

# INTEGRATING BUILDING INFORMATION MODELING AND BUILDING PERFORMANCE SIMULATION

---

A BIM-based simulation tool generating automated building energy performance analysis with explanatory feedback at the early design stages

## Personal information

Author	Miltiadis Gkatzios
Student ID	1071544
email	miltos.gatzios@gmail.com

## Institute

University	Eindhoven University of Technology (TU/e)
Faculty	Faculty of the Built Environment
Department	Construction Management and Engineering (CME)

## Graduation committee

Chairman	prof. dr. ir. B. (Bauke) de Vries
Graduation Supervisor (TU/e)	L.A.J. (Wiet) Mazairac
Graduation Supervisor (TU/e)	dr. ir. P. (Pieter - Jan) Hoes

Date of final colloquium	: 23 August 2018
--------------------------	------------------



## CONTENTS

Preface .....	7
Summary .....	9
Samenvatting .....	11
Abstract .....	13
List of Abbreviations .....	15
List of Figures .....	15
List of Tables .....	16
List of Diagrams .....	16
List of Appendix Figures .....	16
List of Appendix Tables .....	16
<b>CHAPTER 1: INTRODUCTION .....</b>	<b>17</b>
1.1 Problem definition .....	18
1.1.1 Background .....	18
1.1.2 Building Performance Simulation .....	19
1.1.3 Integrated BPS tools with Building Information Modeling .....	19
1.1.4 Research objective and limitations .....	20
1.2 Research questions .....	20
1.3 Research model .....	21
1.4 Scientific importance .....	22
1.5 Reading guide .....	22
<b>CHAPTER 2: LITERATURE REVIEW .....</b>	<b>23</b>
2.1 Environmental impact of Architecture, Engineering & Construction industry .....	24
2.1.1 Environmental impact of AEC industry .....	24
2.1.2 Sustainable construction concepts .....	24
2.2 Building Performance Simulation .....	26
2.2.1 Introduction to Building Performance Simulation tools .....	26
2.2.2 BPS tools' limitations .....	29
2.3 Building Information Modeling .....	33
2.3.1 Introduction to BIM .....	33
2.3.2 Building Information Modeling's benefits .....	34
2.3.3 Industry Foundation Classes .....	36

2.4 Integration of BIM and BPS.....	39
2.4.1 The purpose of BIM and BPS integration.....	39
2.4.2 Examples of integrated BIM-BPS tools .....	40
2.4.3 Benefits and limitations of current BIM-BPS integrated tools.....	41
2.4.4 Available integrated BIM-BPS tools on market.....	42
2.4.5 Integrated BIM-BPS tools and decision making.....	43
2.5 Discussion on the Literature Review.....	43
<b>CHAPTER 3: DEVELOPMENT APPROACH .....</b>	<b>47</b>
3.1 Approach in integrating BIM and BPS.....	48
3.2 Tool design .....	49
3.2.1 Dutch Normative 7120.....	50
3.3 Overview of the developed tool approach .....	52
<b>CHAPTER 4: TOOL DEVELOPMENT .....</b>	<b>55</b>
4.1 Proposed BPMN process.....	56
4.2 Tool methodology .....	57
4.2.1 Creation of building characteristics .....	57
4.2.2 Calculation of the energy demand.....	64
4.3 Tool interface .....	67
4.4 Tool script.....	69
4.5 Test case.....	70
4.5.1 Test case in the developed tool .....	70
4.5.2 Test case in VABI elements .....	71
4.5.3 Accuracy comparison of the two test cases.....	72
4.5.4 Process comparison of the two test cases.....	72
4.6 Tool validation.....	74
4.7 Tool's limitations.....	75
4.8 Future improvements .....	76
<b>CHAPTER 5: CONCLUSION .....</b>	<b>77</b>
5.1 Scientific relevance .....	78
5.2 Societal relevance .....	79
5.3 Recommendations for further research .....	80
<b>CHAPTER 6: REFERENCES .....</b>	<b>81</b>

<b>APPENDIX .....</b>	<b>91</b>
APPENDIX I: Advantages and disadvantages of early design BPS tools.....	92
APPENDIX II: Process map.....	93
APPENDIX III: Exchange requirement .....	99
APPENDIX IV: Data required for the developed tool .....	103
APPENDIX V: Temperature values .....	104
APPENDIX VI: Internal heat values for each building’s functionality.....	105
APPENDIX VII: Solar radiation values .....	107
APPENDIX VIII: Summary of the python script.....	108
APPENDIX IX: Python script.....	110
APPENDIX X: Case study energy results.....	154
APPENDIX XI: Comparison of the results of the two test cases.....	155



## Preface

This report shows the research undertaken during my graduation period at the Eindhoven University of Technology (TU/e) over the past six months. After this period, I am proud to present the graduation research which concludes the Masters of Science - Construction Management and Engineering.

Inspired by the innovation encouraged in TU/e, I have selected a topic that combines one of the master's themes (Building Information Management), my aspiration in contributing to the world's sustainability and my academic background in Architecture Engineering. During the conduction of the thesis I have also gained insight into the environmental aspects of a building design.

Significant contribution of the final result belongs to my two supervisors. I would like to thank L.A.J. (Wiet) Mazairac for his recommendations towards the topic selection, his practical guidance, and the feedback given during the entire process. His motivation towards creative thinking and approaching a development has determined largely the entire research. Moreover I would like to thank P. (Pieter-Jan) Hoes for his guidance towards understanding the theory of building physics since there was no relevant background in place.

Finally I would like to thank family and friends for their practical and moral support during the last six months. I am grateful to my parents, my brother, my sister and my aunt who they have always supported me and have believed in me. My special thanks goes also to Enrico, Evgenia, Jad, Ludo, Marianiki & Niels who hold a great share in the graduation research; their contribution determined in a great degree the final outcome of this graduation research.

I hope you find the thesis interesting.

Miltiadis Gkatzios

Eindhoven, August 2018





## Summary

The environmental impact of the Architecture Engineering and Construction (AEC) industry is severe; the industry produces at least 30% of all greenhouse gas emissions while approximately half of the solid waste is generated by demolishing buildings (El-Dirably & al., 2017; Weisheng, 2017; Zhao & al., 2012). The detrimental environmental effects of the AEC industry have resulted in mobilizing the scientific community to reach sustainable development goals and solutions. However, the research subject of attention has been given on the design, on the construction, and on the operational phase. More specifically, it has been examined how the operational energy demands can be measured and eventually be reduced.

Building Performance Simulation (BPS) is a means to assess the environmental impact of the built environment. In a nutshell, energy BPS tools predict a building's energy performance (BEP) before its actual erection. Due to this prediction, construction stakeholders understand the complexity of the building's performance, and they develop sustainable solutions. However, the low data interoperability between BPS tools and the 3D design model is an essential limitation regarding the use of BPS tools. This limitation creates problems such as the increased time to (re)create the 3D model in the BPS tool and possible assumptions regarding thermal or Heating Ventilation and AirConditioning (HVAC) data. Another limitation is that the role of BPS tools is merely evaluative. In this context the simulation tools provide insight regarding the energy performance, but it cannot send any feedback in order to ensure that the users will take optimal decisions.

The limitations of BPS tools have been overcome with Building Information Modeling (BIM). BIM produces an nD geometric design, displaying semantic information (Jrade & al., 2013). This data reveals spatial, thermal and HVAC values. It makes sense, that the incorporation of these data in the BPS tools could resolve issues regarding the data interoperability, decreasing time constraints, achieving higher accuracy, and overall enhancing the BPS tools' efficiency. Moreover the involvement of BIM in the BPS practice has allowed BPS tools to send feedback to the users. For example sensitivity analyses, comparison of outcomes, and what-if scenarios represent some ways of sending feedback to the users.

The ultimate purpose of this graduation study is to encourage sustainability in the AEC industry since the earlier design stages. Sustainability can be indirectly achieved through influencing the design decision-making. To reach this target, a BPS tool should constantly monitor a 3D model, conduct automatically energy assessments and finally provide feedback to the architects guiding the design decision making. These assessments with associated feedback can be seen as input data for the BIM model. In fact, the architect can rely on them and optimize the 3D building model in terms of its energy performance.

The tool design was based on three basic building blocks. Briefly, spatial and thermal values of 3D models are exported in IFC format, they are integrated with energy norms and values through the development of a Python programming code, and finally a BEP analysis is conducted. The IFC format is considered crucial, as the main goal is to use an open source data format composed of objects with properties, while it is capable of self-extension. Furthermore, the IFC is a recognized international standard supporting use of non-vendor specific applications. Therefore, the developed tool can be compatible with each 3D vendor-based platform that can export IFC files. Spatial, thermal and HVAC values found in this format can be used as input data for the energy calculations. These calculations are taken from the Dutch Normative 7120. It contains norms and values assessing, among others, the annual heating and cooling demand of a building. In fact, the method proposed in NEN 7120, is simplified, requiring a few variables only, all of which can be retrieved directly from the IFC format. Finally, the core of the tool development is a Python script. Python is selected as the preferred language for this development, due to the fact that its syntax is easy to implement, and it offers flexibility in extracting information.

Based on this model, a new design process is proposed. The architect at the beginning of the design process connects the 3D model with the developed tool. The developed tool monitors the 3D model for design changes, and it conducts an energy assessment when the design has been modified. Therefore the tool reveals what is the environmental impact of each decision made. Finally the procedure does not interrupt the design process, is automatic, and it is activated every time the file is saved.

The developed tool, similar to other practical integrated BIM-BPS tools, can handle issues relevant to the BPS practice (e.g. data interoperability, time constraints, accuracy, efficiency). Moreover it has additional benefits derived from its automatic process. The tool does not conduct an on-demand energy analysis, instead, it operates independently. This real-time operation, allow the tool to send feedback to the architects evaluating and informing the impact of each decision made. Through constant feedback, architects can study, understand and evaluate every decision making, check and suggest alternatives solutions since the early design phases, and finally possess knowledge in the domain of building's energy performance.

## Samenvatting

The Dutch summary is a service for the reader.  
It was not written by the author. The English summary is leading.

De bouwindustrie belast in ernstige mate het milieu; deze industrie stoot minimaal 30% van alle broeikasgassen uit en ongeveer de helft van alle vaste afvalstoffen blijven over bij de sloop van gebouwen (El-Dirably & al., 2017; Weisheng, 2017; Zhao & al., 2012). De schadelijke effecten op het milieu van de bouwindustrie heeft ertoe geleid dat vanuit de wetenschap op zoek is gegaan naar duurzame doelen en oplossingen. Tot nu toe is onderzoek met name gericht geweest op het ontwerp, de constructie, en op de operationele fase. Oftewel, er is onderzocht hoe de het energiegebruik bij gebruik van een gebouw gemeten kan worden en eventueel beperkt kan worden.

Gebouw Prestatie Simulaties (Building Performance Simulation; BPS) is een meetmethode om de milieu-invloeden van een gebouwde omgeving te bepalen. Deze methode komt er in het kort op neer dat door middel van BPS de energieprestaties van een gebouw voorspeld kunnen worden voordat het daadwerkelijk is gebouwd. Dankzij deze voorspelling kunnen de betrokkenen in de bouw inzicht krijgen in de complexiteit die achter de prestaties van een gebouw zitten en zij kunnen zich richten op duurzame oplossingen. De tekortkoming is echter dat er cruciale beperkingen liggen op het gebied van data-uitwisseling tussen BPS tools en het 3D ontwerpmodel. Dit heeft bijvoorbeeld tot gevolg dat het extra tijd vraagt om de 3D modellen (opnieuw) te modelleren in BPS tools en er gewerkt wordt op basis van aannames ten aanzien van thermische of HVAC data. Een andere beperking is dat de rol van BPS nauwelijks evaluatief is. In deze context betekent het dat de simulatietool inzicht geeft in de toekomstige energieprestaties van een gebouw, maar het geeft geen inzicht of de keuzes die aan de hand van de uitkomsten gemaakt worden, ook daadwerkelijk optimaal zijn.

Bouw-Informatie-Modellering (Building Information Modeling; BIM) komt tegemoet aan beperkingen van BPS-tools. BIM produceert een nD geometrisch ontwerp dat de semantische informatie toont (Jrade & al., 2013). Deze data levert ruimtelijke, thermische en HVAC waarden op. Het is te verwachten dat het betrekken van deze data in BPS-tools tegemoet kan komen aan genoemde beperkingen zoals data-uitwisseling, tijdsinvesteringen, nauwkeurigheid en de efficiëntie van BPS-tools kan verhogen. Sterker nog, het betrekken van BIM in het gebruik van BPS heeft het mogelijk gemaakt om BPS-tools feedback naar de gebruiker te sturen over de optimalisatiekeuzes die zij maken. Voorbeelden van deze feedback zijn sensitivity analyses, vergelijkingen van uitkomsten en what-if scenario's.

Het hoofddoel van deze afstudeeropdracht is om duurzaamheid aan te moedigen in de bouwindustrie vanaf de vroege ontwerpfasen. Duurzaamheid kan indirect worden bereikt door besluitvorming in het ontwerpproces te beïnvloeden. Om dit doel te bereiken zou een BPS-tool het

3D model continue moeten monitoren, automatische energiebeoordelingen uitvoeren om ten slotte feedback te geven aan de architecten dat zij mee kunnen nemen in hun besluitproces. Deze beoordelingen met daaraan gerelateerde feedback kan worden gezien als inputgegevens voor het BIM model. De architect kan als het ware zich laten leiden door deze gegevens en het 3D gebouwwontwerp optimaliseren ten aanzien van de energieprestaties.

De tool was gebaseerd op drie basis elementen. Ruimtelijke en thermische waarden van 3D modellen worden geëxporteerd in IFC-formaat welke geïntegreerd zijn met energienormen en-waarden door de ontwikkeling van een Python code. Dit als input voor het kunnen uitvoeren van een BEP-analyse. Het IFC-formaat van groot belang omdat het doel is om open source dataformaat te gebruiken dat is gebruik maakt van objecten zodat het uitbreidbaar is. Ruimtelijke, thermische en HVAC waardes bestaan in dit formaat en kunnen gebruikt worden als inputdata voorde energieberekeningen. Deze berekeningen zijn gebaseerd op Nederlands 7120 norm. Deze bevat normen en waarden die onder andere de jaarlijkse verwarming- en koelcapaciteit bepalen. Feitelijk is de methode zoals in de NEN 7120 word voorgesteld een dusdanige vereenvoudiging dat alle benodigde waarden beschikbaar zijn in het IFC formaat. De kern van de toolontwikkeling is een Python script. Python is gekozen als een geschikte programmeertaal voor deze ontwikkeling omdat de Python syntax eenvoudig te implementeren is en een grote flexibiliteit biedt als het om het ophalen van informatie gaat.

Op basis van dit model is een nieuw ontwerpproces voorgesteld. De architect zou aan het begin van ontwerpproces zijn 3D model kunnen gaan koppelen aan de ontwikkelde tool. De ontwikkelde tool houdt het 3D model in de gaten wat betreft veranderingen in het ontwerp en voert energiebeoordelingen uit wanneer het ontwerp wordt aangepast. Zo onthult de tool de belasting op het milieu van elke keuze die wordt gemaakt in het ontwerp. De procedure onderbreekt het ontwerpproces niet. Het werkt automatisch en wordt elke keer geactiveerd op het moment dat de ontwerpfile wordt opgeslagen.

Net als andere in de praktijk geïntegreerde BIM-BPS tools, kan de ontwikkelde tool oplossingen bieden voor problemen in de BPS praktijk (e.g. bijvoorbeeld data uitwisseling, beperkingen in tijd, nauwkeurigheid en efficiëntie). Omdat het een geautomatiseerd proces is, kent het nog meer voordelen. De tool voert de analyses niet alleen uit wanneer de gebruiker daar om vraagt, maar de tool werkt onafhankelijk. Deze real-time activering laat de tool feedback genereren voor de architecten waardoor de impact van hun keuzes constant geëvalueerd wordt en zij daarover worden geïnformeerd. Door deze constante feedback kunnen architecten eenvoudiger reflecteren op hun beslissingen en de impact van hun beslissingen op het milieu beter begrijpen waardoor zij vroeg in het ontwerpproces andere aan andere oplossingen kunnen denken. Dit leidt uiteindelijk bij hen tot kennis over de energieprestaties van gebouwen.

## Abstract

The environmental impact of the Architecture Engineering and Construction (AEC) industry is severe. This graduation thesis addresses this phenomenon, and it suggests an approach to influencing architects to take sustainable design decisions optimizing a building's energy performance. The objective of this thesis was the development of an interactive tool which supports and guides the decision making during the earlier design phases. To achieve this target, the integration of Building Performance Simulation (BPS) and Building Information Modeling (BIM) was considered crucial. The developed tool was based on three basic building blocks. First, it retrieves or creates automatically necessary (spatial, thermal and HVAC) data for the energy assessment from Industry Foundation Classes (IFC) formats. Second, it applies the norms and values found in the Dutch Normative 7120, which assesses, among others, the annual heating and cooling demand of a building. Finally, the information retrieval, and energy assessment is completed with the aid of the Python programming language.

The final assessment, associated with feedback, is included in a user-friendly interface. In essence this feedback shows how each decision making has affected or will affect the energy demand. In that way the architect can evaluate the assessment and the feedback, and he can take optimal design decisions. Thus, the tool supports and guides the decision making. Finally, the entire process is automatic and it does not interrupt the design process. Therefore the tool has promoted a complete integration of BIM and BPS, and it has enabled the possibility of creating an object-oriented sustainable design.



## List of Abbreviations

3D	: Three Dimensional
AEC	: Architecture Engineering and Construction
BEP	: Building Energy Performance
BIM	: Building Information Modeling
BPMN	: Business Process Model and Notation
BPS	: Building Performance Simulation
CAD	: Computer-Aided Design
DCF	: Drawing Interchange Format
DPM	: Design Performance Modeling
EPC	: Energy Performance Coefficient
gbXML	: Green Building XML
GHG	: Greenhouse gas
HVAC	: Heating, Ventilation and Air-Conditioning
IFC	: Industry Foundation Classes
LEED	: Leadership in Energy and Environmental Design
ISSO	: Instituut voor Studie en Stimulering van Onderzoek (NL)
nD	: n-Dimensional
MEP	: Mechanical, Electrical and Plumbing
NEN	: Nederlands Normalisatie Instituut (NL)
NZEB	: Neutral Zero Energy Buildings
RDF	: Resource Description Framework
STEP	: Standard for the Exchange of Product
XML	: Extensible Markup Language

## List of Figures

Figure 1: Research model.....	21
Figure 2: Timeline of BPS tools (Aia.org, 2012; Huang, 2015) .....	27
Figure 3: History of BPS tools from 2000 to 2010 (Attia & al., 2012) .....	31
Figure 4: The role of BPS tools in 2010 (Attia & al., 2012).....	31
Figure 5: IFC Structure regarding a building's property values (BuildingSMART).....	37
Figure 6: Conversion roundtrip (Torma, 2013) .....	38
Figure 7: Development approach .....	48
Figure 8: Method overview.....	52
Figure 9: Proposed design process .....	57
Figure 10: Available and calculated data dependencies.....	58
Figure 11: Extract a wall's length with Python.....	59
Figure 12: Wall and opening relationship under IFC structure (BuildingSMART).....	63
Figure 13: Standard Layout procedure .....	68
Figure 14: Test case results.....	71

## List of Tables

Table 1: Data required for early design simulation (Hemsath, 2013).....	30
Table 2: Building blocks for the tool design.....	49
Table 3: Orientation possibilities .....	62

## List of Diagrams

Diagram 1: Wall end point .....	60
Diagram 2: End points alternatives.....	60
Diagram 3: Wall orientation.....	62
Diagram 4: Differences in Revit-Archicad in wall joints.....	75

## List of Appendix Figures

Figure 15: Energy results of case study.....	154
Figure 16: Developed tool results.....	155

## List of Appendix Tables

Table 4: Early design BPS tools .....	92
Table 5: Exchange requirements .....	100
Table 6: Available and calculated Data .....	103
Table 7: Outside temperatures and time for each month (NEN, 2012) .....	104
Table 8: Inside temperatures for each building functionality (NEN, 2012) .....	104
Table 9: Internal heat by people (NEN, 2012) .....	105
Table 10: Internal heat by equipment (NEN, 2012).....	105
Table 11: Internal heat by electricity (NEN, 2012).....	105
Table 12: Internal heat by fans (NEN, 2012).....	106
Table 13: People per square meter (Bouwbesluit, 2012) .....	106
Table 14: Incident solar radiation (NEN, 2012).....	107
Table 15: Shadow reduction factor, Form factor (NEN, 2012) .....	107
Table 16: VABI elements results .....	155



CHAPTER 1: INTRODUCTION



## 1.1 Problem definition

### 1.1.1 Background

The Architecture, Engineering, and Construction (AEC) industry is solely responsible for tailoring the built environment to society needs. However, the environmental impact of this industry is tremendous, and it has raised the awareness of the scientific community. Recent research has indicated that manufacturing and building materials have exacerbated environmental effects, such as greenhouse gas emission, noise, dust, and solid (Weisheng, 2017).

More specifically, the building industry accounts for 30% of the greenhouse gas (GHG) emissions globally per year, and it is responsible for 40% of the global energy consumption. GHG emissions are generated at the entire life-cycle, starting from the material production to the end of the building's life (Pöyry & al., 2015). Moreover, people spent most of their daily live indoors (office, home and other activities), and they heavily rely on comfort provided by mechanical heating and air-conditioning. This tendency has increased the ratio of building energy consumption to total energy consumption by 10 percentage points within the last 30 years (Cao & al, 2016). Finally, materials used for construction activities have a major impact on the environmental degradation. More specifically, building construction is responsible for 24% of the total material extractions from the lithosphere (Bribián & al., 2011).

The environmental impact of the AEC industry is not a novel insight; an extensive academic research (Adeli, 2009; Bribián & al., 2011; Cao & al, 2016; ElDiraby & al. 2017; Levitt, 2007; Pöyry & al., 2015; Weisheng, 2017; Zhao & al., 2012) has been undertaken from various sectors warning of this phenomenon. The studies have investigated the different phases of a facility's life-cycle (Kravari, 2017). However, the studies' subject of attention has been given mostly on the design, the construction and the operational phase. More specifically, it has been examined the environmental impact of building materials. Secondly it was investigated how the operational energy demands can be reduced; in the operational phase the energy consumed accounts for the largest share of a building's energy consumption (Pomponi & al, 2016).

To lessen the environmental impact, passive energy saving strategies have been proposed. These strategies focus on the building envelope, on the thermal storage, and on passive heating or cooling systems. In fact, these are strategies that an architect should consider during the earliest design stage. Therefore the policies taken so far have encouraged -in advance- the assessment and reduction of the environmental impact of buildings at their operational phase. A sustainable design, based on the operational energy consumption contributes to low operational costs, and it ultimately reduces the total environmental impact of the AEC industry.

### 1.1.2 Building Performance Simulation

An effective building performance simulation (BPS) can drastically reduce the environmental impact of the built environment. More specifically, BPS tools predict a facility's performance in terms of energy, day lighting and comfort indicators (Aia.org, 2012). In this graduation project the term "BPS tools" explicitly refers to the category of energy simulation tools. The BPS tools predict, analyze and illustrate the energy performance of a facility before this one has been literally erected. The primary goal of such tools was the simulation of the late design phase, but gradually their simulation has shifted to the entire building's life-cycle (Coackley & al., 2014). In essence BPS tools request a set of inputs and variables that insert in a calculation engine (either existed or their own) and deliver an outcome to the users (Aia.org, 2012). Therefore, the user can understand the impact of design decisions on a building's energy performance.

Most of the BPS tools are comprehensive, informative and usually have been designed for engineers or energy modelers. Energy engines such as EnergyPlus verify this model's context. Therefore, questions have been posed whether these tools could be simplified and be effective in the earlier design stages (Bazjanac & al., 2011; Hemsath, 2013; Wen & al. 2016), whether they could provide feedback to the users (Athienitis & al., 2015; Attia & al., 2012; Østergård & al., 2016), and finally whether they could meet other parties' needs such as the architects' (Attia & al., 2012). Finally, a Building Energy Performance (BEP) analysis is usually undertaken on-demand (Kumar, 2008). Particularly, BEP analyses can analyze the design outcome, but they evaluate neither the design progress nor the decision making.

### 1.1.3 Integrated BPS tools with Building Information Modeling

Building Information Modeling (BIM) has facilitated the planning, design, construction and operation of a facility in the AEC industry (Azhar & al., 2014). It produces an object-based geometry design in an nD interactive environment, displaying supplementary semantic "texture" information about the various building components (Jrade & al., 2013). Particularly, BIM has the capabilities to create "smart" objects that compose a building design.

The BIM capabilities, have allowed its incorporation in the BPS practice in order to resolve BPS issues. The examination of integrated BIM-BPS tools has shown that this incorporation has handled properly data interoperability issues among modeling software. Consequently, faster analyses can be conducted for a sustainable design (Kriegel & al., 2008), even at the earliest design stages. This particular functionality allows the verification of the energy performance of alternative design proposals. Based on this verification, users such as architects can gradually gain a deep understanding of the complexity of a building's energy performance.

#### 1.1.4 Research objective and limitations

This brief introduction has allowed formulating the research objective of this graduation study. The research objective was to examine the process and the development of a vendor-based neutral and interactive BIM tool that could conduct BEP analyses and send direct feedback to the user. The developed, for the graduation purposes, tool will enable a constant monitoring and continuous energy assessment of a BIM 3D model. Industry Foundation Classes (IFC), which is a non-proprietary standard data model for 3D building representation, could be used to connect the developed tool with any 3D design platform. Therefore, data interoperability, flexibility and use of more automated processes can be achieved.

The main limitation associated with the research objective is the absence of an established methodology in combining Building Information Modeling and Building Performance Simulation. Particularly, there is no a structure way that reveals how integrated BIM-BPS tools can be created, which is their process, or what kind of feedback they can produce. Therefore, the capabilities and the methodology of the BIM-BPS integration can be found only by evaluating and criticizing existing integrated approaches and tools.

#### 1.2 Research questions

With respect to the problem definition and the research objective, the main research question of this graduation project has been formulated as follows:

*How BIM and BPS tools can be integrated in order to provide an interactive system which guides and supports the design decision making for a sustainable building during the design process?*

The following sub-questions are related to the key components of the main question. Their examination has suggested a systematic approach to test the research question.

- i. How do current BPS systems operate? Which are their benefits and constraints?*
- ii. How can BIM be integrated into the BPS practice?*
- iii. How do current BIM-BPS tools affect the design decision making for a sustainable building during the design process?*

### 1.3 Research model

The work and delivery schedule of the graduation project has included four parts: the Literature Review, the Development Approach, the Tool Development and the Conclusion. The *Literature Review* has focused on the examination of the domains of Building Performance Simulation and Building Information Modeling, in order to cover the three sub-questions. With respect to the *Literature Review*, the *Development Approach* seeks for the method where a BIM-BPS tool can be constructed in order to apply the requirements set in the main question. The development, testing and validation of the tool have been the major elements of the *Tool Development*. Finally conclusions are made in order to verify the main question. A graphic illustration of the research model is given in Figure 1.

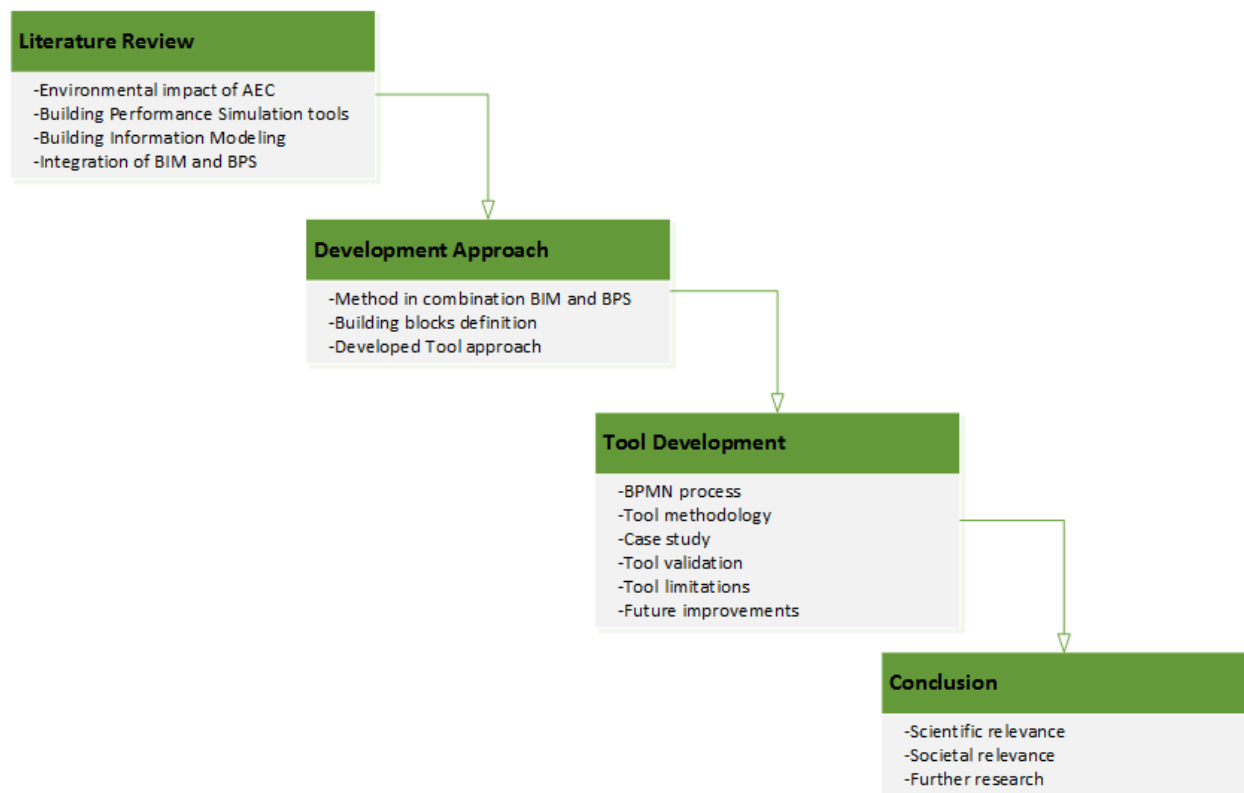


Figure 1: Research model

## 1.4 Scientific importance

The scientific importance and impact of the proposed tool is reflected in various levels. First, the developed tool assesses the building's energy performance during the design process, and it sends feedback to the architect. Therefore the design is improved in its quality, since energy assessments are taken into consideration. Moreover less time is required for its energy validation. Second, the developed tool has been built on the structure of IFC format. Thus, the tool is vendor-based neutral, which makes it compatible with each platform that exports IFC files. Finally, new contemporary theories have been applied that suggest an innovative approach towards the integration of BIM and BPS tools. This allows BIM developers to conduct further research in the BIM's contribution to BEP analyses.

As far as the societal importance is concerned, the tool allows architects to gain valuable experience in the complexity of building's energy performance. The tool predicts the environmental effect of each design decision made, and it can send direct feedback to the architect. The latter can therefore evaluate how his decisions affect the building's energy performance. Thus, it is most likely that the architect designs sustainably and he contributes to the creation of a healthy built environment, with an occupants' comfort and society's welfare (Kravari, 2017).

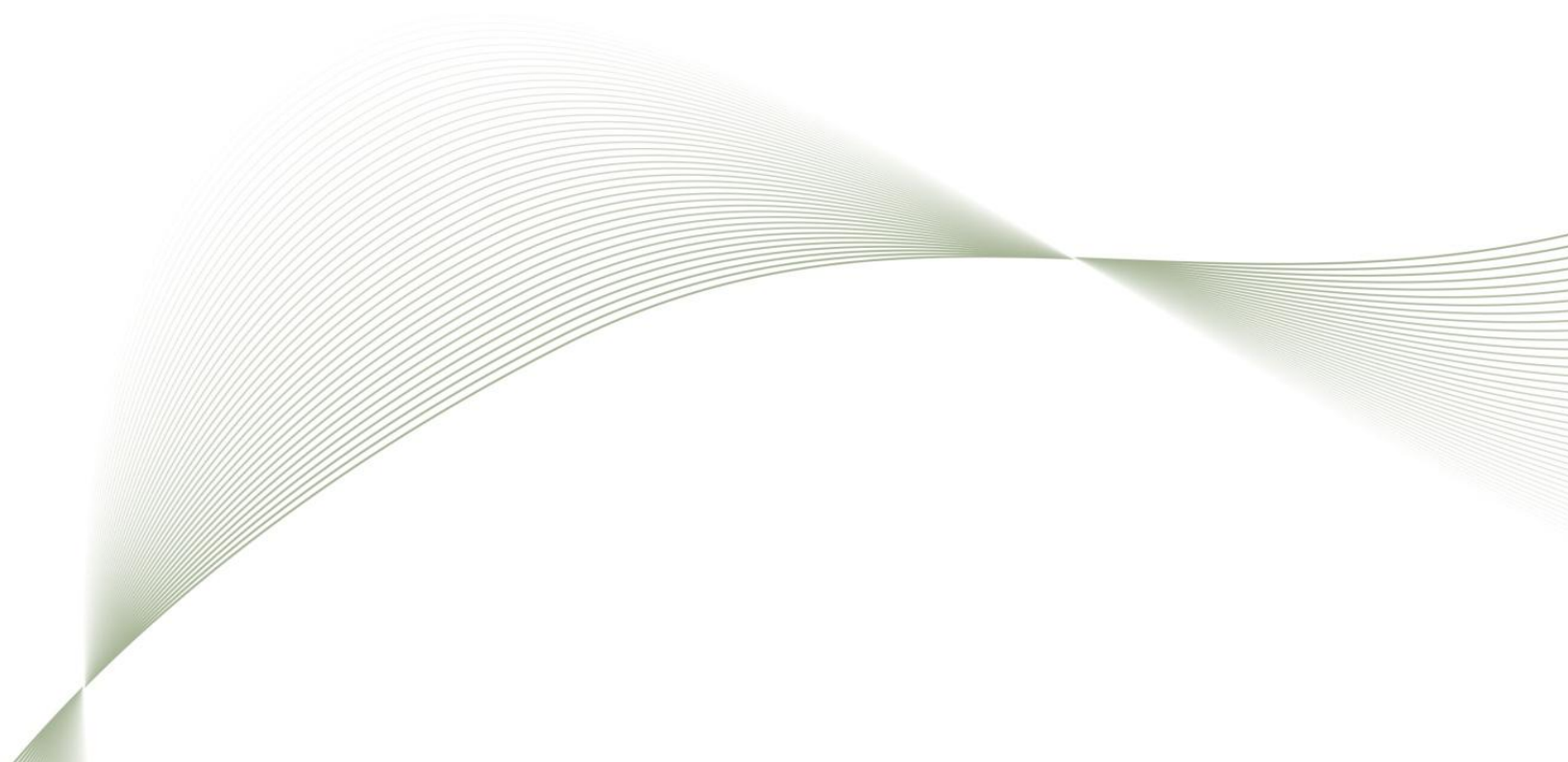
## 1.5 Reading guide

The following chapters outline the implementation of the research model. Chapter 2 covers the literature review conducted. The two themes, related to the graduation project, are introduced and explored: Building Performance Simulation (BPS) and Building Information Modeling (BIM). Furthermore the direct relationship of these two themes is further described by existing integrated BIM-BPS tools. Chapter 3 outlines in theory how to approach the development of the tool. The building blocks of the tool design have been also defined. The combination of the *Literature Review* and the *Development Approach*, has allowed the actual development of the tool, which is provided in Chapter 4. The latter contains the stages from the tool's construction, to its practical testing and its comparison with current techniques. Furthermore, the tool's limitations and its possible improvements are also mentioned. Chapter 5 includes the critical discussion on the research questions, and it stresses the scientific and societal importance of the graduation project. Finally, conclusions are drawn to further research on key aspects of this graduation project.

## CHAPTER 2: LITERATURE REVIEW

This chapter presents a summary and an evaluation of scientific and technical literature that has been related to the graduation project. State-of-art studies, white papers of vendor-based platforms, and discussions about existing 3D platforms consist primarily the literature study regarding the integration of two domains: Building Performance Simulation (BPS) and Building Information Modeling (BIM).

To conduct a comprehensive and in-depth analysis, the following chapter is organized as follows. The first section examines the environmental impact of the construction industry (2.1). The following two sections focus exclusively on the two domains of the graduation topic. Specifically, it is studied the practice of the existing BPS tools, and it is recognized its benefits and limitations (2.2). In the same context, the field of BIM is covered (2.3), and it is analyzed how it has improved the practice in the Architecture Engineering and Construction (AEC) industry. A critical review of the Industry Foundation Classes (IFC) model is also conducted. The fourth section examines the relationship between these BIM and BPS, and therefore existing integrated BIM-BPS tools are analyzed (2.4). Finally a conclusion is drawn from the literature review (2.5).





## 2.1 Environmental impact of Architecture, Engineering & Construction industry

In the specific section it is shown the impact of the Architecture Engineering and Construction (AEC) industry on the environment. Based on this impact, sustainable concepts are discussed, such as the reduction of the operational energy demands, and the embodied impact of building materials.

### 2.1.1 Environmental impact of AEC industry

Within the last years, the environmental impact of the Architecture Engineering and Construction (AEC) industry is severe and substantial. Recent research has indicated that manufacturing and building materials alone account for 10% of global energy consumption. Furthermore the operation phase produces at least 30% of all greenhouse gas emissions while approximately half of the solid waste is generated by demolishing buildings (El-Dirably & al., 2017; Weisheng, 2017; Zhao & al., 2012).

The detrimental environmental effects of the AEC industry have resulted in mobilizing the scientific community to reach sustainable development goals and solutions. In theory, studies have indicated that the AEC industry should plan and shape developments in terms of construction engineering and management. According to Levitt (2007) sustainability can be achieved by boarding the frame of integrated construction, by shaping a global construction industry and finally through new approaches, methods, and information technology. Similarly, Adeli (2009) suggests that environmental engineering technology should promote sustainability, focus both on environment and infrastructure, and advocate an intelligent approach to keep their momentum in attracting students and resources.

In practice, many studies have investigated the different phases of a facility's life-cycle, from conceptual design to complete refurbishment (Kravari, 2017). However, the studies' subject of attention has been given on the design, the construction or the operational phase. More specifically, it was investigated how the operational energy demands can be reduced, and secondly, it has been examined the environmental impact of building materials.

### 2.1.2 Sustainable construction concepts

The policies taken so far have encouraged the assessment and reduction of the environmental impact of buildings at their operational phase. The reason of this effort is that the operational phase accounts for the largest share of a building's energy consumption (Pomponi & al, 2016). More specifically, the operational heating and energy demand of buildings, leads to about 1/3



of the total energy demand globally, and produces 30% of the global CO<sub>2</sub> emissions (Koezjakov & al., 2018). Such worrying numbers have made European countries such as the Netherlands, to include in their national agenda and agreement the erection of neutral zero energy buildings (NZEBS) (Government.nl, 2017). To achieve such goals, effort has been concentrated in improving the energy efficiency of buildings.

Cao & al. (2016) mention that applying energy saving (passive) technologies is a fundamental way to improve a building's energy efficiency. Particularly, passive strategies primarily focus on sustainable building envelopes, on passive heating or cooling systems, and on the storage of the thermal energy. Building envelope is crucial for energy savings because it separates indoor and outdoor environments. According to Cao & al., improving a building envelope primarily relies on two approaches: reducing its thermal transmittances (U-values) and combining it with passive heating or cooling (Cao & al., 2016). The U-values of a building envelope significantly influence building energy consumption levels by reducing energy demands. Shan & al. (2015) have investigated technical methods in increasing energy savings by the thermal insulation. Similarly, fenestration is a vital way for comfort and cooling loads in buildings. To minimize the energy demands, one must reduce the window/wall ratio and select environmental-friendly glazing materials with low U-values. Other passive technologies include solar heat collectors, solar heat prevention and passive cooling and ventilation systems (Cao & al., 2016).

Another means of improving building energy efficiency, is to assess the environmental embodied impact of building materials. This impact, known also as embodied carbon, includes the emissions produced by the energy used concerning the processing the manufacture and the distribution of the building materials (Kravari, 2017). Even if this assessment is related to the construction stage, it can play an important role to the selection of materials during the design phase. Actually, the comparison between operational energy use and embodied energy use, allow architects to understand the relevance of the choice of construction materials. In this way they can optimize a building's energy performance and eventually contribute to the reduction of CO<sub>2</sub> emissions (Koezjakov & al., 2018).

The aforementioned methods should be a main concern of architects, who should include sustainable approaches since the design stage (Shoubi & al., 2015). An architect, who is familiar with these approaches, can make optimal sustainable decisions since the earliest design phases. In fact, reducing operational energy demands has social and economic benefits. By creating an energy efficient building, the thermal conditions within the interior space improve, and therefore users are satisfied in terms of comfort. A healthier built environment has also economic benefits. Sustainable buildings require lower fuel and electricity for the owners, reducing their housing costs (Kravari, 2017). Moreover, this reduction benefits also the

governments, since there is a low demand for new energy infrastructure (Federal Energy Management Program).

Overall, it has been exposed that a sustainable design is an aspect which reduces the operational building costs, and therefore the environmental impact of the AEC industry. The next step is a comprehensive analysis on methodologies of how to approach this aspect. Some of these methodologies are based on statistics, while others are based on simulations (Shoubi & al., 2015). Concerning simulation, existing energy tools, which can assess the environmental impact of the AEC industry, play a major role in creating a sustainable environment. And since sustainable decisions, are taken by architects at the earlier design stages, then these tools should evaluate, but also guide an architect towards implementing such sustainable decisions. An insight on the importance of such tools is given in the following sections.

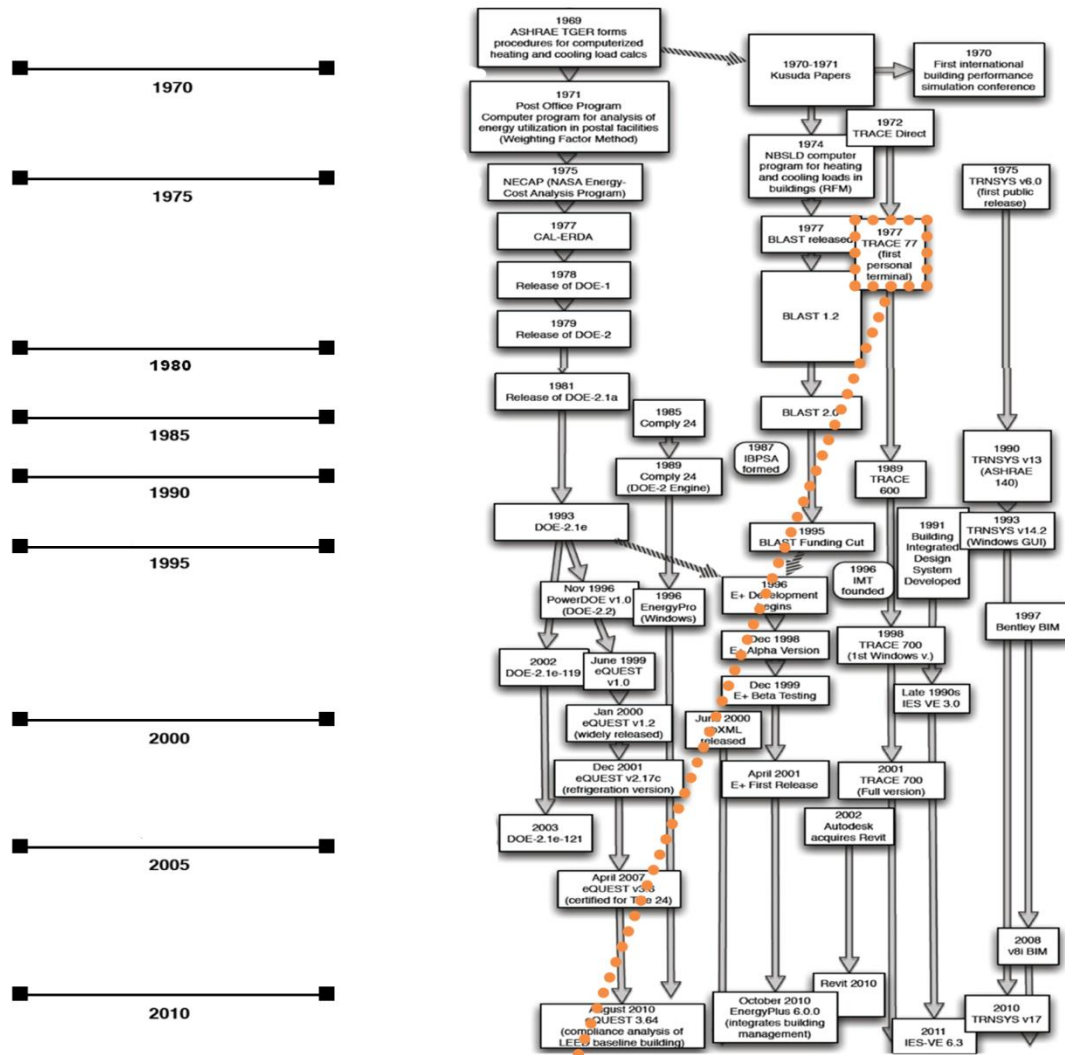
## 2.2 Building Performance Simulation

In the specific section it is explored the role of Building Performance Simulation (BPS) tools. Their history and scientific background are mentioned and their benefits are evaluated. However the section has laid emphasis on BPS tools' limitations.

### 2.2.1 Introduction to Building Performance Simulation tools

Building Performance Simulation (BPS) tools' approach has been based on computational theories; the complexity of energy performance is illustrated by the sciences of physics, mathematics, material science and so on (Hensen, 2011). Coackley & al. (2011) recognize that there are two scientific simulation models: the law-driven and data-driven. Their difference lies in the fact that law-driven models incorporate a given set laws, while, on the contrary, data-driven models predict an energy analysis by monitoring and analyzing a system behavior. Most BPS tools are law-driven and thus they define a building's energy behavior by applying building physics laws (Coackley, 2011).

BPS tools made their appearance at the 1970s (Figure 2), where they were systematically used for more than a decade, followed by a disillusion era with regard to their usage and accuracy. Nowadays, simulation systems have been evolved and BPS tools have steadily been improved on their accuracy, speed, performance and efficiency (Hensen, 2011). Recent studies (Coackley, 2011; Hemsath, 2013; Zhao & al., 2012), have indicated that the various input parameters used in BPS tools can be classified in two categories: the environment characteristics and the building components. Furthermore BPS tools have included and measured variables such as the building's occupancy and functionality (Coackley, 2011; Zhao & al., 2012).



in its simplest form, an energy model is...



a set of inputs  
and default  
variables

(Building geometry/  
massing/form, system,  
type, operation,  
schedules, etc.)



a calculation  
engine

(DOE-2, EnergyPlus  
Apache, etc.)



the results or  
output the  
program delivers

(Performance  
comparison graphs,  
compliance reports etc.)

Figure 2: Timeline of BPS tools (Aia.org, 2012; Huang, 2015)

The dynamic interaction of the building itself, its occupancy and the outside weather conditions can describe the behavior of energy performance. Two types of calculation methods can be found. First, quasi dynamic methods calculate the energy balance over a sufficiently long period of time (month or season), and second, dynamic methods provide analyses with short time periods (hourly calculations) (Gervasio & al., 2010). The selection of the method depends on the BPS tool's scope (Gervasio & al., 2010), but in each case, the tools provide the users key building performance indicators such as energy use and demand (Coackley, 2011).

Two separate categories can cover the broad area of BPS tools. On the one hand, energy simulation engines incorporate simulation principles and use appropriate energy formulas, and, on the other hand, various interfaces, which might conform to these engines. In fact, the first BPS tools have quickly reached maturity, and it was necessary to expand their system. Therefore, energy engines have offered technical solutions allowing for flexibility of simulation approaches (Crawley & al., 2000), and they have accounted for the development of new interfaces.

DOE-2 (US Department of Energy foundation) is one of these energy engines; it is a building Energy Model that assesses a building's energy performance and its life-cycle operations (Coakley & al., 2011). It provides an hourly comprehensive analysis for existing buildings, or for new-construction but conventional ones. EnergyPro and eQuest are known graphic interfaces that incorporate DOE-2 principles, and both of them assume detailed knowledge solely by engineers or energy modelers (Aia.org, 2012).

The predominant energy engine, nowadays, is EnergyPlus which has largely replaced DOE-2. EnergyPlus provides higher accuracy, fewer code-loops, it enables the analysis of complex building designs (Aia.org, 2012) and it conforms to airflows and HVAC specifications (Coakley & al., 2011). The advanced operating system of EnergyPlus has been widely adopted by researchers, who develop new interfaces and methods to measure a building's performance (Aia.org, 2012). Among EnergyPlus features, there is a time-step simulation of 15 minutes, code modification possibilities regarding input data, and report generation (Crawley & al., 2000). Sefaira is a prime example that has complied with the EnergyPlus model. In fact, Sefaira is a comprehensive interface that has first built its own engine (Aia.org, 2012) in order to support decision making. However, its credibility has been established after it has incorporated principles of energy engines such as EnergyPlus and Radiance (Sefaira, 2017). In the wide range of interfaces conforming to EnergyPlus principles, it can be also found the OpenStudio and Simergy (Aia.org, 2012).

Overall, the initial goal of the BPS tools was the simulation on the design phase, but it was realized that the BEP analysis could yield insights to the entire building lifecycle. BPS tools can observe a system's behavior given a set pattern (Coackley & al., 2011), and the output can be evaluated by different stakeholders. The latter can achieve interrelated improvements and produce a unified design, optimized in its lifecycle operations and performance (Aia.org, 2012).

### 2.2.2 BPS tools' limitations

Studies around BPS tools and their produced Building Energy Performance (BEP) analysis have raised many questions of what is simulated, at which moment during the design phase, and for whom (Athienitis, 2015; Hemsath, 2013; Bazjanac & al., 2011). Regarding these questions, Hensmath (2013) has suggested that the design stage has been divided into four different phases in which a building's energy performance can be assessed. These phases are the following: pre-design, schematic design, discrete modeling and post design validation. Most BPS tools have enabled engineers to evaluate a building's energy performance on post-design validation and discrete modeling phase; these are phases of late design stage (Hemsath, 2013). In fact, an adequate preparation for a BEP analysis should include fundamental architectural and HVAC elements in the design (Bazjanac, 2008). However, during these phases, design modifications are minor or scarce and thus a BEP analysis provides little guidance on a sustainable design (Hemsath, 2013).

Despite the abundance of BPS tools, there are missing noteworthy features that could enhance the tools' productivity and effectiveness. Hensen (2011) has highlighted a list of issues; among others issues have arisen in the early design phase support, in uncertainty and sensitivity analysis and in how-to and what-if scenarios. Such issues have been related to the absence of features that could inform the user parallel to the energy analysis evaluation.

The BPS tools' limitations can be further exposed by analyzing different types of these tools.

#### **A. Comprehensive vs. simplified**

BPS tools grasp fundamental physical principles, and they therefore require building details and environmental parameters as the input data (Zhao & al., 2012). A large amount of data can be retrieved by reliable sources but most spatial, thermal and HVAC data are stored in the design itself (Bazjanac 2008; Bazjanac & al., 2011). Therefore, an accurate energy prediction can commonly be made at the later design phases (Bazjanac, 2008), when all these data are incorporated in the 3D model. However, at this phase, vital and usually unchanged design decisions have been already been taken; such decisions significantly affect a building's energy

performance. Therefore the design could slightly be modified without yielding any remarkable results. Eventually, BEP analyses have minimal impact on the design process (Hemsath, 2013). To overcome such a limitation, a large number of simplified approaches to energy prediction have been proposed. These approaches produce a BEP analysis in the early design stage and specifically in schematic phase. A comprehensive description of this phase has been given by Hemsath (2013): “the phase that verifies conceptual design decisions such as massing, site orientation and building form”. At this phase, the architect is in constant search for design directions to make a well informed decision. In fact, architects begin with rules of thumb to create a design, and then constantly modify it to ensure its compliance with energy goals (Aia.org, 2012). However this systematic approach is not intuitive. There is a tedious work based on principles of material physics, mechanical or passive heating, cooling, air circulation, lighting, and human conveyance (Aia.org, 2012). Therefore, it is crucial the integration of BPS tools on the early design stage to support informed decision making (Attia & al., 2012).

Table 1 summarizes the findings of basic data required by early design BPS tools in six comprehensive and scientific studies (Hemsath, 2013). As can be seen in the table, most BPS tools extract input data from a building’s orientation, geometry and envelope, and from its natural ventilation. Similarly, in a current research, Wen & al. (2016), discuss the primary input data handled among ten simplified BPS tools. These data, are mainly related to building’s geometry, weather conditions and building’s properties, from materials to occupancy. For example, design decisions, such as building form, orientation, fenestration, and construction materials, influence a building energy performance (Wen & al., 2016).

Table 1: Data required for early design simulation (Hemsath, 2013)

	Warren (2002)	Hayter & al. (2000)	Xia&al. (2008)	Attia (2011)	Bambardekar & al. (2009)	Gero (1984)
Orientation	X	X	X		X	X
Massing			X	X	X	X
Function			X			
Geometry/ Shape		X	X	X	X	X
Envelope	X	X		X	X	X
Wall Ratio		X		X		X
Interior Space	X	X	X		X	
Shading		X		X	X	
Natural Ventilation	X	X	X	X	X	
Thermal Mass		X		X	X	
Daylight	X	X		X	X	
Renewable Energy	X			X	X	
Infiltration				X		

Nowadays, the amount of BPS tools is not limited to scientific researches. Instead developed vendor-based tools provide BEP analysis since the earlier design stage. Aia organization (2012) has published a list of these tools including both their major advantages and challenges. Appendix I shows this list, and as can be seen, the success of a tool has been largely depended on conditions irrelevant to the data requirements. Among others these conditions have included: their [the BPS tools'] compatibility with existing 3D vendor software (Revit, Autocad, etc), their compliance with existing energy engines (EnergyPlus and DOE-2), the structure of their visualization system, the possibility of comparing outcomes and finally their usability. Such conditions have inevitably affected the BPS tools' development.

Overall, early design BPS tools handle less input data, and they can therefore assess a building's energy performance at the earlier design stages. These BPS tools evaluate the impact of early design decisions (Wen & al., 2016), where the more energy saving potentials can be designed (Hemsath, 2013). Energy predictions can be made on the whole building or sub-level components by analyzing each influencing factor (Zhao & al., 2012).

## B. Engineering (Evaluative) / Architecture (Informative)

The extensive knowledge required to use a BPS tool, is a condition that prevent non-specialist users such as architects to involve a BEP analysis in the design process (Lin & al., 2014). Furthermore, most BPS tools available on the market are not design oriented. Attia & al. (2012) have conducted an extended review on BPS tools, and they have discovered that approximately only 10% of them are designed for architects. Figure 3 reveals the use of both engineer and architecture BPS tools from 1987 to 2010, while Figure 4 reveals the role of these tools in 2010. As can be seen, 40 BPS tools were available to architects and only 4 kept a pre-design informative role (Attia & al., 2012). These disproportionate numbers have posed further questions of the level of contribution of BPS tools in the architecture domain.

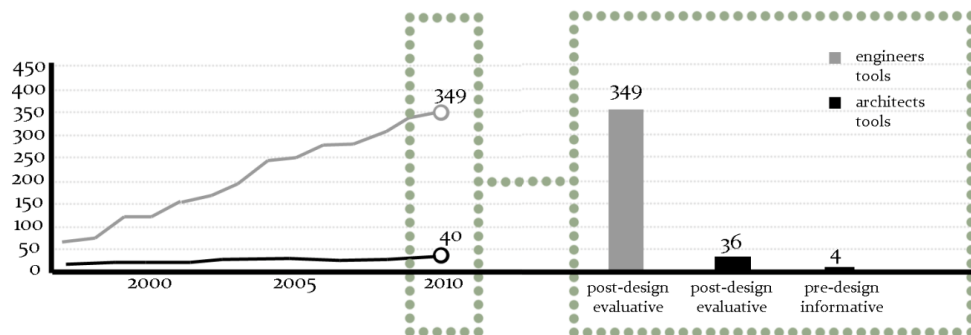


Figure 3: History of BPS tools from 2000 to 2010 (Attia & al., 2012)

Figure 4: The role of BPS tools in 2010 (Attia & al., 2012)

The minimum contribution of BPS tools in architecture can be recognized by a further research that Attia & al. (2012) have undertaken. 240 architects have evaluated key characteristics of BPS tools; particularly *intelligence* and *usability* have been preferred than *interoperability* or *accuracy* (Attia & al., 2012). A similar statement has been issued by Østergård & al. (2016) who have confirmed the necessity of BPS tools providing guidance on the architectural design. However such tools are limited and less used. The majority of BPS tools has remained evaluative than informative.

The intelligence term has been defined as: *“The software’s ability to provide this kind of active support and allow decision making on building performance and cost”* (Attia & al., 2012). This can be succeeded, if a BPS tool provides feedback explaining further the energy results. The feedback term might indicate a range of activities and possibilities. This range might vary from providing further explanation subject to the BEP analysis to implementing corrective actions.

In recent literature, it has been stressed the urgent necessity of BPS tools providing feedback to the users (Athienitis, 2015; Attia & al., 2012; Kumar, 2008). Athienitis & al. (2015) have conducted a research about the strengths and shortcomings of existing BPS tools. Based on this research they have created a list of features that they can assist the decision making. Among the significant features, the list has included parametric analysis and optimization, rapid evaluations and useful graphical feedback (Athienitis, 2015). Similarly Lin & al., (2014) have set a series of requirements, in a way to find a suitable BPS tool that could at least alleviate interoperability issues, provides analysis to inform the impact of decision making and provides data interpretation and guidance.

The example of a tool developed by Attia & al. (2012) is a prime example to illustrate the necessity of a BPS analysis associated with explanatory feedback. Actually the authors have developed a based-simulation tool that can run sensitivity analysis for design parameters and guide the decision making. However, the limitation associated with this tool is the nonexistent data interoperability with the 3D model. The building geometry can be only made in the developed tool, and thermal data are selected from a limited material library (Attia & al., 2012).

Wen & al. (2016) have also placed an emphasis on DPM (Design Performance Modeling) tools, in order to stress the necessity of feedback. These tools have provided architects with a range of BEP analysis results accordingly to design decision variations (Wen & al., 2016). For the architectural community, the goal is to gain experience with the aid of BPS tools. For example, designing in a simulation tool that provides proper feedback could allow architects to gradually possess knowledge in the domain of energy performance and therefore apply this knowledge in each project (Aia.org, 2012).



The aforementioned researches (Athienitis, 2015; Attia & al., 2012; Lin & al., 2014; Wen & al., 2016), have actually demonstrated that feedback can be effectively produced only if the simulation context is different. First there is need for interoperability among the existing 3D models and BPS tools. Currently, changes of a 3D model can be only implemented manually in a BPS tool (Lin & al., 2014). Ideally, a change in the building design would cause a reaction to the BPS tool, and the latter could produce a subsequent analysis. Second this procedure should be automatic. Therefore the analysis is not undertaken under the demand of the user, but instead is incorporated into the 3D design model. With the incorporation of these two features (data interoperability and automation) architects could optimize their own performance and design efficient buildings (Kumar, 2008).

Overall, different BPS tools assess differently the building's environmental performance, and they refer to different users. Most of them, conduct a BEP analysis at the latest design stages, and this analysis is not efficient for architects. Particularly, BPS tools cannot evaluate the energy demand at the earlier stages, and therefore they cannot assist architects to take optimal design decisions. In the same context, BPS tools do not send feedback to the users, since their role is mostly evaluative than informative. Other limitations are associated with the way a BEP analysis is conducted. First there is no data interoperability between 3D design model and energy tools. As a result, time is required to re-design the 3D model in a BPS tool. Second an energy assessment is made only by the demand of the user. Therefore, it is not an automated process, and the energy assessment interrupts actually the design process.

## 2.3 Building Information Modeling

The specific section focuses on the domain of Building Information Modeling (BIM), and specifically on its background, its definition, its benefits, and its limitations. In the same context, an insight into the Industry Foundation Classes (IFC) data model is also given.

### 2.3.1 Introduction to BIM

3D modeling has made its appearance in the 1970s, however, Building Information Modeling (BIM) has been initially developed in 2000s, as a means to visualize the built environment (Ghaffarianhoseini et al., 2017; Jrade & al., 2013). BIM has clearly marked the transition from line-based geometry to an object-based one; this change can be reflected by the different approach on contrary to the commonly used Computer-Aided Design (CAD) tools. This approach has allowed the development and access of semantic "texture" information expressing a reliable "truth", at any given moment, about a BIM facility (Jrade & al., 2013). BIM, therefore produces nD models, and it facilitates a building's development in its entire lifecycle

(planning, design, construction and operation) (Azhar & al., 2014). In this context, BIM has the potentials to enhance project efficiency meeting the needs of the Architecture, Engineering and Construction (AEC) industry. The latter has always sought ways to improve project efficiency by enhancing productivity and quality, reducing costs, and minimizing the delivery time (Azhar & al., 2008; Kravari, 2017).

The considerable complexity of BIM can be recognized by BIM's various definitions (Visschers, 2016). Briefly, there are definitions that associate BIM with the human activity (Eastman & al. 2011), alternative ones associate BIM with the efficiency among stakeholders' collaboration (Migilinskas & al 2011), and finally comprehensive definitions include the generation of data relevant to a building's lifecycle (Succar, 2009). All of these definitions cover the various benefits accrued by the established object-based geometry. It is crucial to understand these various definitions; however, the graduation research focuses on BIM data and therefore the literature review focuses on this BIM aspect.

### **2.3.2 Building Information Modeling's benefits**

A BIM model combines information relevant to the building, including its physical features, its functional characteristics, and project life cycle information, in a series of "smart objects". To illustrate a smart object, Azhar & al. (2014) provide the definition of an air-conditioning unit. This component is not described simply by its physical features and characteristics, but it also contains data of its supplier, operation and maintenance procedures, flow rates and clearance requirements (Azhar & al., 2014). This example verifies that BIM has provided a comprehensive solution to the AEC industry with numerous applications. These applications can be categorized on two domains: the graphical features and the construction sequencing with the lifecycle services (Azhar & al., 2008).

Graphical features have improved the total quality of 3D models. For example, a traditional 3D CAD format describes a building in independent and separate 2D or 3D views. Any modification of these views would not affect or update the rest. Thus, limitations and risks were associated with time efficiency but also error probabilities (Kravari, 2017). Such risks have been eliminated by the establishment of BIM which provides a complete interoperability among the 3D model viewports.

BIM has enhanced the efficiency in the data generation regarding a building's lifecycle. In the field of knowledge and information management, BIM has allowed users to share detailed information and knowledge of the building, in the design phase. Therefore, the AEC industry can manage the design, the construction, and the maintenance itself, it can control and reduce

costs, it can arrange schedules (Azhar & al., 2014; Azhar & al., 2015; Ghaffarianhoseini et al., 2017), and it can review maintenance data, energy usage, and occupancy patterns (Dankers& al., 2014). A prime example is the time reduction in producing a cost estimate. BIM applications have enabled to the user to attain automatically an accurate quantification of the building components, the materials, but also areas and volumes. The known quantity takeoff can be successfully accomplished by BIM, during the design and construction stages (Dankers& al., 2014; Grilo & al., 2011; Kravari, 2017).

BIM has supported engineers' decision making about designing an integrated project at an optimum level. More specifically, decision support can be obtained by the collective knowledge on a project and by constant information management and control (Ghaffarianhoseini et al., 2017). For example, Nowak & al., (2016), conducted a test case which demonstrates how an office-for rent BIM model has been adjusted to information regarding the profits it could generate (Nowak & al., 2016). Moreover, a BIM model includes necessary semantic information about the project itself. Based on this information, decision making can also be encouraged by "smart" design feedback and solutions so that each user optimizes his performance. For example BIM allows an engineer to design and estimate Mechanical, Electrical and Plumbing (MEP) systems in order to reduce risk and waste (G2Crowd, 2018).

Ghaffarianhoseini et al. (2017) have mentioned that BIM has almost been completely adopted, yet its potentials have not been fully exploited. The true reason includes the requirements in significant investments by AEC firms in software, hardware and training (Ghaffarianhoseini et al., 2017). Moreover, the problem of low usage is further aggravated by technical reasons. Interoperability difficulties and lack of software usability consist two of the main technical barriers (Kravari, 2017). Finally, literature review has suggested that BIM concept has not been embraced yet since there is no clear vision regarding BIM application (Dankers & al., 2014; Ghaffarianhoseini et al., 2017). There is a need to standardize the BIM process and to define the guidelines for its implementation.

Overall, BIM has defined new engineering opportunities such as extraction and generation of information (Ghaffarianhoseini et al., 2017; Azhar & al., 2014), collaboration encouragement (Jrade & al., 2013), and interoperability solutions (Beetz & al., 2015; Zhang & al., 2014). These opportunities enhance the productivity and quality of a project, reduce its life-cycle costs and minimize delivery times (Azhar & al., 2008; Kravari, 2017). Moreover BIM encourages and supports the decision made toward an optimal design both in an individual and a collaborative way. However, technical constraints and a lack of a standardized way to implement BIM are reasons that prevent the adoption of BIM (Kravari, 2017).

### 2.3.3 Industry Foundation Classes

BuildingSMART International, previously known as International Alliance for Interoperability, is a shared platform regarding the creation and adoption of BIM standards. It has actually developed and established the specification of the Industry Foundation Classes (IFC) form (BuildingSMART). IFC is an open (vendor-neutral) object-based file format which facilitates the the exchange of consistent data among various platforms used in the AEC industry (Dankers & al., 2014; Ghaffarianhoseini et al., 2017, BuildingSMART; Hitchcock & al., 2011). IFC format can be both exported and imported in most used BIM platforms; for example, in 2015, more than 160 vendors have incorporated IFC format in their systems due to its neutrality (Kravari 2017). Sharing information among vendor-based platforms is not an entirely novel approach. Neutral formats, such as Drawing Interchange Format (DXF), have been widely used for geometric information exchange (Kravari, 2017; Venugopal et al., 2012); on the contrary the IFC format covers also semantic domain content in a structured way (Dankers & al., 2014).

The complete IFC specification has been developed according to the EXPRESS data definition language, and it has adopted the STEP (Standard for the Exchange of Product model data) exchange format. Briefly, the specification allows a taxonomical classification among classes, entities, properties and attributes (BuildingSMART; Visschers, 2016). Each component is represented as a different entity, but also their relationships have been treated as separate, independent entities, which function as intermediary objects. To illustrate these relationships, Figure 5 reveals how a wall in a building project has a property value; a property value can be any spatial, thermal or HVAC datum. The IFC specification definitely is of the largest EXPRESS models since it contains approximately 800 entity definitions with their further properties and attributes (Zhang, & al., 2015). This version (IFC2x3) has been slowly replaced by IFC4 (Borrmann & al., 2015), and since 2017 the latest specifications have been released (Library of Congress, 2016; Kravari, 2017).

Nowadays, there is a lot of discussion whether IFC format could be seen as a standard in the BIM design. The IFC specification has limitations and drawbacks; the most significant lie in the data interoperability between IFC format and 3D vendor-based platforms. First, exporting an IFC format is highly flexible, according to each vendor's needs. For example, an *IfcDoor* is defined only by two mandatory attributes: *GlobalId* and *OwnerHistory*, whilst additional semantic information, is syntactically unnecessary (Zhang & al., 2015). Second, an IFC file import may include graphical errors or information lost. For example, TeklaStructures converts linear IFC objects to native TeklaStructures objects but only few of those are supported (*IfcBeam*, *IfcColumn*, *IfcPile*) (Tekla Structures, 2018). Thus, IFC schema enables a conceptual, type level compatibility between vendor-based platforms; yet, its flexibility in exporting and its limitations in importing impede this compatibility (Borrmann & al., 2015; Torma, 2013).

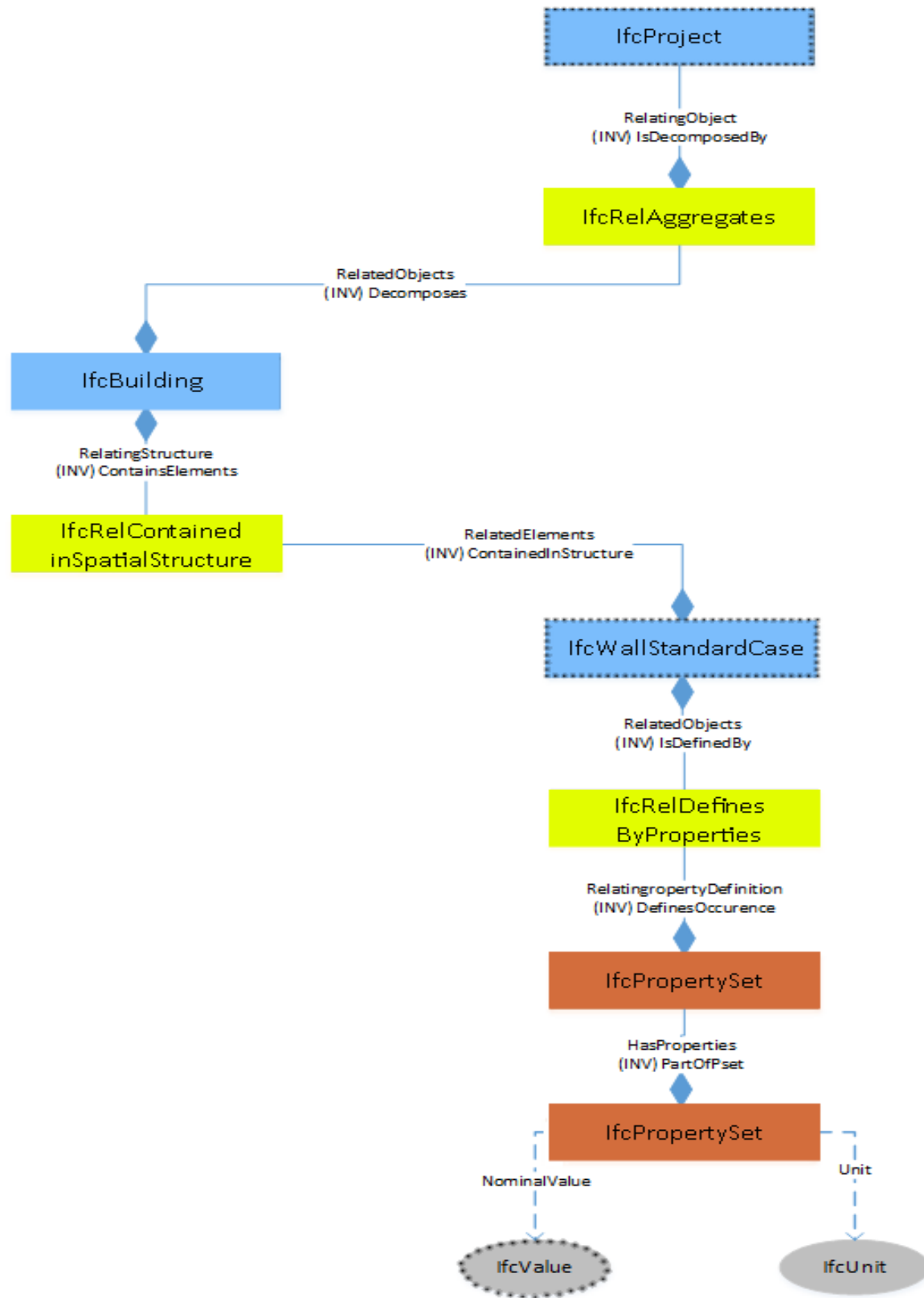


Figure 5: IFC Structure regarding a building's property values (BuildingSMART)

The low compatibility between IFC and 3D vendor-based platforms is further discussed in recent literature (Borrmann & al., 2015; Torma, 2013). Torma (2013) has introduced an approach-term roundtrip conversion as an unsuccessful design attempt (Figure 6). An IFC file can be well used in a read-mode but any design changes should be better implemented in a native file and then a new IFC file can be exported. Therefore discreet actions of load-save and export-import need to be taken (Torma, 2013) with the limitations mentioned before. McPhee, based on these necessary actions, (2013) mentions that the advantages of IFC format can be gained in fields that do not anticipate any design changes such as facilities management, coordination and analysis. As far as the analysis field is concerned, relevant data can be isolated and extracted. For example a structural analysis requires structural components whilst a thermal analysis requires the dimensional space and envelope information. The EXPRESS data definition is simpler than a vendor-based platform one and thus it is simpler to access, extract and manipulate data (McPhee, 2013).

Overall, IFC format carries, in a structured way all the data that are found in a BIM model from general building information to specific numerical values. The format is open, and it can be imported and exported in every vendor-based platform. Both its structure and its neutrality allow the BIM data interoperability in the AEC industry, however, IFC's flexibility might prevent this possibility. Therefore, the *Literature Review* has indicated that the IFC format can nowadays be perfectly used by tools that do not anticipate design changes, such as analysis tools.

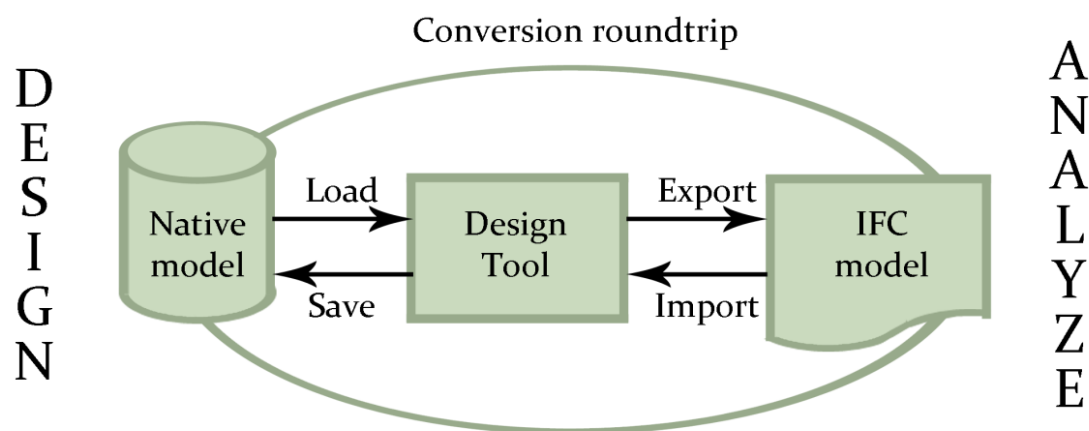


Figure 6: Conversion roundtrip (Torma, 2013)

## 2.4 Integration of BIM and BPS

The *Literature Review* has barely covered two vast fields that have affected the AEC industry the latest years. On the one hand, Building Performance Simulation (BPS) tools that produce a Building Energy Performance (BEP) analysis have caught the attention of the researchers in an urgent need to satisfy global aspirations to design, construct and occupy buildings sustainably. On the other hand Building Information Modeling (BIM) has allowed for further possibilities such as visualizing the built environment (Ghaffarianhoseini et al., 2017; Jrade & al., 2013) with semantic information before its actual creation.

This section focuses on the combination of BIM and BPS as a means to design and construct the built environment sustainably. Nowadays, this combination has been achieved mostly by independent practical tools. Therefore, this section is limited to the presentation of these tools by emphasizing both in their benefits and limitations. Finally these benefits and limitations are criticized whether they affect the design decision making and whether they have resolved other practical issues associated with the use of BPS tools.

### 2.4.1 The purpose of BIM and BPS integration

Klitgaard & al. (2006) have described the traditional collaborative approach between architects and engineers regarding the building's energy performance (BEP) evaluation. In essence, each stakeholder controls his own domain, and he uses his own toolsets. Particularly, an architect works on Building Information Modeling (BIM) design tools and an engineer on Building Performance Simulation (BPS) tools. This implies that all building performance assessment is done by the engineer, who afterwards consults the architect on the outcome and on the matter of eventual design modifications (Klitgaard & al., 2006). As can be seen, such a procedure generates risk of misinterpretations, it requires time, and finally no direct feedback options are available (Negendahl, 2015).

According to Negendahl (2015), a solution to this traditional collaborative approach would be an integrated BIM-BPS tool that could replace the communication between architects and engineers. Architects can therefore design respecting the building's energy performance since even the earliest design stages. Such an integrated tool would change the design process into a faster, performance-aware and flexible process. Questions have been posed however, what an integrated tool could do, how does it fit in the design process, and finally what can it provide to the users (Negendahl, 2015). Since there is not a standardized theory to answer these questions, a critical assessment on existing integrated tools can provide insights regarding the tools' role.

#### 2.4.2 Examples of integrated BIM-BPS tools

Scientific research has been conducted how BIM and BPS can be combined and specifically how can BIM facilitate a BPS process. Kim & al. (2011) have described a BIM-based tool as an integrator format of a building's 3D geometry, materials, building structures, Mechanical, Electrical, and Plumbing (MEP) systems and occupants. In their study, they have demonstrated how a Heating, Ventilation and Air-Conditioning (HVAC) energy performance analysis can be improved significantly using a BIM-based energy simulation tool. Compared results of the BIM simulation method and a detailed BPS tool have revealed the advantages and the disadvantages of the BIM method. The energy performance was improved in terms of time, but missing essential data regarding HVAC systems and simplified assumptions produced non-realistic results. Moreover other interoperability issues have been raised regarding the building geometry (Kim & al., 2011).

Industry Foundation Classes (IFC) has also attracted the attention of the BEP simulations. Business and financial constraints prevent the opportunity for IFC to be completely developed; yet, this format seems promising in providing significant value by exchanging data between BIM and BPS tools (Hitchcock, 2011). The influence of IFC format in BEP analyses, is relatively new, and it can be examined by five scientific examples (Andriamamonjy & al. 2018; Bazjanac, 2008; EIDiraby & al., 2017; Kravari 2017; Visschers 2016).

According to Bazjanac (2008), a methodology to a semi-automated IFC-BEP simulation preparation and execution makes the BEP process much more effective. To confirm his hypothesis, Bazjanac has used two tools. The first one allows the incorporation of HVAC and schedule data from EnergyPlus to a BIM model. In that way these data become IFC compatible. The second tool transforms an IFC file in a format that can be read by EnergyPlus (including the information of the incorporated HVAC data). Therefore spatial, thermal and HVAC data are transformed from an IFC format to a format that EnergyPlus can read. This eliminates any need for human intervention. Mistakes from vague assumptions or human errors are avoided since such a process incorporates principles of information management. Moreover, the time consumption in producing a BEP is substantially reduced; information is automatically retrieved, and the data interoperability between the applications is improved (Bazjanac, 2008).

The study of Bazjanac has revealed that the immediate connection between IFC and BPS tools has not yet been efficiently accomplished. The expansion of the IFC data and structure could possible resolve interoperability difficulties; however, it would complicate and compound management activities (EIDiraby & al., 2017; Kravari, 2017). Therefore even recent studies (Andriamamonjy & al., 2018; EIDiraby & al., 2017) have presented methodologies in which the connection between IFC and BPS tools has been established with the aid of a third tool.



ElDiraby & al. (2018) have created an algorithm that has been used to transmit IFC data in thermal zones applicable for available on market energy analysis tools. The proposed method involves a developed tool that transform an architectural, structural or a unified 3D model into a thermal analysis one. The developed tool re-creates the geometry of the 3D model so that it can be imported in OpenStudio; OpenStudio supports a BEP analysis using EnergyPlus and a daylight analysis using Radiance (OpenStudio). An almost identical approach is demonstrated by Andriamamonjy & al. (2018), who adapt an IFC format to Modelica simulation engines. Modelica's flexibility in simulating in the entire life-cycle has been the main advantage of their research study. Particularly, through an IFC format, energy analyses can be produced at early and late design stages (Andriamamonjy & al., 2018).

Finally, graduation researches (Kravari, 2017; Visschers, 2016) have provided interesting insights in the information management from IFC formats towards sustainability solutions. Visschers (2016) has developed a tool that converts an IFC file to a validated gbXML file, which can be used for a BEP simulation in DesignBuilder. A different approach can be found in the research of Kravari (2017). A tool enables the link of an IFC format with online RDF material libraries in order to find the embodied environmental impact of the construction materials. The proposed research promotes a tool that is not intermediary between existing BIM and BPS platforms; instead it is a BPS tool itself which reads only IFC data.

#### **2.4.3 Benefits and limitations of current BIM-BPS integrated tools**

All of the integrated tools, described in the previous section (2.4.2), have common benefits. The time consumption in creating the 3D model and producing a BEP analysis is substantially reduced since data are received automatically. The improved data interoperability between BIM and BPS tools, also limits the possibility of human errors. Moreover, the fast and accurate re-construction of the 3D model in energy tools allows the test of BIM design alternatives. Finally, all studies retrieve data from IFC files, therefore the methodologies proposed are not vendor oriented. Instead, they can be applied to any BIM model that can export IFC formats (Bazjanac, 2008).

The limitations of current integrated tools give an insight on the existing gaps regarding the integration of BIM and BPS. First, the immediate connection of BIM and BPS has not been accomplished yet. Instead, the developed tools are intermediary ones. They actually facilitate the transformation of IFC formats into applicable ones for existing energy engines. The tools resolve mostly interoperability issues between BIM and BPS, since they allow an accurate transmission of spatial, thermal and even HVAC data to a BPS tool. However, this method still disturbs the design process and the energy analysis is undertaken only under the demand of

the user. Therefore issues such as the automation in production of BEP analyses during the design process cannot be resolved by intermediary platforms. Ideally, a change in the building design should cause a reaction to the BPS tool.

#### 2.4.4 Available integrated BIM-BPS tools on market

Currently there are more than 160 vendor-based platforms on market, that are used on the AEC industry (Kravari, 2017); the most known platforms are Bentley AECOsim Building Designer, ArchiCAD, TeklaStructures, Autodesk Revit, SynchroPRO and VectorWorks. Most of them belong to the families of two corporations: Autodesk and NemetschekGroup in which numerous BPS plug-in operate. For example, at the moment of the research conduction, Autodesk has accommodated more than 50 plug-in relevant to BEP analysis. Therefore an analysis to all of the existing BPS tools, plug-in and applications is neither possible nor efficient. Instead an analysis of two tools is presented, which they have also shaped the development of the graduation tool. These two tools are Sefaira and PriMusIFC.

One of the most comprehensive tools nowadays is Sefaira; a tool where the domains of BPS and BIM are intimately connected. The tool retrieves automatic data for the 3D model, it creates a series of BEP analyses regarding energy, thermal comfort, daylight and HVAC design, and it also compares options about envelope design. Sefaira is fast, credible and enhances the collaboration between stakeholders (Sefaira, 2013). In fact, at the earlier design phase (after a closed building envelope) it can accurately produces an analytical BEP analysis based on the industry-standard EnergyPlus, Radiance and DAYSIM analysis engines (Sefaira, 2017). Thus, it provides the means for the entire design team to take optimal decisions and develop in collaboration a sustainable facility. To achieve an accurate result, Sefaira currently operates in Revit and SketchUp (even if it can retrieve data from IFC files), and it cannot be used in the rest vendor-based platforms. Finally, Sefaira offers free daylight and energy guidance, with the aid of hyperlinks to the web browser. The user in that way he can gain objective information on how he can reduce heating and cooling loads.

Similar to Sefaira, PriMusIFC has also proposed an alternative way in managing BIM and more specific IFC features. The specific tool read IFC files and can efficiently operate in BuildingSMART certified software such as Edificius, Revit, ArchiCad, Allplan, AutoCad and Vectorworks. It has to be mentioned that PriMusIFC analysis is irrelevant to a building's energy performance. The development of this tool, has assumed that every variation and decision making of the 3D BIM model corresponds to a seamless variation in the bill of quantities and project costs. PriMusIFC efficiently detect these changes, since it acquires information from a 3D BIM model and mechanically produces the relating QuantitiesTakeOff and cost estimations.

The tool also allows the user to maintain project's history, and it is perfectly updated with any changes made to the 3D BIM model (PriMusIFC).

In contrast to the scientific integrated tools mentioned in section 2.4.2, the available on-market tools (section 2.4.4) have managed to automate their procedure. To adapt such features, Sefaira and PriMusIFC adopt new contemporary processes and propose a developed, systematic and efficient way to connect BIM and BPS. They are not intermediary tools; instead they read by themselves BIM files. Therefore, they can actually be automatically updated as the design evolves. The user primarily designs a 3D model and his decisions inevitably produce a parallel output regarding the sustainability domain: energy consumption or cost estimation respectively. As a result, the user can evaluate comparable alternatives and finally optimize his design.

#### 2.4.5 Integrated BIM-BPS tools and decision making

One of the critical issues stressed by the *Literature Review* was the necessity of incorporating feedback in the BEP analysis (section 2.2.2). Definitely the integrated BIM-BPS tools can resolve such an issue. All of the aforementioned integrated BIM-BPS tools imply that the fast (re)construction of the 3D model in a BPS tool offers the possibility that more design alternatives can be tested. The examples of available on market tools, where 3D modeling and analysis is directly (without intermediary platforms) connected, show how these series of analyses could also allow the generation of feedback. More specifically, the integrated tools can keep a history of the changes and the respectively outcomes. In this way the user can compare the outcomes and see if the changes had a positive effect on the building's energy performance. Other ways of designing sustainably can be seen by Sefaira, which includes hyperlinks to general information of designing properly. However this kind of feedback is not connected to the BIM project itself. Finally, BPS tools could run sensitivity analyses on design parameters (Attia & al., 2008) as a means to guide the decision making. However, none of the integrated tools have included such a mechanism.

### 2.5 Discussion on the Literature Review

An analysis in the existing literature of Building Performance Simulation (BPS) tools has revealed the benefits of these tools. Energy (law-driven) engines, such as EnergyPlus have set a comprehensive framework that various interfaces use to assess a building's energy performance. A diversity of BPS tools focus on different aspects of the domain of building's energy performance, and they therefore provide a detailed analysis on the various professions existing in the AEC industry. Engineers, energy modeler and architects can gain a deep insight in

sustainable fields such as: thermal analysis, HVAC specifications, daylight analysis, and occupancy behaviors. By understanding the complexity of energy performance, they can implement sustainable solutions regarding the building's life-cycle.

Nowadays, each simulation tools, regardless its complexity (comprehensive or simplified), provides an on-demand energy efficiency analysis. That means that the user at a moment of the design phase requests for a BEP analysis according to the input data. This process is time costly, because the 3D model usually is re-created in the BPS tools. Moreover, the (re)creation of the 3D model interrupts the design process. Finally, current BEP tools do not provide guidance toward better designs; they merely provide predicted performance indicators based on the data input (Athienitis & al., 2015).

The absence of guidance makes it difficult for architects to utilize BPS tools during the design phase. Actually, an architect may not be able always to identify the reasons which influence and produce the results. Accordingly to his experience only, he can work back to the design file where he incorporates changes that could possibly improve the energy performance. Overall, this approach is tedious, costly and forces architects to rely either on their instinct or on simulation experts during the early design stages (Attia & al., 2012). In both cases, it is concluded that energy simulation tools do not aid architects in design sustainably.

Building Information Modeling (BIM) could play a major role in resolving issues concerning BPS. BIM generates models with semantic information facilitating the planning, design, construction, operation and maintenance of a facility. A BIM model can be described by 3D geometries such as materials, structure, Mechanical, Electrical, and Plumbing (MEP) systems but lately other features are involved such as occupancy and functionality. In this context, BIM could allow the rapid and effective transmission of building semantic data in the BPS tools. More specifically the IFC format facilitates the transmission of data among BIM tools due to its neutrality. IFC is used as a standard format regarding the information management, and lately its compliance with BPS tools has been extensively studied.

BIM and IFC capabilities have already improved the quality and effectiveness of BPS tools. Research studies have tested the incorporation of the BIM in the existing BPS tools. In general, an integrated BIM-BPS tool can be approached in two ways. On the one hand, an integrated tool can be an intermediary one between existing design and analysis tools. In this context, it actually facilitates the transformation of BIM formats into applicable ones for existing energy engines. On the other hand, an integrated tool can read BIM files and produce by itself an energy analysis. Therefore there is an immediate connection between the design and the analysis, that it also allows the integrated tool to be updated as the design evolves.

Integrated tools which are intermediary between BIM and BPS facilitate the automatic re-creation of a 3D model, with accurate data, in a BPS environment. Therefore issues regarding data interoperability, time reduction and 3D model (creation) accuracy have been solved. However, other limitations stressed, such as the automation in process and the generation of feedback, cannot be tackled.

The practice of available on market tools, however, has managed to automate the BPS procedure. To achieve this target, these tools have deviated from the current practice of the BPS procedure, which is applied separately from the design process. Instead, they proposed a contemporary and efficient way where the production of a BEP analysis is solely affected by the design progress. The main limitation of these tools is that they are vendor and market oriented, and they can therefore be used in specific BIM platforms. Moreover, even if they can capture the environmental impact of each decision making, they do not provide this information to the user. The limited features that can be seen as feedback are a project's history or a comparison of design alternatives.

The *Literature Review* has offered an insight in the current practice of integrating BIM and BPS. However this practice is relatively new, and the potentials and capabilities of this integration are yet to be discovered. The gaps found in the *Literature Review* have revealed that nowadays, integrated BIM-BPS tools provide limited guidance to the architects. Therefore the research objective of this study was the examination of a BPS tool that can constantly monitor a 3D model, conduct energy assessments and finally provide feedback that can guide an architect's design decisions.



### CHAPTER 3: DEVELOPMENT APPROACH

The practical integrated tools presented in the *Literature Review*, have given insights in the integration of Building Information Modeling (BIM) and Building Performance Simulation (BPS). Based on this integration, the theoretical approach for the tool creation has been established.

The following chapter is organized as follows. First an overview of the basic method in combining a BIM Model and a BPS tool is provided (3.1). Based on this method the building blocks required for the tool design are mentioned, and explained further (3.2). Finally an overview regarding the developed tool's approach is concluded (3.3).



### 3.1 Approach in integrating BIM and BPS

The ultimate purpose of this graduation study is to encourage sustainability in the construction industry since the earlier design stages. The sustainability can be indirectly achieved through influencing the design decision-making. To reach this target, a Building Performance Simulation (BPS) tool should constantly monitor a 3D model, conduct automatic energy assessments and finally include feedback mechanisms that can guide the design decision making.

The *Literature Review* has shown that these targets can be achieved by connecting Building Information Modeling (BIM) and BPS tools. This interaction involves a BIM model providing (directly or not) data to a BPS tool so that the latter can produce a Building Energy Performance (BEP) analysis. In that way, data interoperability issues are resolved, human intervention is minimized, the time of the 3D model re-creation is substantially reduced.

The proposed method to approach the development of the tool is based on this BIM-BPS interaction. Particularly, BIM data can be seen as input data for a BPS tool. However, an additional connection between BIM and a BPS tool is established. In a nutshell, the BPS tool provides a BEP analysis with associated feedback that is used as an input data for the BIM model. Based on this analysis, an optimized 3D model is produced. Therefore a causal-effect loop is created, which represents the dynamic relationship between the BPS tool and the BIM model. The approach is shown in Figure 7.

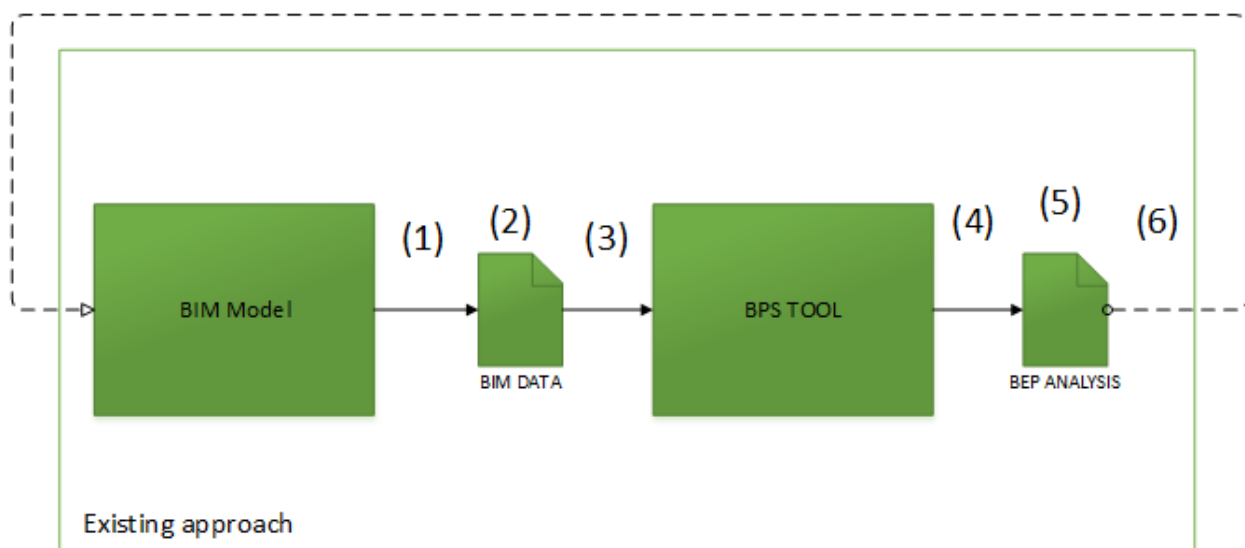


Figure 7: Development approach



As can be seen in the figure, six different points/questions are addressed. Their examination is necessary in order to develop a tool. The first three questions examine what a BIM model produces. In fact, the semantic information of a BIM model can be extracted in a different format (1), including specific information (2) and can be imported in specific applications (3). For example a vendor-based platform's file (i.e. .rvt) includes a 3D building design that can be only rendered in a 3D environment. On the other hand, an .ifc file includes semantic information written in the EXPRESS data definition language, and therefore more applications can import and read this format. Moreover, the *Literature Review* has shown that BPS tools need specific thermal analysis models, which demand the existence of intermediary tools that convert a 3D model in such a format.

The second series of questions are relevant to the BPS tool. According to its scope the tool can produce a series of analyses regarding the energy demand. Therefore the next questions reflect in the analysis that a BPS tool produces (4), and in its format (5). These questions affect immediately the last question (6), which concerns how this information can be retrieved by the user and how it affects the design decision making.

### 3.2 Tool design

The *Literature Review* has demonstrated that the integration of BIM and BPS can be achieved through two different ways. First, intermediary tools can read IFC files and transmit them into formats that can be imported in existing BPS tools (Andriamamonjy & al. 2018; Bazjanac, 2008; ElDiraby & al., 2017; Visschers 2016). Based on the new format, a BEP analysis is conducted. Second, BPS tools can read IFC formats, import them in their own system and produce a BEP analysis by themselves (Kravari, 2017; PriMusIFC; Sefaira). The second way has revealed that this procedure can be also automated and potentially send feedback to the users. Therefore such a procedure, combined with the approach described in section 3.1, implies that the developed tool should both read IFC files and produce BEP analyses. The building blocks used for the tool development in terms of resources are given in Table 2 below:

Table 2: Building blocks for the tool design

BIM design	Energy Norms	Programming Language
IFC models	NEN 7120	Python 2.7
Model 1 etc.	Bouwbesluit	Python modules
		- Python OCC 0.16
		- IfcOpenShell 2.7-0.5.0
		- PyQt4
		-Matplotlib
Version IFC2X3		

In essence, building models are exported in IFC format, which are combined with energy norms and values through the development of a code with the use of Python programming language in order to conduct the BEP analysis.

The IFC format marks a significant choice of resource, as the goal is to use an open source data format that is defined by objects with properties, while it is capable of self-extension. Furthermore, the IFC is a recognized international standard supporting use of non-vendor specific applications. The Energy Norms used is mainly the Dutch Normative 7120, which is explained in detail in the next section. The core of the tool development is a Python script. Python is selected as the preferred language for this development, due to the fact that its syntax is easy to comprehend and implement, especially where no existing programming background is in place. Furthermore, it offers flexibility in extracting and creating data.

### 3.2.1 Dutch Normative 7120

The tool development requires a valid source of norms and values regarding Energy Performance Indicators. These norms and values can be derived from the Dutch Normative 7120, which establishes the maximum allowed Energy Performance Coefficient (EPC) for buildings. The method used complies with the standards of Building Regulations and the results reflect the Dutch reality. The Dutch Normative has eventually replaced the assessment methods currently laid down in the ISSO 82 (Energy Performance Certificate Houses) and ISSO 75 (Energy Performance Certificate). The particular Normative is construction-sector oriented, and it is simplified maintaining a broad scope, legal certainty and reproducibility (NEN, 2012).

#### i. Scope

The considerable scope of the Dutch Normative includes the definition of a set of principles to describe the Dutch reality. However, construction parties may use the Normative in order to review energy topics, and achieve a balance between the expectations of clients and stakeholders, including the government. In the future, practical support and detailed guidance will be also included for the erection of new buildings (NEN, 2012). In this context, the application of the standards can be seen as an analysis tool to take fast and accurate measurements, and conduct BEP analyses.

#### ii. Structure

Two main themes can be found in the structure of the Dutch Normative. The first theme reflects on the thermal analysis and the heat transfer both by means of transmission and ventilation. It is followed by a chapter evaluating a building's internal heat (people, equipment,

lighting, and fans), whilst the third part of the thermal analysis involves the heat by solar radiation. Finally, a special chapter is relevant to the dynamic building properties and the utilization factor. The second theme involves an assessment technique to ventilation, lighting, humidification and hot water. Even if the main goal is the investigation of the EPC factor, the well structured formulas can be used for the evaluation of isolated energy cases. The early design stage can provide information relevant to the building envelope and thermal values; therefore for the developed tool, NEN7120 has enabled the computation of a building's thermal energy balance.

### **iii. Method**

Concerning the method, this is quasi-dynamic since it calculates the heat balance monthly. Independent calculation methods, for each energy aspect, involve a complex formula that it is decomposed in a series of sub-formulas with additional explanations. The values of the formulas can be characterized either as fixed or flat values. Fixed values are defined values (i.e. hours per month) or scientific accepted values modified and adjusted in the Dutch principles (i.e. weather conditions in the Netherlands). Moreover constraints and contingency errors have also been incorporated into the fixed values. On the other hand, flat values are calculated values that vary according to the building itself. Electricity types, ventilation types, and materials determine such values. In the same context, flat values are derived by a building's orientation or its construction year. Finally, the influence of the occupancy behavior affects greatly a building's energy performance. Identical buildings have different energy demands because of their occupancy. NEN 7120 has recognized such discrepancies and illustrates them with standardized values for each building's functionality.

### **iv. Limitations**

As it is clearly mentioned in the NEN 7120: *"there is a need for a legal certainty and reproducibility, still the method used is a simplified one"* (NEN, 2012). This simplification is closely aligned to the needs of the graduation topic in contrast with the abundant formulas found in the official documentation of energy engines such as EnergyPlus (EnergyPlus). However, a simplified method has placed additional limitations, which are discussed further.

NEN7120 has a small material library with standardized thermal values. Two errors arise from this limitation. First, a user might not always identify precisely the material he wants. Second, a mix of materials found in the construction work, in example brick with insulation, is associated with complicated formulas for the computation of their thermal values. BIM technology and the IFC format allow the user to overcome this limitation. In fact, a BIM component is associated with one (or more) material(s), which by default have derived thermal values (heat transfer

coefficient, thermal mass, etc); the computation of such values is already completed by the BIM model. Therefore, the developed tool relies on the NEN 7120 prime formulas, but it avoids using flat thermal values. Instead it seeks these values in the BIM model. That increases both the speed and the accuracy of the tool while it also minimized error possibilities.

The building's functionality and its influence on thermal analysis is an additional limitation in the NEN 7120. The latter, has a variety of appendixes regarding occupancy behavior; however few of them, essential for the thermal analysis balance, are missing. For example the direct connection between occupancy and people per square meter is absent. This missing information can be retrieved by other Dutch documentations such as the Bouwbesluit (Bouwbesluit, 2012).

### 3.3 Overview of the developed tool approach

The development approach aims at developing a tool which contains information and formulas about the exchange of thermal energy between the building and its surroundings. The tool can be developed upon the determination methods set by the Dutch standards for the energy performance of buildings (NEN 7120). With this information set, the tool can extract and use IFC data in order to produce a BEP analysis. The energy performance is illustrated to the architect, and it is further supported by feedback related to the current and “previous” (if exist) results. Figure 8 shows the relation among those elements. As can be seen the Dutch normative are set data while the IFC model is the input one that will eventually influence the energy results and the feedback. Conversely, the feedback provided is the output that will indirectly affect the IFC model and therefore a causal-effect loop is created.

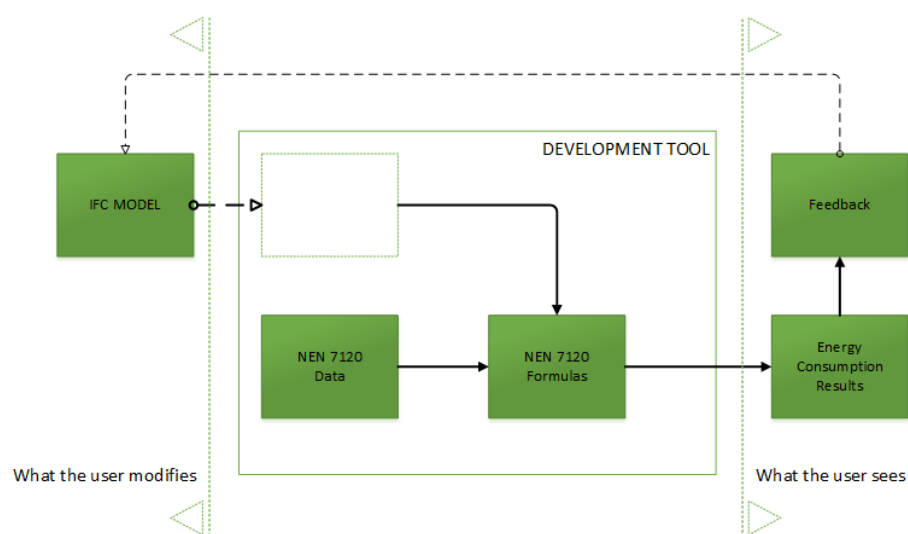


Figure 8: Method overview

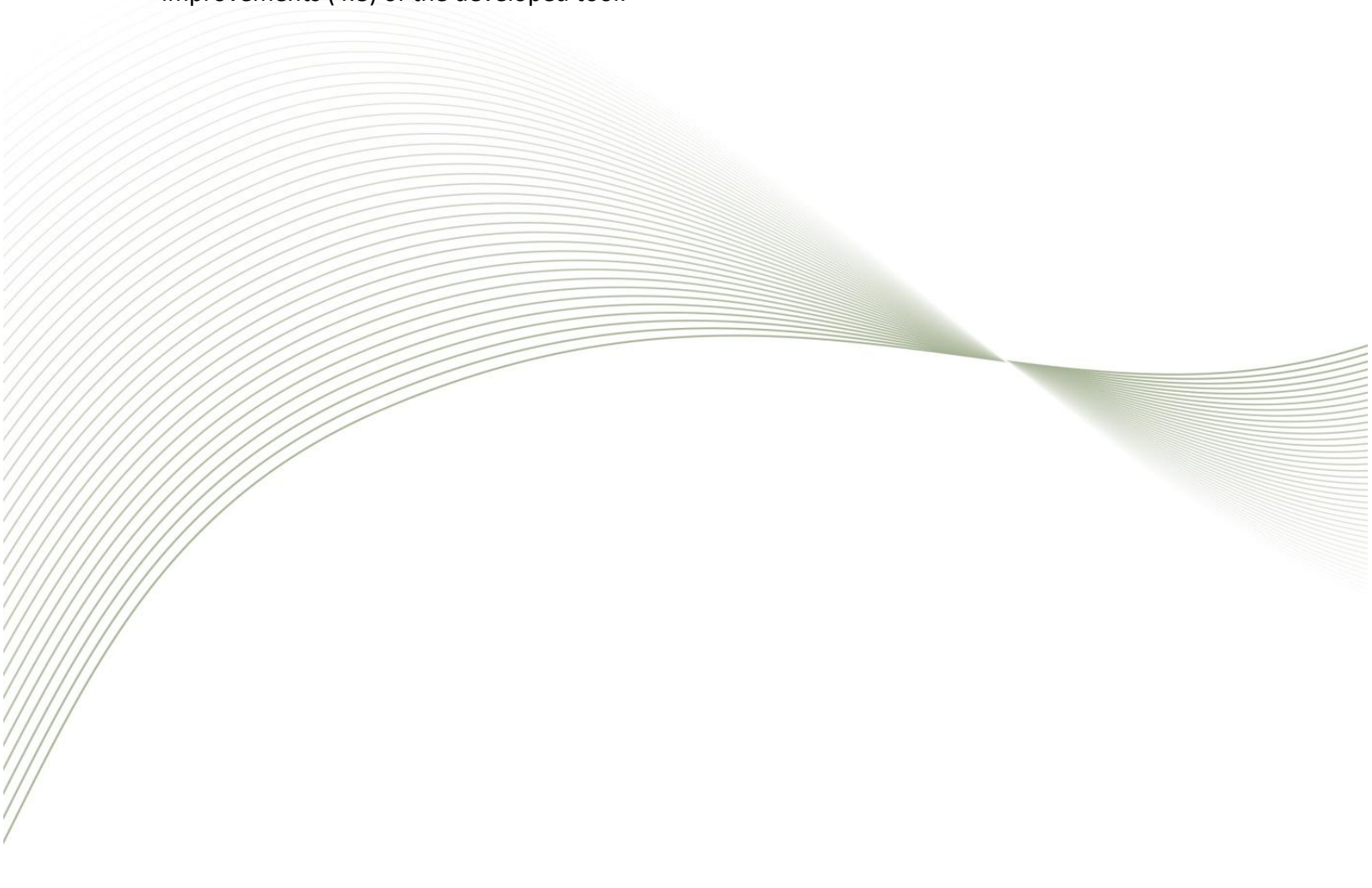
The goal of the developed tool is to actively support the decision making during the early design phase in respect of reducing a building's energy demand. Early design decisions largely determine the level of a building's energy consumption and are crucial on the sustainability features (Jrade & al., 2013). During the early and pre-construction phase, the building structure and envelope are shaped (shape, orientation, the wall ratio and/or the materials). Such design decisions are hasty and fragile and an architect may subsequently modify them with respect to creating a perfect harmony between the building and its surroundings. However, as the design evolves, the building envelope is shaped and those decisions are vital, fundamental. After this phase (discrete modeling, post design) the design can be slightly modified, without yielding any remarkable results in energy optimization. Therefore it is considered crucial that those decisions are supported and validated with the tool since the early design phase, so optimal sustainable solutions are found.



## CHAPTER 4: TOOL DEVELOPMENT

In this chapter, it is described the development of the tool. The *Development Approach* described in the previous chapter has established the framework in which the tool has been developed. In a nutshell, the tool aims in facilitating an architect to involve a Building Energy Performance (BEP) analysis in the early design phase, and it also encourages the decision-making towards sustainable solutions. The developed tool is intimately connected with 3D programs, resolving interoperability issues. Finally the tools collects data from IFC files in order to preserve its independence from the 3D vendor-based platforms.

To conduct a comprehensive analysis, the following chapter is organized as follows. First the role of the developed tool is shown in the design process (4.1). An insight in the tool's methodology follows (4.2). In this section, the actions that the tool engages in are fully described. Moreover, the interface of the developed tool and its features are illustrated and discussed (4.3). The entire methodology is also provided through a brief summary of the python script (4.4). The next section provides a test case study (4.5) that demonstrates the capabilities of the developed tool, and its benefits compared to existing BPS tools. Afterwards a tool validation (4.6) verifies whether the tool operates in each 3D vendor-based platform. Finally, the last two sections adequately reflect in the limitations (4.7), and the future possible improvements (4.8) of the developed tool.

A decorative graphic at the bottom of the page consisting of numerous thin, curved, light green lines that sweep across the width of the page, creating a sense of motion and flow.

## 4.1 Proposed BPMN process

The illustrated BPMN schema (Figure 9) describes the design process occurred during the early design phase with the involvement of the developed tool. The main objective of this schema was to capture an alternative way of the design process as contemporary plug-in (Sefaira and PriMusIFC) have suggested. The process involves one actor, the architect; the second element is the tool itself. As can be seen in the schema the architect engages in two set of actions, whilst one automatic procedure corresponds to the tool. The type and the documentation of each activity are given in the Process Map (Appendix II).

The first sequence of actions that the architect engages in is identified in the first half of the BPMN schema. These actions essentially involve the export of the 3D model as an IFC format and its import in the developed tool. The export options should include basic IFC elements so that the tool operates effectively. These IFC elements are further explained in the Exchange Requirements (Appendix III). Finally, it has to be mentioned, that this sequence is followed once; the architect should only modify the 3D model afterwards.

The import of the IFC file activates the tool, which mechanically produces a BEP analysis. Three different tasks are undertaken, which are further described in the following section (4.2). The latest task summarizes the BEP analysis; an independent window illustrates the energy results with the associated feedback.

The second series of actions that the architect accomplishes is found in the second half of the BPMN schema. In this sequence, the architect evaluates the feedback, and he modifies the design seeking sustainable solutions. The following export of the BIM model re-activates the tool's automatic procedure. Therefore, the architect can understand how the new outcome was affected by the decisions he had made between the two exports. It makes sense that often exports can even capture the individual decisions the architect takes.

The process is terminated when the architect designs the BIM model in a more detailed level. The illustrated window will slightly change and other comprehensive tools are therefore recommended.



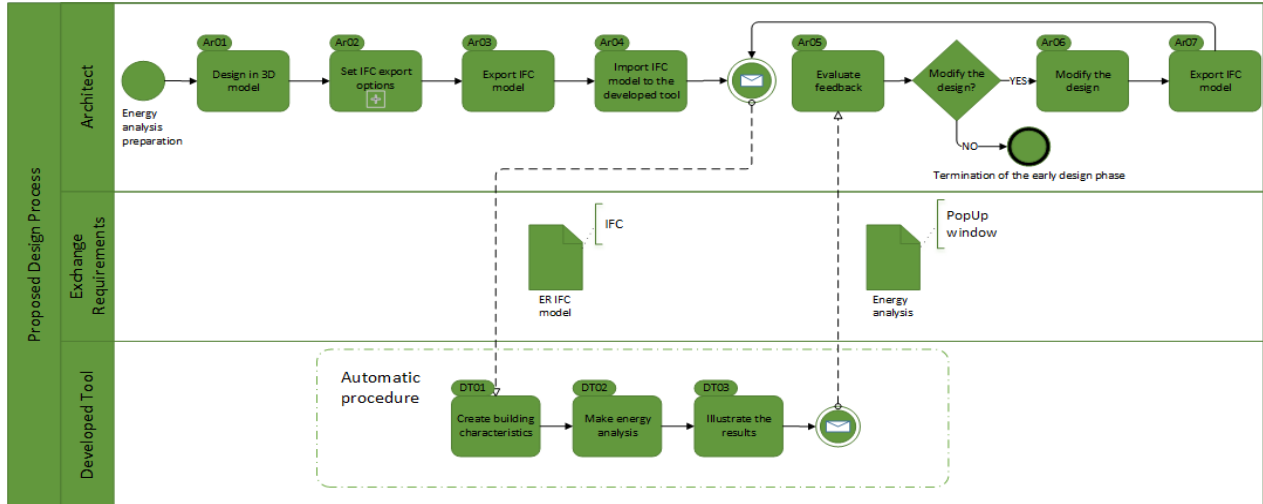


Figure 9: Proposed design process

## 4.2 Tool methodology

The automatic procedure of the tool involves a combination of three tasks and thereby provides the user a BEP analysis. These tasks are the following:

- i. Creation of the building characteristics
- ii. Calculation of the energy consumption
- iii. Illustration of the energy results

In the following sub-sections the approach of the first two tasks is further analyzed. The illustration of the energy results is discussed in the section relevant to the tool interface (4.3).

### 4.2.1 Creation of building characteristics

As it was discussed in the *Literature Review*, the IFC standard is an exchange data model, in which it is allowed the development and access of semantic texture information of a BIM facility (Jrade& al., 2013). To calculate the environmental impact of the building envelope, this information is received, manipulated and used as input in the energy calculations.

An IFC file might miss essential information that is required for the applications of the energy formulas. A wide range of information cannot be modeled, and therefore it is not established at the schema level (Zhang, & al., 2015). The required information is summarized in two fields, the available and calculated data. Appendix IV lists these data, while Figure 10 illustrates their interaction pattern. First, a brief example shows the extraction of the *Available Data*, and the problems arise from this procedure. Second, it is provided the procedure for calculating each of the *Calculated Data* with illustrated examples.

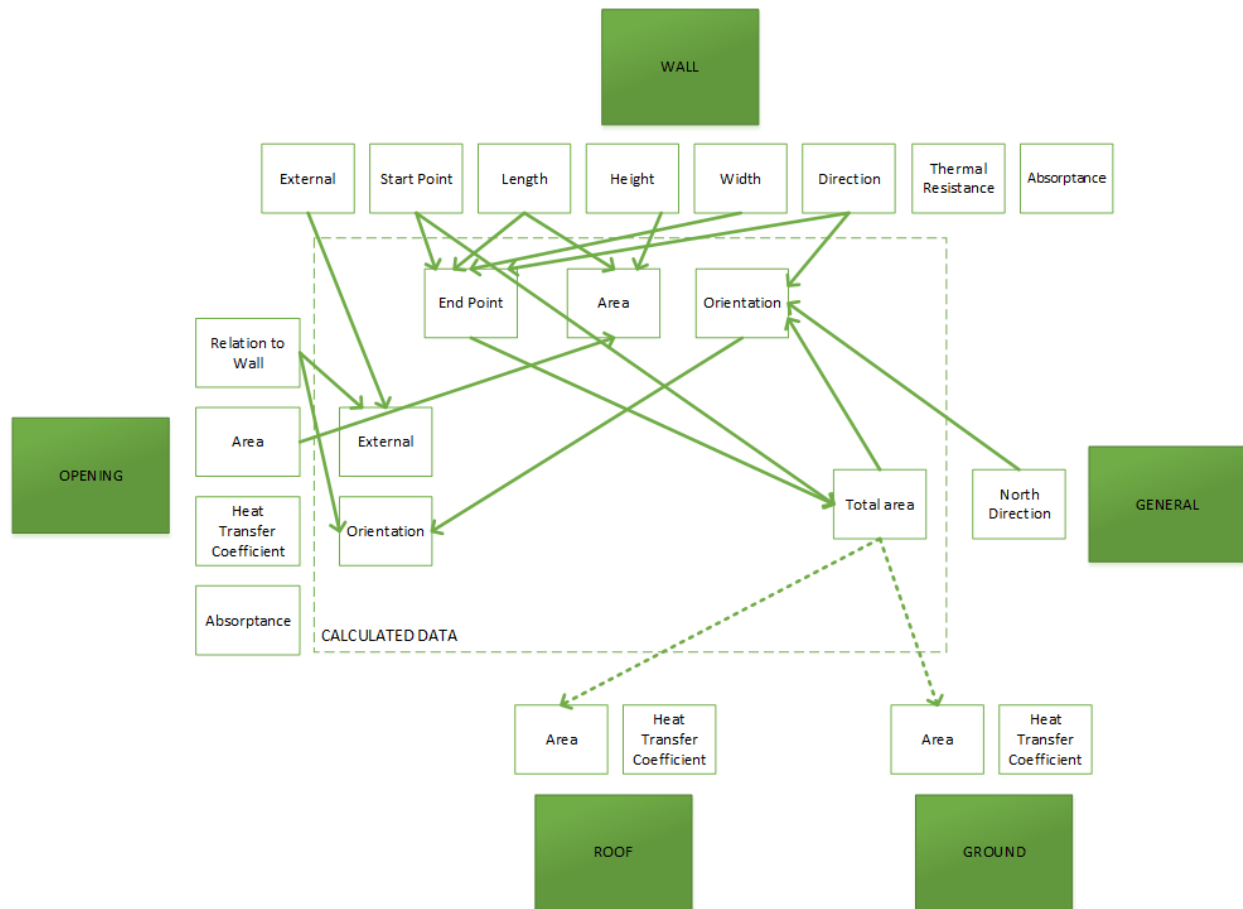


Figure 10: Available and calculated data dependencies

## A. Available Data

The code (Figure 11) contains (line: 915) and calls (line: 1661) a function: “getWallLengthWidthHeight” in order to obtain quantitative measurements such as the length of wall (line: 950). It temporarily considers the length value at 0 (line: 947), and it substitutes this value (line: 929 or line: 940) with data retrieved from the semantic component IfcWallStandardCase. This procedure involves hidden dangers, few of which were also demonstrated in *the Literature Review*.

- i. Vendor-based platforms that conduct an IFC export, allow users to customize this export. This export flexibility means that same values can be found in different entities. A considerable confusion arises over the identification of values either as properties (IfcProperty) or quantities (IfcQuantity) (Kravari, 2017). The code applies an approach where it seeks the necessary data in possible locations (line: 923 & line: 935).
- ii. Increased flexibility in the IFC export can be also provided in the properties name. The specific code was written based on the IFC format of Revit. Thus, the length value of a

wall is named as “Length” (line: 928 & line: 939). However, other vendor-based platforms might specify the same value name as “LENGTH”, or “length”. The differences spotted in capital letters, alphabet, punctuations and apostrophes could bug the code since a precise spelling is required. In the graduation research, the code limited in Revit’s needs. However, an extensive research in other platforms’ format could detect these spelling errors and enrich the code.

- iii. Another limitation that has been exposed in the *Literature Review* is the flexibility that IFC allows in defining an entity. The case of IfcDoor has been previously examined, which includes only the two mandatory attributes of GlobalId and OwnerHistory (Zhang & al., 2015). Similarly an IfcWall’s semantic information, from dimensions to material, is syntactically unnecessary. Therefore the absence of basic measurements would not allow the tool to calculate the energy performance. The code could apply alternative calculation methods; however, in the graduation project such an approach was rejected.
- iv. In the *Literature Review*, it was mentioned that IFC2x3 version is slowly replaced by IFC4 (Borrmann & al., 2015), and that could influence the IFC format’s structure. For example, as it is mentioned in the BuildingSMART documentation, the IfcElementQuantity has been sub-typed from a new intermediate IfcQuantitySet supertype (BuildingSMART). That could bug the code between line: 922 and line: 923.

---

```

1606: wallStandardCaseList = self.ifc_file.by_type("IfcWall")
1658: for wall in wallStandardCaseList:
1661:     length = getWallLengthWidthHeight(wall, "length")

915: def getWallLengthWidthHeight(wall, lengthOrWidthOrHeightOrAreaOrVolume):
916:
917:     length = 0
918:
921:     for relDefinesByProperties in wall.IsDefinedBy:
922:         if relDefinesByProperties.is_a("IfcRelDefinesByProperties"):
923:             if relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcElementQuantity"):
924:                 for properties in relDefinesByProperties.RelatingPropertyDefinition.Quantities:
925:                     if properties.is_a("IfcQuantityLength"):
926:
928:                         if properties.Name == "Length":
929:                             length = properties.LengthValue
930:
935:             elif relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcPropertySet"):
936:                 for properties in relDefinesByProperties.RelatingPropertyDefinition.HasProperties:
937:
939:                     if properties.Name == "Length":
940:                         length = properties.NominalValue.wrappedValue
941:
949:     if lengthOrWidthOrHeightOrAreaOrVolume == "length":
950:         return (length)

```

---

Figure 11: Extract a wall's length with Python

## B. Calculated Data

### i. (wall) end point:

*Requirements:* (wall) start point, (wall) axis direction, (wall) length, (wall) width

*Necessary for:* (general) total area

*Description:* An *IfcWall* has properties related to its placement in the BIM model. These properties include the *start point* and the *axis direction*, both measured by the three Cartesian points (x, y, z). However, the *end point* is not defined, and it is crucial in calculating the total external area. It has to be mentioned, that a wall's *end point* is not always the next wall's *start point*. The object placement characteristics are intimately related to the way that the wall has been designed. Diagram 2 illustrates this phenomenon. To precisely define each wall's *end point*, the tool needs to define first the wall's *start point*, *axis direction*, *length* and *width* (Equation 1, Diagram 1). Since each *end point* is defined, the tool adjusts the *axis directions* so all walls are similarly designed (either clockwise or counter-clockwise). This adjustment is required for the mathematical calculation of the total area, as it will be explained later.

*Equation - Illustrations:*

$$EP_i = SP_i * AD_i * (L_i - W_{i+1}) \quad (1)$$

EP = End Point

SP = Start Point

L = length

W = width

i = current wall

i + 1 = next wall



Diagram 1: Wall end point

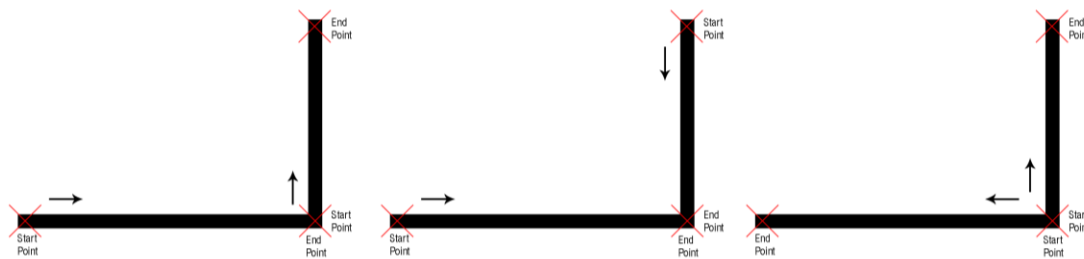


Diagram 2: End points alternatives

*Limitations:* The formula applied for the calculation of the *end point* has no limitations. However, the adjustment of the *axis directions* requires identical values between *start* and *end points*. An irregular axis direction prevents this identification because it creates many decimal digit numbers. Therefore the formula can be applied only if walls are drawn parallel to the two main axes (x, y).

## ii. (general) total area:

*Requirements:* (wall) start point, (wall) end point

*Necessary for:* possible for (roof) and (ground) area, energy calculations

*Description:* The shoelace formula (Equation 2) is a mathematical algorithm that can calculate the area of any polygon regardless its irregularity and points, as long as it is not a non-self intersecting polygon (Distributed Wikipedia, 2017). The application of the formula requires ordering the polygon's vertices either clockwise or counter-clockwise. Hence, the developed tool has already adjusted the walls' *axis direction*. The *total area* is an important generated value; it offers the advantage of performing energy calculations without requiring horizontal planes. For example, Sefaira cannot conduct any analysis if BIM components are missing, including a roof or ground floor from which the value of total area can be estimated. In the developed tool, the total area value temporarily replaces the area of these floors that are crucial elements for the application of the energy formulas. Once the architect has designed entities of horizontal planes (roof and ground), then the tool retrieves the necessary information from these entities.

*Equation - Illustration:*

$$A = \frac{1}{2} * \left( \sum_{i=1}^{n-1} (x_i * y_{i+1} + x_n * y_1) - \sum_{i=1}^{n-1} (x_{i+1} * y_i + x_1 * y_n) \right) \quad (2)$$

A = area of polygon

n = the number of walls

(x, y) = Cartesian points of start points

*Limitations:* If the points are labeled sequentially in a counterclockwise direction, then the total area will be positive. However, a clockwise direction will produce a negative value. This limitation can be overcome by requesting the area's absolute value. However, the polarity of the total area determines each wall's *orientation*.

### iii. (wall) orientation:

*Requirements:* (wall) axis direction, (general) total area, (general) north direction

*Necessary for:* (opening) orientation, energy calculations

*Description:* The *orientation* of an *IfcWall* (south, north, west and east) is absent in an IFC format even if the general *north direction* is given. However, a wall's orientation is included in energy calculations, such as the solar radiation. A wall's orientation cannot be estimated by a specific equation. Instead, there is a reasonable estimate based on three requirements: a wall's axis direction, the given north direction and the polarity of the total area.

*Illustration:* The following diagram (Diagram 3), illustrates the estimation of a wall's orientation, while Table 3 below presents the possible alternatives. As can be seen in the diagram, each axis direction corresponds to one of the four cardinal points. In Diagram 3a, the axis directions move counter-clockwise and the north direction is defined as default. Thus, when an axis direction is identified as the north direction, then the wall is west. If the axis directions move clockwise (Diagram 3b), then the same identification makes the wall east. As it has already been mentioned the rotation of the axis directions can be recognized by the total area's polarity.

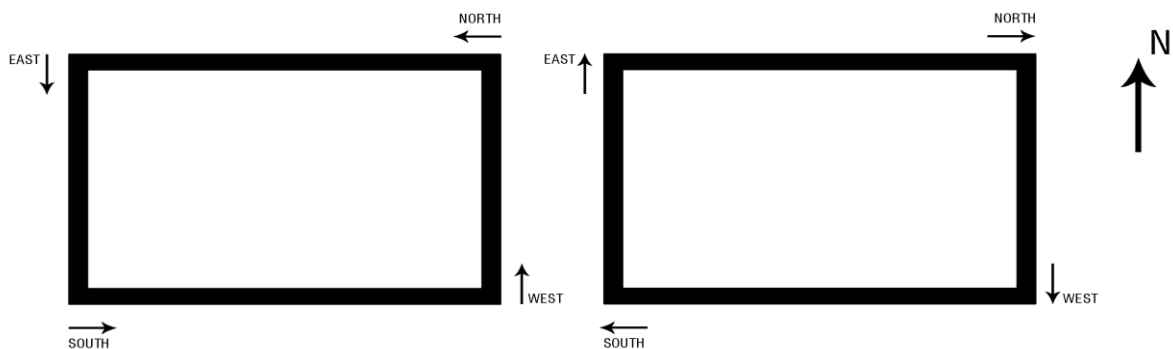


Diagram 3: Wall orientation

Table 3: Orientation possibilities

	total area's polarity positive	total area's polarity negative
Axis Direction 0° clockwise than North Direction	west	east
Axis Direction 90° clockwise than North Direction	south	north
Axis Direction 180° clockwise than North Direction	east	west
Axis Direction 270° clockwise than North Direction	north	south

*Limitations:* The estimation of a wall's *orientation* derives from a reasonable assumption based on three aspects, and the application of an equation is absent. This absence restricts the orientation values to the four cardinal points. Therefore labels as “southwest” cannot be born.

#### ix. (opening) external, orientation:

*Requirements:* (opening) relation to wall, (wall) external, (wall) orientation

*Necessary for:* energy calculations

*Description:* An *IfcOpening*, either door or window, is associated to an *IfcWall* through an intermediate entity: *IfcOpeningElement* (BuildingSMART) (Figure 12). This connection allows each opening to inherit identical orientation and position (external/internal) to its related wall.

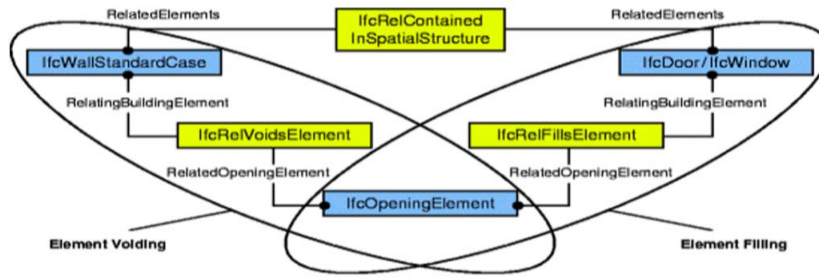


Figure 12: Wall and opening relationship under IFC structure (BuildingSMART)

#### x. (wall) area:

*Requirements:* (wall) length, (wall) height, (opening) area, (opening) relation to wall

*Necessary for:* energy calculations

*Description:* An *IfcWall* has a derived value of its area as the product of its length and height. However the specific value is not altered if an opening is attached to the wall. To define an accurate wall area, the following equation is applied (Equation 3).

*Equation:*

$$WA_i = L_i * H_i - \sum OA_i \quad (3)$$

WA = wall area

L = length

H= height

OA =opening area

i = current wall

#### 4.2.2 Calculation of the energy demand

Thermal engineering refers to heat transfer as the process of the generation, conversion and the exchange of the thermal energy between different physical systems (Designing Buildings, 2018). Four different mechanisms are identified; namely these are the thermal conduction, thermal convection, thermal radiation and the transfer of energy by phase changes. A facility's thermal behavior is a function of a dynamic relationship among those systems (Designing Buildings, 2018). Based on these mechanisms, the Dutch Normative has identified a structured way where a facility's heat energy demand derives from four different methods (NEN, 2012):

- i. Heat by transmission
- ii. Heat by ventilation
- iii. Internal Heat
- iv. Solar Radiation

Finally a utilization factor needs to be taken into account.

##### **i. Heat by transmission:**

Heat conduction is the phenomenon in which heat is dissipated because of the temperature difference between two different systems. It is defined as the fundamental and crucial process in which heat can be either gained or lost. In the built environment, heat conduction is related to materials' heat transfer coefficient (Designing Buildings, 2018). However in the complex building structures, many additional parameters affect the heat transfer such as the building's area, functionality, and usage.

The Dutch normative has identified heat by transmission as the heat conduction between the inside and outside environment. Thus it is determined by the materials that separate those two environments, or in other words by the materials of the building envelope. The overall heat by transmission is the summation of the following equation (NEN, 2012) (Equation 4):

$$Q_i = A_i * R_i * (\Theta_i - \Theta_o) * t \quad (4)$$

Q = heat transmission

A = area

R = Heat transfer coefficient

$\Theta_i$  = inside temperature

$\Theta_o$  = outside temperature

t = calculated value of the length of time in ms

i = current element



As far as the temperatures and the associated length of time are concerned, NEN 7120 provides these values for each month. Regarding the inside temperature, this depends on the building functionality. For most cases the inside temperature is fixed and set as 20°K for the winter months, while, during May to September the temperature increases to 24°K (NEN, 2012). Appendix V reveals how the outside and inside temperature is affected by each month and building functionality respectively. Finally, the spatial (area) and thermal (heat transfer coefficient) data are mechanically retrieved by the IFC model.

## **ii. Heat by ventilation**

Heat convection is the heat transfer mechanism dependent on the air movement of different temperatures. Such a movement can be forced by human activities and fans, or can be natural (Designing Buildings, 2018).

The Dutch Normative has expressed the heat by ventilation as the heat loss, associated with air flow through a building by natural means. It is mainly determined by the ventilation heat transfer coefficient which is a product of the heat capacity of air, the density of air, and the time-average temperature of supply air volume flow. The heat by ventilation is calculated as follows (NEN, 2012) (Equation 5):

$$Q = H * (\theta_i - \theta_o) * t \quad (5)$$

Q = heat transmission

H = ventilation heat transfer coefficient

$\theta_i$  = inside temperature

$\theta_o$  = outside temperature

t = calculated value of the length of time in ms

## **iii. Internal heat:**

Similarly to the heat by transmission, the internal heat represents the heat conduction between different systems occurring inside the building envelope. According to the Dutch Normative in a considered area the internal heat is calculated as follows (NEN 7120) (Equation 6):

$$Q = i_{int} * t \quad (6)$$

Q = the internal heat production

i = the sum of heat flows from the internal heat production

t = calculated value of the length of time in ms

Four different heat flows are determined (NEN 7120):

- i. Heat flow from the internal heat because of people
- ii. Heat flow from the internal heat because of equipment
- iii. Heat flow from the internal heat because of lighting
- iv. Heat flow from the internal heat because of fans

The parameter that affects each flow is the building's functionality; a parameter that it is not incorporated in the IFC format. Therefore, the tool requests the determination of this parameter before it conducts any analysis. In the developed tool, there are six (6) predefined building functionalities; namely these are: residence, office, store, school, hospital and gym. After the functionality selection, corresponding values are set. These values can be found in Appendix VI (NEN, 2012, Bouwbesluit 2012).

#### **iv. Solar radiation:**

Physical elements hotter than 0°K emit and absorb thermal radiation by their surroundings. The net difference of these exposures, reflect to heat transfer. However, this heat transfer it is almost negligible in contrast to the relative one derived from the sun (Designing Buildings, 2018). Solid surfaces in the built environment, are exposed to the sun, and therefore they have an effective collector area which absorbs an amount of the solar radiation. Similarly, transparent surfaces, such as windows, absorb an amount of this radiation, whilst another amount is transmitted immediately in the building's interior. The variables that determine the solar heat gain or loss are among others the surfaces' orientation, window blinds and heat transfer properties. The overall heat gain by solar radiation is the summation of the following formula (NEN 7120) (Equation 7):

$$Q = (F_{sr_i} * a_i * l_i - F_k * I_k t) * t \quad (7)$$

Q = the solar radiation heat

$F_{sr}$  = the dimensionless shading reduction factor of the external surface

a = the effective collector area of the surface

I = the incident solar radiation of the surface determined by its orientation and slope

$F_k$  = the dimensionless form factor between the building and the sky

$I_k$  = the additional heat flow through thermal radiation of the assembly k to the sky

t = calculated value of the length of time in ms

i = current element

All of the dimensionless factors and values regarding solar radiation, which are not found in an IFC format, are shown in Appendix VII.

#### v. Utilization factor:

To calculate the energy demand for both the heating and the cooling season, a utilization factor is necessary. Two utilization factors can be found. First, the gain utilization factor (for the internal and solar heat gains) estimates the part of the gains needed to decrease the energy needs for heating, whilst the rest part leads an undesired increase of the internal temperature. Similarly, a loss utilization factor estimates the portion of the transmission and ventilation heat transfer used to decrease the cooling needs (Gervasio & al., 2010). The calculation of the factors is a function based on the portion of the energy gain and loss, and on the time inertia of the building. The factors are used in the final energy calculations as follows (NEN, 2012) (Equation 8, 9):

For the heating period:

$$Q = Q_{HL} - (\eta_g * Q_{HG}) \quad (8)$$

For the cooling period

$$Q = Q_{HG} - (\eta_l * Q_{HL}) \quad (9)$$

Q = the energy demand (heating or cooling)

$Q_{HL}$  = the heat losses (heat by transmission + heat by ventilation)

$Q_{HG}$  = the heat gains (internal heat + solar radiation)

$\eta_g$  = the gain utilization factor

$\eta_l$  = the loss utilization factor

### 4.3 Tool interface

The following chapter describes the function of the interface system which is involved in the tool. Two different layouts can be found, namely the standard and the results layout.

#### A. Standard Layout:

The standard layout is the basic layout that appears when an architect uses the developed tool. The simple layout consists of a graphical presentation of the IFC model, and various options included in corresponding buttons. In fact, the architect is asked to accomplish three discreet actions. A successful attempt results in the access of the necessary information, so that the tool produces a BEP analysis. Figure 13 illustrates the layout process and its incorporated features.

First, the architects load an IFC file (Figure 13a). Thus, he connects the developed tool with the 3D design. In fact, the developed tool memorizes the file's path and it constantly look for changes. Therefore, it is crucial that each export overwrites the given IFC file and its path. Second, the data IFC file activates further options (Figure 13b). The architect can visually verify whether the 3D design is the correct one. Further actions such as zoom and rotate are also possible. The architect should also select the type of the facility among the six (6) pre-defined building's functionalities. Third, the import of the IFC file and the selection of the building functionality enable the tool to retrieve all the necessary spatial, thermal and factors data. Therefore a BEP analysis can be conducted. Such an option activates the second layout, regarding the energy results (Figure 13c).

The implementation of these three actions is accomplished once. Afterwards the architect updates the IFC file (export new versions), and hence the tool retrieves the new data, it produces the new results, it captures the differences, and it visualizes the new analysis with the associative feedback. The architect re-takes these actions only in order to upload a new 3D model, or to select a different building functionality.

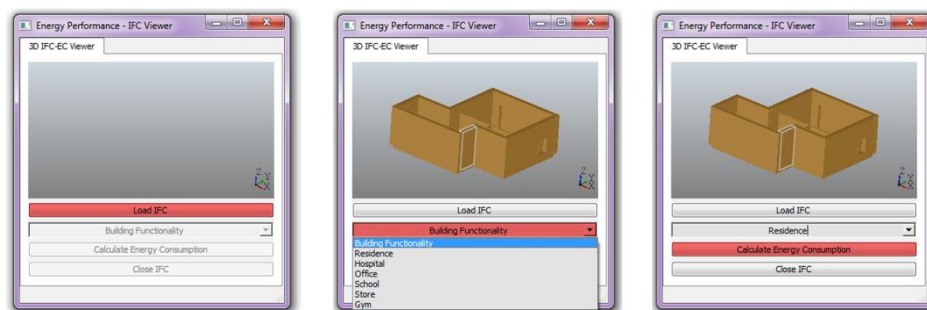


Figure 13: Standard Layout procedure

## B. Results Layout

The results layout indicates the building's energy demand, found in the IFC model, and it is associated with explanatory feedback. Similarly to the standard layout, the results layout includes graphical representations, blocks of texts and further selection options. The layout can be seen in section 4.5, Figure 14. The results layout is divided in four different sections:

- i. **Energy results button:** By pressing the button, the tool provides a graphic and table illustration of the final outcome. BPS tools enable engineers and energy modelers to assess a building's energy performance by such features. However, the *Literature Review* has indicated that such a visualization system is not efficient to determine architects' design choices (Attia & al., 2012). Thus this function is hidden, and is illustrated only under request.
- ii. **Feedback box:** The specific section shows how each element contributes different to the building's energy demand. The tool allows the architect to assess the impact of each BIM elements by numerical, positive or negative, values. This quantitative assessment is the first feedback provided to the architect.
- iii. **Illustration box:** The final outcome is recorded in a diagrammatic architectural building's section. The four thermal methods are incorporated in the diagram (heat by transmission & ventilation, internal heat and solar radiation). The overall heat requirement is also included in the same diagram in terms of heating and cooling demand. The second feedback that an architect receives can be found in *the illustration box*. Once the architect modifies the BIM model, and exports a new IFC file, the diagram reveals both the new and old outcome. Their comparison is drawn and illustrated in contrasting colors (red & green), reflecting the impact of the decision making. Finally the individual illustrations of the thermal methods allow the architect to understand how the interior environment, the exterior environment, and the building's envelope affect the heat balance.
- iv. **What-if button and what-if box:** A series of possible design actions is incorporated in the tool. The architect can examine these possibilities by pressing the *What-If button*. A selection of an action conducts a theoretical BEP analysis. Therefore the architect can investigate the impact of design decisions before he actually implements them. The outcome is temporarily available in the text block, allowing the architect to gain a better understanding of how the heat balance can be optimized. The *What-if* feature is the third interactive feedback provided to the architect.

#### 4.4 Tool script

The last section describing the methodology of the tool is an examination of its script. Python has been used as the programming language since it offers flexibility and it can easily collect data from an IFC model. A summary of the BEP analysis procedure can be found in Appendix VIII, while the entire written script is given in Appendix IX.

## 4.5 Test case

To verify the tool, a test case was conducted. The test case describes the creation of a 3D model in Revit, and its connection with the developed tool in order to guide the design decision making. Furthermore, the similar test case has been conducted in VABI Elements, a BPS tool that uses the same formulas indicated in NEN 7120. The similar test case was conducted for two reasons. First, the comparison of the outcomes of the two cases can show if the developed tool is accurate and its results are reasonable. Second, and most importantly, a comparison can be made between the processes of the two tools (developed tool - VABI Elements). This comparison reveals the benefits of the process of the developed tool, and it shows how these benefits can optimize the design procedure.

### 4.5.1 Test case in the developed tool

In Revit 2017, a simple building model was built (10 m. x 12 m. x 3 m.), with a door in the west (2.20 m. \* 0.90m.) ( $U = 3.70 \text{ W}/(\text{m}^2 \cdot \text{K})$ ), a window in the west (1.20 m. 0.90 m.) ( $U = 3.68 \text{ W}/(\text{m}^2 \cdot \text{K})$ ), three windows in the north (1.20 m. 0.90 m.) ( $U = 3.68 \text{ W}/(\text{m}^2 \cdot \text{K})$ ), and one window in the east (1.20 m. 0.90m.) ( $U = 3.68 \text{ W}/(\text{m}^2 \cdot \text{K})$ ). The material selected for the walls was the pre-defined by Revit “Exterior Wall – Brick on CMU” ( $U = 0.18 \text{ W}/(\text{m}^2 \cdot \text{K})$ ). The file was exported as an IFC file and imported in the developed tool. The building functionality was selected as *Office* and the results have been provided immediately. To get more detailed results, and to check if the tool works, the energy results were requested. The results, regarding energy and cooling demand per month, can be shown in Appendix X.

Figure 14a shows the results that the developed tool produced. As can be seen, the outcome in the *Illustration Box* suggests that the energy and cooling demand of the building are approximately 24.000 MJ and 5.000 MJ respectively. With respect to the final outcome, the *Feedback Box* shows how each of the building (BIM) components has affected the final outcome. As can be seen, each square meter of a north opening has increased both the (heating and cooling demand) approximately by 1000MJ. Since the design is still in its earlier phases, and the design decisions are not vital, the architect can realize that the north openings increase greatly the demand. Therefore, their position can be limited, and instead openings should be placed to other walls. In this context, two of the three north openings (almost two square meters) have been removed and the file has been re-exported. The results should reveal a total energy demand reduced by 2000MJ (2 x 1000MJ). Indeed, the immediate outcome verified this assumption. The heating demand was reduced approximately by 1600MJ and the cooling demand by 400MJ (Figure 14b).

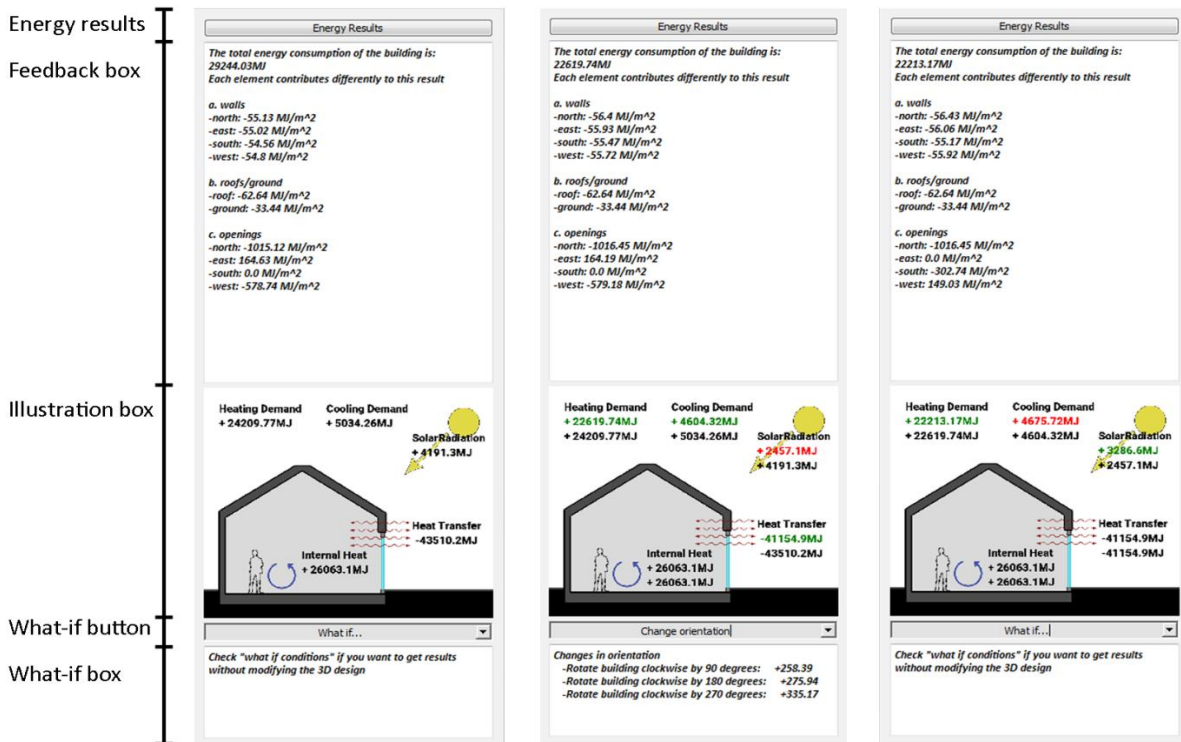


Figure 14: Test case results

A second way to check if the tool works properly was the usage of a *What-if* scenario. More specifically it was “asked” what would be the heat consequences by rotating the building. The results, appearing in the *What-if box* proved that the current building orientation was the worst alternative. The best scenario was the rotation of the building by 270° clockwise. Such an option would save approximately 335 MJ annually. Since this decision optimizes the building’s energy performance, the 3D model was rotated and re-exported. The final outcome has proved this feedback; the heating demand increased by 405 MJ, and the cooling demand decreased by 70 MJ (Figure 14c).

#### 4.5.2 Test case in VABI elements

The same test case (section 4.5.1) was conducted in VABI Elements since it makes use of the same formulas indicated in NEN 7120. From the fixed material library of the program, the walls and the windows were selected so that they have similar U-value as the ones in the case study. Once again, the three design “moments” were validated in VABI Elements. The first was the original 3D model, then two north windows were removed, and finally the building was rotated by 270° clockwise. Finally it has to be mentioned, that the complexity of the use of VABI Elements combined with no existing simulation background in place, has allowed the request of outcomes in the field of heat loss (heat by transmission and heat by ventilation).

#### 4.5.3 Accuracy comparison of the two test cases

The results produced by VABI Elements compared to the ones of the developed tool are found in Appendix XI. As can be seen, a discrepancy in the final results can be found. This discrepancy can be explained by many factors. First of all, many assumptions in parameters that affect the heat loss were taken. These parameters included among others a heating and cooling generator, the HVAC system and building operation requirements. Moreover VABI Elements retrieves weather condition from NEN 5060, while the developed tool retrieves this information by NEN 7120. The main difference is that NEN 5060 provides an optimal calendar of hourly weather conditions throughout the year. On the other hand, the developed tool uses monthly weather values and a utilization factor (NEN 7120). Finally VABI elements do not have similar material library with Revit. Therefore it was not possible that identical materials are found for both cases.

Despite the discrepancies of the results, the comparison showed that the developed tool assess the building's environmental impact correctly. In both test cases, the removal of the two north openings has reduced the heat loss. On the other hand, changing the building's orientation has no impact in heat loss, since the orientation affects neither the heat by transmission nor the heat by ventilation.

#### 4.5.4 Process comparison of the two test cases

In the *Literature Review*, it was demonstrated that integrated BIM-BPS tools allow the data interoperability, they do not request human intervention and therefore they are faster. The comparison of the processes of the two test cases, has also justified these benefits.

First, in VABI Elements there was a need for human intervention. The problems that can arise from this intervention are two. First the human intervention can be associated with human errors. The user might evaluate wrong some requirements, might miss the completion of requirements, or might fall into data assumptions. Second, much time is spent on preparing the model for analysis. For example, time is needed to re-create the 3D model. Moreover any change made in the 3D BIM model, must be repeated again in the 3D model of VABI elements. Furthermore, to determine the building's functionality, it was necessary to download and select from VABI elements the office requirements, office use, and an office ventilation system. Similarly, the selection of each material has been done manually, again through incorporated libraries in VABI elements.

On the developed tool, the re-creation of the 3D model was unnecessary. Instead the tool was seeking to the 3D BIM model and retrieved only the necessary information. In the same



context, there was no need to select manually neither the material nor the thermal values. All these data were found mechanically. Finally the selection of the building's functionality was done immediately. Therefore, the example of the developed tool shows that there is no need for human intervention that minimizes both the human errors, and it reduces drastically the time needed for the energy analysis.

The developed tool has additional benefits associated with its process. These benefits are relevant to its automation and to the feedback sent to the architects.

On contrary to most BIM-BPS platforms, the developed tool does not conduct an on-demand energy analysis. Instead, it operates independently and automatically. As it was shown in the case study, the user selects once the building functionality, and afterwards he only modifies and exports new designs. The rest of the procedure is automatic. Automation can be found both in re-starting the analysis, and in actions such as generating the geometry, generating results and generating feedback. All of these actions are remained in the background and activated when the design is re-exported. The advantage of this automation procedure is that the developed tool does not interrupt the design process. More specifically, the user does not swap tools or windows; instead the outcome is revealed mechanically next to the 3D model.

The automation in restarting the analysis and the automation in receiving the data allow the generation of feedback that evaluate and inform the impact of the decision made. In the case study, all the decisions made were supported and guided by the feedback received. On the contrary, VABI elements have provided no evidence regarding the reasons that produce the final outcome. Therefore, through constant feedback, the developed tool allows architects to study, understand and evaluate every decision they make, check and suggest alternatives solutions since the early design phases, and finally gain knowledge regarding a building's energy performance.

Finally the developed tool allows an architect to take optimal decision without the consultancy of other engineers. The comprehensiveness of VABI elements has shown that it is difficult for architects to incorporate BPS tools in the early design. The performance evaluation is done by the engineer, who later consults the architect on the matter of eventual design modifications (Klitgaard & al., 2006). However, the test case has shown that the developed tool can be easily used by the architects. The final outcome can be evaluated and the architect can rely on it to take sustainable design decisions. Therefore, the tool reduces the communication between architects and consultants at the early stages, allowing the architect to design faster. This communication can be restored at the latest design stages, but by then, optimal decisions will have already been made regarding the building's envelope and materials.

Overall, the comparison between the processes of the developed tool and VABI elements has provided insights in the benefits of the developed tool. The energy analysis is faster than a normal BPS analysis, since there is no human intervention. Moreover the entire procedure is automatic, which allows the generation of feedback sent to the architect. Finally the tool can be easily used. All these benefits combined, compose a tool that can be used by architects since the earliest design stages, replacing the role of a consultant engineer, since the tool supports and guides the design decision making.

#### 4.6 Tool validation

A key component of the tool is the use of IFC formats. Among others, the specific format supports the use of non-vendor specific applications. Particularly, the application can be used by any architect, regardless his preference in 3D vendor-based platforms. The tool was constructed on Revit-IFC format. Therefore it was tested if the IFC formats exported by other vendor-platforms could activate the same procedure and allow the tool to operate smoothly. Time-wise restrictions reduce the number of these platforms, and thus only one additional commonly-used 3D platform was selected: Archicad.

The validation revealed that Archicad creates a similar IFC file to the Revit one. Most of the basic properties and attributes are similarly named, and therefore there is no any error in retrieving spatial data such as a wall's length, width and height. However, thermal data such as the *Heat Transfer Coefficient* are produced solely by Revit. Therefore the tool was modified to find equivalent values. For example the *Heat Transfer Coefficient* can be found in Archicad as *Thermal Transmittance*.

Archicad, in contrast to Revit, suggests a different approach to establishing the walls' joints. In Archicad the length of the wall includes a part of the previous wall. Diagram 4 demonstrates the difference in Archicad and Revit. Thus errors arise regarding the data creation, and more specifically the estimation of a wall's end point value. To overcome this limitation, the script was modified, and an additional variable has been added in the existing calculation formula (Equation 1). A Boolean value indicates whether a wall is dependent or not to the following wall, and it is expressed as 0 or 1. Therefore it maintains or eliminates a part of the formula. The latter is re-devised as follows (Equation 10):

$$EP_i = SP_i * AD_i * (L_i - BV * W_{i+1}) \quad (10)$$

EP = End Point

SP = Start Point

L = length

**BV = Boolean Value (0 or 1)**

W = width

i = current wall

i +1 = next wall

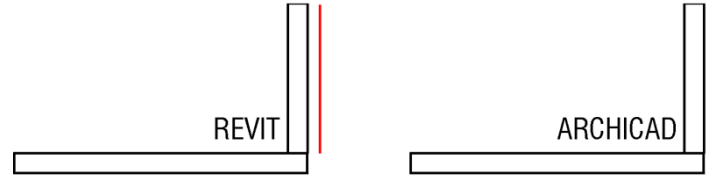


Diagram 4: Differences in Revit-Archicad in wall joints

Moreover, Archicad IFC exports may create IFC backups instead of overwriting the existing files. That interrupts the iteration of the process; the tool is re-activated only when IFC files are exported in the same file-path with the same file-name. However, this limitation does not lie in the developed tool's process but in the functionality of Archicad exports. To resolve this limitation, the tool should check for possible generation of back-up files. However, time-wise restrictions did not allow this possibility.

#### 4.7 Tool's limitations

Regardless the tool's benefits, as those expressed in section 4.5.4, existing technical problems are recognized. The tool was constructed according to the specification of IFC 2x3 format, produced by a Revit file. Three errors can arise because of this choice. First, the tool might not run smoothly in the newer version 2x4 even if the IFC structure remains the same (Kravari, 2017). Second, the IFC's flexibility in naming properties and attributes (in different vendor-based platforms) may prevent the tool to retrieve all the necessary data. The tool operates efficiently in Revit, but the validation in Archicad has already revealed this limitation. Third, the tool operates independently by the 3D vendor-based platforms. Therefore the tool is re-activated only when IFC files are exported in the same file-path with the same file-name. Once again this procedure might hide errors as in the case of Archicad validation.

The limitations of each datum creation were exposed in section 4.3.1. Overall, it was stressed the necessity that the architect draws parallel to the two main axes. As far as the extraction of the thermal values is concerned, the tool retrieves these values from fixed material libraries designed and provided by manufacturing companies (Jrade & al., 2013). However, this fixed library does not cover all of the possible materials combinations, and secondly it reduces the possibility for the architect to customize this library.

Regarding the energy calculations two limitations are recognized. First, the developed tool applies equations and retrieves data from the Dutch Normative. Therefore the geographic range is restricted to the region of the Netherlands. Second, the calculations are based on limited variables that might reduce the tool's accuracy. Section 4.5.3 has revealed the discrepancies between the developed tool and VABI Elements.

Finally the developed tool cannot be applied at the later design stages, since it was developed to capture the changes on design principles such as the building's shape and envelope.

## 4.8 Future improvements

Achieving improvements in the aforementioned limitations (4.7) could allow the developed tool to operate with a higher technical accuracy. First, a deep insight in the IFC format produced by different 3D vendor-based platforms, would resolve issues regarding the data extraction. Second, another improvement could reflect in the possibility that the architect selects the geographic region of the building; this selection would activate formulas and data from this region's Normative. Third, loading a material library to the tool, could allow the architect to select material samples with their associated thermal values. The architect could even replace building elements, such as walls or openings with comparable elements; a similar application has already been implemented in the tool developed by ElDiraby & al. (2017). Finally, in the same context, this library could allow the tool to seek design alternatives and send them as feedback.

The developed tool allows the architect to accomplish simple actions such as zoom and rotate. However existing IFC –read tools (i.e. BIM Vision) incorporate further interactive features (illustration of entities structure, entities details etc). A future improvement could include such features.

An improvement that could increase the tool's efficiency is its operation in cloud services such as Software-as-a-Service platforms. By this incorporation, common limitations associated with a traditional software use can be overcome. Among the benefits of cloud computing, is the easier extraction of data, integration with third party protocols allowing higher accuracy, and finally, the production of faster outcomes (Andriamamonjy & al., 2018; El-Diraby & al., 2017). Moreover all these benefits, could allow the tool to run during the entire design phase.

## CHAPTER 5: CONCLUSION



## 5.1 Scientific relevance

The scientific relevance of the research study is portrayed through a critical evaluation of the research questions, as those were formulated at the introduction of this study. The main research question is discussed after an assessment of the sub-questions.

### **How do current BPS systems operate? Which are their benefits and constraints?**

Nowadays, Building Performance Simulation (BPS) tools are abundant, with numerous applications and specific targets for each of a building's lifecycle phase. The formulas and principles used are law-driven describing the behavior of the energy use and demand. In a nutshell, BPS tools predict a building's energy performance before its actual erection. Due to this prediction, construction stakeholders (engineers, architects etc) understand the complexity of the energy balance, and they develop sustainable solutions. However, BPS tools have technical problems. These problems include the low data interoperability between them and 3D models, and second the "on-demand" energy performance analysis which evaluates neither the design progress nor the decision making. Other limitations, associated to the BPS tools, are relevant to their scope; most of BPS tools are informative, and they refer to engineers or energy modelers. Thus, they cannot provide feedback to architects so that they produce more sustainable buildings.

### **How can BIM technology be integrated into the BPS practice?**

A building's energy performance (BEP) is affected through a dynamic interaction of many parameters such as the building, the occupancy and weather conditions. Building Information Modeling (BIM) allows the development of nD models with incorporated semantic information about each construction element. The semantic information involves a building's spatial, thermal and HVAC data that BPS tools require. The *Literature Review* has revealed a range of scientific BPS tools that are constructed around this information and receive data from BIM models. The direct data transfer resolves interoperability issues between BIM and BPS tools allowing the latter to operate faster, more accurate, whilst human errors are minimized.

### **How do current BIM-BPS tools affect the design decision making for a sustainable building during the design process?**

The integration of BIM and BPS reduces time and interoperability issues. The current BIM-BPS tools support that this integration can produce faster comparable results from which an architect (himself) can evaluate the building's energy performance and design sustainably. Moreover, few examples, mostly vendor-based plug-in which operate in the BIM environment can be updated in every decision that the architect makes. Therefore the architect can understand whether his decision has improved the building's energy performance. Moreover this plug-in can keep history or provide comparison results; these are mechanisms that can be

seen as feedback. Overall, BIM-BPS tools can improve the design decision making, since the architect receives information in order to optimize the building's energy performance.

The aforementioned three questions allow evaluating and discussing the context of the main research question.

### **How BIM and BPS tools can be integrated in order to provide an interactive system which guides and supports the design decision making for a sustainable building during the design process?**

The ultimate purpose of this graduation study is to encourage sustainability in the construction industry. The main research question was that this sustainability can be indirectly succeeded through influencing the design decision-making. To exercise this influence, an integrated BIM-BPS tool was developed aiming in guiding the architects toward a sustainable design during the earlier design stages. By illustrating energy results immediately and in interactive way as the design evolves, the architect can identify how the building elements affect the energy demand. Based on that, he can take optimal sustainable solutions.

For the development of this tool, the current approach of BIM-BPS tools was modified. A causal-effect loop was created where output of BIM models are input for the BPS tool and consequently, the output of the BPS tool is the BIM model's input. Moreover the process is automated so that it is updated as the design is modified. Finally, in order to support the design decision making, the tool is interactive by generating and sending feedback to the user. In essence the feedback shows how each decision making has affected or will affect the energy demand. It includes simple numeric values that reveal how each component determines the energy demand. Secondly, the tool can produce in advance design alternatives that could improve the building's energy performance. Finally the comparison of previous and current outcomes is visible.

## **5.2 Societal relevance**

The societal relevance of this research emanates from the necessity of the construction industry to deliver sustainable solutions. Sustainability implementations and practices in the AEC industry, ensures a healthy built environment, with an occupants' comfort and society's welfare (Kravari, 2017). The developed tool facilitates this sustainability by providing law-driven results to architects who can evaluate them and create efficient and environmental-friendly buildings. Moreover, the tool can be used for educational practices, so that the architects gain valuable experience in the complexity of building's performance.

### 5.3 Recommendations for further research

In the introduction, the research objective of this study was expressed as the development of a vendor-neutral and evaluative BIM tool that could conduct BEP analyses. Even if the objective was reached, further research is recommended in two key aspects of the developed tool: its automated process and its generated feedback.

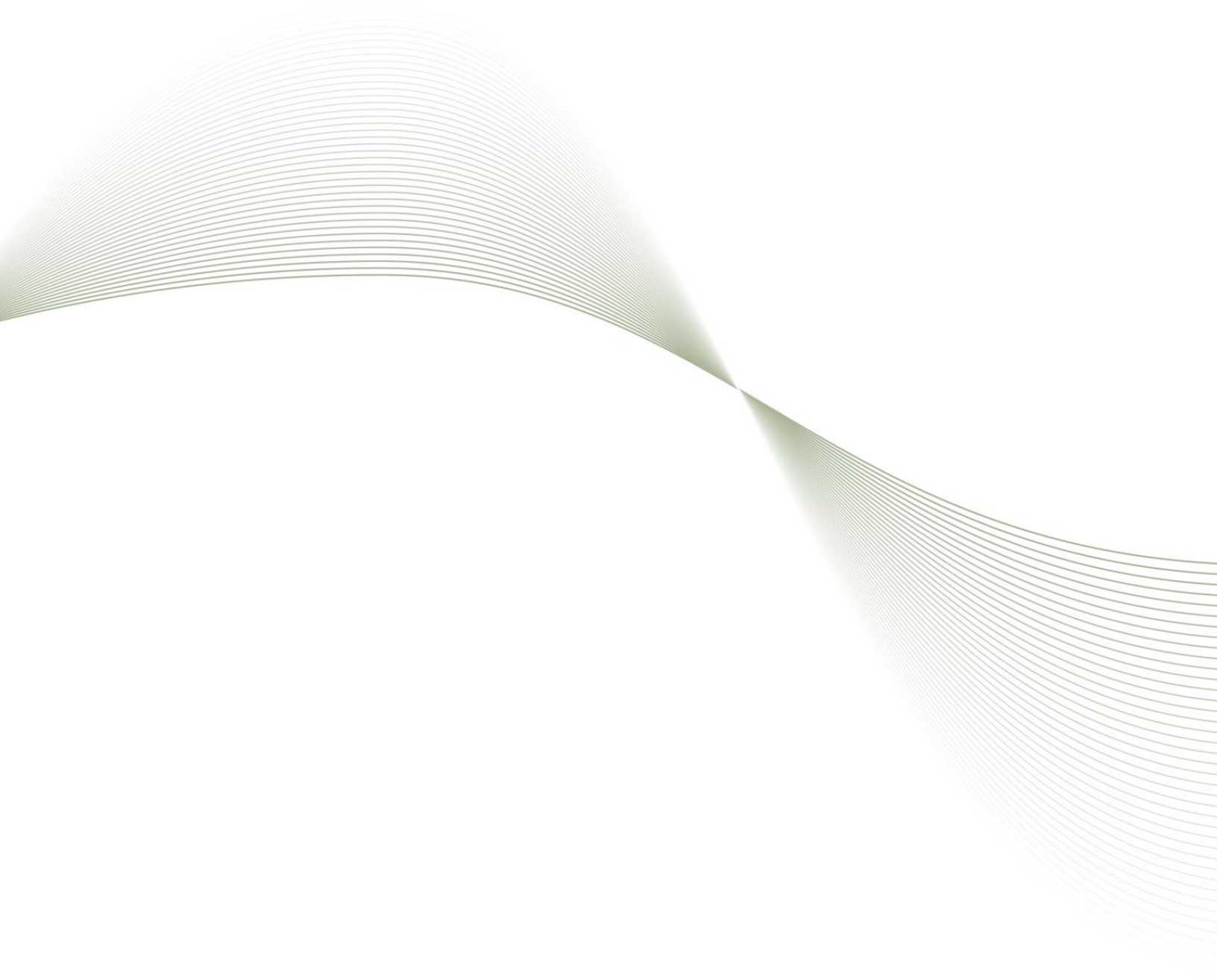
The tool's process has already been described, and it has been proved that a universal and neutral-vendor tool can incorporate the BIM information in other domains such as the Building Performance Simulation. In fact, the tool development has demonstrated that the IFC format, can resolve interoperability issues between (BIM) platforms. As a design program evolves, the IFC format is modified, which in consequence (re)activates a second system. In this research study, the second system generated BEP analysis at the early design phases.

Further research can be conducted in the aforementioned process. For example it can be studied how such a BPS system can operate in the entire building's lifecycle. Moreover, a similar automated process can be used for tools that are intermediary between existing BIM and BPS tools. This could resolve accuracy limitations, as these were found in the developed tool. Finally, research can be conducted on how such an interactive process can be implemented in other management applications that are affected directly by the design evolution. These can be the quantities take off, cost estimations, risk management and time management.

The second area of research is the generation of feedback by BPS tools. The feedback provided to the user reveals the environmental impact of each building component, and secondly shows the environmental impact of design decisions. In fact, *feedback* is an essential feature enabling BPS tools to gain an evaluative character and therefore guide the user to design sustainable. This possibility addressed in this research can be fully explored further. For example they can accomplish how-to scenarios or even take themselves corrective actions. These features are not yet implemented, and therefore further research is recommended to explore the capabilities of feedback provided by BPS tools.



## CHAPTER 6: REFERENCES



- [1]. Adeli, H., (2009), *Vision for civil and environmental engineering departments in the 21st century*, Journal of Professional Issues in Engineering Education and Practice, vol.135.11. 1–3, [https://doi.org/10.1061/\(ASCE\)1052-3928\(2009\)135:1\(1\)](https://doi.org/10.1061/(ASCE)1052-3928(2009)135:1(1))
- [2]. Aia.org., (2012), *Architect's guide to integrating energy modeling in the design process*, [online], Available at: <https://www.aia.org.org/resources/8056-architects-guide-to-integrating-energy-modeli>, [Accessed 24 Jan. 2018].
- [3]. Alwan, Z., Greenwood, D. & Gledson, B., (2015), *Rapid LEED evaluation performed with BIM based sustainability analysis on a virtual construction project*, Construction Innovation, Vol. 15 Issue: 2 (pp.134-150), <https://doi.org/10.1108/CI-01-2014-0002>
- [4]. Andriamamonjy A., Saelens D. & Klein R., *An automated IFC-based workflow for building energy performance simulation with Modelica*, Automation in Construction, vol.91 (pp.166-181), <https://doi.org/10.1016/j.autcon.2018.03.019>
- [5]. Ariyaratne, C. I., & Moncaster, A. M., (2014), *Stand-alone Calculation Tools are not the Answer to Embodied Carbon Assessment*, Energy Procedia, 62 (pp.150–159), <https://doi.org/10.1016/j.egypro.2014.12.376>
- [6]. Athienitis A., O' Brien, W., (2015), *Modeling, Design, and Optimization of Net-Zero Energy Buildings*, <https://doi.org/10.1002/9783433604625>
- [7]. Attia S., Gratia E., de Herde, A. & Hensen J.L.M., (2012), *Simulation-based decision support tool for early stages of zero-energy building design*, Energy and Buildings 49 (pp.2-15), <https://doi.org/10.1016/j.enbuild.2012.01.028>
- [8]. Azhar, S., (2011), *Building information modeling (BIM): Trends, benefits, risks, and challenges for the AEC industry*, Leadership and Management in Engineering, 11(3) (pp. 241–252), <https://ascelibrary.org/doi/10.1061/%28ASCE%29LM.1943-5630.0000127>
- [9]. Azhar, S., Hein, M.F. & Sketo, B., (2014), *Building Information Modeling(BIM): Benefits, risks and challenges*, Available at: [https://www.researchgate.net/publication/237569739\\_Building\\_Information\\_Modeling\\_BIM\\_Benefits\\_Risks\\_and\\_Challenges](https://www.researchgate.net/publication/237569739_Building_Information_Modeling_BIM_Benefits_Risks_and_Challenges)
- [10]. Azhar, S., Nadeem, A., Mok, J. Y., & Leung, B. H., (2008), *Building Information Modeling (BIM): A new paradigm for visual interactive modeling and simulation for construction projects*, In Proc., First International Conference on Construction in Developing Countries (pp. 435–446), Available at: [https://www.researchgate.net/publication/283118367\\_Building\\_Information\\_Modeling\\_BIM\\_A\\_New\\_Paradigm\\_for\\_Visual\\_Interactive\\_Modeling\\_and\\_Simulation\\_for\\_Construction\\_Projects](https://www.researchgate.net/publication/283118367_Building_Information_Modeling_BIM_A_New_Paradigm_for_Visual_Interactive_Modeling_and_Simulation_for_Construction_Projects)
- [11]. Bambardekar, S. and Poerschke, U., (2009), *The Architect as performer of Energy Simulation in the Performance Based Design*, Building Simulation: Eleventh International IBPSA Conference (pp. 1306-1313), Available at: [http://www.ibpsa.org/proceedings/BS2009/BS09\\_1306\\_1313.pdf](http://www.ibpsa.org/proceedings/BS2009/BS09_1306_1313.pdf)

- [12]. Bazjanac, V., (2008), *IFC BIM-Based Methodology for Semi-Automated Building Energy Performance Simulation*, Proceedings of the 25th International Conference on Information Technology in Construction, Available at: <https://simulationresearch.lbl.gov/sites/all/files/919e.pdf>
- [13]. Bazjanac, V., Maile, T., Rose, C., O'Donnell, J.T., Mrazovic, N., Morissey, E. & Welle, B.R., (2011), *An assessment of the use of building energy performance simulation in early design*, Proceedings of Building Simulation 2011: 12th Conference of International Building Performance Simulation Association, (pp.1579-1585), Available at: [http://www.ibpsa.org/proceedings/BS2011/P\\_1531.pdf](http://www.ibpsa.org/proceedings/BS2011/P_1531.pdf)
- [14]. Becerik-Gerber, B. & Kensek, K., (2010), *Building Information Modeling in Architecture, Engineering, and Construction: Emerging Research Directions and Trends*, Journal of Professional Issues in Engineering Education and Practice, vol.136.10.1061, Available at: [https://www.researchgate.net/publication/238311037\\_Building\\_Information\\_Modeling\\_in\\_Architecture\\_Engineering\\_and\\_Construction\\_Emerging\\_Research\\_Directions\\_and\\_Trends](https://www.researchgate.net/publication/238311037_Building_Information_Modeling_in_Architecture_Engineering_and_Construction_Emerging_Research_Directions_and_Trends)
- [15]. Becerik-Gerber B, Rice S (2010) *The perceived value of building information modeling in the U.S. building industry*, Journal of Information Technology in Construction (ITcon), Vol. 15 (pg. 185-201), Available at: <https://www.itcon.org/paper/2010/15>
- [16]. Beetz, J., Braak, W. C., Botter, R., Zlatanova, S., & Laat, R. d., (2015). *Interoperable data models for infrastructural artifacts – a novel IFC extension method using RDF vocabularies exemplified with quay wall structures for harbors*, eWork and eBusiness in Architecture, Engineering and Construction (pp. 135-140), Available at: <https://repository.tudelft.nl/islandora/object/uuid:bb9a7dff-52c7-4aaf-a6b8-898432270620/datastream/OBJ/download>
- [17]. Berlo, L., (2018), *KUBUS openBIM roadshow Rotterdam*, [online], Available at: [https://www.slideshare.net/berlotti/kubus-openbim-roadshow-rotterdam?next\\_slideshow=1](https://www.slideshare.net/berlotti/kubus-openbim-roadshow-rotterdam?next_slideshow=1), [Accessed 20 July 2018]
- [18]. Borrmann, A., Beetz, J., Koch, C., & Liebich, T., (2015), *Industry Foundation Classes: a vendor-independent data model for the entire lifecycle of a building*, In A. Borrmann, M. Koenig, C. Koch, & J. Beetz (Eds.), *Building Information Modeling: Technological Fundamentals and Industrial Practice* (pp. 83-127), (VDI-Buch). Berlin: Springer, [https://doi.org/10.1007/978-3-658-05606-3\\_6](https://doi.org/10.1007/978-3-658-05606-3_6)
- [19]. Bozic, B., Mendel-Gleason, G., Debruyne, C., & O'Sullivan, D., (2016), *Computational History and Data-Driven Humanities*, Dublin: Springer.
- [20]. Bouwbesluit, (2012), *Tekst van het Bouwbesluit 2012 zoals dit luidt op 1 maart 2013. Deze tekst is samengesteld uit de Staatsbladen 2011, 416; 2011, 676; 2012, 441 en 2013, 75*, [online] Available at: <https://www.quiverte.com/wp-content/uploads/2015/06/integrale-versie-bouwbesluit-2012-per-1-3-2013.pdf>, [Accessed 15 May 2018]
- [21]. Bragança, L., Vieira, S. M., & Andrade, J. B., (2014), *Early Stage Design Decisions: The Way to Achieve Sustainable Buildings at Lower Costs*, The Scientific World Journal, 2014 (pp. 1–8) <https://doi.org/10.1155/2014/365364>

- [22]. Bribián, I.Z., Capilla, V.A., & Usón, A. A., (2011), *Life cycle assessment of building materials: Comparative analysis of energy and environmental impacts and evaluation of the eco-efficiency improvement potential*, Building and Environment, 46(5) (pp. 1133–1140), <https://doi.org/10.1016/j.buildenv.2010.12.002>
- [23]. BuildingSMART. (n.d.), *IFC Technology*, [online], Available at: <http://www.buildingsmart-tech.org/specifications/ifc-overview/ifc-technology>, [Accessed 18 July 2018]
- [24]. BuildingSMART. (n.d.), *Technical Vision*, [online], Available at: <https://www.buildingsmart.org/standards/technical-vision/>, [Accessed 18 July 2018]
- [25]. Cao, X., Dai, X. & Liu, J., (2016), *Building energy-consumption status worldwide and the-state-of-art technologies for zero-energy buildings during the past decade*, Energy and Buildings, 128 (pp. 198–213), <https://doi.org/10.1016/j.enbuild.2016.06.089>
- [26]. Chong, H.-Y., Lee, C.-Y., & Wang, X., (2017), *A mixed review of the adoption of Building Information Modeling (BIM) for sustainability*, Journal of Cleaner Production, 142 (pp. 4114–4126), <https://doi.org/10.1016/j.jclepro.2016.09.222>
- [27]. Coakley, D., Raftery P. & Keane M., (2014), *A review of methods to match building energy simulation models to measured data*, Renewable and Sustainable Energy Reviews 37 (pp. 123–141), <https://doi.org/10.1016/j.rser.2014.05.007>
- [28]. Crawley, D.B., Lawrie, L.K., Pedersen, C.O. & Winkelmann, F.C., (2000), *EnergyPlus: Energy simulation program*, Ashrae Journal 42 (pp.49-56), Available at: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.122.6852&rep=rep1&type=pdf>
- [29]. G2Crowd, (2018), *Best building design and building information modeling software*, [online], Available at: <https://www.g2crowd.com/categories/building-design-and-building-information-modeling-bim>, [Accessed 09 August 2018]
- [30]. Dankers, M., van Geel, F., & Segers, N. M. (2014). *A Web-platform for Linking IFC to External Information during the Entire Lifecycle of a Building*. Procedia Environmental Sciences, 22 (pp. 138–147), <https://doi.org/10.1016/j.proenv.2014.11.014>
- [31]. Designing Buildings, (2018), *Heat Transfer in buildings*, [online], Available at: [https://www.designingbuildings.co.uk/wiki/Heat\\_transfer\\_in\\_buildings#Conduction](https://www.designingbuildings.co.uk/wiki/Heat_transfer_in_buildings#Conduction)
- [32]. Distributed Wikipedia, (2017), *Shoelace Formula*, [online], Available at: [https://ipfs.io/ipfs/QmXoyvizjW3WknFIJnKLwHCnL72vedxjQkDDP1mXWo6uco/wiki/Shoelace\\_formula.html](https://ipfs.io/ipfs/QmXoyvizjW3WknFIJnKLwHCnL72vedxjQkDDP1mXWo6uco/wiki/Shoelace_formula.html), [Accessed 22 May 2018]
- [33]. Eastman, C., Teicholz, P., Sacks, R., & Liston, K. (2011), *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors*, Building (Vol.2), <http://doi.org/10.1002/9780470261309>

- [34]. El-Diraby, T., Krijnen, T. & Papagelis, M., (2017), *BIM-based collaborative design and socio-technical analytics of green buildings*, Automation in Construction 82 (pp. 59-74), <https://doi.org/10.1016/j.autcon.2017.06.004>
- [35]. EnergyPlus, (2016), *Getting Started*, vrs.8.7 [online documentation], Available at: [https://energyplus.net/sites/all/modules/custom/nrel\\_custom/pdfs/pdfs\\_v8.7.0/GettingStarted.pdf](https://energyplus.net/sites/all/modules/custom/nrel_custom/pdfs/pdfs_v8.7.0/GettingStarted.pdf) [Accessed 19 June 2018]
- [36]. EnergyPlus. (n.d.), *Engineering Reference, the reference to EnergyPlus calculations*, Available at: [https://energyplus.net/sites/default/files/pdfs\\_v8.3.0/EngineeringReference.pdf](https://energyplus.net/sites/default/files/pdfs_v8.3.0/EngineeringReference.pdf) [Accessed 18 June 2018]
- [37]. van Enk, E.E.E., (2016), *Guidelines for selecting the 'fit for-purpose' model complexity regarding building energy performance prediction*, Eindhoven University of Technology, Available at: <https://research.tue.nl/en/studentTheses/guidelines-for-selecting-the-fit-for-purpose-model-complexity-reg>
- [38]. Federal Energy Management Program. (n.d.). *The Business Case for Sustainable Design in Federal Facilities. Energy Efficiency and Renewable Energy*, [online] Available at: <https://www.energy.gov/sites/prod/files/2013/10/f3/bcsddoc.pdf> [Accessed 11 August 2018]
- [39]. GBPN, (n.d), *The Netherlands*, [online], Available at: <http://www.gbpn.org/databases-tools/bc-detail-pages/netherlands#Summary>, [Accessed 10 July 2018]
- [40]. Gerrish T., Ruikar K., Cook, M., Johnson M., Phillip M., & Lowry C., (2017), *BIM application to building energy performance visualization and management: challenges and potential*, Energy and Buildings 144 (pp. 218 – 228), <https://doi.org/10.1016/j.enbuild.2017.03.032>
- [41]. Gervasio H., Santos P. & da Silva, L.S., (2010), *Influence of thermal insulation on the energy balance for cold formed buildings*, Advanced Steel Construction 6 (2) (pp. 742-766), Available at: [https://www.researchgate.net/publication/260601744\\_Influence\\_of\\_thermal\\_insulation\\_on\\_the\\_energy\\_balance\\_for\\_cold-formed\\_buildings](https://www.researchgate.net/publication/260601744_Influence_of_thermal_insulation_on_the_energy_balance_for_cold-formed_buildings)
- [42]. Ghaffarianhoseini, A., Tookey, J., Ghaffarianhoseini, A., Naismith, N., Azhar, S., Efimova, O., & Raahemifar, K. (2017). *Building Information Modelling (BIM) uptake: Clear benefits, understanding its implementation, risks and challenges*, Renewable and Sustainable Energy Reviews, 75 (pp. 1046–1053), <https://doi.org/10.1016/j.rser.2016.11.083>
- [43]. Government.nl, (2017), *Central government encourages sustainable energy*, [online] Available at: <https://www.government.nl/topics/renewable-energy/central-government-encourages-sustainable-energy>, [Accessed 15 Dec. 2017].
- [44]. Groeneveld, J. J. & Zeiler, W., (2016), *Building performance simulation from a BIM: gebouw presatie simulatie vanuit BIM*, Available at: <https://research.tue.nl/en/publications/building-performance-simulation-from-a-bim-gebouw-presatie-simula>

- [45]. Grilo, A., & Jardim-Goncalves, R., (2011), *Challenging electronic procurement in the AEC sector: A BIM-based integrated perspective*, Automation in Construction, 20(2) (pp. 107–114), <https://doi.org/10.1016/j.autcon.2010.09.008>
- [46]. Gruber, T., (1993), *A Translation Approach to Portable Ontology Specifications*, Knowledge Acquisition, 5 (pp. 199-220), <https://doi.org/10.1006/knac.1993.1008>
- [47]. Hemsath, T., (2013), *Conceptual energy modeling for architecture, planning and design: impact of using building performance simulation in early design stages*, Proceedings of BS2013: 13th Conference of International Building Performance Simulation Association (pp. 376-384), Available at: [http://www.ibpsa.org/proceedings/BS2013/p\\_2015.pdf](http://www.ibpsa.org/proceedings/BS2013/p_2015.pdf)
- [48]. Hitchcock, R.J. & Wong J., (2011), *Transforming IFC architectural views BIMs for energy simulation*, 12th Conference of International Building Performance Simulation Association (1089-1095), Available at: [http://www.ibpsa.org/proceedings/BS2011/P\\_1391.pdf](http://www.ibpsa.org/proceedings/BS2011/P_1391.pdf)
- [49]. Hensen, J.L.M., (2011), *Building performance simulation for sustainable building design and operation*, Proceedings of the 60<sup>th</sup> anniversary Environmental Engineering Department, Available at: <http://purl.tue.nl/727421556883814.pdf>
- [50]. Huang, J., (2015), *A high-level overview of DOE-2 and related simulation platforms*, [online], Available at : <http://www.cpuc.ca.gov/WorkArea/DownloadAsset.aspx?id=5334>, [Accessed 18 July 2018]
- [51]. Jrade, A. & Jalaei, F., (2013), *Integrating building information modeling with sustainability to design building projects at the conceptual stage*, Building Simulation. 6. 10.1007, <https://doi.org/10.1007/s12273-013-0120-0>
- [52]. Kim, S. & Woo, J.H., (2011). *Analysis of the Differences in Energy Simulation Results between Building Information Modeling ( BIM )-based Simulation Method and the Detailed Simulation Method*, Proceedings of the Winter Simulation Conference (pp.3550-3561), Available at: <https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6148049>,
- [53]. Klitgaard, J., Kirkegaard P.H. & Mullins M., *On the integration of digital design and analysis tools*, Digital architecture and construction, 2006, (pp. 187-196), Seoul, <https://doi.org/10.2495/DARC060191>
- [54]. Koezjakov A., Urge-Vorsatz D., Crijns-Graus W. & van der Broek M., (2018), *The relationship between operational energy demand and embodied energy in Dutch residential buildings*, Energy and Buildings (2018) 165, (pp.233-245), <https://doi.org/10.1016/j.enbuild.2018.01.036>
- [55]. Kravari, M.N., (2017), *Environmental impact assessment of building materials in BIM models Computation of embodied carbon in building materials with the use of Industry Foundation Classes and Semantic Web technologies*, Eindhoven University of Technology, Available at: <https://www.ofcoursecme.nl/?mdocs-file=3806>
- [56]. Kriegel, E. & Nies, B., (2008), *Green BIM*, Indianapolis, USA: John Wiley & Sons.

- [57]. Kumar, S., (2008), *Interoperability between building information models (BIM) and energy analysis programs*, University of South California, Available at: <http://digitallibrary.usc.edu/cdm/ref/collection/p15799coll127/id/64743>
- [58]. Levitt, R. E., (2007), *CEM research for the next 50 years: Maximizing economic, environmental, societal value of the built environment*, Journal of Construction Engineering and Management vol. 133 Issue 9, [https://doi.org/10.1061/\(ASCE\)0733-9364\(2007\)133:9\(619\)](https://doi.org/10.1061/(ASCE)0733-9364(2007)133:9(619))
- [59]. Library of Congress. (2016), *Industry Foundation Classes (IFC), Clear Text Family*, [online], Available at: <https://www.loc.gov/preservation/digital/formats/fdd/fdd000447.shtml>, [Accessed 20 July 2018]
- [60]. Lin, S.H. & Gerber, D.J., (2014), *Evolutionary energy performance feedback for design: Multidisciplinary design optimization and performance boundaries for design decision support*, Energy and buildings 84 (pp. 426-441), <https://doi.org/10.1016/j.enbuild.2014.08.034>
- [61]. McPhee, A., (2013), *IFC, What is it good for?*, [online], Available at: <http://practicalbim.blogspot.com/2013/06/ifc-what-is-it-good-for.html> [Accessed 18 July 2018]
- [62]. Migilinskas, D., Popov, V., Juocevicius, V., & Ustinovichius, L., (2013), *The benefits, obstacles and problems of practical BIM implementation*, Procedia Engineering, 57 (pp. 767–774), <http://doi.org/10.1016/j.proeng.2013.04.097>
- [63]. Negendahl, K., (2015), *Building Performance Simulation in the early design stage: An introduction to integrated dynamic models*, Automation in Construction 54 (pp.39-53), <https://doi.org/10.1016/j.autcon.2015.03.002>
- [64]. NEN, (2008), *Hygrothermische eigenschappen van gebouwen – Referentieklimaatgegevens*
- [65]. NEN,(2008), *NEN-EN-ISO 13790: Energieprestatie van gebouwen- Berekening van het energiegebruik voor verwarming en koeling (ISO 13790:2008,IDT)*
- [66]. NEN, (2012), *NEN 7120 +C2: Energieprestatie van gebouwen-Bepalingsmethode*
- [67]. Nowak P., Ksiazek M., Draps M. & Zawistowski J., (2016), *Decision Making with use of Building Information Modeling*, Procedia Engineering 135 (2016) (pp. 519 -526), <https://doi.org/10.1016/j.proeng.2016.08.177>
- [68]. Omnirie, (2017), *Unleash the power of digital collaboration with Building Information Modeling*, [online], Available at: <http://omnirie.com/BIM-Building-Information-Modeling.html>, [Accessed, 31 July 2018]
- [69]. OpenStudio (n.d), *What is OpenStudio*, [online], Available at: <https://www.openstudio.net/>, [Accessed, 17 August 2018]

- [70]. Østergård, T., Jensen, R.L. & Maagaard, S.E., (2016), *Building simulations supporting decision making in early design – A review*, Renewable and Sustainable Energy Reviews 61(pp.187–201), <https://doi.org/10.1016/j.rser.2016.03.045>
- [71]. Pomponi, F. & Moncaster, A. (2016), *Embodied carbon mitigation and reduction in the built environment – What does the evidence say?*, Journal of Environmental Management, 181, (pp. 687–700), <https://doi.org/10.1016/j.jenvman.2016.08.036>
- [72]. Pöyry, A., Säynäjoki, A., Heinonen, J., Junnonen, J.-M., & Junnila, S., (2015), *Embodied and Construction Phase Greenhouse Gas Emissions of a Low-energy Residential building*, Procedia Economics and Finance, 21, (pp. 355–365), [https://doi.org/10.1016/S2212-5671\(15\)00187-2](https://doi.org/10.1016/S2212-5671(15)00187-2)
- [73]. PriMusIFC (n.d), *The BIM Quantity Takeoff and construction estimating software from BIM model*, [online], Available at: <https://www.accasoftware.com/en/bim-quantity-takeoff-software>, [Accessed 16 June 2018]
- [74]. Revit, (2005), *Building Information Modeling for sustainable design*, [White Paper], Available at: [http://images.autodesk.com/latin\\_am\\_main/files/bim\\_for\\_sustainable\\_design\\_oct08.pdf](http://images.autodesk.com/latin_am_main/files/bim_for_sustainable_design_oct08.pdf) [Accessed 03 July 2018]
- [75]. Sefaira (2013), *4 Rules of thumb that could lead you astray*, [White Paper], Available at: <http://sefaira.com/resources/4-rules-of-thumb-that-could-lead-you-astray/> [Accessed 15 June 2018]
- [76]. Sefaira (2017), *Collaborative software for high-performance design*, [online], Available at: <http://sefaira.com/sefaira-architecture/>, [Accessed 15 June 2018]
- [77]. Shan M., Wang P., Li, J., Yue G. & Yang X., (2015), *Energy and environment in Chinese rural buildings: Situations, challenges and intervention strategies*, Building and Environment (2015), 91 (pp. 271–282), <https://doi.org/10.1016/j.buildenv.2015.03.016>
- [78]. Succar, B., (2009), *Building information modeling framework: A research and delivery foundation for industry stakeholders*, Automation in Construction, 18(3) (pp. 357–375), <http://doi.org/10.1016/j.autcon.2008.10.003>
- [79]. Tekla Structures, (2018), *Limitations in IFC object conversion*, [online], Available at: [https://teklastructures.support.tekla.com/2018/en/int\\_IFC\\_object\\_conversion\\_limitations](https://teklastructures.support.tekla.com/2018/en/int_IFC_object_conversion_limitations) [Accessed 18 July 2018]
- [80]. Torma, S., (2013), *Semantic linking of building information models. In Semantic Computing (ICSC)*, 2013 IEEE Seventh International Conference on (pp. 412–419), Available at: <https://ieeexplore.ieee.org/abstract/document/6693555/>
- [81]. Tukker, A., (2000), *Life Cycle Assessment as a tool in environmental impact assessment*, Environmental Impact Assessment Review, 20 (2000) (pp. 435–456), [https://doi.org/10.1016/S0195-9255\(99\)00045-1](https://doi.org/10.1016/S0195-9255(99)00045-1)



- [82]. Venugopal, M., Eastman, C. M., Sacks, R., & Teizer, J., (2012), Semantics of model views for information exchanges using the industry foundation class schema. *Advanced Engineering Informatics*, 26(2) (pp.411–428), <https://doi.org/10.1016/j.aei.2012.01.005>
- [83]. Visschers, M.G., (2016), *Building Information Modeling (BIM) based whole-building energy analysis towards an improved interoperability a conversion from the IFC file format to a validated gbXML file format*, Eindhoven University of Technology, Available at: <https://www.ofcoursecme.nl/?mdocs-file=3201>
- [84]. Wang S., Wainer, G. A., Rajus V. & Woodbury R., (2013), *Occupancy analysis using building information modeling and cell-DEVS simulation*, Proceedings of the Symposium on theory of modeling & simulation, Available at: [https://www.researchgate.net/publication/262361081\\_Occupancy\\_analysis\\_using\\_building\\_information\\_modeling\\_and\\_cell-DEVS\\_simulation](https://www.researchgate.net/publication/262361081_Occupancy_analysis_using_building_information_modeling_and_cell-DEVS_simulation)
- [85]. Weisheng L, C. W. (2017). *Computational Building Information Modelling for construction waste management: Moving from rhetoric to reality*. *Renewable and Sustainable Energy Reviews*, 68 (pp. 587-595), <https://doi.org/10.1016/j.rser.2016.10.029>
- [86]. Wen L. & Hiyama K., (2016), *A review: simple tools for evaluating the energy performance in early design stages*, *Procedia Engineering* 146 (pp.32–39), <https://doi.org/10.1016/j.proeng.2016.06.349>
- [87]. Zhang, C., Beetz, J. & Weise M., (2014), *Interoperable validation for IFC building models using open standards*, *Journal of Information Technology in Construction*, vol. 20. Issue number ECPPM-2014 (pp. 24-39), Available at: <https://www.itcon.org/paper/2015/2>
- [88]. Zhao, H. & Magoulès, F., (2012), *A review on the prediction of building energy consumption*, *Renewable and Sustainable Energy Reviews* 16 (pp. 3586–3592), <https://doi.org/10.1016/j.rser.2012.02.049>



APPENDIX



## APPENDIX I: Advantages and disadvantages of early design BPS tools

Table 4: Early design BPS tools

Early Design Performance Modeling Tool	Advantages	Challenges
Ecotect	<ul style="list-style-type: none"> <li>Somewhat Revit/ Autocad compatible</li> <li>Graphic intuitively understandable results</li> <li>Easy to learn/use</li> </ul>	<ul style="list-style-type: none"> <li>For-purchase software</li> <li>Not well supported</li> <li>Models not back-compatible</li> </ul>
OpenStudio	<ul style="list-style-type: none"> <li>SketchUp-style input models</li> <li>Freeware</li> <li>EnergyPlus analysis engine</li> <li>Easy to use if familiar with SketchUp</li> </ul>	<ul style="list-style-type: none"> <li>Only provides numeric output (currently to graphic results)</li> <li>SketchUp not yet compatible with Autocad/Revit/Archicad etc.</li> <li>Some training on defining components that E+ understands required</li> </ul>
COMFEN(Commercial Projects) REFSEN (Residential Projects)	<ul style="list-style-type: none"> <li>Very easy to use</li> <li>Freeware</li> <li>Well supported (LBNL-helpdesk)</li> <li>EnergyPlus analysis engine</li> <li>Graphic intuitively understandable results</li> <li>Provides the broadest range of performance implications (including energy)</li> </ul>	<ul style="list-style-type: none"> <li>Provides only envelope alternatives analysis (doesn't address mechanical or electrical system alternatives specifically)</li> <li>Only assesses performance of a single-zone (does not address a whole building)</li> <li>Not yet compatible with Autocad / Revit / Archicad, etc</li> </ul>
Spreadsheets (Rmi-Emit 1.2)	<ul style="list-style-type: none"> <li>Easy to use if familiar with spreadsheets</li> <li>Freeware</li> </ul>	<ul style="list-style-type: none"> <li>Only assesses specific components (does not address a whole building)</li> <li>Provides numeric output with only basic graphic results</li> </ul>
Sefaira	<ul style="list-style-type: none"> <li>User-friendly, with SketchUp-type input environment</li> <li>Is a whole-building model (can specify different conceptual mechanical systems)</li> <li>Graphic intuitively understandable results</li> <li>Allows comparisons of multiple options side-by-side</li> <li>Uses its own engine (faster, multiple models)</li> </ul>	<ul style="list-style-type: none"> <li>For-purchase software</li> <li>Not yet compatible with Autocad / Revit / Archicad etc.*</li> <li>Uses its own engine (black box / invalidated by ASHRAE?)*</li> <li>Doesn't provide code compliance information</li> </ul>
Whole-Building Em-Tools (Equest, Simergy, etc.)	<ul style="list-style-type: none"> <li>Some are somewhat Revit / Autocad / Archicad / SketchUp compatible</li> <li>Early component assessment can easily transition into a whole-building energy analysis</li> </ul>	<ul style="list-style-type: none"> <li>Some are for-purchase software, some are freeware</li> <li>Not easy to learn / use without training</li> <li>Typically only provides numeric output</li> </ul>

\* Currently this challenge has been met

## APPENDIX II: Process map

Name	Process Map [connect 3D model with software]
------	--

Identifier	PC 01
------------	-------

Change Log		
2018 – 04 - 25	PM Created	m.gkatzios@student.tue.nl
2018 – 07 - 25	PM Modified	m.gkatzios@student.tu.nl

Exchange Requirements	ER Design model
	Energy analysis

### Scope

The following process map describes the interaction between the architect and the developed tool, during the early design process. By connecting a 3D design model with the tool, the architect can see how his decisions affect the energy analysis. The only party involved in the process map, is the architect.

### General Description

The main objective of the process map is to suggest the way that the documentations exchange flows, (e.g. designs, reports, requirements etc.) including the corresponding formats, amongst the architect and the developed tool.

The process map describes the necessary steps that the architect has to accomplish. However those steps actually include the export of the 3d model as an IFC file and the latter's import in the tool.

POOL : Architect

LANE : Architect

Design 3D model [ID:Ar01]

Type	activity
Name	Design 3D model
Documentation	This activity has no prior requirements as it illustrates the beginning of the process map. In general the architect should create the first 3D design model, that it should include at least 1 wall so that the developed tool produces results.

POOL : Architect

LANE : Architect

Set IFC export options [ID: Ar02]

Type	activity
Name	Set IFC export options
Documentation	This activity has no prior requirements since it can happen either before or after the first design. It is an important procedure because the user has to set up the IFC export so all of the required elements to be exported, as these are also mentioned later in the ER.

POOL : Architect

LANE : Architect

Export IFC model [ID:Us03]

Type	activity
Name	Export IFC model
Documentation	This activity requires that the architect has already designed a 3D model with the minimum requirements [Ar01] and that he has already set the IFC export options [Ar02]. It is recommended that the architect sets a shortcut for the export so that it becomes faster.

POOL : Architect  
 LANE : Architect  
 Import IFC model to the developed tool [ID:Ar04]

Type	activity
Name	Import IFC model to the developed tool
Documentation	This activity requires that the architect has already exported an IFC model [Ar03]. This activity need to be done only once, and it actually illustrates the connection of the tool with the 3D model.

POOL : Developed Tool  
 LANE : Developed Tool  
 Create the building characteristics [ID:DT01]

Type	activity
Name	Create the building characteristics
Documentation	This activity requires that the architect has already imported the IFC file to the developed tool [Ar04]. The tool extracts the data that it needs, and it creates the data that they cannot be found in the IFC format.

POOL : Developed Tool  
 LANE : Developed Tool  
 Make energy analysis [ID:DT02]

Type	activity
Name	Make energy analysis
Documentation	This activity requires that the developed tool has all of the data that it needs [DT01]. Based on these data, and the formulas of NEN7120 it can make the energy analysis, and create the associated feedback.

POOL : Developed Tool  
 LANE : Developed Tool  
 Illustrate the results [ID:DT03]

Type	activity
Name	Illustrate the results
Documentation	This activity requires that the tool has already made the energy analysis [DT02]. The illustration of the results is necessary so that the architect can evaluate his decisions.

POOL : Architect  
 LANE : Architect  
 Evaluate the feedback [ID:Ar05]

Type	activity
Name	Evaluate the feedback
Documentation	This activity requires that the architect can see the illustrated results from the tool [DT03]. The architect doesn't have to evaluate the results; instead the evaluation is included in the feedback. Therefore he needs to read how his decision(s) modified the building's energy performance.

POOL : Architect  
 LANE : Architect  
 Modify the design [ID:Ar06]

Type	activity
Name	Modify the design
Documentation	This activity requires that the architect has already evaluated the feedback [Ar05]. Based on those, he can modify the design in order to improve the building's energy performance.



POOL : Architect  
 LANE : Architect  
 Export IFC model [ID:Ar07]

Type	activity
Name	Export IFC model
Documentation	This activity requires that the architect has already modified the design [Ar06]. This activity is really important because it restarts automatically the tool, and it generates the new results and feedback. For better results it is recommended that this activity is completed after each important decision making, so that the results show how this decision affected the building's energy performance.

Modify the design?

Type	Gateway
Name	Modify the design?
Documentation	This gateway requires that the architect has already evaluated the feedback [Ar05]. Based on those he can choose whether he will modify the design on the basic architecture principles (building envelope, orientation etc) or he will take more detailed design decisions. The second option will terminate the process map.

Energy analysis preparation

Type	Event
Name	Energy analysis preparation
Documentation	This is a start event and initiates the process.

### Termination of the early design phase

Type	Event
Name	Termination of the early design phase
Documentation	This is the event which terminates the process. It occurs when the architect does not interfere with basic architecture principles, but instead he takes more detailed design decisions. This event can be only occurred by the gateway “ <i>Modify the design?</i> ”.

### ER Design model

Type	Data Object
Name	ER IFC model
Documentation	The client ought to import in the software the IFC file derived from the design program. That document includes all the IFC elements that the developed tool needs in order to complete the energy analysis. These can be simple IFC elements such as doors or windows, but it should include their analytical values such as the Heat Transfer Coefficient

### Energy analysis

Type	Data Object
Name	Energy analysis
Documentation	These are the results that the tool gives back to the architect. The results include graph results and the feedback, in an independent pop-up window.

### APPENDIX III: Exchange requirement

Name	IFC Model
------	-----------

Identifier	ER IFC model
------------	--------------

Change Log		
2018 – 04 - 25	PM created	m.gkatzios@student.tue.nl

Project Stage	0	Portfolio requirements	
	1	Conception of need	
	2	Outline feasibility	
	3	Substantive feasibility	
	4	Outline conceptual design	✓
	5	Full conceptual design	
	6	Coordinated design and procurement	
	7	Production information	
	8	Construction	
	9	Operation and maintenance	
	10	Disposal	

Table 5: Exchange requirements

Exchange Requirement					
Element	Property concept	Property name	Definition	Example and explanation test using a rule in pseudo code	
Wall					
	General wall requirements				
		Identification	A unique ID for the identification of each wall	ID number of a total 22 numbers and/or symbols and/or letters	
	Specific wall requirements				
		Width	The width of the wall	0.3	
		Length	The length of the wall	1.2	
		Height	The height of the wall	2.4	
		Area	The area of the wall	5	
		Placement	The insert placement of the wall	(0, 0, 0)	
		Axis direction	The direction the wall moves to	(0, 1)	
		External wall	The building shall have external walls	External	
		Heat Transfer Coefficient	The heat transfer coefficient of the wall	0.2	
		Solar absorptance	The solar absorptance of the wall	0.7	
Opening (Door or Window)					
	General opening requirements				
		Identification	A unique ID for the identification of each opening	ID number of a total 22 numbers and/or symbols and/or letters	
	Specific opening requirements				
		Length	The length of the opening	1.2	
		Height	The height of the opening	2.2	
		Placement	The placement of the opening related to the wall	Wall building element	
		Heat Transfer Coefficient	The heat transfer coefficient of the opening	2.5	
		Solar absorptance	The solar absorptance of the opening	0.7	

- 
1. Either one of: M - Mandatory / O – Optional / R – Recommended / N - Shall not be used
  2. Either one of: Boolean, Integer, Real, String, Relation, Object type, etc.
  3. Where applicable: Number of items per property such as <=2 (less or equal to two items per property)
  4. Where applicable: (e.g. m, m<sup>2</sup>, m<sup>3</sup>, etc.)

	Existence <sup>1</sup>	Value Type <sup>2</sup>	Cardinality <sup>3</sup>	Unit <sup>4</sup>	IFC mapping
	M	String	n/a	n/a	IfcWall → IfcWall.GlobalId
	M	Real	>= 0.1	m	IfcWall → IfcPropertySet → Property.Name = "Width"
	M	Real	>= 0.3	m	IfcWall → IfcPropertySet → Property.Name = "Length"
	M	Real	>= 2.4	m	IfcWall → IfcPropertySet → Property.Name = "Height"
	O	Area	>= 0.8	m <sup>2</sup>	IfcWall → IfcPropertySet → Property.Name = "Area"
	M	Real	List of three points	n/a	IfcWall → IfcWall.ObjectPlacement.RelativePlacementLocation.Coordinates
	M	Real	List of two points	n/a	IfcWall → IfcWall.ObjectPlacement.RelativePlacementRefDirection.DirectionRatios
	M	Boolean	n/a	n/a	IfcWall → IfcPropertySet → Property.Name = "IsExternal"
	M	Real	>= 0	W / (m <sup>2</sup> * K)	IfcWall → IfcPropertySet → Property.Name = "Heat Transfer Coefficient (U)"
	R	Real	>= 0 & <= 1	n/a	IfcWall → IfcPropertySet → Property.Name = "Absorptance"
	M	String	n/a	n/a	IfcDoor/Window → IfcDoor/Window.GlobalId
	M	Real	>= 0.9	m	IfcDoor/Window → IfcDoor/Window.OverallWidth
	M	Real	>= 2.2	m	IfcDoor/Window → IfcDoor/Window.OverallHeight
	M	String	n/a	n/a	IfcDoor/Window → IfcDoor/Window.FillVoids.RelatingOpeningElement.VoidsElement
	M	Real	>= 0	W / (m <sup>2</sup> * K)	IfcDoor/Window → IfcPropertySet → Property.Name = "Heat Transfer Coefficient (U)"
	R	Real	>= 0 & <= 1	n/a	IfcDoor/Window → IfcPropertySet → Property.Name = "Absorptance"

## **OVERVIEW**

### **Scope**

The 3D design model and by consequence the IFC model is made by the architect in a 3D vendor-based platform such as Revit or Archicad. As a matter of fact it is the model out of which automated data will be retrieved, so that the energy analysis is conducted. When the architect designs at the 3D platform, most of these data are there, but the architect should ensure that this information will be also exported in the IFC model. Therefore, he must export an IFC with its IFC Guid, its analytical information of the model, and with a high level of detail.

### **General Description**

The IFC model is the programming representation of the design model. It is actually a structured form where all building elements are related to each other.

### **Information Description**

The minimum information requirements are a closed building envelope and at least one opening element. All of the other information can be automatically either found or created.

## **INFORMATION REQUIREMENTS**

### **Preconditions**

The user in order to export correctly the information he has to set up the IFC export options.

### **WALL**

This part includes the relevant information about the wall elements found in the 3D model.

#### **Wall General Requirements**

This part provides the general requirements for a wall.

#### **Opening Specific Requirements**

This part provides more specific requirements for a wall.

### **OPENINGS (Door/Window)**

This part includes the relevant information about the opening elements found in the 3D model.

#### **Opening General Requirements**

This part provides the general requirements for an opening.

#### **Opening Specific Requirements**

This part provides more specific requirements for an opening.

## APPENDIX IV: Data required for the developed tool

Table 6: Available and calculated Data

Building Characteristics	Available	Calculated
<b>General</b>		
-North Direction	✓	
-Total area		✓
<b>Wall</b>		
-External/Internal	✓	
-Start Point	✓	
-End Point		✓
-Direction	✓	
-Orientation		✓
-Length	✓	
-Height	✓	
-Width	✓	
-Area		✓
-Heat Transfer Coefficient	✓	
-Absorptance	✓	
<b>Opening</b>		
-Relation to wall	✓	
-External/Internal		✓
-Area	✓	
-Orientation		✓
-Heat Transfer Coefficient	✓	
-Absorptance	✓	
<b>Roof/Ground</b>		
-Area	✓	
-Heat Transfer Coefficient	✓	

## APPENDIX V: Temperature values

Table 7: Outside temperatures and time for each month (NEN, 2012)

	Temperature [ °C]	t <sub>mi</sub> [ms]
January	2.6	2.6784
February	5.0	2.4192
March	6.8	2.6784
April	9.3	2.5920
May	13.3	2.6784
June	16.0	2.5920
July	17.4	2.6784
August	17.4	2.6784
September	14.6	2.5920
October	11.3	2.6784
November	7.1	2.5920
December	4.0	2.6784

Table 8: Inside temperatures for each building functionality (NEN, 2012)

Building Functionality	Inside Temperature Heating Season [ °C]	Inside Temperature Cooling Season [ °C]
Residence	20	24
Hospital	22	24
Office	20	24
School	20	24
Store	20	24
Gym	13	24



## APPENDIX VI: Internal heat values for each building's functionality

Table 9: Internal heat by people (NEN, 2012)

Building Functionality	Heat Production by people / Square Meter [W/m <sup>2</sup> ]	Correction factor for occupation time
Residence	-	-
Hospital	5	0.8
Office	5	0.5
School	10	0.3
Store	3	0.4
Gym	3	0.3

Table 10: Internal heat by equipment (NEN, 2012)

Building Functionality	Heat Production by equipment / Square Meter [W/m <sup>2</sup> ]
Residence	-
Hospital	4
Office	4
School	2
Store	3
Gym	1

Table 11: Internal heat by electricity (NEN, 2012)

Building Functionality	Electricity demand/ square meter / year [W/m <sup>2</sup> ]	Total hours use / year
Residence	-	-
Hospital	17	4500
Office	16	2350
School	16	1750
Store	30	2800
Gym	16	2350

Table 12: Internal heat by fans (NEN, 2012)

Building Functionality	Ventilation Demand per Square Meter [W/m <sup>2</sup> ]
Residence	-
Hospital	12
Office	6.5
School	8.5
Store	6.5
Gym	4

Table 13: People per square meter (Bouwbesluit, 2012)

Building Functionality	Person /square meter [1/m <sup>2</sup> ]
Residence	-
Hospital	0.125
Office	0.05
School	0.125
Store	0.03
Gym	0.05

## APPENDIX VII: Solar radiation values

Table 14: Incident solar radiation (NEN, 2012)

	Roof	Ground	North	South	East	West
January	26.8	5.8	10.5	56.1	19.6	22.3
February	49.4	10.1	18.8	68.5	37.4	32.6
March	79.6	16.2	30.1	82.9	50.5	51.6
April	164.1	33.2	52.6	140.2	112.1	109.5
May	212.3	42.7	68.6	134.6	122	132.8
June	225.2	45.4	81.7	123.4	130.6	143.6
July	199.1	40.1	70.4	119.2	121.8	112.5
August	184.9	37.3	60.8	135.5	121.5	109.2
September	117.5	23.8	40.3	115.1	79.3	73.5
October	72.7	14.8	25.4	101.6	51	49
November	32.6	7	12.9	56.4	23.4	23.5
December	20.9	4.6	8.3	47.2	16.1	15.4

Table 15: Shadow reduction factor, Form factor (NEN, 2012)

	Roof	Ground	North	South	East	West
Shadow Reduction Factor	1	1	1	0.9	0.85	0.85
Form Factor	1	0	0.5	0.5	0.5	0.5

## APPENDIX VIII: Summary of the python script

After the selection of the building functionality the model operates in three steps as follows:

### Step 1: Define building elements' characteristics

1. Stores all the values relevant to the building functionality selection
2. Creates its own BIM elements (walls, windows, roof & ground). This is made to facilitate the access to data information when is necessary
3. Creates lists of these elements (2)
4. Gets north direction

For the walls:

5. Gets global Id
6. Gets if the wall is external or internal - maintains only the external walls
7. Get the axis direction
8. Get basic measurements (length, width, height & area)
9. Get thermal values (heat transfer coefficient, thermal mass)
10. Includes these data in the new walls (2)
11. Gets a lists for all widths(8)
12. Gets the start coordinates
13. Creates possible end points (6, 7, 8, 11, 12)
14. Defines final coordinates (13)
15. Gets external area (14)
16. Gets axis orientation(12,14)
17. Gets orientation (16)

For the roofs:

18. Gets global Id
19. Gets area (either by *IfcRoof* or by 14)
20. Gets thermal values (either by *IfcRoof* or 9)
21. Stores these data (18, 19, 20) in the new roofs(2)

For the grounds:

22. Find the lowest plane level (if exist)
23. Take same values (18, 19, 20) as roof (either by the lowest *IfcPlane* or 21)
24. Stores these data (23) in the new grounds (2)

For the openings:

25. Gets if it is window or door
26. Gets basic measurements (length, height & area)
27. Gets in which wall the opening is associated to
28. Gets if the opening is external or internal (6, 27) - maintains only the external openings
29. Gets orientation (17, 27)
30. Gets thermal values
31. Stores these data (25, 26, 28, 29, 30) in the new openings (2)

## **Step 2: Make energy analysis**

32. Reads daily temperatures by NEN 5060 file
33. Calculates heat transmission by each elements area and thermal values and (32)
34. Calculates heat by ventilation by elements areas and (32)
35. Calculates internal heat (1, 15)
36. Calculates solar radiation by each elements orientation, area and thermal values
37. Calculates utilization factor (33, 34, 35, 36)

## **Step 3: Illustrate the energy results**

38. Stores the energy results (33, 34, 35, 36) in *results list*
39. Stores the energy results (33, 34, 35, 36) in *what if list*
40. (in case of a new analysis) Stores new energy results (33, 34, 35, 36) in results list
41. Stores energy results for each element
42. Creates the energy graphs and tables (38)
43. Creates the feedback box (41)
44. Creates the images (38, 40)
45. Creates the what if scenarios
46. Produces a new *what if* analysis in request (39, 45)

## APPENDIX IX: Python script

```
#IMPORTS
import sys
import os,time
import win32file
import win32con

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

from PyQt4 import QtCore, QtGui
from PyQt4.QtGui import *
from PyQt4.QtCore import *
from OCC.Display.backend import get_backend
get_backend("qt-pyqt4")
import OCC.Display.qtDisplay
from OCC.Display.qtDisplay import qtViewer3d

from OCC.gp import *
import OCC.Bnd, OCC.BRepBndLib

from matplotlib.backends.backend_qt4agg import FigureCanvasQTAff as FigureCanvas
from matplotlib.backends.backend_qt4agg import NavigationToolbar2QT as NavigationToolbar
from matplotlib.figure import Figure
import matplotlib.pyplot as plt; plt.rcParamsDefaults()
import numpy as np
import pandas as pd

from PIL import Image, ImageDraw, ImageFont
from PIL.ImageQt import ImageQt

#INITIATE SELECTION
guid_selection = None

class ProductViewer(qtViewer3d):
    def __init__(self, *args):
        qtViewer3d.__init__(self)
        self.objects = {}

    @staticmethod
    def Hash(shape):
        return shape.HashCode(1 << 30)

    displayed_shapes = {}

    def Show(self, key, shape, color=None):
        self.objects[ProductViewer.Hash(shape)] = key
        qclr = OCC.Quantity.Quantity_Color(.35, .25, .1, OCC.Quantity.Quantity_TOC_RGB)
        ais = self._display.DisplayColoredShape(shape, qclr)
```

```

self.displayed_shapes[key] = ais
self._display.FitAll()

```

```

def Color(self, key):
    ais = self.displayed_shapes[key]
    qclr = OCC.Quantity.Quantity_Color(1, 0, 0, OCC.Quantity.Quantity_TOC_RGB)
    ais.GetObject().SetColor(qclr)

```

```

def ColorBack(self, key):
    ais = self.displayed_shapes[key]
    qclr = OCC.Quantity.Quantity_Color(.35, .25, .1, OCC.Quantity.Quantity_TOC_RGB)
    ais.GetObject().SetColor(qclr)

```

```

def ColorWhenElementAcceptable(self, key):
    try:
        ais = self.displayed_shapes[key]
        qclr = OCC.Quantity.Quantity_Color(0, 0.7, 0, OCC.Quantity.Quantity_TOC_RGB)
        ais.GetObject().SetColor(qclr)
    except KeyError as e:
        print( e )

```

```

def ColorWhenElementNotAcceptable(self, key):
    ais = self.displayed_shapes[key]
    qclr = OCC.Quantity.Quantity_Color(1, 0, 0, OCC.Quantity.Quantity_TOC_RGB)
    ais.GetObject().SetColor(qclr)

```

```

def mouseReleaseEvent(self, *args):
    qtViewer3d.mouseReleaseEvent(self, *args)
    if self._display.selected_shape:
        global guid_selection
        global selected_shape
        selected_shape = self._display.selected_shape
        guid_selection = (self.objects[ProductViewer.Hash(self._display.selected_shape)])

```

*#MAIN CLASS OF THE APPLICATION*

```

class initUI(object):

```

```

    def __init__(self, *args):
        # Constructing an application
        app = QtGui.QApplication(sys.argv)
        # Viewer initialization
        self.main = Main(self)
        self.main.show()
        self.main.canvas.InitDriver()
        self.main.statusBar()
        self.display = self.main.canvas._display
        # Raise a system exit
        sys.exit(app.exec_())

```

*#MAIN CLASS OF THE GRAPHICAL USER INTERFACE*

```

class ResultsDialog(QtGui.QWidget):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, None)

```

```

        self.setGeometry(100, 100, 800, 500)

class PopUpDialog(QtGui.QWidget):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, None, QtCore.Qt.WindowStaysOnTopHint)
        self.setGeometry(1250, 50, 300, 800)
        self.setWindowTitle("Energy Calculations - Feedback")

class Main(QtGui.QMainWindow):

    def __init__(self, parent=None):

        self.parent = parent
        QtGui.QMainWindow.__init__(self)

        # Instantiating the tabs
        global filename
        self.filename = None

        self.PopUpDialog = PopUpDialog(self)
        self.ResultsDialog = ResultsDialog(self)
        self.tabs = QtGui.QTabWidget()
        self.setCentralWidget(self.tabs)

        #VIEWER TAB
        self.viewerTab = QtGui.QWidget()
        self.tabs.addTab(self.viewerTab, "3D IFC-EC Viewer")

        # Implementing the OCC viewer
        self.canvas = ProductViewer(self)
        self.setGeometry(100, 100, 400, 400)
        self.setWindowTitle("Energy Performance - IFC Viewer")

        # Calling the tab

        self.tab3dView()

    @QtCore.pyqtSlot(str)
    def directory_changed(self, path):
        print('Directory Changed!!!')
        self.refreshIfcFile()

    @QtCore.pyqtSlot(str)
    def file_changed(self, path):
        print('File Changed!!!')
        self.refreshIfcFile()

    #DEFINE THE TAB (Boxes, Layout, Buttons etc)
    def tab3dView(self):
        # Initializing a split-view layout
        font = QtGui.QFont("Calibri", 9, QtGui.QFont.Bold, True)
        sizePolicy = QtGui.QSizePolicy(QtGui.QSizePolicy.Fixed, QtGui.QSizePolicy.MinimumExpanding)

```



```

self.visualizationBox = QtGui.QTextBrowser()
self.visualizationBox.setFont(font)
self.visualizationBox.setSizePolicy(sizePolicy)

self.whatIfBox = QtGui.QTextBrowser()
self.whatIfBox.setFont(font)
self.visualizationBox.setSizePolicy(sizePolicy)
self.whatIfBox.setFixedHeight(100)

self.feedbackBox = QtGui.QTextBrowser()
self.feedbackBox.setFont(font)
self.feedbackBox.setSizePolicy(sizePolicy)

#for the diagram
self.figure = Figure()
self.figureCanvas = FigureCanvas(self.figure)
self.figureToolBar = NavigationToolBar(self.figureCanvas, self)
self.figureCanvas.setFont(font)
self.figureCanvas.setSizePolicy(sizePolicy)

#for the image
self.label = QtGui.QLabel()
self.pixmap = QtGui.QPixmap()

#for the table
self.table = QtGui.QTableWidget()

# Define a widget for the 3D viewer
center = QtGui.QWidget()
center2 = QtGui.QWidget()
center3 = QtGui.QWidget()

# Define and set layout
mainLayout = QtGui.QHBoxLayout(center)
viewerFeedbackResults = QtGui.QVBoxLayout()
viewerEnergyResults = QtGui.QVBoxLayout()

#BUTTONS

self.openIfcButton = QtGui.QPushButton("Load IFC", self)
self.openIfcButton.clicked.connect(self.openIfcFile)
self.openIfcButton.setFixedHeight(20)

# self.refreshIfcButton = QtGui.QPushButton("Refresh IFC", self)
# self.refreshIfcButton.clicked.connect(self.refreshIfcFile)
# self.refreshIfcButton.setFixedHeight(20)

self.calculateEnergyConsumptionButton = QtGui.QPushButton("Calculate Energy Consumption", self)
self.calculateEnergyConsumptionButton.clicked.connect(self.calculateEnergyConsumption)
self.calculateEnergyConsumptionButton.setFixedHeight(20)

self.closeIfcButton = QtGui.QPushButton("Close IFC", self)
self.closeIfcButton.clicked.connect(self.closeIfcFile)

```

```

self.closeIfcButton.setFixedHeight(20)

self.viewEnergyResultsButton = QtGui.QPushButton("Energy Results", self)
self.viewEnergyResultsButton.clicked.connect(self.viewEnergyResults)
self.viewEnergyResultsButton.setFixedHeight(20)

self.buildingFunctionalityButton = QtGui.QComboBox ()
self.buildingFunctionalityButton.setEditable(True)
self.buildingFunctionalityButton.lineEdit().setAlignment(QtCore.Qt.AlignCenter)
self.buildingFunctionalityButton.setFixedHeight(20)
self.buildingFunctionalityButton.setStyleSheet("background-color: #f0f0f0")

self.buildingFunctionalityButton.addItem("Building Functionality")
self.buildingFunctionalityButton.addItem("Residence")
self.buildingFunctionalityButton.addItem("Hospital")
self.buildingFunctionalityButton.addItem("Office")
self.buildingFunctionalityButton.addItem("School")
self.buildingFunctionalityButton.addItem("Store")
self.buildingFunctionalityButton.addItem("Gym")
self.buildingFunctionalityButton.currentIndexChanged.connect(self.getBuildingFuctionality)

self.whatIfButton = QtGui.QComboBox()
self.whatIfButton.setEditable(True)
self.whatIfButton.lineEdit().setAlignment(QtCore.Qt.AlignCenter)
self.whatIfButton.setFixedHeight(20)
self.whatIfButton.setStyleSheet("background-color: #e3e3e3")

self.whatIfButton.addItem("What if...")
self.whatIfButton.addItem("Change wall area")
self.whatIfButton.addItem("Change opening area")
self.whatIfButton.addItem("Change orientation")
self.whatIfButton.currentIndexChanged.connect(self.whatIf)

splitterVer1 = QtGui.QSplitter(QtCore.Qt.Vertical)
splitterVer2 = QtGui.QSplitter(QtCore.Qt.Vertical)
splitterVer3 = QtGui.QSplitter(QtCore.Qt.Vertical)

splitterVer1.addWidget(self.table)
splitterVer1.addWidget(self.figureCanvas)
splitterVer1.addWidget(self.figureToolbar)

splitterVer2.addWidget(self.canvas)
splitterVer2.addWidget(self.openIfcButton)
splitterVer2.addWidget(self.buildingFunctionalityButton)
splitterVer2.addWidget(self.calculateEnergyConsumptionButton)
splitterVer2.addWidget(self.closeIfcButton)

#splitterVer3.addWidget(self.refreshIfcButton)
splitterVer3.addWidget(self.viewEnergyResultsButton)
splitterVer3.addWidget(self.feedbackBox)
splitterVer3.addWidget(self.label)
splitterVer3.addWidget(self.whatIfButton)
splitterVer3.addWidget(self.whatIfBox)

```

```

mainLayout.addWidget(splitterVer2)
secondLayout = QtGui.QVBoxLayout(center2)
secondLayout.addLayout(viewerFeedbackResults)
viewerFeedbackResults.addWidget(splitterVer3)
thirdLayout = QtGui.QVBoxLayout(center3)
thirdLayout.addLayout (viewerEnergyResults)
viewerEnergyResults.addWidget(splitterVer1)

self.viewerTab.setLayout(mainLayout)
self.PopUpDialog.setLayout(secondLayout)
self.ResultsDialog.setLayout(thirdLayout)

#to make buttons unavailable until IFC file is loaded

if not self.filename:
    self.calculateEnergyConsumptionButton.setEnabled(False)
    self.closeIfcButton.setEnabled(False)
    self.viewEnergyResultsButton.setEnabled(False)
    self.buildingFunctionalityButton.setEnabled(False)
    self.buildingFunctionalityButton.setStyleSheet("background-color: #f0f0f0")
    self.whatIfButton.setEnabled(False)
    self.whatIfButton.setStyleSheet("background-color: #f0f0f0")

self.count = 0
self.rdf_graph = None

#LOAD IFC
def openIfcFile(self):
    self.counter = 1
    self.filename = QtGui.QFileDialog.getOpenFileName(self, 'Open file', ".", "Industry Foundation Classes (*.ifc)")
    self.fs_watcher = QtCore.QFileSystemWatcher([self.filename])
    self.fs_watcher.connect(self.fs_watcher, QtCore.SIGNAL('fileChanged(const QString &)'), self.file_changed)

    if self.filename:
        self.parent.display.EraseAll()
        self.visualizationBox.clear()
        self.whatIfBox.clear()
        self.feedbackBox.clear()
        self.buildingFunctionalityButton.setEnabled(True)
        self.buildingFunctionalityButton.setStyleSheet("background-color: #e3e3e3")
        self.closeIfcButton.setEnabled(True)

#create two lists that will contain all energy results(1) and tip results(2) of specific file
self.energyResults = []
self.tipResults = []
self.parse_ifc(self.filename)

#WHAT IF AND THE SCENARIOS
def whatIf(self):

    self.whatIfBox.clear()
    if self.whatIfButton.currentText() == "What if..." :

```

```

        self.whatIfBox.append(r'Check "what if conditions" if you want to get results')
        self.whatIfBox.append("without modifying the 3D design")
    if self.whatIfButton.currentText() == "Change wall area":
        self.getWhatIfChangeWallArea()
    if self.whatIfButton.currentText() == "Change opening area":
        self.getWhatIfChangeOpeningArea()
    if self.whatIfButton.currentText() == "Change orientation":
        self.getWhatIfChangeOrientation()

def getWhatIfChangeWallArea(self):

    self.whatIfBox.append("Changes in wall area")

    for wall in self.wallList:
        wall.area = wall.area * 1.049
    for roof in self.roofList:
        roof.area = roof.area * 1.1
    for ground in self.groundList:
        ground.area = ground.area * 1.1
    self.totalExternalArea = self.totalExternalArea * 1.1
    self.getAllEnergyResults()
    self.tipResultsSaved()

    oldResults = self.tipResults[-2]
    newResults = self.tipResults[-1]

    differences = tuple(x - y for x, y in zip(newResults, oldResults))
    if differences[3] >= 0:
        self.whatIfBox.append((" -" + "External area increase by 10%:" + " " + "+" + str(differences[3] + "MJ")))
    else:
        self.whatIfBox.append((" -" + "External area increase by 10%:" + " " + str(differences[3] + "MJ")))

    for wall in self.wallList:
        wall.area = (wall.area / 1.049) * 0.9486
    for roof in self.roofList:
        roof.area = (roof.area / 1.1) * 0.9
    for ground in self.groundList:
        ground.area = (ground.area / 1.1) * 0.9
    self.totalExternalArea = (self.totalExternalArea / 1.1) * 0.9
    self.getAllEnergyResults()
    self.tipResultsSaved()

    oldResults = self.tipResults[-3]
    newResults = self.tipResults[-1]

    differences = tuple(x - y for x, y in zip(newResults, oldResults))
    if differences[3] >= 0:
        self.whatIfBox.append((" -" + "External area decrease by 10%:" + " " + "+" + str(differences[3] + "MJ")))
    else:
        self.whatIfBox.append((" -" + "External area decrease by 10%:" + " " + str(differences[3] + "MJ")))

    for wall in self.wallList:

```

```

        wall.area = wall.area / 0.9486
    for roof in self.roofList:
        roof.area = roof.area / 0.9
    for ground in self.groundList:
        ground.area = ground.area / 0.9
    self.totalExternalArea = self.totalExternalArea / 0.9

    del self.tipResults[1:]

def getWhatIfChangeOpeningArea(self):

    self.whatIfBox.append("Changes in opening area")

    for opening in self.openingList:
        opening.area = opening.area * 0.9
        unique = True
        for wall in self.wallList:
            if unique:
                if wall.orientation == opening.orientation:
                    wall.area = wall.area + ((opening.area / 0.9) - opening.area)
                    unique = False

    self.getAllEnergyResults()
    self.tipResultsSaved()

    oldResults = self.tipResults[-2]
    newResults = self.tipResults[-1]

    differences = tuple(x - y for x, y in zip(newResults, oldResults))
    if differences[3] >= 0:
        self.whatIfBox.append(
            (" -" + "External area of openings decrease by 10%:" + " " + " " + "+" + str(differences[3])))
    else:
        self.whatIfBox.append(
            (" -" + "External area of openings decrease by 10%:" + " " + " " + str(differences[3])))

    for opening in self.openingList:
        temporaryValue = opening.area
        opening.area = opening.area/0.9
        unique = True
        for wall in self.wallList:
            if unique:
                if wall.orientation == opening.orientation:
                    wall.area = wall.area -(opening.area - temporaryValue)
                    unique = False

    del self.tipResults[1:]

def getWhatIfChangeOrientation(self):

    self.whatIfBox.append ("Changes in orientation")

    for index in range(1, 4):

```

```

for wall in self.wallList:
    if wall.orientation == "North":
        wall.orientation = "East"
    elif wall.orientation == "East":
        wall.orientation = "South"
    elif wall.orientation == "South":
        wall.orientation = "West"
    elif wall.orientation == "West":
        wall.orientation = "North"
for opening in self.openingList:
    if opening.orientation == "North":
        opening.orientation = "East"
    elif opening.orientation == "East":
        opening.orientation = "South"
    elif opening.orientation == "South":
        opening.orientation = "West"
    elif opening.orientation == "West":
        opening.orientation = "North"

self.getAllEnergyResults()
self.tipResultsSaved()

oldResults = self.tipResults[(-1 - index)]
newResults = self.tipResults[-1]

if index == 1:
    rotation = "90 degrees:  "
elif index == 2:
    rotation = "180 degrees:  "
elif index == 3:
    rotation = "270 degrees:  "

differences = tuple(x - y for x, y in zip(newResults, oldResults))
if (differences[3] + differences[4]) <= 0:
    self.whatIfBox.append(
        (" -" + "Rotate building clockwise by" + " " + rotation + "+" + str(differences[3] + differences[4])))
else:
    self.whatIfBox.append(
        (" -" + "Rotate building clockwise by" + " " + rotation + "-" + str(differences[3] + differences[4])))

for wall in self.wallList:
    if wall.orientation == "North":
        wall.orientation = "East"
    elif wall.orientation == "East":
        wall.orientation = "South"
    elif wall.orientation == "South":
        wall.orientation = "West"
    elif wall.orientation == "West":
        wall.orientation = "North"
for opening in self.openingList:
    if opening.orientation == "North":

```

```

        opening.orientation = "East"
    elif opening.orientation == "East":
        opening.orientation = "South"
    elif opening.orientation == "South":
        opening.orientation = "West"
    elif opening.orientation == "West":
        opening.orientation = "North"

del self.tipResults[1:]

#BUILDING FUNCTIONALITY
def getBuildingFuctionality(self):

    # set building functionality
    self.buildingFunctionality = ""
    if self.buildingFunctionalityButton.currentText() != "Building Functionality":
        self.buildingFunctionality = str(self.buildingFunctionalityButton.currentText())
    else:
        self.buildingFunctionality == ""

    #enable other buttons
    if self.buildingFunctionality != "":
        self.calculateEnergyConsumptionButton.setEnabled(True)
        self.viewEnergyResultsButton.setEnabled(True)
        self.whatIfButton.setEnabled(True)
        self.whatIfButton.setStyleSheet("background-color: #e3e3e3")
    elif self.buildingFunctionality == "":
        self.calculateEnergyConsumptionButton.setEnabled(False)
        self.viewEnergyResultsButton.setEnabled(False)
        self.whatIfButton.setEnabled(False)
        self.whatIfButton.setStyleSheet("background-color: #f0f0f0")

    # persons per square meter by Bouwbesluit 2012 (tabel 1.2), ventilation demand by new buildings, electricity
    demand by table 16.2, total hours 16.1
    # set values according to functionality
    illuminationReductionFactor = 0.3
    # heat internal values by people
    self.timeCorrectionFactor = 0
    self.heatProductionPeople = 0
    # heat internal values by equipment
    self.heatProductionEquipment = 0
    #all 0 for case of residence
    self.ventilationDemand = 0
    self.electricityDemand = 0
    self.personPerSquareMeter = 0
    self.totalHours = 0
    self.insideTemperatureHeatingSeason = 20
    if self.buildingFunctionality == "Hospital":
        self.timeCorrectionFactor = 0.80
        self.heatProductionPeople = 5
        self.heatProductionEquipment = 4
        self.personPerSquareMeter = 0.125
        self.ventilationDemand = 12

```

```

        self.electricityDemand = 17 * illuminationReductionFactor
        self.totalHours = 4500
        self.insideTemperatureHeatingSeason = 22
    elif self.buildingFunctionality == "Office":
        self.timeCorrectionFactor = 0.30
        self.heatProductionPeople = 5
        self.heatProductionEquipment = 4
        self.personPerSquareMeter = 0.05
        self.ventilationDemand = 6.5
        self.electricityDemand = 16 * illuminationReductionFactor
        self.totalHours = 2350
        self.insideTemperatureHeatingSeason = 20
    elif self.buildingFunctionality == "School":
        self.timeCorrectionFactor = 0.30
        self.heatProductionPeople = 10
        self.heatProductionEquipment = 2
        self.personPerSquareMeter = 0.125
        self.ventilationDemand = 8.5
        self.electricityDemand = 16 * illuminationReductionFactor
        self.totalHours = 1750
        self.insideTemperatureHeatingSeason = 20
    elif self.buildingFunctionality == "Store":
        self.timeCorrectionFactor = 0.40
        self.heatProductionPeople = 3
        self.heatProductionEquipment = 3
        self.personPerSquareMeter = 0.03
        self.ventilationDemand = 4
        self.electricityDemand = 30 * illuminationReductionFactor
        self.totalHours = 2800
        self.insideTemperatureHeatingSeason = 20
    elif self.buildingFunctionality == "Gym":
        self.timeCorrectionFactor = 0.30
        self.heatProductionPeople = 3
        self.heatProductionEquipment = 1
        self.personPerSquareMeter = 0.05
        self.ventilationDemand = 6.5
        self.electricityDemand = 16 * illuminationReductionFactor
        self.totalHours = 2350
        self.insideTemperatureHeatingSeason = 13

#CLOSE IFC
def closeIcFile(self):
    self.filename = None

    self.parent.display.EraseAll()
    self.visualizationBox.clear()
    self.whatIfBox.clear()
    self.feedbackBox.clear()
    self.figure.clf()
    self.figureCanvas.draw()
    self.calculateEnergyConsumptionButton.setEnabled(False)
    self.closeIcFileButton.setEnabled(False)
    self.viewEnergyResultsButton.setEnabled(False)

```



```

self.buildingFunctionalityButton.setCurrentIndex(0)
self.buildingFunctionalityButton.setEnabled(False)
self.buildingFunctionalityButton.setStyleSheet("background-color: #f0f0f0")
self.whatIfButton.setCurrentIndex(0)
self.whatIfButton.setEnabled(False)
self.whatIfButton.setStyleSheet("background-color: #f0f0f0")

#REFRESH IFC
@QtCore.pyqtSlot(str)
def refreshIfcFile(self):
    time.sleep(1)
    if self.filename:
        self.counter = self.counter + 1
        print self.counter
        self.parent.display.EraseAll()
        self.visualizationBox.clear()
        self.feedbackBox.clear()
        self.whatIfBox.clear()
        self.figure.clf()
        self.tipResults = []
    self.parse_ifc(self.filename)
    if self.calculateEnergyConsumptionButton.isEnabled():
        self.calculateEnergyConsumption()

#PARSE IFC
def parse_ifc(self, filename):
    self.created_shapes = {}
    self.ifc_file = ifcopenshell.open(filename)
    rooms = self.ifc_file.by_type("IfcBuildingElement")
    for room in rooms:
        if room.Representation:
            ifcgeom = ifcopenshell.geom.create_shape(settings, room).geometry
            shp = self.canvas.Show(room.GlobalId, ifcgeom, None)
    print "IFC file successfully loaded!"

#CALCULATE ENERGY CONSUMPTION
#TOTALLY 3 STEPS

#STEP 1:GET MY OWN WALLS/OPENINGS/ROOF/GROUND AND DEFINE THE CHARACTERISTICS
def calculateEnergyConsumption(self):

    #Pop up window only once
    if not self.PopUpDialog.isVisible():
        self.PopUpDialog.show()
    self.whatIfButton.setCurrentIndex(0)

    self.whatIfBox.clear()
    self.whatIfBox.append(r'Check "what if conditions" if you want to get results')
    self.whatIfBox.append("without modifying the 3D design")
    #ENTIRE CALCULATIONS#

    def getWallForTotal(wall, areaOrVolume):
        forTotalVolume = 0

```

```

forTotalArea = ((wall.startPoint[0] * wall.endPoint[1]) - (wall.startPoint[1] * wall.endPoint[0]))

if areaOrVolume == "area":
    return (forTotalArea / 2)
elif areaOrVolume == "volume":
    return (forTotalVolume / 2)

def getWallForPart(wall):
    for connect in wall.connectedTo:
        if wall.connectedTo != []:
            wall.forPart.append((((connect[1] * wall.startPoint[0]) - (connect[0] * wall.startPoint[1])) / 2))

counter = 0
if len(wall.forPart) > 1:
    for index in range(1, len(wall.forPart)):
        wall.forPart.append((wall.forPart[index] - wall.forPart[(index - 1)]))
        counter = counter + 1
    for index in range(0, counter):
        wall.forPart.remove(wall.forPart[1])

wall.forPart.append(round(((wall.forTotal) - sum(wall.forPart)), 3))

class myWall(object):

    def __init__(self, globalId, internalOrExternal,
                 startPoint, endPoint, possibleStartPoint,
                 possibleEndPoint, axisDirection, length, width, height,
                 axisOrientation, orientation, area, connectedTo, forTotal, forPart,
                 thermalMass, thermalResistance, heatTransferCoefficient, absorptance,
                 thermalTransmittance, solarHeatGainCoefficient):
        self.globalId = globalId
        self.internalOrExternal = internalOrExternal
        self.startPoint = startPoint
        self.possibleStartPoint = possibleStartPoint
        self.possibleEndPoint = possibleEndPoint
        self.endPoint = endPoint
        self.axisDirection = axisDirection
        self.length = length
        self.width = width
        self.height = height
        self.axisOrientation = axisOrientation
        self.orientation = orientation
        self.area = area
        self.connectedTo = connectedTo
        self.forTotal = forTotal
        self.forPart = forPart
        self.thermalMass = thermalMass
        self.thermalResistance = thermalResistance
        self.heatTransferCoefficient = heatTransferCoefficient
        self.absorptance = absorptance
        self.thermalTransmittance = thermalTransmittance
        self.solarHeatGainCoefficient = solarHeatGainCoefficient

```

```

class myOpening(object):

    def __init__(self, globalId, doorOrWindow, internalOrExternal,length, height, area, orientation,
        thermalMass, thermalResistance, heatTransferCoefficient, absorptance,
        thermalTransmittance,solarHeatGainCoefficient):
        self.globalId = globalId
        self.doorOrWindow = doorOrWindow
        self.internalOrExternal = internalOrExternal
        self.length = length
        self.height = height
        self.area = area
        self.orientation = orientation
        self.thermalMass = thermalMass
        self.thermalResistance = thermalResistance
        self.heatTransferCoefficient = heatTransferCoefficient
        self.absorptance = absorptance
        self.thermalTransmittance = thermalTransmittance
        self.solarHeatGainCoefficient = solarHeatGainCoefficient

class myRoof (object):

    def __init__(self, globalId, area, thermalMass,thermalResistance, heatTransferCoefficient, absorptance,
thermalTransmittance,solarHeatGainCoefficient):
        self.globalId = globalId
        self.area = area
        self.thermalMass = thermalMass
        self.thermalResistance = thermalResistance
        self.heatTransferCoefficient = heatTransferCoefficient
        self.absorptance = absorptance
        self.thermalTransmittance = thermalTransmittance
        self.solarHeatGainCoefficient = solarHeatGainCoefficient

class myGround (object):

    def __init__(self, globalId, area, thermalMass,thermalResistance, heatTransferCoefficient,
absorptance,thermalTransmittance,solarHeatGainCoefficient):
        self.globalId = globalId
        self.area = area
        self.thermalMass = thermalMass
        self.thermalResistance = thermalResistance
        self.heatTransferCoefficient = heatTransferCoefficient
        self.absorptance = absorptance
        self.thermalTransmittance = thermalTransmittance
        self.solarHeatGainCoefficient = solarHeatGainCoefficient

# GENERAL FUNCTIONS #
def getNorthDirection(mainProject):

    northDirection = [0.0, 1.0]
    counter = 0
    for Project in mainProject:
        for RepresentationContext in Project.RepresentationContexts:
            if counter == 0:

```

```

        if RepresentationContext.is_a("IfcGeometricRepresentationContext"):
            if RepresentationContext.TrueNorth != None:
                northDirection[0] = round((RepresentationContext.TrueNorth.DirectionRatios[0]), 1)
                northDirection[1] = round((RepresentationContext.TrueNorth.DirectionRatios[1]), 1)
            counter = 1

    return (northDirection)

def getAllWidths(wallList):

    widths = []
    for wall in wallList:
        width = wall.width
        if width not in widths:
            widths.append(width)

    return (widths)

# FUNCTIONS FOR myWall CHARACTERISTICS #

def getWallAxisDirection(wall):

    wallAxisDirection = [1.0, 0.0]
    if wall.ObjectPlacement.is_a("IfcLocalPlacement"):
        if wall.ObjectPlacement.RelativePlacement.RefDirection != None:
            wallAxisDirection[0] = wall.ObjectPlacement.RelativePlacement.RefDirection.DirectionRatios[0]
            wallAxisDirection[1] = wall.ObjectPlacement.RelativePlacement.RefDirection.DirectionRatios[1]

    return (wallAxisDirection)

def getWallLengthWidthHeight(wall, lengthOrWidthOrHeightOrAreaOrVolume):
    width = 0
    length = 0
    height = 0
    volume = 0

    for relDefinesByProperties in wall.IsDefinedBy:
        if relDefinesByProperties.is_a("IfcRelDefinesByProperties"):
            if relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcElementQuantity"):
                for properties in relDefinesByProperties.RelatingPropertyDefinition.Quantities:
                    if properties.is_a("IfcQuantityLength"):
                        if properties.Name == "Width":
                            width = properties.LengthValue
                        if properties.Name == "Length":
                            length = properties.LengthValue
                        if properties.Name == "Height":
                            height = properties.LengthValue
                        if properties.Name == "Volume":
                            volume = properties.LengthValue

                    elif relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcPropertySet"):
                        for properties in relDefinesByProperties.RelatingPropertyDefinition.HasProperties:
                            if properties.Name == "Width":

```

```

        width = properties.NominalValue.wrappedValue
    if properties.Name == "Length":
        length = properties.NominalValue.wrappedValue
    if properties.Name == "Height":
        height = properties.NominalValue.wrappedValue
    if properties.Name == "Volume":
        volume = properties.NominalValue.wrappedValue

area = length * height

```

```

if lengthOrWidthOrHeightOrAreaOrVolume == "length":
    return (length)
elif lengthOrWidthOrHeightOrAreaOrVolume == "width":
    return (width)
elif lengthOrWidthOrHeightOrAreaOrVolume == "height":
    return (height)
elif lengthOrWidthOrHeightOrAreaOrVolume == "area":
    return (area)
elif lengthOrWidthOrHeightOrAreaOrVolume == "volume":
    return (volume)

```

```

def getThermalValues(element):

```

```

    thermalMass = 0
    thermalResistance = 0
    heatTransferCoefficient = 0
    absorptance = 0.8
    solarHeatGainCoefficient = 0
    for relDefinesByType in element.IsDefinedBy:
        if relDefinesByType.is_a("IfcRelDefinesByType"):
            if relDefinesByType.RelatingType.is_a("IfcTypeProduct"):
                for PropertySets in relDefinesByType.RelatingType.HasPropertySets:
                    if PropertySets.is_a("IfcPropertySet"):
                        if PropertySets.Name == "Analytical Properties":
                            for Properties in PropertySets.HasProperties:
                                if Properties.Name == "Heat Transfer Coefficient (U)":
                                    heatTransferCoefficient = Properties.NominalValue.wrappedValue
                                if Properties.Name == "Thermal mass":
                                    thermalMass = Properties.NominalValue.wrappedValue
                                if Properties.Name == "Thermal Resistance (R)":
                                    thermalResistance = Properties.NominalValue.wrappedValue
                                if Properties.Name == "Absorptance":
                                    absorptance = Properties.NominalValue.wrappedValue
                                if Properties.Name == "Solar Heat Gain Coefficient":
                                    solarHeatGainCoefficient = Properties.NominalValue.wrappedValue
            elif relDefinesByType.is_a("IfcRelDefinesByProperties"):
                if relDefinesByType.RelatingPropertyDefinition.is_a("IfcPropertySet"):
                    for Properties in relDefinesByType.RelatingPropertyDefinition.HasProperties:
                        if Properties.Name == "Heat Transfer Coefficient (U)":
                            heatTransferCoefficient = Properties.NominalValue.wrappedValue
                        if Properties.Name == "Thermal mass":
                            thermalMass = Properties.NominalValue.wrappedValue

```

```

        if Properties.Name == "Thermal Resistance (R)":
            thermalResistance = Properties.NominalValue.wrappedValue
        if Properties.Name == "Absorptance":
            absorptance = Properties.NominalValue.wrappedValue
        if Properties.Name == "Solar Heat Gain Coefficient":
            solarHeatGainCoefficient = Properties.NominalValue.wrappedValue
    if heatTransferCoefficient == 0:
        heatTransferCoefficient = 0.1428
    if thermalResistance == 0:
        thermalResistance = 1/0.1428

    return (thermalMass, thermalResistance, heatTransferCoefficient, absorptance, solarHeatGainCoefficient)

def getThermalTransmittance (opening):

    thermalTransmittance = 0

    for relDefinesByProperties in opening.IsDefinedBy:
        if relDefinesByProperties.is_a("IfcRelDefinesByProperties"):
            if relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcPropertySet"):
                for properties in relDefinesByProperties.RelatingPropertyDefinition.HasProperties:
                    if properties.Name == "ThermalTransmittance":
                        thermalTransmittance = properties.NominalValue.wrappedValue

    return (thermalTransmittance)

def checkIfElementIsExternal(wall):

    external = False
    for relDefinesByProperties in wall.IsDefinedBy:
        if relDefinesByProperties.is_a("IfcRelDefinesByProperties"):
            if relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcPropertySet"):
                for properties in relDefinesByProperties.RelatingPropertyDefinition.HasProperties:
                    if properties.Name == "IsExternal":
                        if properties.NominalValue.wrappedValue == True:
                            external = True
    return (external)

def getPossibleCoordinates(wall, allWidths):

    firstCoordinates = getFirstCoordinates(wall)
    isConnectedTo = getHowManyWallsIsConnectedTo(wall)
    for possible in allWidths:
        # first coordinates
        Case1x1 = round((firstCoordinates[0] + (wall.axisDirection[0] * possible / 2)), 10)
        Case1y1 = round((firstCoordinates[1] + (wall.axisDirection[1] * possible / 2)), 10)
        Case2x1 = round((firstCoordinates[0] - (wall.axisDirection[0] * possible / 2)), 10)
        Case2y1 = round((firstCoordinates[1] - (wall.axisDirection[1] * possible / 2)), 10)
        Case3x1 = round ((firstCoordinates[0]),10)
        Case3y1 = round((firstCoordinates[1]), 10)

```

```

wall.possibleStartPoint.append([Case1x1, Case1y1, firstCoordinates[2]])
wall.possibleStartPoint.append([Case2x1, Case2y1, firstCoordinates[2]])
wall.possibleStartPoint.append([Case3x1, Case3y1, firstCoordinates[2]])

# second get_coordinates
Case1x2 = round(
    (firstCoordinates[0] + (wall.axisDirection[0] * (wall.length + (isConnectedTo * possible / 2)))),
    10)
Case1y2 = round(
    (firstCoordinates[1] + (wall.axisDirection[1] * (wall.length + (isConnectedTo * possible / 2)))),
    10)
Case2x2 = round(
    (firstCoordinates[0] + (wall.axisDirection[0] * (wall.length - (isConnectedTo * possible / 2)))),
    10)
Case2y2 = round(
    (firstCoordinates[1] + (wall.axisDirection[1] * (wall.length - (isConnectedTo * possible / 2)))),
    10)
Case3x2 = round(
    (firstCoordinates[0] + (wall.axisDirection[0] * wall.length)),
    10)
Case3y2 = round(
    (firstCoordinates[1] + (wall.axisDirection[1] * wall.length)),
    10)

wall.possibleEndPoint.append([Case1x2, Case1y2, firstCoordinates[2]])
wall.possibleEndPoint.append([Case2x2, Case2y2, firstCoordinates[2]])
wall.possibleEndPoint.append([Case3x2, Case3y2, firstCoordinates[2]])

```

**def** getFirstCoordinates(wall):

```

unique = True
for wall2 in wallStandardCaseList:
    if unique:
        if wall.globalId == wall2.GlobalId:
            x1 = round((wall2.ObjectPlacement.RelativePlacement.Location.Coordinates[0]), 10)
            y1 = round((wall2.ObjectPlacement.RelativePlacement.Location.Coordinates[1]), 10)
            z1 = round(((wall2.ObjectPlacement.RelativePlacement.Location.Coordinates[
                2]) + getHeightAfterLevel(wall2)), 10)
            unique = False

firstCoordinates = [x1, y1, z1]

return (firstCoordinates)

```

**def** getHeightAfterLevel(wall):

```

level = 0
for relDefinesByProperties in wall.IsDefinedBy:
    if relDefinesByProperties.is_a("IfcRelDefinesByProperties"):
        if relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcPropertySet"):
            for properties in relDefinesByProperties.RelatingPropertyDefinition.HasProperties:
                if properties.Name == "Base Constraint":
                    baseConstraintName = properties.NominalValue.wrappedValue

```

```

        baseConstraintName = baseConstraintName.replace('Level: ', '')
        level = level + getLevelHeight(baseConstraintName)
    if properties.Name == "Base Offset":
        level = level + properties.NominalValue.wrappedValue

    return (level)

def getLevelHeight(baseConstraintName):

    buildingStoreyList = self.ifc_file.by_type("IfcBuildingStorey")
    for buildingStorey in buildingStoreyList:
        if baseConstraintName == buildingStorey.Name:
            baseConstraint = buildingStorey.Elevation

    return (baseConstraint)

def getHowManyWallsIsConnectedTo(wall):
    unique = True
    for wall2 in wallStandardCaseList:
        if unique:
            if wall.globalId == wall2.GlobalId:
                counter = 0
                for RelConnectsElements in wall2.ConnectedTo:
                    if RelConnectsElements.is_a("IfcRelConnectsPathElements"):
                        if RelConnectsElements.RelatedElement.is_a("IfcBuildingElement"):
                            if RelConnectsElements.RelatedElement.is_a("IfcWallStandardCase"):
                                if checkIfElementIsExternal(RelConnectsElements.RelatedElement):
                                    counter = counter + 1

                if counter == 2:
                    isConnectedTo = 1
                else:
                    isConnectedTo = -1
                unique = False

    return (isConnectedTo)

def getConnectedToInCorrectOrder(wallList):

    for wall in wallList:
        if len(wall.connectedTo) > 1:
            for index in range(0, (len(wall.connectedTo) - 1)):
                if abs(wall.connectedTo[index][0] - wall.startPoint[0]) >= \
                    abs(wall.connectedTo[index + 1][0] - wall.startPoint[0]) and \
                    abs(wall.connectedTo[index][1] - wall.startPoint[1]) >= \
                    abs(wall.connectedTo[index + 1][1] - wall.startPoint[1]):
                    (wall.connectedTo[index], wall.connectedTo[index + 1]) = (
                        wall.connectedTo[index + 1], wall.connectedTo[index])

# FOR EXTERNAL WALLS startPoint, endPoint, orientation, AND totalExternalArea #

def defineFinalCoordinates_defineAxisOrientation_getTotalExternalArea(checklist, totalExternalArea,
    totalExternalVolume):

```



```

finalWallList = [checklist[0]]
getFinalCoordinatesForExternalWall(finalWallList, externalWallList)
externalArea = 0
externalVolume = 0
for wall in finalWallList:
    externalArea = externalArea + getWallForTotal(wall, "area")
    externalVolume = externalVolume + getWallForTotal(wall, "volume")
orientation = externalArea
totalExternalArea = totalExternalArea + abs(externalArea)
totalExternalVolume = totalExternalVolume + abs(externalVolume)

getOrientation(finalWallList, orientation)

# to get for two different buildings

for wall in finalWallList:
    if wall in checklist:
        checklist.remove(wall)

if checklist != []:
    (totalExternalArea,
     totalExternalVolume) = defineFinalCoordinates_defineAxisOrientation_getTotalExternalArea(checklist,
                                                                                               totalExternalArea,
                                                                                               totalExternalVolume)

    return (round(totalExternalArea,2), round(totalExternalVolume,2))

def getFinalCoordinatesForExternalWall(finalWallList, externalWallList):

    for wall in finalWallList:
        wall.axisOrientation = wall.axisDirection
        unique = True
        for wall2 in externalWallList:
            if wall2 not in finalWallList:
                for index in range(0, len(wall.possibleEndPoint)):
                    if wall.startPoint in wall.possibleEndPoint:
                        (wall.possibleStartPoint, wall.possibleEndPoint) = (
                            wall.possibleEndPoint, wall.possibleStartPoint)
                        if unique:
                            wall.axisOrientation = [-wall.axisDirection[0], -wall.axisDirection[1]]
                            unique = False
                    else:
                        if unique:
                            wall.axisOrientation = wall.axisDirection
                            unique = False

                if wall.possibleEndPoint[index] in wall2.possibleStartPoint:
                    wall.endPoint = wall.possibleEndPoint[index]
                    wall2.startPoint = wall.possibleEndPoint[index]
                    finalWallList.append(wall2)

                elif wall.possibleEndPoint[index] in wall2.possibleEndPoint:

```

```

        wall.endPoint = wall.possibleEndPoint[index]
        wall2.startPoint = wall.possibleEndPoint[index]
        finalWallList.append(wall2)

for wall in finalWallList:
    if wall.startPoint == []:
        for wall2 in finalWallList:
            for index in range(0, len(wall.possibleStartPoint)):
                if wall.possibleStartPoint[index] in wall2.possibleEndPoint:
                    wall.startPoint = wall.possibleStartPoint[index]

    if wall.endPoint == []:
        for wall2 in finalWallList:
            for index in range(0, len(wall.possibleEndPoint)):
                if wall.possibleEndPoint[index] in wall2.possibleStartPoint:
                    wall.endPoint = wall.possibleEndPoint[index]

def getOrientation(finalWallList, orientation):

    if orientation < 0:
        for wall in finalWallList:
            wall.axisOrientation = [-wall.axisOrientation[0], -wall.axisOrientation[1]]

def getExternalWallOrientation(northDirection, wall):

    if wall.axisOrientation == [1, 0]:
        wall.orientation = "South"
    elif wall.axisOrientation == [-1, 0]:
        wall.orientation = "North"
    elif wall.axisOrientation == [0, 1]:
        wall.orientation = "East"
    elif wall.axisOrientation == [0, -1]:
        wall.orientation = "West"

# FOR INTERNAL WALLS startPoint, endPoint, orientation #

def getFinalCoordinatesForInternalWall(internalWallList, externalWallList):

    # get internal with external

    for wall in internalWallList:
        for wall2 in externalWallList:

            for index in range(0, len(wall.possibleStartPoint)):
                if abs(wall2.startPoint[0] - wall.possibleStartPoint[index][0]) <= abs(
                    wall2.startPoint[0] - wall2.endPoint[0]) and abs(
                    wall2.startPoint[1] - wall.possibleStartPoint[index][1]) <= abs(
                    wall2.startPoint[1] - wall2.endPoint[1]):
                    wall.startPoint = wall.possibleStartPoint[index]
                    wall2.connectedTo.append(wall.possibleStartPoint[index])
            for index in range(0, len(wall.possibleEndPoint)):
                if abs(wall2.startPoint[0] - wall.possibleEndPoint[index][0]) <= abs(
                    wall2.startPoint[0] - wall2.endPoint[0]) and abs(

```

```

        wall2.startPoint[1] - wall.possibleEndPoint[index][1]) <= abs(
        wall2.startPoint[1] - wall2.endPoint[1]):
    wall.endPoint = wall.possibleEndPoint[index]
    wall2.connectedTo.append(wall.possibleEndPoint[index])

```

*# get internal with internal at edges*

```

for wall in internalWallList:
    if wall.startPoint in wall.possibleEndPoint:
        (wall.possibleStartPoint, wall.possibleEndPoint) = (wall.possibleEndPoint, wall.possibleStartPoint)
    for wall2 in internalWallList:
        if wall != wall2:
            if wall.endPoint == []:
                for index in range(0, len(wall.possibleEndPoint)):
                    if wall.possibleEndPoint[index] in wall2.possibleStartPoint:
                        wall.endPoint = wall.possibleEndPoint[index]
                        wall2.startPoint = wall.possibleEndPoint[index]
                    elif wall.possibleEndPoint[index] in wall2.possibleEndPoint:
                        wall.endPoint = wall.possibleEndPoint[index]
                        wall2.endPoint = wall.possibleEndPoint[index]
            if wall.startPoint == []:
                for index in range(0, len(wall.possibleStartPoint)):
                    if wall.possibleStartPoint[index] in wall2.possibleStartPoint:
                        wall.startPoint = wall.possibleStartPoint[index]
                        wall2.startPoint = wall.possibleStartPoint[index]
                    elif wall.possibleStartPoint[index] in wall2.possibleEndPoint:
                        wall.startPoint = wall.possibleStartPoint[index]
                        wall2.endPoint = wall.possibleStartPoint[index]

```

*# get internal with internal at midway*

```

def getLastPart(self):
    for wall in internalWallList:
        for wall2 in internalWallList:
            if wall2.startPoint != [] and wall2.endPoint != []:
                if wall != wall2:
                    for index in range(0, len(wall.possibleStartPoint)):
                        if abs(wall2.startPoint[0] - wall.possibleStartPoint[index][0]) <= abs(
                            wall2.startPoint[0] - wall2.endPoint[0]) and abs(
                            wall2.startPoint[1] - wall.possibleStartPoint[index][1]) <= abs(
                            wall2.startPoint[1] - wall2.endPoint[1]):
                            wall.startPoint = wall.possibleStartPoint[index]
                            wall2.connectedTo.append(wall.possibleStartPoint[index])
                    for index in range(0, len(wall.possibleEndPoint)):
                        if abs(wall2.startPoint[0] - wall.possibleEndPoint[index][0]) <= abs(
                            wall2.startPoint[0] - wall2.endPoint[0]) and abs(
                            wall2.startPoint[1] - wall.possibleEndPoint[index][1]) <= abs(
                            wall2.startPoint[1] - wall2.endPoint[1]):
                            wall.endPoint = wall.possibleEndPoint[index]
                            wall2.connectedTo.append(wall.possibleEndPoint[index])

    for wall in internalWallList:
        if wall.startPoint == [] or wall.endPoint == []:
            getLastPart(internalWallList)

```

```

getLastPart(internalWallList)

# remove unnecessary connectedTo from internal and duplicates
for wall in internalWallList:
    wall.connectedTo = sorted(wall.connectedTo)
    wall.connectedTo = [wall.connectedTo[i] for i in range(len(wall.connectedTo)) if
        i == 0 or wall.connectedTo[i] != wall.connectedTo[i - 1]]
    if wall.connectedTo != []:
        for connect in wall.connectedTo:
            if connect == wall.startPoint or connect == wall.endPoint:
                wall.connectedTo.remove(connect)

for wall in internalWallList:
    getWallForTotal(wall, "area")
for wall in self.wallList:
    getWallForPart(wall)

def getInternalWallAxisDirection(internalWallList):

    for wall in internalWallList:
        unique = False
        if wall.axisDirection[0] < 0:
            wall.axisDirection[0] = abs(wall.axisDirection[0])
            unique = True
        if wall.axisDirection[1] < 0:
            wall.axisDirection[1] = abs(wall.axisDirection[1])
            unique = True
        if unique:
            (wall.startPoint, wall.endPoint) = (wall.endPoint, wall.startPoint)

def getAverageValues(ElementList):

    thermalMassAverage = 0
    thermalResistanceAverage = 0
    heatTransferCoefficientAverage = 0
    absorptanceAverage = 0
    thermalTransmittanceAverage = 0
    counter = 0
    for wall in ElementList:
        thermalMassAverage = thermalMassAverage + wall.thermalMass
        thermalResistanceAverage = thermalResistanceAverage + wall.thermalResistance
        heatTransferCoefficientAverage = heatTransferCoefficientAverage + wall.heatTransferCoefficient
        absorptanceAverage = absorptanceAverage + wall.absorptance
        thermalTransmittanceAverage = thermalTransmittance + wall.thermalTransmittance
        counter = counter + 1

    if counter != 0:
        thermalMassAverage = thermalMassAverage / counter
        thermalResistanceAverage = thermalResistanceAverage / counter
        heatTransferCoefficientAverage = heatTransferCoefficientAverage / counter
        absorptanceAverage = absorptanceAverage / counter
        thermalTransmittanceAverage = thermalTransmittanceAverage / counter

```

```

    return
(thermalMassAverage,thermalResistanceAverage,heatTransferCoefficientAverage,absorptanceAverage,thermalTransmittanceAverage)

```

```

# FOR ROOFS #

```

```

def getAreaRoof (roof):

```

```

    area = 0
    for relDefinesByProperties in roof.IsDefinedBy:
        if relDefinesByProperties.is_a("IfcRelDefinesByProperties"):
            if relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcPropertySet"):
                for Properties in relDefinesByProperties.RelatingPropertyDefinition.HasProperties:
                    if Properties.Name == "Area":
                        area = Properties.NominalValue.wrappedValue
                    elif Properties.Name == "TotalArea":
                        area = Properties.NominalValue.wrappedValue
                    elif Properties.Name == "Gross Area":
                        area = Properties.NominalValue.wrappedValue

    return (area)

```

```

# For GROUND FLOOR #

```

```

def getFloorArea (slab):

```

```

    area = 0
    for relDefinesByProperties in slab.IsDefinedBy:
        if relDefinesByProperties.is_a("IfcRelDefinesByProperties"):
            if relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcPropertySet"):
                for Properties in relDefinesByProperties.RelatingPropertyDefinition.HasProperties:
                    if Properties.Name == "TotalArea":
                        area = Properties.NominalValue.wrappedValue
                    elif Properties.Name == "Area":
                        area = Properties.NominalValue.wrappedValue

    return (area)

```

```

def getFloorLevel (slab):

```

```

    level = 0
    for decomposes in slab.Decomposes:
        for relDefinesByProperties in decomposes.RelatingObject.IsDefinedBy:
            if relDefinesByProperties.is_a("IfcRelDefinesByProperties"):
                if relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcPropertySet"):
                    for properties in relDefinesByProperties.RelatingPropertyDefinition.HasProperties:
                        if properties.Name == "Level" or properties.Name == "Base Level":
                            levelName = properties.NominalValue.wrappedValue
                            levelName = levelName.replace('Level: ', '')
                            level = level + getLevelHeight(levelName)
                        if properties.Name == "Base Offset From Level" or properties.Name == "Height Offset From

```

```

Level":
    level = level + properties.NominalValue.wrappedValue

else:
    for relDefinesByProperties in slab.IsDefinedBy:
        if relDefinesByProperties.is_a("IfcRelDefinesByProperties"):
            if relDefinesByProperties.RelatingPropertyDefinition.is_a("IfcPropertySet"):
                for properties in relDefinesByProperties.RelatingPropertyDefinition.HasProperties:
                    if properties.Name == "Level" or properties.Name == "Base Level":
                        levelName = properties.NominalValue.wrappedValue
                        levelName = levelName.replace('Level: ', '')
                        level = level + getLevelHeight(levelName)
                    if properties.Name == "Base Offset From Level" or properties.Name == "Height Offset From
Level":
                        level = level + properties.NominalValue.wrappedValue

    return (level)
# FOR OPENINGS #

def getOpeningOrientation(opening):

    openingOrientation = []
    for relFillsElement in opening.FillsVoids:
        for relVoidsElement in relFillsElement.RelatingOpeningElement.VoidsElements:
            if relVoidsElement.RelatingBuildingElement.is_a("IfcBuildingElement"):
                if relVoidsElement.RelatingBuildingElement.is_a("IfcWallStandardCase"):
                    wallId = relVoidsElement.RelatingBuildingElement.GlobalId
                    unique = True
                    for wall in externalWallList:
                        if unique:
                            if wallId == wall.globalId:
                                openingOrientation = wall.orientation
                                wall.area = wall.area - (opening.OverallWidth * opening.OverallHeight)
                                unique = False

    return (openingOrientation)

def checkIfOpeningIsExternal(opening):

    openingLocation = "internal"
    for relFillsElement in opening.FillsVoids:
        for relVoidsElement in relFillsElement.RelatingOpeningElement.VoidsElements:
            if relVoidsElement.RelatingBuildingElement.is_a("IfcBuildingElement"):
                if relVoidsElement.RelatingBuildingElement.is_a("IfcWallStandardCase"):
                    wallId = relVoidsElement.RelatingBuildingElement.GlobalId
                    unique = True
                    for wall in self.wallList:
                        if unique:
                            if wallId == wall.globalId:
                                openingLocation = wall.internalOrExternal
                                unique = False
    return (openingLocation)

```

```

def getOpeningLengthOrHeightOrArea (opening,lengthOrWidthOrHeightOrAreaOrVolume ):

    length= opening.OverallWidth
    height= opening.OverallHeight
    area= length * height

    if lengthOrWidthOrHeightOrAreaOrVolume == "length":
        return (length)
    if lengthOrWidthOrHeightOrAreaOrVolume == "height":
        return (height)
    if lengthOrWidthOrHeightOrAreaOrVolume == "area":
        return (area)

# in order to find northDirection
Main_Project = self.ifc_file.by_type("IfcProject")
northDirection = getNorthDirection(Main_Project)
print northDirection

# in order to create the main list of walls, and the main list of openings
wallStandardCaseList = self.ifc_file.by_type("IfcWall")
openingsList = self.ifc_file.by_type("IfcWindow") + self.ifc_file.by_type("IfcDoor")
roofsList = self.ifc_file.by_type("IfcRoof") + self.ifc_file.by_type("IfcPlate")
slabsList = self.ifc_file.by_type("IfcSlab")

self.wallList = []
for wall in wallStandardCaseList:
    internalOrExternal = "internal"
    axisDirection = getWallAxisDirection(wall)
    length = getWallLengthWidthHeight(wall, "length")
    width = getWallLengthWidthHeight(wall, "width")
    height = getWallLengthWidthHeight(wall, "height")
    area = getWallLengthWidthHeight(wall, "area")
    (thermalMass, thermalResistance, heatTransferCoefficient, absorptance,solarHeatGainCoefficient) =
getThermalValues(wall)
    thermalTransmittance = getThermalTransmittance(wall)
    forTotal = 0
    if checkIfElementIsExternal(wall):
        internalOrExternal = "external"
    temporaryWall = myWall \
    (wall.GlobalId, internalOrExternal, [], [], [], [],
    axisDirection, length, width, height, [], [], area, [], forTotal, [],
    thermalMass, thermalResistance, heatTransferCoefficient, absorptance,
    thermalTransmittance,solarHeatGainCoefficient)
    self.wallList.append(temporaryWall)

allWidths = getAllWidths(self.wallList)

for wall in self.wallList:
    getPossibleCoordinates(wall, allWidths)

# in order to create lists for external and internal myWall
externalWallList = []

```

```

internalWallList = []
for wall in self.wallList:
    if wall.internalOrExternal == "external":
        externalWallList.append(wall)

for wall in self.wallList:
    if wall.internalOrExternal == "internal":
        internalWallList.append(wall)

# in order to create the checklist and it does not affect the externalWallList
checklist = []

for index, wall in enumerate(externalWallList):
    checklist.append(externalWallList[index])
print externalWallList
print checklist
# in order to take coordinates and external area
self.totalExternalArea = 0
self.totalExternalVolume = 0
(self.totalExternalArea, self.totalExternalVolume) =
defineFinalCoordinates_defineAxisOrientation_getTotalExternalArea(
    checklist, self.totalExternalArea, self.totalExternalVolume)

# in order to take wall orientation
for wall in externalWallList:
    getExternalWallOrientation(northDirection, wall)

getFinalCoordinatesForInternalWall(internalWallList, externalWallList)
getInternalWallAxisDirection(internalWallList)
getConnectedToInCorrectOrder(self.wallList)

#in Order to take average thermal values

(self.thermalMassAverageWall, self.thermalResistanceAverageWall, self.heatTransferCoefficientAverageWall,
self.absorptanceAverageWall, self.thermalTransmittanceAverageWall) = getAverageValues(externalWallList)

# in Order to take wallsArea per orientation

self.northWallArea = 0
self.southWallArea = 0
self.eastWallArea = 0
self.westWallArea = 0

for wall in externalWallList:
    if wall.orientation == "North":
        self.northWallArea = self.northWallArea + wall.area
    if wall.orientation == "South":
        self.southWallArea = self.southWallArea + wall.area
    if wall.orientation == "East":
        self.eastWallArea = self.eastWallArea + wall.area
    if wall.orientation == "West":
        self.westWallArea = self.westWallArea + wall.area

```



```

#for the roof
self.roofList = []
if roofsList == []:
    globalId = 0
    area = self.totalExternalArea
    (thermalMass, thermalResistance, heatTransferCoefficient, absorptance, thermalTransmittance) = \
    (self.thermalMassAverageWall, self.thermalResistanceAverageWall,
self.heatTransferCoefficientAverageWall,
    self.absorptanceAverageWall, self.thermalTransmittanceAverageWall)
    solarHeatGainCoefficient = 0
    temporaryRoof = myRoof \
    (globalId, area, thermalMass, thermalResistance, heatTransferCoefficient,
absorptance, thermalTransmittance, solarHeatGainCoefficient)
    self.roofList.append(temporaryRoof)
else:
    for roof in roofsList:
        area = getAreaRoof(roof)
        (thermalMass, thermalResistance, heatTransferCoefficient, absorptance, solarHeatGainCoefficient) =
getThermalValues(roof)
        thermalTransmittance = getThermalTransmittance(roof)
        if thermalMass == 0:
            thermalMass = self.thermalMassAverageWall
        if thermalResistance == 0:
            thermalResistance = self.thermalResistanceAverageWall
        if heatTransferCoefficient == 0:
            heatTransferCoefficient = self.heatTransferCoefficientAverageWall
        if absorptance == 0:
            absorptance = self.absorptanceAverageWall
        if thermalTransmittance == 0:
            thermalTransmittance = self.thermalResistanceAverageWall

        temporaryRoof = myRoof \
        (roof.GlobalId, area, thermalMass, thermalResistance, heatTransferCoefficient, absorptance,
thermalTransmittance, solarHeatGainCoefficient)
        self.roofList.append(temporaryRoof)

(self.thermalMassAverageRoof, self.thermalResistanceAverageRoof, self.heatTransferCoefficientAverageRoof,
self.absorptanceAverageRoof, self.thermalTransmittanceAverageRoof) = getAverageValues(self.roofList)

#for the ground
self.groundList = []
if len(slabsList) == len(roofsList):
    for roof in self.roofList:
        area = roof.area
        globalId = roof.globalId
        thermalMass = roof.thermalMass
        thermalResistance = roof.thermalResistance
        heatTransferCoefficient = roof.heatTransferCoefficient
        absorptance = roof.absorptance
        thermalTransmittance = roof.thermalTransmittance
        solarHeatGainCoefficient = 0
        temporaryGround = myGround \
        (globalId, area, thermalMass, thermalResistance, heatTransferCoefficient, absorptance,

```

```

        thermalTransmittance,solarHeatGainCoefficient)
    self.groundList.append(temporaryGround)
else:
    levels = []
    for slab in slabsList:
        level = getFloorLevel(slab)
        levels.append(level)
    for slab in slabsList:
        level = getFloorLevel(slab)
        if level == min(levels):
            area = getFloorArea (slab)
            (thermalMass, thermalResistance, heatTransferCoefficient, absorptance,solarHeatGainCoefficient) =
getThermalValues(slab)
            thermalTransmittance = getThermalTransmittance(slab)
            if thermalMass == 0:
                thermalMass = self.thermalMassAverageWall
            if thermalResistance == 0:
                thermalResistance = self.thermalResistanceAverageWall
            if heatTransferCoefficient == 0:
                heatTransferCoefficient = self.heatTransferCoefficientAverageWall
            if absorptance == 0:
                absorptance = self.absorptanceAverageWall
            if thermalTransmittance == 0:
                thermalTransmittance = self.thermalResistanceAverageWall
            temporaryGround = myGround \
                (slab.GlobalId, area, thermalMass, thermalResistance, heatTransferCoefficient, absorptance,
                thermalTransmittance,solarHeatGainCoefficient)
            self.groundList.append(temporaryGround)

    (self.thermalMassAverageGround, self.thermalResistanceAverageGround,
self.heatTransferCoefficientAverageGround,
    self.absorptanceAverageGround, self.thermalTransmittanceAverageGround) =
getAverageValues(self.groundList)

    #for the openings
    self.openingList = []
    for opening in openingsList:
        doorOrWindow = "door"
        if opening.is_a("IfcWindow"):
            doorOrWindow = "window"
        length = getOpeningLengthOrHeightOrArea(opening, "length")
        height = getOpeningLengthOrHeightOrArea(opening, "height")
        area = getOpeningLengthOrHeightOrArea(opening, "area")
        orientation = getOpeningOrientation(opening)
        internalOrExternal = checkIfOpeningIsExternal(opening)
        (thermalMass, thermalResistance, heatTransferCoefficient, absorptance,solarHeatGainCoefficient) =
getThermalValues(opening)
        thermalTransmittance = getThermalTransmittance(opening)
        temporaryOpening = myOpening(opening.GlobalId, doorOrWindow, internalOrExternal,length, height, area,
orientation,
            thermalMass, thermalResistance, heatTransferCoefficient, absorptance,
            thermalTransmittance,solarHeatGainCoefficient)
        self.openingList.append(temporaryOpening)

```

```

externalOpeningList = []
for opening in self.openingList:
    if opening.internalOrExternal == "external":
        externalOpeningList.append(opening)

(self.thermalMassAverageOpening, self.thermalResistanceAverageOpening,
self.heatTransferCoefficientAverageOpening,
self.absorptanceAverageOpening, self.thermalTransmittanceAverageOpening) = getAverageValues(
    externalOpeningList)

# in Order to take wallsArea per orientation

self.northOpeningArea = 0
self.southOpeningArea = 0
self.eastOpeningArea = 0
self.westOpeningArea = 0

for opening in externalOpeningList:
    if opening.orientation == "North":
        self.northOpeningArea = self.northOpeningArea + opening.area
    if opening.orientation == "South":
        self.southOpeningArea = self.southOpeningArea + opening.area
    if opening.orientation == "East":
        self.eastOpeningArea = self.eastOpeningArea + opening.area
    if opening.orientation == "West":
        self.westOpeningArea = self.westOpeningArea + opening.area

# CALCULATION PART 2
self.getAllEnergyResults()
# CALCULATION PART 3
self.lastStep()

#take by temperature difference

#STEP2:CALCULATE ENERGY CONSUMPTIONS
def getAllEnergyResults(self):

    #SET TEMPERATURES

    temperatures = []
    data = pd.read_csv(r"C:\Users\Miltos\Desktop\energy_performance_viewer\app_env\Coppy of NEN5060-
A2.csv")
    for dat in data["T"].values:
        temperatures.append(round((dat * 0.1), 1))

    # TAKE HEAT TRANSFER

    wallHeatTransfer = 0
    openingHeatTransfer = 0
    ventilationHeatTransfer = 0
    densityOfAir = 1.205

```

```

heatCapacityOfAir = 1008
infiltrationRate = 0.43
ventilationHeatTransferCoefficient = (self.totalExternalArea * infiltrationRate * densityOfAir *
heatCapacityOfAir) / 1000
self.wallCalculationPerMonth = []
self.windowCalculationPerMonth = []
self.ventilationCalculationPerMonth = []
self.wallTotalHeatTransfer = 0
self.openingTotalHeatTransfer = 0
self.roofTotalHeatTransfer = 0
self.groundTotalHeatTransfer = 0

#create each month results
for index, temp in enumerate(temperatures):
    if index > 1:
        if data["M"].values[index] != data["M"].values[index - 1]:
            self.wallCalculationPerMonth.append(round(wallHeatTransfer, 1))
            self.windowCalculationPerMonth.append(round(openingHeatTransfer, 1))
            self.ventilationCalculationPerMonth.append(round(ventilationHeatTransfer, 1))
            wallHeatTransfer = 0
            openingHeatTransfer = 0
            ventilationHeatTransfer = 0

#find heating or cooling inside temperature
if data["M"].values[index] >= 5 and data["M"].values[index] <= 9:
    insideTemperature = 20
else:
    insideTemperature = self.insideTemperatureHeatingSeason
#for ventilation
ventilationHeatTransferCalculation = 0.0036 * ventilationHeatTransferCoefficient * (temp -
insideTemperature)
ventilationHeatTransfer = ventilationHeatTransfer + ventilationHeatTransferCalculation

#for the walls
for wall in self.wallList:
    heatTransferCalculation = (0.0036 * (wall.area * (temp - insideTemperature))) / wall.thermalResistance
    wallHeatTransfer = wallHeatTransfer + heatTransferCalculation
    self.wallTotalHeatTransfer = self.wallTotalHeatTransfer + heatTransferCalculation
#for the roof
for roof in self.roofList:
    heatTransferCalculation = (0.0036 * (roof.area * (temp - insideTemperature))) / roof.thermalResistance
    wallHeatTransfer = wallHeatTransfer + heatTransferCalculation
    self.roofTotalHeatTransfer = self.roofTotalHeatTransfer + heatTransferCalculation
#for the ground
for ground in self.groundList:
    soilTemperature = 14
    heatTransferCalculation = (0.0036 * (ground.area * (soilTemperature - insideTemperature))) /
ground.thermalResistance
    wallHeatTransfer = wallHeatTransfer + heatTransferCalculation
    self.groundTotalHeatTransfer = self.groundTotalHeatTransfer + heatTransferCalculation
#for the widnows and doors
for opening in self.openingList:
    heatTransferCalculation = 0.0036 * opening.area * (temp - insideTemperature) *

```

```

opening.heatTransferCoefficient
    openingHeatTransfer = openingHeatTransfer + heatTransferCalculation
    self.openingTotalHeatTransfer = self.openingTotalHeatTransfer + heatTransferCalculation

#create last month result
self.wallCalculationPerMonth.append(round(wallHeatTransfer, 1))
self.windowCalculationPerMonth.append(round(openingHeatTransfer, 1))
self.ventilationCalculationPerMonth.append(round(ventilationHeatTransfer,1))

#TAKE INTERNAL HEAT

internalHeatFactorLighting = 3.6 * ((self.electricityDemand * self.totalExternalArea * self.totalHours/1000) +
(3*self.totalExternalArea))
internalHeatFactorLightingPerMonth = internalHeatFactorLighting / 12

if self.buildingFunctionality == "Residence":
    internalHeatFactorTotal = 0.0036*(230 + 1.8* self.totalExternalArea)
else:
    internalHeatFactorPeople =0.0036* self.heatProductionPeople * self.timeCorrectionFactor *
self.totalExternalArea
    internalHeatFactorEquipment = 0.0036 * self.heatProductionEquipment * self.totalExternalArea

    internalHeatFactorTotal = internalHeatFactorEquipment + internalHeatFactorPeople

self.internalHeatPerMonth = []
hoursPerMonth = 0
self.hoursPerMonthList = []
for index, temp in enumerate(temperatures):

    if index > 1:
        if data["M"].values[index] != data["M"].values[index - 1]:
            self.hoursPerMonthList.append(hoursPerMonth)
            hoursPerMonth = 0
            hoursPerMonth = hoursPerMonth + 1

self.hoursPerMonthList.append(hoursPerMonth)
for month in self.hoursPerMonthList:
    self.internalHeatPerMonth.append((round(month * internalHeatFactorTotal,1)) +
internalHeatFactorLightingPerMonth)

#TAKE SOLAR RADIATION

#set standard values
heatTransferResistanceOutside = 0.04
heatTransferCoefficientOutside = 5*0.90
timeAverageTemperatureDifference = 11

#set standard lists according to dutch legislation
roofIncidentSolarRadiationPerMonth = [26.8, 49.4, 79.6, 164.1, 212.3, 225.2, 199.1, 184.9, 117.5, 72.7, 32.6,
20.9]
basementIncidentSolarRadiationPerMonth = [5.8, 10.1, 16.2, 33.2, 42.7, 45.4, 40.1, 37.3, 23.8, 14.8, 7, 4.6 ]
northIncidentSolarRadiationPerMonth = [10.5, 18.8, 30.1, 52.6, 68.6, 81.6, 70.4, 60.8, 40.3, 25.4, 12.9, 8.3]
eastIncidentSolarRadiationPerMonth = [19.6, 37.4, 50.5, 112.1, 122, 130.6, 121.8, 121.5, 79.3, 51, 23.4, 16.1]

```

```

southIncidentSolarRadiationPerMonth = [56.1, 68.5, 82.9, 140.2, 134.6, 123.4, 119.2, 135.5, 115.1, 101.6, 56.4,
47.2]
westIncidentSolarRadiationPerMonth = [22.3, 32.6, 51.6, 109.5, 132.8, 143.6, 112.5, 109.2, 73.5, 49, 23.5,
15.4]

roofShadowReductionFactor = 1
basementShadowReductionFactor = 1
northShadowReductionFactor = 1
eastShadowReductionFactor = 0.85
southShadowReductionFactor = 0.9
westShadowReductionFactor = 0.85

roofFormFactor = 1
basementFormFactor = 0
verticalFormFactor = 0.5

self.wallSolarRadiationPerMonth = []
solarRadiationPerMonth = 0
self.northSolarRadiationCalculation = 0
self.southSolarRadiationCalculation = 0
self.westSolarRadiationCalculation = 0
self.eastSolarRadiationCalculation = 0
self.roofSolarRadiationCalculation = 0
self.groundSolarRadiationCalculation = 0
self.northOpeningSolarRadiationCalculation = 0
self.southOpeningSolarRadiationCalculation = 0
self.westOpeningSolarRadiationCalculation = 0
self.eastOpeningSolarRadiationCalculation = 0
for index, temp in enumerate(temperatures):
    if index > 1:
        if data["M"].values[index] != data["M"].values[index - 1]:
            self.wallSolarRadiationPerMonth.append(round((solarRadiationPerMonth*0.0036), 1))
            solarRadiationPerMonth = 0
#for the walls

    for wall in self.wallList:
        if wall.orientation == "North":
            shadowReductionFactor = northShadowReductionFactor
            incidentSolarRadiationPerMonth = northIncidentSolarRadiationPerMonth
            wallCollectorArea = wall. absorptance * heatTransferResistanceOutside * wall.heatTransferCoefficient *
wall.area
            heatFlow = heatTransferResistanceOutside * wall.heatTransferCoefficient * wall.area *
heatTransferCoefficientOutside * timeAverageTemperatureDifference
            solarRadiationCalculation = (shadowReductionFactor * wallCollectorArea *
            incidentSolarRadiationPerMonth[(data["M"].values[index] - 1)]) - (
                verticalFormFactor * heatFlow)
            self.northSolarRadiationCalculation = self.northSolarRadiationCalculation + solarRadiationCalculation
            solarRadiationPerMonth = solarRadiationPerMonth + solarRadiationCalculation
        elif wall.orientation == "East":
            shadowReductionFactor = eastShadowReductionFactor
            incidentSolarRadiationPerMonth = eastIncidentSolarRadiationPerMonth
            wallCollectorArea = wall. absorptance * heatTransferResistanceOutside * wall.heatTransferCoefficient *
wall.area

```

```

        heatFlow = heatTransferResistanceOutside * wall.heatTransferCoefficient * wall.area *
heatTransferCoefficientOutside * timeAverageTemperatureDifference
        solarRadiationCalculation = (shadowReductionFactor * wallCollectorArea *
            incidentSolarRadiationPerMonth[(data["M"].values[index] - 1)]) - (
                verticalFormFactor * heatFlow)
        self.eastSolarRadiationCalculation = self.eastSolarRadiationCalculation + solarRadiationCalculation
        solarRadiationPerMonth = solarRadiationPerMonth + solarRadiationCalculation
    elif wall.orientation == "South":
        shadowReductionFactor = southShadowReductionFactor
        incidentSolarRadiationPerMonth = southIncidentSolarRadiationPerMonth
        wallCollectorArea = wall.absorptance * heatTransferResistanceOutside * wall.heatTransferCoefficient *
wall.area
        heatFlow = heatTransferResistanceOutside * wall.heatTransferCoefficient * wall.area *
heatTransferCoefficientOutside * timeAverageTemperatureDifference
        solarRadiationCalculation = (shadowReductionFactor * wallCollectorArea *
            incidentSolarRadiationPerMonth[(data["M"].values[index] - 1)]) - (
                verticalFormFactor * heatFlow)
        self.southSolarRadiationCalculation = self.southSolarRadiationCalculation + solarRadiationCalculation
        solarRadiationPerMonth = solarRadiationPerMonth + solarRadiationCalculation
    elif wall.orientation == "West":
        shadowReductionFactor = westShadowReductionFactor
        incidentSolarRadiationPerMonth = westIncidentSolarRadiationPerMonth
        wallCollectorArea = wall.absorptance * heatTransferResistanceOutside * wall.heatTransferCoefficient *
wall.area
        heatFlow = heatTransferResistanceOutside * wall.heatTransferCoefficient * wall.area *
heatTransferCoefficientOutside * timeAverageTemperatureDifference
        solarRadiationCalculation = (shadowReductionFactor * wallCollectorArea *
            incidentSolarRadiationPerMonth[(data["M"].values[index] - 1)]) - (
                verticalFormFactor * heatFlow)
        self.westSolarRadiationCalculation = self.westSolarRadiationCalculation + solarRadiationCalculation
        solarRadiationPerMonth = solarRadiationPerMonth + solarRadiationCalculation

    #for the roof
    shadowReductionFactor = roofShadowReductionFactor
    incidentSolarRadiationPerMonth = roofIncidentSolarRadiationPerMonth
    for roof in self.roofList:
        roofCollectorArea = roof.absorptance * heatTransferResistanceOutside * roof.heatTransferCoefficient *
roof.area
        heatFlow = heatTransferResistanceOutside * roof.heatTransferCoefficient * roof.area *
heatTransferCoefficientOutside * timeAverageTemperatureDifference
        solarRadiationCalculation = (shadowReductionFactor * roofCollectorArea *
            incidentSolarRadiationPerMonth[(data["M"].values[index]-1)]) - (roofFormFactor * heatFlow)
        solarRadiationPerMonth = solarRadiationPerMonth + solarRadiationCalculation
        self.roofSolarRadiationCalculation = self.roofSolarRadiationCalculation + solarRadiationCalculation

    #for the ground
    shadowReductionFactor = basementShadowReductionFactor
    incidentSolarRadiationPerMonth = basementIncidentSolarRadiationPerMonth
    for ground in self.groundList:
        basementCollectorArea = ground.absorptance * heatTransferResistanceOutside *
ground.heatTransferCoefficient * ground.area
        heatFlow = heatTransferResistanceOutside * ground.heatTransferCoefficient * ground.area *
heatTransferCoefficientOutside * timeAverageTemperatureDifference
        solarRadiationCalculation = (shadowReductionFactor * basementCollectorArea *

```

```

incidentSolarRadiationPerMonth[(data["M"].values[index]-1))] - (basementFormFactor * heatFlow)
solarRadiationPerMonth = solarRadiationPerMonth + solarRadiationCalculation
self.groundSolarRadiationCalculation = self.groundSolarRadiationCalculation + solarRadiationCalculation

frameFactor = 0.1
for opening in self.openingList:
    if opening.orientation == "North":
        shadowReductionFactor = northShadowReductionFactor
        incidentSolarRadiationPerMonth = northIncidentSolarRadiationPerMonth
        if opening.solarHeatGainCoefficient == 0:
            openingCollectorArea = opening.absorptance * heatTransferResistanceOutside *
opening.heatTransferCoefficient * opening.area
        else:
            openingCollectorArea = (1-frameFactor) * opening.solarHeatGainCoefficient * opening.area
            heatFlow = heatTransferResistanceOutside * opening.heatTransferCoefficient * opening.area *
heatTransferCoefficientOutside * timeAverageTemperatureDifference
            solarRadiationCalculation = (shadowReductionFactor * openingCollectorArea *
            incidentSolarRadiationPerMonth[(data["M"].values[index] - 1))] - (
                verticalFormFactor * heatFlow)
            self.northOpeningSolarRadiationCalculation = self.northOpeningSolarRadiationCalculation +
solarRadiationCalculation
            solarRadiationPerMonth = solarRadiationPerMonth + solarRadiationCalculation
        elif opening.orientation == "East":
            shadowReductionFactor = eastShadowReductionFactor
            incidentSolarRadiationPerMonth = eastIncidentSolarRadiationPerMonth
            if opening.solarHeatGainCoefficient == 0:
                openingCollectorArea = opening.absorptance * heatTransferResistanceOutside *
opening.heatTransferCoefficient * opening.area
            else:
                openingCollectorArea = (1-frameFactor) * opening.solarHeatGainCoefficient * opening.area
                heatFlow = heatTransferResistanceOutside * opening.heatTransferCoefficient * opening.area *
heatTransferCoefficientOutside * timeAverageTemperatureDifference
                solarRadiationCalculation = (shadowReductionFactor * openingCollectorArea *
                incidentSolarRadiationPerMonth[(data["M"].values[index] - 1))] - (
                    verticalFormFactor * heatFlow)
                self.eastOpeningSolarRadiationCalculation = self.eastOpeningSolarRadiationCalculation +
solarRadiationCalculation
                solarRadiationPerMonth = solarRadiationPerMonth + solarRadiationCalculation
            elif opening.orientation == "South":
                shadowReductionFactor = southShadowReductionFactor
                incidentSolarRadiationPerMonth = southIncidentSolarRadiationPerMonth
                if opening.solarHeatGainCoefficient == 0:
                    openingCollectorArea = opening.absorptance * heatTransferResistanceOutside *
opening.heatTransferCoefficient * opening.area
                else:
                    openingCollectorArea = (1 - frameFactor) * opening.solarHeatGainCoefficient * opening.area
                    heatFlow = heatTransferResistanceOutside * opening.heatTransferCoefficient * opening.area *
heatTransferCoefficientOutside * timeAverageTemperatureDifference
                    solarRadiationCalculation = (shadowReductionFactor * openingCollectorArea *
                    incidentSolarRadiationPerMonth[(data["M"].values[index] - 1))] - (
                        verticalFormFactor * heatFlow)
                    self.southOpeningSolarRadiationCalculation = self.southOpeningSolarRadiationCalculation +
solarRadiationCalculation

```



```

        solarRadiationPerMonth = solarRadiationPerMonth + solarRadiationCalculation
    elif opening.orientation == "West":
        shadowReductionFactor = westShadowReductionFactor
        incidentSolarRadiationPerMonth = westIncidentSolarRadiationPerMonth
        if opening.solarHeatGainCoefficient == 0:
            openingCollectorArea = opening.absorptance * heatTransferResistanceOutside *
opening.heatTransferCoefficient * opening.area
        else:
            openingCollectorArea = (1-frameFactor) * opening.solarHeatGainCoefficient * opening.area
            heatFlow = heatTransferResistanceOutside * opening.heatTransferCoefficient * opening.area *
heatTransferCoefficientOutside * timeAverageTemperatureDifference
            solarRadiationCalculation = (shadowReductionFactor * openingCollectorArea *
            incidentSolarRadiationPerMonth[(data["M"].values[index] - 1)]) - (
                verticalFormFactor * heatFlow)
            self.westOpeningSolarRadiationCalculation = self.westOpeningSolarRadiationCalculation +
solarRadiationCalculation
            solarRadiationPerMonth = solarRadiationPerMonth + solarRadiationCalculation

self.wallSolarRadiationPerMonth.append(round((solarRadiationPerMonth*0.0036), 1))
self.northSolarRadiationCalculation = self.northSolarRadiationCalculation * 0.0036
self.eastSolarRadiationCalculation = self.eastSolarRadiationCalculation * 0.0036
self.southSolarRadiationCalculation = self.southSolarRadiationCalculation * 0.0036
self.westSolarRadiationCalculation = self.westSolarRadiationCalculation * 0.0036
self.roofSolarRadiationCalculation = self.roofSolarRadiationCalculation * 0.0036
self.groundSolarRadiationCalculation = self.groundSolarRadiationCalculation * 0.0036
self.northOpeningSolarRadiationCalculation = self.northOpeningSolarRadiationCalculation * 0.0036
self.eastOpeningSolarRadiationCalculation = self.eastOpeningSolarRadiationCalculation * 0.0036
self.southOpeningSolarRadiationCalculation = self.southOpeningSolarRadiationCalculation * 0.0036
self.westOpeningSolarRadiationCalculation = self.westOpeningSolarRadiationCalculation * 0.0036

#TAKE UTILIZATION FACTOR
utilizationGainFactor = 0.65

#TAKE HEAT AND COOLING DEMAND
self.heatGains = []
self.heatLosses = []
for row in range(0, len(self.wallSolarRadiationPerMonth)):
    self.heatGains.append((self.internalHeatPerMonth[row] + self.wallSolarRadiationPerMonth[row]))
    self.heatLosses.append ((self.wallCalculationPerMonth[row] + self.windowCalculationPerMonth[row] +
self.ventilationCalculationPerMonth[row]))
self.heatingDemand = []
self.coolingDemand = []
for row in range(0, len(self.heatGains)):
    if row >=4 and row <=8:
        self.heatingDemand.append(0)
        self.coolingDemand.append(((1-utilizationGainFactor)*self.heatGains[row]))
    else:
        self.heatingDemand.append((self.heatLosses[row]+(utilizationGainFactor*self.heatGains[row])))
        self.coolingDemand.append(0)
print "SelfHD", self.heatingDemand
print "CD", self.coolingDemand

```

```

print "internal heat = ", self.internalHeatPerMonth
print "solar radiation= ", self.wallSolarRadiationPerMonth
print " wallcm", self.wallCalculationPerMonth
print "windowcm", self.windowCalculationPerMonth
print "ventilcm", self.ventilationCalculationPerMonth
print "heat gains = ", self.heatGains
print "heat losses = ", self.heatLosses
print sum(self.heatingDemand), sum(self.coolingDemand)
# Get total wall and window area
totalWallArea = 0
totalWindowArea = 0
for wall in self.wallList:
    totalWallArea = totalWallArea + wall.area
for opening in self.openingList:
    totalWindowArea = totalWindowArea + opening.area

#STEP3: SAVE AND SHOW THE RESULTS
def lastStep(self):

    # save the results in the two lists
    self.resultsSaved()

    #image
    self.getResultsOnImage()

    #feedback
    self.getFeedback()

#SAVE RESULTS FOR IMAGES
def resultsSaved(self):
    # put the final results to the result list
    self.energyResults.append((round(sum(self.internalHeatPerMonth),2),
round(sum(self.windowCalculationPerMonth) + sum(self.wallCalculationPerMonth)
+sum(self.ventilationCalculationPerMonth),2),
round(sum(self.wallSolarRadiationPerMonth),2), round(-(sum (self.heatingDemand)),2),
round(sum(self.coolingDemand),2)))

    self.tipResultsSaved()

#SAVE RESULTS FOR WHATIF SCENARIOS
def tipResultsSaved(self):

    self.tipResults.append((round(sum(self.internalHeatPerMonth),2),
round(sum(self.windowCalculationPerMonth) + sum(self.wallCalculationPerMonth)
+sum(self.ventilationCalculationPerMonth),2),
round(sum(self.wallSolarRadiationPerMonth),2), round(-(sum (self.heatingDemand)),2),
round(sum(self.coolingDemand),2)))

#DRAW RESULTS ON THE IMAGE
def getResultsOnImage(self):
    image = Image.open(r"C:\Users\Miltos\Desktop\energy_performance_viewer\app_env\Diagram.png")
    draw = ImageDraw.Draw(image)

```

```
font = ImageFont.truetype(r"C:\Users\Miltos\Desktop\energy_performance_viewer\app_env\Roboto-Black.ttf",
```

```
    size=36)
```

```
colorBlack = 'rgb(0, 0, 0)'
```

```
colorRed = 'rgb(255, 0, 0)'
```

```
colorGreen = 'rgb(0, 128, 0)'
```

```
text0 = "Heating Demand"
```

```
(x10, y10) = (50, 50)
```

```
text01 = "Cooling Demand"
```

```
(x101, y101) = (400, 50)
```

```
text1 = "Internal Heat"
```

```
(x11, y11) = (320, 550)
```

```
text2 = "Heat Transfer"
```

```
(x12, y12) = (680, 450)
```

```
text3 = "SolarRadiation"
```

```
(x13, y13) = (680, 150)
```

```
draw.text((x10, y10), text0, fill=colorBlack, font=font)
```

```
draw.text((x101, y101), text01, fill=colorBlack, font=font)
```

```
draw.text((x11, y11), text1, fill=colorBlack, font=font)
```

```
draw.text((x12, y12), text2, fill=colorBlack, font=font)
```

```
draw.text((x13, y13), text3, fill=colorBlack, font=font)
```

```
# If these are the first results
```

```
if len(self.energyResults) == 1:
```

```
    for row in self.energyResults:
```

```
        (x1, y1) = (320, 600)
```

```
        internalHeatText = "+" + " " + str(row[0]) + "MJ"
```

```
        (x2, y2) = (680, 500)
```

```
        heatTransfer = row[1]
```

```
        if heatTransfer >= 0:
```

```
            mark = "+"
```

```
        else:
```

```
            mark = ""
```

```
        heatTransferText = mark + " " + str(row[1]) + "MJ"
```

```
        (x3, y3) = (680, 200)
```

```
        solarHeatText = "+" + " " + str(row[2]) + "MJ"
```

```
        (x4, y4) = (50, 100)
```

```
# totalHeatEnergyConsumption = row[3]
```

```
# if totalHeatEnergyConsumption >= 0:
```

```
#     mark = "+"
```

```
# else:
```

```
#     mark = ""
```

```
        HeatingDemandText = "+" + " " + str(row[3]) + "MJ"
```

```
        (x5, y5) = (400, 100)
```

```
        CoolingDemandText = "+" + " " + str(row[4]) + "MJ"
```

```
draw.text((x1, y1), internalHeatText, fill=colorBlack, font=font)
```

```
draw.text((x2, y2), heatTransferText, fill=colorBlack, font=font)
```

```
draw.text((x3, y3), solarHeatText, fill=colorBlack, font=font)
```

```
draw.text((x4, y4), HeatingDemandText, fill=colorBlack, font=font)
```

```
draw.text((x5, y5), CoolingDemandText, fill=colorBlack, font=font)
```

```

# after refresh (auto einai epeidi kanei 2 refreshes)
elif len(self.energyResults)==2:
    differences = tuple(x - y for x, y in zip(self.energyResults[-1], self.energyResults[-2]))
    color1 = colorBlack
    color2 = colorBlack
    color3 = colorBlack
    color4 = colorBlack
    color5 = colorBlack
    if differences[0] > 0:
        color1 = colorGreen
    elif differences[0] < 0:
        color1 = colorRed
    if differences[1] > 0:
        color2 = colorGreen
    elif differences[1] < 0:
        color2 = colorRed
    if differences[2] > 0:
        color3 = colorGreen
    elif differences[2] < 0:
        color3 = colorRed
    # the total is absolute value
    if differences[3] < 0:
        color4 = colorGreen
    elif differences[3] > 0:
        color4 = colorRed
    if differences[4] < 0:
        color5 = colorGreen
    elif differences[4] > 0:
        color5 = colorRed

    oldResults = self.energyResults[-2]
    (x1, y1) = (320, 650)
    internalHeatText = "+" + " " + str(oldResults[0]) + "MJ"
    (x2, y2) = (680, 550)
    heatTransfer = oldResults[1]
    if heatTransfer >= 0:
        mark = "+"
    else:
        mark = ""
    heatTransferText = mark + " " + str(oldResults[1]) + "MJ"
    (x3, y3) = (680, 250)
    solarHeatText = "+" + " " + str(oldResults[2]) + "MJ"
    (x4, y4) = (50, 150)
    HeatingDemandText = "+" + " " + str(oldResults[3]) + "MJ"
    (x5, y5) = (400, 150)
    CoolingDemandText = "+" + " " + str(oldResults[4]) + "MJ"
    draw.text((x1, y1), internalHeatText, fill=colorBlack, font=font)
    draw.text((x2, y2), heatTransferText, fill=colorBlack, font=font)
    draw.text((x3, y3), solarHeatText, fill=colorBlack, font=font)
    draw.text((x4, y4), HeatingDemandText, fill=colorBlack, font=font)
    draw.text((x5, y5), CoolingDemandText, fill=colorBlack, font=font)

```

```

newResults = self.energyResults[-1]
(x6, y6) = (320, 600)
internalHeatTextNew = "+" + " " + str(newResults[0]) + "MJ"
(x7, y7) = (680, 500)
heatTransferNew = newResults[1]
if heatTransferNew >= 0:
    mark = "+"
else:
    mark = ""
heatTransferTextNew = mark + " " + str(newResults[1]) + "MJ"
(x8, y8) = (680, 200)
solarHeatTextNew = "+" + " " + str(newResults[2]) + "MJ"
(x9, y9) = (50, 100)
HeatingDemandTextNew = "+" + " " + str(newResults[3]) + "MJ"
(x10, y10) = (400, 100)
CoolingDemandTextNew = "+" + " " + str(newResults[4]) + "MJ"

draw.text((x6, y6), internalHeatTextNew, fill=color1, font=font)
draw.text((x7, y7), heatTransferTextNew, fill=color2, font=font)
draw.text((x8, y8), solarHeatTextNew, fill=color3, font=font)
draw.text((x9, y9), HeatingDemandTextNew, fill=color4, font=font)
draw.text((x10, y10), CoolingDemandTextNew, fill=color5, font=font)

```

**else:**

```

differences = tuple(x - y for x, y in zip(self.energyResults[-1], self.energyResults[-3]))
color1 = colorBlack
color2 = colorBlack
color3 = colorBlack
color4 = colorBlack
color5 = colorBlack
if differences[0] > 0:
    color1 = colorGreen
elif differences[0] < 0:
    color1 = colorRed
if differences[1] > 0:
    color2 = colorGreen
elif differences[1] < 0:
    color2 = colorRed
if differences[2] > 0:
    color3 = colorGreen
elif differences[2] < 0:
    color3 = colorRed
# the total is absolute value
if differences[3] < 0:
    color4 = colorGreen
elif differences[3] > 0:
    color4 = colorRed
if differences[4] < 0:
    color5 = colorGreen
elif differences[4] > 0:
    color5 = colorRed

```

```

oldResults = self.energyResults[-3]
(x1, y1) = (320, 650)
internalHeatText = "+" + " " + str(oldResults[0]) + "MJ"
(x2, y2) = (680, 550)
heatTransfer = oldResults[1]
if heatTransfer >= 0:
    mark = "+"
else:
    mark = ""
heatTransferText = mark + " " + str(oldResults[1]) + "MJ"
(x3, y3) = (680, 250)
solarHeatText = "+" + " " + str(oldResults[2]) + "MJ"
(x4, y4) = (50, 150)
HeatingDemandText = "+" + " " + str(oldResults[3]) + "MJ"
(x5, y5) = (400, 150)
CoolingDemandText = "+" + " " + str(oldResults[4]) + "MJ"
draw.text((x1, y1), internalHeatText, fill=colorBlack, font=font)
draw.text((x2, y2), heatTransferText, fill=colorBlack, font=font)
draw.text((x3, y3), solarHeatText, fill=colorBlack, font=font)
draw.text((x4, y4), HeatingDemandText, fill=colorBlack, font=font)
draw.text((x5, y5), CoolingDemandText, fill=colorBlack, font=font)

newResults = self.energyResults[-1]
(x6, y6) = (320, 600)
internalHeatTextNew = "+" + " " + str(newResults[0]) + "MJ"
(x7, y7) = (680, 500)
heatTransferNew = newResults[1]
if heatTransferNew >= 0:
    mark = "+"
else:
    mark = ""
heatTransferTextNew = mark + " " + str(newResults[1]) + "MJ"
(x8, y8) = (680, 200)
solarHeatTextNew = "+" + " " + str(newResults[2]) + "MJ"
(x9, y9) = (50, 100)
HeatingDemandTextNew = "+" + " " + str(newResults[3]) + "MJ"
(x10, y10) = (400, 100)
CoolingDemandTextNew = "+" + " " + str(newResults[4]) + "MJ"

draw.text((x6, y6), internalHeatTextNew, fill=color1, font=font)
draw.text((x7, y7), heatTransferTextNew, fill=color2, font=font)
draw.text((x8, y8), solarHeatTextNew, fill=color3, font=font)
draw.text((x9, y9), HeatingDemandTextNew, fill=color4, font=font)
draw.text((x10, y10), CoolingDemandTextNew, fill=color5, font=font)

imageFirstConvertToQ = ImageQt(image)
imageSecondConvertToQ = QImage(imageFirstConvertToQ)

self.pixmap = QtGui.QPixmap(imageSecondConvertToQ)
self.pixmap = self.pixmap.scaled(300, 300, QtCore.Qt.KeepAspectRatio)
self.label.setPixmap(self.pixmap)
self.label.show()

```

*#CREATE THE FEEDBACKS*

```
def getFeedback(self):

    totalRoofArea = 0
    totalGroundArea = 0

    for roof in self.roofList:
        totalRoofArea = totalRoofArea + roof.area
    for ground in self.groundList:
        totalGroundArea = totalGroundArea + ground.area

    wallHeatTransfer = self.wallTotalHeatTransfer / (self.northWallArea + self.southWallArea + self.eastWallArea +
self.westWallArea)
    roofHeatTransfer = self.roofTotalHeatTransfer / totalRoofArea
    groundHeatTransfer = self.groundTotalHeatTransfer / totalGroundArea
    if (self.northOpeningArea + self.southOpeningArea + self.eastOpeningArea + self.westOpeningArea) == 0:
        openingHeatTransfer = 0
    else:
        openingHeatTransfer = self.openingTotalHeatTransfer / (self.northOpeningArea + self.southOpeningArea +
self.eastOpeningArea + self.westOpeningArea )

    northConsumptionWall = self.northSolarRadiationCalculation / self.northWallArea + wallHeatTransfer
    eastConsumptionWall = self.eastSolarRadiationCalculation / self.eastWallArea + wallHeatTransfer
    southConsumptionWall = self.southSolarRadiationCalculation / self.southWallArea + wallHeatTransfer
    westConsumptionWall = self.westSolarRadiationCalculation / self.westWallArea + wallHeatTransfer

    roofConsumption = self.roofSolarRadiationCalculation / totalRoofArea + roofHeatTransfer

    groundConsumption = self.groundSolarRadiationCalculation / totalGroundArea + groundHeatTransfer
    if self.northOpeningArea == 0:
        northConsumptionOpening = 0
    else:
        northConsumptionOpening = self.northOpeningSolarRadiationCalculation / self.northOpeningArea +
openingHeatTransfer)

    if self.eastOpeningArea == 0:
        eastConsumptionOpening = 0
    else:
        eastConsumptionOpening = self.eastOpeningSolarRadiationCalculation / self.eastOpeningArea +
openingHeatTransfer

    if self.southOpeningArea == 0:
        southConsumptionOpening = 0
    else:
        southConsumptionOpening = self.southOpeningSolarRadiationCalculation / self.southOpeningArea +
openingHeatTransfer

    if self.westOpeningArea == 0:
        westConsumptionOpening = 0
    else:
        westConsumptionOpening = self.westOpeningSolarRadiationCalculation / self.westOpeningArea +
openingHeatTransfer
```

```

results = self.energyResults[-1]
self.feedbackBox.clear()
self.feedbackBox.append("The total energy consumption of the building is: " + str(results[3]) + "MJ")
self.feedbackBox.append("Each element contributes differently to this result")
self.feedbackBox.append("")
self.feedbackBox.append("a. walls")
self.feedbackBox.append("-north: " + str(round(northConsumptionWall,2)) + " MJ/m^2")
self.feedbackBox.append("-east: " + str(round(eastConsumptionWall,2)) + " MJ/m^2")
self.feedbackBox.append("-south: " + str(round(southConsumptionWall,2)) + " MJ/m^2")
self.feedbackBox.append("-west: " + str(round(westConsumptionWall,2)) + " MJ/m^2")
self.feedbackBox.append("")
self.feedbackBox.append("b. roofs/ground")
self.feedbackBox.append("-roof: " + str(round(roofConsumption,2)) + " MJ/m^2")
self.feedbackBox.append("-ground: " + str(round(groundConsumption,2)) + " MJ/m^2")
self.feedbackBox.append("")
self.feedbackBox.append("c. openings")
self.feedbackBox.append("-north: " + str(round(northConsumptionOpening,2)) + " MJ/m^2")
self.feedbackBox.append("-east: " + str(round(eastConsumptionOpening, 2)) + " MJ/m^2")
self.feedbackBox.append("-south: " + str(round(southConsumptionOpening, 2)) + " MJ/m^2")
self.feedbackBox.append("-west: " + str(round(westConsumptionOpening, 2)) + " MJ/m^2")

#VIEW AND DRAW THE ENERGY RESULTS
def viewEnergyResults (self):

    # Pop up window only once
    if not self.ResultsDialog.isVisible():
        self.ResultsDialog.show()
    # get the graph
    self.getVisualGraphAsBarChart(self.heatGains, self.heatLosses)
    # get the table results
    self.getTableResults()

def getTableResults(self):
    months = ["January", "February", "March", "April", "May", "June", "July", "August", "September",
"October",
    "November", "December"]
    results = ["Heat Transmission", "Heat Ventilation", "Internal Heat Gain", "Solar Radiation"]
    horHeaders = []
    verHeaders = []
    rowList = []
    heatTransmission = []
    for row in range(0, len(self.wallCalculationPerMonth)):
        heatTransmission.append((self.wallCalculationPerMonth[row] + self.windowCalculationPerMonth[row]))

    self.table.setRowCount(0)
    self.table.setColumnCount(0)
    for i,month in enumerate (months):
        columnPosition = self.table.columnCount()
        if columnPosition <= 11:
            self.table.insertColumn(columnPosition)
            horHeaders.append(month)
    for k, result in enumerate (results):

```



```

rowPosition = self.table.rowCount()
if rowPosition <= 3:
    self.table.insertRow(rowPosition)
    verHeaders.append(result)
    if k == 0:
        for row in heatTransmission:
            rowList.append(row)
    elif k == 1:
        for row in self.ventilationCalculationPerMonth:
            rowList.append(row)
    elif k == 2:
        for row in self.internalHeatPerMonth:
            rowList.append(row)
    elif k == 3:
        for row in self.wallSolarRadiationPerMonth:
            rowList.append(row)
    for j, month in enumerate(rowList):
        self.table.setItem(rowPosition, j, QtGui.QTableWidgetItem(str(rowList[j])))
    rowList = []

```

```

self.table.setHorizontalHeaderLabels(horHeaders)
self.table.setVerticalHeaderLabels(verHeaders)
self.table.resizeColumnsToContents()
self.table.resizeRowsToContents()
self.table.setFixedHeight(140)
self.table.show()

```

```

def getVisualGraphAsBarChart(self, heatGains, heatLosses):

```

```

    heatLosses2 = []
    for row in heatLosses:
        loss = -row
        heatLosses2.append(loss)

```

```

    consumptionObjects = (heatLosses2, heatGains)
    y_pos = np.arange(12)
    barWidth = 0.35

```

```

    self.ax = self.figure.add_subplot(111)
    Consumption1 = self.ax.bar(y_pos - barWidth, heatLosses2, barWidth, color='r')

```

```

    Gain1 = self.ax.bar(y_pos, heatGains, barWidth, color='g')

```

```

    self.ax.set_ylabel("MJ")
    self.ax.set_title("Heat Gains & Losses")
    self.ax.set_xticks(y_pos + barWidth / 2)
    self.ax.set_xticklabels(("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"))

```

```

init = initUI()

```

## APPENDIX X: Case study energy results

	January	February	March	April	May	June	July	August	September	October	November	December
Heat Transmission	-3645.3	-2870.9	-2834.1	-2282.7	-1603.9	-1052.3	-818.8	-820.2	-1311.6	-1977.3	-2699.7	-3368.1
Heat Ventilation	-2826.7	-2193.1	-2134.1	-1672.6	-1083.8	-622.1	-413.5	-414.7	-843.5	-1402.6	-2028.6	-2590.0
Internal Heat Gain	2204.1	2038.8	2204.1	2148.9	2204.1	2148.9	2204.1	2204.1	2148.9	2204.1	2148.9	2204.1
Solar Radiation	-77.8	34.8	188.0	586.7	793.5	890.4	752.3	680.1	357.2	152.1	-49.5	-116.5

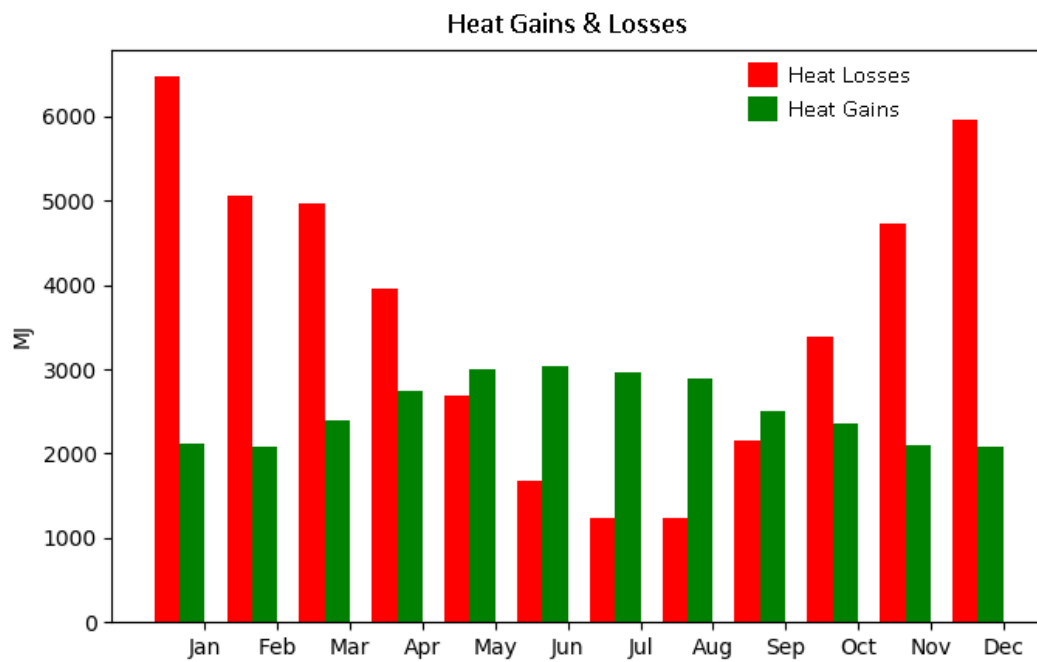


Figure 15: Energy results of case study

## APPENDIX XI: Comparison of the results of the two test cases

Table 16: VABI elements results

Number Room	Name Room	Area [m <sup>2</sup> ]	Volume [m <sup>3</sup> ]	Design temperature [°C]	Transmission loss [W]	Ventilation loss [W]	Reheat [W]	Total heat losses [W]
1	<New room>	105.32	251.35	20	1155	2018	1568	4741

Number Room	Name Room	Area [m <sup>2</sup> ]	Volume [m <sup>3</sup> ]	Design temperature [°C]	Transmission loss [W]	Ventilation loss [W]	Reheat [W]	Total heat losses [W]
1	<New room>	105.32	251.35	20	1052	1978	1568	4598

Number Room	Name Room	Area [m <sup>2</sup> ]	Volume [m <sup>3</sup> ]	Design temperature [°C]	Transmission loss [W]	Ventilation loss [W]	Reheat [W]	Total heat losses [W]
1	<New room>	105.32	251.35	20	1052	1978	1568	4598

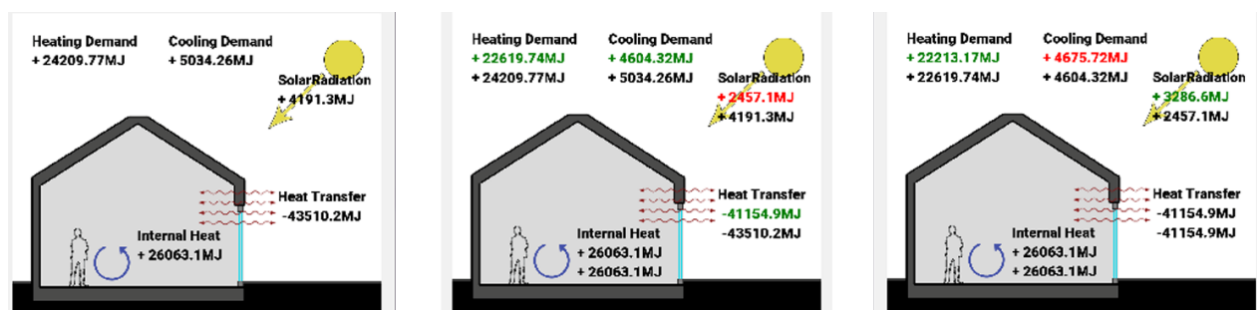


Figure 16: Developed tool results

\* 1W = 31.5 MJ/ye