

**TECHNISCHE
UNIVERSITÄT
DRESDEN**

Institute of Construction Informatics, Faculty of Civil Engineering

BIM Model Enrichment for Energy Performance Simulations

by

Khyati Pravin Patel

from

Vadodara, India

Master Thesis submitted to the Faculty of Civil Engineering, Institute of Construction
Informatics of the University of Technology Dresden in partial fulfillment of the requirements
for the degree of

Master of Science

Responsible Professor: Prof. Dr.-Ing. habil. Karsten Menzel

Second Examiner: Prof. Dr.-Ing. Rainer Scherer

Advisor: Janakiram Karlapudi, MSc; Dipl.-Ing. Johannes Frank Schüler

Dresden, April, 2020

Declaration

I confirm that this assignment is my own work and that I have not sought or used the inadmissible help of third parties to produce this work. I have fully referenced and used inverted commas for all text directly quoted from a source. Any indirect quotations have been duly marked as such.

This work has not yet been submitted to another examination institution – neither in Germany nor outside Germany – neither in the same nor in a similar way and has not yet been published.

Dresden,

Place, Date

(Signature)

Acknowledgment

I would like to take this opportunity to thank everyone who helped me complete my thesis. This thesis would not have been possible but for the input of others in directing me and helping me to refocus on the end goal.

I would like to thank Professor Karsten Menzel for providing me with the opportunity to work on this thesis and for his support. I would also like to thank Mr. Janakiram Karlapudi and Mr. Johannes Frank Schüler, my supervisors, for their time, guidance, and valuable advice.

Finally, I would like to thank my family and friends for their continuous support and patience throughout the year and my academic life.

Abstract

The need to perform building energy simulations is increasing due to the escalating environmental crisis. The data required to perform these simulations need to be acquired from other stakeholders of the project. With the increasing usage of BIM throughout the AEC industry, the exchange of data between the stakeholders has improved significantly throughout the life-cycle of the project. BIM-based energy performance assessments are being performed. However, one of the fundamental issues encountered is data interoperability between BIM and BEM. Inaccurate geometry creation, software not being able to read the data, and incorrect mapping of data are some of the major interoperability issues faced. To address such issues, a python script is developed to translate accurately and precisely the data from IFC to IDF (EnergyPlus Input Data File). The python script ensures that the data read from IFC is appropriately converted to EnergyPlus readable format. This not only improves data translation accuracy but also minimizes the loss of information. The python script also takes into account the additional data (space occupancy data) that is modeled in ArchiCAD which is then exported to IFC. The thesis demonstrates the potentials and possibilities of the developed script with respect to geometry creation, material assignment, etc. Energy performance for model with and without internal gain (space occupancy) is performed. The quantity and accuracy of data available is vital to obtain reliable energy performance of a project.

Table of Contents

Declaration.....	iv
Acknowledgment.....	v
Abstract.....	vi
List of Tables	ix
List of Figures.....	x
List of Abbreviations	xii
1. Introduction	1
1.1 Motivation and Objective	2
1.2 Organization of Thesis.....	3
2. Background and Related Work.....	4
2.1 BIM.....	4
2.1.1 Level of Development.....	5
2.1.2 Life-Cycle Stages.....	7
2.2 BEM.....	9
2.3 Interoperability.....	12
2.3.1 Industry Foundation Class	12
2.3.2 Model View Definition (MVD).....	14
2.3.3 IfcDoc	15
2.3.4 Interoperability Gap.....	15
3. Research Approach.....	18
4. Data Requirements and Model Preparation.....	19
4.1 EnergyPlus Data Requirement.....	19
4.2 ArchiCAD Model	20
5. Script Development and Validation.....	24
5.1 Data Mapping	24
5.1.1 IFC Entity Relationships.....	24
5.1.2 IFC Validation	25
5.1.3 Mapping IFC and IDF.....	25
5.2 Data Extraction	26
5.3 IDF Validation	36
6. Building Energy Performance	39
6.1 Simulation Results	39

6.2	Enriched Model Simulation	43
6.2.1	BIM Enrichment	43
6.2.2	Enriched Model Simulation Results	48
7.	Conclusion	52
8.	Future Work.....	53
9.	References	54
10.	Appendices	57
A.	ArchiCAD Model	57
A.1	Model Data.....	57
A.2	IFC Export Settings.....	59
B.	Class Object – Detailed Input	60
B.1	Input Data.....	60
B.2	Simulation Class Objects	62
B.3	HVAC Class Objects.....	63
B.4	People and Schedule Class Objects.....	68
C.	Simulation Results	70
D.	Python Script.....	74
D.1	Building.....	74
D.2	Site:Location.....	74
D.3	Zone	74
D.4	Material	75
D.5	Construction.....	76
D.6	BuildingSurface and Shading.....	77
D.7	FenestrationSurface.....	79
D.8	People.....	81
D.9	Schedule.....	81

List of Tables

Table 1: Spaces at different LOD [9].....	6
Table 2: IDF-IFC Mapping.....	25
Table 3: Site and Source Energy.....	43
Table 4: Site and Source Energy (Enriched Model)	51

List of Figures

Figure 1: Traditional Method vs. BIM [2]	1
Figure 2: Effects of BIM over Life-Cycle Stages [8]	4
Figure 3: Life-cycle Phases [11]	7
Figure 4: Architectural (Left) and Thermal Model (Right)	10
Figure 5: Space Boundaries Data Exchange [15]	11
Figure 6: IFC conceptual layers [17]	13
Figure 7: IFC Structure of a Wall [10].....	14
Figure 8: Data Exchange Schema and MVD.....	15
Figure 9: SBT-1 Mapping Process [20].....	16
Figure 10: Methodology	18
Figure 11: ArchiCAD BIM Model	21
Figure 12: Floor Plans.....	21
Figure 13: Zone Volume.....	22
Figure 14: North in ArchiCAD and EnergyPlus.....	23
Figure 15: IFC information model [25]	23
Figure 16: <i>IfcWall</i>	24
Figure 17: IFC Validation.....	25
Figure 18: Process flow for 'Building' data extraction.....	27
Figure 19: 'Building' Output	27
Figure 20: Process flow for 'Site:Location' data extraction	28
Figure 21: 'Site:Location' Output.....	28
Figure 22: Process flow for 'Material' data extraction	29
Figure 23: 'Material' Output.....	30
Figure 24: Process flow for 'Construction' data extraction	30
Figure 25: 'Construction' Output.....	31
Figure 26: Process flow for 'Zone' data extraction	32
Figure 27: 'Zone' Output	32
Figure 28: Process Flow for 'BuildingSurface' data extraction.....	33
Figure 29: 'Building:Surface' Output	34
Figure 30: Process flow for 'FenestrationSurface' data extraction.....	35
Figure 31: 'FenestrationSurface' Output	35
Figure 32: IDF Geometry in OpenStudio	36
Figure 33: IDF Geometry in OpenStudio (Outward Normal Render)	37
Figure 34: IDF Geometry in OpenStudio (Outward Normal Render) – Incorrect.....	37

Figure 35: Material Schedule (ArchiCAD).....	38
Figure 36: 'Materials' IDF Class Objects	38
Figure 37: Fossil Fuel Reduction [26]	39
Figure 38: DXF Drawing including Shading Element	40
Figure 39: Air Temperature - Meeting 22 (HVAC off)	41
Figure 40: Zone Air Temperature - Meeting 22 (HVAC on)	42
Figure 41: Heating Load.....	42
Figure 42: Cooling Load.....	43
Figure 43: Slab Dimension Property Set	44
Figure 44: Process Flow for Slab Dimensions Extraction	45
Figure 45: 'BuildingSurface' Output (Floor Slab).....	46
Figure 46: Space Occupancy Property Set.....	46
Figure 47: Process Flow for 'People' and 'Schedule:Compact' data extraction.....	47
Figure 48: 'People' Output	48
Figure 49: 'Schedule' Output.....	48
Figure 50: Zone Air Temperature - Meeting 22 (with and without 'People')	49
Figure 51: People Heating Energy and Heating Load	50
Figure 52: People Heating Energy and Cooling Load.....	50
Figure 53: Ground Floor Dimensions	57
Figure 54: First Floor Dimensions.....	57
Figure 55: Second Floor Dimensions	58
Figure 56: Modeled Data (Building Element and Properties)	58
Figure 57: IFC Export Settings	59
Figure 58: Geometry and Data Conversion (IFC Export Settings).....	59
Figure 59: Zone Air Temperature - Laboratory 01 (HVAC off)	70
Figure 60: Zone Air Temperature - Laboratory 01 (HVAC on)	70
Figure 61: Zone Air Temperature - Laboratory 01 (with and without 'People')	71
Figure 62: Zone Air Temperature - Office 12 (HVAC off).....	71
Figure 63: Zone Air Temperature - Office 12 (HVAC on).....	72
Figure 64: Zone Air Temperature - Office 12 (with and without 'People').....	72
Figure 65: Heating Load (with and without 'People')	73
Figure 66: Cooling Load (with and without 'People').....	73

List of Abbreviations

AEC	Architecture, Engineering, and Construction
ASHRAE	American Society of Heating, Refrigerating and Air-Conditioning Engineers
BEM	Building Energy Model
BIM	Building Information Modeling
BPIE	Building Performance Institute Europe
CO	Change Order
CO ₂	Carbon Dioxide
EU	European Union
gbXML	Green Building Extensible Markup Language
GST	Geometry Simplification Tool
HTML	Hypertext Markup Language
HVAC	Heating, Ventilation, and Air-Conditioning
IDF	Input Data File
IFC	Industry Foundation Classes
LC	Life-Cycle
LOD	Level of Development
MVD	Model View Definition
Pset	Property Set
SBT	Space Boundary Tool

1. Introduction

Building Information Modeling (BIM) is one of the recent developments in the Architecture, Engineering, and Construction (AEC) industry that has shown potential for economic and sustainable construction by providing design and analysis support throughout the life-cycle (LC) of the project. BIM encourages an integrated approach by collaborating with different stakeholders involved in the AEC industry throughout the LC of the project.

The traditional method of a construction project is fragmented and is quickly becoming obsolete with the introduction of BIM which provides a more integrated approach. The US National Building Information Model Standard Project Committee defines BIM as “a digital representation of physical and functional characteristics of a facility. A BIM is a shared knowledge resource for information about a facility forming a reliable basis for decisions during its life-cycle; defined as existing from earliest conception to demolition” [1]. The primary purpose of BIM is to ensure that the required data, quantitatively as well as qualitatively, is available for the concerned stakeholder in a suitable format. This is essential to make better decisions.

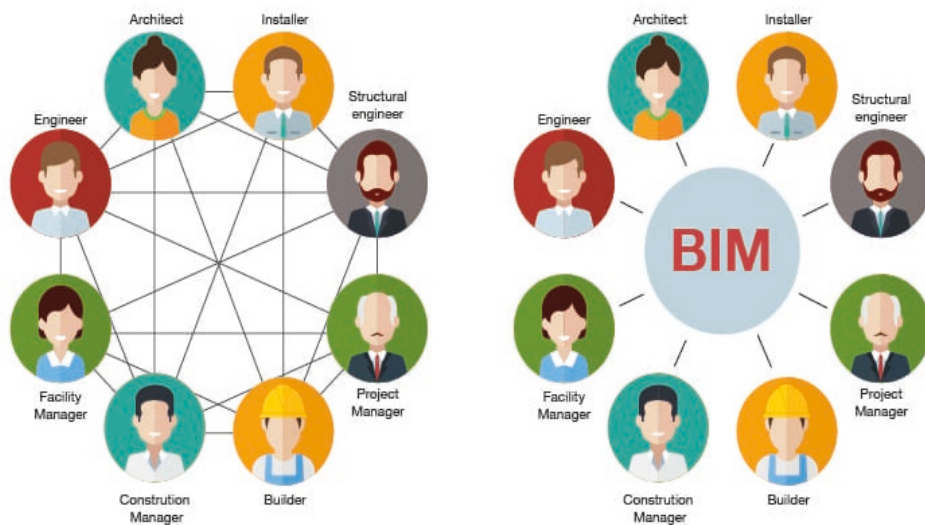


Figure 1: Traditional Method vs. BIM [2]

BIM is being implemented by firms across the AEC industry. BIM not only improves the coordination and communication of the involved stakeholders but it also improves productivity and quality control of the project. The usage of BIM tools during building energy performance analysis may prove to be financially beneficial due to improved communication and reduction in time consumption.

Building energy performance needs to be significantly improved to reduce energy consumption and CO₂ emissions. According to a report by Building Performance Institute Europe (BPIE), 50% of the EU's final energy consumption is used for heating and cooling of which 80% is used in buildings [3]. Since 40% of the final energy consumption is due to the building sector, reducing the energy consumption in buildings will drastically reduce the final energy consumption in the EU which will help achieve the defined energy goals. Under the Paris Agreement, the EU strives to achieve the goal of reduction of greenhouse gas emissions by at least 40% by 2030 [4]. In November 2018, the European Commission published its long-term 2050 goals of reducing greenhouse gas emissions between 80% - 100% compared to the levels in 1990. The long-term goal aims to achieve a climate-neutral economy by 2050 [5].

The building energy performance for a project can be monitored over the life-cycle of the building using the BIM tools. The most significant requirement for monitoring is the availability of accurate and essential data at all the life-cycle stages. The absence or inaccuracy of data will lead to unreliable energy assessment of the project.

1.1 Motivation and Objective

With the increasing need for integration and coordination between all the stakeholders of the AEC industry during various life-cycle phases of a project, the data that is provided for analysis to the concerned individual is crucial. The quality and quantity of data that is provided will determine the accuracy of the analysis. The interoperability between different software tools is required to ensure that the data exchanged between different stakeholders is as accurate as possible. There is a lack of complete compatibility and integration between software. [6]

Each organization that is a part of the project uses different software to perform analysis. The communication between different stakeholders is heavily dependent on the data that is exchanged and the interoperability between their software. When importing a file in the simulation software using an exchange file format (for example, IFC or gbXML) which was generated from modeling software, there is a possibility that the data read by the simulation software is wrongly mapped. At times, the data is insufficient or incomprehensible by the simulation engine. This will require the data to be manually entered in the simulation engine even when the data is originally available within the exchange schema. This process of entering data is time consuming and inefficient.

Taking into account the limited interoperability and possibility of a loss of data, this thesis aims to achieve an accurate and efficient translation of data between BIM (ArchiCAD) and BEM (EnergyPlus) software using IFC as an interoperable file format.

1.2 Organization of Thesis

The thesis is divided into eight sections. The first section is the introduction which explains BIM in brief and includes the statistics about the energy consumption by the AEC industry. The section also includes the motivation and objective of this research. The second section covers the background research which explains BIM in detail with respect to Level of Development (LOD) and life-cycle stages of the project with a focus on energy simulation. The section also explains the different types of space boundaries and their importance for energy calculations. The second section also throws light on the concept of interoperability. The third section explains the approach undertaken in this research to overcome the interoperability gap between BIM and BEM software discussed in the previous section. The fourth section explains the BEM software requirements and the modeling method in BIM software to obtain optimum data for extraction for BEM on export to IFC. The fifth section explains the data extraction process for possible BEM data requirements. The sixth section demonstrates the energy simulation results using the extracted data and enrichment of the BIM model and its simulation results. Lastly, the seventh and the eight sections present the conclusion and the future work respectively.

2. Background and Related Work

2.1 BIM

The traditional design process of the involved stakeholders of a project was fragmented and strongly hierarchical whereas the BIM process involves an integrated approach of the concerned teams. The traditional design process is also linear, distinct, segregated, and the information is only acquired on a need-to-know basis whereas the BIM process is concurrent, multi-level, and the information is shared and contributed early on. Using BIM ensures greater efficiencies. The usage of BIM process over the traditional design process is beneficial for all the concerned teams. For example, for the owners of the project, it gives them the ability to choose design options and meet their goals financially and environmentally as the other teams have a better understanding of the owners' desired outcomes. For constructors, BIM allows them to share their expertise of the current techniques during the early phases which results in improved project quality and economic construction phase. Early project phases do have a crucial impact on the future performance of a building throughout its life-cycle. Up to 80% of operational costs, as well as environmental impacts, are determined in an early design phase [7]. Figure 2 below shows the value of decision making (graph 1) and the cost incurred (graph 2) due to changes throughout the life-cycle of a project. Additionally, the relationship between design effort and time for both the traditional design process (graph 4) and the BIM process (graph 3) are visualized.

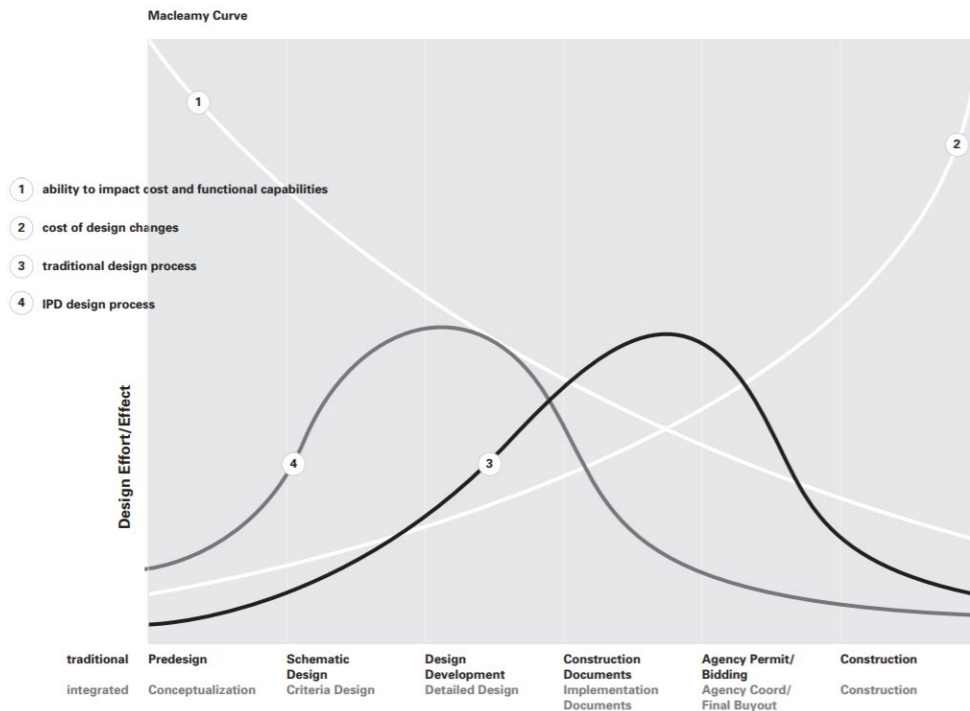


Figure 2: Effects of BIM over Life-Cycle Stages [8]

2.1.1 Level of Development

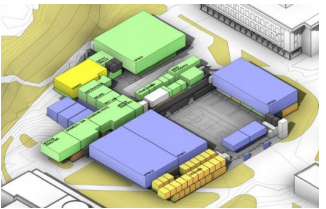
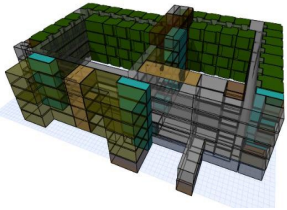
The collaborative nature of BIM requires that there is a clear understanding of the amount detail of the model at any given phase of the project. The different level of development in BIM represents the completeness and the level of refinement of a digital model at a given point in time. LOD sets the minimum amount of geometric, spatial, quantitative, as well as any attribute information necessary for modeling at a particular stage of the life-cycle of the project. Thus, LOD consists of two components: a geometric (mass modeling) LOD and an attributive (operation and maintenance) LOD.

There are six basic levels of development as defined by BIMForum in collaboration with the American Institute of Architects. These definitions are a generic definition guideline of model content and not for any particular software. These definitions need not be strictly adhered to but rather be used as a base guideline to improve communication amongst the stakeholders about the characteristics of the model elements. The levels of development are [9]:

- LOD 100 – The element is graphically represented with a generic representation such as line work is used. The related information of the element may be derived from other elements.
- LOD 200 – At this level, the elements are modeled as generic elements or systems of elements and with approximate quantities, location, orientation, etc. The modeled element acts as a placeholder for more detailed element modeling. LOD 200 can be related to the design phase of a project life-cycle.
- LOD 300 – The model elements represented are specific with respect to the location, size, shape, quantity, and orientation of the project.
- LOD 350 – The model elements are modeled as modeled in LOD 300 along with additional data of the relation between different model elements that are attached to a particular element.
- LOD 400 – This level of development is considered to be suitable for fabrication and assembly as the data available at this level is at sufficient detail and accuracy. The elements modeled are very specific. LOD 400 can be related to the construction phase of a project life-cycle.
- LOD 500 – The final level of development represents the project as it has been constructed - the as-built conditions. The model is suitable for the maintenance and operations of the facility. It is a field verified representation of the project. LOD 500 can be related to the operations phase of a project life-cycle.

Table 1 below shows spaces at different LOD.

Table 1: Spaces at different LOD [9]

LOD	Description	Illustration
100	<p>Spaces are modeled as generic objects with approximate size, shape and location. This level is typically appropriate for design of spatial requirements where space objects are placed either in random manner for quantification or in a 'blocking and stacking' process. Bounding elements are not required but may be needed if specific dimensions are desired.</p> <p>Element modeling to include:</p> <ul style="list-style-type: none"> • Space object based on area required by program or brief 	
200	<p>Spaces are modeled or placed with bounding elements such as walls and columns that are at a minimum of LOD200. Perimeter and area of spaces are calculated with respect to the bounding elements. LOD of spaces shall not exceed the LOD of the bounding elements. For example, if interior partitions are defined at LOD200, the space objects for the project cannot be delivered at LOD300.</p> <p>Element modeling to include:</p> <ul style="list-style-type: none"> • Vertical bounding elements at LOD200 • Space objects that automatically associate with vertical bounding elements 	
300	<p>Spaces are modeled or placed with bounding elements that are at a minimum of LOD300. Perimeter and area of spaces are calculated with respect to the bounding elements.</p> <p>Element modeling to include:</p> <ul style="list-style-type: none"> • Vertical bounding elements at LOD300 • Space objects that automatically associate with vertical bounding elements 	

350	<p>Comply with the LOD300 requirements. Volume of the space is accurately calculated to the nearest horizontal finish surface such as a ceiling or underside of slab above.</p> <p>Element modeling to include:</p> <ul style="list-style-type: none"> • Vertical bounding elements to minimum LOD300 • Horizontal bounding elements such as ceilings or slabs • Space objects that automatically associate with vertical and horizontal bounding elements 	
-----	---	--

To put the definition of different levels of development in perspective, an architect may model a wall system that is highly detailed whereas a contractor may model the same wall as a single component as the sequencing of walls is of importance and system of installation of the wall is trivial [10].

2.1.2 Life-Cycle Stages

The life-cycle stages of a project are broadly classified into three phases. Namely, the design phase, the construction phase, and the operations phase. These phases are further divided into sub-phases as shown in Figure 3.

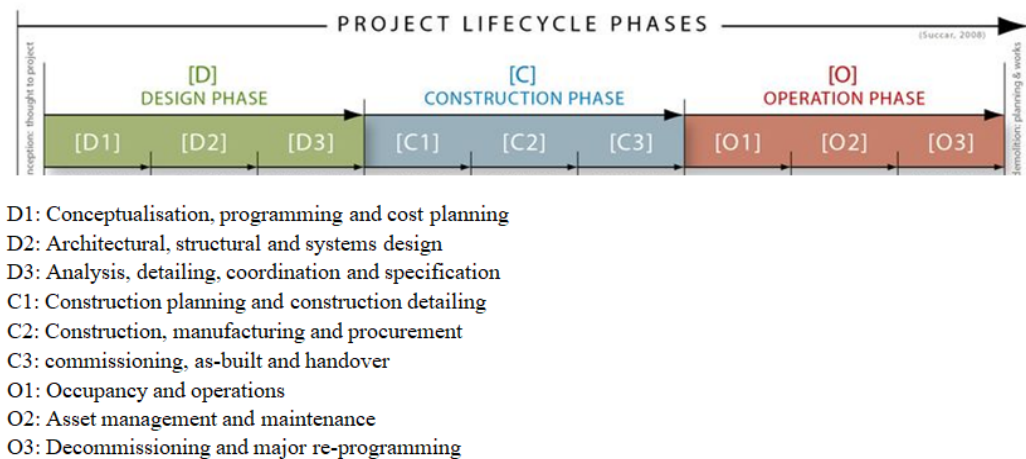


Figure 3: Life-cycle Phases [11]

The whole building energy simulation can be divided into 11 modeling cycles/stages, as defined by ASHRAE Standard 209, corresponding to the life-cycle stages described above. The stages of energy simulation are [12]:

- Simple Box Modeling – This stage identifies and evaluates the energy end uses during

the conceptual design phase. The models are generated using identical HVAC systems for all zones and the loads are analyzed. The energy performance is compared and approximated by varying the building geometry (orientation, size, shape, etc), window-to-wall ratio, shadings.

- Conceptual Design Modeling – The evaluation of energy improvements is carried out in this stage. The model is prepared to evaluate annual building energy by end use with identical HVAC systems. Recommendations are provided to improve the energy performance of each conceptual design option.
- Load Reduction – The purpose of this stage to identify the distribution of energy by end use by evaluating different (at least three) load reduction plans of action. The load can be reduced by implementing strategies from a single category or a combination of the following categories:
 - a. Changing the building envelope
 - b. Daylighting and artificial lighting
 - c. Internal equipment loads
 - d. Outdoor air temperature
 - e. Natural and passive ventilationWhen internal equipment loads exceed 60 percent of the building energy end use, at least two of the strategies shall be selected from the internal equipment loads category
- HVAC System Selection Modeling – Various HVAC system options are analyzed at this stage of energy modeling.
- Design Refinement – The model is prepared to evaluate systems in the building. At least one of the following building system is refined and further developed:
 - a. HVAC system
 - b. Lighting system
 - c. Envelope system
 - d. Service water heating system
 - e. Process and plug load system
- Design Integration and Optimization – This stage aims at integrating building systems by optimization to achieve the desired goals of building energy performance by investigating complex interactions of multiple variables. The energy performance goals are identified and various models are prepared by varying at least two variables that affect the energy performance of a building. An optimization analysis is carried out as well by defining objectives, design variables and design constraints.

- Energy Simulation Aided Value Engineering – To provide information on the holistic implications of value engineering measures on project performance goals to ensure more informed design decisions. The project alternatives are identified from various proposals and the first-cost and operating-cost consequences are identified as well. The energy model is prepared and analyzed for every proposal.
- As-Designed Energy Performance – The final energy model is prepared and analyzed to compare as-designed performance to project goals. The inputs are provided which represent the as-designed configuration of the building systems.
- Change Orders – Feedback is provided on all requests for change orders (CO) at this stage of energy simulation. The energy model is qualitatively reviewed and analyzed for at least one CO and a report is prepared about the impact of the CO on the energy performance. The model is then updated to meet the desired energy performance goal and re-evaluated qualitatively.
- As-Built Performance – An energy model of the as-built building systems with accurate inputs of the as-built conditions due to any changes taking place during the construction process is prepared and analyzed and compared with the project goals. The occupancy and process-dependent schedules and loads shall reflect design phase inputs or should be adjusted to reflect new information.
- Post-Occupancy Energy Performance Comparison – The energy model prepared during the last design- or construction-phase is compared with the actual measured energy use during the operation with actual weather conditions. This is done to improve future energy assumptions while modeling and to identify the potential measures to meet the project energy goals. The analysis for the measured and simulated energy data is performed for typical weather year simulation results.

2.2 BEM

The geometry created for architectural purposes is different than the geometry created for energy analysis. The architectural model, various spaces (rooms) are modeled that are separated by walls whereas, for energy analysis, thermal zones (thermal spaces) are modeled that are separated by thermal space boundaries. These thermal space boundaries may not necessarily be the same as the walls in the architectural model. A large room in an architectural model can be modeled as single or multiple thermal zones, i.e., there can be multiple thermal zones in the energy model representing the large architectural space. Figure 4 shows the difference between the architectural wall boundaries (left) and thermal space boundaries (right) for energy analysis.

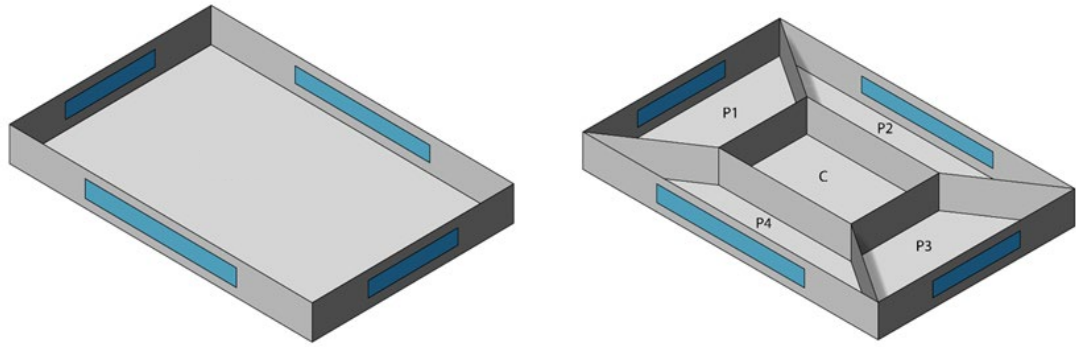


Figure 4: Architectural (Left) and Thermal Model (Right)

A space boundary is a virtual boundary that defines the relationship between spaces and the building element. Space boundaries are of importance for various calculations, namely, energy analysis, lighting calculation, and facility management. In energy analysis, space boundaries are used to estimate the amount of energy that will be used by a building during operation. There are several types of space boundaries that can be distinguished, also known as “levels”. The type of space boundary will estimate energy usage. The type of space boundary affects the amount of heat transfer through the building elements.

The three types of space boundaries are [13], [14]:

- First (1st) Level Space Boundary: These are boundaries of a space defined by the physical surface of the building elements or by providing a virtual boundary by an adjacent space without a wall. 1st level space boundaries do not influence the energy flow (heat transfer) through the modeled building elements to affect the simulation result of the adjacent space.
 - 1st level space boundaries do not take into account any change of material in the bounding building elements, or different spaces/zones behind a wall or slab (floor or ceiling).
 - 1st level space boundaries are differentiated in two ways: virtual or physical and internal/external, or undefined (internal and external).
 - 1st level space boundaries form a closed shell around the space (so long as the space is completely enclosed) and include overlapping boundaries representing openings (filled or not) in the building elements.
- Second (2nd) Level Space Boundary: These boundaries are still defined by the physical or virtual building surfaces separating two zones but the changes in materials/material assemblies or changes in the adjacent zone are taken into account. 2nd level space boundaries influence the energy flow (heat transfer) through the modeled building elements but the adjacent surfaces of the building

element should be found and are combined to form a single heat transfer surface.

- 2nd level space boundaries are differentiated in two ways: virtual or physical and internal or external, whereby any space boundary that is both internal and external has to be split into segments being either
- 2nd level space boundaries form a closed shell around the space (so long as space is completely enclosed).
- Special (2nd) Level Space Boundary: There are two types of special 2nd level space boundaries, namely,
 - Type 2a: This space boundary is represented when space is observed on both sides of the space boundary, i.e., a building element that provides space boundary is common to two spaces.
 - Type 2b: This space boundary is represented when a part of the building element does not play a part in the energy flow (heat transfer) as there is no adjacent space observed. A building element is observed on the other side of the space boundary.

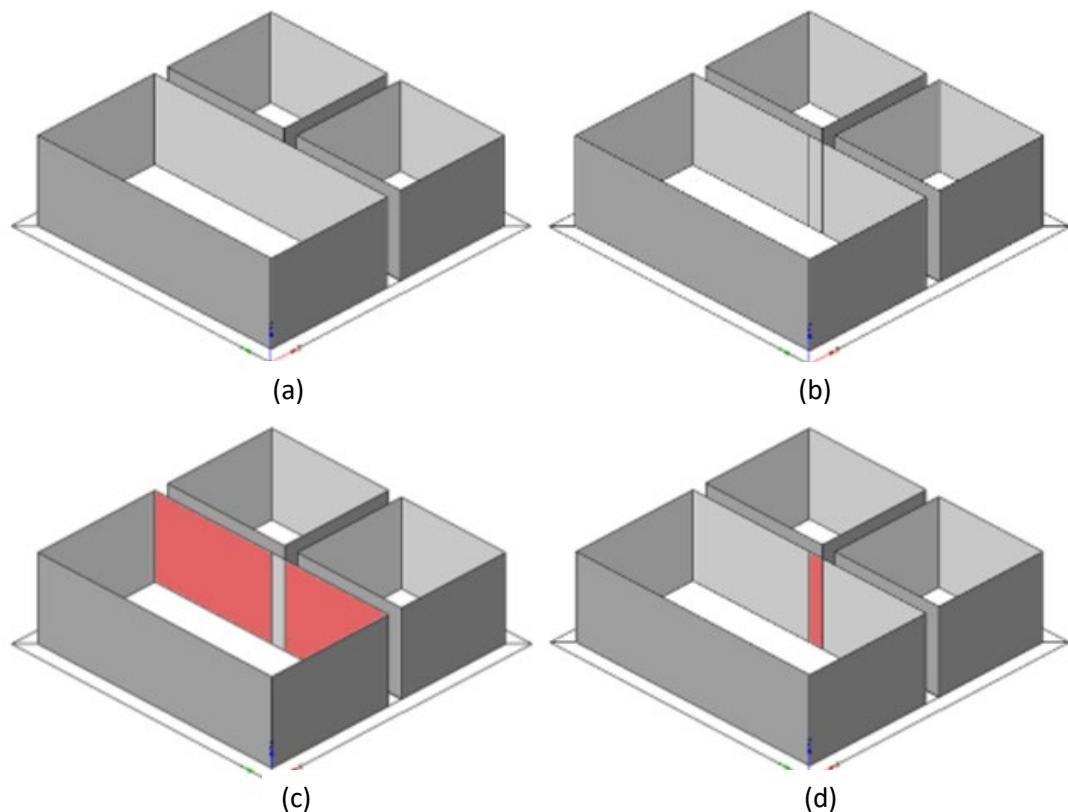


Figure 5: Space Boundaries Data Exchange [15]

2.3 Interoperability

The need for integration of different stakeholders during the life-cycle of the project requires that the data is exchanged efficiently between different software to avoid the time consuming task of preparing the model in every software, certain open data exchange schema have been developed to ease the process. The most commonly used data exchange formats are IFC and gbXML. This thesis uses IFC as the exchange format.

2.3.1 Industry Foundation Class

The Industry Foundation Class (IFC) is an international open standard schema developed for BIM data to be exchanged among various AEC software applications. It is developed based on the ISO-STEP EXPRESS language. IFC has been developed to incorporate building information throughout the life-cycle of the project, from planning through design (including analysis and simulation), construction to post-occupancy and operation and maintenance. “The IFC schema is a standardized data model that codifies, in a logical way...

...the **identity** and **semantics** (*name, machine-readable unique identifier, object type or function*)...

...the **characteristics** or **attributes** (*such as material, color, and thermal properties*)...

...and **relationships** (including locations, connections, and ownership)...

...of **objects** (like columns or slabs)...

...**abstract concepts** (performance, costing)...

...**processes** (installation, operations)...

...and **people** (owners, designers, contractors, suppliers, etc.).” [16]

IFC has been classified into four conceptual layers, namely, the resource layer, the core layer, the interoperability layer, and the domain layer. In the resource layer are the base definitions defining base constructs such as geometry, topology, materials, etc. These definitions do not have a global unique ID (GUID) assigned to them and are not to be used independently. The core layer includes the kernel and core extension schema containing entity definitions. The interoperability layer includes schema that contains entity definition that is specific to a general product, process or resource specialization. The domain layer includes schemas that provide entity definition specific to a certain discipline.

The structure of IFC is hierarchical. All the objects are nested within a subentity definition tree. All physical objects, abstract objects, actors, and other basic constructs are abstractly represented. The properties and quantities of a building element are contained in optional property sets (Pset) and quantity set (Qto) respectively.

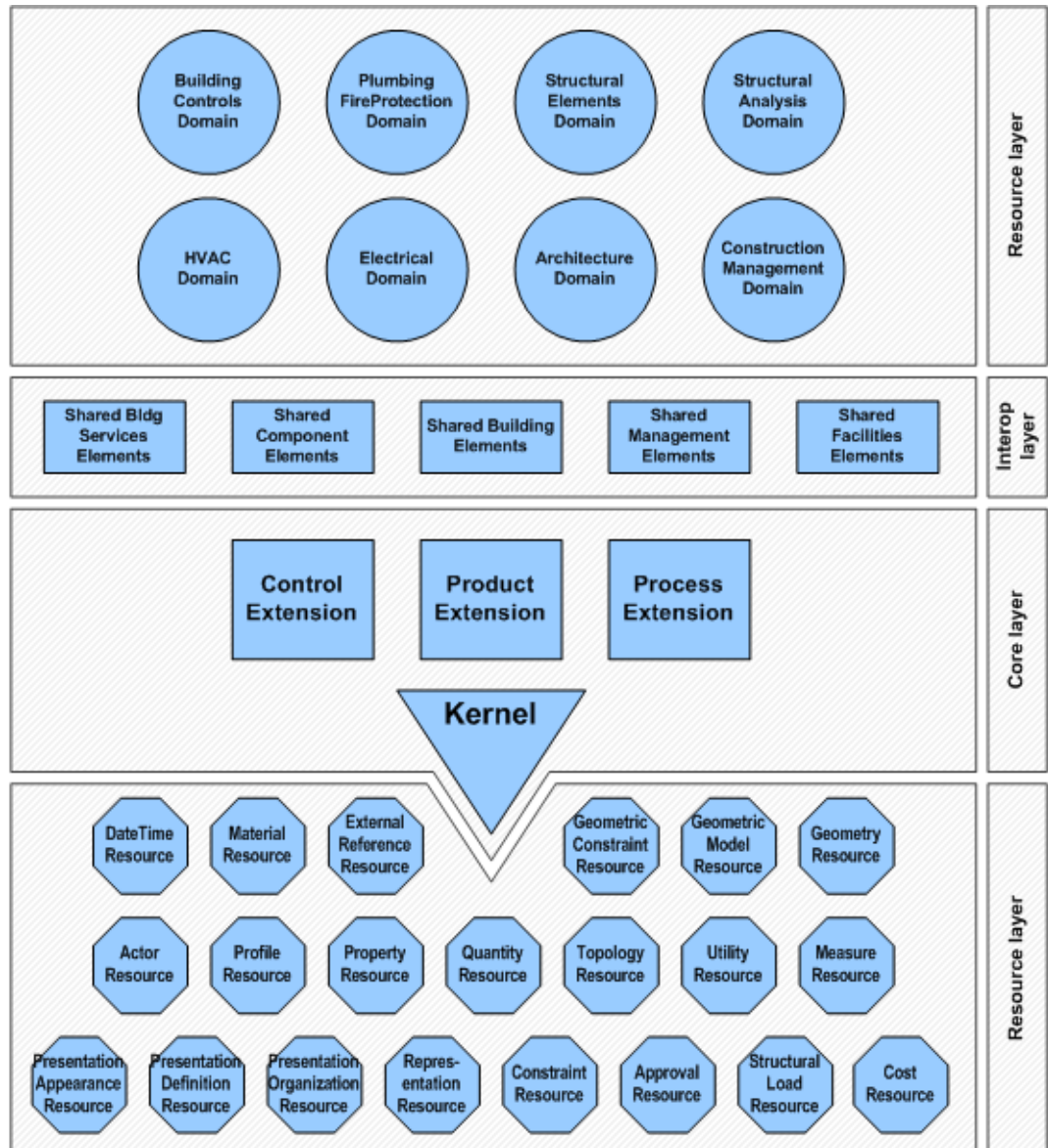


Figure 6: IFC conceptual layers [17]

There are three fundamental sub types of *IfcRoot*, namely, *IfcObjectDefinition*, *IfcPropertyDefinition* and *IfcRelationship*.

“*IfcObjectDefinition* is the generalization of any semantically treated thing or process, which can be either a type or an occurrence. Object definitions can be named by means of the inherited *Name* attribute, which should be a user recognizable label for the object occurrence.” [17], [18]

“*IfcPropertyDefinition* defines the generalization of all characteristics (i.e. grouping of individual properties), that may be assigned to objects. At present, sub types of *IfcPropertyDefinition* include property set occurrences, property set templates, and property templates.” [17], [18]

“*IfcRelationship* is the abstract generalization of all objectified relationships in IFC. Objectified relationships has the priority when it comes to handling relationships among objects. This allows to keep relationship specific properties directly at the relationship and opens the possibility to later handle relationship specific behavior. There are two different types of relationships, 1-to-1 relationships and 1-to-many relationships used within the subtypes of *IfcRelationship*.” [17], [18]

Figure 7 shows the IFC structure of how a wall is defined and the inheritance tree of the entity. Since the structure of IFC is hierarchical, each level of the inheritance tree introduces different attributes and relations to the *IfcWall* entity. *IfcElement* carries the relationship of the wall with other entities, such as opening elements.

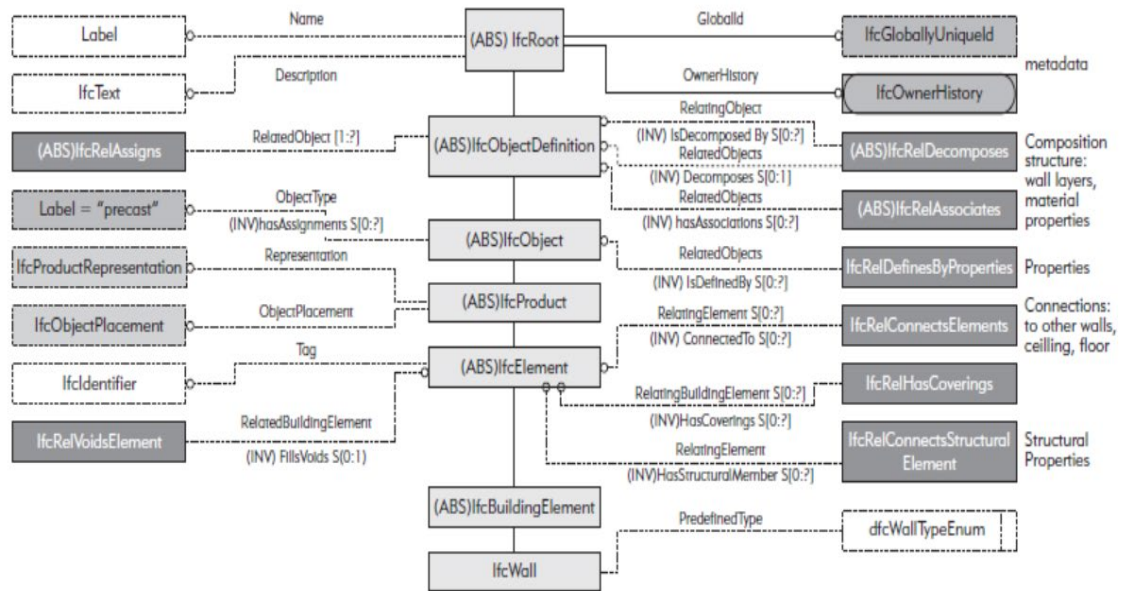


Figure 7: IFC Structure of a Wall [10]

IFC, developed to cater to the needs of all the stakeholders of the AEC industry, has multiple types of geometry, relationships, properties, etc and therefore has an abundance of data that can be trivial to different stakeholders. For example, each material layer of a wall is required by an energy simulation engineer but it is redundant for a structural engineer. This raises the need for the specific workflow of data exchange for each stakeholder. Hence, model view definitions are developed.

2.3.2 Model View Definition (MVD)

Model View Definitions are a subset of the overall IFC schema that is used to describe the exchange of data pertaining to a specific use. MVDs help the exchange to be effective by informing the end users about the type of data that should be received. MVDs are data-centric

rather than application-centric, i.e., MVDs do not depend on the software used by the stakeholders but rather depend on the need for data by the end user. [10]

There is already a database of MVDs available from buildingSMART. A few of the MVDs that are available are: Coordination View, Space Boundary Addon View, Structural Analysis View, Design Transfer View. This thesis uses IFC2x Edition 3 Technical Corrigendum 1 as the schema and Coordination View Version 2.0 along with Space Boundary Addon View as the MVD for the data exchange as shown in Figure 8.

```
ISO-10303-21;  
HEADER;FILE_DESCRIPTION(('ViewDefinition [CoordinationView_V2.0, SpaceBoundary2ndLevelAddOnView  
FILE_NAME('C:\\Users\\KHYATI\\Desktop\\Thesis\\Thesis.ifc', '2020-02-26T19:36:05', ('Architect'), (  
FILE_SCHEMA('IFC2X3'));  
ENDSEC;
```

Figure 8: Data Exchange Schema and MVD

Even though MVDs require certain data to be included in the exchange process, it does not guarantee the correctness and accuracy of the data. A data validation tool shall be used for this purpose.

2.3.3 IfcDoc

IfcDoc is a tool developed by buildingSMART for the validation of the correctness and accuracy (as explained in 3.3.2) of the data against an MVD [19]. The tool can be used to generate a custom MVD with custom property sets, quantity sets, etc and validate the data against it or a baseline MVD can be used to validate the data.

2.3.4 Interoperability Gap

EnergyPlus is currently not capable of directly importing any data exchange format (IFC or gbXML). This requires a third tool that has the ability to import a data exchange format and converts the file to an EnergyPlus compatible file format or a tool that uses EnergyPlus as an energy simulation engine.

A Geometry Simplification Tool (GST) which is now superseded by Space Boundary Tool (SBT-1), developed by Lawrence Berkley National Laboratory, can read the IFC file and convert it to IDF file. SBT-1 generates space boundaries and creates the geometry as a surface based on these space boundaries. It uses the IDF material library to assign materials to building elements. Figure 9 shows the mapping process of SBT-1.

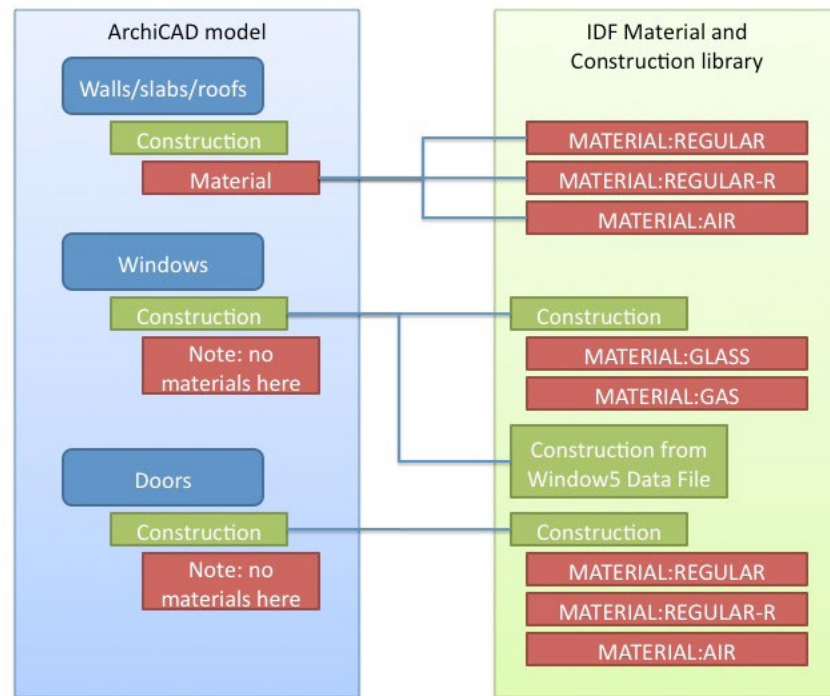


Figure 9: SBT-1 Mapping Process [20]

The limitations of using SBT-1 to are that to correctly map materials objects of an IDF file from the IFC file, the names have to be identical, and it only supports IFC2x3.

Various works of literature also mention errors encountered in the IDF file generated using SBT-1. Some of these errors are [21], [22]:

- Pitched and sloped roof geometry is not correctly converted
- Missing floors and ceiling
- Mispositioned windows

The latest version of Space Boundary Tool is currently not available for download and further development of the tool has been suspended. The unavailability of the tool requires another method for the conversion of IFC to IDF.

OpenStudio and Simergy are two software applications that run whole building energy simulation using EnergyPlus as the simulation engine. OpenStudio runs lighting analysis based on Radiance and a OpenStudio plug-in is used in SketchUp for geometry creation. These software have a graphical interface that is lacking in EnergyPlus. Both, OpenStudio and Simergy, can import IDF, IFC, and gbXML but when import data from these files, either the certain required data is missing in the exchange file format (IFC and gbXML) or the software is not able to read and map the data correctly.

There are already certain limitations with the import of IFC in OpenStudio. The import process of IFC to OpenStudio is not direct. It requires the IFC file to be uploaded to BIMServer which will then download an OpenStudio file (.osm). The limitations are [23]:

- BIMServer only supported IFC2x3 and is not able to read large IFC files
- Import of Sloped Wall and Roofs
- Import of Outside boundary conditions of ceilings, floors, doors, and windows.
- Import of Curtain Walls

Simergy has not stated any such limitations for the import. Even with the capabilities of importing other IFC data, there are still errors in the data that are imported in both OpenStudio and Simergy as stated in various works of literature. Some of these errors are:

- BIMServer is not able to read all the relevant data from IFC (OpenStudio)
- All of the geometry is not correctly generated (OpenStudio)
- Some walls are missing or the floors extend over the walls. Some of the windows are not sized correctly (OpenStudio) [24]
- Material data is not or incorrectly mapped and default materials are used instead (Simergy)

To overcome these interoperability gaps, i.e., loss of data, incorrect mapping of data, incorrect generation of geometry, etc, a research approach of extracting the data using python is undertaken that is explained in brief in the next section.

3. Research Approach

There are semi-automatic methodologies available to transfer data between BIM and BEM as discussed in the section 2.3.4. As previously explained, at times the data available within the interoperable exchange format is not graspable by the simulation engine; a methodology is developed for the smooth transfer of data to avoid the loss of data during the transfer process.

The research approach undertaken in this thesis involves using IFC as an interoperable data exchange file format. The IFC file is generated from ArchiCAD (BIM software) using various export options according to the input data requirements of EnergyPlus (BEM software). A direct translation of data between IFC and IDF (EnergyPlus input data file) is not available currently. Consequently, a python script is developed to read the available data in IFC and to write that data in the format and order that is decipherable by EnergyPlus to run the simulation successfully. Before the extraction is begun, the generated IFC file is viewed in IFC Java viewer to visualize and understand the hierarchy of the IFC entities. Once the data is extracted and the IDF is generated, the IDF is opened in OpenStudio for geometric visualization and verification of the building. Upon verification through OpenStudio, manual verification is done by comparing original data in ArchiCAD to the extracted data in IDF. The BIM model is enhanced with internal gains (space occupancy data) to improve the reliability of the energy simulation results. This data is also extracted using python. The simulation results of the energy model with and without internal gain are compared to understand the effect of additional data. Figure 10 shows the methodology that is developed and adopted in this thesis.

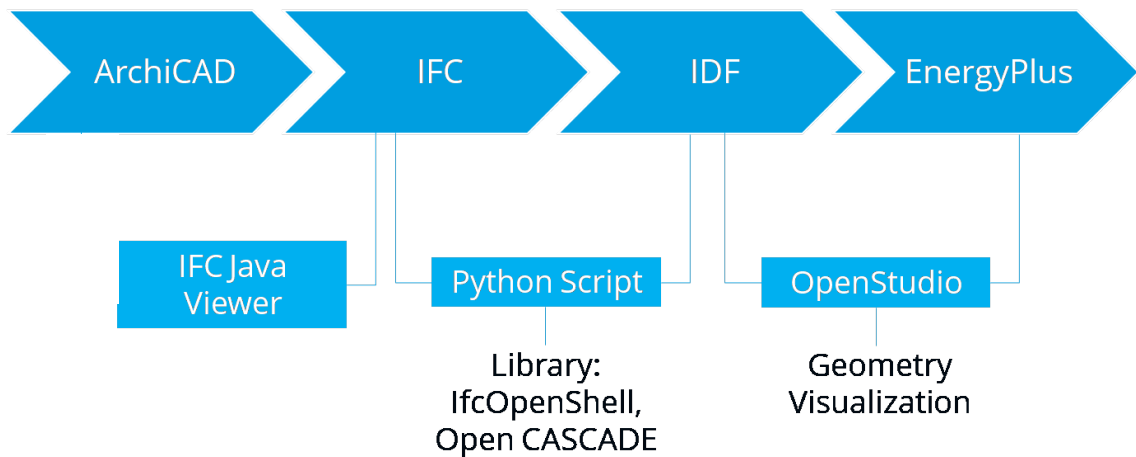


Figure 10: Methodology

Each step of the methodology is explained in detail in Chapter 4, Chapter 5 and Chapter 6. The data requirements of EnergyPlus and the modeling techniques used in ArchiCAD to obtain optimum and accurate data are further explained in the next section.

4. Data Requirements and Model Preparation

The architectural model and the energy model are different as explained in 3.2. The architectural model and the energy model are almost indistinguishable during the early design phase of the project. As the project progresses to further stages of the life-cycle, the architectural model and the energy model begin to differ. The building geometry (architectural data), available through the architectural model, is the basic input required by energy simulation engines. The differences occur due to the input data that is required by energy simulation software (for example, HVAC details).

The data required to perform energy simulation of a project can be roughly classified into the following five categories:

- Architectural Data: Building location and orientation, floor data (height, number of floors, etc), construction materials, and fenestration data
- Mechanical Data: HVAC zones, activities in zones, design flow rates, equipment detail, and control sequences
- Electrical Data: Electrical equipment (artificial lighting, kitchen equipment, etc)
- Internal Loads Data: Peak occupancy, peak lighting load, and peak equipment load
- Operations Data: Occupancy schedules, and lighting and equipment schedules

The data available for energy simulation determines the comprehensiveness of simulation results obtained and the accuracy of the simulated results. The more the availability of the data is, the more detailed the simulation results will be. For example, if only the architectural data is available, the simulation engine can only provide zone temperatures and weather/solar calculations.

The accuracy of the energy simulation also depends on the model prepared in the BIM application and the export of data to IFC. For example, the length of the walls interpreted by IFC will be incorrect if the walls are not connected correctly which will eventually result in an error in energy simulation as space/zone will not be fully enclosed.

4.1 EnergyPlus Data Requirement

The minimum input data required (class object) from an IFC file by EnergyPlus to successfully run a simulation are:

- Building
- Site:Location

This minimum input data will run weather/solar calculations. Building geometry is not required to run this type of simulation.

The input data (class object) that can be further obtained from an IFC file to run a more detailed simulation in EnergyPlus are:

- Material
- Construction
- Zone
- BuildingSurface
- FenestrationSurface

The data (parameters) further required by each of these class objects are mentioned in Appendix B.

The simulation settings (class object) need to be manually entered in EnergyPlus to successfully simulate as this data is not available in ArchiCAD for export to IFC. These class objects are:

- Simulation Control
- Timestep
- RunPeriod/SizingPeriod
- GlobalGeometryRules

The data (parameters) further required by each of these simulation class objects are mentioned in Appendix B.

4.2 ArchiCAD Model

The ArchiCAD BIM model is a three storey building with different types of spaces. The building consists of a laboratory, office spaces, etc. The building is located in Berlin, Germany. Figure 12 shows the floor plans and Figure 11 shows the 3D view of the modeled building.

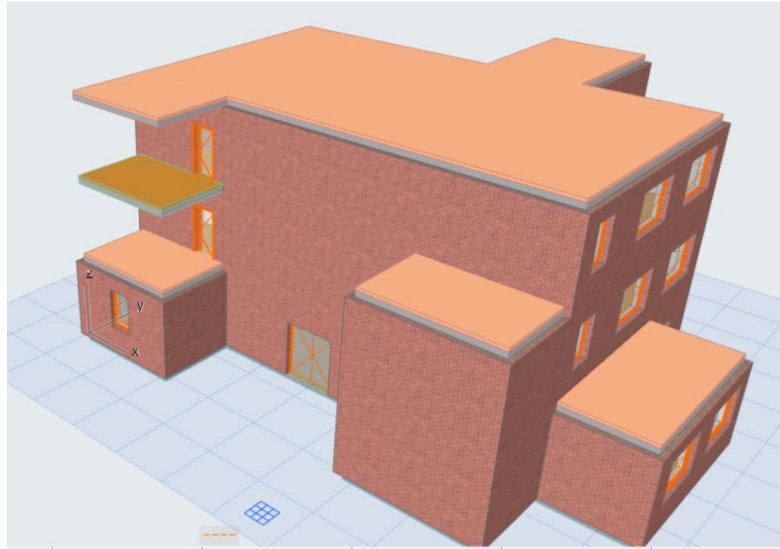


Figure 11: ArchiCAD BIM Model



Figure 12: Floor Plans

As explained previously, the modeling of the project also plays a part in the accuracy of the export of data from BIM to IFC; the following modeling options are used.

- The walls are modeled using the wall centerline as the reference line
- The direction of the reference line is important. The direction should be the same for all the walls in a particular elevation (north – right to left, south – left to right, east – bottom to top, and west – top to bottom)
- The reference line for slabs and roof is the bottom of the building element except for ground slab. The reference line for the ground slab is the top of the building element
- The slabs are modeled on the reference line of the walls and not till the outer face of the wall
- The slabs are modeled separately for each zone
- The zones are generated from the inner face of the wall and from the top of the slab till the bottom of the slab of the floor above to represent true volume as shown in Figure 13

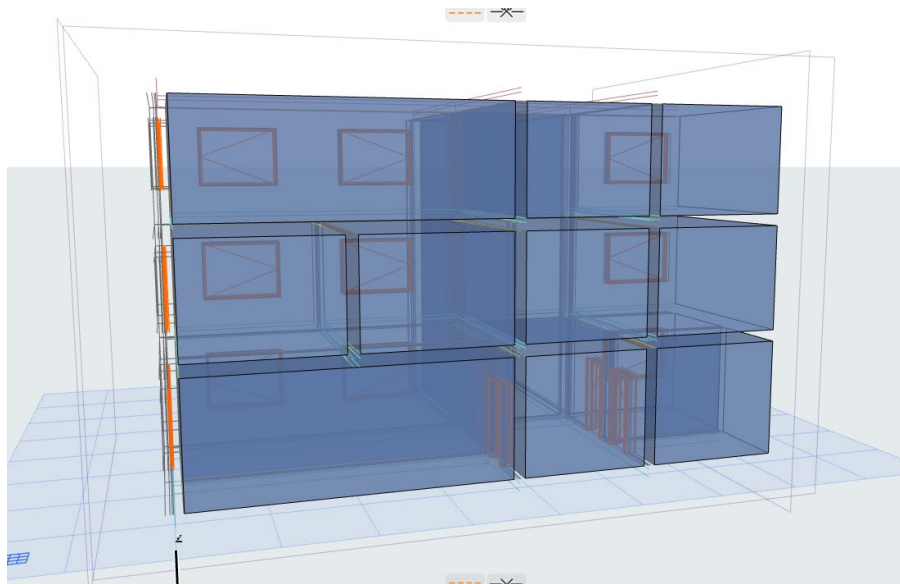


Figure 13: Zone Volume

- If the material has a different thickness in the construction layers of the building element, a material with a different name but the same properties is created
- The projecting slabs on the south elevation which act as shading surfaces are modeled as roof
- True North angle is set to 90° to match 0° north of EnergyPlus The 0° north of ArchiCAD (green arrow) is different than of EnergyPlus (red arrow) as shown in Figure 14.

EnergyPlus North



ArchiCAD North



Figure 14: North in ArchiCAD and EnergyPlus

Figure 15 shows IFC entities that are required for efficient energy simulation. Considering this information and the data requirements of EnergyPlus, IFC export settings are configured. The IFC export options used are mentioned in Appendix A.

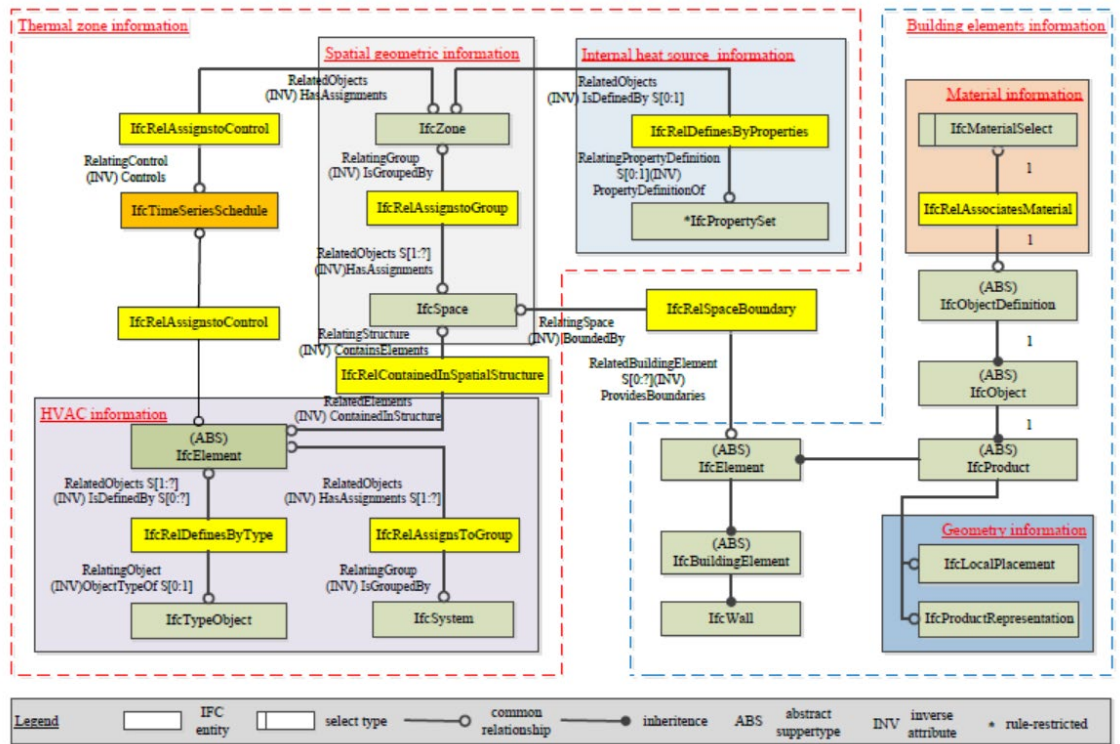


Figure 15: IFC information model [25]

Once the BIM model is accurately and correctly developed and the data is exported to IFC, the extraction of data in EnergyPlus format is carried out using python which is further explained in the next section.

5. Script Development and Validation

5.1 Data Mapping

The importing of BIM data into a BEM application requires that the data is correctly mapped between an IFC file that is generated from BIM software and BEM software. The terminology used to define building elements and other necessary data may be different in the source software (ArchiCAD), data exchange file (IFC) and the target software (EnergyPlus) but the data must be comprehended and assigned correctly. One of the gaps faced during interoperability is that there is a loss of data between different software because either the data is not readable or the data is mapped incorrectly. To overcome this gap, it is imperative to study how elements are defined in ArchiCAD and EnergyPlus. It is also important to study how relationships are formed between IFC entities and to corroborate that the required entities are exported from ArchiCAD.

5.1.1 IFC Entity Relationships

As explained earlier in 3.3.1, the IFC structure is hierarchical. It uses attributes and inverse attributes of the previous levels and certain attributes of its own to define an entity. The IFC schema divides all entities into rooted and non-rooted entities. Rooted entities derive from *IfcRoot* and each has a GUID, whereas non-rooted entities do not have a GUID. *IfcWall* entity does not have properties linked to it directly but rather has to be accessed using the attributes it inherits from the previous entities. Figure 16 shows how the *IfcWall* entity is linked using *IfcPropertySet* to obtain its properties (e.g., thermal properties). *IfcWall* uses *IsDefinedBy* inverse attribute which is inherited from the *IfcObject* entity to access its properties.

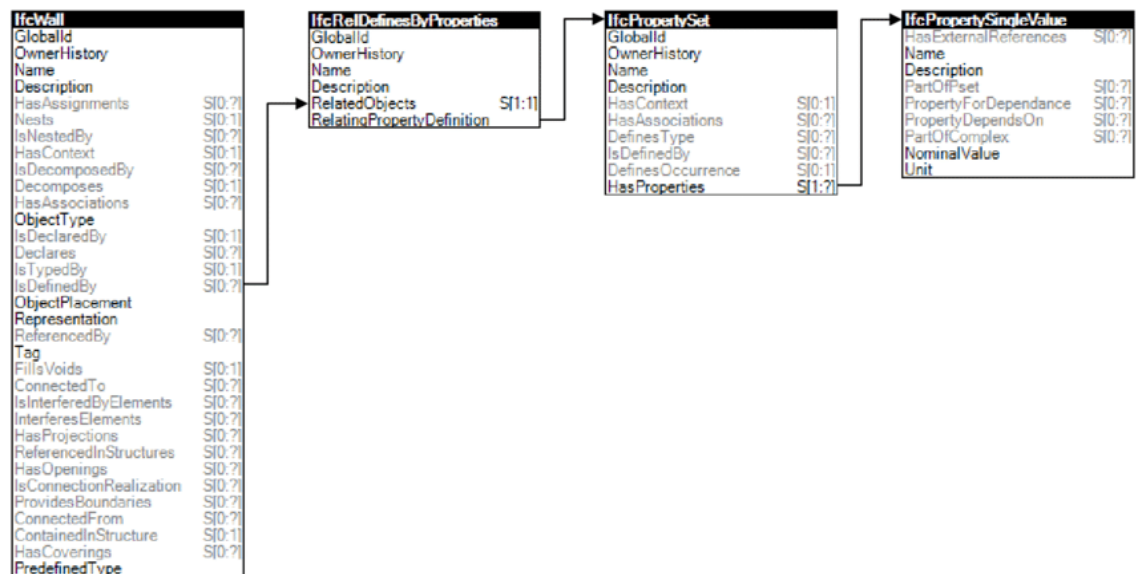


Figure 16: *IfcWall*

5.1.2 IFC Validation

Before the data is extracted using python for the generation of IDF, it is essential to check whether the required data is exported from ArchiCAD to IFC and if the IFC file generated is coherent to the IFC schema. The validation of the IFC file is done using the IFCDOC tool that is developed by buildingSMART. The IFC file is validated against the IFC2x3 schema and the Coordination View 2.0 MVD. Figure 17 shows that the file generated is correct and the entities are correctly mapped. There is no incorrect mapping of the data from ArchiCAD to IFC.

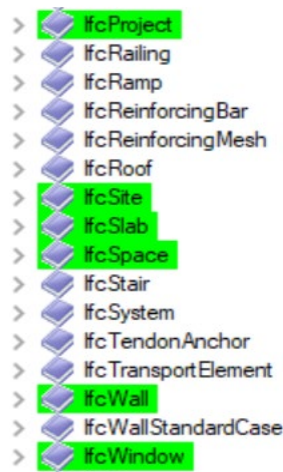


Figure 17: IFC Validation

5.1.3 Mapping IFC and IDF

As explained previously, the data must be correctly assigned in the simulation engine from the IFC file. To ensure this, IDF class objects and IFC entities are studied. There are various ways an element can be defined in IDF, e.g., a wall can be defined under BuildingSurface:Detailed IDF class object or under Wall:Detailed IDF class object. Table 2 below shows the IFC entities that correspond to the IDF class objects used in this thesis.

Table 2: IDF-IFC Mapping

IDF Class Object	IFC Entity	Comment
Building	- IfcProject	
Site:Location	- IfcSite	
Material	- IfcRelDefinesByProperties	<p>- Materials are obtained by iterating through <i>IfcBuildingElement</i> entity</p> <p>- Only <i>IfcWall</i> and <i>IfcSlab</i> entities are iterated</p>

Construction	- IfcRelAssociatesMaterial	<ul style="list-style-type: none"> - Construction layers are obtained by iterating through <i>IfcBuildingElement</i> entity - Only <i>IfcWall</i> and <i>IfcSlab</i> entities are iterated
Zone	- IfcSpace	
BuildingSurface:Detailed	- IfcWall, IfcSlab	<ul style="list-style-type: none"> - Geometry is obtained by iterating through <i>IfcBuildingElement</i> entity - Only <i>IfcWall</i> and <i>IfcSlab</i> entities are iterated
FenestrationSurface:Detailed	- IfcDoor, IfcWindow	<ul style="list-style-type: none"> - Geometry is obtained by iterating through <i>IfcBuildingElement</i> entity - Only <i>IfcDoor</i> and <i>IfcWindow</i> entities are iterated
Shading	- IfcSlab	<ul style="list-style-type: none"> - Geometry is obtained by iterating through <i>IfcBuildingElement</i> entity - Only <i>IfcSlab</i> entity is iterated

5.2 Data Extraction

As explained earlier that correct mapping of data is important for accurate simulations, this section explains how the data required by EnergyPlus is extracted from IFC using python and the generation of an IDF file with the correct mapped data.

The first IDF class object required for which the data can be extracted from an IFC file is 'Building'. Figure 18 shows the process flow of the developed python script to extract the data. The 'Building' class object requires varies parameters but only the 'Name' and 'True North' parameter can be extracted from IFC. The 'Name' parameter is obtained using the *Name* attribute of the *IfcProject* entity. *IfcProject* has an attribute named *RepresentationContext* which has a type, the *IfcGeometricRepresentationContext* entity. This entity has an attribute *TrueNorth*

which has accesses the *IfcDirection* entity. This entity has an attribute *DirectionRatios* which provides the rectangular coordinates of the direction of TrueNorth. EnergyPlus requires the input for ‘True North’ in degrees. These rectangular coordinates are converted to polar coordinates (degrees). Figure 19 shows the output after extraction.

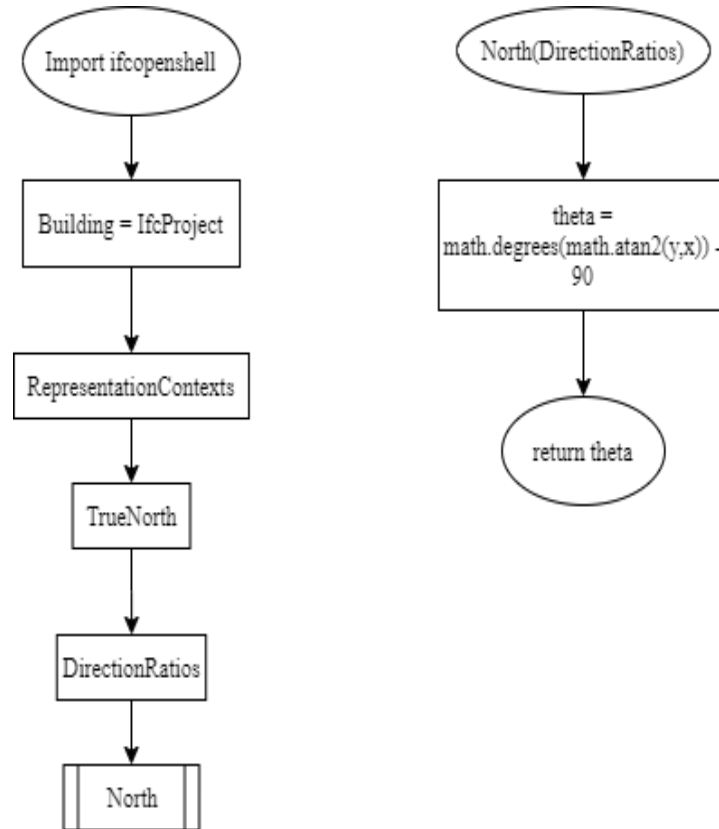


Figure 18: Process flow for 'Building' data extraction

<pre> Building, Thesis, 0.0, Suburbs, 0.04, 0.4, FullInteriorAndExterior, 25, 6; </pre>	<pre> !- Name !- North Axis {deg} !- Terrain !- Loads Convergence Tolerance Value !- Temperature Convergence Tolerance Value {deltaC} !- Solar Distribution !- Maximum Number of Warmup Days !- Minimum Number of Warmup Days </pre>	<div style="border: 1px solid blue; width: 30px; height: 15px; display: inline-block;"></div> manual/default inputs
---	--	---

Figure 19: 'Building' Output

The IDF class object ‘Site:Location’ is also required to run a simulation. This class object defines the geographical location of the project. The parameters required by ‘Site:Location’ are extracted from the *IfcSite* entity. This entity has *SiteAddress* as an attribute that has an assigned value of the *IfcPostalAddress* entity. This entity has an attribute *Town* which is mapped to the ‘Name’ input parameter in ‘Site:Location’ class object. Figure 20 shows the process flow. The input parameters ‘Latitude’, ‘Longitude’, and ‘Elevation’ in EnergyPlus are mapped to *RefLatitude*, *RefLongitude*, and *RefElevation* attributes of *IfcSite* entity respectively. The

latitude and longitude values in IFC are stored in degrees-minutes-seconds format whereas Energyplus requires it in degrees so *RefLatitude* and *RefLongitude* values are converted to the required format. Figure 21 shows the output after extraction.

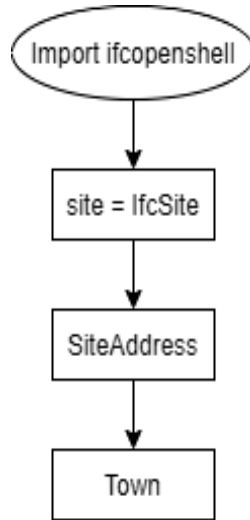


Figure 20: Process flow for 'Site:Location' data extraction

Site:Location,	
Berlin,	!- Name manual/default inputs
52.47,	!- Latitude {deg}
13.4,	!- Longitude {deg}
1,	!- Time Zone {hr}
49;	!- Elevation {m}

Figure 21: 'Site:Location' Output

The IDF class object 'Material' is required as the objects defined here will be used in the 'Construction' class object to define the construction layers of a building element. Figure 22 shows the process of extracting the required input parameters. The process shows that *IfcDoor* and *IfcWindow* entities are ignored. This is because ArchiCAD does not allow construction layers to be defined. Consequently, the material data for doors and windows are not available in IFC upon export. Also, the 'Roughness' parameter of the 'Material' class object is not available in IFC.

The materials' properties are extracted from the *IsDefinedBy* inverse attribute of each the *IfcBuildingElement* (abstract supertype of *IfcWall*, *IfcSlab*, *IfcDoor*, and *IfcWindow*). The inverse attribute is an objectified relationship that accesses the *IfcRelDefinesByProperties* and the *IfcRelDefinesByType* entities. *IfcRelDefinesByProperties* has a *RelatingPropertyDefinition* attribute with which has a type, the *IfcPropertySetDefinition* which is a supertype of the *IfcPropertySet* entity and the *IfcElementQuantity* entity. The 'thickness' parameter is extracted

using the *Quantities* attribute of the *IfcElementQuantity* entity. The rest of the parameters are extracted using *HasProperties* attribute of the *IfcPropertySet* entity. Figure 23 shows the output after extraction.

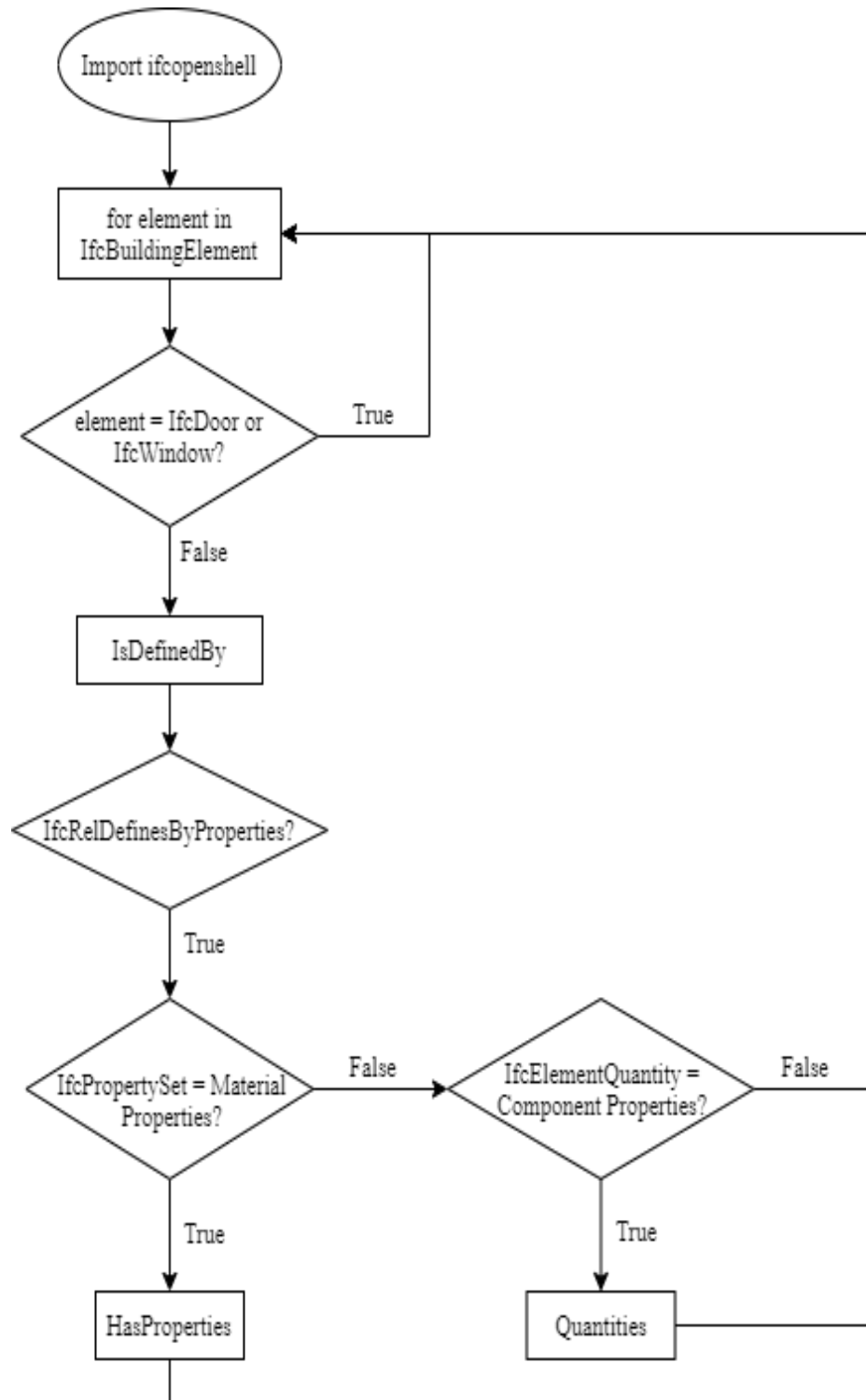


Figure 22: Process flow for 'Material' data extraction

Material,		
Plaster - Gypsum,	!- Name	 manual/default inputs
MediumSmooth,	!- Roughness	
0.01,	!- Thickness {m}	
0.57,	!- Conductivity {W/m-K}	
1300.0,	!- Density {kg/m3}	
1000.0;	!- Specific Heat {J/kg-K}	

Figure 23: 'Material' Output

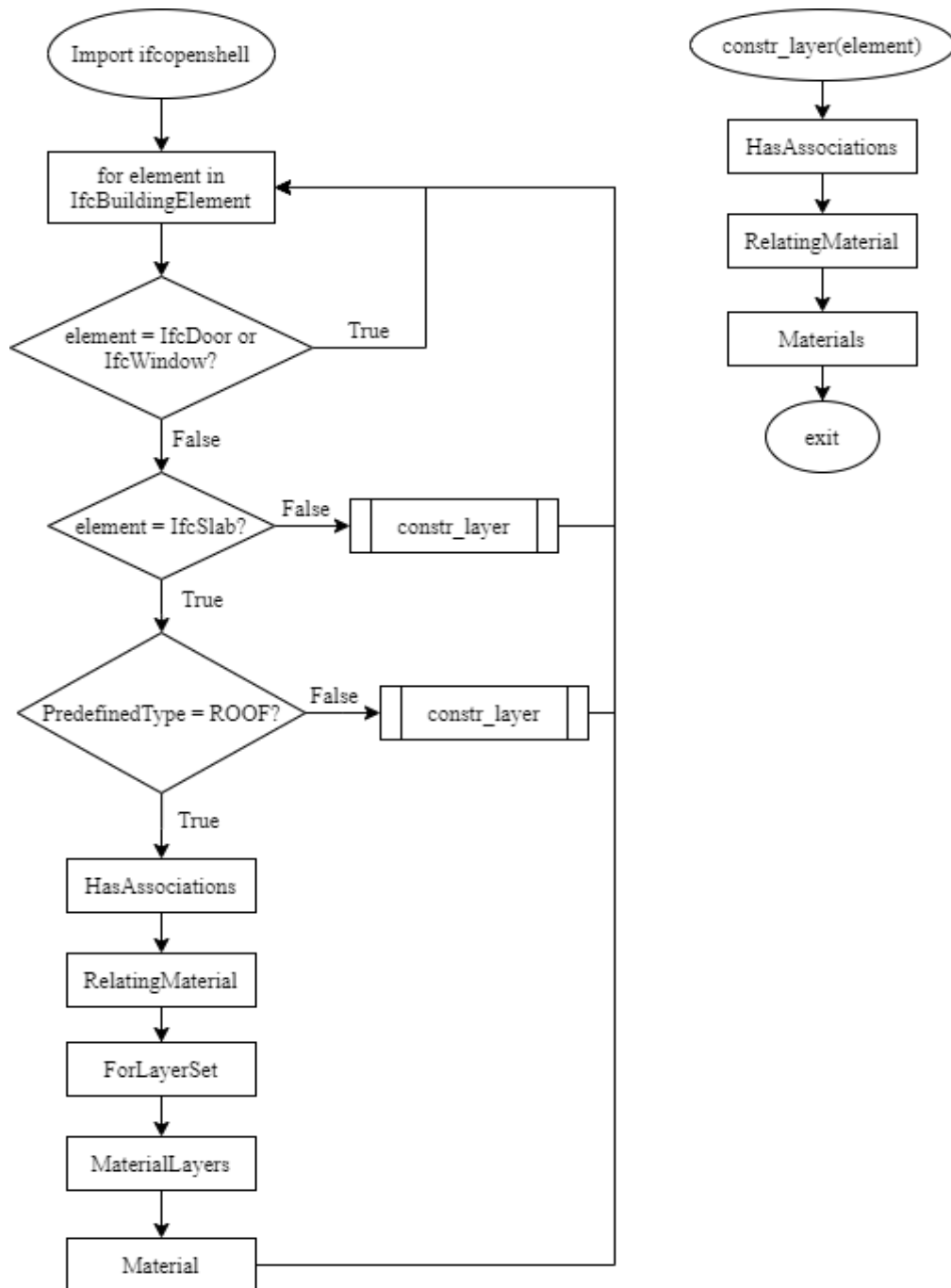


Figure 24: Process flow for 'Construction' data extraction

The next IDF class object that is essential is 'Construction'. As explained earlier for 'Material' class object, *IfcDoor* and *IfcWindow* entities are ignored as shown in Figure 24 due to the unavailability of the corresponding data.

The list of materials that construction is composed of is extracted through the *HasAssociations* inverse attribute. This attribute, an objectified relationship, accesses the *IfcRelAssociatesMaterial* entity. In the case of roof elements, the list of materials is extracted through the *RelatingMaterial* attribute of *IfcRelAssociatesMaterial* which accesses the *IfcMaterialLayerSet* entity with a *ForLayerSet* attribute which in turn accesses *IfcMaterialLayer* entity. In the case of other elements, the *RelatingMaterial* attribute accesses the *IfcMaterialList* entity. The layers are extracted through the *Materials* attribute of *IfcMaterialList*. Figure 25 shows the output after extraction.

```
Construction,
  215 Block Insulated Cavity Plastered 400,  !- Name
  Brick,                                     !- Outside Layer
  Air Space,                               !- Layer 2
  Insulation - Plastic Hard,               !- Layer 3
  Concrete Block - Structural (1),         !- Layer 4
  Plaster - Gypsum;                        !- Layer 5
```

Figure 25: 'Construction' Output

'Zone' is the next IDF class object that is necessary. Figure 26 shows the process flow of data extraction. 'Zone' corresponds to the *IfcSpace* entity of IFC. The extraction of data is done using the *IsDefinedBy* inverse attribute of each the *IfcSpace* entity. The inverse attribute, an objectified relationship, accesses the *IfcRelDefinesByProperties*. *IfcElementQuantity* is accessed using the *RelatingPropertyDefinition* attribute. The parameters, 'Ceiling Height', 'Volume', and 'Floor Area', are obtained using the *Quantities* attribute of the *IfcElementQuantity* entity.

EnergyPlus can auto calculate the parameters 'Volume', 'Ceiling Height', and 'Floor Area' but these calculated values will not be calculated correctly as the geometry of the building is created using the centerline of the walls and the height of the zone is modeled only till the bottom of the slab. The IFC file contains the correct values and these values are used as input. EnergyPlus will use the input values and not the calculated values for calculation. Figure 27 shows the output after extraction.

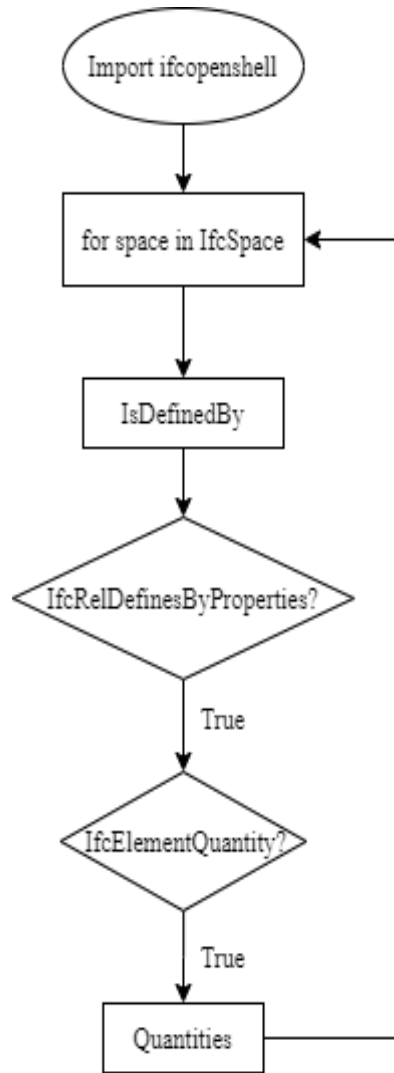


Figure 26: Process flow for 'Zone' data extraction

```

Zone,
  Laboratory 01,           !- Name
  ,                       !- Direction of Relative North {deg}
  ,                       !- X Origin {m}
  ,                       !- Y Origin {m}
  ,                       !- Z Origin {m}
  1,                      !- Type
  1,                      !- Multiplier
  ,                       !- Ceiling Height {m}
  44.058,                 !- Volume {m3}
  14.73999999464,         !- Floor Area {m2}
  ,                       !- Zone Inside Convection Algorithm
  ,                       !- Zone Outside Convection Algorithm
  Yes;                    !- Part of Total Floor Area
  
```

Figure 27: 'Zone' Output

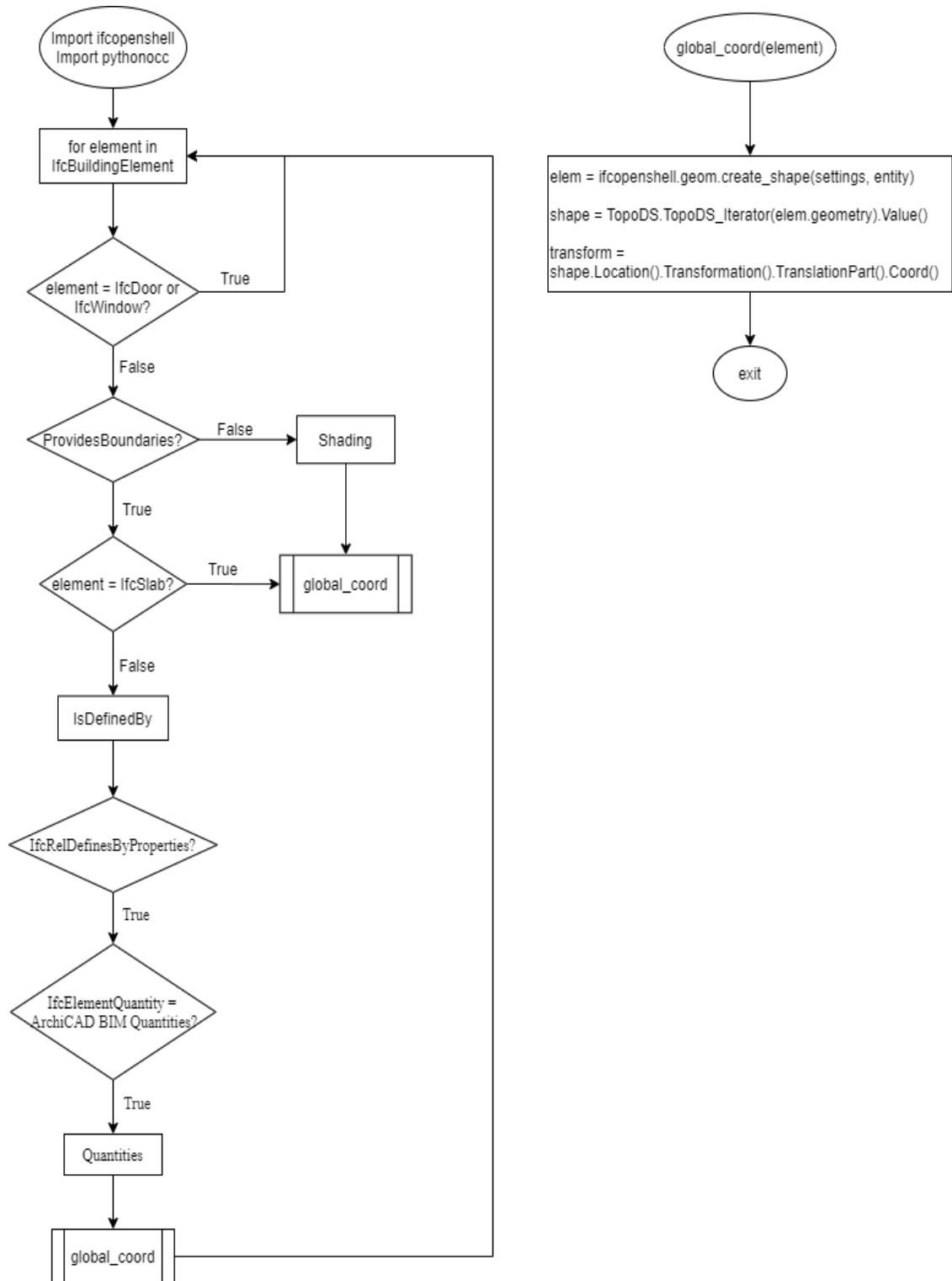


Figure 28: Process Flow for 'BuildingSurface' data extraction

'BuildingSurface:Detailed' class object is used to define the building geometry (walls, roofs, and floors). Figure 28 shows the data extraction (vertices) process flow. The shading elements are modeled as *IfcSlab* entities but these elements do not enclose any space/zone. These shading elements are extracted using the *ProvidesBoundaries* attribute of the *IfcBuildingElement*

(abstract) entity. The *ProvidesBoundaries* attribute accesses a list of *IfcRelSpaceBoundary* entities of the element which has an attribute of *InternalOrExternalBoundary* that is used to determine the ‘Outside Boundary Condition’, ‘Outside Boundary Condition Object’, ‘Sun Exposure’, and ‘Wind Exposure’ parameters of the IDF class object. IFC stores the starting coordinate (lower left vertex) of the *IfcBuildingElement* (abstract) entity in a local coordinate system. Python OPCASCADE library is used to obtain the global coordinates of the building element. The length and height of the building elements are used to determine the other three vertices. These properties are obtained by accessing the *Quantities* attribute of the *IfcElementQuantity* entity through the *IsDefinedBy* attribute of the *IfcBuildingElement* (abstract) entity. *IfcSlab* entities do not contain the length and width properties of the element. Consequently, the other three vertices of roofs and floors are required to be entered manually. Figure 29 shows the output after extraction.

```
BuildingSurface:Detailed,
    SW - 005,                !- Name
    Wall,                    !- Surface Type
    215 Block Insulated Cavity Plastered 400, !- Construction Name
    Corridor 01,              !- Zone Name
    Outdoors,                 !- Outside Boundary Condition
    ,                         !- Outside Boundary Condition Object
    SunExposed,               !- Sun Exposure
    WindExposed,              !- Wind Exposure
    0.5,                      !- View Factor to Ground
    ,                         !- Number of Vertices
    0.0,                      !- Vertex 1 X-coordinate {m}
    5.0,                      !- Vertex 1 Y-coordinate {m}
    0.0,                      !- Vertex 1 Z-coordinate {m}
    0.0,                      !- Vertex 2 X-coordinate {m}
    3.0,                      !- Vertex 2 Y-coordinate {m}
    0.0,                      !- Vertex 2 Z-coordinate {m}
    0.0,                      !- Vertex 3 X-coordinate {m}
    3.0,                      !- Vertex 3 Y-coordinate {m}
    3.0,                      !- Vertex 3 Z-coordinate {m}
    0.0,                      !- Vertex 4 X-coordinate {m}
    5.0,                      !- Vertex 4 Y-coordinate {m}
    3.0;                     !- Vertex 4 Z-coordinate {m}
```

Figure 29: 'Building:Surface' Output

‘FenestrationSurface:Detailed’ class object is used to define the building geometry (windows and doors). Figure 30 shows the data extraction (vertices) process flow. As previously explained, the starting coordinate of *IfcDoor* and *IfcWindow* entities are also stored in a local coordinate system and it is necessary to transform it into the global coordinate system. The width and height properties are extracted using the *OverallWidth* and *OverallHeight* attributes respectively of the *IfcDoor* and *IfcWindow* entities. These properties are then used to determine the other three vertices of the elements. Figure 31 shows the output after extraction.

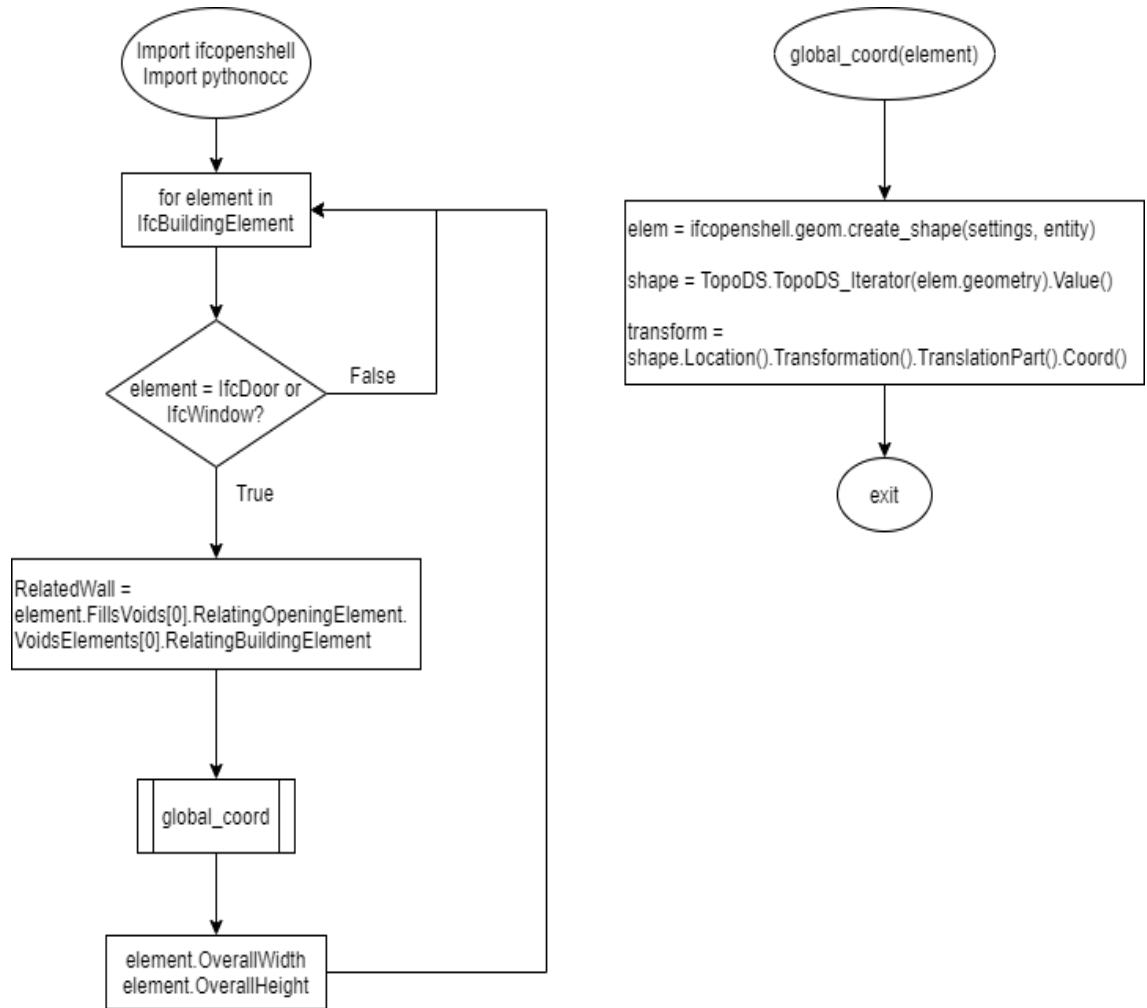


Figure 30: Process flow for 'FenestrationSurface' data extraction

```

FenestrationSurface:Detailed,
    WD - 006,                !- Name
    Window,                  !- Surface Type
    WinLib_2003,             !- Construction Name
    SW - 018,                !- Building Surface Name
    ,                        !- Outside Boundary Condition Object
    ,                        !- View Factor to Ground
    ,                        !- Frame and Divider Name
    1,                       !- Multiplier
    ,                        !- Number of Vertices
    16.0,                    !- Vertex 1 X-coordinate {m}
    10.05,                   !- Vertex 1 Y-coordinate {m}
    1.0,                     !- Vertex 1 Z-coordinate {m}
    16.0,                    !- Vertex 2 X-coordinate {m}
    10.950000000000001,      !- Vertex 2 Y-coordinate {m}
    1.0,                     !- Vertex 2 Z-coordinate {m}
    16.0,                    !- Vertex 3 X-coordinate {m}
    10.950000000000001,      !- Vertex 3 Y-coordinate {m}
    2.5,                     !- Vertex 3 Z-coordinate {m}
    16.0,                    !- Vertex 4 X-coordinate {m}
    10.05,                   !- Vertex 4 Y-coordinate {m}
    2.5;                     !- Vertex 4 Z-coordinate {m}
  
```

Figure 31: 'FenestrationSurface' Output

5.3 IDF Validation

Once the IDF file generated through python, it is important to check that the geometry created is accurate. Since EnergyPlus does not have a graphical interface, OpenStudio is used to visually check the accuracy of the geometry in IDF. The generated IDF is imported into OpenStudio. Before importing the IDF file, the coordinates for the floor, roof, and shading elements need to be entered manually.

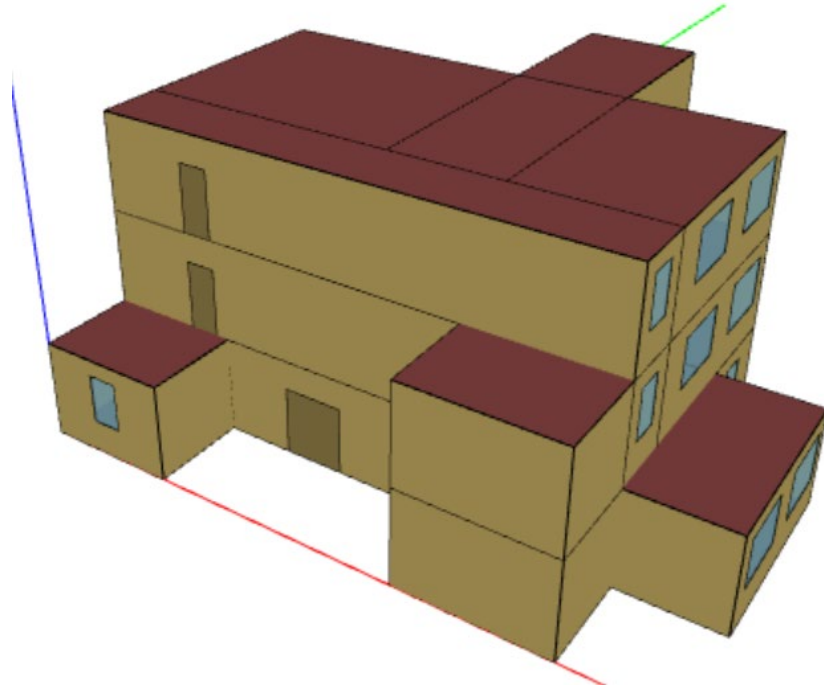


Figure 32: IDF Geometry in OpenStudio

Figure 32 shows the geometry of the model generated by OpenStudio by interpreting the IDF file. As seen in the figure, the shading elements are not imported but the rest of the geometry is imported and the correctly generated. OpenStudio can render the 3D view with respect to the outward normal of the building elements. The outward normal of a building element is important as it determines how the construction layers are layered in a building element. The outside face of a heat transfer surface will render white and the inside face will render red. Figure 33 shows that the geometry with respect to outward normal is also correct.

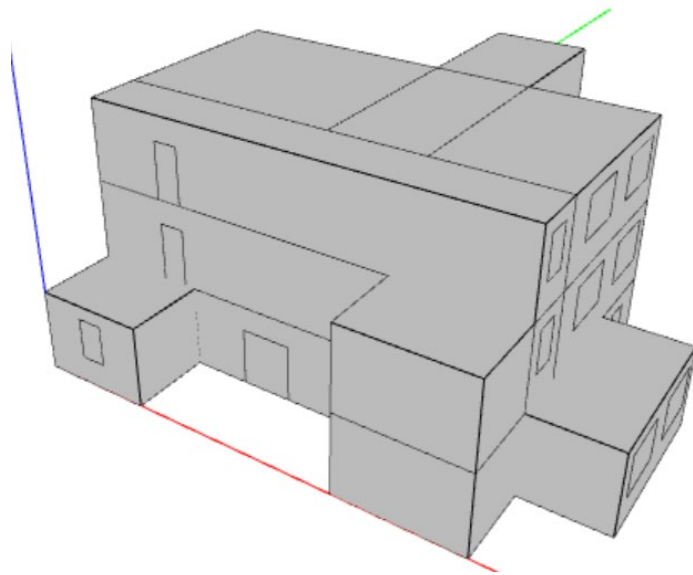


Figure 33: IDF Geometry in OpenStudio (Outward Normal Render)

The direction of the reference line in the ArchiCAD model plays an essential role in the outward normal direction as stated earlier. If the direction line is inverted than the stated direction, the geometry with respect to outward normal will be incorrectly generated. Figure 34 shows that if the direction of the reference line is not right (top), IDF creates the outward normal inside which implies that when construction is assigned to this wall, the outside layer mentioned in the ‘Construction’ IDF class object will be inside which is incorrect as it the outer layer of the wall. This option of rendering view by normal in OpenStudio works as a quick diagnostic tool before the simulation is run.

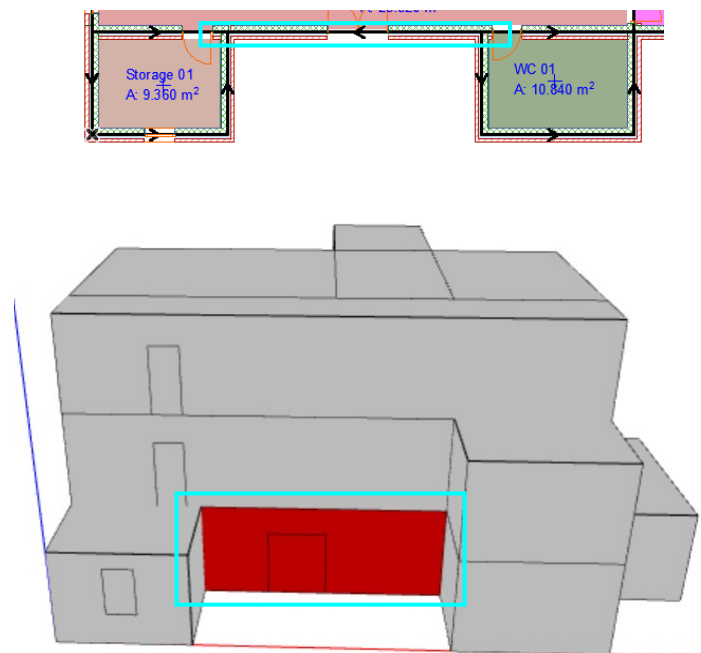


Figure 34: IDF Geometry in OpenStudio (Outward Normal Render) – Incorrect

The IDF class objects are also verified against schedules in ArchiCAD. The most important IDF class object to be checked is 'Materials' as it will determine the U-Values of the building elements. As mentioned previously, if a material has two different thicknesses, it needs to have different names. Therefore, these materials' thicknesses and the related 'Construction' class objects are verified. As observed in Figure 35 and Figure 29, the material and its corresponding thicknesses are accurately extracted (materials in ArchiCAD schedule and IDF class objects are highlighted in the same color). The other IDF class objects are also verified before the simulation run.

All Components Schedule		
ID	Name	Skin Thickness
EF-04	Brick	0.100
EF-06	Air Space	0.050
EM-00	GENERIC - EXTERNAL MEMBRANE	0.100
IC-00	GENERIC - INTERNAL CLADDING	0.100
IC-01	Plaster - Gypsum	0.010
IN-06	Insulation - Plastic Hard	0.025
IN-06	Insulation - Plastic Hard (1)	0.030
PR-00	GENERIC - PREFABRICATED	0.200
ST-00	GENERIC - STRUCTURAL	0.200
ST-05	Concrete Block - Structural	0.100
ST-05	Concrete Block - Structural (1)	0.215

Figure 35: Material Schedule (ArchiCAD)

Field	Units	Obj1	Obj2	Obj3	Obj4	Obj5	Obj6
Name		Plaster - Gypsum	Concrete Block - Structural (1)	Insulation - Plastic Hard	Brick	Concrete Block - Structural	GENERIC - INTERNAL CLADDING
Roughness		MediumSmooth	MediumRough	MediumRough	MediumRough	MediumRough	MediumRough
Thickness	m	0.01	0.215	0.025	0.1	0.1	0.1
Conductivity	W/m-K	0.57	0.6	0.032	0.58	0.6	0.25
Density	kg/m3	1300	1400	28	1500	1400	900
Specific Heat	J/kg-K	1000	880	1450	840	880	1000
Thermal Absorptance							
Solar Absorptance							
Visible Absorptance							

Figure 36: 'Materials' IDF Class Objects

Upon validation of the extracted data, the energy simulation of the building is performed. The results of the energy simulation are further explained and illustrated in the next section.

6. Building Energy Performance

Building performance simulation is a decisive tool to improve both the quality and performance of architectural design. It is necessary to achieve the holistic design and energy goals of the owner of the building. Building performance simulation is the use of physics-based software to calculate potential design impacts such as annual energy use and thermal and visual comfort. It includes building energy simulation or building energy modeling such as solar, shading, daylight, glare, thermal comfort, and natural ventilation.

Building energy performance is greatly influenced by the design strategies used. The energy performance of the building should be as efficient as possible. The major reason for the inefficiency of building energy performance is the extent of usage of fossil fuels. The reduction in fossil fuel use will reduce the CO₂ emissions which will consequently improve the building energy performance rating. Figure 37 shows that 50% of the energy generated by fossil fuels can be reduced by implementing different design strategies. Design strategies can include optimizing the orientation of the building using solar analysis, changing the building envelop, etc.

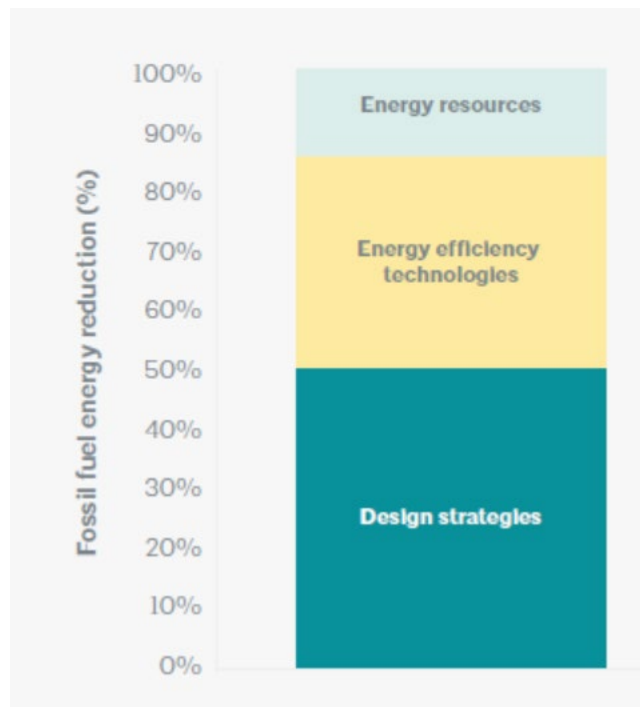


Figure 37: Fossil Fuel Reduction [26]

6.1 Simulation Results

The generated IDF file contains the required geometry along with the assigned thermal envelope. As previously explained, the construction of doors and windows cannot be extracted

from the IFC as it is not exported from ArchiCAD. Consequently, 'Construction' class object for doors and windows are manually added and later assigned to the 'construction name' parameter in the 'FenestrationSurface:Detailed' class object. The 'roughness' parameter in the 'Material' class object is also not available in IFC and is manually added.

The IDF class objects required for simulation are added to the generated IDF file. The entered parameters are mentioned in Appendix B. The default year for 'RunPeriod' available through the weather file is used. The year for 'RunPeriod' is 2017. When the simulation is run with the generated IDF file and the simulation parameters, only the weather data (site air temperature, etc) and zone air temperatures can be obtained. This type of simulation can be used to analyze the zone air temperatures and to manipulate it by changing the building envelope. This can help in the reduction of load on the HVAC systems. The simulation is run to understand the zone temperatures and to verify the inclusion of shading elements during the simulation. EnergyPlus can create a .dxf file of the geometry as an output if specified.

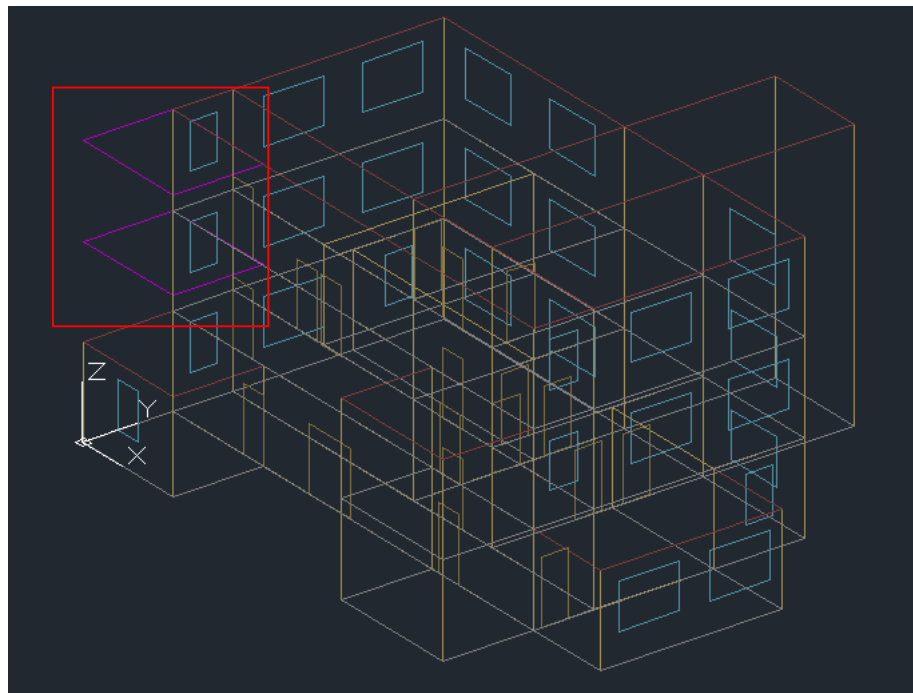


Figure 38: DXF Drawing including Shading Element

As seen from Figure 38, the two shading elements, represented in purple, are created in the IDF file through the extraction process using python even though it is not visible when the IDF file is imported in OpenStudio.

Once the geometry is verified using the output drawing (.dxf) from EnergyPlus, the HVAC system is manually entered into the existing generated IDF file. An HVAC template of the Fan Coil system is used in this thesis. The HVAC and related IDF class objects entered are

mentioned in Appendix B. The zones named Laboratory 01, Office 12, and Meeting 22 are conditioned by the HVAC system and the rest of the building is unconditioned.

The zone air temperature, heating energy, and cooling energy are requested as output results. An HTML file with a summary of all the data is also requested. The variables that can be requested as output depends on the IDF class objects used in the file.

The zone air temperature is shown in Figure 39 and Figure 40 is for Meeting 22. The output observed in the figure is at every timestep, i.e., at every 15 minutes as the timestep is set to 4. The air temperature of the zone in Figure 39 is observed when the simulation is run without HVAC IDF class objects or the HVAC system is off.



Figure 39: Air Temperature - Meeting 22 (HVAC off)

Figure 40 shows the zone air temperature when the HVAC system is on and infiltration is permitted. The heating setpoint is set to 18 °C and the cooling setpoint is set to 20 °C. The zone heat balance represents a well-stirred model for a zone, i.e., it takes into consideration the temperature change due to heating and cooling of the zone using the HVAC system when calculating the air temperature of the zones. The factors that affect the air temperature of the zone are the U-value of the building elements and also their thermal mass. During summer, the maximum air temperature observed when the HVAC system is off is 34.5 °C from Figure 39 but when the HVAC system is on, the maximum temperature observed is 30.5 °C from Figure 40. During winter, the minimum temperature observed is 1 °C when the HVAC system is off and 5.5 °C when the HVAC system is on.

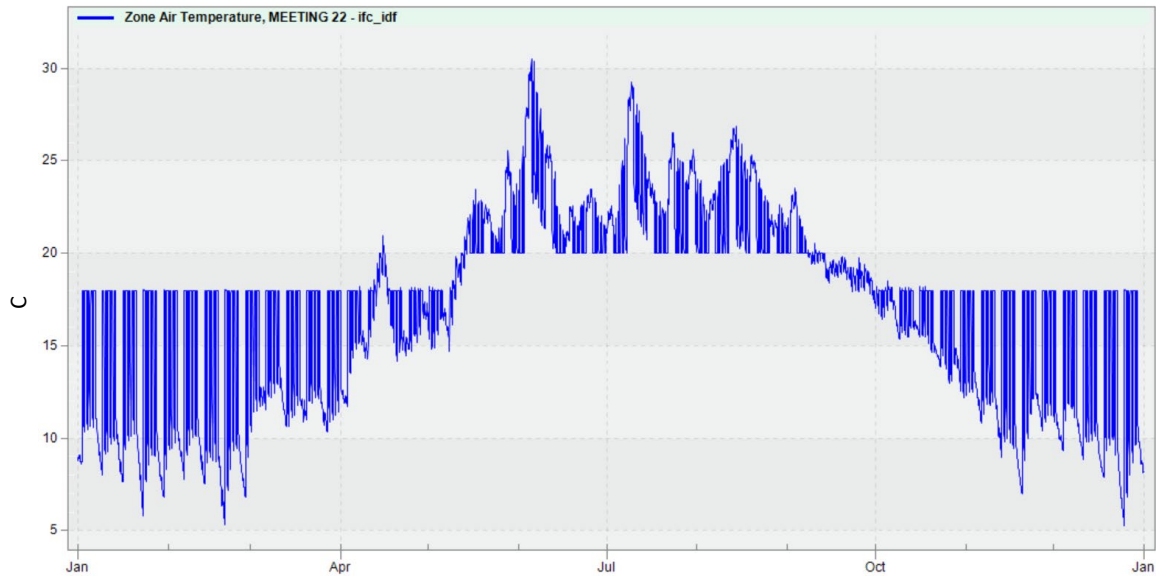


Figure 40: Zone Air Temperature - Meeting 22 (HVAC on)

The air temperature for zones Office 12 and Laboratory 01 is mentioned in Appendix C.

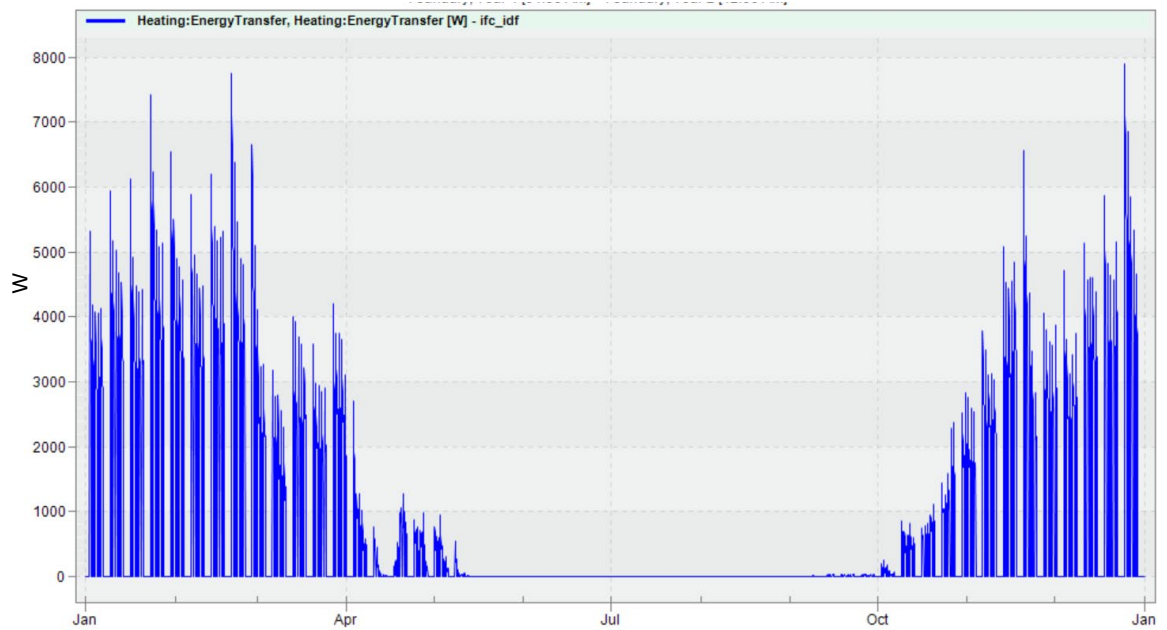


Figure 41: Heating Load

Figure 41 shows the heating energy required to maintain the set heating setpoint of 18 °C throughout the year and for the entire building. The heating energy is higher during December through February as expected. The heating energy during the summer months is zero as expected.

Figure 42 shows the cooling energy required to maintain the set cooling setpoint of 20 °C. The energy required is for the entire building. The cooling energy is zero as expected during the winter months.

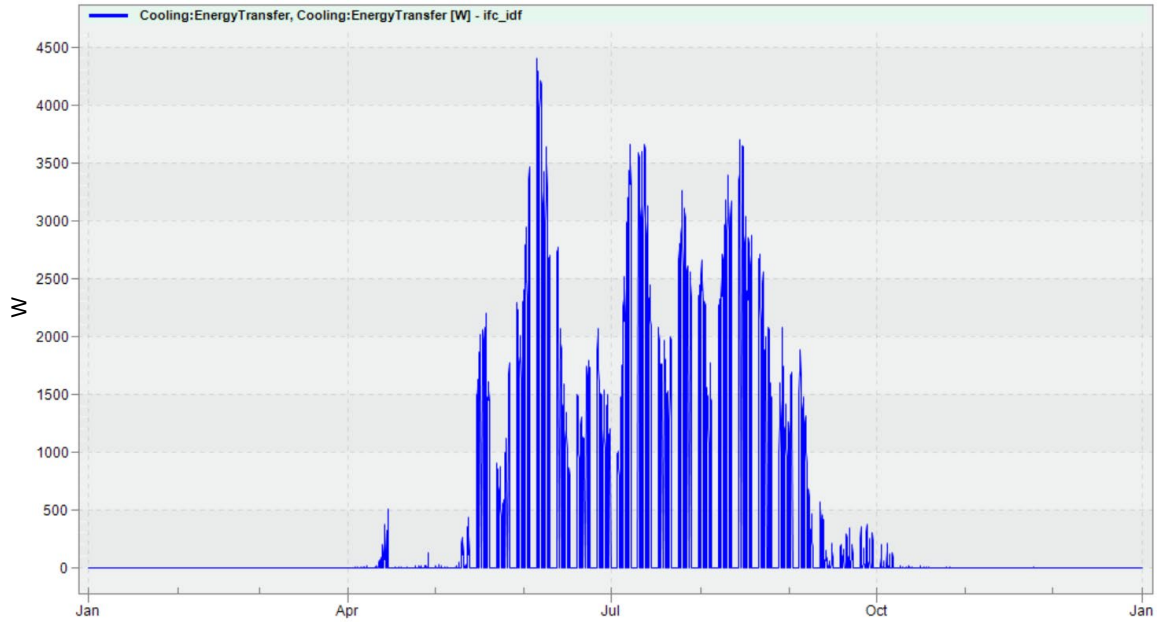


Figure 42: Cooling Load

Table 3 shows the total site and source energy used for the entire building.

Table 3: Site and Source Energy

	Total Energy [GJ]	Energy Per Total Building Area [MJ/m ²]	Energy Per Conditioned Building Area [MJ/m ²]
Total Site Energy	30.25	70.5	454.27
Net Site Energy	30.25	70.5	454.27
Total Source Energy	44.33	103.3	665.68
Net Source Energy	43.33	103.3	665.68

The simulation run does not include any internal gains. Consequently, the results of the simulations are not accurate enough. The internal gains like people, equipment, and artificial lighting also have a major impact on the heating and cooling loads and the energy performance of the building. The BIM model can be enriched to include data related to internal gains for the simulation engine to read.

6.2 Enriched Model Simulation

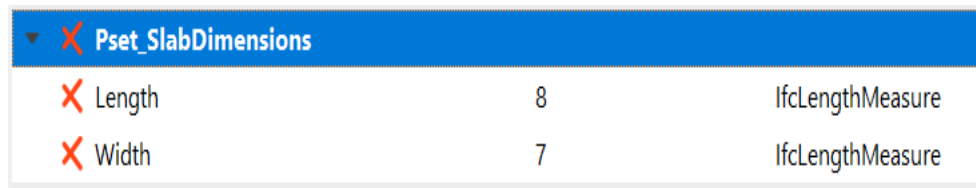
6.2.1 BIM Enrichment

Semantic enrichment of building models refers to the automatic or semiautomatic addition of meaningful information to a digital model of a building or other structure by software that can deduce new information by processing rules [27]. The enrichment of building models is especially important for as-is or as-built BIM models. The enrichment process can help in

bridging the interoperability gap between software. The enrichment process is generally applied to add the missing information in the BIM model but it can also be used to extend the schema of the building information models [27]. It has been previously used to identify the missing concepts in the IFC schema related to the building codes. The enrichment of a BIM model can also improve the communication between the stakeholders.

The BIM model in this thesis is also enriched with information that is required by an energy simulation software to perform a more accurate analysis. The information added to the existing BIM model is related to the internal gains of a space. Internal gains (occupancy, artificial lighting, and other types of equipment) along with their operational schedules are required to assess the building energy performance and to obtain the heating and cooling demand of the building as close to reality as possible.

As mentioned previously, the coordinates of the *IfcSlab* entities are entered manually in the IDF file. The BIM model is enriched with a custom property set named ‘Pset_SlabDimensions’ with two properties, namely, the length and width of the slab. The properties are of the type *IfcLengthMeasure* as shown in Figure 43.



Pset_SlabDimensions		
Length	8	IfcLengthMeasure
Width	7	IfcLengthMeasure

Figure 43: Slab Dimension Property Set

The enriched BIM model is then exported to IFC and the extraction of the coordinates of the *IfcSlab* entities is also carried out using python. The python code for this extraction is incorporated in the existing script. The same extraction process is used as the other *BuildingSurface* elements. The restructured extraction process is highlighted in red in Figure 44.

The same process flow is followed to extract the global coordinate of the *IfcSlab* entity. The custom property set added to the *IfcSlab* entity is accessed through the *HasProperties* attribute of the *IfcPropertySet* entity which is accessed through the *IsDefinedBy* attribute of the *IfcBuildingElement* (abstract) entity. These properties (dimensions) are then used to determine the other three vertices of the floor, roof, and shading elements. Figure 45 shows the output after extraction.

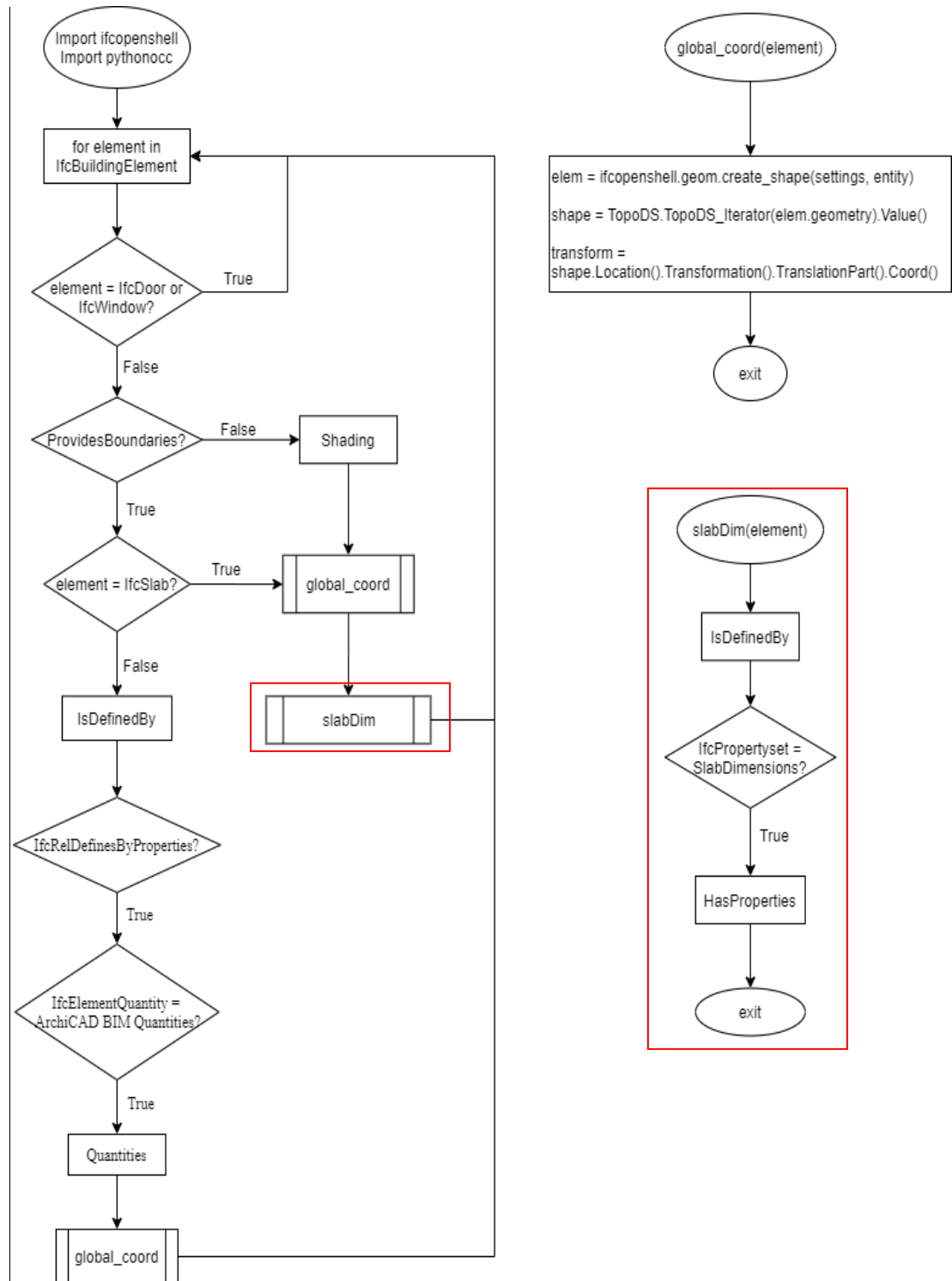


Figure 44: Process Flow for Slab Dimensions Extraction

```

BuildingSurface:Detailed,
  SLA - 016,                !- Name
  Floor,                    !- Surface Type
  Generic Slab/Roof 300,    !- Construction Name
  Office 11,                !- Zone Name
  Zone,                     !- Outside Boundary Condition
  Office 04,                !- Outside Boundary Condition Object
  NoSun,                    !- Sun Exposure
  NoWind,                   !- Wind Exposure
  1,                        !- View Factor to Ground
  ,                          !- Number of Vertices
  11.5,                     !- Vertex 1 X-coordinate {m}
  5.0,                      !- Vertex 1 Y-coordinate {m}
  3.0,                      !- Vertex 1 Z-coordinate {m}
  11.5,                     !- Vertex 2 X-coordinate {m}
  12.0,                     !- Vertex 2 Y-coordinate {m}
  3.0,                      !- Vertex 2 Z-coordinate {m}
  16.0,                     !- Vertex 3 X-coordinate {m}
  12.0,                     !- Vertex 3 Y-coordinate {m}
  3.0,                      !- Vertex 3 Z-coordinate {m}
  16.0,                     !- Vertex 4 X-coordinate {m}
  5.0,                      !- Vertex 4 Y-coordinate {m}
  3.0;                      !- Vertex 4 Z-coordinate {m}

```

Figure 45: 'BuildingSurface' Output (Floor Slab)

The BIM model is also enriched with occupancy details and its operational schedule. The enrichment is done by using the existing ArchiCAD custom property set options. ArchiCAD has the capabilities of exporting its property sets as well as any custom created property sets to IFC. ArchiCAD already has a custom property set named 'Pset_SpaceOccupancyRequirements' wherein the occupancy information can be added. The property set does not have a property for schedule. Therefore, a custom property named 'Schedule' is added to the property set of space occupancy requirements. The property is of the type *IfcPropertyTableValue* as shown in Figure 46.

▼ Pset_SpaceOccupancyRequirements		
<input type="checkbox"/> AreaPerOccupant		IfcAreaMeasure
<input type="checkbox"/> IsOutlookDesirable		IfcBoolean
<input type="checkbox"/> MinimumHeadroom		IfcLengthMeasure
<input checked="" type="checkbox"/> OccupancyNumber	15	IfcCountMeasure
<input type="checkbox"/> OccupancyNumberPeak		IfcCountMeasure
<input type="checkbox"/> OccupancyTimePerDay		IfcTimeMeasure
<input type="checkbox"/> OccupancyType		IfcLabel
<input checked="" type="checkbox"/> Schedule	Through: 12/31:; F...	IfcPropertyTableValue

Figure 46: Space Occupancy Property Set

The property 'OccupancyNumber' and 'Schedule' is used to define the 'People' and 'Schedule:Compact' class objects in IDF respectively.

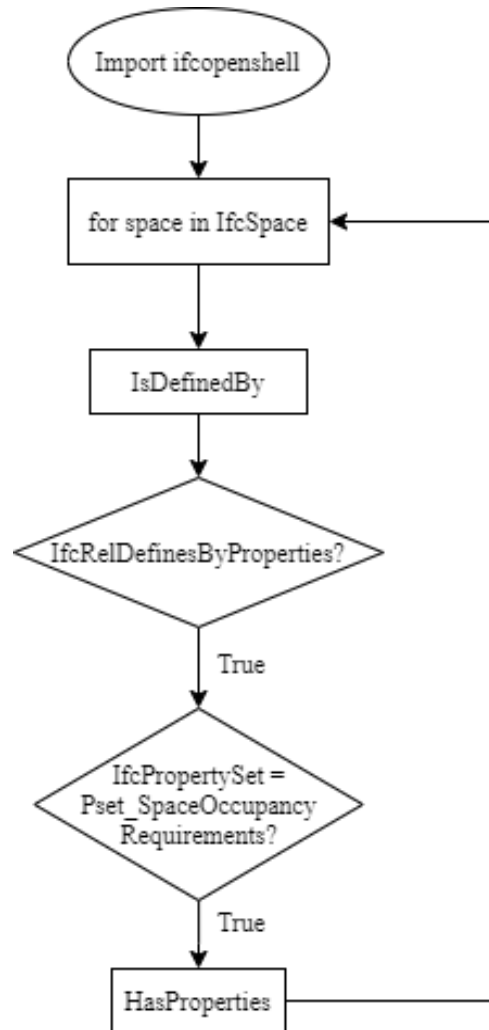


Figure 47: Process Flow for 'People' and 'Schedule:Compact' data extraction

'People' and 'Schedule:Compact' class objects are extracted from the enriched IFC file. Figure 47 shows the process flow of data extraction. The data is added to the zone elements in ArchiCAD which correspond to the *IfcSpace* entity of IFC. Therefore, *IfcSpace* is iterated during the extraction process. The extraction of data is done using the *IsDefinedBy* inverse attribute of each the *IfcSpace* entity. The inverse attribute, an objectified relationship, accesses the *IfcRelDefinesByProperties*. *IfcPropertySet* is accessed using the *RelatingPropertyDefinition* attribute. If the *Name* attribute of *IfcPropertySet* is *Pset_SpaceOccupancyRequirements*, *IfcPropertySingleValue* entity is accessed using the *HasProperties* attribute to extract the occupancy number and *IfcPropertyTableValue* entity is accessed to extract the operational schedule of the occupancy. Figure 48 and Figure 49 shows the output after extraction for 'People' and 'Schedule' class objects respectively.

```

People,
  Laboratory 01People,      !- Name
  Laboratory 01,           !- Zone or ZoneList Name
  Laboratory 01Schedule,   !- Number of People Schedule Name
  People,                  !- Number of People Calculation Method
  10.0,                    !- Number of People
  ,                         !- People per Zone Floor Area {person/m2}
  ,                         !- Zone Floor Area per Person {m2/person}
  ,                         !- Fraction Radiant
  ,                         !- Sensible Heat Fraction
  ActSchd;                 !- Activity Level Schedule Name

```

Figure 48: 'People' Output

```

Schedule:Compact,
  Laboratory 01Schedule,   !- Name
  Fraction,                !- Schedule Type Limits Name
  Through: 12/31,          !- Field 1
  For: WeekDays CustomDay1 CustomDay2 , !- Field 2
  Until: 08:00,            !- Field 3
  0,                       !- Field 4
  Until: 11:00,            !- Field 5
  1,                       !- Field 6
  Until: 14:00,            !- Field 7
  0.5,                     !- Field 8
  Until: 17:00,            !- Field 9
  0.8,                     !- Field 10
  Until: 24:00,            !- Field 11
  0,                       !- Field 12
  For: Weekends Holiday,   !- Field 13
  Until: 24:00,            !- Field 14
  0,                       !- Field 15
  For: SummerDesignDay,    !- Field 16
  Until: 24:00,            !- Field 17
  1,                       !- Field 18
  For: WinterDesignDay,    !- Field 19
  Until: 24:00,            !- Field 20
  1;                       !- Field 21

```

Figure 49: 'Schedule' Output

6.2.2 Enriched Model Simulation Results

The internal gains and their respective schedules are modeled for the three zones that are discussed in this thesis, i.e., Office 12, Laboratory 01, and Meeting 22.

Occupancy of the zone is one of the major sources of sensible and latent heat gains that affect the heating and cooling loads of the building which in turn influence the building energy performance. To better understand the effect of occupancy and its schedule (occupancy and activity schedule), the simulation is run with the HVAC system off. Figure 50 shows the comparison of zone air temperature with (red graph) and without (blue) internal gains (people) for Meeting 22 zone. As observed in the figure, the occupancy of the zone has a significant

influence on the air temperature. Due to this increase in air temperature, the heating load for this zone decreases and the cooling load increases. The zone air temperature for Laboratory 01 and Office 12 zones is mentioned in Appendix C.

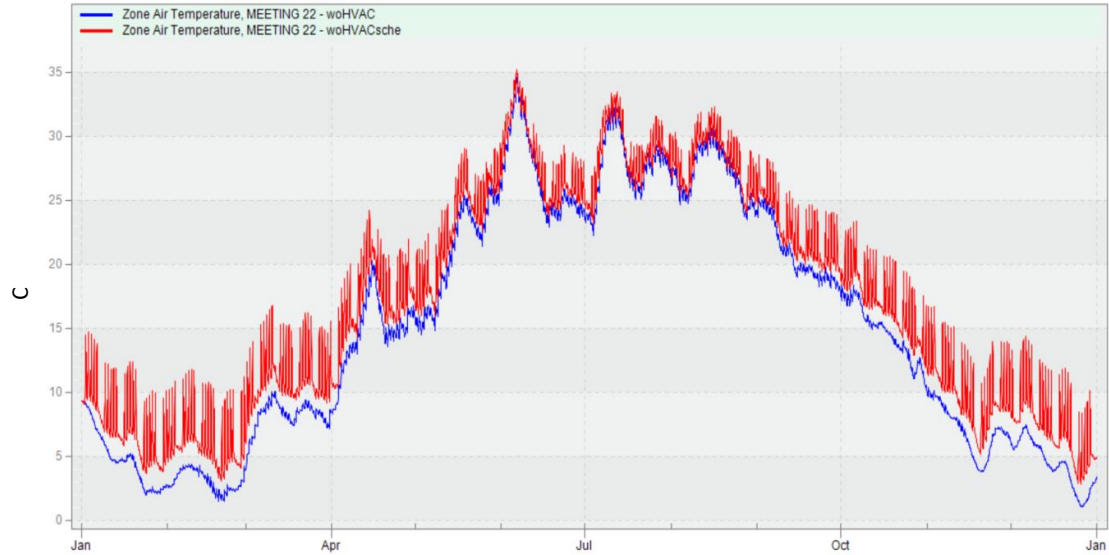


Figure 50: Zone Air Temperature - Meeting 22 (with and without 'People')

Figure 51 shows the heating load and heat gain due to the ‘People’ class object for two days (February 20 and February 21). The yellow graph represents the heating load without internal gains (people and associated occupancy schedule) and the green graph represents the heating load with internal gains. The purple graph, orange graph, and light orange graph represent heat gain due to people in zone Office 12, Laboratory 01, and Meeting 22 respectively. As observed, the heating load decreases (green graph) to maintain the heating setpoint when the internal gains are taken into consideration when compared to the heating load (yellow graph) in absence of internal gains. This is due to the increase in air temperature of the zone as a result of the increase in heat gain due to the occupancy (purple graph, orange graph, and light orange graph).

Figure 52 shows the cooling load and heat gain due to the ‘People’ class object for two days (June 5 and June 6). The blue graph represents the cooling load without internal gains (people and associated occupancy schedule) and the red graph represents the cooling load with internal gains. The purple graph, orange graph, and light orange graph represent heat gain due to people in zone Office 12, Laboratory 01, and Meeting 22 respectively. As observed, the cooling load increase (blue graph) to maintain the cooling setpoint when the internal gains are taken into consideration when compared to the cooling load (red graph) in absence of internal gains. This is due to the increase in air temperature of the zone as a result of the increase in heat gain due to the occupancy (purple graph, orange graph, and light orange graph).

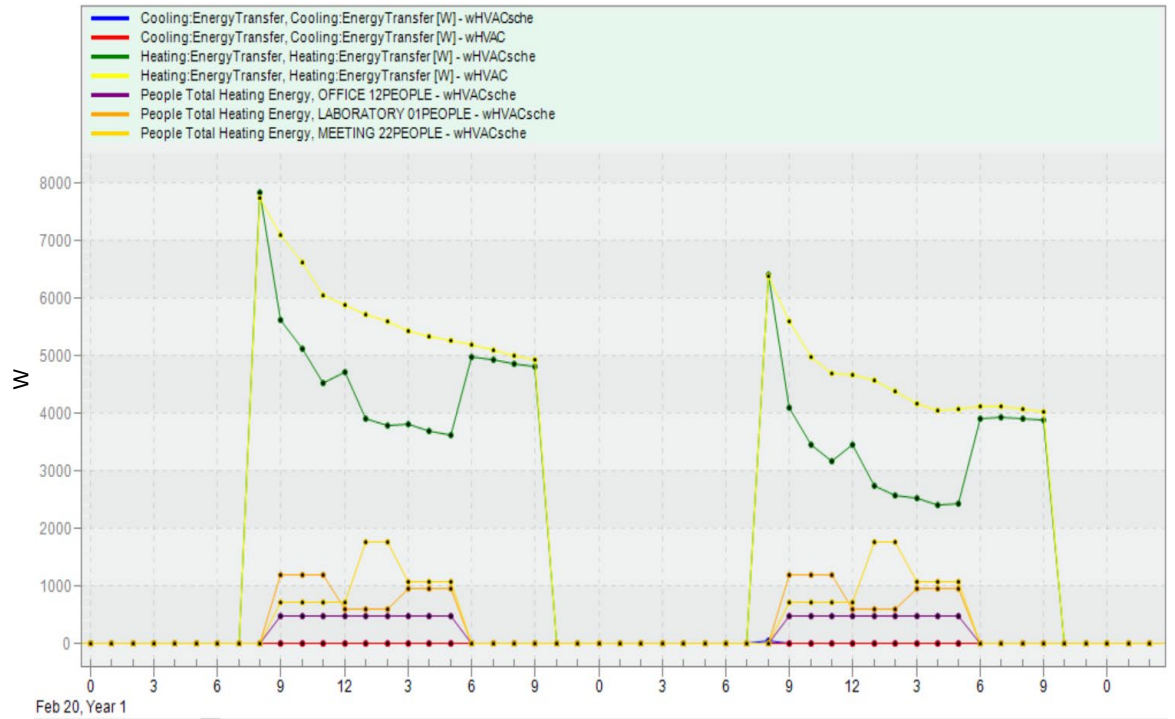


Figure 51: People Heating Energy and Heating Load

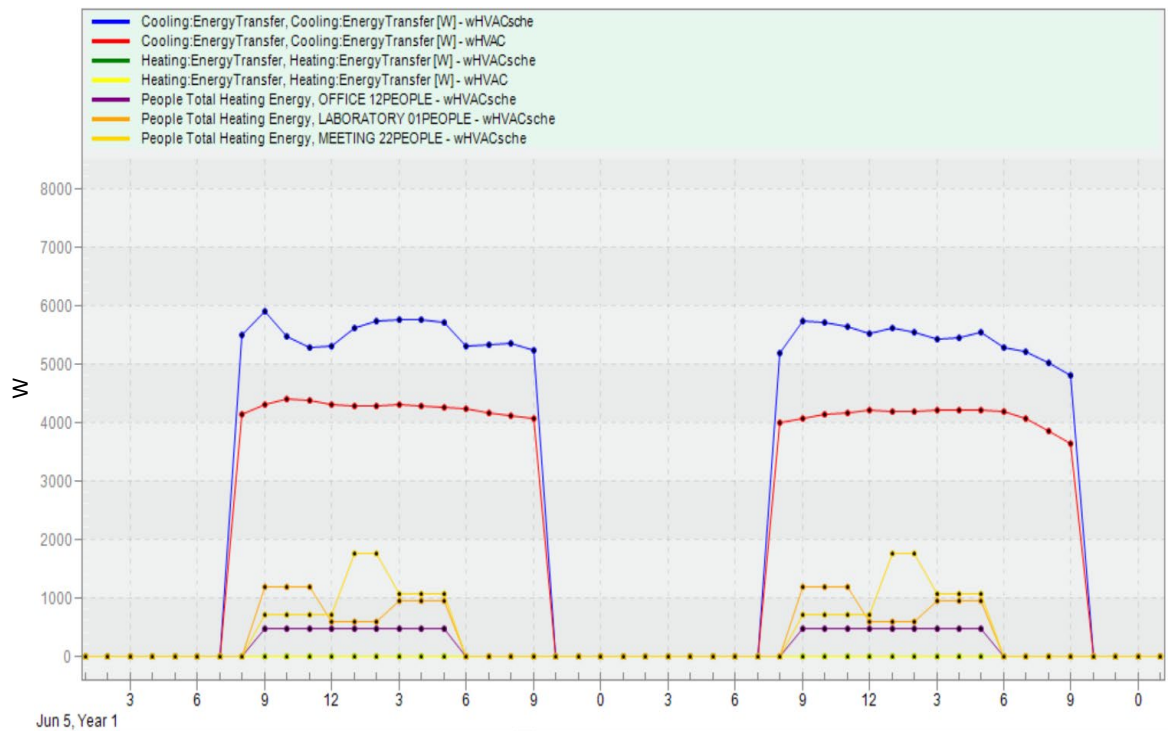


Figure 52: People Heating Energy and Cooling Load

Refer to Appendix C for the heating load and cooling load for the entire year with and without internal gains ('People').

Table 4 shows the total site and source energy consumption. As observed, the energy consumption is almost twice when the internal gains (people) are taken into consideration for the energy analysis in comparison to when they are not considered (Table 3).

Table 4: Site and Source Energy (Enriched Model)

	Total Energy [GJ]	Energy Per Total Building Area [MJ/m²]	Energy Per Conditioned Building Area [MJ/m²]
Total Site Energy	64.89	151.2	974.32
Net Site Energy	64.89	151.2	974.32
Total Source Energy	91.25	212.62	1370.12
Net Source Energy	91.25	212.62	1370.12

7. Conclusion

BIM is a rapidly growing process concept in the AEC due to its many advantages compared to the traditional process. BIM improves communication and coordination between all the stakeholders of a project. With the growing focus on energy simulations in the AEC industry due to environmental concerns, it is important that the data is exchanged effortlessly. The interoperable file formats that have been developed (for example, IFC and gbXML) help achieve improved coordination and data exchange between the stakeholders. Although the coordination and data transfer is drastically improved, there still exists a gap.

The main objective of this research was to improve or bridge the interoperability gap between ArchiCAD (BIM) and EnergyPlus (BEM). The major interoperability gap between the two software is the direct translation of data from IFC (generated from ArchiCAD) to EnergyPlus. This gap is overcome by developing a python script that reads the IFC data and translates it to EnergyPlus compatible file format. To develop the script for seamless integration between BIM and BEM, the following steps are identified: (1) recognizing relevant BIM input data; (2) BEM input data requirement; and (3) concise mapping between the two datasets.

The modeling of the building in ArchiCAD is one of the crucial steps. If the modeling is incorrect or inconsistent, the model generated in EnergyPlus will be inaccurate leading to incorrect simulation results. The following geometric IFC entities can be translated to IDF: *IfcWall*, *IfcSlab* (only rectangular floors and roofs), *IfcWindow*, *IfcDoor*. Subsequently, the thermal properties of the materials and the construction layers of the building element are accurately assigned. Using python to translate the data from IFC to EnergyPlus ensured that there is no incorrect mapping of data or any loss of data which are some of the issues observed when using other graphical interfaces based on EnergyPlus. This research approach initiates the potential and possibilities of translating IFC data to EnergyPlus readable data.

Finally, the ability to add custom *IfcPropertySet* with additional relevant data for simulation to the BIM model allows for improved integration between BIM and BEM. The enrichment of the BIM model with additional data (for example, space occupancy, lighting load, etc) also improves the reliability of the energy simulation results. This additional data can also be extracted using python to ensure there is no loss of data. The comparison of the simulation results of energy models with and without internal gains (space occupancy data) shows that the estimation of energy consumption strongly depends on the accuracy of the available input data.

8. Future Work

The approach used in this thesis presents a way to bridge the interoperability gap between BIM and BEM software but it still has its limitations. Some of the steps that can be taken to improve this approach and reduce the limitations are:

- The data extraction process can be extended to incorporate IFC4.
- The study currently only takes into account some IFC entities like *IfcWall*, *IfcSlab* but other entities can be used to represent a wall (*IfcWallStandardCase*) or a roof (*IfcRoof*). The python script can be further extended to include other entities.
- The study only works with a flat roof. The python script can be further extended to include a pitched roof and other types of roofs.
- The python script can be further extended to incorporate circular walls and walls at an angle (in plan view).
- Other internal gains (artificial lighting, electric appliances) data can be added to the BIM model for the enrichment and the extension of python script can be implemented to extract these data. This will improve the accuracy of the simulation results.

The current research approach has multiple areas of possible enhancements. Future work might be able to eliminate barriers and establish a fully automated and well-functioning tool which is usable in practice. Drawn conclusions and some of the formulated recommendations can function as input for future studies. Ultimately an improved conversion tool can contribute to a more developed BIM environment.

9. References

- [1] National Institute of Building Sciences, “FREQUENTLY ASKED QUESTIONS ABOUT THE NATIONAL BIM STANDARD-UNITED STATES.” [Online]. Available: <https://www.nationalbimstandard.org/faqs>. [Accessed: 05-Mar-2020].
- [2] “IFC what’s it for? What’s its connection with BIM?” [Online]. Available: <http://biblus.accasoftware.com/en/ifc-whats-it-for-whats-its-connection-with-bim/>. [Accessed: 09-Apr-2020].
- [3] Building Performance Institute Europe, *Europe’s Buildings Under The Microscope*. 2011.
- [4] “Paris Agreement.” [Online]. Available: https://ec.europa.eu/clima/policies/international/negotiations/paris_en. [Accessed: 05-Mar-2020].
- [5] Directorate-General for Climate Action (European Commission), “Going Climate-Neutral by 2050.”
- [6] M. Rahmani Asl, S. Zarrinmehr, M. Bergin, and W. Yan, “BPOpt: A framework for BIM-based performance optimization,” *Energy Build.*, vol. 108, pp. 401–412, 2015.
- [7] I. Kovacic and V. Zoller, “Building Life Cycle Optimization Tools for Early Design Phases.”
- [8] AIA California Council, “Integrated Project Delivery: A Guide,” 2007.
- [9] BIM Forum, “Level of Development (LOD) Specification Part I & Commentary,” no. April, 2019.
- [10] C. Eastman, P. Teicholz, R. Sacks, and K. Liston, *BIM Handbook*, 2nd Editio. John Wiley & Sons, Inc., 2011.
- [11] “Project Lifecycle Phases,” 2013. [Online]. Available: <https://www.bimframework.info/2013/12/project-lifecycle-phases.html>. [Accessed: 08-Mar-2020].
- [12] ASHRAE, “ANSI/ASHRAE Standard 209-2018 Energy Simulation Aided Design for Buildings Except Low-Rise Residential Buildings,” 2018.
- [13] V. Bazjanac, “Space boundary requirements for modeling of building geometry for energy and other performance simulation,” *Lawrence Berkeley Natl. Lab. Univ. California, Berkeley, California, USA*, no. November, pp. 16–18, 2010.

- [14] M. Weise, T. Liebich, R. See, V. Bazjanac, and T. Laine, "Implementation guide: Space boundaries for energy analysis," 2011.
- [15] buildingSMART, "IfcRelSpaceBoundary." [Online]. Available: https://standards.buildingsmart.org/IFC/RELEASE/IFC4/ADD2_TC1/HTML/schema/ifcproductextension/lexical/ifcrelspaceboundary.htm. [Accessed: 10-Mar-2020].
- [16] buildingSMART, "Industry Foundation Classes (IFC) - An Introduction." [Online]. Available: <https://technical.buildingsmart.org/standards/ifc>. [Accessed: 11-Mar-2020].
- [17] buildingSMART, "Industry Foundation Classes Version 4.1.0.0." [Online]. Available: https://standards.buildingsmart.org/IFC/RELEASE/IFC4_1/FINAL/HTML/. [Accessed: 27-Mar-2020].
- [18] buildingSMART, "IFC2x Edition 3." [Online]. Available: <https://standards.buildingsmart.org/IFC/RELEASE/IFC2x3/FINAL/HTML/>. [Accessed: 27-Mar-2020].
- [19] buildingSMART, "IfcDoc." [Online]. Available: <https://technical.buildingsmart.org/resources/ifcdoc/>. [Accessed: 14-Mar-2020].
- [20] J. T. O'Donnell *et al.*, "Transforming BIM to BEM: Generation of Building Geometry for the NASA Ames Sustainability Base BIM," California, 2013.
- [21] I. Ivanova, K. Kiesel, and A. Mahdavi, "BIM-generated data models for EnergyPlus: A comparison of gbXML and IFC formats," *Build. Simul. Appl.*, vol. 2015-Febru, pp. 407–414, 2015.
- [22] H. Gao, C. Koch, and Y. Wu, "Building information modelling based building energy modelling: A review," *Appl. Energy*, vol. 238, no. March, pp. 320–343, 2019.
- [23] "Import IFC." [Online]. Available: http://nrel.github.io/OpenStudio-user-documentation/tutorials/tutorial_ifcimport/. [Accessed: 16-Mar-2020].
- [24] N. Yu, "INFORMATION INTEROPERABILITY BETWEEN BUILDING INFORMATION MODELING AUTHORIZING TOOLS AND SIMULATION TOOLS TO SUPPORT ENERGY EFFICIENT BUILDING DESIGN," The Pennsylvania State University, 2014.
- [25] M. Zhiliang, Z. Tonghua, W. Zhenhua, and F. Yan, "TRANSFORMATION FROM IFC DATA OF DESIGN RESULTS TO IDF DATA FOR ANALYSIS OF BUILDING'S ENERGY CONSUMPTION."
- [26] T. Nahan, "Architect 's Guide to Building Performance," 2019.

- [27] R. Sacks, L. Ma, R. Yosef, A. Borrmann, S. Daum, and U. Kattel, “Semantic Enrichment for Building Information Modeling: Procedure for Compiling Inference Rules and Operators for Complex Geometry,” *J. Comput. Civ. Eng.*, vol. 31, no. 6, pp. 1–12, 2017.
- [28] Graphisoft, “ARCHICAD 23 Reference Guide.”
- [29] U.S. Department of Energy, *Input Output Reference*. 2019.

10. Appendices

A. ArchiCAD Model

A.1 Model Data

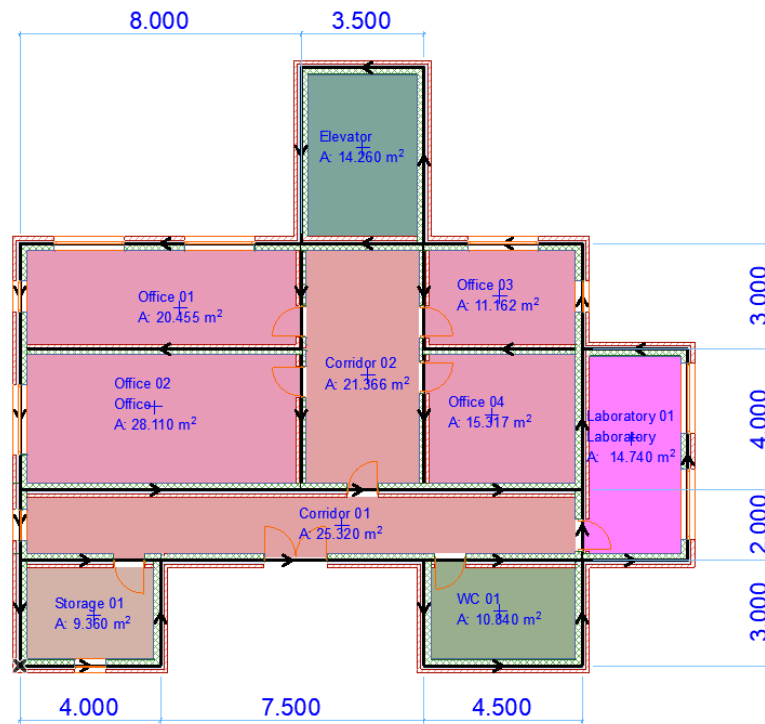


Figure 53: Ground Floor Dimensions

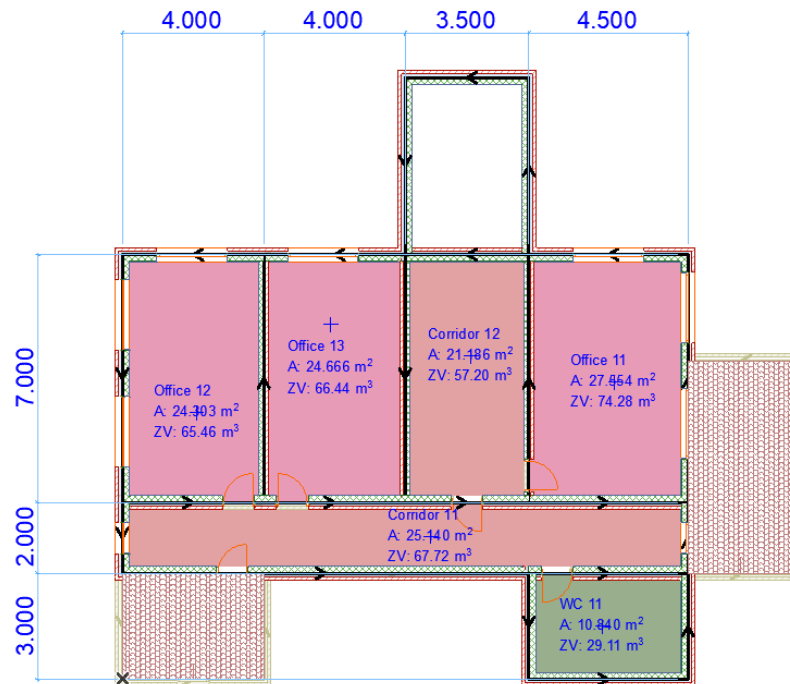


Figure 54: First Floor Dimensions

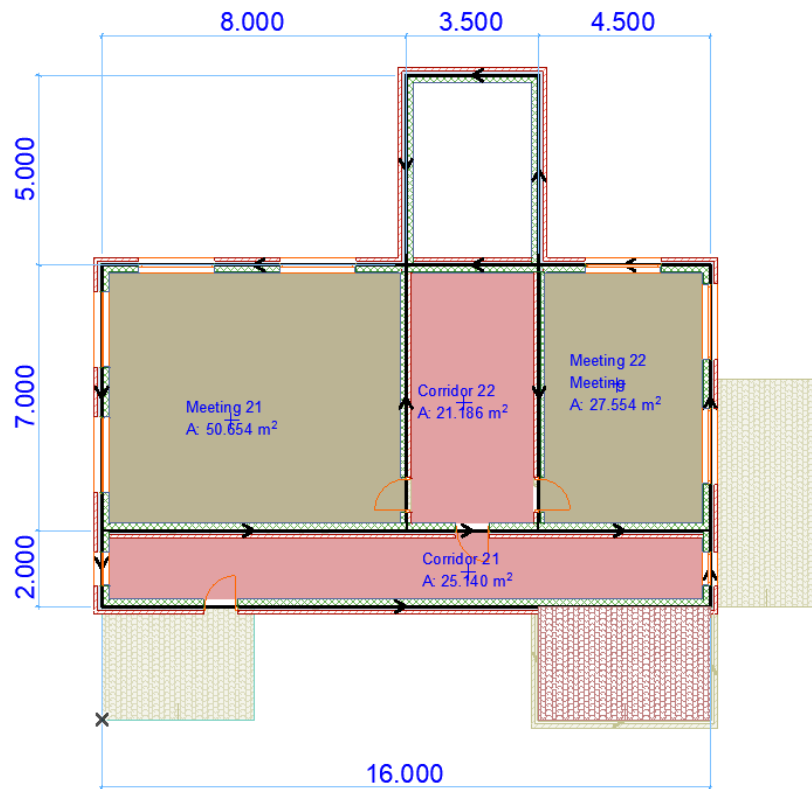


Figure 55: Second Floor Dimensions

Components by Layers					
Layer Name	Name	Density	Heat Capacity	Thermal Conductivity	Skin Thickness
External Walls					
	Air Space	1.20 kg/m ³	1008.00 J/kgK	0.15 W/mK	0.05
	Brick	1,500.00 kg/m ³	840.00 J/kgK	0.58 W/mK	0.10
	Concrete Block - Structural (1)	1,400.00 kg/m ³	880.00 J/kgK	0.60 W/mK	0.22
	Insulation - Plastic Hard	28.00 kg/m ³	1450.00 J/kgK	0.03 W/mK	0.03
	Plaster - Gypsum	1,300.00 kg/m ³	1000.00 J/kgK	0.57 W/mK	0.01
Internal Walls					
	Air Space	1.20 kg/m ³	1008.00 J/kgK	0.15 W/mK	0.05
	Brick	1,500.00 kg/m ³	840.00 J/kgK	0.58 W/mK	0.10
	Concrete Block - Structural	1,400.00 kg/m ³	880.00 J/kgK	0.60 W/mK	0.10
	Insulation - Plastic Hard	28.00 kg/m ³	1450.00 J/kgK	0.03 W/mK	0.03
	Plaster - Gypsum	1,300.00 kg/m ³	1000.00 J/kgK	0.57 W/mK	0.01
Roof					
	GENERIC - EXTERNAL MEMBRANE	1,390.00 kg/m ³	900.00 J/kgK	0.17 W/mK	0.10
	GENERIC - PREFABRICATED	2,400.00 kg/m ³	1000.00 J/kgK	2.50 W/mK	0.20
Slab					
	GENERIC - INTERNAL CLADDING	900.00 kg/m ³	1000.00 J/kgK	0.25 W/mK	0.10
	GENERIC - STRUCTURAL	2,300.00 kg/m ³	1000.00 J/kgK	2.30 W/mK	0.20

Figure 56: Modeled Data (Building Element and Properties)

A.2 IFC Export Settings

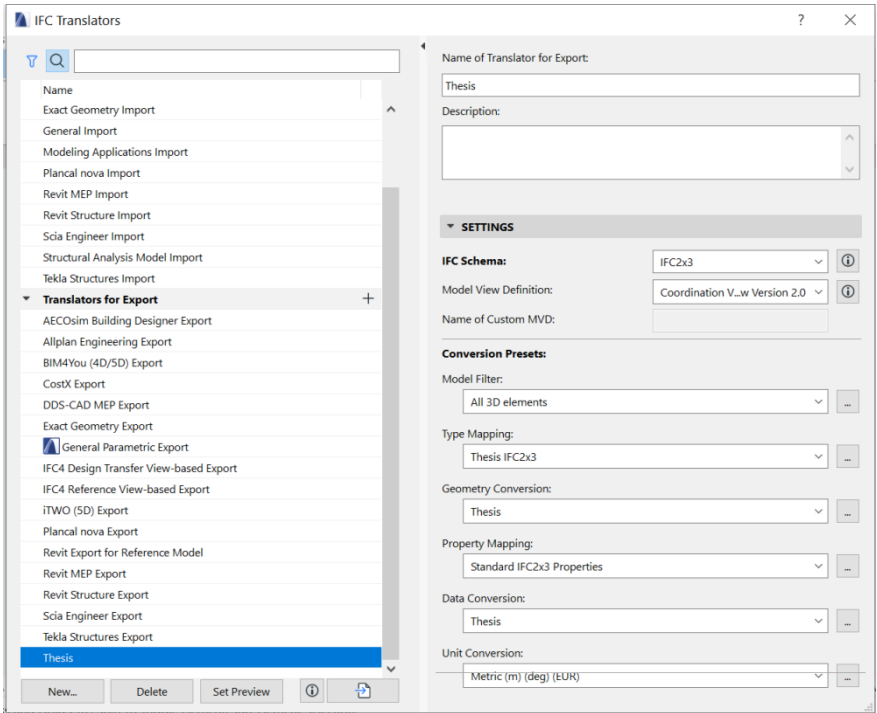


Figure 57: IFC Export Settings

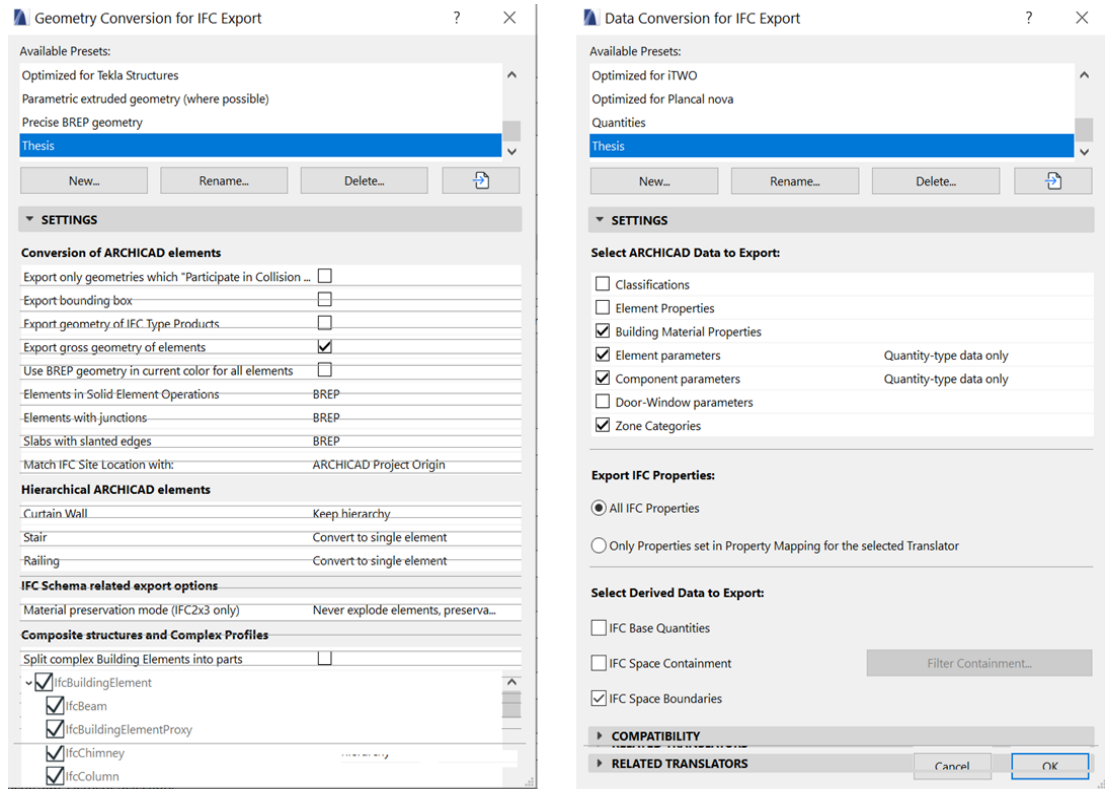


Figure 58: Geometry and Data Conversion (IFC Export Settings)

B. Class Object – Detailed Input

B.1 Input Data

IDF Class Object	Parameter	Required Parameter	Extracted from IFC
Building	Name	Yes	Yes
	North Axis {deg}	Yes	Yes
	Terrain	Yes	No
	Loads Convergence Tolerance Value	Yes	No
	Temperature Convergence Tolerance	Yes	No
	Solar Distribution	Yes	No
	Maximum Number of Warmup Days	Yes	No
	Minimum Number of Warmup Days	Yes	No
Site:Location	Name	Yes	Yes
	Latitude {deg}	No	Yes
	Longitude {deg}	No	Yes
	Time Zone {hr}	No	No
	Elevation {m}	No	Yes
Material	Name	Yes	Yes
	Roughness	Yes	No
	Thickness {m}	Yes	Yes
	Conductivity {W/m-K}	Yes	Yes
	Density {kg/m3}	Yes	Yes
	Specific Heat {J/kg-K}	Yes	Yes
	Thermal Absorptance	No	No
	Solar Absorptance	No	No
	Visible Absorptance	No	No

Construction	Name	Yes	Yes
	Outside Layer	Yes	Yes
	Layer 2	Yes/No	Yes/No
	.	Yes/No	Yes/No
	.	Yes/No	Yes/No
Zone	Name	Yes	Yes
	Direction of Relative North {deg}	No	No
	X Origin {m}	No	No
	Y Origin {m}	No	No
	Z Origin {m}	No	No
	Type	No	No
	Multiplier	No	No
	Ceiling Height {m}	No	Yes
	Volume {m3}	No	Yes
	Floor Area {m2}	No	Yes
	Zone Inside Convection Algorithm	No	No
	Zone Outside Convection Algorithm	No	No
	Part of Total Floor Area	No	No
BuildingSurface:Detailed	Name	Yes	Yes
	Surface Type	Yes	Yes
	Construction Name	Yes	Yes
	Zone Name	Yes	Yes
	Outside Boundary Condition	Yes	Yes
	Outside Boundary Condition Object	Yes/No	Yes/No
	Sun Exposure	No	No
	Wind Exposure	No	No
	View Factor to Ground	No	No

	Number of Vertices	No	No
	Vertex 1 X-,Y-,Z-Coordinate	Yes	Yes
	Vertex 2 X-,Y-,Z-Coordinate	Yes	No
	Vertex 3 X-,Y-,Z-Coordinate	Yes	No
	Vertex 4 X-,Y-,Z-Coordinate	Yes	No
FenestrationSurface:Detailed	Name	Yes	Yes
	Surface Type	Yes	Yes
	Construction Name	Yes	Yes
	Building Surface Name	Yes	Yes
	Outside Boundary Condition	Yes/No	Yes/No
	View Factor to Ground	No	No
	Frame and Divider Name	No	No
	Multiplier	No	No
	Number of Vertices	No	No
	Vertex 1 X-,Y-,Z-Coordinate	Yes	Yes
	Vertex 2 X-,Y-,Z-Coordinate	Yes	No
	Vertex 3 X-,Y-,Z-Coordinate	Yes	No
	Vertex 4 X-,Y-,Z-Coordinate	Yes/No	No

B.2 Simulation Class Objects

IDF Class Object	Parameter	Value
Version	Version	9.2
SimulationControl	Do Zone Sizing Calculation	Yes
	Do System Sizing Calculation	No
	Do Plant Sizing Calculation	No
	Run Simulation for Sizing Periods	No
	Run Simulation for Weather File Run Periods	Yes

Timestep	Number of Timesteps per Hour	4
RunPeriod	Name	RunPeriod
	Begin Month	1
	Begin Day of Month	1
	Begin Year	
	End Month	12
	End Day of Month	31
	End Year	
	Day of Week for Start Day	
	Use Weather File Holidays and Special Days	Yes
	Use Weather File Daylight Saving Period	Yes
	Apply Weekend Holiday Rule	No
	Use Weather File Rain Indicators	Yes
	Use Weather File Snow Indicators	Yes
	Treat Weather as Actual	No
GlobalGeometryRules	Starting Vertex Position	LowerLeftCorner
	Vertex Entry Direction	Counterclockwise
	Coordinate System	Relative

B.3 HVAC Class Objects

IDF Class Object	Parameter	Value
ZoneInfiltration: DesignFlowRate	Name	
	Zone or ZoneList Name	
	Schedule Name	INFIL-SCH
	Design Flow Rate Calculation Method	flow/zone
	Design Flow Rate {m3/s}	0.0167
	Flow per Zone Floor Area {m3/s-m2}	

	Flow per Exterior Surface Area {m3/s-m2}	
	Air Changes per Hour {1/hr}	0
	Constant Term Coefficient	0
	Temperature Term Coefficient	0
	Velocity Term Coefficient	0.2237
	Velocity Squared Term Coefficient	0
HVACTemplate:Thermostat	Name	All Zones
	Heating Setpoint Schedule Name	Htg-SetP-Sch
	Constant Heating Setpoint {C}	
	Cooling Setpoint Schedule Name	Clg-SetP-Sch
	Constant Cooling Setpoint {C}	
HVACTemplate: Zone:FanCoil	Zone Name	
	Template Thermostat Name	All Zones
	Supply Air Maximum Flow Rate {m3/s}	Autosize
	Zone Heating Sizing Factor	
	Zone Cooling Sizing Factor	
	Outdoor Air Method	flow/person
	Outdoor Air Flow Rate per Person {m3/s}	0.00944
	Outdoor Air Flow Rate per Zone Floor Area {m3/s-m2}	0
	Outdoor Air Flow Rate per Zone {m3/s}	0
	System Availability Schedule Name	FanAvailSched
	Supply Fan Total Efficiency	0.7
	Supply Fan Delta Pressure {Pa}	75
	Supply Fan Motor Efficiency	0.9
	Supply Fan Motor in Air Stream Fraction	1
	Cooling Coil Type	ChilledWater

	Cooling Coil Availability Schedule Name	
	Cooling Coil Design Setpoint {C}	12.5
	Heating Coil Type	HotWater
	Heating Coil Availability Schedule Name	
	Heating Coil Design Setpoint {C}	50
	Dedicated Outdoor Air System Name	
	Zone Cooling Design Supply Air Temperature Input Method	SupplyAirTemperature
	Zone Cooling Design Supply Air Temperature Difference {deltaC}	
	Zone Heating Design Supply Air Temperature Input Method	SupplyAirTemperature
HVACTemplate:Plant: ChilledWaterLoop	Name	Chilled Water Loop
	Pump Schedule Name	
	Pump Control Type	INTERMITTENT
	Chiller Plant Operation Scheme Type	Default
	Chiller Plant Equipment Operation Schemes Name	
	Chilled Water Setpoint Schedule Name	
	Chilled Water Design Setpoint {C}	7.22
	Chilled Water Pump Configuration	ConstantPrimaryNoSecondary
	Primary Chilled Water Pump Rated Head {Pa}	179352
	Secondary Chilled Water Pump Rated Head {Pa}	179352
	Condenser Plant Operation Scheme Type	Default
	Condenser Equipment Operation Schemes Name	
	Condenser Water Temperature Control Type	
	Condenser Water Setpoint Schedule Name	

	Condenser Water Design Setpoint {C}	29.4
	Condenser Water Pump Rated Head {Pa}	179352
	Chilled Water Setpoint Reset Type	OutdoorAirTemperatureReset
	Chilled Water Setpoint at Outdoor Dry-Bulb Low {C}	12.2
	Chilled Water Reset Outdoor Dry-Bulb Low {C}	15.6
	Chilled Water Setpoint at Outdoor Dry-Bulb High {C}	6.7
	Chilled Water Reset Outdoor Dry-Bulb High {C}	26.7
	Chilled Water Primary Pump Type	
	Chilled Water Secondary Pump Type	
	Condenser Water Pump Type	
	Chilled Water Supply Side Bypass Pipe	
	Chilled Water Demand Side Bypass Pipe	
	Condenser Water Supply Side Bypass Pipe	
	Condenser Water Demand Side Bypass Pipe	
	Fluid Type	
	Loop Design Delta Temperature {deltaC}	
	Minimum Outdoor Dry Bulb Temperature {C}	7.22
HVACTemplate:Plant:Chiller	Name	Main Chiller
	Chiller Type	ElectricReciprocatingChiller
	Capacity {W}	Autosize
	Nominal COP {W/W}	3.2
	Condenser Type	WaterCooled
	Priority	1
	Sizing Factor	
HVACTemplate:Plant:Tower	Name	Main Tower
	Tower Type	SingleSpeed

	High Speed Nominal Capacity {W}	Autosize
	High Speed Fan Power {W}	Autosize
	Low Speed Nominal Capacity {W}	Autosize
	Low Speed Fan Power {W}	Autosize
	Free Convection Capacity {W}	Autosize
	Priority	1
	Sizing Factor	
HVACTemplate:Plant: HotWaterLoop	Name	Hot Water Loop
	Pump Schedule Name	
	Pump Control Type	INTERMITTENT
	Hot Water Plant Operation Scheme Type	Default
	Hot Water Plant Equipment Operation Schemes Name	
	Hot Water Setpoint Schedule Name	
	Hot Water Design Setpoint {C}	82
	Hot Water Pump Configuration	ConstantFlow
	Hot Water Pump Rated Head {Pa}	179352
	Hot Water Setpoint Reset Type	OutdoorAirTemperatureReset
	Hot Water Setpoint at Outdoor Dry-Bulb Low {C}	82.2
	Hot Water Reset Outdoor Dry-Bulb Low {C}	-6.7
	Hot Water Setpoint at Outdoor Dry-Bulb High {C}	65.6
	Hot Water Reset Outdoor Dry-Bulb High {C}	10
HVACTemplate: Plant:Boiler	Name	Main Boiler
	Boiler Type	HotWaterBoiler
	Capacity {W}	Autosize
	Efficiency	0.8
	Fuel Type	NaturalGas

	Priority	1
	Sizing Factor	
Sizing:Parameters	Heating Sizing Factor	1.2
	Cooling Sizing Factor	1.2
Site:WaterMainsTemperature	Calculation Method	CORRELATION
	Temperature Schedule Name	
	Annual Average Outdoor Air Temperature {C}	9.69
	Maximum Difference In Monthly Average Outdoor Air Temperatures {deltaC}	28.10

B.4 People and Schedule Class Objects

IDF Class Object	Parameter	Value
People	Name	Office 12People
	Zone or ZoneList Name	Office 12
	Number of People Schedule Name	Office 12Schedule
	Number of People Calculation Method	4
	Number of People	
	People per Zone Floor Area {person/m2}	
	Zone Floor Area per Person {m2/person}	
	Fraction Radiant	
	Sensible Heat Fraction	
	Activity Level Schedule Name	ActSchd
	Carbon Dioxide Generation Rate {m3/s-W}	3.82E-08
	Enable ASHRAE 55 Comfort Warnings	No
Schedule:Compact	Name	Office 12Schedule
	Schedule Type Limits Name	Fraction
	Field 1	Through: 12/31
	Field 2	For: WeekDays CustomDay1 CustomDay2

	Field 3	Until: 8:00
	Field 4	0
	Field 5	Until: 17:00
	Field 6	1
	Field 7	Until: 24:00
	Field 8	0
	Field 9	For: Weekends Holiday
	Field 10	Until: 24:00
	Field 11	0
	Field 12	For: SummerDesignDay
	Field 13	Until: 24:00
	Field 14	1
	Field 15	For: WinterDesignDay
	Field 16	Until: 24:00
	Field 17	1

C. Simulation Results

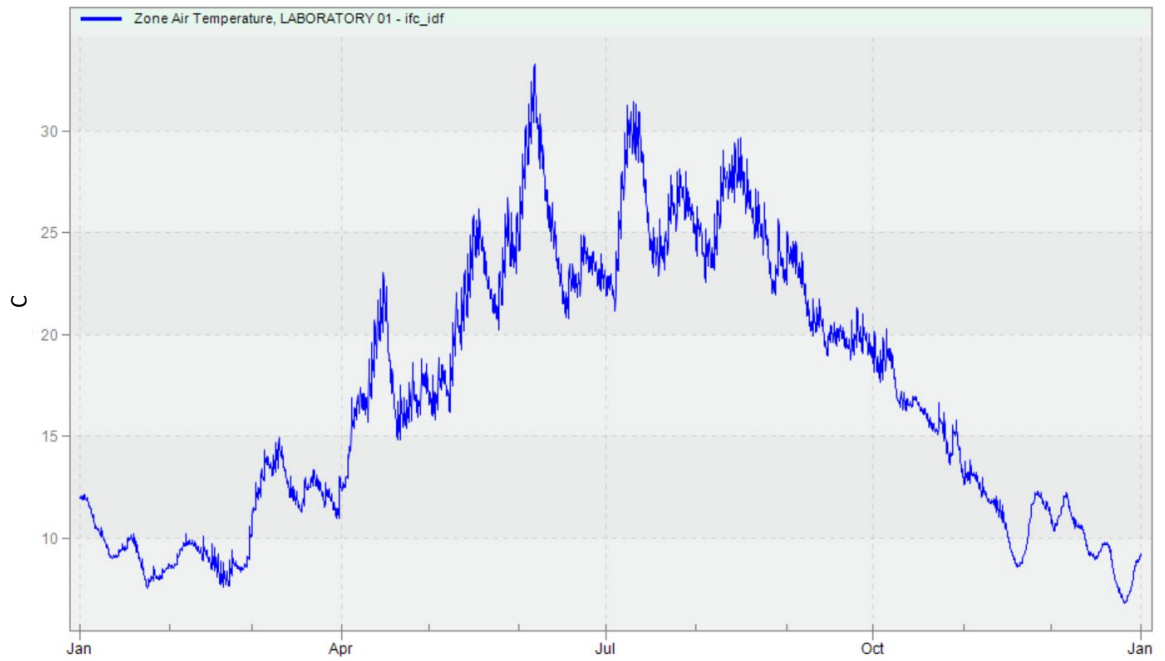


Figure 59: Zone Air Temperature - Laboratory 01 (HVAC off)

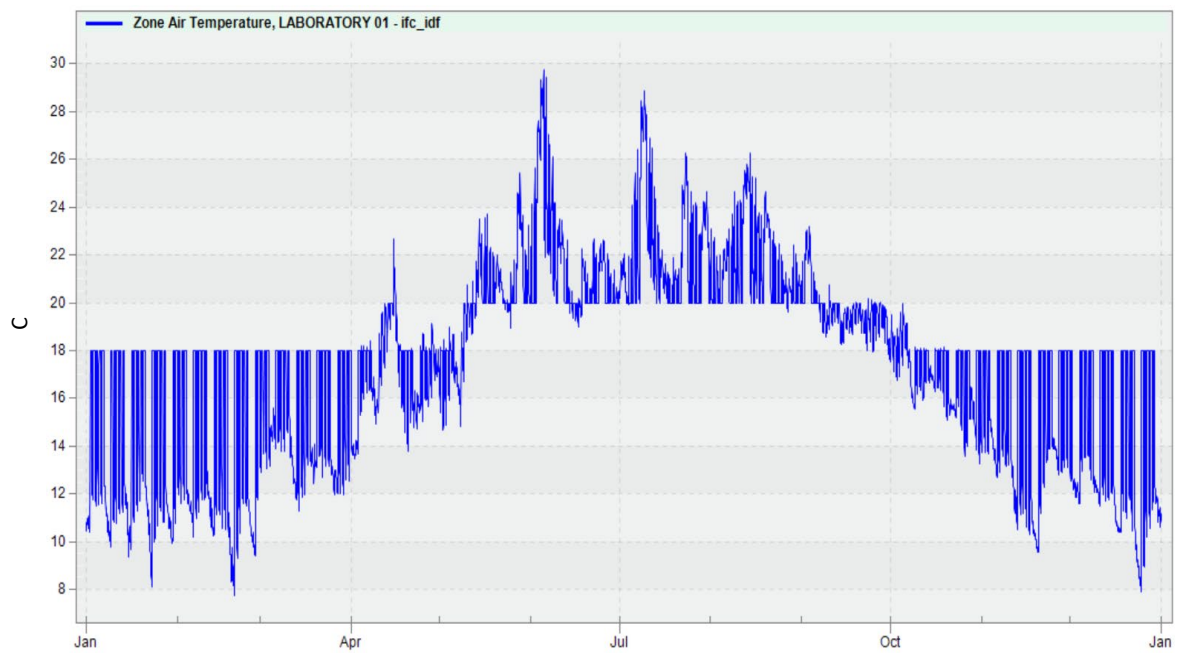


Figure 60: Zone Air Temperature - Laboratory 01 (HVAC on)

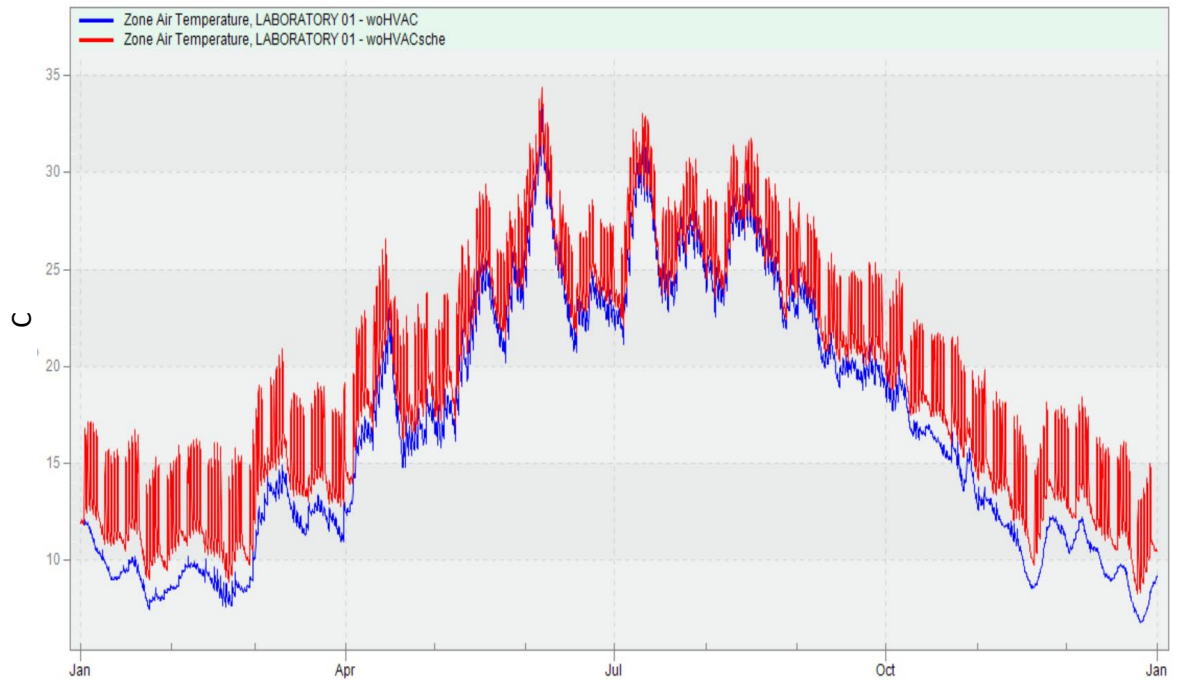


Figure 61: Zone Air Temperature - Laboratory 01 (with and without 'People')



Figure 62: Zone Air Temperature - Office 12 (HVAC off)

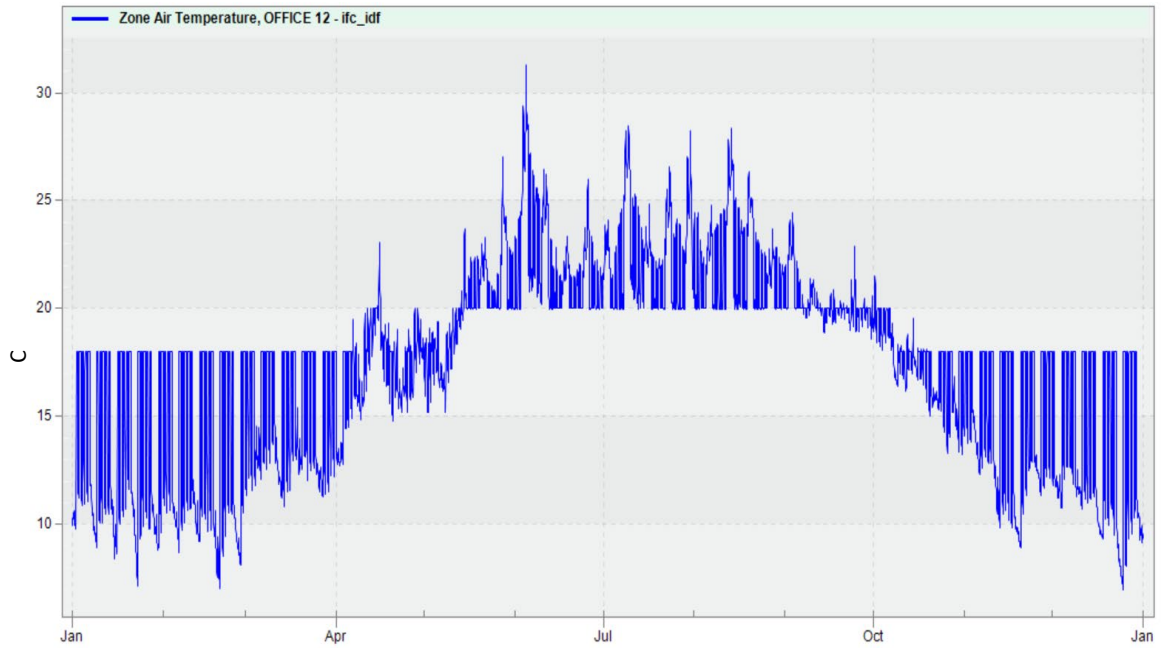


Figure 63: Zone Air Temperature - Office 12 (HVAC on)

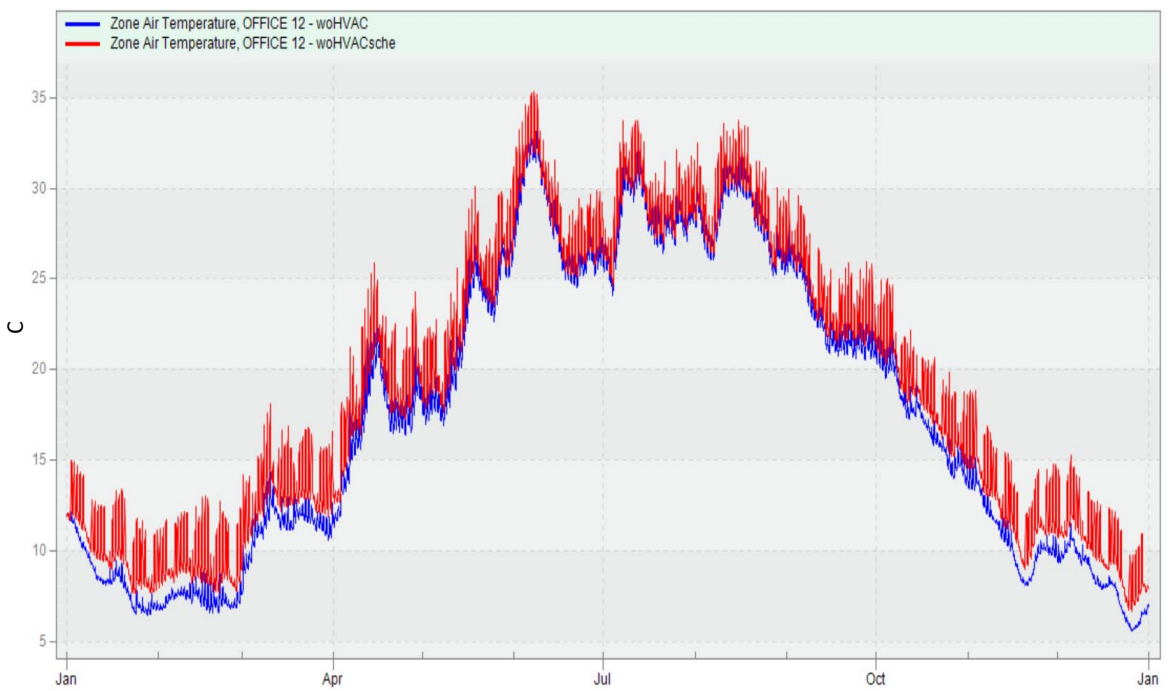


Figure 64: Zone Air Temperature - Office 12 (with and without 'People')

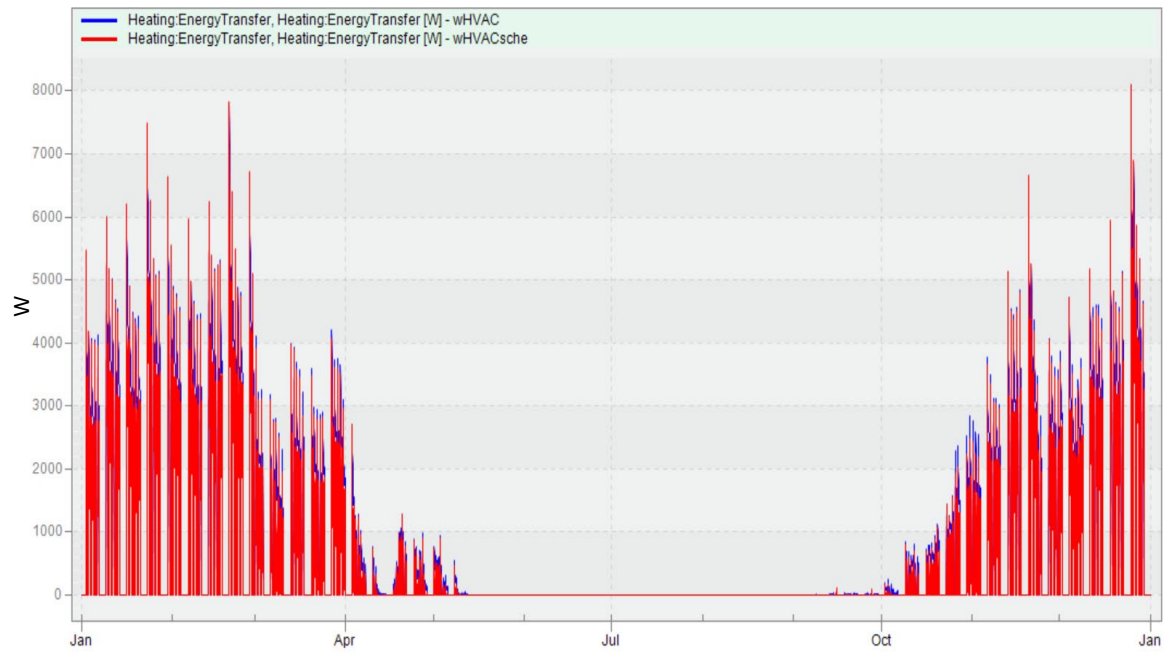


Figure 65: Heating Load (with and without 'People')

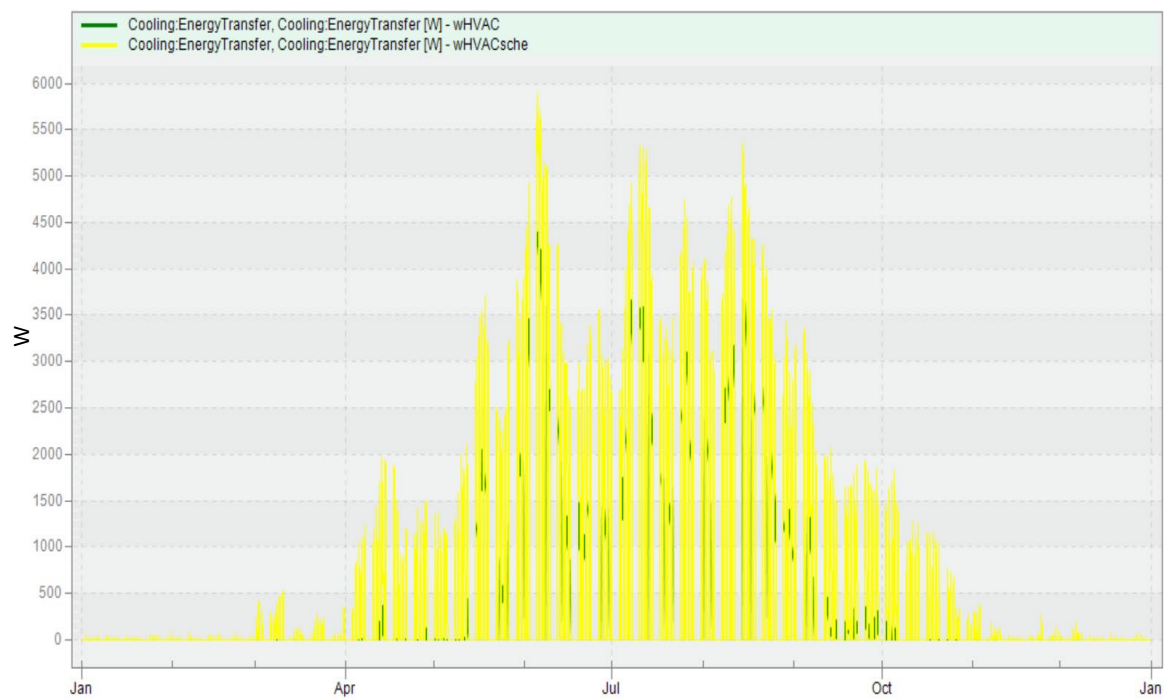


Figure 66: Cooling Load (with and without 'People')

D. Python Script

D.1 Building

```
import math
import ifcopenshell

def north(x, y):
    theta = math.degrees(math.atan2(y,x)) - 90
    return theta

ifc_file = ifcopenshell.open('thesis.ifc')
idf = open('ifc_idf.idf', 'w')

building = ifc_file.by_type('IfcProject')[0]

idf.write('Building,\n')

name = building.Name
idf.write('\t'+str(name)+'\n')

trueNorth = building.RepresentationContexts[0].TrueNorth.DirectionRatios
deg = north(trueNorth[0], trueNorth[1])
idf.write('\t'+str(deg)+';\n')

idf.close()
```

D.2 Site:Location

```
import ifcopenshell

ifc_file = ifcopenshell.open('thesis.ifc')
idf = open('ifc_idf.idf', 'a')

def dms_dd(d, m, s):
    dd = d + float(m)/60 + float(s)/3600
    return dd

site = ifc_file.by_type('IfcSite')

idf.write('Site:Location,\n')

for element in site:
    Name = element.SiteAddress.Town

    idf.write('\t'+str(Name)+'\n')

    a,b,c = element.RefLatitude[0:3]
    lat = dms_dd(a,b,c)
    idf.write('\t'+str(lat)+'\n')

    x,y,z = element.RefLongitude[0:3]
    long = dms_dd(x,y,z)
    idf.write('\t'+str(long)+'\n')

    idf.write('\t,\n')

    idf.write('\t'+str(element.RefElevation)+';\n\n')

idf.close()
```

D.3 Zone

```
import ifcopenshell

ifc_file = ifcopenshell.open('thesis.ifc')
idf = open('ifc_idf.idf', 'a')

Spaces = ifc_file.by_type('IfcSpace')

for space in Spaces:

    idf.write('Zone,\n')

    idf.write('\t'+str(space.LongName)+'\n') #Name

    idf.write('\t,\n') #RelativeNorth
```

```

#Origin
idf.write('\t,\n')
idf.write('\t,\n')
idf.write('\t,\n')

#Type and Multiplier
idf.write('\t1,\n')
idf.write('\t1,\n')

#CeilingHeight,Volume,Area
idf.write('\t,\n')

spa = space.IsDefinedBy

for sp in spa:
    if sp.is_a('IfcRelDefinesByProperties'):
        if sp.RelatingPropertyDefinition.is_a('IfcElementQuantity'):
            prop = sp.RelatingPropertyDefinition.Quantities
            for pro in prop:
                if pro.Name == 'Net Volume':
                    volume = pro.VolumeValue
                    idf.write('\t'+str(volume)+',\n')
            for pro in prop:
                if pro.Name == 'Area':
                    area = pro.AreaValue
                    idf.write('\t'+str(area)+',\n')

#ConvectionAlgorithm
idf.write('\t,\n')
idf.write('\t,\n')

idf.write('\tYes;\n\n') #PartOfFloorArea

idf.close()

```

D.4 Material

```

import ifcopenshell

ifc_file = ifcopenshell.open('thesis.ifc')
idf = open('ifc_idf.idf','a')

BuildingElement = ifc_file.by_type('IfcBuildingElement')

mList = []
for element in BuildingElement:
    if element.is_a('IfcDoor') or element.is_a('IfcWindow'):
        continue

    material = element.IsDefinedBy

    mat_list=[]
    thick_list=[]
    for mat in material:
        if mat.is_a('IfcRelDefinesByProperties'):
            if mat.RelatingPropertyDefinition.Name == 'Material Properties':
                prop = mat.RelatingPropertyDefinition.HasProperties
                for compProp in prop:
                    mat_n = compProp.HasProperties
                    for x in mat_n:
                        matList = []
                        if x.Name not in mList:
                            if x.Name.find('Air')!=-1:
                                matList.append('Material:AirGap')
                                matList.append(x.Name)
                                mList.append(x.Name)
                                xy = x.HasProperties
                                for y in xy:
                                    if y.Name == "ThermalConductivity":
                                        matList.append(y.NominalValue.wrappedValue)
                            else:
                                matList.append('Material')
                                matList.append(x.Name)
                                mList.append(x.Name)
                                matList.append('\t,\n') #Roughness
                                xy = x.HasProperties
                                for y in xy:
                                    if y.Name == "ThermalConductivity":
                                        matList.append(y.NominalValue.wrappedValue)

```

```

        for y in xy:
            if y.Name == "MassDensity":
                matList.append(y.NominalValue.wrappedValue)
        for y in xy:
            if y.Name == "SpecificHeatCapacity":
                matList.append(y.NominalValue.wrappedValue)

        mat_list.append(matList)

    if mat.RelatingPropertyDefinition.Name == 'Component Quantities':
        thick = mat.RelatingPropertyDefinition.Quantities
        if element.is_a('IfcSlab'):
            for compQ in thick:
                thic = compQ.HasQuantities
                for t in thic:
                    if t.Name == 'Skin Thickness':
                        thick_list.append(t.LengthValue)
        else:
            for compQ in reversed(thick):
                thic = compQ.HasQuantities
                for t in thic:
                    if t.Name == 'Skin Thickness':
                        thick_list.append(t.LengthValue)

    for i in range(len(mat_list)):
        if mat_list[i] == []:
            continue
        else:
            li=list(mat_list[i])
            if mat_list[i][1].find('Air') != -1:
                li[2] = thick_list[i]/mat_list[i][2]
            else:
                li.insert(3,thick_list[i])

            for j in range(len(li)):
                if j == len(li) - 1:
                    idf.write('\t'+str(li[j])+';\n\n')
                else:
                    idf.write('\t'+str(li[j])+',\n')

idf.close()

```

D.5 Construction

```

import ifcopenshell

def constr_layer(entity):
    construction = entity.HasAssociations[0].RelatingMaterial.Materials

    for layer in construction:
        layer_list.append(layer.Name)

    if element.is_a('IfcWall'):
        layer_list.reverse()

    for i in range(len(layer_list)):
        if i == len(layer_list) - 1:
            idf.write('\t'+str(layer_list[i])+';\n\n')
        else:
            idf.write('\t'+str(layer_list[i])+',\n')

ifc_file = ifcopenshell.open('thesis.ifc')
idf = open('ifc_idf.idf','a')

BuildingElement = ifc_file.by_type('IfcBuildingElement')

constr = []
for element in BuildingElement:
    if element.is_a('IfcDoor') or element.is_a('IfcWindow'):
        continue

    Name = element.IsDefinedBy

    for name in Name:
        if name.is_a('IfcRelDefinesByType'):
            if name.RelatingType.Name not in constr:

                idf.write('Construction,\n')

                constr.append(name.RelatingType.Name)

```

```

idf.write('\t'+str(name.RelatingType.Name)+'\n')

layer_list = []
if element.is_a('IfcSlab'):
    if element.PredefinedType == 'ROOF':
        construction = element.HasAssociations[0].RelatingMaterial.ForLayerSet.MaterialLayers

        material = []
        for mat in construction:
            material.append(mat.Material.Name)

        for i in range(len(material)):
            if i == len(material) - 1:
                idf.write('\t'+str(material[i])+';\n\n')
            else:
                idf.write('\t'+str(material[i])+',\n')

    else:
        constr_layer(element)

else:
    constr_layer(element)

idf.close()

```

D.6 BuildingSurface and Shading

```

import ifcopenshell
import ifcopenshell.geom
from OCC.Core import TopoDS

settings = ifcopenshell.geom.settings()
settings.set(settings.USE_PYTHON_OPENCASCADE, True)

def global_coord(entity):
    elem = ifcopenshell.geom.create_shape(settings, entity)
    shape = TopoDS.TopoDS_Iterator(elem.geometry).Value()
    transform = shape.Location().Transformation().TranslationPart().Coord()

    x1,y1,z1 = transform[0],transform[1],transform[2]
    idf.write('\t'+str(x1)+' '+str(y1)+' '+str(z1)+'\n')
    return x1,y1,z1

def slabDim(entity):
    for props in entity.IsDefinedBy:
        if props.is_a('IfcRelDefinesByProperties'):
            if props.RelatingPropertyDefinition.is_a('IfcPropertySet'):
                prop = props.RelatingPropertyDefinition.HasProperties
                for pro in prop:
                    if pro.Name == 'Length':
                        Length = pro.NominalValue.wrappedValue
                    if pro.Name == 'Width':
                        Width = pro.NominalValue.wrappedValue
                return Length,Width

def internal(entity):
    if entity.is_a('IfcSlab'):
        if entity.ContainedInStructure[0].RelatingStructure.Name == 'Ground Floor':
            idf.write('\t,\n')
        else:
            idf.write(str(entity.ProvidesBoundaries[0].RelatingSpace.LongName)+'\n')
    else:
        idf.write(str(entity.ProvidesBoundaries[-1].RelatingSpace.LongName)+'\n')
    idf.write('\tNoSun,\n')
    idf.write('\tNoWind,\n')

def external(entity):
    idf.write('\tOutdoors,\n')
    idf.write('\t,\n')
    idf.write('\tSunExposed,\n')
    idf.write('\tWindExposed,\n')

def xpos():
    x2,y2,z2 = x1+Length, y1, z1
    x3,y3,z3 = x1+Length, y1, z1+Height
    x4,y4,z4 = x1, y1, z1+Height
    idf.write('\t'+str(x2)+' '+str(y2)+' '+str(z2)+'\n')
    idf.write('\t'+str(x3)+' '+str(y3)+' '+str(z3)+'\n')

```

```

idf.write('\t'+str(x4)+','+str(y4)+','+str(z4)+';\n\n')

def xneg():
    x2,y2,z2 = x1-Length, y1, z1
    x3,y3,z3 = x1-Length, y1, z1+Height
    x4,y4,z4 = x1, y1, z1+Height
    idf.write('\t'+str(x2)+','+str(y2)+','+str(z2)+';\n')
    idf.write('\t'+str(x3)+','+str(y3)+','+str(z3)+';\n')
    idf.write('\t'+str(x4)+','+str(y4)+','+str(z4)+';\n\n')

def ypos():
    x2,y2,z2 = x1, y1+Length, z1
    x3,y3,z3 = x1, y1+Length, z1+Height
    x4,y4,z4 = x1, y1, z1+Height
    idf.write('\t'+str(x2)+','+str(y2)+','+str(z2)+';\n')
    idf.write('\t'+str(x3)+','+str(y3)+','+str(z3)+';\n')
    idf.write('\t'+str(x4)+','+str(y4)+','+str(z4)+';\n\n')

def yneg():
    x2,y2,z2 = x1, y1-Length, z1
    x3,y3,z3 = x1, y1-Length, z1+Height
    x4,y4,z4 = x1, y1, z1+Height
    idf.write('\t'+str(x2)+','+str(y2)+','+str(z2)+';\n')
    idf.write('\t'+str(x3)+','+str(y3)+','+str(z3)+';\n')
    idf.write('\t'+str(x4)+','+str(y4)+','+str(z4)+';\n\n')

ifc_file = ifcopenshell.open('thesis.ifc')
idf = open('ifc_idf.idf','a')

BuildingElement = ifc_file.by_type('IfcBuildingElement')

for element in BuildingElement:
    #ShadingElement
    if element.ProvidesBoundaries == ():
        idf.write('Shading:Building,\n')
        idf.write('\t'+str(element.Name)+';\n')
        idf.write('\t,\n')
        idf.write('\t0,\n')
        global_coord(element)
        Length, Height = slabDim(element)
        idf.write('\t'+str(Length)+';\n')
        idf.write('\t'+str(Height)+';\n')

    else:
        #BuildingSurface or FenestrationSurface
        if element.is_a('IfcDoor') or element.is_a('IfcWindow'):
            continue
        else:
            idf.write('BuildingSurface:Detailed,\n')

            idf.write('\t'+str(element.Name)+';\n')

        #SurfaceType
        if element.is_a('IfcWall'):
            idf.write('\tWall,\n')
        else:
            if element.is_a('IfcSlab'):
                if element.PredefinedType == 'ROOF':
                    idf.write('\tRoof,\n')
                else:
                    idf.write('\tFloor,\n')

        #Construction
        Name = element.IsDefinedBy
        for name in Name:
            if name.is_a('IfcRelDefinesByType'):
                idf.write('\t'+str(name.RelatingType.Name)+';\n')

        #Zone
        if element.is_a('IfcWall'):
            Space = element.ProvidesBoundaries[0]
        else:
            Space = element.ProvidesBoundaries[-1]

        idf.write('\t'+str(Space.RelatingSpace.LongName)+';\n')

        #Outside Boundary Condition/Outside Boundary Condition Object/Sun and Wind Conditions
        if element.is_a('IfcSlab'):
            if element.PredefinedType != 'ROOF':
                if element.ContainedInStructure[0].RelatingStructure.Name == 'Ground Floor':

```



```

        idf.write('\tGround,\n')
    else:
        idf.write('\tZone,\n')
        internal(element)
    else:
        external(element)
else:
    if Space.InternalOrExternalBoundary == 'EXTERNAL':
        external(element)

    else:
        idf.write('\tZone,\n')
        internal(element)

#View Factor to Ground
if element.is_a('IfcWall'):
    idf.write('\t0.5,\n')
else:
    if element.PredefinedType == 'ROOF':
        idf.write('\t0,\n')
    else:
        idf.write('\t1,\n')

idf.write('\t,\n') #Number of Vertices

#Vertices
x1,y1,z1 = global_coord(element)

if element.is_a('IfcSlab'):
    Length, Width = slabDim(element)

    if element.PredefinedType != 'ROOF':
        x2,y2,z2 = x1,y1+Width,z1
        x3,y3,z3 = x1+Length,y1+Width,z1
        x4,y4,z4 = x1+Length,y1,z1
        idf.write('\t'+str(x2)+','+str(y2)+','+str(z2)+',\n')
        idf.write('\t'+str(x3)+','+str(y3)+','+str(z3)+',\n')
        idf.write('\t'+str(x4)+','+str(y4)+','+str(z4)+',';\n\n')
    else:
        x4,y4,z4 = x1,y1+Width,z1
        x3,y3,z3 = x1+Length,y1+Width,z1
        x2,y2,z2 = x1+Length,y1,z1
        idf.write('\t'+str(x2)+','+str(y2)+','+str(z2)+',\n')
        idf.write('\t'+str(x3)+','+str(y3)+','+str(z3)+',\n')
        idf.write('\t'+str(x4)+','+str(y4)+','+str(z4)+',';\n\n')

else:
    for props in Name:
        if props.is_a('IfcRelDefinesByProperties'):
            if props.RelatingPropertyDefinition.is_a('IfcElementQuantity'):
                prop = props.RelatingPropertyDefinition.Quantities
                for pro in prop:
                    if pro.Name == 'Length of Reference Line':
                        Length = pro.LengthValue
                    if pro.Name == 'Height':
                        Height = pro.LengthValue

WallDir = element.ObjectPlacement.RelativePlacement.RefDirection.DirectionRatios
if WallDir == (1.0,0.0,0.0):
    xpos()
elif WallDir == (-1.0,0.0,0.0):
    xneg()
elif WallDir == (0.0,1.0,0.0):
    ypos()
else:
    yneg()

idf.close()

```

D.7 FenestrationSurface

```

import ifcopenshell
import ifcopenshell.geom
from OCC.Core import TopoDS

settings = ifcopenshell.geom.settings()
settings.set(settings.USE_PYTHON_OPENCASCADE, True)

ifc_file = ifcopenshell.open('thesis.ifc')
idf = open('ifc_idf.idf','a')

```

```

BuildingElement = ifc_file.by_type('IfcBuildingElement')

for element in BuildingElement:

    #BuildingSurface or FenestrationSurface
    if element.is_a('IfcWindow') or element.is_a('IfcDoor'):
        idf.write('FenestrationSurface:Detailed,\n')
    else:
        continue

    print ('\t',element.Name,',')
    idf.write('\t'+str(element.Name)+',\n')

    #SurfaceType
    if element.is_a('IfcWindow'):
        idf.write('\tWindow,\n')
    else:
        idf.write('\tDoor,\n')

    idf.write('\t,\n') #Construction

    #Building Surface (Wall Relation)
    RelatedWall = element.FillsVoids[0].RelatingOpeningElement.VoidsElements[0].RelatingBuildingElement
    idf.write('\t'+str(RelatedWall.Name)+',\n')

    #Outside Boundary Condition Object
    space = element.ProvidesBoundaries[0]
    if space.InternalOrExternalBoundary == 'INTERNAL':
        RelSpace = element.ProvidesBoundaries[1]
        idf.write('\t'+str(RelSpace.RelatingSpace.LongName)+',\n')
    else:
        idf.write('\t,\n')

    idf.write('\t,\n') #View Factor to Ground

    idf.write('\t,\n') #Frame and Divider Name

    idf.write('\t1,\n') #Multiplier

    idf.write('\t,\n') #Number of Vertices

    #Vertices
    elem = ifccopshell.geom.create_shape(settings, element)
    shape = TopoDS.TopoDS_Iterator(elem.geometry).Value()
    transform = shape.Location().Transformation().TranslationPart().Coord()
    x1,y1,z1 = transform[0],transform[1],transform[2]

    direction = RelatedWall.ObjectPlacement.RelativePlacement.RefDirection.DirectionRatios

    if direction == (0.0,-1.0,0.0):
        x1,y1,z1 = x1,y1+(element.OverallWidth/2),z1
        x2,y2,z2 = x1, y1-element.OverallWidth, z1
        x3,y3,z3 = x1, y2, z1+element.OverallHeight
        x4,y4,z4 = x1, y1, z1+element.OverallHeight
        idf.write('\t'+str(x1)+','+str(y1)+','+str(z1)+',\n')
        idf.write('\t'+str(x2)+','+str(y2)+','+str(z2)+',\n')
        idf.write('\t'+str(x3)+','+str(y3)+','+str(z3)+',\n')
        idf.write('\t'+str(x4)+','+str(y4)+','+str(z4)+',\n\n')
    elif direction == (0.0,1.0,0.0):
        x1,y1,z1 = x1,y1-(element.OverallWidth/2),z1
        x2,y2,z2 = x1, y1+element.OverallWidth, z1
        x3,y3,z3 = x1, y2, z1+element.OverallHeight
        x4,y4,z4 = x1, y1, z1+element.OverallHeight
        idf.write('\t'+str(x1)+','+str(y1)+','+str(z1)+',\n')
        idf.write('\t'+str(x2)+','+str(y2)+','+str(z2)+',\n')
        idf.write('\t'+str(x3)+','+str(y3)+','+str(z3)+',\n')
        idf.write('\t'+str(x4)+','+str(y4)+','+str(z4)+',\n\n')
    elif direction == (1.0,0.0,0.0):
        x1,y1,z1 = x1-(element.OverallWidth/2),y1,z1
        x2,y2,z2 = x1+element.OverallWidth, y1, z1
        x3,y3,z3 = x2, y1, z1+element.OverallHeight
        x4,y4,z4 = x1, y1, z1+element.OverallHeight
        idf.write('\t'+str(x1)+','+str(y1)+','+str(z1)+',\n')
        idf.write('\t'+str(x2)+','+str(y2)+','+str(z2)+',\n')
        idf.write('\t'+str(x3)+','+str(y3)+','+str(z3)+',\n')
        idf.write('\t'+str(x4)+','+str(y4)+','+str(z4)+',\n\n')
    else:
        x1,y1,z1 = x1+(element.OverallWidth/2),y1,z1
        x2,y2,z2 = x1-element.OverallWidth, y1, z1

```

```

x3,y3,z3 = x2, y1, z1+element.OverallHeight
x4,y4,z4 = x1, y1, z1+element.OverallHeight
idf.write('\t'+str(x1)+' '+str(y1)+' '+str(z1)+'\n')
idf.write('\t'+str(x2)+' '+str(y2)+' '+str(z2)+'\n')
idf.write('\t'+str(x3)+' '+str(y3)+' '+str(z3)+'\n')
idf.write('\t'+str(x4)+' '+str(y4)+' '+str(z4)+'\n\n')

idf.close()

```

D.8 People

```

import ifcopenshell

ifc_file = ifcopenshell.open('thesis.ifc')
idf = open('ifc_idf.idf', 'a')

spaces = ifc_file.by_type('IfcSpace')

for space in spaces:
    number = space.IsDefinedBy
    for peep in number:
        if peep.is_a('IfcRelDefinesByProperties'):
            if peep.RelatingPropertyDefinition.Name == 'Pset_SpaceOccupancyRequirements':
                idf.write('People,\n')

                idf.write('\t'+str(space.LongName)+'People,\n') #Name

                idf.write('\t'+str(space.LongName)+'\n') #Zone/ZoneList Name

                people = peep.RelatingPropertyDefinition.HasProperties
                for p in people:
                    if p.Name == 'Schedule':
                        idf.write('\t'+str(space.LongName)+'Schedule,\n')
                idf.write('\tPeople,\n')
                for p in people:
                    if p.Name == 'OccupancyNumber':
                        idf.write('\t'+str(p.NominalValue.wrappedValue)+';\n')

idf.close()

```

D.9 Schedule

```

import ifcopenshell

ifc_file = ifcopenshell.open('thesis.ifc')
idf = open('ifc_idf.idf', 'a')

spaces = ifc_file.by_type('IfcSpace')

for space in spaces:
    schedule = space.IsDefinedBy

    for s in schedule:
        if s.is_a('IfcRelDefinesByProperties'):
            if s.RelatingPropertyDefinition.Name == 'Pset_SpaceOccupancyRequirements':
                value = s.RelatingPropertyDefinition.HasProperties
                for val in value:
                    if val.is_a('IfcPropertyTableValue'):

                        idf.write('Schedule:Compact,\n')
                        idf.write('\t'+str(space.LongName)+str(val.Name)+'\n')
                        idf.write('\tFraction,\n')
                        #Schedule
                        for i in range(len(val.DefiningValues)):
                            idf.write('\t'+str(val.DefiningValues[i][0])+'\n')
                            if val.DefinedValues[i][0] == '':
                                continue
                            else:
                                if i == len(val.DefiningValues) - 1:
                                    idf.write('\t'+str(val.DefinedValues[i][0])+';\n')
                                else:
                                    idf.write('\t'+str(val.DefinedValues[i][0])+';\n')

idf.close()

```