BIM models. Then, the main geometry representation types defined in IFC data models are mentioned with examples demonstrating the way they are represented. Afterward, the concept of version control is explained including an overview of some commercial software products that adopt the concept of version control.

Chapter 3 mentions the methodology of the paper in which it is stated how the experiments are being conducted to compare the results of the proposed approaches. Additionally, the chapter includes a detailed explanation of the two proposed approaches for geometry comparison from the concept and implementation to the advantages each approach can offer.

Chapter 4 starts with an overview of the tool that was created during the research explaining the interface, inputs, and outputs. Then, a brief description of the test models that are used in testing both approaches is given.

In chapter 5, a comparison between the results of the two introduced methods and the results from commercial software products is reviewed.

In chapter 6, a discussion about results is held to identify the limitations and weaknesses of each approach according to the results.

Chapter 7 concludes the manuscript with a summary of the research findings.

2 Background and Related Works

2.1 Building Information Modeling Overview

Complexity is appearing in built facilities when it comes to planning and construction. It is appearing due to the involvement of several stakeholders from diverse backgrounds as well as different expertise. Moreover, Information exchange between these several stakeholders in a fruitful way is necessary in order to achieve and sustain a successful construction project (Borrmann et al., 2018).

In fact, uncontrollable small design changes can create many kinds of errors if they are not tracked continuously with respect to the related plans, inconsistencies can arise and stay undiscovered until construction takes place where then approaches for solving the errors will be associated with excessive costs and huge delays. Furthermore, due to the limited scope of information resulting from the technical drawings, the building design was impacted negatively since it cannot be directly used for the purpose of analysis, calculation, and simulation (Borrmann et al., 2018).

This limited scope also increased the manual workload and the degree of errors because building design data resulting from technical drawings is subjected to be entered manually into the downstream application which reduces the efficiency of the overall project (Borrmann et al., 2018).

Additionally, it increases the workload on the building owners as they need to put a huge amount of effort to extract the needed information for operating the facility from the drawings and insert it again into the facility management system. So, essential information could be lost due to the high complexity of the information flow model and will consequently boost inefficiencies (Borrmann et al., 2018).

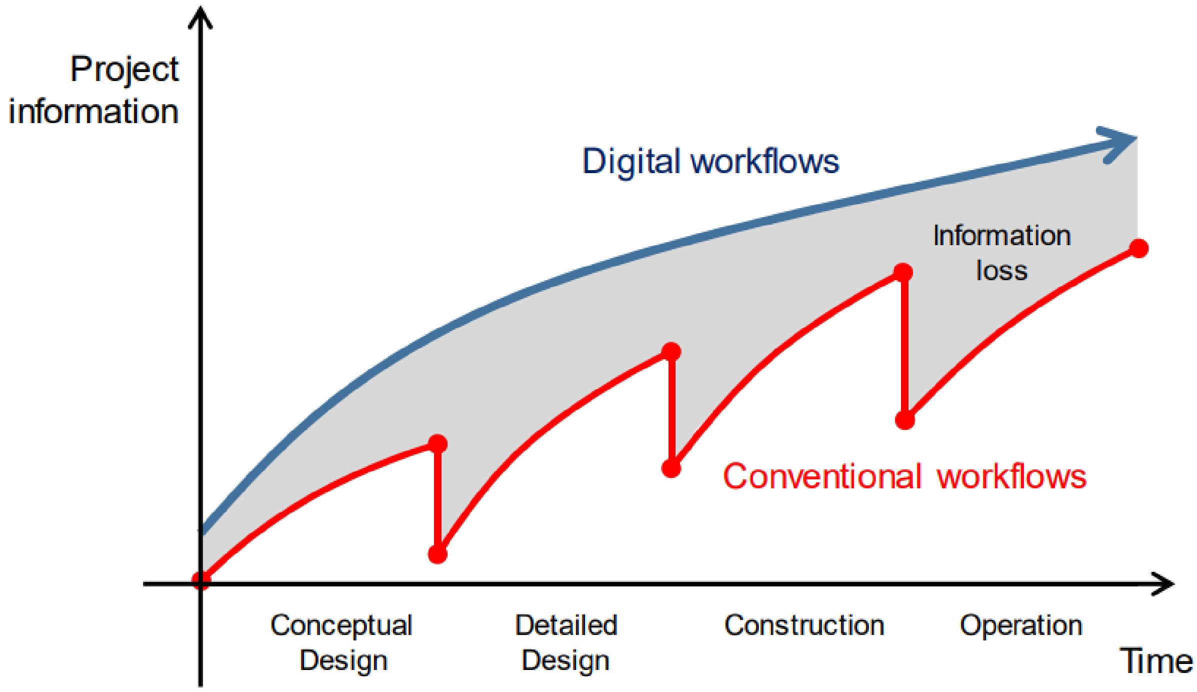


Figure 2 - Loss of information comparison between Digital and Conventional workflows (Borrmann et al., 2018)

Figure 2 demonstrates how digital workflows promote the exchange of information in a way that loss of information is minimized. This figure provides actual motivation for adopting Building Information Modeling Technology (BIM) which relies on the concept of digitalization for creating 3D digital models for designing, planning, and managing construction projects.

Building Information Modeling (BIM) is a methodology implemented for the cross-disciplinary design of buildings that are based on the management and creation of 3D models that store geometric information in addition to semantic information regarding each element. With the evolution of the AEC industry, the adoption of BIM has increased rapidly because it enhances the efficiency through the exchange of 3D models in the early design stages (Abdualdenien and Borrmann, 2019).

A Building Information Model is a digital representation of a facility that includes three-dimensional geometry of each component the building has at specific level of detail. Additionally, it also represents non-physical elements of a building like zones, schedules, or structure of the project. Besides the geometry, each element is associated with a group of semantic information as well like materials, types, technical parameters as well as different logical relationships between elements as shown in figure 3 (Borrmann et al., 2018).

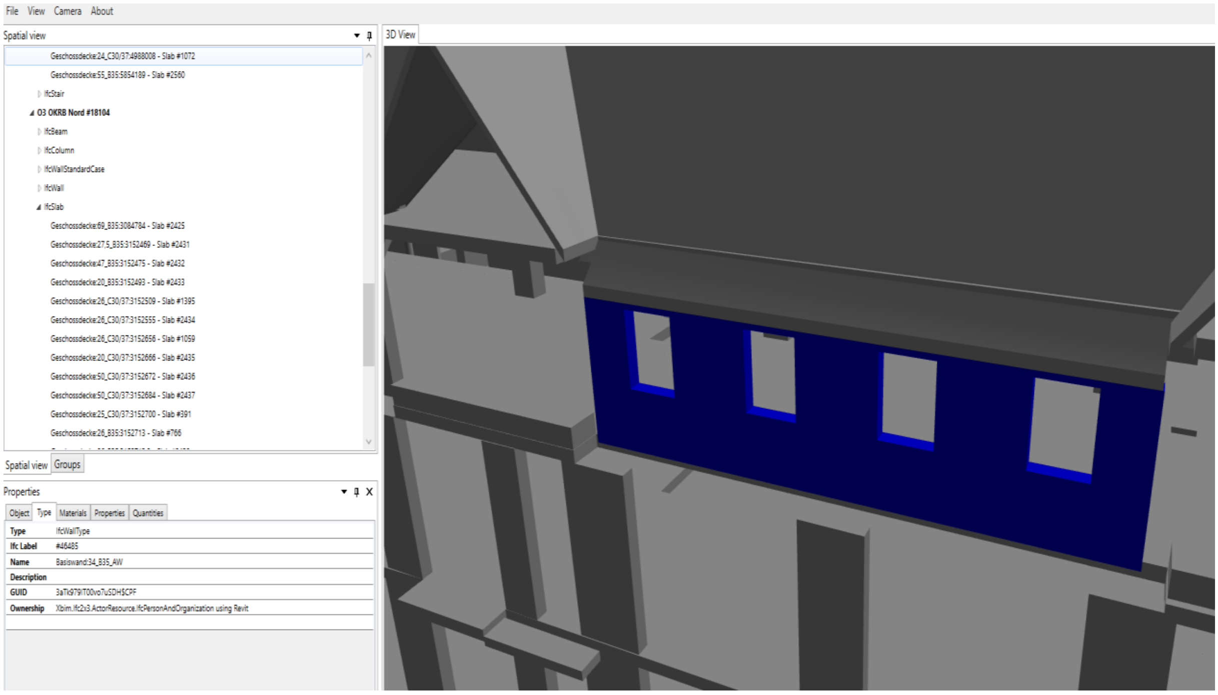


Figure 3 - BIM model showing both geometry and semantic information of building element

The transition between shifting from drawing-based workflows to model-based workflows needs a huge number of changes regarding internal and cross-company workflows. In order to avoid conflict in the functioning of workflows, a transition step is needed. Accordingly, two main concepts in the BIM spectrum are defined, known as BIM Maturity and Level of development/detail (LOD).

BIM maturity levels refer to the technological progress obtained according to the extent of collaboration and information exchange between different stakeholders working on a project (Pillay, Musonda and Makabate, 2018).

In 2008 Bew and Richards developed the BIM maturity level model which become one of the main components of the implementation strategy in the UK (Succar, 2015).

Furthermore, the BIM maturity model is composed of four levels, and they are explained as follows according to Dakhil and Underwood (2015):

- BIM level 0: In this level, information is produced via CAD drawings, but model information is not shared.
- BIM level 1: In this level, a transition is taking place from CAD information to 3D one. Additionally, models created are not shared among different stakeholders.
- BIM level 2: In this level, two new dimensions are added which are the 4D, Schedules, and the 5D, Budget estimation. Although, this level focuses on the

collaboration of work, users might be working on different 3D models but in the end, models are shared via a common file type to allow for collaboration. In this way, all users involved in the organization are up to date regarding the information available and have the possibility to edit it accordingly.

- BIM level 3: In this level, the main target is to achieve full integration of available information in a cloud-based environment. In this way, the life cycle of a facility from the early design stage to the construction and maintenance can be efficiently managed.

Figure 4 represents the maturity levels graphically.

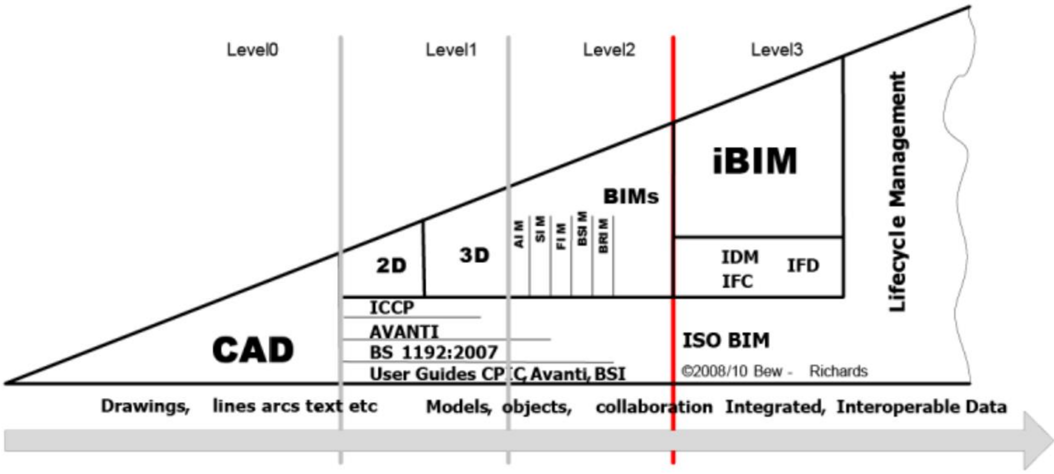


Figure 4 - BIM Maturity Level Model by Bew & Richards (2008).

Level of Development (LOD) is the level at which the model has been created with respect to the information stored in it (Mekawy and Petzold, 2018). LOD levels are described in figure 5.

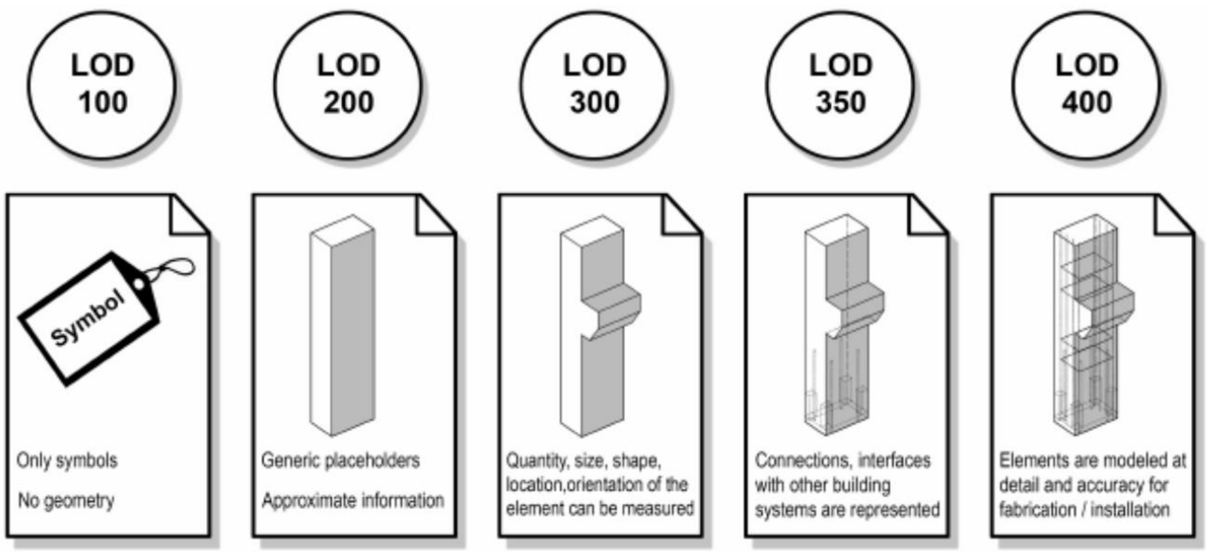


Figure 5 - LOD levels overview (BIMForum, 2016).

Additionally, two essential terms “BIG BIM” and “little BIM” are introduced to aid in understanding the BIM implementation process.

Moreover, little BIM describes the use of BIM software specifically by an individual user to perform a specific task like, creating a building model using software and then extracting drawings from it. The model is not used by different software solutions and is not submitted to other stakeholders. So, external communications take place only using drawings (Borrmann et al., 2018).

On the other hand, BIG BIM allows for comprehensive collaboration between stakeholders through model-based communication across the lifecycle of the project. A wide variety of technologies supports coordination works such as databases, servers for models, or project platforms (Borrmann et al., 2018).

In the context of BIM usage, the question of whether software solutions from one vendor are used “Closed BIM” or open vendor-neutral formats are used to allow for exchange of data and information between software products of different software vendors “Open BIM” as demonstrated in figure 6. Even though some software companies provide wide libraries of software products that performs many tasks regarding designing and operating a facility, there will be a need to exchange data with other solutions from other companies that might be used by other stakeholders (Borrmann et al., 2018).

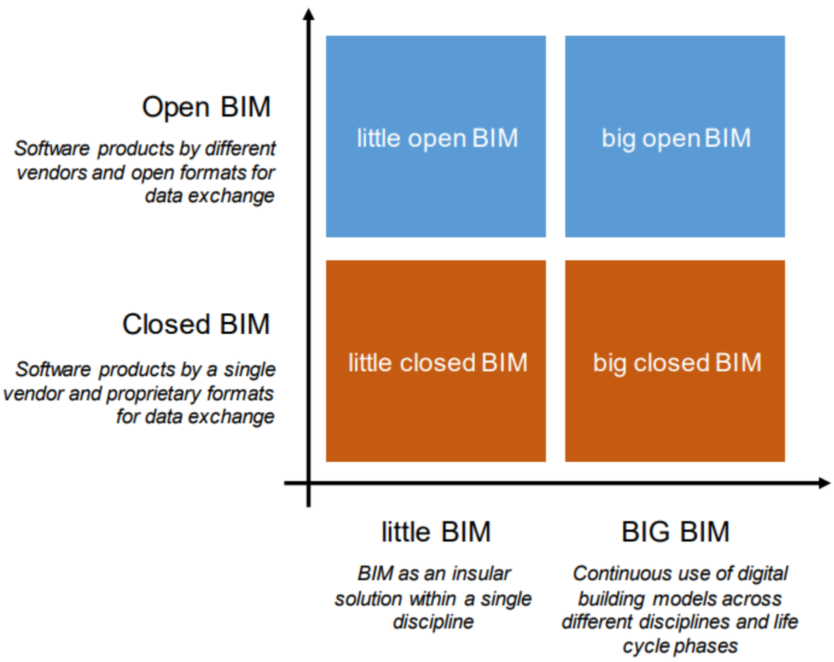


Figure 6 - The terms "BIG BIM", "little BIM", "Open BIM" and "Closed BIM" (Borrmann et al., 2018).

To overcome the main challenges associated with loss of data and enhance exchange of data between various software products in the AEC sector, some of the software vendors, public authorities, and users across the world created “the International Alliance for Interoperability” in 1994. The name was changed in 2003 to *buildingSMART* for marketing issues. The non-profit international organization had success in creating a vendor-independent data format that has the capability of exchanging digital building models. Industry Foundation Classes (IFC) was the resulting object-oriented data model that provides data structures that can cover almost every aspect of built facilities. The data format now forms the basis for several national standards and guidelines that support the implementation and adoption of Open BIM (Borrmann et al., 2018).

2.2 Industry Foundation Classes (IFC)

2.2.1 History of IFC

In 1970, functions and methods for data exchange among different CAD tools started to be implemented as the demand by major groups for a loss-free data exchange interface increased. These initial prototypes were limited to only support the exchange of geometric data with no support for the exchange of semantic data such as IGES which stands for Initial Graphics Exchange Specifications. Afterward, with more research conducted to enhance the standardization process, the basics of Standard for the Exchange of Product Model Data (STEP) were introduced (Borrmann, Beetz, Koch and Liebich, 2018). STEP is supported by many countries with researchers involved in the development of engineering product standards for data exchange (Liu, Lovett, Godwin and Fletcher, 2002).

Since its start in 2005, buildingSMART evolved rapidly across the world as more than 800 organizations, institutes and companies are now official members of the organization and are holding the responsibility to create, develop and maintain standards for the industry. The first version of Industry foundation classes (1.0) was invented in 1997. Moreover, version 1.5.1 was the first version that was established to be used by dedicated software applications supporting the construction industry (Laasko and Kiviniemi, 2012).

The first version was followed by many extensions and revisions, which were adopted and implemented by different software vendors in their software products. Figure 7 summarizes the development of IFC versions across time. Unlike other popular vendor-specific formats like Autodesk's DWG format, IFC data models are available with no licensing fees which was a motivating factor for several software products to adopt the IFC model. Currently, version 2x3 is the most used version across the globe but it is currently being replaced gradually by version 4 (Borrmann, Beetz, Koch and Liebich, 2018).

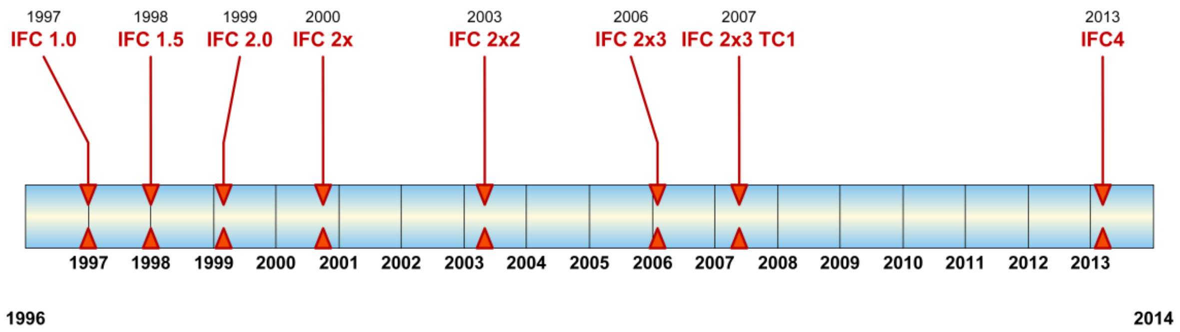


Figure 7 - Version development of IFC format (Borrmann, Beetz, Koch and Liebich, 2018).

In recent years, IFC has become the basic format for implementing Open BIM, many countries adopted the IFC format as a main data exchange format for the coordination of digital models in the construction industry. IFC is already supported by many BIM software products due to the neutrality of the format. Hence, it became the basis of most of the sectors that prescribe the implementation of BIM for building facilities (Borrmann, Beetz, Koch and Liebich, 2018).

2.2.2 EXPRESS as data modeling language for IFC

EXPRESS as a language for data modeling is mentioned in part 11 of ISO standard (10303, 2004) as a standard for exchange of product manufacturing data. It is a language that was specifically designed to represent product data through constraints and schemas. The language is based on object-oriented data models which means that it can apply abstraction of objects into several classes that share relations with each other and can have attributes as well (Borrmann, Beetz, Koch and Liebich, 2018).

EXPRESS adopts the idea of creating an entity type which is the same as classes in object-oriented theory. Relationships of each entity type can be defined to other entity types, also the same holds for attributes.

The concept of inheritance is a basic concept in the standard of EXPRESS so that attributes and relationships can be applied to sub-types, a relationship between two objects can either be defined as a direct relationship or an inverse relationship which is a special kind of relationship in EXPRESS standard (Borrmann, Beetz, Koch and Liebich, 2018).

Defining relationships between groups of objects is possible as the language adopts a wide range of aggregation types such as:

- List: Collection of objects with specific order
- Array: Collection of objects with fixed size and order
- Bag: Collection of objects with allowed duplication and no order
- Set: Collection of objects with no duplication allowed and no order

Typically, sets and lists are the most aggregation types used across IFC data models (buildingSMART).

EXPRESS also offers a block called WHERE which is an optional block to define algorithmic conditions. Furthermore, EXPRESS uses the enumerations to assign fixed values from predefined selection to attributes that can only hold specific values (Borrmann, Beetz, Koch and Liebich, 2018). Algorithm 1 shows the typical description of an entity by EXPRESS.

```
ENTITY IfcMaterialLayerSet;
       SetName          : OPTIONAL IfcString;
       Offsets          : LIST [1:?] OF IfcLengthMeasure;
       Thicknesses      : LIST [1:?] OF IfcLengthMeasure;
       Materials        : LIST [1:?] OF IfcMaterial;
   WHERE
       WR1 : (HIINDEX(SELF.Offsets) = HIINDEX(SELF.Thicknesses)) AND
             (HIINDEX(SELF.Thicknesses) = HIINDEX(SELF.Materials));
END_ENTITY;
```

Algorithm 1 - Entity description according to EXPRESS language (BuildingSMART, 2022)

The textual notation is not the only possibility to represent description of entities, EXPRESS offers graphical notation as well for the description of entities and highlighting relationships among them. This notation is known as EXPRESS-G.

2.2.3 IFC Layers

As result of the numerous amounts of entities and complex relationships inside the IFC structure, it is divided into 4 main layers:

- Core layer
- Interoperability layer
- Domain layer
- Resource layer

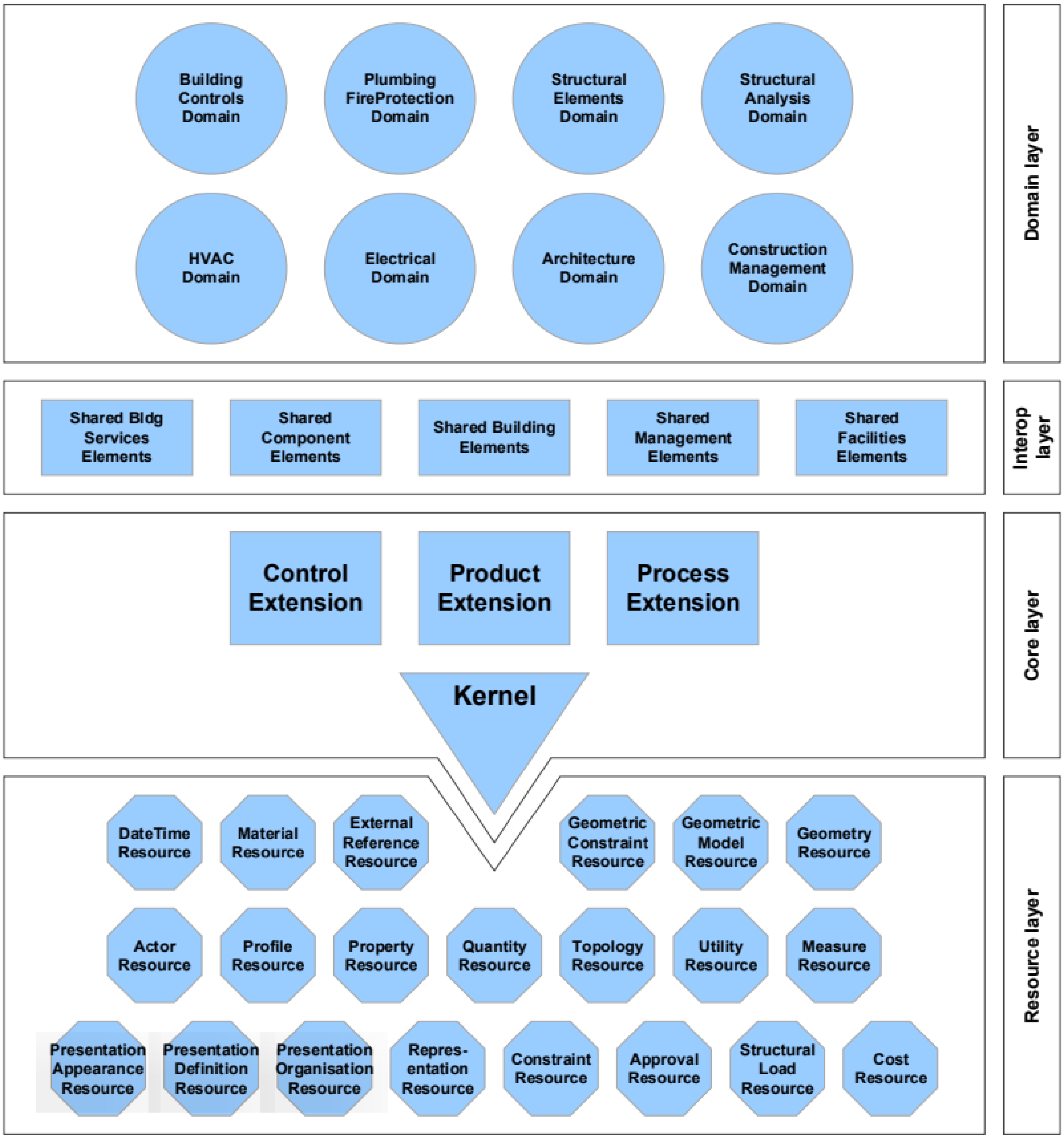


Figure 8 - IFC layers (buildingSMART).

The relationships between these layers are demonstrated in figure 8. These four main layers share strict referencing among each other, meaning that referencing is only possible downwards while referencing upwards in the hierarchy is not allowed. For example, classes inside the resource layer need to be independent as it cannot reference classes above them. On the other hand, all the remaining layers can reference from the resource layer (Laasko and Kiviniemi, 2012).

Core Layer

The Core Layer includes the main abstract classes of the IFC data model such as *IfcProject, IfcRoot, IfcObject, IfcProcess, IfcRelatioship, and IfcProduct*. These classes identify the structures, general concepts, and relationships that can be used and referenced by the layers above. The Kernel module is defined in the Core Layer, this module is responsible for providing relationships, attributes, roles, and concepts regarding objects. Generally, every entity defined in this layer or layers above has a global unique id as a required attribute while other attributes like owner or history information are optional attributes. Additionally, the Core Layer offers three more extension modules which are Control Extension, Product Extension, and Process Extension (Borrmann, Beetz, Koch and Liebich, 2018).

- Control Extension: Provides classes for control of objects and allocating them to spatial and physical elements such as *IfcControl.*
- Product Extension: Provides classes for defining the physical and non-physical elements of the building model such as *IfcElement, IfcBuildingElement, IfcOpeningElement, IfcBuilding, IfcBuildingStorey, IfcSpace, IfcSite* in addition to other classes responsible for the declaration of relationships between objects such as *IfcRelVoidsElement.*
- Process Extension: Provides classes for the description of processes and operations.

Interoperability Layer

The Interoperability Layer is located above the Core Layer and it contains many classes that are defined in *IfcProductExtension* schema to help in increasing the details level of the information represented for different entities (Noardo, Ohori, Krijnen and Stoter, 2021).

Furthermore, the Interoperability Layer which is also known as Shared Layer acts as an intermediate layer between the domain schemas and the core of the data model.

Hence, it includes classes which are partially derived from classes found in the Core Layer. For example, the main classes for building elements such as *IfcSlab, IfcColumn, and IfcBeam* (Borrmann, Beetz, Koch and Liebich, 2018).

Domain Layer

This layer in general provides specialized particular classes that are only limited to be applied by a particular domain. It is implemented to be found at the highest layer in the hierarchical structure which includes the definition of entities of a specific process, product, or resources. These objects are used for sharing of information and intra-domain exchange. Generally, the classes defined in this layer cannot be referenced by any other layer. Currently, IFC4 has additional defined domain than those listed in figure 8 (buildingSMART, 2021, Borrmann, Beetz, Koch and Liebich, 2018).

Resource Layer

The Resource layer is at the bottom level and is composed of schemes that identify the fundamental data structures which can be used throughout the whole data model. Unlike classes included in other layers, the entities of this are not derived from *IfcRoot.* Hence, they have to be referenced by elements that instantiate from *IfcRoot class* (Borrmann, Beetz, Koch and Liebich, 2018). Examples of resource schemes are as follow:

- Geometric Model Resource: Provides classes for the description of geometric models such as *IfcSweptAreaSolid, IfcCsgSolid.*
- Geometry Resource: Provides geometric elements such as curves, points, swept surfaces.
- Topology Resource: Provides classes for describing and representing the topology of a solid mass.
- Utility Resource: Provides elements that describe version history and ownership of objects in IFC.

Moreover, this layer provides more schemes such as Measure, Representation, Cost, and Material.

2.2.4 Inheritance hierarchy

Inheritance hierarchy plays an essential role in defining both generalization and specialization relationships in IFC data models. It is the basis for identifying which attributes of which entities can be accessed by other entities. (Borrmann, Beetz, Koch and Liebich, 2018).

Figure 9 highlight the most important classes which are found in the upper layers of the inheritance hierarchy. Moreover, several classes from this figure that are of high importance in the IFC data models will be clarified such as *IfcRoot, IfcObject, and IfcProduct.*

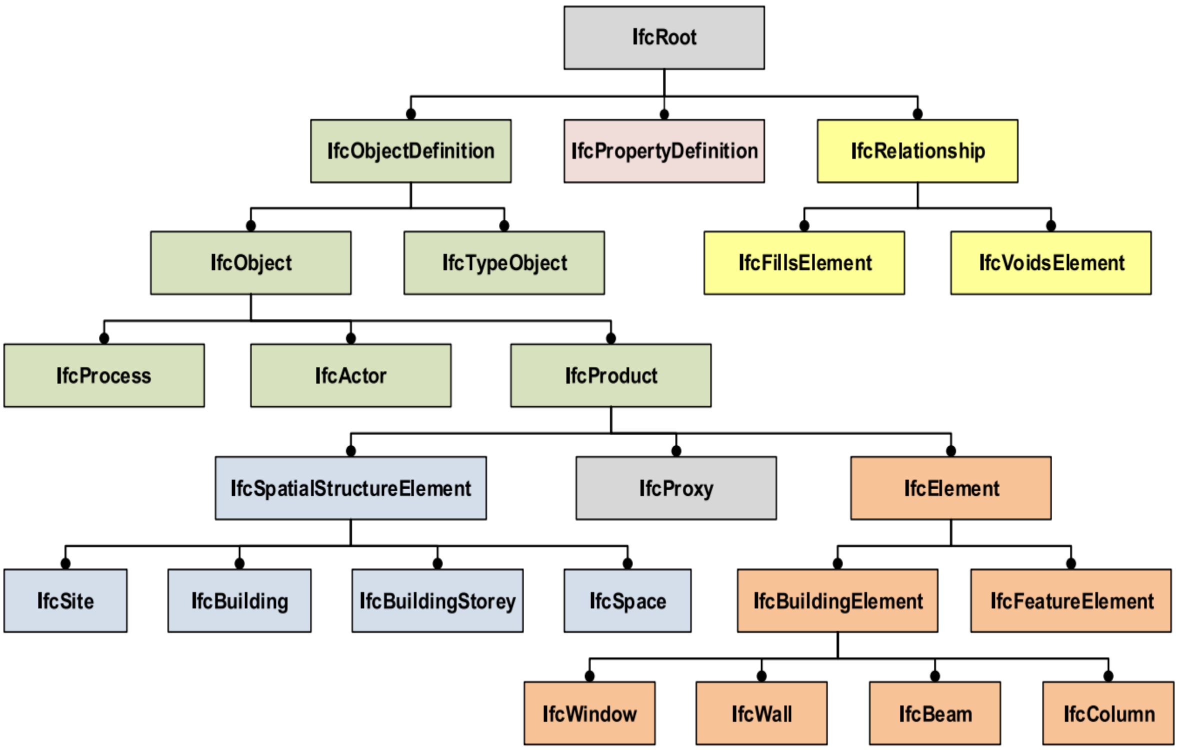


Figure 9 - Part of the inheritance hierarchy of the IFC demonstrating the most important classes and their relation (Borrmann, 2018).

IfcRoot Class

The inheritance tree starts with the abstract class *IfcRoot.* Normally, all classes must be derived either directly or indirectly from *IfcRoot* except the classes found in the resource layer as they cannot reference from other layers. This class provides helpful attributes for describing objects such as Globally Unique Identifier (GUID) which helps in identifying ownership of an object to track the history of changes through the development of an object. Furthermore, three classes are directly derived from *IfcRoot* which are *IfcRelationship, IfcPropertyDefinition, and IfcObjectDefinition* (Borrmann, Beetz, Koch and Liebich, 2018).

- *IfcObjectDefinition* class is a superclass comprising all classes that ae used to describe and define physical (e.g., columns, beams), non-physical (e.g., site, spaces), or conceptual objects (e.g., costs).
- *IfcPropertyDefinition* class is responsible for defining properties that are not already part of the data model.
- *IfcRelationship* class comprises subclasses that have a specific role in describing relationships among objects.

IfcObject Class

The class *IfcObject* is a subclass of the superclass *IfcObjectDefinition* and is used to represent an object of a building project, six important classes are subclasses of this class (Borrmann, Beetz, Koch and Liebich, 2018).

- *IfcActor:* Represent participant in the building project.
- *IfcControl:* An object which controls another object.
- *IfcGroup:* Represent aggregation of objects.
- *IfcProcess:* Describe any process that can be encountered in a project (e.g. construction, planning).
- *IfcProduct:* Describe physical or spatial objects. Geometric shape representation can be assigned to *IfcProduct* objects.
- *IfcResource:* Identify object used as part of a process.

IfcProduct Class

The class *IfcProduct* is the basis for representing physical or spatial objects. Typically, all classes that are used to define or describe virtual models are derived from *IfcProduct.* As mentioned earlier, objects from this class can be assigned different types of geometric shape representation and position as well. This class includes *IfcElement* which is a superclass that comprises essential classes that is necessary for describing building elements like *IfcBuildingElement* class, all elements that are available in a built facility are defined within the *IfcBuildingElement* superclass as subclasses such as *IfcColumn*, *IfcSlab*, *IfcBeam* etc (Borrmann, Beetz, Koch and Liebich, 2018). Figure 10 shows the classes derived from *IfcBuildingElement* superclass.

Moreover, the *IfcSpatialElement* class has a specific role in describing spatial objects. Subclasses of this class include *IfcBuilding, IfcBuildingStorey, IfcSite, and IfcSpace* which are classes frequently found in any IFC data model. Additionally, *IfcProxy* which is also a subclass of *IfcProduct* is used to describe objects that do not match any of the semantic types defined in the IFC model (Borrmann, Beetz, Koch and Liebich, 2018).

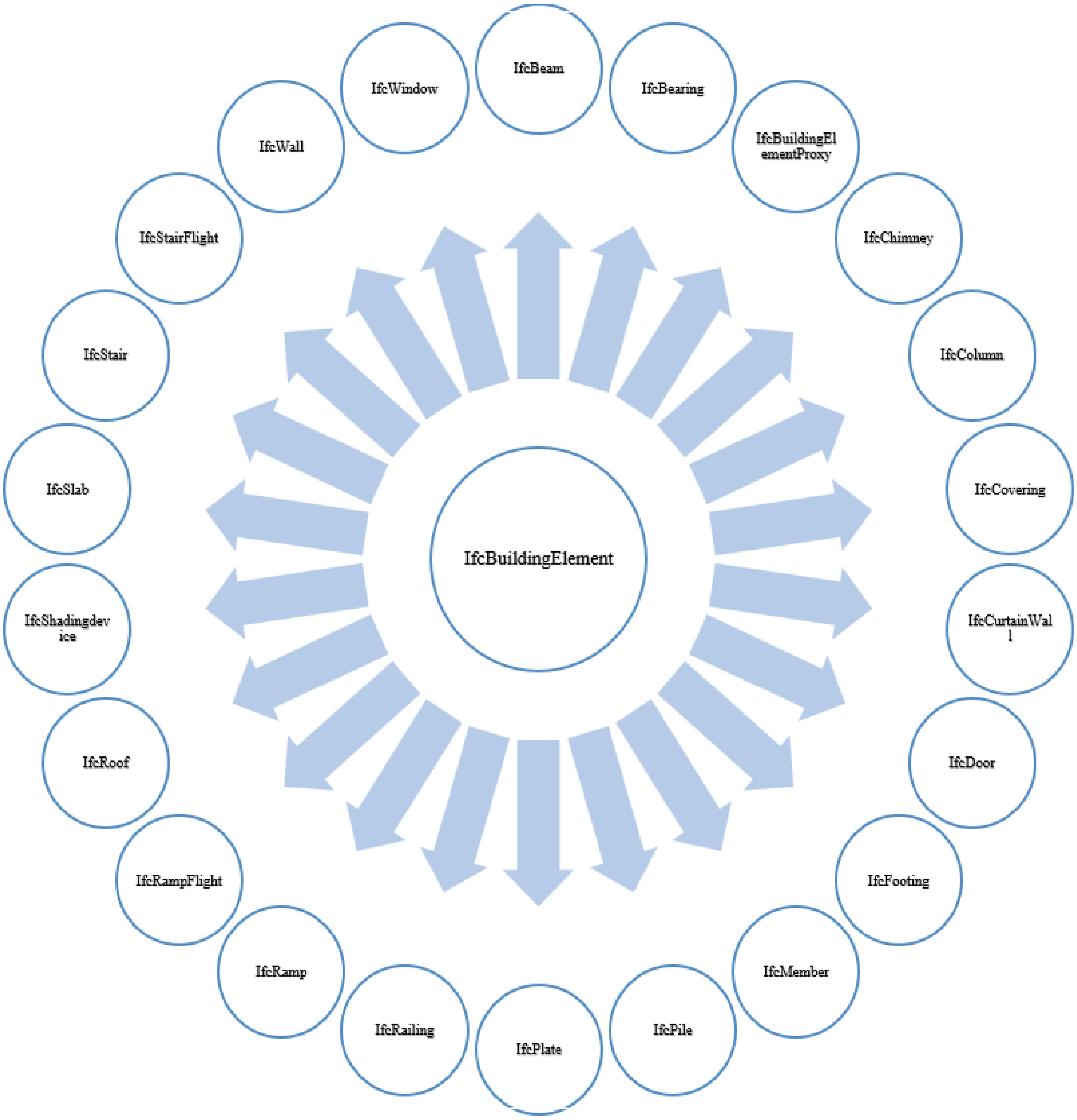


Figure 10 - Classes derived from IfcBuildingElement superclass.

2.2.5 Geometry Representation

Geometric description has a specific division when compared to the semantic description in IFC data models. All objects within a model are defined as a semantic identity and then linking with geometric representations takes place. Hence, identity is only applied to semantic objects (Borrmann, Beetz, Koch and Liebich, 2018). Figure 11 shows the division between the semantic structure and the description of geometry.

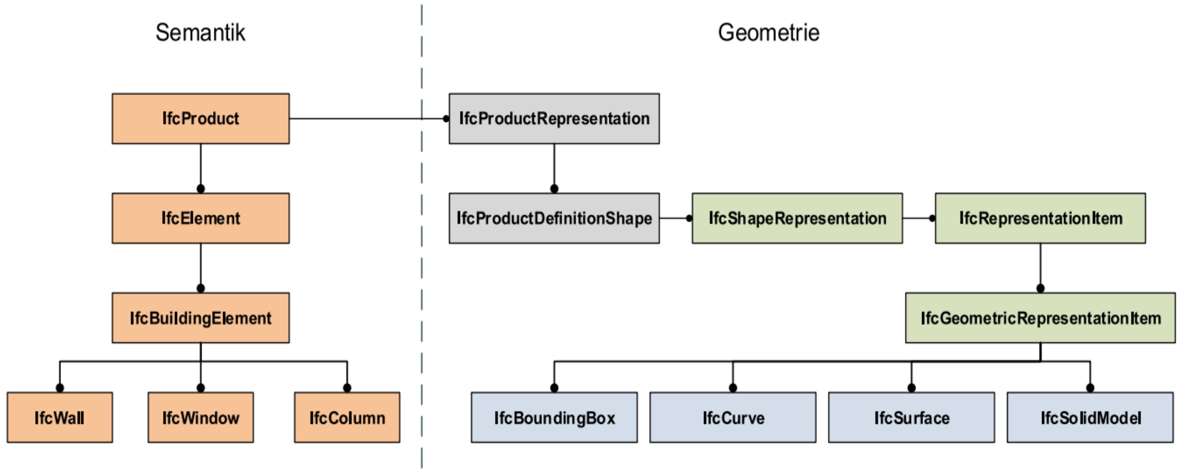


Figure 11 - Division between semantic structure and description of geometry (Borrmann, Beetz, Koch and Liebich, 2018).

Linking geometry representations with objects raise the need for geometric representations for various applications. Currently, modern tools that support BIM modeling demand high quality when using boundary representation (Brep) or constructive solid geometry (CSG) for the description of objects in order to apply modifications to the model. Furthermore, IFC provides a wide range of geometric representation types which inherit from the superclass *IfcGeometricRepresentationItem.* Basically, it contains classes for the description of solids "*IfcSolidModel*", classes for representing surfaces "*IfcSurface*" and classes for describing curves "*IfcCurve*" (Borrmann, Beetz, Koch and Liebich, 2018).

Since a lot of classes for different geometry representation types are defined in the IFC documentation, this chapter will focus on the most important classes for the description of geometry by discussing their concept with the help of some examples.

2.2.5.1 Solid Modeling

Modeling 3D solid objects is supported by IFC data model in different ways. The superclass *IfcSolidModel* is the basis for multiple subclasses, each class comprises subclasses that are applicable to certain cases for the representation of elements. The main classes derived from *IfcSolidModel* are as follows:

- *IfcManifoldSolidBrep*
- *IfcCsgSolid*
- *IfcSweptAreaSolid*

Boundary Representation

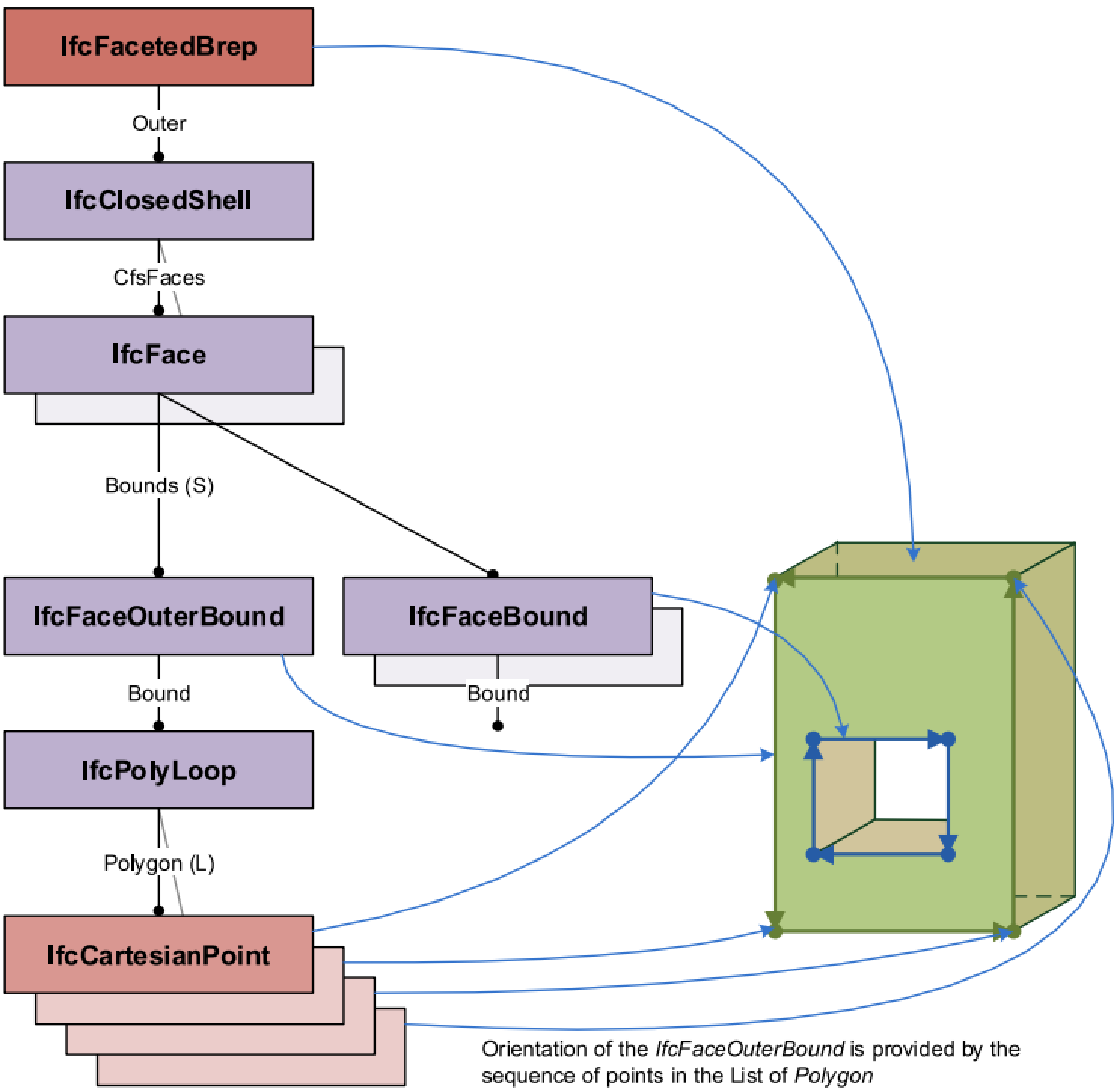


Figure 12 - Example demonstrating the use of IfcFacetedBrep (buildingSMART).

The Boundary Representation (Brep) is considered to be the most powerful approach for modeling solids. *IfcFacetedBrep* and *IfcAdvancedBrep* are the two main subclasses derived from *IfcManifoldSolidBrep* (Borrmann, Beetz, Koch and Liebich, 2018). Moreover, the major difference between those two subclasses is the capability of modeling surfaces as *IfcFacetedBrep* can only model flat surfaces while *IfcAdvancedBrep* offers the possibility to model surfaces with curves along the edges. However, both types do not support voiding in geometric objects, as they are only limited to describe shells. Two variations of the classes mentioned before are implemented to extend the use of Brep to cover cases with voids, these classes are known as *IfcFacetedBrepWithVoids* and *IfcAdvancedBrepWithVoids* (Borrmann, Beetz, Koch and Liebich, 2018). Figure 12 shows an example that demonstrates the use of *IfcFacetedBrep* for boundary representation.

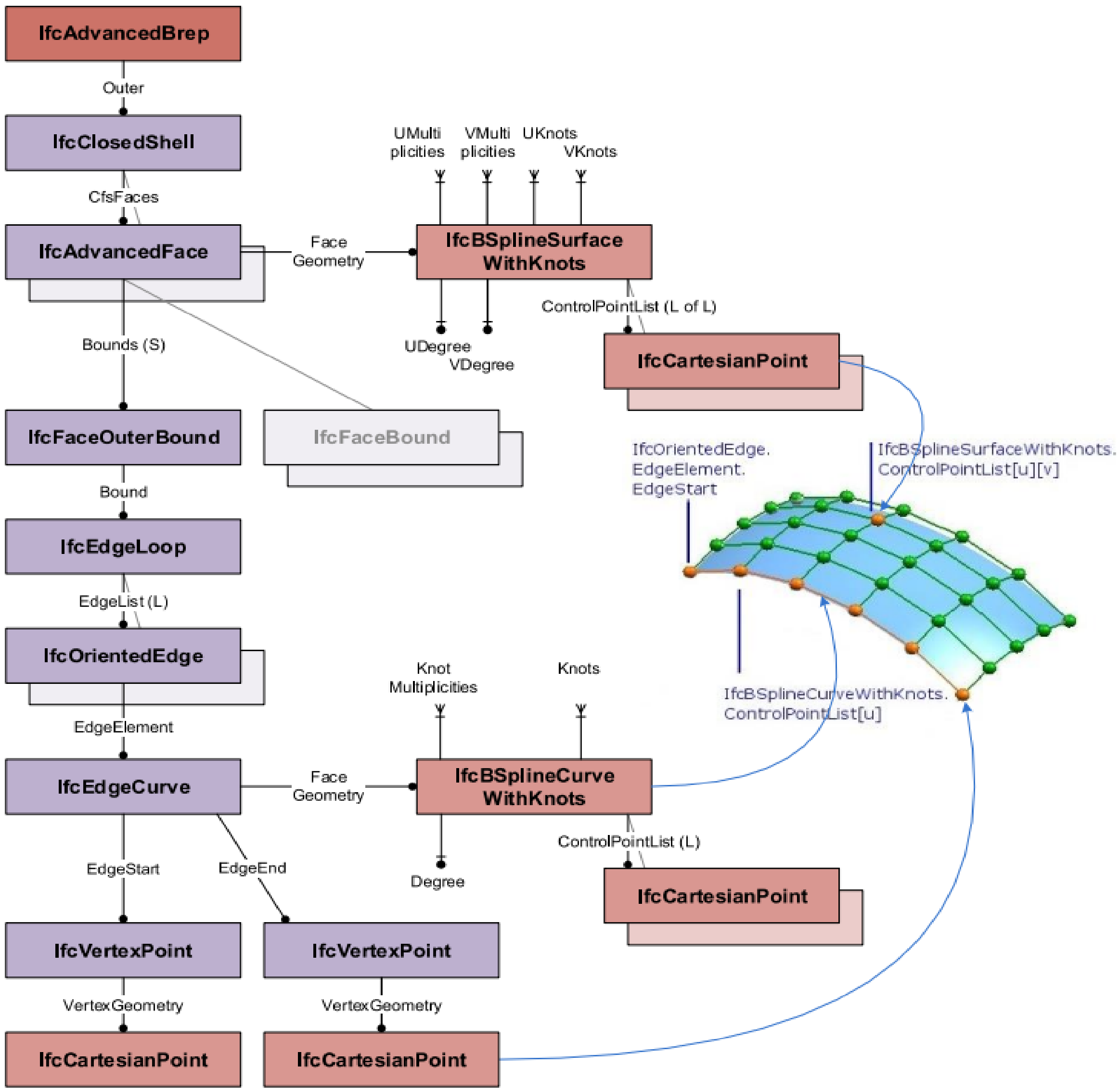


Figure 13 - Example demonstrates the use of IfcAdvancedBrep (buildingSMART).

It is shown in figure 13 that the class *IfcFacetedBrep* refers to *IfcClosedShell* class which consists of a set of faces described through the *IfcFace* class, each face described in this set can acquire any numerous amounts of bounding surfaces defined by the class *IfcFaceBound,* which in turn refers to an object that includes a list of points that define the vertices of the solid known as *IfcLoop.*

A data structure for describing solids with curves extends this basic topological data structure with elements for modeling the geometric sequence of surfaces and edges. The basis for this is the *IfcAdvancedBrep* class. As above, this is linked to the *IfcClosedShell* object and references a surface object of type *IfcAdvancedFace*. In contrast to his *IfcFace* objects above, these objects contain explicit geometric descriptions. This can be described as a NURBS surface modeled as *IfcBSplineSurface*. An object of this class must reference the corresponding control point and provide all parameters necessary for the description of a NURBS surface (Borrmann, Beetz, Koch and Liebich, 2018). Figure 13 shows the data structure for modeling a curved surface using *IfcAdvancedBrep* class.

Constructive Solid Geometry and Clipping

The Constructive geometry (CSG) approach is modeling solids by combining predefined fixed solid objects (primitives) using Boolean operations. Operations like union, intersection, and difference. The IFC data model provides the *IfcCsgPrimitive3D* class along with its subclasses *IfcBlock*, *IfcRectangularPyramid*, *IfcRightCircularCone*, *IfcRightCircularCylinder*, and *IfcSphere*. The *IfcBooleanResult* class is employed to model the results of combinatorial operations. This class provides an operator attribute that may have one in all three values: UNION, INTERSECTION, or DIFFERENCE, together with FirstOperand and SecondOperand attributes that refer to the 2 operands. The operand is of type *IfcSolidModel*, *IfcHalfSpaceSolid*, *IfcCsgPrimitive3D*, or *IfcBooleanResult*. CSG models are exclusively described by the latter two classes. Users are able to use the *IfcBooleanResult* class recursively to define a tree-like structure. Figure 14 demonstrates the concept of solid modeling using CSG primitives with Boolean operators (Borrmann, Beetz, Koch and Liebich, 2018).

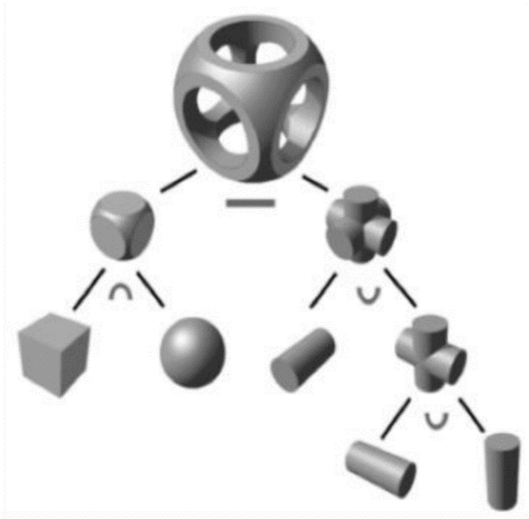


Figure 14 - Showing the concept of solid modeling using CSG approach (Zhu et al., 2020).

Clipping approach is taken into account to be a special kind of CSG, as only the Boolean difference operation is involved. As a matter of fact, several studies simply remarked clipping as CSG. However, IFC explicitly differentiates them from one another, as two distinctive classes are defined, i.e., *IfcBooleanClippingResult* and *IfcCsgSolid* (Zhu et al., 2020).

Clipping is often used in modeling elements that are cut off by a plane. the primary operand is usually a volumetric solid (*IfcSolidModel*) and the second operand may be a so-called half-space solid (*IfcHalfSpaceSolid*), that's defined along a plane and in one direction. Clippings can occur anywhere as a node in an exceedingly CSG tree (Borrmann, Beetz, Koch and Liebich, 2018). Figure 15 shows examples of elements modeled by the clipping approach.

Extrusion and Swept Solids

Various ways of 3D solid modeling are provided by the IFC model by means of rotation or extrusion of a profile defined in a two-dimensional perspective through the superclass *IfcSweptAreaSolid* and its subclasses *IfcRevolvedAreaSolid*, *IfcExtrudedAreaSolid*, *IfcFixedReferenceSweptAreaSolid*, and *IfcSurfaceCurveSweptAreaSolid*. additionally, there's also the class *IfcSweptDiskSolid*, which inherits directly from *IfcSolidModel*. the idea for every operation is that the definition of a profile within the kind of an *IfcProfileDef* object referenced by the SweptArea attribute.

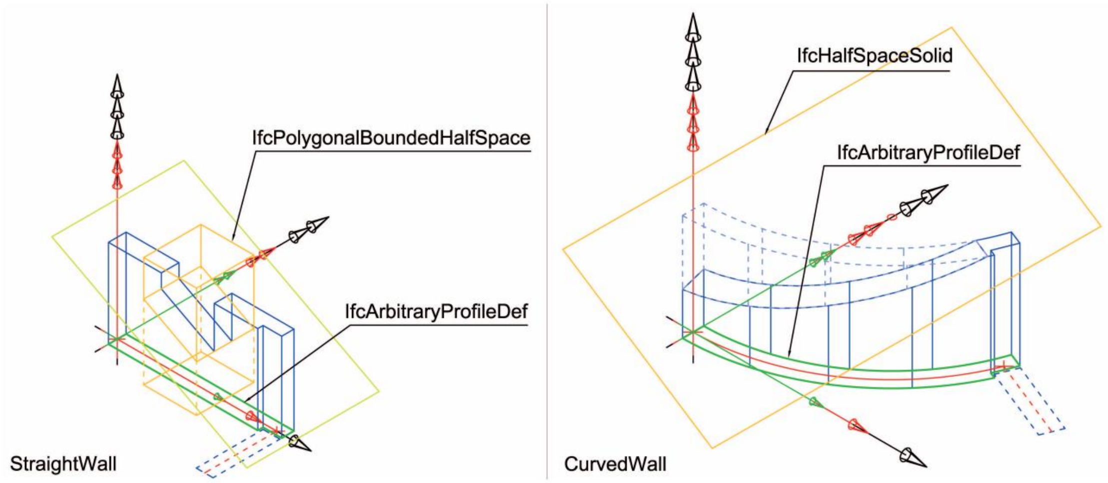


Figure 15 - Examples showing the use of clipping approach for modeling (buildingSMART).

The foremost common subclass of *IfcProfileDef* is known as *IfcArbitrary-ClosedProfileDef*, which defines a closed profile by referencing an *IfcCurve* object. By using the *IfcExtrudedAreaSolid* class, this profile can then be used for an operation in which extrusion is applied for a specific distance described by “Depth” attribute along a specified direction represented as “ExtrudedDirection” attribute as shown in figure 16. When using the *IfcRevolvedAreaSolid* class, the profile is rotated around a specific axis with a specific angle defined as “Angle” attribute as presented in figure 16. The *IfcFixedReferenceSweptAreaSolid* class can be used to model an object as the result of sweeping a defined profile along a specified curve with the help of “Directrix” attribute. An important feature of this representation is that the profile cannot be twisted but remains oriented to a fixed reference vector while sweeping (Borrmann, Beetz, Koch and Liebich, 2018).

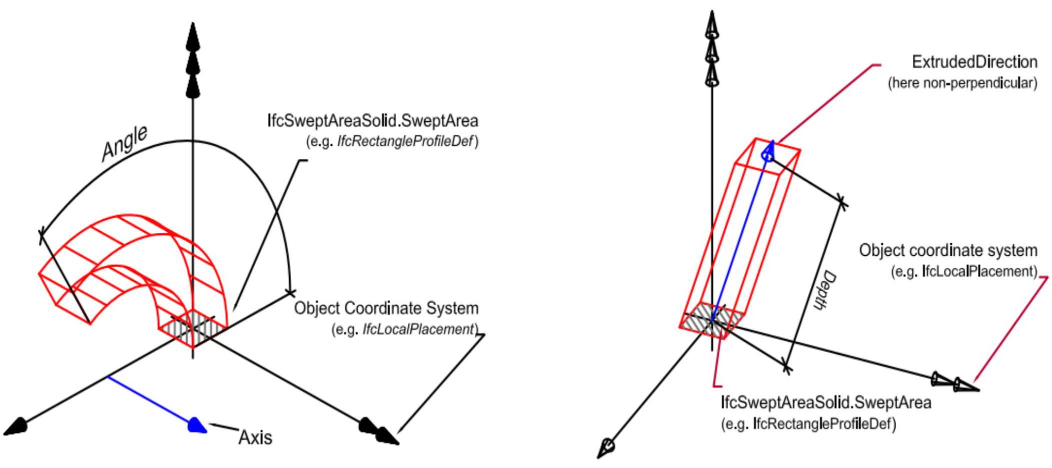


Figure 16 - Examples for modeling solids using IfcRevolvedAreaSolid and IfcExtrudedAreaSolid (buildingSMART).

2.2.5.2 Surface Modeling

A surface model provides a way to describe a composite surface made up of multiple surfaces. The main use of this representation type is to represent terrain models or flat surfaces. Moreover, solids described in 3D can also be represented via surfaces. The advantage of this method over Brep representation is the simplicity of the data structure. While the drawback is the limited possibilities for checking the correctness of modeled solids. This can be foreseen in improper intersections between surfaces such as gaps and overlaps. The IFC data model implements two variants of the surface model as seen in figure 17. *IfcFaceBasedSurfaceModel* allows for modeling bodies without voids presented among them while *IfcShellBasedSurfaceModel* allows for voiding in the modeling of solids with the help of *IfcShell* objects which can be used as open shells or closed shells (Borrmann, Beetz, Koch and Liebich, 2018).

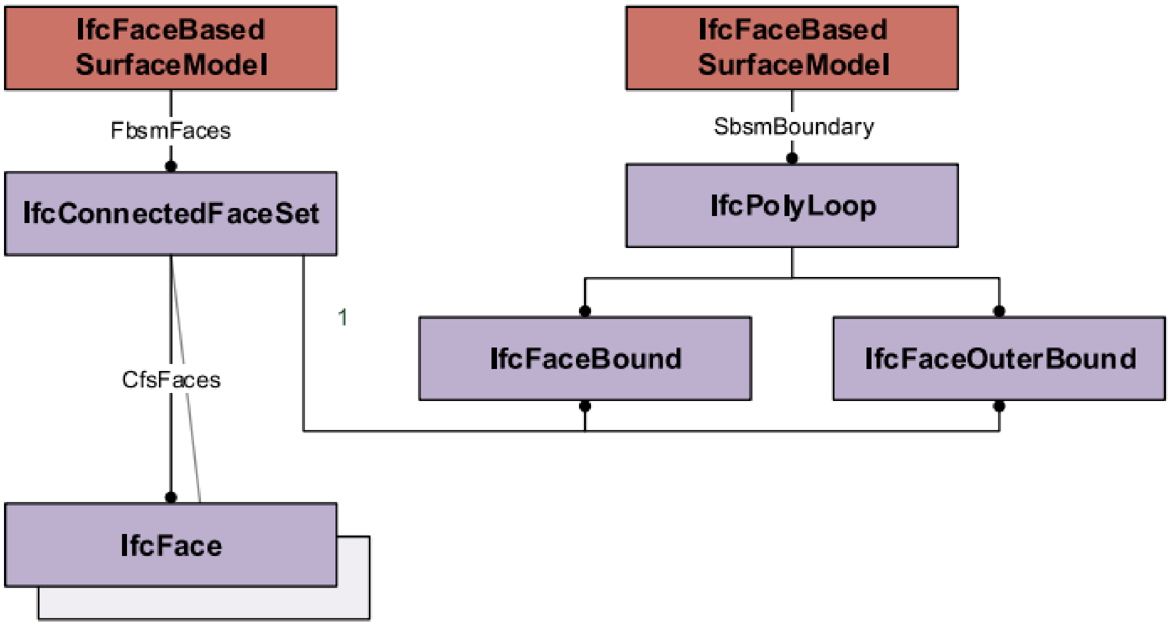


Figure 17 - Data structure for description of surface models (Borrmann, Beetz, Koch and Liebich, 2018).

Tessellation

Triangulation approach is a widely used approach for representing geometric shapes using triangulated mesh. This very common and simple form of geometric representation can be interpreted by almost any visualization software product. Its main disadvantage is the approximation of curved surfaces by triangular facets. Surfaces are data-intensive, and many applications offer limited possibilities for editing these types of elements. So, this geometric representation is not always the perfect form to create geometry. Furthermore, Digital Terrain Models (DTMs) are described by means of triangulated surfaces. For such applications, the IFC data model provides the *IfcTriangulatedFaceSet* class. It derives from the *IfcTessellatedFaceSet* class, which represents the general concept and idea of tessellated surfaces such as polygons with any number of edges. *IfcTessellatedFaceSet* inherits from *IfcTesselatedItem* instead of deriving from *IfcSolidModel* class (Borrmann, Beetz, Koch and Liebich, 2018). Figure 18 represents an example of a curved element with a fixed profile described by IfcTriangulatedFaceSet class.

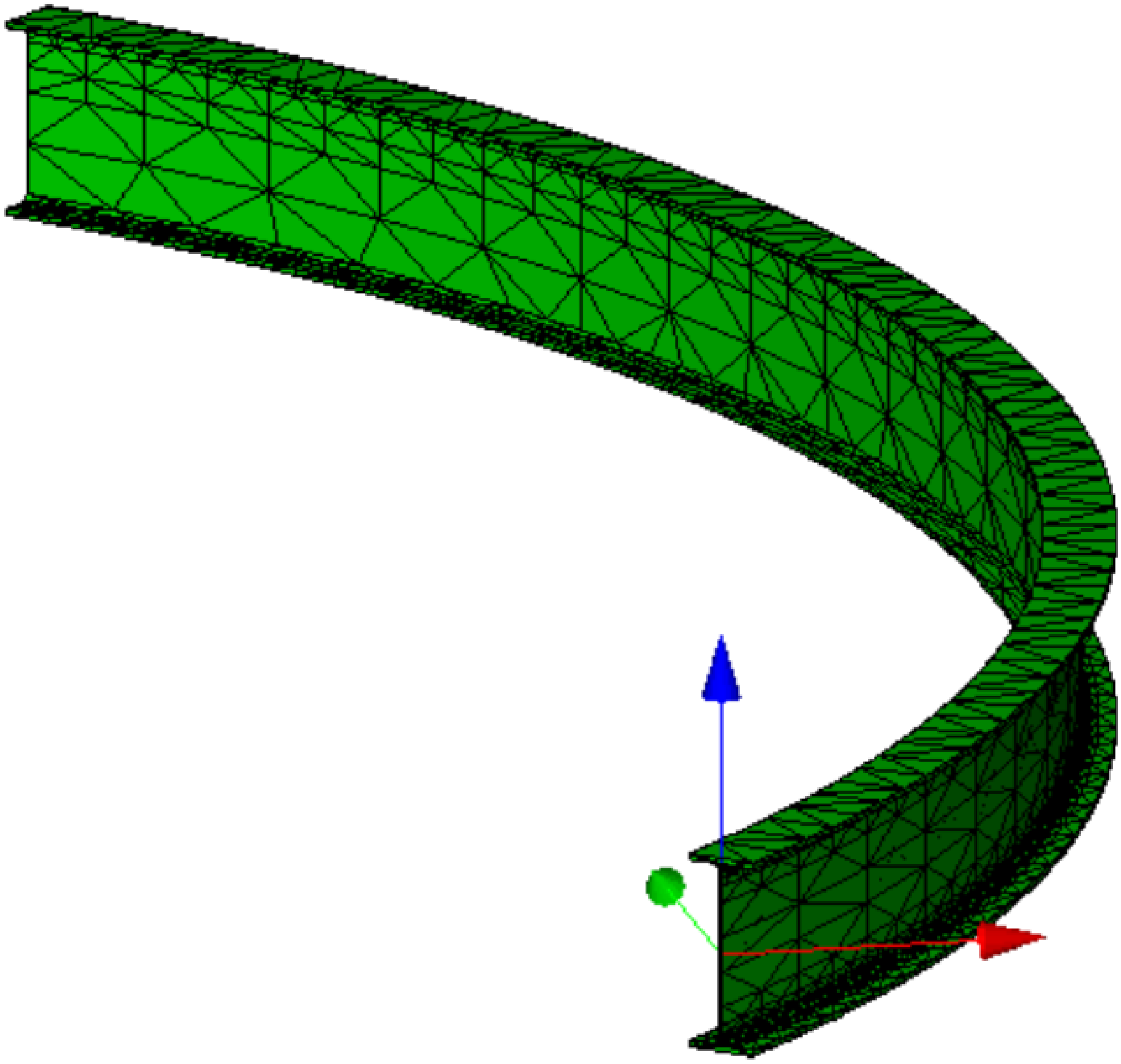


Figure 18 - Example showing curved beam described by IfcTriangulatedFaceSet (buildingSMART).

2.2.5.3 Additional Types

Bounding Box

The Bounding Box is the simplest geometric representation type available within the IFC data model. The idea is to approximate any object into a bounding box through the exterior geometry of the object which determines the dimensions or size of the bounding box. In the *IfcBoundingBox* class, three attributes are found that determine the dimensions of the bounding box known as XDim, YDim, and ZDim in addition to another attribute called Corner which represents the bottom left corner of the box in terms of coordinates as shown in figure 19.

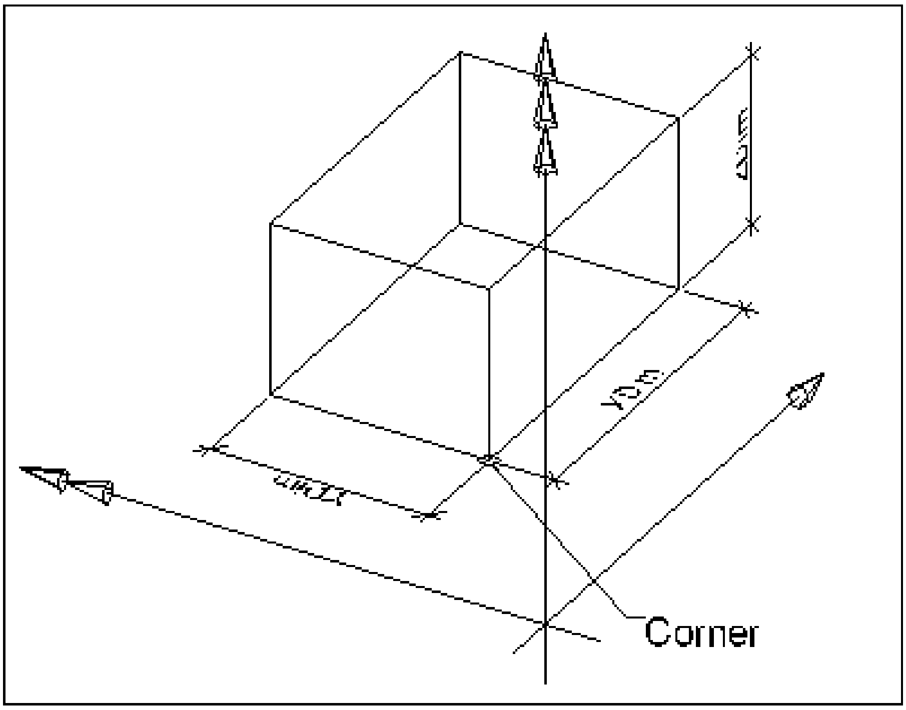


Figure 19 - Bounding box and its attributes (buildingSMART).

Understanding the possible geometry representation types in the IFC models is essential to present a valid geometry evaluation concept that will hold the responsibility for comparing the geometry of elements in version control systems.

2.3 Version Control

The term VCS which stands for version control system is basically defined as a system that manages the development of evolving objects across different versions. (Zolkipli, Ngah and Deraman, 2018). That is a system that is able to record all changes and modifications made by users working together on a project. The versioning concept is already implemented universally as it helps in enhancing communications and tracking project development through identifying changes applied regarding every key aspect in the project (Zolkipli, Ngah and Deraman, 2018).

Versioning concept is implemented in many industries, the concept itself is adopted in different industries with different implementation aspects and guidelines. However, version control systems are mostly used in the software development industry for a long time now.

Regarding the software development process, software developers frequently tend to modify code and other files to implement certain features or enhance the functionality of the software. It is clear that several revisions will be made before the final version is produced. As the system becomes larger and more complex and the number of revisions increases, it becomes difficult to manage and organize code and files. Therefore, the existence of VCS really helps software developers to speed up and simplify their development process. Without a VCS, software developers are tricked into keeping multiple copies of their code on their computers. This is dangerous because documents and files can easily be changed or deleted with the wrong copy of code and your work can be lost. A version control system addresses this problem by keeping track of all versions of the codes developed by users to identify changes and modifications between versions. Adoption of a VCS will be mandated in phases so that all software developers working on the same project can work in an efficient way towards project milestones. Typically, version control systems have two main types which are Centralized Version Control System “CVCS” and Distributed Version Control System “DVCS”. CVCS has on a central repository which is the server while DVCS has a local repository for each user as shown in figure 20 (Zolkipli, Ngah and Deraman, 2018).

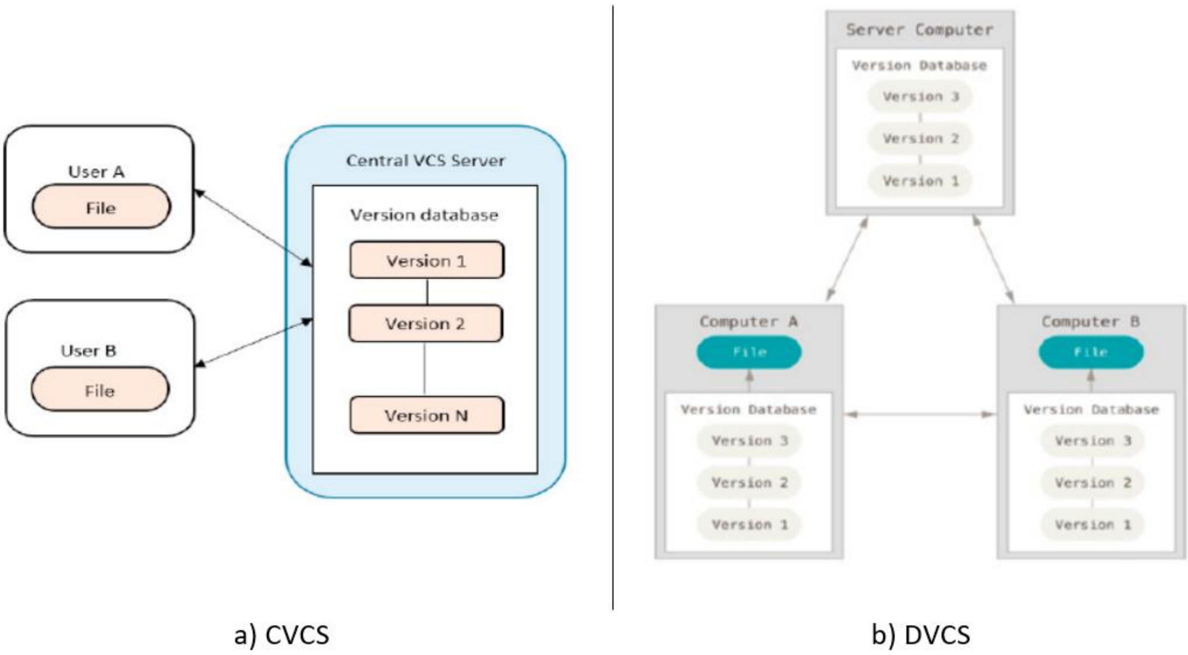


Figure 20 - Concept of CVCS and DVCS (Zolkipli, Ngah and Deraman, 2018).

There are various methods, systems, and protocols that enable distributed version control of text files. Examples include products such as Mercurial and Git. Most approaches apply the same workflow presented in storing a global history of changes in a central database, consolidate incoming modifications through "commit and push", and allow users to view the full history of changes and clone them on their local devices. So, each user can read and understand the entire history and evaluate changes locally. When users are ready to share their changes with others, they resynchronize their local state with the central database. A series of update messages form the complete history of the project. Incoming updates can be automatically merged if they do not have any conflict with existing or concurrent local changes. Only in cases where certain conflicts are encountered, users should resolve them and manually choose the desired content to eliminate all kinds of conflicts presented (Blischak, Davenport and Wilson, 2016).

2.3.1 Versioning of BIM models

Existing version control services use line-based data comparison in addition to tracking of text lines that have been added, deleted, or changed. However, the data models used in the AEC industry usually define complex and highly interconnected data structures that cannot be easily versioned. For example, the order of entities in two different versions of a physical STEP file “SPF” can be completely different, whether or not both versions provide the same content in terms of information. Despite these limitations, text-based serialization of is widely used in data models to transfer BIM data in file-based delivery scenarios. Typically, collaboration is primarily achieved through file-based data exchange in the current practice of AEC projects. Users from different domains integrate their work together through a central database known as the Common Data Environment “CDE” (Esser, Vilgertshofer and Borrmann, 2021).

BIM models are mainly communicated through IFC data models in the AEC sector nowadays, IFC provides capabilities to exchange information and data through different platforms and systems (Muller et al., 2017). Furthermore, versioning of IFC data models is challenging since it is hard to come up with certain criteria that can be applied to all data models, some tools apply versioning of entities under the assumption that their order does not change which is not true in many cases, other assumption might be associated with GUID stability throughout the models in which GUID comparison is performed between instances to identify if changes were made or not. Additionally, GUIDs of objects might be changed even if the model did not undergo any type of modifications. Therefore, comparisons which are based on GUID are not reliable and can provide misleading results (Shi et al., 2018).

Shi et al. (2018) introduced an approach for detecting the differences presented in two IFC models in terms of similarity rate. Their approach is based on the use of a recursive depth-first search after normalizing all instances found in the model to compute the similarity between the models. the main drawback is that the rate presented at the end as output does not expose any means of understanding for the user regarding the changes happened. Figure 21 shows an example for the output of this tool on two models, and it is not clear what kind of changes were detected per aspect like the “Geometry” or “Positioning” rather it gives numbers describing only how many instances were added or deleted which is not practical for many use cases in the AEC industry where users from different domains would expect more detailed explanation of changes to understand the actual difference between two models regarding several aspects.

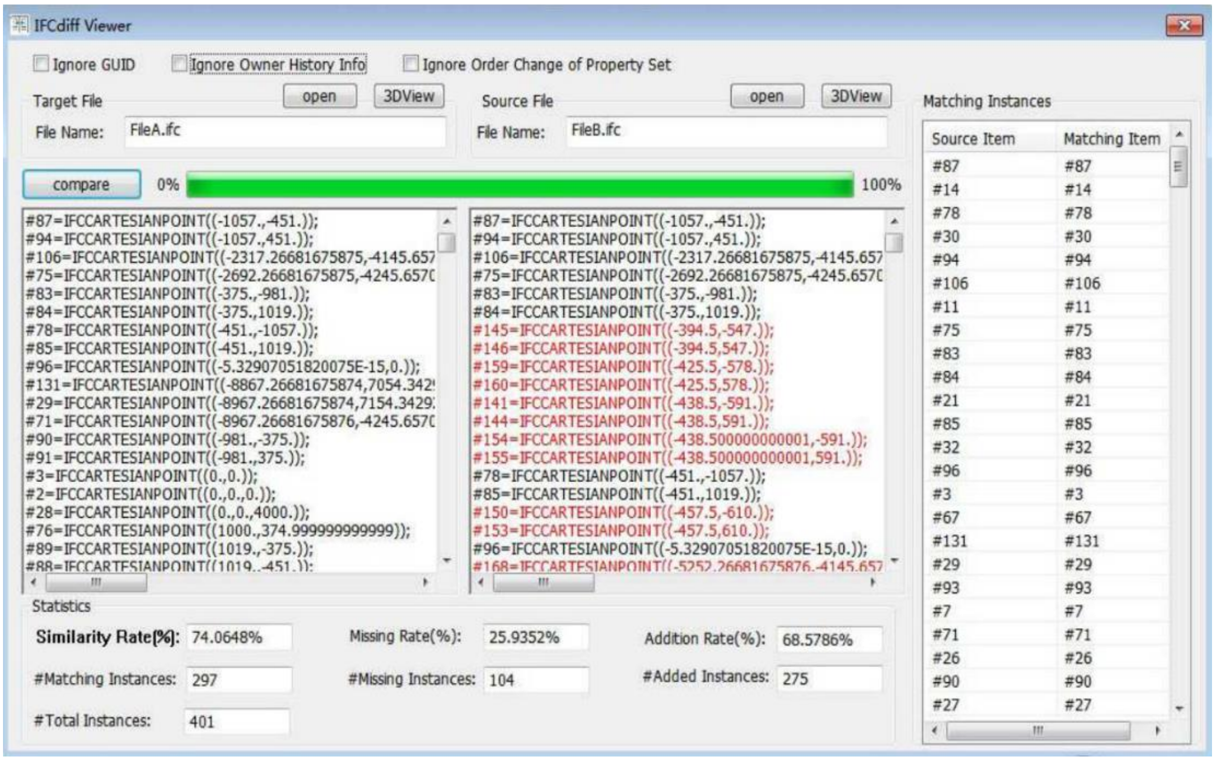


Figure 21 - Output from IFCdiff tool (Shi et al., 2018).

Currently, in the market, there are some commercial tools that provide versioning features to identify differences between two models such as Autodesk BIM 360 and BIMvision through the module "Compare".

Figure 22 - Output from comparing two model versions in BIM 360 (Autodesk).

Autodesk BIM 360 is a product by Autodesk which support the coordinators with features that help in coordination and collaboration tasks in addition to other features which support model and document control tasks. With the comparison module available on BIM 360, a user can compare two versions of a model and evaluate the differences. As shown in figure 22, the differences are classified into three categories representing the elements added, deleted, and modified between the two versions. Results are presented in a visual form where elements are associated with color codes based on the type of difference they have. Furthermore, by clicking on modified elements the differences are described in terms of internal quantities like volume, total area, or placement.

BIMvision performs the comparison process exactly like BIM 360, after loading the models the differences are expressed with color codes assigned to the elements and the user can see quantifiable differences after selecting elements. However, results from the two products are not always the same especially for detecting geometric differences. The reason behind this is the different approaches each product use for evaluating geometry.

2.4 Research Gaps

Currently, the software products available in the market for version control of BIM models offer a detailed explanations for users working with models by describing modifications with quantities so that project coordinators can identify the actual changes within each aspect. However, the results from this comparison processes are not always accurate and reliable since the criteria for evaluating different aspects within these tools remain mysterious. For example, “ifcwebserver” product compare the geometry of objects between two models based on a direct comparison of volumes which does not make sense as the object might change in dimensions while keeping the same value for the volume, or the element itself might be represented as a two-dimensional surface which has a volume equal to zero. So, internal quantities like volume cannot be a general method for the comparison process and the evaluation of geometry need to rely on more practical approaches.

3 Methodology

3.1 Overview

As proven by the conducted literature, the need for versioning of BIM models is increasing with the evolvement in the AEC industry. Methods for versioning models vary in nature and depend mainly on the intended target of the comparison process. This study focuses on identifying geometric differences between elements of two IFC models through proposing and evaluating methods suitable for evaluating geometry. The main idea is to introduce geometry evaluating techniques that can be applied to any geometrical shape regardless of the representation type which describes the element. The following paragraphs cover details about the two proposed approaches which were introduced in this research to evaluate geometric differences between elements in IFC models, covering the concept, implementation, and motivation for each approach.

3.2 Approach 1: Bounding Box Comparison

3.2.1 Concept and Implementation

The concept behind the bounding box comparison approach is to construct a bounding box for each geometrical shape found in the models as shown in figure 23, then comparison can be held based on comparing dimensions of the constructed bounding boxes.

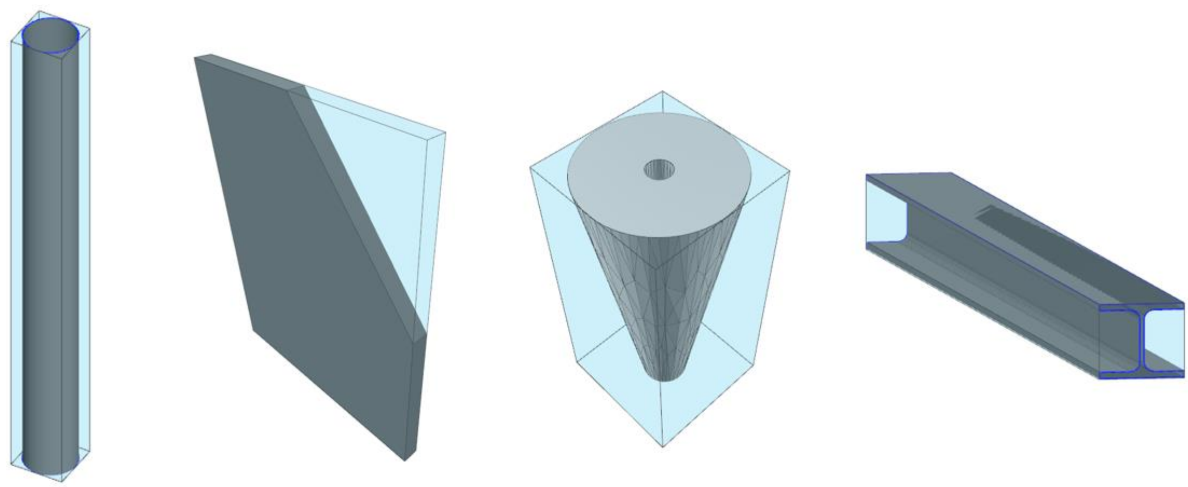


Figure 23 - Examples for Bounding Boxes of different geometries.

Typically, bounding boxes are simple cubes with three dimensions that are created based on the exterior geometry of an element in the three-dimensional space. The exterior geometry of elements is the main influencing factor that determines to which extent the dimensions of the bounding box will stretch.

Bounding box concept is common and can be found in many BIM tools and IFC viewers, some tools define it as one of the internal properties of elements and it can be expressed with parameters in different ways, usually, it is identified by size which represents the numeric value for the three dimensions or by min/max which represent the bottom left and top right corner of the box in terms of coordinates which are referenced to the model coordinates. Furthermore, the bounding box concept is also common in other industries, and it has different use cases. For example, it is used in object detection, and it is one of the most popular techniques among image processing methods.

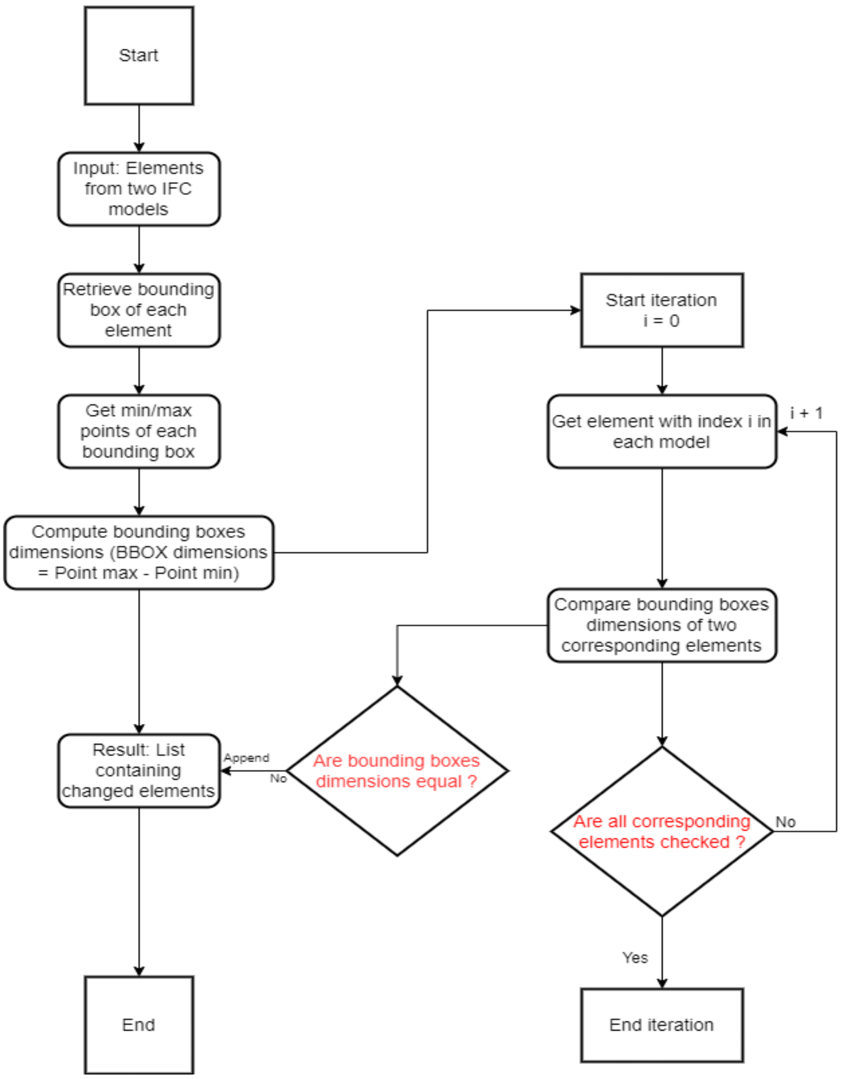


Figure 24 - Bounding box comparison flow chart.

Moreover, figure 24 illustrates the process of geometry comparison using this approach which starts by getting common elements between two IFC models. The term "common" means the elements which are found in both versions which kept the same element id unchanged. Then retrieving their bounding box with the help of xBIM library package which formulates the bounding box and expresses it as a direct property for every instance.

Afterward, bounding box dimensions can be calculated for every instance by subtracting the coordinates of the point at the top right corner (max) from the point at the bottom left corner (min), and then the comparison is held for the two corresponding elements to state whether the geometry of the element in the old versioned was changed in the new version or not. If the element was subjected to changes regarding geometry it is added to the list of differences between the two models and the process is repeated until all common elements are evaluated.

This approach also covers an additional feature regarding elements replacement. For example, when an element is removed from the first version, it captures its bounding box and searches for an equivalent element with the same bounding box dimensions and position as the new elements of the second version, if the equivalent element is found, then both elements will be removed from the lists "New" and "Removed" and will be reported as "Semantically modified".

3.2.2 Motivation and Advantages

The choice of this approach was based on several advantages and benefits that it offers which were the main motivation for implementing this approach and they can be summarized as follows:

- Easy to implement: Retrieving the bounding boxes for elements in IFC models is easy with the help of certain external packages.
- Applicable to any geometry: The concept of bounding boxes can be applied with the same criteria to any geometrical shape of any geometry representation type in IFC data models.
- Suitable for models with specific LOD levels: This approach can afford reliable results for models of low LOD levels in which elements are represented just with simple geometry that can be easily handled with this approach.

- Independent of element positioning: The idea in the geometry comparison process is to compare the difference in bounding boxes dimensions, not the difference between the max/min point coordinates of two bounding boxes as in this case any translational movement of an element in updated versions will be reported as a change in the geometry.
- Results are quantifiable: Results from this approach are expressed by the difference in the dimensions of the two bounding boxes that were compared which gives the user a clear idea of what type of modification was encountered for a specific element.

3.3 Approach 2: Mesh Comparison

3.3.1 Concept and Implementation

The mesh comparison approach is based on formulating triangulated mesh for elements and then retrieving vertices as a list of points of known coordinates in the three-dimensional space from the formulated triangular mesh. So, at the end, the comparison is made between point lists of two elements to evaluate the geometric differences between them.

Furthermore, the triangulated mesh can be applied to any geometrical shape in IFC models. However, the description of the element will vary depending on the geometrical shape to be described. For example, a simple rectangular column which is described by extrusion of a rectangular profile along a specific path will have a triangulated mesh consisting of 12 triangles and 8 vertices as shown in figure 25, while more complex shapes with non-uniform surfaces and curves might tend to have triangulated mesh consisting of thousands of triangles and vertices.

The concept of this approach is adopted by the IFC scheme with the class *IfcTriangulatedFaceSet* where it is used to describe objects by means of triangulated mesh composed of a collection of triangles as described in chapter 2. The class can be applied to represent many geometrical shapes. However, it is mainly used for the representation of topological surfaces and digital terrain models.

Generally, triangulated mesh offers a lot of use cases across several industries, a lot of modeling and visualization tools adopt this concept for the representation of objects. It is also used by technical tools like 3D printers in identifying geometrical shapes for 3D printing of objects.

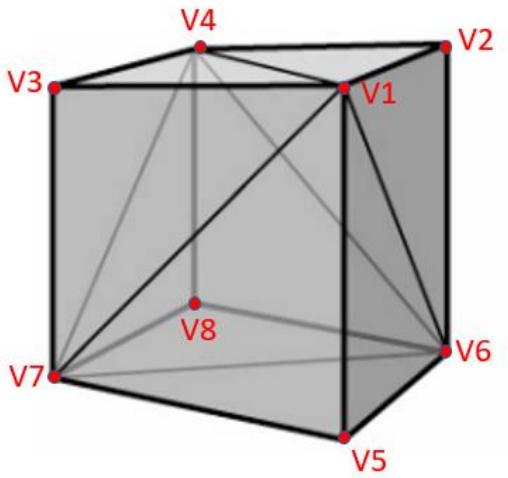


Figure 25 - Triangulated mesh of a cube.

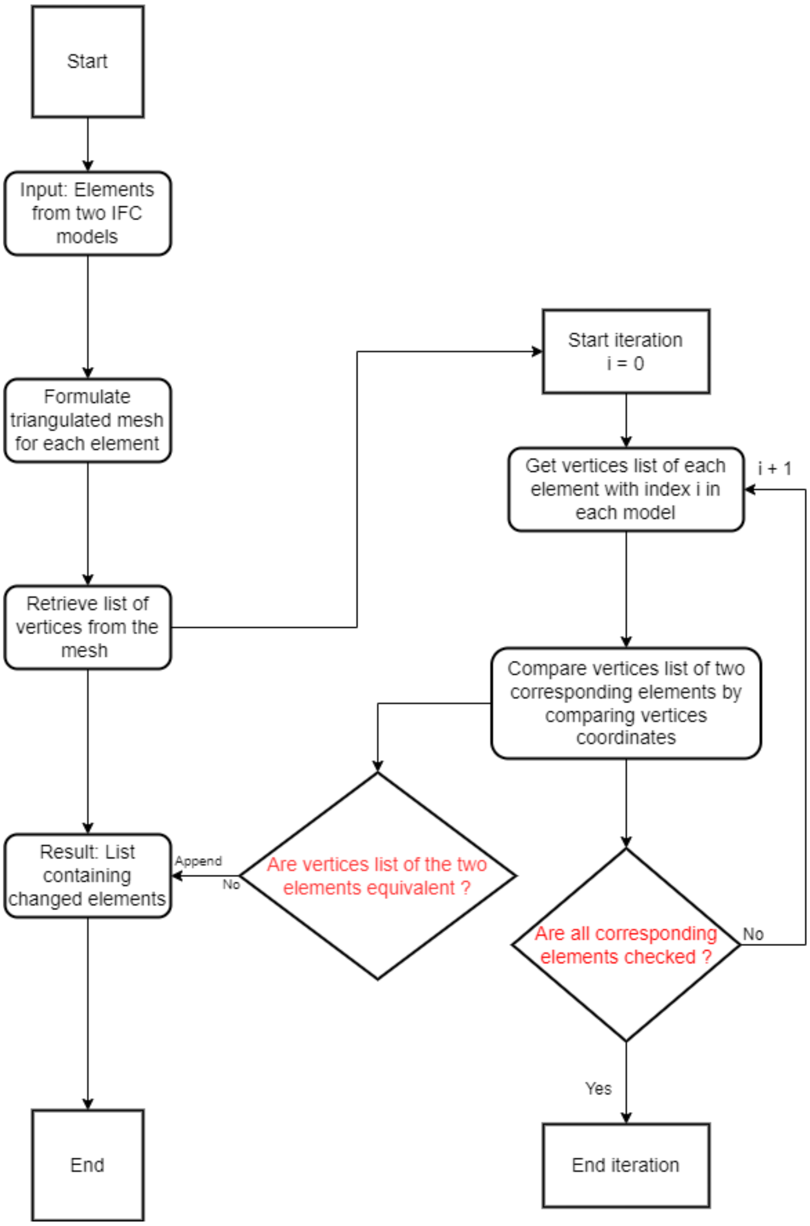


Figure 26 - Mesh comparison flow chart.

The process of geometry comparison using this approach is presented in figure 26 by a flow chart describing the whole process for geometry comparison. The process starts by retrieving common elements between two IFC models. then the triangulated mesh is constructed for each element by an internal function of the geometry engine of xBIM library package as shown below.

```
private List<XbimPoint3D> extractGeometryVertices(XbimShapeInstance instance, Xbim3DModelContext ctx)
{
    List<XbimPoint3D> vertices;
    var transformation = instance.Transformation;
    XbimShapeGeometry geometry = ctx.ShapeGeometry(instance);
    XbimRect3D box = geometry.BoundingBox;

    byte[] data = ((IXbimShapeGeometryData)geometry).ShapeData;

    using (var stream = new MemoryStream(data))
    {
        using (var reader = new BinaryReader(stream))
        {
            var mesh = reader.ReadShapeTriangulation();

            vertices = mesh.Vertices as List<XbimPoint3D>;
        }
    }

    return vertices;
}
```

Algorithm 2 - Part of the code showing the method developed to retrieve vertices list from triangulated mesh.

Afterward, a comparison of two vertices lists of two corresponding elements takes place by comparing the coordinates of points of the first list with the coordinates of points of the second list, if the two elements have a difference in terms of geometry, it will impact the coordinates of one or more point in the second list. Hence, the element will be added to the results list which contains all the elements with geometrical differences. The process will be repeated until all the corresponding elements are evaluated and compared.

3.3.2 Motivation and Advantages

This approach is practically powerful in evaluating various geometrical shapes as it offers a lot of benefits and advantages that are summarized as follows:

- Applicable to any geometry: The triangulated mesh can be formed for any geometrical shape. Hence, this approach is valid for all elements.
- Support voiding: The formulated mesh is affected by the presence of voids in evaluated elements as it influences how the mesh is built. Since voiding elements are common in any built project in the AEC industry, they need to be taken into consideration while evaluating associated elements as they affect the geometry of the element at the end.

- Independent of LOD level of the model: This approach can deal easily with any geometric shape. So, it is applicable to models with any LOD level starting from evaluating simple geometries to complex and irregular geometry found in models with LOD 300 and above.
- Changes easily detected: evaluating geometry with this approach is highly sensitive as the coordinates of each vertex in the list of vertices of an element are compared to the corresponding vertex of the new version of the element. Hence, any slight deflection in coordinates of one vertex in the updated version will be reported as a change in geometry.

3.4 Summary

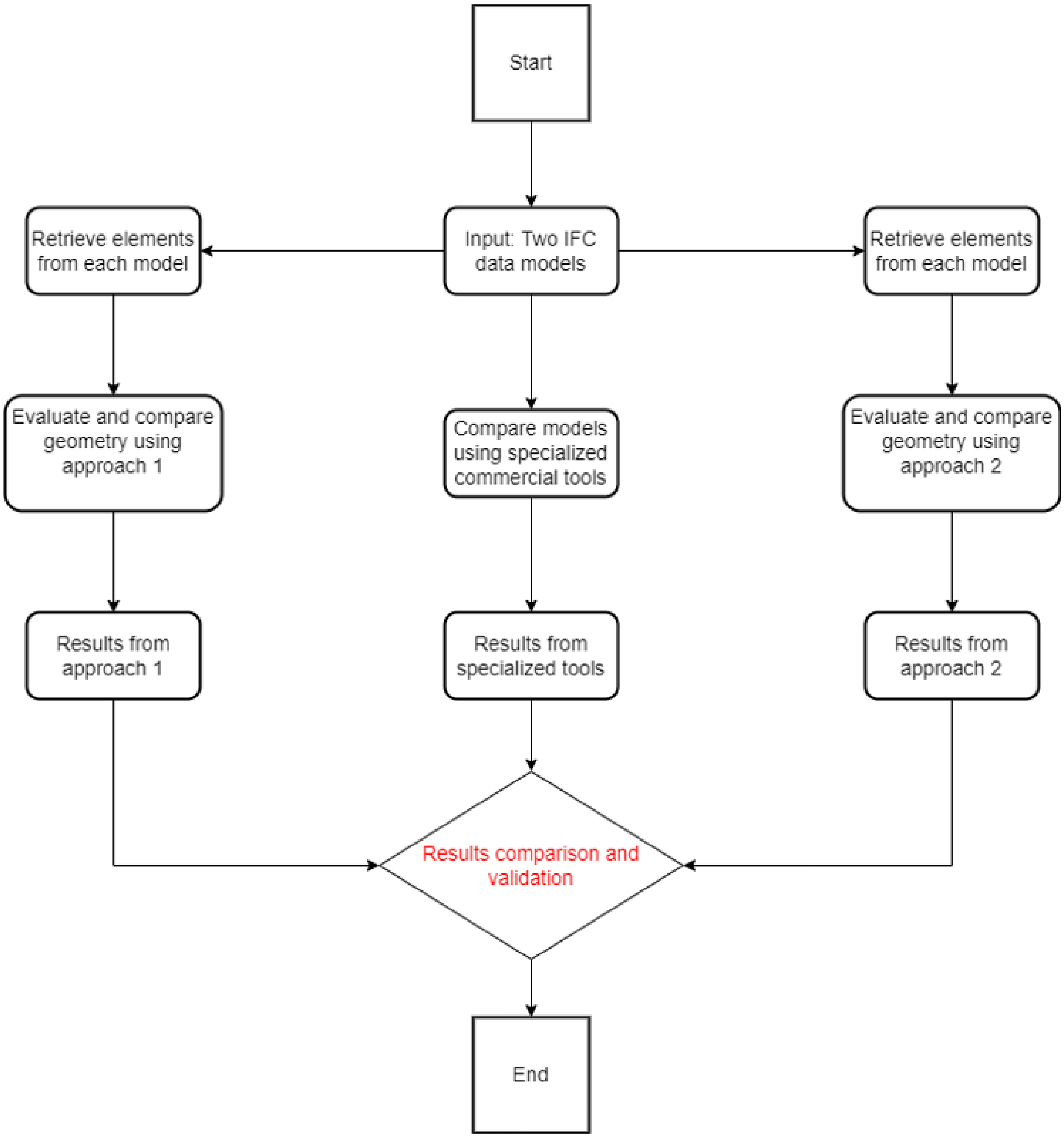


Figure 27 - Flow chart summarizing the proposed methodology.

The proposed approaches are based on completely different concepts. So, they need to be tested and their results shall be compared to identify limitations and gaps. Figure 27 summarizes the proposed methodology described in this chapter.

The process starts by obtaining two IFC models representing two versions of one model as the input. Afterward, the common element which can be found in both models are retrieved to perform the evaluation and comparison, then the evaluation and comparison of geometry take place using the two prescribed approaches. Finally, results from each approach are obtained and are to be compared to results from specialized products which perform this kind of comparison like Autodesk BIM 360 to validate the results coming from each approach to ensure that the results are reliable. Moreover, the comparison of the results is an essential step to identify the main limitations of each approach which will be the basis for future developments and research.

4 Case Study / Prototype

4.1 Prototype: User Interface, Inputs, and Outputs

The tool that was created during this research for testing the versioning concepts mentioned before is called *IFCcomparison*, and it was developed using C# language on visual studio as a Windows Presentation Foundation "WPF" application. The main supporting external package which was added to the tool is xBIM package which helps in reading and manipulating data stored in IFC files and processing geometric instances using their geometry engine. Figure 28 shows the user interface of the tool where inputs and outputs are highlighted.

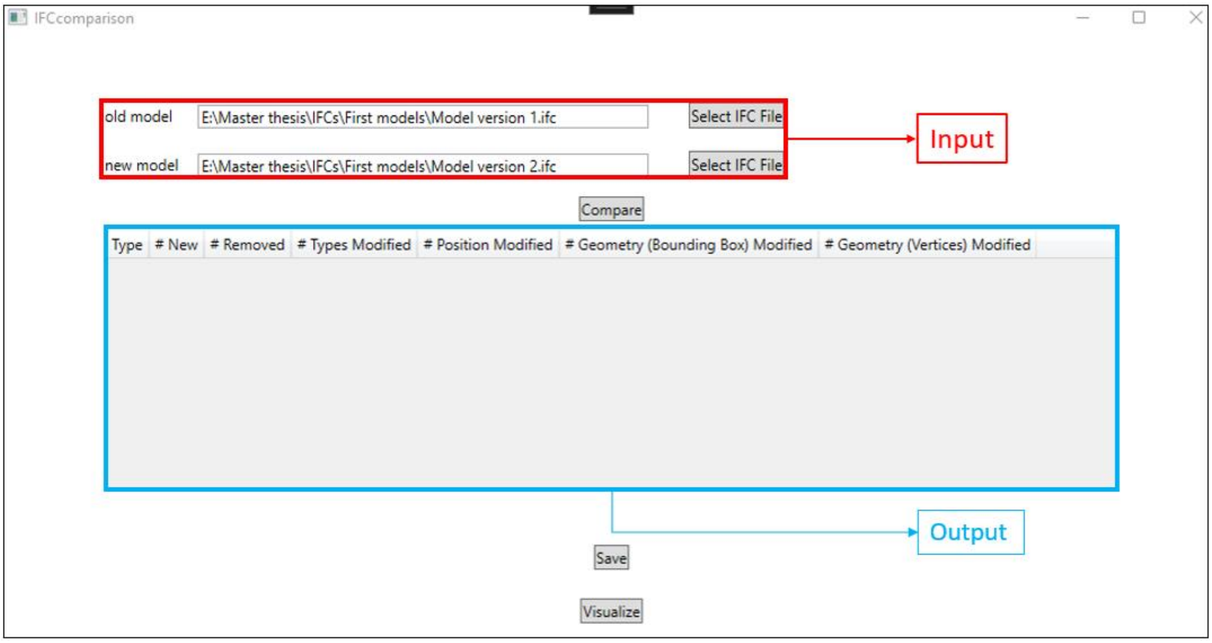


Figure 28 - IFCcomparison user interface.

The tool requires two IFC files which represent two versions of a model to perform the comparison procedure by clicking "Compare", then the comparison starts, and a summary of the results is presented in a table that provide a counter for differences per each aspect and element type as shown in figure 29. The main intention was to create a tool that applies the versioning concept only to detect geometric differences between versions. However, the implementation has covered other aspects besides the geometry to make the tool more comprehensive and to determine all differences between the two IFC models.

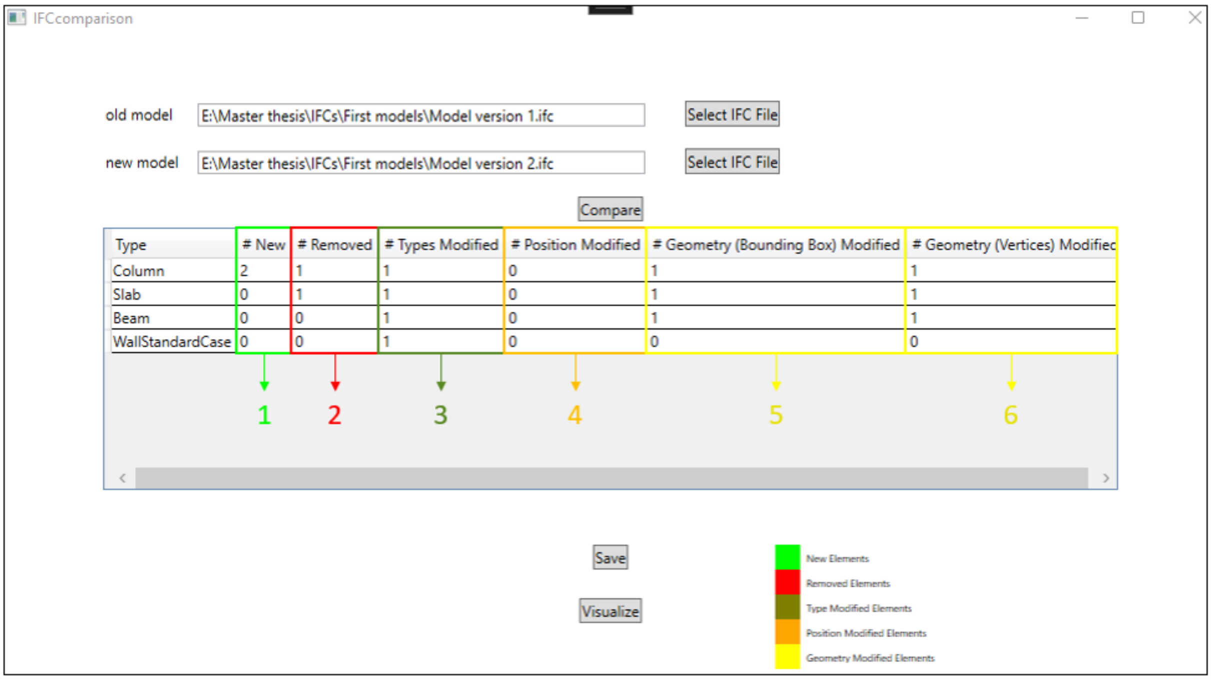


Figure 29 - Output for sample model from IFCcomparison tool (1/3).

The output presented in the table is determined per each modification type and per each element category as shown in figure 29. The modification types from 1 to 6 can be described as follows:

1. Elements that were added in the second version and are not found in the first version.
2. Elements that were removed in the second version and found only in the first version.
3. Elements that their “type name” which is a property in the modeling tools were changed in the updated version.
4. Elements that experienced a translational movement in x, y, or z direction in the updated version.
5. Elements that experienced modification in geometry according to approach 1 (Bounding Box comparison) in the updated version.
6. Elements that experienced modification in geometry according to approach 2 (Mesh comparison) in the updated version.

The initial results presented in the table are just given to give a user some clear understanding of the number of modifications found between two versions for each modification type but after clicking the button “Save”, the user will get an excel file that provides a detailed explanation for each modification type as shown in figure 30.

	A	B	C	E	F	G
1	New	Removed	Type Modified	Geometry Modified	Geometry 2 Modified	Semantically Modified
2	IFCColumn : 2297619	IFCColumn : 2296345	__TRG-UEZ-Rechteck:20x20_C30/37 -> TRG-HEA:HEA100_S235 : 2297257	__3.033,0.2,0.2 -> 3.008,0.1,0.096 : 2297257	__Vol Old: 0.12133 m3 -> Vol New: 0.00602 m3 : 2297257	IFCColumn (2297633) -> IFCWallStandardCase(2296613)
3	IFCColumn : 2297681	IFCSlab : 2296023	__STZ-Rechteck:20x20_C30/37_AS -> STZ-Rund:20_C30/37_AS : 2296587	__0.2,0.2,3.2 -> 0.21,0.209,3.21 : 2296587	__Vol Old: 0.128 m3 -> Vol New: 0.09956 m3 : 2296587	
4			__Geschossdecke:20_C30/37 -> Geschossdecke:30_C30/37 : 2296630	__3.433,6.367,0.2 -> 3.433,6.367,0.3 : 2296630	__Vol Old: 4.37178 m3 -> Vol New: 6.55767 m3 : 2296630	
5			__Basiswand:20_C30/37_AW -> Basiswand:20_C30/37_IW : 2296625			

Figure 30 - Output for sample model from IFCcomparison tool (2/3).

The results are described in detail in the excel file which shows the exact change with the element id for each modification type. Geometric modifications are presented in columns “E” and “F” based on approaches 1 and 2 respectively. For approach 1, the bounding box dimension of the element in the first and second version are mentioned to identify and quantify the difference in addition to the element ID which helps in finding the associated element in the model. While the results are interpreted per volume or area for approach 2. As it is hard to quantify the difference in approach 2 to be understandable by the user, internal quantities like volume and area are used instead to quantify the change in the element geometry. However, this does not mean that those internal quantities are used in the comparison process to find the difference in geometry between elements, they are only used to quantify the change in the geometry of elements.

Additionally, the tool in this sample reported a modification regarding the replacement of element under column “G”, approach 1 detected a replacement of a structural column by a wall that had the same dimensions and same placement position. In this case, neither the column is reported in the “Removed” list nor the wall is reported in the “New” list.

Furthermore, the tool offers a visualization option through the button “Visualize” which overrides graphics of changed elements according to the defined color code from the IFC files, then the results are exported to a “. xBIM” file which is readable by xBIM explorer that the user can open to visualize the differences between the models as shown in figure 31.

The visual comparison of results is practical when the evaluated model is a mega-scale model that contains thousands of elements with a huge number of modifications as it will be hard for a user to evaluate the results in the excel file for each element in each modification type manually.

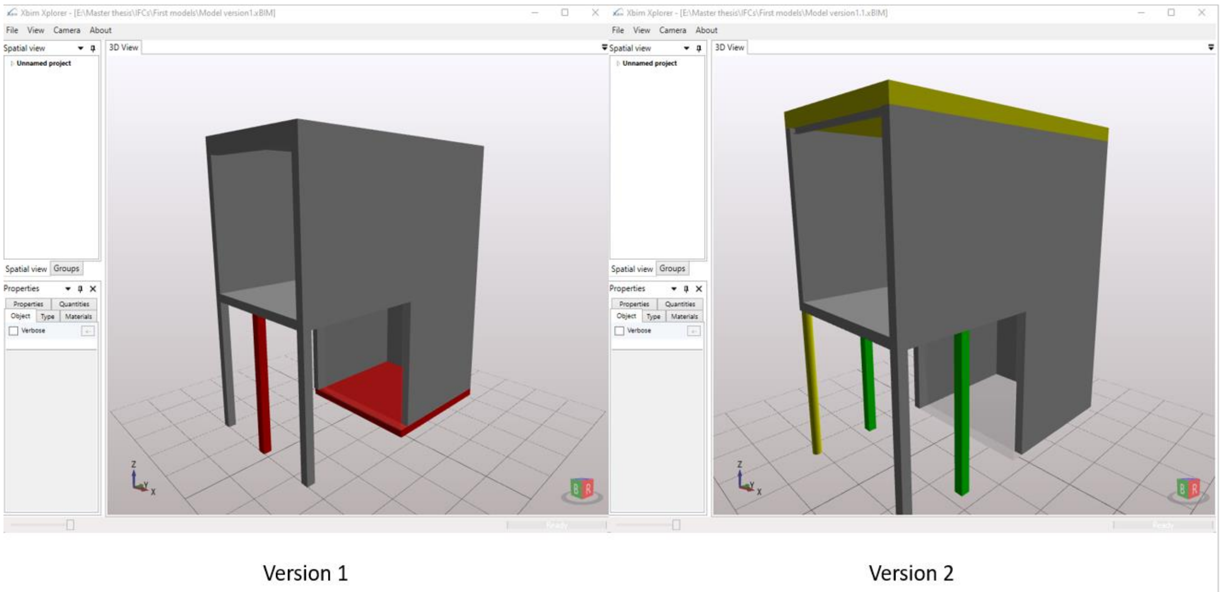


Figure 31 - Output for sample model from IFCcomparison tool (3/3).

Figure 31 shows a visualization comparison example of a sample model, in the first version the user can see only the elements removed from the updated version highlighted, while in the second version the user can see all other differences regarding each aspect highlighted.

4.2 Case Study

The test models that will be used in testing and evaluating the accuracy of the results are created by Zilch + Müller Ingenieure GmbH. Basically, two models M1 & M2 are going to be tested on both IFCcomparison tool and Autodesk BIM 360 docs and the results will be shown in chapter 5.

4.2.1 Model M1

Model M1 is a building model that consists of three levels including the foundation level, it is not the full building but the two versions of this model which will be tested had many modifications and changes in these levels. Hence, it is useful to conduct the experiment on such a case to validate the results of the tool.

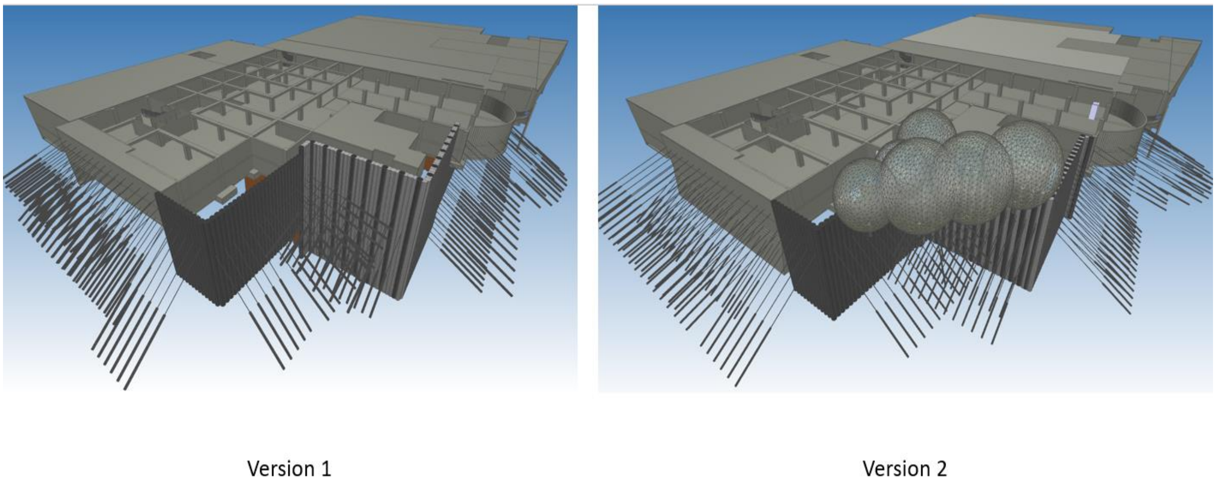


Figure 32 - 3D view for the two versions of model M1.

Figure 32 shows the two versions of the model. The first version of the model consists of 1471 elements while the second version consists of 1478 and the composition of elements is defined in table 1.

Type	Version 1	Version 2
Generic Object	447	442
Wall	424	437
Slab	121	121
Beam	197	196
Column	275	275
Stair	7	7
Total	**1471**	**1478**

Table 1 - Composition of elements in the two versions of model M1.

The generic objects in the table represent the main foundation elements in this model as these elements were modeled using custom Revit families. Moreover, the total number of elements in the table represents only the physical elements without considering spatial elements like grids or detailing lines

4.2.2 Model M2

Model M2 is a commercial building consisting of 7 levels including the foundation level, this model was selected as a test model since the two versions had many diverse differences across all types of elements on all levels. Moreover, it is clear from figure 33 that the two versions of the model are not similar at all. Therefore, many changes and modifications are expected to be documented in the comparison process.

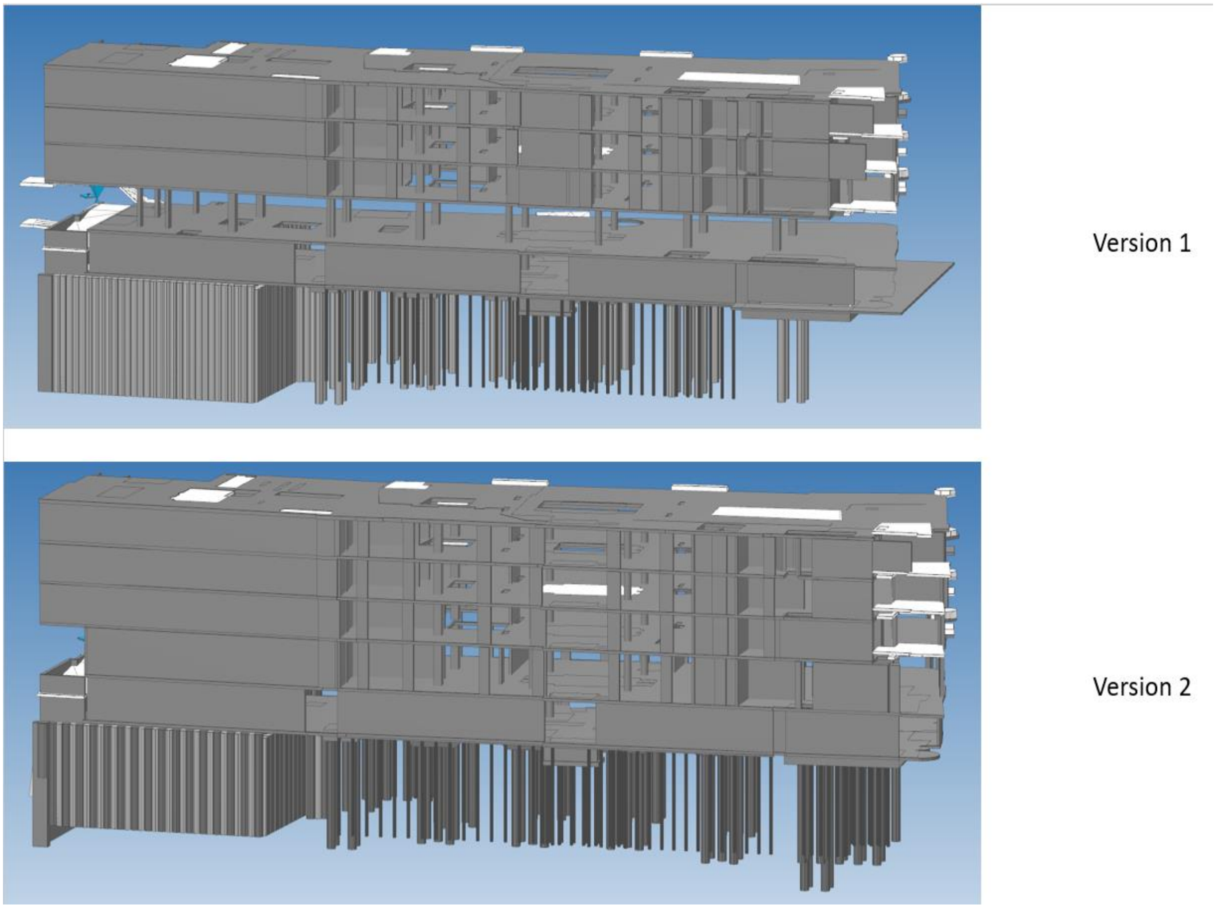


Figure 33 - 3D view for the two versions of model M2.

The first version of the model consists of 914 elements while the second version consists of 973 and the composition of elements is defined in table 2.

Type	Version 1	Version 2
Generic Object	247	288
Wall	313	347
Slab	212	193
Beam	6	7
Column	136	138
Total	**914**	**973**

Table 2 - Composition of elements in the two versions of model M2.

The generic object category includes all the Revit custom families instances in the model which are mostly foundation elements in addition to some other additional elements like retaining walls.

5 Results

Model M1 showed unrealistic results when the two IFC files were added to Autodesk BIM 360 as shown in figure 34. Therefore, two Revit files “. rvt” of model M1 are added instead on Autodesk BIM 360 to achieve more realistic and reliable results. On the other hand, the IFC files of model M2 produced realistic results that can be used directly to validate the results of IFCcomparison tool.

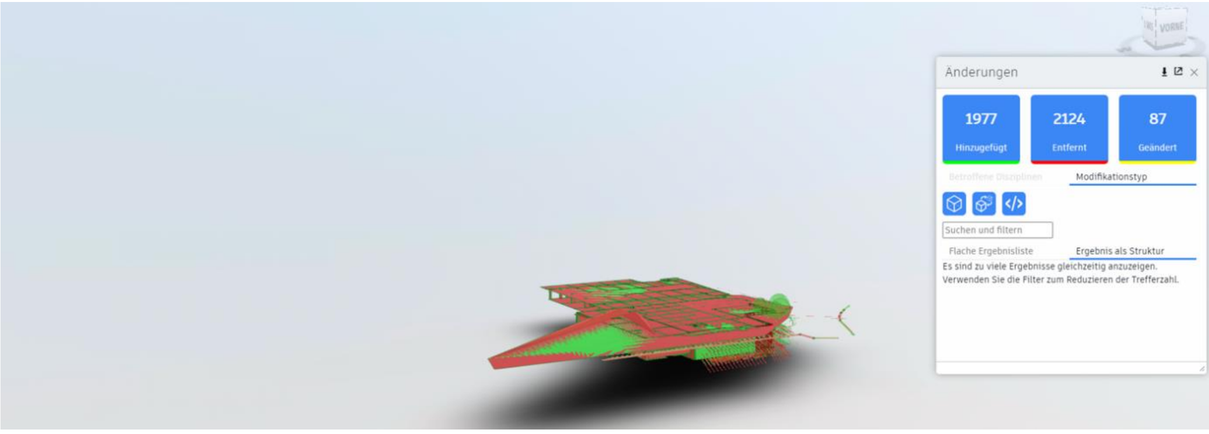


Figure 34 – Unrealistic results from IFC files of model M1 on Autodesk BIM 360.

5.1 Model M1 Results

The two versions of the model were added as an input in IFCcomparison tool and Autodesk BIM 360 docs to evaluate and identify the differences in terms of output of both tools.

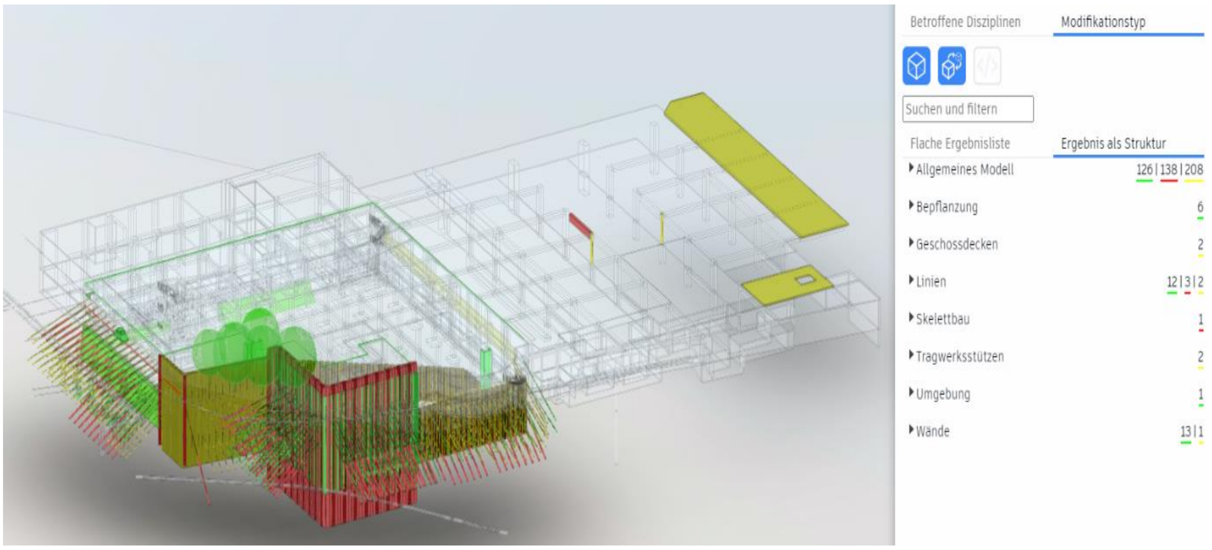
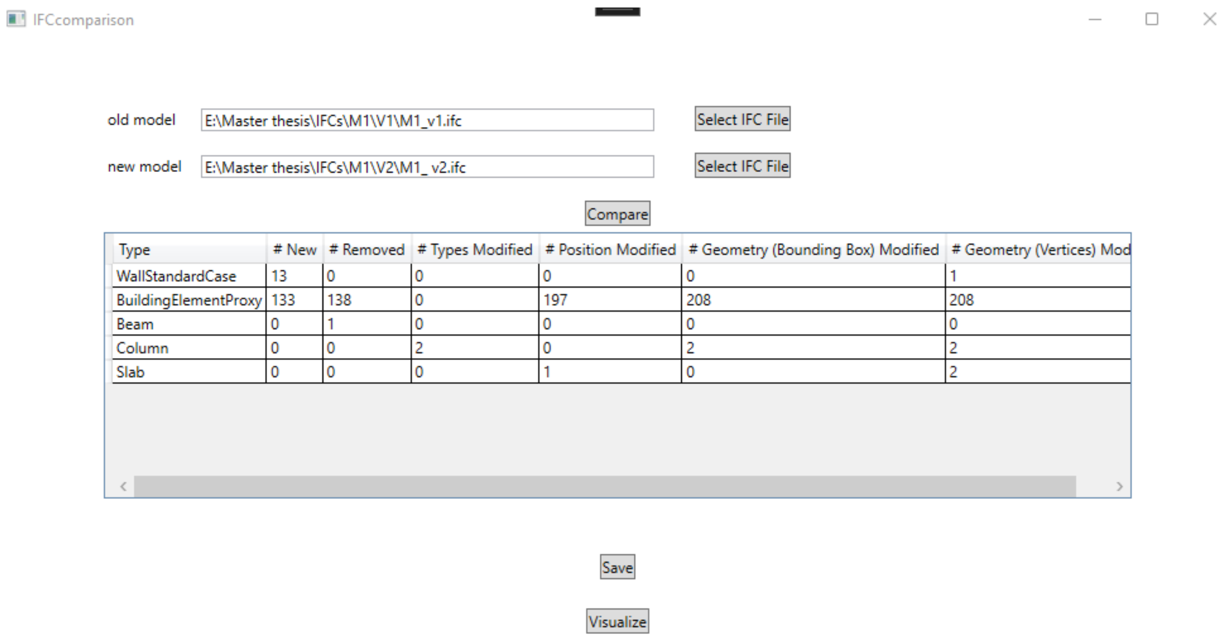


Figure 35 - Output from Autodesk BIM 360 for model M1.

Figure 35 shows the output generated by Autodesk BIM 360 with classification according to the modification type and element category. Additionally, color code is assigned to modified elements to help in visualizing the types of modification where green color represents the new elements, red color represents removed elements, and yellow color represents the modified elements.

Elements with only changes in values of parameters were filtered out as shown in figure 35 since they are irrelevant to the comparison process which aims to report changes in geometry. Hence, the total number of modified elements between the two versions of the model according to Autodesk BIM 360 is 213 excluding lines which is not physical building element.



Type	# New	# Removed	# Types Modified	# Position Modified	# Geometry (Bounding Box) Modified	# Geometry (Vertices) Mod
WallStandardCase	13	0	0	0	0	1
BuildingElementProxy	133	138	0	197	208	208
Beam	0	1	0	0	0	0
Column	0	0	2	0	2	2
Slab	0	0	0	1	0	2

Figure 36 - Output from IFCcomparison tool for model M1.

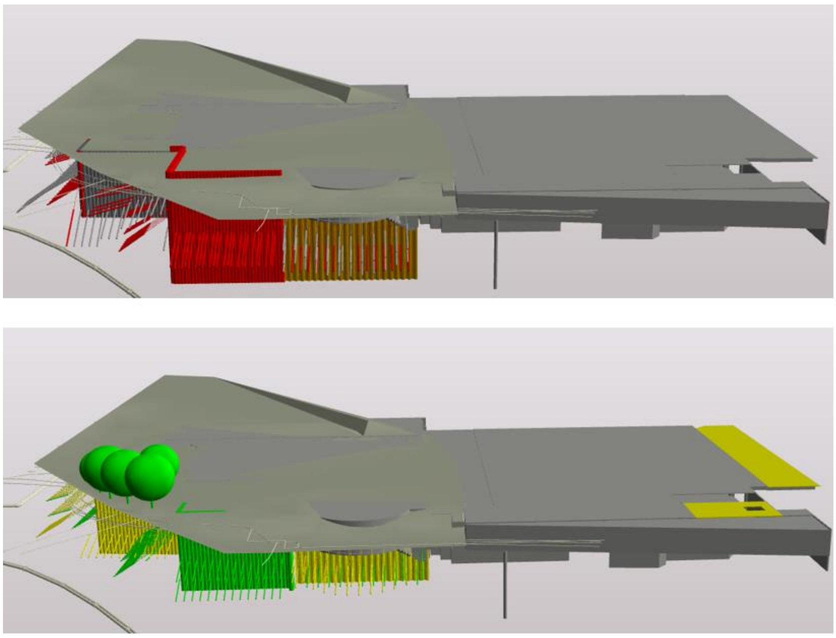
As shown in figure 36, approach 1 reported a total of 210 elements that were changed in terms of geometry between the two versions while approach 2 reported 213 elements which matches exactly the output from Autodesk BIM 360.

	A	B	E	F
1	New	Removed	Geometry Modified (Approach 1)	Geometry Modified (Approach 2)
2	IFCWallStandardCase : 2800748	IFCBeam : 1853155	__1,1,4.81 -> 0.61,0.606,4.82 : 1817671	__Vol Old: 24.90769 m3 -> Vol New: 24.90771 m3 : 1844350
3	IFCWallStandardCase : 2800749	IFCBuildingElementProxy : 2226702	__1,1,4.81 -> 0.61,0.606,4.82 : 1817716	__Vol Old: 42.50602 m3 -> Vol New: 41.84563 m3 : 2178973
4	IFCWallStandardCase : 2800750	IFCBuildingElementProxy : 2240995	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229051	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507741
5	IFCWallStandardCase : 2800751	IFCBuildingElementProxy : 2241000	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229173	__Vol Old: 6.13323 m3 -> Vol New: 6.42237 m3 : 2229051
6	IFCWallStandardCase : 2800752	IFCBuildingElementProxy : 2241004	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229176	__Vol Old: 6.13323 m3 -> Vol New: 6.42237 m3 : 2229173
7	IFCWallStandardCase : 2800753	IFCBuildingElementProxy : 2241008	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229179	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507745
8	IFCWallStandardCase : 2801436	IFCBuildingElementProxy : 2241012	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229182	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507749
9	IFCWallStandardCase : 2801437	IFCBuildingElementProxy : 2241016	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229185	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507754
10	IFCWallStandardCase : 2801438	IFCBuildingElementProxy : 2241024	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229851	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507759
11	IFCWallStandardCase : 2801439	IFCBuildingElementProxy : 2241028	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229893	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507763
12	IFCWallStandardCase : 2801440	IFCBuildingElementProxy : 2241032	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229896	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507767
13	IFCWallStandardCase : 2801441	IFCBuildingElementProxy : 2241036	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229899	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507771
14	IFCWallStandardCase : 2801442	IFCBuildingElementProxy : 2241040	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229902	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507775
15	IFCBuildingElementProxy : 2606291	IFCBuildingElementProxy : 2241044	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229905	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507779
16	IFCBuildingElementProxy : 2609687	IFCBuildingElementProxy : 2241048	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229908	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507783
17	IFCBuildingElementProxy : 2611122	IFCBuildingElementProxy : 2241052	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229911	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507787
18	IFCBuildingElementProxy : 2611204	IFCBuildingElementProxy : 2269559	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229914	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507791
19	IFCBuildingElementProxy : 2611213	IFCBuildingElementProxy : 2269697	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229917	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507795
20	IFCBuildingElementProxy : 2611222	IFCBuildingElementProxy : 2269702	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229920	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507799
21	IFCBuildingElementProxy : 2611231	IFCBuildingElementProxy : 2269706	__0.76,0.76,14.01 -> 0.76,0.76,14.67 : 2229923	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507803
22	IFCBuildingElementProxy : 2611240	IFCBuildingElementProxy : 2269710	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240624	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507807
23	IFCBuildingElementProxy : 2611249	IFCBuildingElementProxy : 2269714	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240629	__Vol Old: 6.13323 m3 -> Vol New: 6.42237 m3 : 2229176
24	IFCBuildingElementProxy : 2611258	IFCBuildingElementProxy : 2269718	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240634	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507811
25	IFCBuildingElementProxy : 2611267	IFCBuildingElementProxy : 2269723	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240638	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507815
26	IFCBuildingElementProxy : 2611276	IFCBuildingElementProxy : 2269728	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240642	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507819
27	IFCBuildingElementProxy : 2611285	IFCBuildingElementProxy : 2269732	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240646	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507823
28	IFCBuildingElementProxy : 2611294	IFCBuildingElementProxy : 2269736	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240650	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507827
29	IFCBuildingElementProxy : 2611340	IFCBuildingElementProxy : 2269740	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240654	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507831
30	IFCBuildingElementProxy : 2611349	IFCBuildingElementProxy : 2269744	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240658	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507835
31	IFCBuildingElementProxy : 2611358	IFCBuildingElementProxy : 2269748	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240662	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507839
32	IFCBuildingElementProxy : 2611367	IFCBuildingElementProxy : 2270133	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240666	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507843
33	IFCBuildingElementProxy : 2611385	IFCBuildingElementProxy : 2270138	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240670	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507847
34	IFCBuildingElementProxy : 2611440	IFCBuildingElementProxy : 2270142	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240674	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507851
35	IFCBuildingElementProxy : 2611444	IFCBuildingElementProxy : 2270146	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240678	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507855
36	IFCBuildingElementProxy : 2614129	IFCBuildingElementProxy : 2270150	__0.647,0.245,15.51 -> 0.647,0.245,8.66 : 2240682	__Vol Old: 0.13303 m3 -> Vol New: 0.058 m3 : 2507859

Figure 37 - Part of Excel sheet from IFCcomparison tool for model M1.

Figure 37 demonstrates part of the excel sheet from the IFCcomparison tool where details are provided for each type of modification with the associated element id to be able to track the development of every reported element in more updated versions.

Additionally, figure 38 shows the visual output from IFCcomparison tool where elements graphics are overloaded with a specific color scheme to group them according to their modification type. However, elements with no changes between the two versions keep their graphic styles.



Version 1

Version 2

Figure 38 - Visual output from IFCcomparison tool for model M1.

Furthermore, table 3 summarizes the findings of both tools to identify the actual differences.

	IFCcomparison Tool				Autodesk BIM 360		
Type	New	Removed	Geometry Approach 1	Geometry Approach 2	New	Removed	Geometry
Generic Object	133	138	208	208	133	138	208
Wall	13	0	0	1	13	0	1
Beam	0	1	0	0	0	1	0
Slab	0	0	0	2	0	0	2
Column	0	0	2	2	0	0	2
Total	**146**	**139**	**210**	**213**	**146**	**139**	**213**

Table 3 - Summary of results from IFCcomparison tool and Autodesk BIM 360 for model M1.

Clearly, approach 2 provided accurate results compared to Autodesk BIM 360 results as, both reported 213 elements which is the actual difference between the two versions while approach 1 reported 210 with 3 elements missing. After evaluating differences manually, it was found that 210 modifications in this model were mostly changes in the dimensions of elements directly. For example, 208 structural piles were modified in diameter and depth, in that case, the first approach was capable of reporting the change, the same holds for the 2 columns which were reported as their dimension was changed while, the remaining 3 elements kept their dimensions unchanged, but voids were added to them to change their geometric shape. Therefore, they were not reported by approach 1 as they don’t influence the dimensions of the bounding boxes used in the comparison process.

5.2 Model M2 Results

The same comparison procedure was applied for model M2 starting with two model versions as inputs then, the results were extracted from both tools.

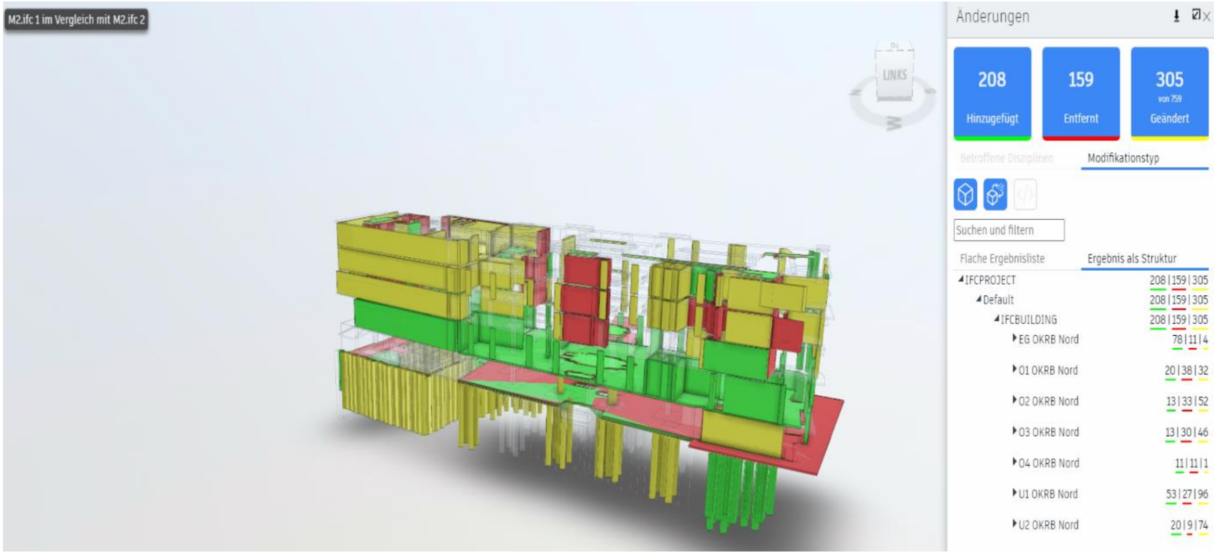
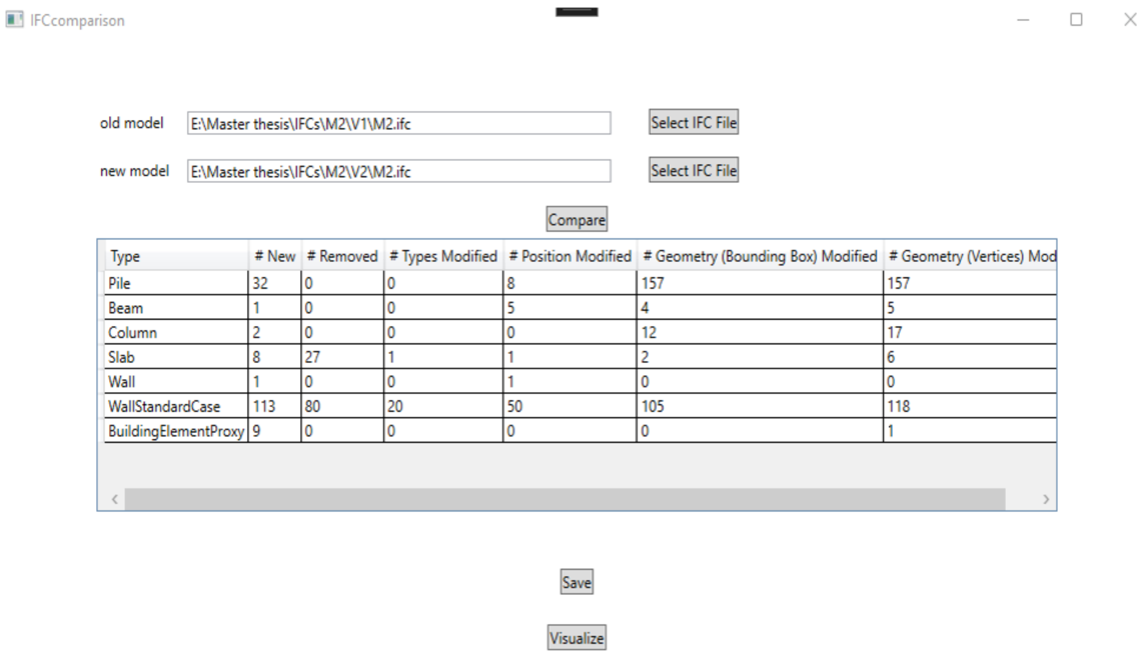


Figure 39 - Output from Autodesk BIM 360 for model M2.

Figure 39 shows the results from Autodesk BIM 360 between the two versions of model M2. Clearly, this model passed through many changes and modifications across all elements categories in terms of geometry with a total of 305 elements modified according to Autodesk BIM 360.



Type	# New	# Removed	# Types Modified	# Position Modified	# Geometry (Bounding Box) Modified	# Geometry (Vertices) Mod
Pile	32	0	0	8	157	157
Beam	1	0	0	5	4	5
Column	2	0	0	0	12	17
Slab	8	27	1	1	2	6
Wall	1	0	0	1	0	0
WallStandardCase	113	80	20	50	105	118
BuildingElementProxy	9	0	0	0	0	1

Figure 40 - Output from IFCcomparison tool for model M2.

As shown in figure 40, approach 1 reported a total of 280 elements that was changed in terms of geometry between the two versions while approach 2 reported 305 modified elements.

	A	B	E	F	G
1	New	Removed	Geometry Modified	Geometry 2 Modified	Semantically Modified
2	IFCPile : 2540199	IFCSlab : 2408688	__ 0.89,0.885,11.01 -> 0.89,0.885,9.41 : 2435806	__ Vol Old: 8.08476 m3 -> Vol New: 8.08719 m3 : 2293833	__ IFCBuildingElementProxy(2475207) -> IFCBuildingElementProxy(2569492)
3	IFCPile : 2540200	IFCSlab : 2408697	__ 0.76,0.89,11.01 -> 0.76,0.89,9.41 : 2435807	__ Vol Old: 0.56226 m3 -> Vol New: 0.57148 m3 : 2336766	
4	IFCPile : 2540201	IFCSlab : 2408707	__ 0.89,0.885,11.01 -> 0.89,0.885,9.41 : 2435808	__ Vol Old: 13.75598 m3 -> Vol New: 13.58893 m3 : 2295100	
5	IFCPile : 2540202	IFCSlab : 2408946	__ 0.76,0.89,11.01 -> 0.76,0.89,9.41 : 2435809	__ Vol Old: 0.50464 m3 -> Vol New: 0.52099 m3 : 2345387	
6	IFCPile : 2540314	IFCSlab : 2408968	__ 0.76,0.89,11.01 -> 0.76,0.89,9.41 : 2435823	__ Vol Old: 3.00628 m3 -> Vol New: 1.87863 m3 : 2306618	
7	IFCPile : 2541298	IFCSlab : 2409490	__ 0.89,0.885,11.01 -> 0.89,0.885,9.41 : 2435824	__ Vol Old: 6.27771 m3 -> Vol New: 6.13363 m3 : 2358561	
8	IFCPile : 2541444	IFCSlab : 2409497	__ 0.824,0.888,11.01 -> 0.824,0.888,9.01 : 2435825	__ Vol Old: 1.25801 m3 -> Vol New: 1.22231 m3 : 2360487	
9	IFCPile : 2541536	IFCSlab : 2409509	__ 0.89,0.885,11.01 -> 0.89,0.885,9.41 : 2435848	__ Vol Old: 1.33359 m3 -> Vol New: 1.33359 m3 : 2370982	
10	IFCPile : 2541640	IFCSlab : 2409551	__ 0.76,0.89,11.01 -> 0.76,0.89,9.41 : 2435849	__ Vol Old: 186.61461 m3 -> Vol New: 186.60993 m3 : 2401123	
11	IFCPile : 2541778	IFCSlab : 2409559	__ 0.89,0.885,11.01 -> 0.89,0.885,9.41 : 2435850	__ Vol Old: 5.34851 m3 -> Vol New: 5.37304 m3 : 2324936	
12	IFCPile : 2541779	IFCSlab : 2409590	__ 0.856,0.89,11.01 -> 0.856,0.89,9.41 : 2435851	__ Vol Old: 13.82985 m3 -> Vol New: 13.601 m3 : 2327837	
13	IFCPile : 2541780	IFCSlab : 2409602	__ 0.89,0.885,11.01 -> 0.89,0.885,9.01 : 2435878	__ Vol Old: 55.11111 m3 -> Vol New: 72.06837 m3 : 2408237	
14	IFCPile : 2541781	IFCSlab : 2409609	__ 0.76,0.89,11.01 -> 0.76,0.89,9.01 : 2435879	__ Vol Old: 2.91945 m3 -> Vol New: 1.463 m3 : 2306899	
15	IFCPile : 2541782	IFCSlab : 2409617	__ 0.89,0.885,11.01 -> 0.89,0.885,9.01 : 2435880	__ Vol Old: 6.64153 m3 -> Vol New: 5.67549 m3 : 2435806	
16	IFCPile : 2541914	IFCSlab : 2409625	__ 0.76,0.89,11.01 -> 0.76,0.89,9.01 : 2435881	__ Vol Old: 5.78184 m3 -> Vol New: 4.94084 m3 : 2435807	
17	IFCPile : 2542124	IFCSlab : 2409689	__ 0.89,0.885,11.01 -> 0.89,0.885,9.01 : 2435882	__ Vol Old: 6.64153 m3 -> Vol New: 5.67549 m3 : 2435808	
18	IFCPile : 2542705	IFCSlab : 2409696	__ 0.76,0.89,11.01 -> 0.76,0.89,9.01 : 2435883	__ Vol Old: 5.78184 m3 -> Vol New: 4.94084 m3 : 2435809	
19	IFCPile : 2542835	IFCSlab : 2409705	__ 0.89,0.885,11.01 -> 0.89,0.885,9.01 : 2435884	__ Vol Old: 5.78184 m3 -> Vol New: 4.94084 m3 : 2435823	
20	IFCPile : 2542927	IFCSlab : 2409791	__ 0.76,0.89,11.01 -> 0.76,0.89,9.01 : 2435885	__ Vol Old: 6.64153 m3 -> Vol New: 5.67549 m3 : 2435824	
21	IFCPile : 2543019	IFCSlab : 2409828	__ 0.89,0.885,11.01 -> 0.89,0.885,9.01 : 2435886	__ Vol Old: 5.78166 m3 -> Vol New: 4.73045 m3 : 2435825	
22	IFCPile : 2543111	IFCSlab : 2409837	__ 0.76,0.89,11.01 -> 0.76,0.89,9.01 : 2435887	__ Vol Old: 6.64153 m3 -> Vol New: 5.67549 m3 : 2435848	
23	IFCPile : 2543203	IFCSlab : 2409869	__ 0.89,0.885,11.01 -> 0.89,0.885,9.01 : 2435888	__ Vol Old: 5.78184 m3 -> Vol New: 4.94084 m3 : 2435849	
24	IFCPile : 2543295	IFCSlab : 2409889	__ 0.76,0.89,11.01 -> 0.76,0.89,9.01 : 2435889	__ Vol Old: 6.64153 m3 -> Vol New: 5.67549 m3 : 2435850	
25	IFCPile : 2543387	IFCSlab : 2409907	__ 0.89,0.885,11.01 -> 0.89,0.885,9.01 : 2435890	__ Vol Old: 5.7818 m3 -> Vol New: 4.94081 m3 : 2435851	
26	IFCPile : 2543479	IFCSlab : 2409917	__ 0.76,0.89,11.01 -> 0.76,0.89,9.01 : 2435891	__ Vol Old: 6.64153 m3 -> Vol New: 5.43398 m3 : 2435878	
27	IFCPile : 2543597	IFCSlab : 2409926	__ 0.89,0.885,11.01 -> 0.89,0.885,9.01 : 2435892	__ Vol Old: 5.78184 m3 -> Vol New: 4.7306 m3 : 2435879	
28	IFCPile : 2543758	IFCSlab : 2459912	__ 0.76,0.89,11.01 -> 0.76,0.89,9.01 : 2435893	__ Vol Old: 6.64153 m3 -> Vol New: 5.43398 m3 : 2435880	
29	IFCPile : 2543759	IFCWallStandardCase : 2288031	__ 0.89,0.885,11.01 -> 0.89,0.885,9.01 : 2435894	__ Vol Old: 5.78184 m3 -> Vol New: 4.7306 m3 : 2435881	
30	IFCPile : 2543760	IFCWallStandardCase : 2288195	__ 0.813,0.889,11.01 -> 0.813,0.889,9.01 : 2435895	__ Vol Old: 6.64153 m3 -> Vol New: 5.43398 m3 : 2435882	
31	IFCPile : 2543761	IFCWallStandardCase : 2288326	__ 0.89,0.885,11.41 -> 0.89,0.885,9.41 : 2435896	__ Vol Old: 5.78184 m3 -> Vol New: 4.7306 m3 : 2435883	
32	IFCPile : 2543762	IFCWallStandardCase : 2288618	__ 0.76,0.89,11.41 -> 0.76,0.89,9.41 : 2436336	__ Vol Old: 6.64153 m3 -> Vol New: 5.43398 m3 : 2435884	
33	IFCPile : 2546621	IFCWallStandardCase : 2289420	0.89,0.885,11.41 -> 0.89,0.885,9.41 : 2436337	Vol Old: 5.78184 m3 -> Vol New: 4.7306 m3 : 2435885	

Figure 41 - Part of Excel sheet from IFCcomparison tool for model M2.

Figure 41 presents part of the Excel sheet from the IFCcomparison tool where details are provided for each type of modification with the associated element id. In this excel sheet two columns “Type Modified” and “Position Modified” are hidden since they are not relevant for comparing the results of both tools.

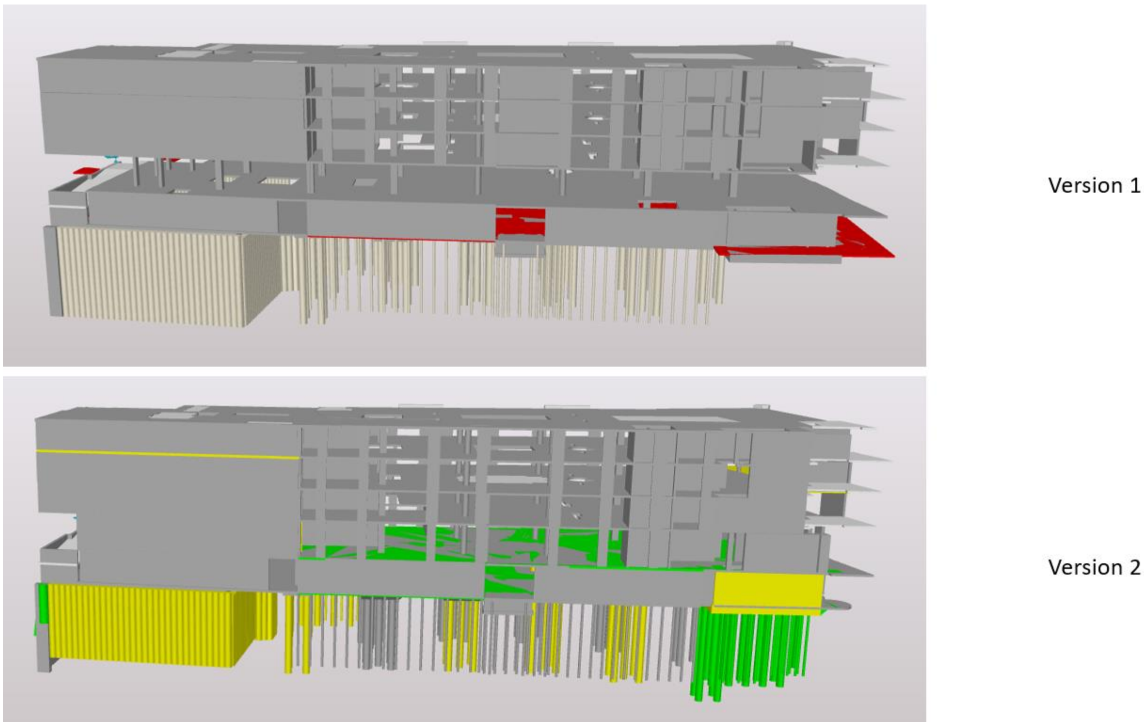


Figure 42 - Visual output from IFCcomparison tool for model M2.

Figure 42 shows the visual output from IFCcomparison tool. In fact, the visual output from the tool is not accurate and, in some cases, does not visually represent all the reported elements as some elements lose their representation reference, hence their graphic styles cannot be overloaded.

Moreover, table 4 summarizes the findings of both tools for the comparison process of model M2.

	IFCcomparison Tool				Autodesk BIM 360		
Type	New	Removed	Geometry Approach 1	Geometry Approach 2	New	Removed	Geometry
Generic Object	41	0	157	158	42	1	158
Wall	114	80	105	118	114	80	118
Beam	1	0	4	5	1	0	5
Slab	8	27	2	7	49	78	7
Column	2	0	12	17	2	0	17
Total	**166**	**107**	**280**	**305**	**208**	**159**	**305**

Table 4 - Summary of results from IFCcomparison tool and Autodesk BIM 360 for model M2.

The results showed differences between both tools regarding the “New” and “Removed” aspects regarding the slabs. To identify which results are right, a manual revision was conducted to calculate the actual numbers for added and removed elements and it was concluded that IFCcomparison tool reported the realistic numbers. GUID changes between the two versions could be the reason behind the false numbers reported by Autodesk BIM 360 regarding those two aspects as shown in figure 43.

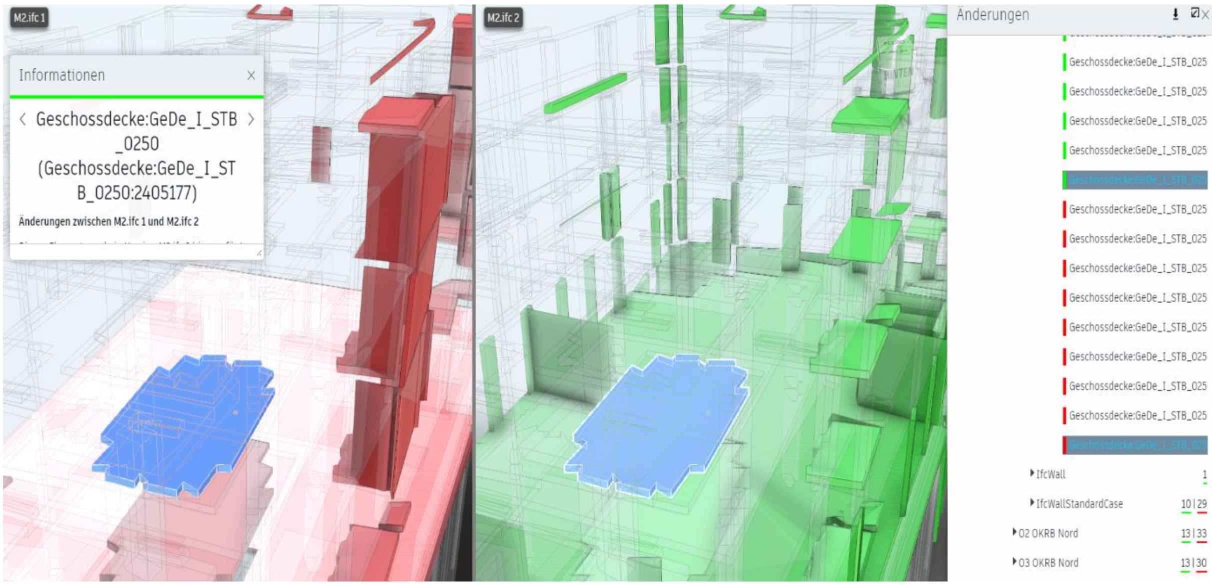


Figure 43 - Unrealistic results due to GUID changes.

Besides the changes in the GUIDs, IFCcomparison tool detected an element replacement case between the two versions where a generic element was removed and another generic element with the same geometry and position was added. On the other hand, Autodesk BIM 360 classified the first element as a removed element and the second element as an added element as shown in figure 44.

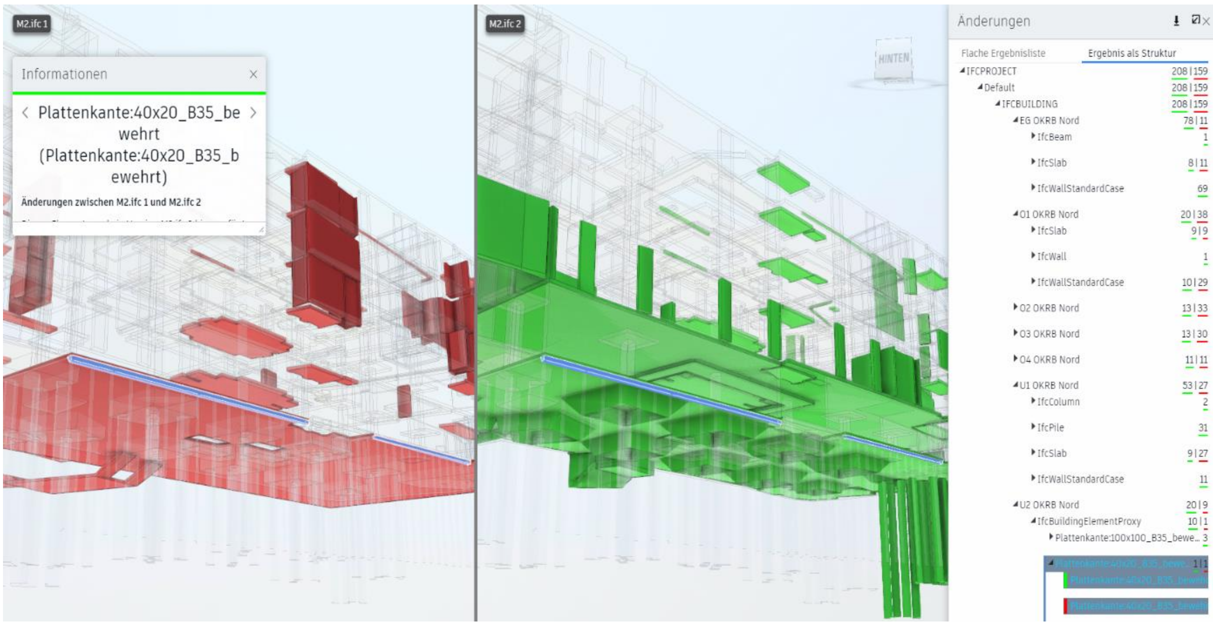


Figure 44 - Element replacement.

Moreover, approach 2 provided exactly the same results as Autodesk BIM 360 regarding geometry modifications. On the other hand, approach 1 reported 280 elements, missing 25 elements as the approach is capable of reporting changes in the dimensions of each element. While the missing ones are cases where voids took place to change the geometric shape of the elements.

6 Discussion on Results

Clearly, the results showed that the second approach can identify and report any kind of geometric modification while it also highlights some limitations of the first approach. Generally, the first approach manages to report direct changes in elements dimensions as these types of changes always affect the size of the bounding boxes but, when it comes to voiding or applying changes regarding the shape of elements through modifying profiles, approach 1 fails to report such cases and this is considered the main limitation of this approach. The following figures represent these cases which were encountered several times while evaluating test models.

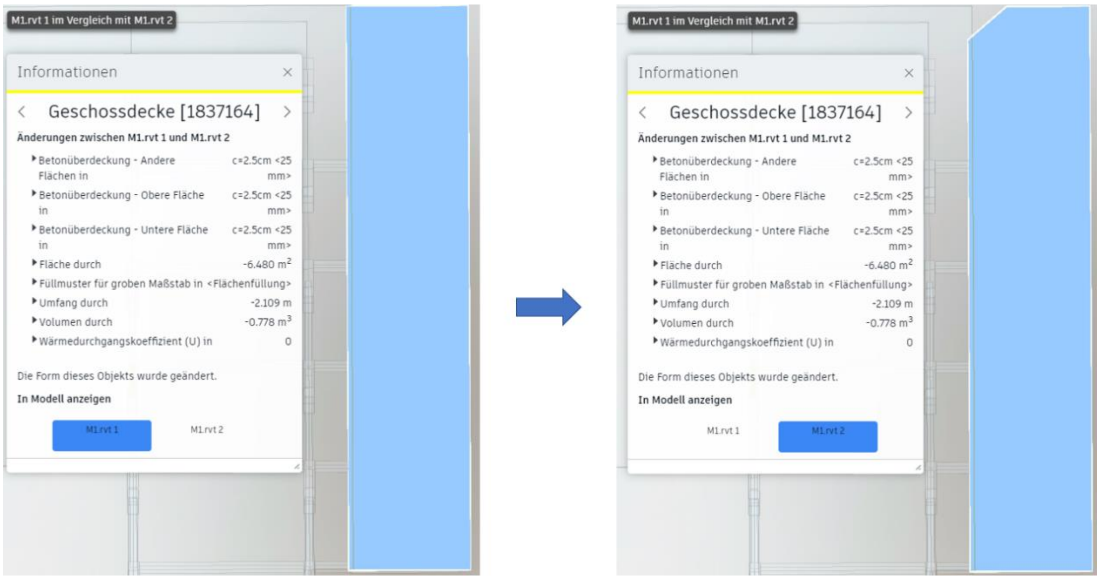


Figure 45 - Example for limitation cases of approach 1 (1/4).

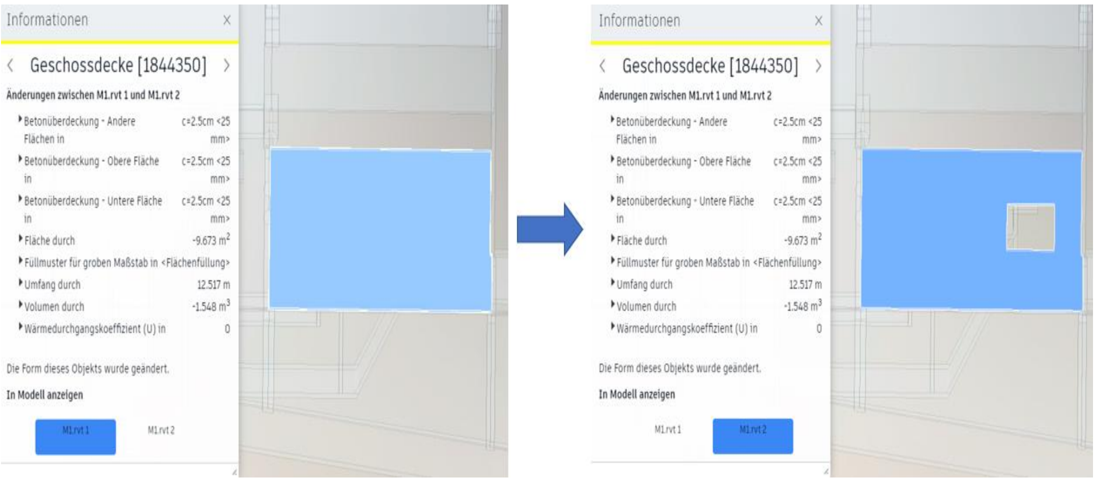


Figure 46 - Example for limitation cases of approach 1 (2/4).

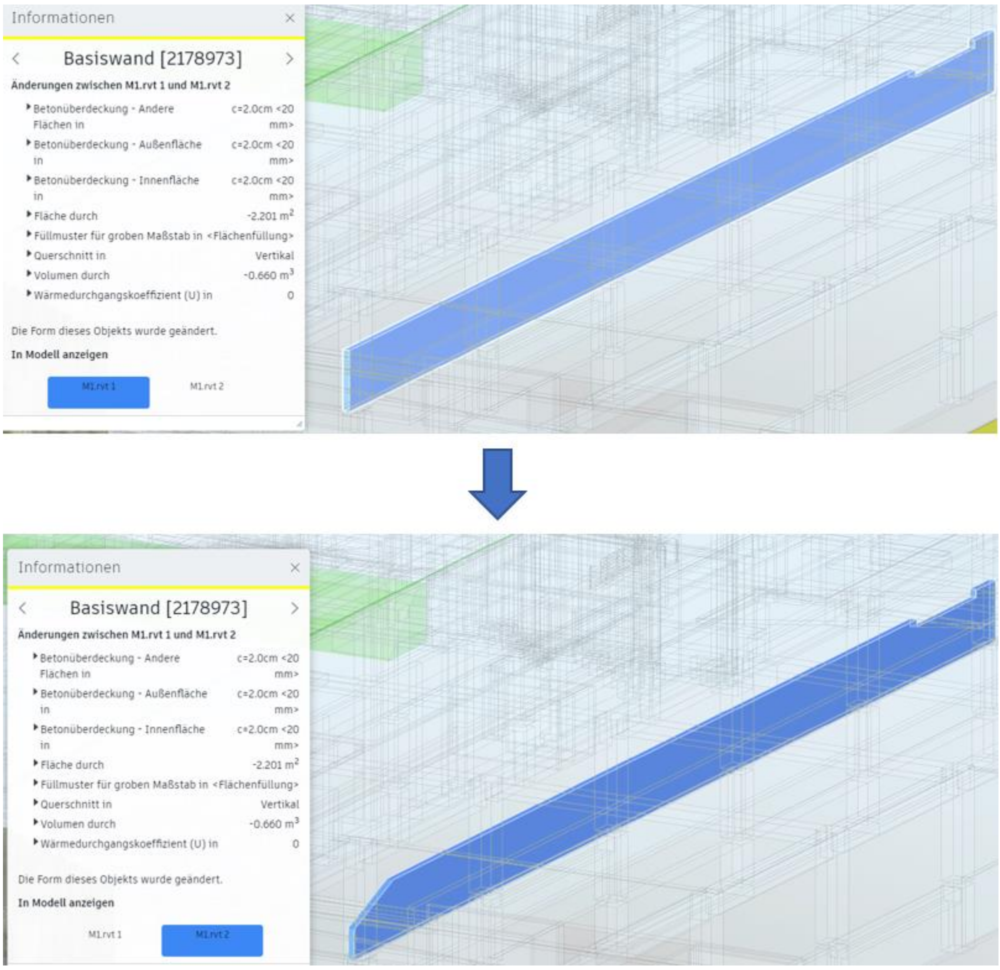


Figure 47 - Example for limitation cases of approach 1 (3/4).

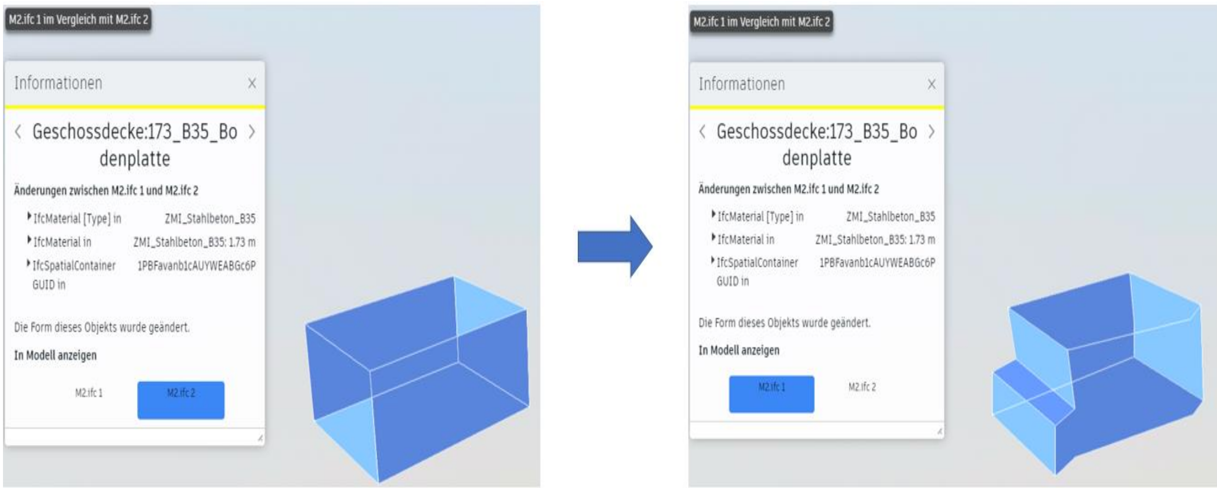


Figure 48 - Example for limitation cases of approach 1 (4/4).

The previous figures show some cases from the test models where the first approach fails to report. Obviously, these cases are the main reason behind the inaccurate results reported by the first approach for both models M1 and M2.

Generally, the accuracy of results from IFC models depends on the stability of the model. Typically, an efficient strategy is required to filter out redundant instances that appear in some IFC models to ensure that the model is stable, and the accuracy of results is not affected, these cases are frequent and can affect the accuracy of both approaches.

7 Conclusion

The aim of this research was to identify effective methods for evaluating and comparing the geometry of elements for versioning applications in BIM models. Based on the experiments conducted, it can be concluded that the mesh comparison method (second approach) has proved to be practical and powerful for evaluating geometry in BIM models. Based on the accuracy of results from this approach, it can serve as a standard and general approach for version control tools.

Moreover, the bounding box comparison method (first approach) is limited to certain use cases, it showed up high accuracy for reporting elements with direct changes in dimensions, but for other cases where voiding or modification of shapes takes place, it provides unreliable results. Hence, this method is recommended to be used for evaluating differences in BIM models with low LOD levels where these cases are not common.

For future development, an efficient strategy for filtering out the redundant instances in IFC models needs to be implemented in the tool to ensure that the results remain reliable among different BIM models.

Enhancing results presentation could be another development as the actual output from the second approach is expressed per volume for solids, and per area for surfaces, this might be not enough for users seeking to know the actual difference between two elements among two versions. Hence, new ideas for presenting results for this approach could provide the user with a better understanding of the changes that occurred.

Finally, implementing the element replacement detection feature for the second approach could enhance the output of the tool. Currently, this feature is only implemented in the first approach, and it is more complicated to integrate it in the second approach.

References

Abualdenien, J., & Borrmann, A. (2019). A meta-model approach for formal specification and consistent management of multi-LOD building models. *Advanced Engineering Informatics*, *40*, 135-153 (Accessed on 01.04.2022).

Autodesk BIM 360 (2022) 'Module: Docs'.

Berwald, S. (2008). From CAD to BIM: The Experience of Architectural Education with Building Information Modeling. in AEI 2008: Building Integration Solutions. doi: 10.1061/41002(328)8 (Accessed on 10.06.2022).

Bew, M., & Richards, M. (2008). BIM Maturity Model. Paper presented at the Construct IT Autumn 2008 Members' Meeting. Brighton, UK (Accessed on 10.06.2022).

BIM Vision (2022) 'Module: Compare'.

BIMForum. 2016. "Level of Development Specification". Available at: LOD Spec 2016 Part I (bimforum.org) (Accessed on 25.04.2022).

Blischak, J. D., Davenport, E. R., & Wilson, G. (2016). A quick introduction to version control with Git and GitHub. *PLoS computational biology*, *12*(1), e1004668 (Accessed on 15.06.2022).

Borrmann, A., Beetz, J., Koch, C., Liebich, T., & Muhic, S. (2018). Industry foundation classes: A standardized data model for the vendor-neutral exchange of digital building models. In *Building information modeling* (pp. 81-126). Springer, Cham (Accessed on 01.04.2022).

BuildingSMART (2013). Industry Foundation Classes, Version 4, Documentation. Available at: http://www.buildingsmart-tech.org/ifc/IFC4/final/html/ (Accessed on 01.04.2022).

BuildingSMART (2013a). IFC and related data model standards documentation. Available at: http://www.buildingsmart-tech.org/ (Accessed on 20.04.2022).

BuildingSMART (2022). The EXPRESS Definition Language for IFC Development. Available at: https://standards.buildingsmart.org/documents/Implementation/The_EXPRESS_Definition_Language_for_IFC_Development.pdf (Accessed on 01.04.2022).

Dakhil, A., Alshawi, M., & Underwood, J. (2015, June). BIM client maturity: literature review. In *12th International Post-Graduate Research Conference* (Accessed on 20.06.2022).

Esser, S., Vilgertshofer, S., & Borrmann, A. (2021, June). Graph-based version control for asynchronous BIM level 3 collaboration. In *Proceedings of the EG-ICE 2021 Workshop on Intelligent Computing in Engineering, Berlin, Germany* (pp. 98-107) (Accessed on 01.04.2022).

ISO (2004) ISO 10303-11:2004: Industrial automation systems and integration - Product data representation and exchange - Part 21: Implementation methods: Clear text encoding of the exchange structure (ISO 10303-11:1994).

Laakso, M., & Kiviniemi, A. O. (2012). The IFC standard: A review of history, development, and standardization, information technology. *ITcon*, *17*(9), 134-161 (Accessed on 18.05.2022).

Mekawy, M., & Petzold, F. (2018). BIM-based model checking in the early design phases of precast concrete structures (Accessed on 10.07.2022).

Muller, M. F., Garbers, A., Esmanioto, F., Huber, N., Loures, E. R., & Canciglieri, O. (2017). Data interoperability assessment though IFC for BIM in structural design–a five-year gap analysis. *Journal of Civil Engineering and Management*, *23*(7), 943-954 (Accessed on 11.08.2022).

Noardo, F., Arroyo Ohori, K., Krijnen, T., & Stoter, J. (2021). An inspection of IFC models from practice. *Applied Sciences*, *11*(5), 2232 (Accessed on 25.06.2022).

Pillay, N., Musonda, I., & Makabate, C. (2018). Use of BIM at higher learning institutions: Evaluating the level of implementation and development of BIM at built environment schools in South Africa. *Singapore: AUBEA*, 227-240 (Accessed on 10.08.2022).

Shi, X., Liu, Y. S., Gao, G., Gu, M., & Li, H. (2018). IFCdiff: A content-based automatic comparison approach for IFC files. *Automation in Construction*, *86*, 53-68 (Accessed on 20.04.2022).

Succar, B. (2015) BIM ThinkSpace: BIM Maturity, BIM Think Space. Available at: http://www.bimthinkspace.com/bim-maturity/ (Accessed on 15.07.2022).

UsBIM (2022) ‘Module: Compare’.

Zada, A. J., Tizani, W., & Oti, A. H. (2014). Building information modelling (BIM)—Versioning for collaborative design (Accessed on 12.04.2022).

Zhu, J., Wu, P., Chen, M., Kim, M. J., Wang, X., & Fang, T. (2020). Automatically processing IFC clipping representation for BIM and GIS integration at the process level. *Applied sciences*, *10*(6), 2009 (Accessed on 01.06.2022).

Zolkifli, N. N., Ngah, A., & Deraman, A. (2018). Version control system: A review. *Procedia Computer Science*, *135*, 408-415 (Accessed on 10.08.2022).

Appendix

The listed items are included in the submission file:

- The original and redacted PDF files
- Title of the thesis written in English and German in a text file
- Abstract in a text file
- Source code for the developed tool
- Excel sheets representing the results from the test models
- Figures used in the thesis

Declaration

I hereby declare that I have completed this Master thesis independently. Only the sources and tools explicitly named in the thesis were used. I have marked as such any ideas that have been taken over literally or analogously.

I also assure you that this thesis has not yet been the subject of any other examination procedure.

München, 27. September 2022

Ahmed Mohamed

Ahmed Mohamed

[redacted]