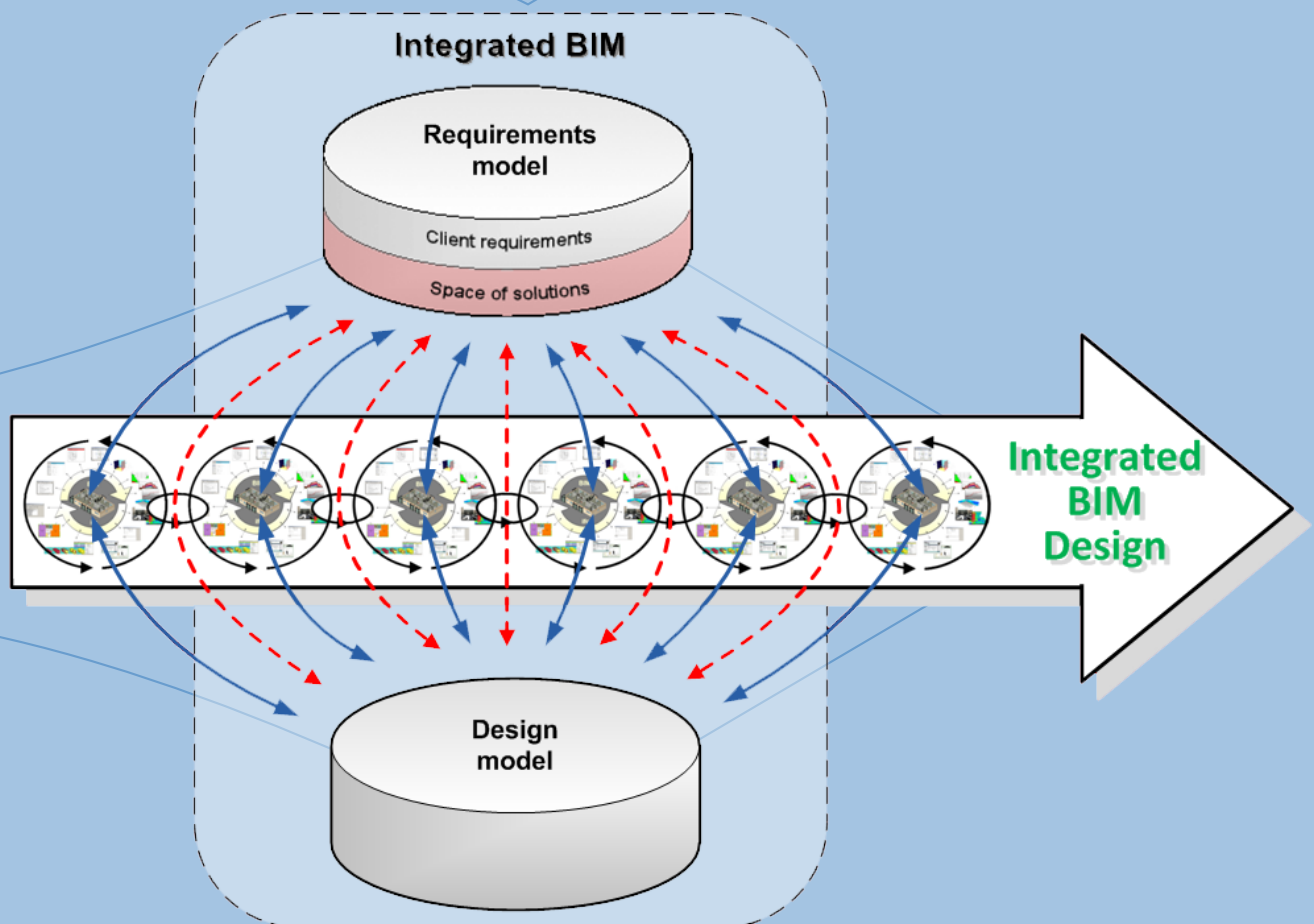


Niels Trelldal

Integrated Data and Process Control During BIM Design

Focused on Integrated Design of Energy and Indoor Climate Conditions

Master's Thesis, February 2008



Integrated Data and Process Control During BIM Design

- focused on Integrated Design of Energy and Indoor Climate Conditions

Master's Thesis

Author	Niels Trelldal, s021820
Project period	September 2007 – February 2008
Submission date	1st February 2008
Supervisors	Flemming Vestergaard, BYG•DTU Carsten Rode, BYG•DTU
External Supervisor	Jan Karlshøj, Rambøll Danmark A/S
Places	Department of Civil Engineering (BYG•DTU) Technical University of Denmark Brovej, Building 118 DK-2800 Kgs. Lyngby Rambøll Danmark A/S Teknikerbyen 31 DK-2830 Virum

**** BIM = Building Information Model ****

Version 2 – March 2008

This second edition includes minor corrections identified after the submission of the thesis.

Abstract

Integration among members of a design team and of their defined tasks in building construction projects has for several decades been considered a suitable way to improve co-operation in construction projects by limiting the fragmented structure usually characterising the AEC industry. Integration can be separated into two areas; data integration and process integration. A concept of data integration which is gaining momentum in the AEC industry is the use of Building Information Models (BIM) in the design process supported by the use of the open standard IFC. Likewise, the process integration concept of integrated design is starting to be used in the AEC industry primarily driven by tightened energy demands.

This thesis aims at identifying how an integrated design process can be improved by using BIM design, and in particular how a shared data repository defined by use of the current IFC specifications may support the improved process.

In order to achieve this goal, the thesis sets out with a literature study on the concept of integrated design and BIM design. Here it is identified that BIM design could improve the integrated design process by allowing for interoperability among simulation tools and new possibilities for management and control of requirements for the design decisions. Hereupon follows an analysis of how identified input and output data of five selected simulation tools and requirements may be defined using the IFC specifications. To further explore the capabilities of the IFC specifications, an IFC interface for the Danish energy calculation tool Be06 is subsequently developed. Finally the findings in the research are used to define a methodology of how BIM design can improve integrated design.

The methodology developed in the thesis is called Integrated BIM Design. It consists of a decision-making process which follows an integrated procedure in order to ensure that the design goals are met. To support the integrated procedure, the project-related information should be defined in an integrated BIM. The integrated BIM must be separated into a design model and a requirements model in order to allow for appropriate information sharing and abilities to control the design development.

It is found that 70 % of input and output data of the five simulation tools can be defined using IFC, and that the concepts for requirements definition in IFC will result in a range of new extensions to be included in the specifications before requirements can be defined. The development of the IFC interface illustrated that it can be complicated to create mapping between information defined in a BIM and in the simulation tools. To allow for Integrated BIM Design to be used, it will require improvements to the IFC specifications, improved development of IFC support for existing tools, and a set of new tools primarily for requirements management.

Contrary to previous ideas for integration the suggested methodology of Integrated BIM Design is based on concepts already being implemented in the AEC industry. Therefore, it could allow for the goal of complete integration to be reached within a reasonable number of years particularly if the clients would ensure that this development is pushed forward.

Preface

This thesis is created as part of the degree for a Master of Science in Engineering (MSc) at the Department of Civil Engineering (BYG•DTU) at the Technical University of Denmark. The thesis constitutes 30 ECTS-points which represents the workload of one semester and has been carried out during the autumn semester 2007 (September 2007 – February 2008).

Because the content of the thesis is related to the international standard IFC, the thesis is written in English to allow for its results to contribute to the future development of the standard.

Simultaneously with the research for this thesis, two students at the IT University of Copenhagen, Ole Berard and Jesper Schulz, have been working on a Master's Thesis examining the geometrical capabilities of IFC in relation to thermal simulation tools. While researching for our separate theses, we have strived for mutual work support; however, a displacement in the research periods for the two theses has prevented the results of the IT University work from being included in the present thesis.

All source references in the thesis are indicated by square brackets. If a quotation is used directly from a source, it will be indicated by quotation marks. All references which are not public available have been included on the CD enclosed with the thesis. References included on the CD are marked in the list of references.

The CD also includes IFC files containing the models described in Chapter 3 and 5. They can be used to test the IFC interface created in Chapter 4. This interface is also included on the CD along with a PDF version of the thesis.

During the research for the thesis, several employees at Rambøll Danmark A/S have assisted with technical guidance and support for the use of various simulation tools. I am very grateful for this support and would like in particular to thank Jørn Trelldal and Frederik Vildbrad Winther for their assistance.

I would also like to thank the supervisors, Flemming Vestergaard, Carsten Rode, and Jan Karlshøj for taking their time to go through the large amount of material generated during this research and for providing valuable feedback.

Copenhagen, February 1, 2008

Niels Trelldal

Table of Contents

ABSTRACT	I
PREFACE	II
TABLE OF CONTENTS	III
LIST OF TERMS.....	VI
CHAPTER 1 INTRODUCTION	1
1.1 BACKGROUND	2
1.2 THESIS STATEMENT	4
1.3 SCOPE OF THESIS	4
1.4 METHOD	4
1.5 INSTRUCTIONS FOR READING	5
CHAPTER 2 LITERATURE STUDY ON INTEGRATED DESIGN AND BIM DESIGN.....	6
2.1 INTRODUCTION.....	7
2.2 INTEGRATED DESIGN.....	7
2.2.1 <i>The Need for Integrated Design</i>	7
2.2.2 <i>Concept of Integrated Design</i>	9
2.2.3 <i>Integrated Design Process</i>	10
2.2.4 <i>Use of Simulation Tools in IDP</i>	12
2.2.5 <i>Space of Solutions</i>	14
2.2.6 <i>Sensitivity Analysis</i>	16
2.2.7 <i>Summary on Integrated Design</i>	17
2.3 BIM DESIGN	17
2.3.1 <i>The Need for BIM Design</i>	17
2.3.2 <i>Concept of BIM Design</i>	19
2.3.3 <i>IFC as the Shared Data Repository</i>	22
2.3.4 <i>Information Delivery Manual and Model View Definitions</i>	25
2.3.5 <i>Centralised Data Repository – Model Server Concept</i>	25
2.3.6 <i>Code Compliance and BIM Evaluation</i>	27
2.3.7 <i>Definition of a Client Requirements Model</i>	28
2.3.8 <i>Summary on BIM Design</i>	31
2.4 POSSIBILITIES TO IMPROVE IDP BY USING BIM DESIGN	31
2.4.1 <i>Challenges for Successful Use of IDP</i>	31
2.4.2 <i>Interoperability and Simulation Tools</i>	32
2.4.3 <i>Information Management and Control</i>	32
2.5 SUMMERY OF POSSIBILITIES.....	33
CHAPTER 3 CAPABILITIES OF THE IFC SPECIFICATIONS.....	34
3.1 INTRODUCTION.....	35
3.2 IFC CAPABILITIES FOR INTEROPERABILITY IN IBD	35

3.2.1	<i>Identification of Input and Output Data for Simulation Tools</i>	36
3.2.1.1	Input/Output Data Analysis of iDbuild	38
3.2.1.2	Input/Output Data Analysis of Riuska	39
3.2.1.3	Input/Output Data Analysis of BSim	40
3.2.1.4	Input/Output Data Analysis of Be06	41
3.2.1.5	Input/Output Data Analysis of Flovent	42
3.2.2	<i>Capabilities of IFC to Contain Information Required for Simulation Tools</i>	43
3.2.2.1	Generation of Information	43
3.2.2.2	Geometrical Representations and Placement in IFC.....	45
3.2.2.3	Definition of Possible Locations for Non-Geometry Information in IFC	46
3.2.2.4	Use of Properties in IFC to Store Information.....	46
3.2.3	<i>Comparison of input and output data and IFC capabilities</i>	49
3.2.3.1	Input Data for Projects, Sites, Buildings and Stories	50
3.2.3.2	Input Data for Spaces	51
3.2.3.3	Input Data for Walls and Slabs	53
3.2.3.4	Input Data for Windows and Doors	55
3.2.3.5	Input Data for Furniture	56
3.2.3.6	Input Data for Ventilation systems	57
3.2.3.7	Input Data for Cooling Systems.....	59
3.2.3.8	Input Data for Heating and Domestic Hot Water Systems	59
3.2.3.9	Input Data for Lighting Systems	60
3.2.3.10	Input Data for Electrical components	61
3.2.3.11	Output Data.....	61
3.2.4	<i>Identified Capabilities for Interoperability</i>	62
3.3	IFC CAPABILITIES FOR MANAGEMENT AND CONTROL IN IBD.....	64
3.3.1	<i>Space of Solutions in a Requirements Model</i>	65
3.3.1.1	Separation of Requirements and Design Model	66
3.3.1.2	Direct and Indirect Linking	66
3.3.1.3	Storing Constraints in IFC	67
3.3.2	<i>Capabilities of IFC for Compliance Checking</i>	69
3.3.3	<i>Identified Capabilities for Information Management and Control</i>	70
3.4	SUMMARY OF IFC CAPABILITIES	71
CHAPTER 4 IFC IMPLEMENTATION		72
4.1	INTRODUCTION.....	73
4.2	IFC TOOLBOX.....	73
4.3	IFC INTERFACE FOR BE06.....	76
4.3.1	<i>Identified Current Possibilities</i>	76
4.3.2	<i>Included Functionality</i>	78
4.3.3	<i>Development of Interface</i>	79
4.3.3.1	Transposing Local Coordinates to Global Coordinates	79
4.3.3.2	Retrieving Information on Spaces	82
4.3.3.3	Retrieving Information on Walls	85

4.3.3.4	Retrieving Information of Orientation	87
4.3.3.5	Retrieving Information on Windows and Doors	90
4.3.3.6	Retrieving Information on Slabs/Roofs	91
4.3.4	<i>Summarising Information</i>	94
4.3.5	<i>Suggested Future Development of Interface</i>	94
4.4	IDENTIFIED IFC CAPABILITIES	95
CHAPTER 5 METHODOLOGY DESCRIPTION		96
5.1	INTRODUCTION.....	97
5.2	ITERATIONS AND DESIGN MODEL	97
5.3	REQUIREMENTS MODEL AND CONTROL	98
5.4	METHODOLOGY OF IBD	100
5.5	USE OF METHODOLOGY IN CURRENT DESIGN	101
5.5.1	<i>Definition of Client Requirements</i>	102
5.5.2	<i>Room Design</i>	103
5.5.3	<i>Building Composition</i>	105
5.5.4	<i>Optimisation of Windows and Ventilation</i>	106
5.5.5	<i>Façade Design</i>	109
5.6	SUMMARY OF POTENTIALS IN THE METHODOLOGY FOR IDB	110
CHAPTER 6 DISCUSSION.....		112
6.1	INTRODUCTION.....	113
6.2	CONSEQUENCES OF IMPLEMENTING IBD	113
6.2.1	<i>Complexity of Information Sharing</i>	113
6.2.2	<i>Effort of Implementation</i>	114
6.2.3	<i>Ensuring Development and Quality in the Design Process</i>	114
6.3	ACHIEVING PROPER INTEROPERABILITY	115
6.3.1	<i>Improvements within a Short Time Frame</i>	115
6.3.2	<i>Improvements within a Longer Time Frame</i>	116
6.4	ALLOWING FOR REQUIREMENTS MANAGEMENT	118
6.4.1	<i>Improvements within a Short Time Frame</i>	118
6.4.2	<i>Improvements within a Longer Time Frame</i>	118
6.5	REASONS FOR SLOW DEVELOPMENT	119
CHAPTER 7 CONCLUSION		121
REFERENCES		124
LIST OF TABLES		129
TABLE OF FIGURES.....		130
APPENDICES		134

List of Terms

Table 1-1. Terms and definitions used in this thesis, partly based on [Erabuild 2008].

AEC	Architecture, Engineering and Construction	A phrase that may be used as an alternative to describe the building construction industry.
BIM	Building Information Model	An object-oriented, AEC-specific model – a digital representation of a building to facilitate exchange and interoperability of information in digital format. The model can be without geometry or with 2D or 3D representations. Integrated BIM, when you share information, requires open standards.
BIM design	Building Information Model design	The act of creating a Building Information Model.
CFD	Computational Fluid Dynamics	Can analyze problems that involve fluid flows. Typically used in the AEC industry to analyze air movements to evaluate indoor environment.
CAD	Computer Aided Design	A class of software applications that performs a design function.
DTU	Technical University of Denmark	
EEDSS	Energy and Environmental Design Decision Support Systems	Different types of tools used to analyse energy and comfort conditions in buildings.
FM	Facilities Management	Describing the area of building operation and maintenance.
HVAC	Heating, ventilation, and air conditioning systems	The abbreviation for heating, ventilation, and air conditioning systems, used in building design and construction.
IAI	International Alliance for Interoperability	International Alliance for Interoperability. An open consortium to develop, promote and support for implementation of IFC.

IBD	Integrated BIM design	The use of BIM design in the integrated design process to support for interoperability for a better decision-making process and requirements management for proper design development.
IDP	Integrated Design Process	IDP is the entire, multi-disciplinary processing of a design task for which a competent design team continuously pursues durability-oriented targets from the beginning and optimises them in terms of ensuring quality through the application of modern methods and tools throughout each individual phase.
IDM	Information Delivery Manual	A manual being developed within IAI to ensure that information is available when required in the process.
IFC	Industry Foundation Classes	An international specification for product data exchange and sharing for AEC/FM. IFC enables interoperability between the computer applications for AEC/FM. A subset of IFC is approved as ISO/PAS 16739.
MVD	Model View Definitions	Intended as a way to document IFC based data exchange capabilities in software

Chapter 1

Introduction

1.1 Background

The AEC industry today is a fragmented industry where project teams in building construction projects are composed by members from several companies typically representing either architects, engineers or contractors. Each member in the team typically has predefined tasks which confine the parts of the project in which each member is intended to contribute, and the separation of team members is thereby substantial. As with any other project including a project team, co-operation within the team is essential for achieving good results.

Integration among team members and their defined tasks has for several decades been considered a suitable way to improve co-operation in construction projects and limiting the fragmented structure, and substantial research has been conducted in this area. [Augenbroe 1992] distinguishes between two different areas of integration; data integration and process integration. Data integration relates to the challenges of defining a standard for describing design objects, preferably in a format neutral to different design domains. Process integration relates to the definition of the design context for any aspect-related task such as performance evaluation and handling of information flow and design decisions.

The COMBINE project initiated in 1990 is an example of research of integration focusing primarily on data integration. The aim was to increase control over energy-related aspects for all members in the design team by enabling multi-criterion design through integration of a range of specific energy and HVAC-related tools and bringing the results to the disposal of the design team [Augenbroe 1992]. The outcome of the project was a prototype of an Integrated Building Design System with a centralised data model from which several design and simulation tools could access and exchange information on the building. Due to lack of funding the project ended in 1995 before proper implementation in design and simulation tools was reached.

The data model in COMBINE was defined using the STEP standard as a neutral format for storing data. STandard for Exchange of Product (STEP) is an international standard for the computer-interpretable representation and exchange of product data [Forester 2001] and several other research projects have also focused on data integration based on the STEP standard.

The General AEC Reference Model (GARM) is one of the first examples of a data model based on STEP for the AEC industry and was developed around 1988 [IAI 2000]. GARM uses a concept of functional unit/technical solution to define objects via STEP. The idea is that the functional unit consists of requirements for each object. For each functional unit one or more technical solutions can be assigned which then include characteristics of the actual object [Gielingh 1988].

Several other research projects refined the data model for the AEC industry based on STEP during the 1990's [IAI 2000], however, only limited implementation was achieved. One of the main challenges for the research is believed to be the lack of consensus in the industry at that time as most companies were only starting to use drawing based 2D CAD leaving other information to be defined in other types of documents. For this reason, among others, the fragmented structure of design teams continued to be used in the industry with limited integration as a result.

In 1996 a new organisation was formed which was backed by large players in the industry such as Autodesk and Bentley, some of the largest companies in the world developing software for the AEC industry. The organisation was called International Alliance for Interoperability (IAI) and used the basis from the development of STEP to define a new standard called Industry Foundation Classes (IFC). IAI has within the last decade managed to ensure that IFC has been implemented in a large range of software ranging from design and simulation tools to construction and maintenance tools. The IFC specifications which define the standard will be described further in the next chapter.

Along with the implementation of IFC, a range of new tools have emerged which allow for data models to be created and which provide a possibility to assign a large amount of information to the objects in the model. This has led to a new commonly accepted term of working with Building Information Models (BIM) which is a type of data model containing building related information. A recent study by [Erabuild 2008] describes that approximately 19 % of the participants in their survey from the Nordic AEC industry now use BIM techniques in the design work and approximately 5 % of these participants use tools which support IFC. Possibilities for data integration in the AEC industry therefore finally seem to gain momentum.

Similar to data integration, process integration has for a long time been absent in the AEC industry. This is among others reflected by the continuous use of task descriptions like [PAR & F.R.I 2002] in Denmark which define how the separate tasks in a construction project should, to a considerable extent, be completed by specific individual members of the design team. Tightened requirements for energy consumption of buildings in Europe caused by [EPBD 2003] and in particular in Denmark by the Danish Building Code [BR95 2007] has, however, initiated a new effort of trying to intensify the use of process integration to primarily ensuring improved building energy performance. One of the main efforts in the development has been within the project Task 23 - Optimization of Solar Energy Use in Large Buildings by The International Energy Agency (IEA). In this project research institutes and consultants from more than 12 countries worked on refining and implementing a concept of integrated design [Larsson 2003]. The intention of integrated design is to implement a large extent of process integration to ensure an improved result of the construction projects. The exact contents of integrated design will be described further in the next chapter. The extent of the use of integrated design in the industry is yet unknown, however, at least in Denmark more and more consultants seem to start implementing at least parts of the concept.

The latest development on integration in the AEC industry therefore relates to both data integration and process integration and considerable consensus in the two areas seems to have been reached. The use of BIMs and IFC in the design process could be the way to allow for data integration, and the use of integrated design could be the way to allow for process integration. So far the development has, however, primarily been focusing on improvements in each of the two areas respectively, and the possibilities of using a combination of data and process integration seem to remain unclarified.

Only a combination of the two areas will represent complete integration in construction projects, and this should constitute the overall goal of improving the project results. For this reason, the aim of this thesis will be to investigate how a combination of data and process integration can be developed to ensure an improved project result. This has led to the following thesis statement:

1.2 Thesis Statement

The use of Building Information Models is gaining momentum in the AEC industry worldwide. At the same time, the demand for energy efficient and well climatized buildings increases. Using an integrated design process within the design team is seen as one of the best ways to achieve better performance of buildings during the design stage.

The aim of this thesis is to identify how an integrated design process can be improved by using BIM design, and in particular how a shared data repository defined by use of the current IFC specifications may support the improved process.

1.3 Scope of Thesis

Since the amount of activities in construction projects can be rather comprehensive, several delimitations apply to the scope of this thesis. The focus of the thesis is within energy and indoor climate conditions, and primarily the activities within the design phase are considered. Definition of client requirements, which should be completed before the design phase, is also of great importance to ensure proper energy and indoor climate conditions. However, in this thesis it will be assumed that proper client requirements have already been defined. The operational phase is also of great importance to energy and indoor climate conditions as the systems can easily be controlled differently than intended by the designers, however, this phase along with the construction phase will not be considered either.

Even within the design phase, the amount of activities can be very extensive and this thesis will therefore mainly describe the overall decision-making process and specific activities which relate to the focus on energy and indoor climate conditions. Other areas such as structural engineering, cost estimation and maintenance are also important when making design decisions, however, these will not be considered.

As mentioned in the preface, detailed research on the abilities to use geometry defined by use of the IFC specifications in simulation tools is ongoing at the IT University of Copenhagen. For this reason, the scope of this current thesis will be limited to only include handling of simple geometry.

1.4 Method

In order to achieve the goal of the thesis statement, the thesis is composed by four main chapters:

- Chapter 2 comprises a literature study on the latest development within research and implementation of integrated design and BIM design to identify the capabilities and demands for successful use of the two concepts. Based on these findings it will be analysed how BIM design could potentially improve integrated design.
- Chapter 3 contains an analysis of the capabilities of the current IFC specifications based on the identified demands from the previous chapter.

- Chapter 4 continues the exploration of capabilities of the IFC specifications by developing an example of an IFC interface for a simulation tool.
- Finally Chapter 5 summarises the findings from the previous chapters to define a methodology of how BIM design could improve integrated design. This is supplemented by a demonstration of how some current tools could be used to implement the methodology in building design.

Chapter 6 supplements the research by a discussion which intends to analyse the effort required to ensure the implementation of the methodology and how it can be ensured that the development in the future will continue with a faster pace than what has been the case in the AEC industry within the last 15 years.

1.5 Instructions for Reading

The author of this thesis previously participated in the creation of a Bachelor's Thesis, [Treldal & Johnsen 2005], concerning the creation of 3D object oriented models. To avoid too much repetition in this current thesis, [Treldal & Johnsen 2005] (written in Danish) will be used as a reference for details of areas already described here. Although this current thesis will briefly touch upon all subjects discussed, it will be assumed that the reader of the thesis has a prior knowledge of the possibilities of using 3D object oriented models. If this is not the case, [Treldal & Johnsen 2005] should be briefly studied to ensure an insight in the terms and methods used in this current thesis.

As the thesis additionally describes advanced solutions for optimising energy and comfort performance, it is expected that the reader furthermore has a prior knowledge in the area of construction engineering as terms of e.g. thermal performance will not be described to maintain simplicity.

For readers primarily interested in the methodology developed in the thesis, attention should be directed to Chapters 2 and 5 where the basis for the methodology is identified and the content defined. For readers primarily interested in the capabilities of the IFC specifications, attention should be directed to Chapters 3 and 4 where these capabilities are analysed.

Chapter 2

Literature Study on Integrated Design and BIM Design

2.1 Introduction

In the following chapter, the concepts of integrated design and BIM design will be defined and the status today of the two concepts will be discussed. Integrated design and BIM design are concepts which have been undergoing significant research and development within the last decade. The intention with the literature study is therefore to try and create an overview of the two concepts and then focus on the areas which are believed to be relevant for this thesis.

The literature study will focus on the processes in integrated design rather than on the technical solutions intended to be the outcome of the design. This relates to the fact that BIM design is believed primarily to be able to support these processes as opposed to the performance of the technical solutions.

For the same reason the literature study on BIM design will focus on its abilities to support and automate work processes within design teams. The concept of BIM design primarily relates to information management and sharing, and the focus will therefore be on solutions and techniques which support such possibilities.

2.2 Integrated design

2.2.1 The Need for Integrated Design

In order to understand integrated design, it is important to understand the work processes practised today in a traditional building construction project. Construction projects today are typically divided into phases relating to client requirements definition, design, construction and operation. The scope of this thesis relates primarily to the design phase, which can again be divided into sub-phases of conceptual design, scheme design and detailed design. The task specifications for each member of the design team during these phases are in Denmark defined in [PAR & F.R.I 2002] and are somewhat similar to the requirements found in other countries.

Such definitions of tasks have traditionally resulted in a process where the architect starts the conceptual design phase by working with different ideas for the building within the framework of the client requirements. The structural engineer and HVAC engineer will typically be involved in the project later in the design process when the architect has decided on an overall framework of the building layout and need expertise in order for the intended design solutions to perform and comply with the requirements from the client and the authorities. This results in a gradual starting point for the decision-making process of architectural, structural and HVAC design as illustrated in Figure 2-1.

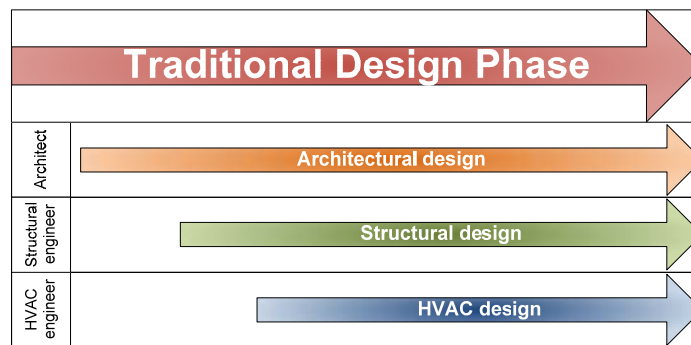


Figure 2-1. Traditional involvement of members in a design team during the design phase.

[Marsh 1997] argues that the use of mechanical services to produce a controllable and comfortable indoor environment within any building is almost unquestioned in modern architecture. It is further argued that together with artificial lighting technology, these capabilities underpin the majority of building design. This allow for the architect to be able to decide on many of the overall design principles without the assistances of engineers, because most design solutions can be supported by modern technologies to create a reasonable indoor environment.

Tightened requirements in building regulations in Europe caused by [EPBD 2003] require that the engineers improve their solutions in order to save energy and at the same time create better indoor environments. For the engineer to achieve this, [Löhnert et al. 2003] describes that because the engineer is bound by the overall design solutions defined by the architect, the engineer is limited to suggesting advanced and high-performance heating, cooling and lighting systems which might have only a marginal performance increase but a considerable capital cost increase. Even though the traditional design process is seen by many as quick and simple, the description above indicates that in respect to comfort and energy design solutions, this process only allow for sub-optimisation with limited performance increase.

To avoid the limitations of sub-optimisation, the focus for many researchers has moved to the conceptual design phase because it is believed that “in this stage, the most important decisions that determine the future of a building project are taken, and [...] 70 % to 80 % of capital resources are committed” [Rizos 2007]. To avoid sub-optimisation at various stages, overall design decisions should therefore be adjusted in the conceptual design phase in order to support solutions that meet the needs of all domains involved in the design team.

Some, e.g. [Rizos 2007] and [Marsh 1997], believe that the solution is to provide the architect with new tools that allow him or her to include consideration to such areas as sun distribution, thermal simulation and lighting calculation in the conceptual design phase. Both [Rizos 2007] and [Marsh 1997] agree that the architect as the sole conceptual designer is not sufficiently skilled to provide all required input to such simulations. For this reason they suggest that the tools used should be able to assist the designer in selecting the correct input data.

[Heiselberg & Brohus 2007] describes how a large range of solutions is required in order to achieve an energy consumption that is within the low energy classes of the Danish Building Code [BR95 2007] for three reference buildings. Advanced sensitivity analyses are used to find the appropriate solutions, and it takes a significant amount of thorough iterations to reach the energy consumption

required. The intention of the Danish Building Code is that the requirements for energy consumption are tightened with an additional 50 % before 2015 [Transport- og Energiministeriet 2005] and the solutions and techniques used in [Heiselberg & Brohus 2007] are therefore prerequisites for building design in the near future. The solutions developed requires the skills of an engineer, and if the solutions are to be implemented in the overall design, it is required that the engineer is engaged at the conceptual design phase along with the architect.

The acknowledgment that “the design of such a building often requires more and/or different types of skills and knowledge” [Hestnes 2007] is one of the main arguments for a new approach to the decision-making process in the conceptual design phase. Taking advantage of skills from several members of the design team from the beginning and throughout the design process is an approach defined as “integrated design”.

2.2.2 Concept of Integrated Design

The concept of integrated design is to include experts from each domain represented in the design team at the very beginning of the design phase in order to ensure that overall design decisions are adjusted to support for solutions that meet the needs of all domains involved right from the beginning of the process. The concept is illustrated in Figure 2-2.

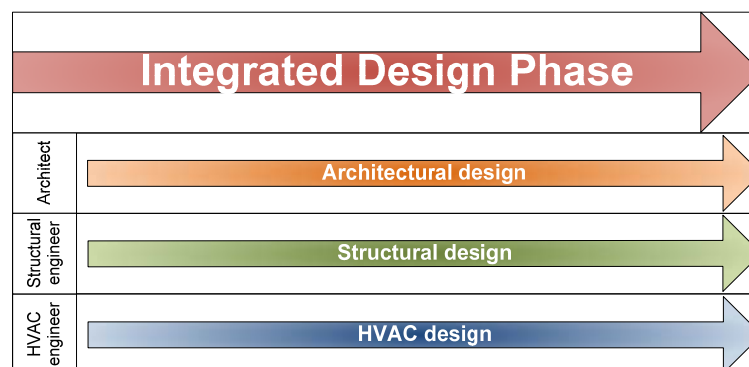


Figure 2-2. Work processes of members in a design team in an integrated design phase.

In order to ensure that integrated design fulfils the design goals required, the concept is refined by well-proven approaches that are put into a systematic process described by [Hestnes 2007] as:

- Uses a holistic approach
- Concentrates the effort on the earliest phases of design
- Focuses on the performance goals throughout the process.

In integrated design, it is further the intention that before the design phase even begins, the entire design team are to define the goals of the project in co-operation with the client and users [Hestnes 2007]. Integrated design is hereby intended to begin already in the client requirements definition phase and continue throughout the design process.

The most extensive research and testing of integrated design has been conducted within the project *Task 23 - Optimization of Solar Energy Use in Large Buildings* by The International Energy Agency as previously mentioned. The aim of this project was to optimise the use of solar energy in large buildings. However, the result of the research is believed to be able to generate improved building performance in many respects and across domains [Poel 2002]. [Löhnert et al. 2003] is one of the main documents describing the results of the research and will therefore be used as a main reference in the following.

2.2.3 Integrated Design Process

According to [Hestnes 2007] and [Löhnert et al. 2003] integrated design should be seen as a process and should therefore be referred to as an Integrated Design Process (IDP).

[Löhnert et al. 2003] describes the process as: "IDP is the entire, multi-disciplinary processing of a design task for which a competent design team continuously pursues durability-oriented targets from the beginning and optimises them in terms of ensuring quality through the application of modern methods and tools throughout each individual phase." This will also be the definition used in this thesis for the understanding of IDP.

The challenges of IDP is to allow a multi-disciplinary design team to develop and refine shared goals and be able to ensure that these goals are met throughout the entire design process. In order for this to happen, it is believed that the decision-making process of the design team must be redefined.

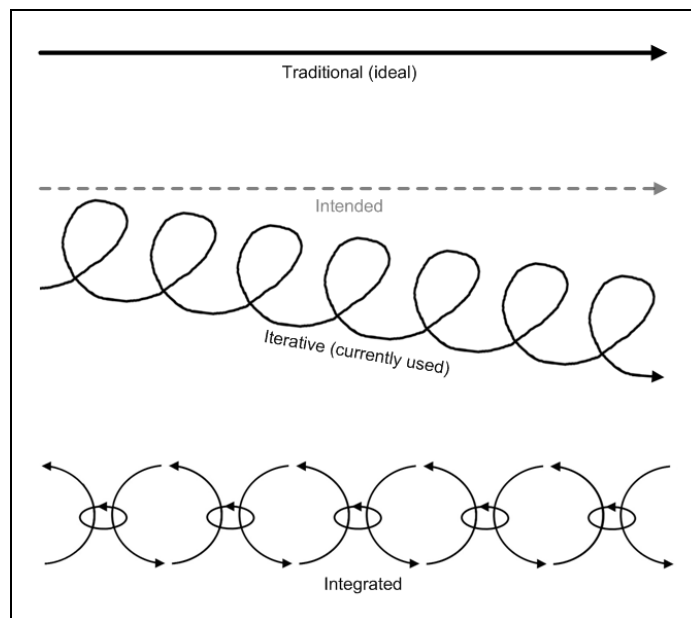


Figure 2-3. The linear procedure, the iterative procedure and the integrated procedure. Illustration inspired by [Löhnert et al. 2003] and [Kiviniemi 2005].

[Löhnert et al. 2003] describes that idealised traditional design usually is a linear procedure which does not allow for design optimisation. If the design needs to be assessed and optimised, it requires iterations which create an iterative procedure. This procedure is typical in modern design. A linear

procedure has a weakness of not being able to include potential improvements to the design and an iterative procedure has a weakness of being complicated to control because the iterations can easily cause the design to deviate from the original goal. The suggestion, illustrated in Figure 2-3, is therefore to combine the two procedures in an integrated procedure which allows for iterations but ensures that from each closed iteration the procedure moves closer to the design goals. The concept of these closed iterations is illustrated in Figure 2-4. The figure illustrates that each iteration is intended to create a basis for taking the next decision with respect to the design goals, any constraints that may apply for the construction (influences) and recommendations from specialists.

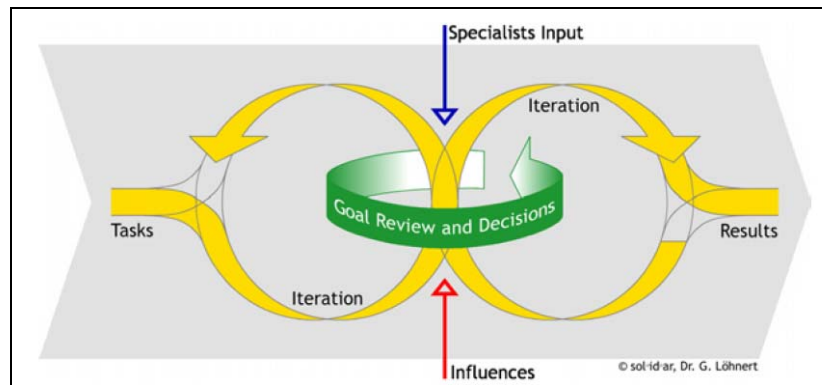


Figure 2-4. Iterations as a provider for problem-orientated analyses of design alternatives and optimisation [Löhnert et al. 2003].

The holistic approach of IDP supports the intentions of constantly moving closer to the design goals by allowing for iterations to vary in the depth of problem consideration from macro to micro and shift between problems and corresponding solutions. Each solution has been evaluated thoroughly and with the right level of detail. These solutions can therefore create milestones which should be understood as “point of no return” [Löhnert et al. 2003] and which ensure that future iterations will not cause deviation from the design goals.

Each phase of the design process consists of minimum one iteration, but it could also consist of several iterations to allow for a steady development of the construction project within each phase. It is believed to be the task of the project management to ensure that decisions on milestones are documented in order to avoid the loss of information in the transitions from one iteration to the next [Löhnert et al. 2003]. It is further argued that a skilled design facilitator should be included in the design team to manage this task and to ensure proper development in the design process.

[Poel 2002] summarises the experiences gathered from five projects where IDP has been used. It is concluded that IDP can be very effective to achieve the design goals and that IDP can be used for various design tasks in different contexts. It is described that “the first stages of the design process may be a little more time consuming and costly, but inefficiencies in the following part of the design process will be avoided and the overall cost performance ratio of the building improves.” In order to use IDP successfully, [Poel 2002] mentions a large range of recommendations which relate to the composition of the design team and the process. Most findings relate to the skills and commitment of the design team members, however, it is also mentioned that quality of communication across domains and special attention to the exchange of information is of great importance for IDP to be successful.

2.2.4 Use of Simulation Tools in IDP

In order to create a proper basis for decision, both [Hestnes 2007] and [Löhnert et al. 2003] argue that it requires the use of modern methods and tools. In relation to this thesis, the methods and tools required are described by [Rizos 2007] as Energy and Environmental Design Decision Support Systems (EEDSS). These are categorised in four groups:

- Design guidelines or rules of thumb – for general design advice
- Traditional physical calculation methods – focusing on a limited number of physical phenomena
- Correlation based methods – considering all physical aspects that influence a certain building performance with restrictions in design specification and performance assessments.
- Building simulation – creating a virtual building where the user in detail can specify parameters that influence the building performance, resulting in performance predictions that are as close to reality as possible.

In [Marsh 1997] it is argued, that in relation to computer (building) simulations “the increased number of design parameters that can be considered and the level of detail to which performance can be simulated is far beyond the scope of any manual method.” The holistic approach of IDP requires that the tools selected are capable of handling such a large range of design parameters and the use of computer simulations throughout IDP is therefore inevitable. The use of more simple EEDSS such as design guidelines are still required in order to have sufficient input data for simulations right from the beginning of the IDP. Therefore it is up to the skilled members of the design team to navigate through the large range of EEDSS in order to find tools that are capable of generating the basis of decisions required.

The need for iterations right from the beginning of IDP further requires that the tools selected are capable of handling different levels of details of the design solutions to be analysed. At the conceptual design stage, it is typically required to analyse various alternatives of building orientation, insulation of building envelope, glazing area, thermal mass, shading devices, floor plan depth, daylight distribution, ventilation concepts and space usage [Rizos 2007]. Later, as the design progresses, a need emerges for detailed evaluation and optimisation of the building envelope, ventilation principles, heating systems, air movements etc.

Many computer simulation tools have been developed over the years to support the various needs of the design team. [DOE 2007] currently holds a list of more than 350 energy simulation tools, which indicates that the range of possibilities is significant. Within the last decade, rapid development has additionally resulted in a range of new simulation tools intended to be used in the conceptual design phase making it possible to use simulation tools throughout IDP.

The following list includes some of the simulation tools which could be used to support the tasks mentioned above. Many other simulation tools include similar capabilities as those mentioned in the list, and there is additionally a large range of specialised tools capable of simulating other design details and issues such as sustainability issues.

Simulation tools

- *Ecotech* – a software package which is intended to be used in the conceptual design phase. It couples a modelling interface with a set of performance analysis functions. It includes abilities to analyse areas such as solar, thermal, lighting, shadows & shading design, energy & building regulations, acoustics, air flow, cost and resources [Square One 2007].
- *Green Building Studio* – a web-based energy analysis solution intended for the conceptual design phase which integrates with many 3D-CAD applications. It is capable of evaluating the CO₂ use of the building and of analysing the influence of different building orientations, ventilation systems and glazing area among others.
- *Energy 10* – an energy simulation system that provides predictions of operating energy performance and identifies the most effective design strategies in reaching this performance level [Poel 2002]. Energy 10 has been developed to support IDP particularly.
- *IES Virtual Environment* – a suite of dynamic performance analysis tools which are all linked to one single building model. It allows for simulations on various levels of details in areas such as thermal, lighting, solar distribution, CFD and cost.
- *BSim* – primarily a thermal simulation tool, however, it also allows for simulations of among others, solar distribution, moisture transport, natural ventilation and energy use.,
- *Flovent* – a CFD tool capable of simulating movements of fluids. Typically used to evaluate air movements in buildings and to assess the indoor climate.
- *Heat 2D/3D* – a specialised set of simulation tools which is capable of analysing transient and steady-state heat conduction in constructions in 2D and 3D respectively. Used primarily to analyse thermal bridges.
- *Be06* – although primarily intended to be used as a tool to prove energy compliance with requirements in the Danish Building Code, the tool can also be used during design to simulate the energy consumption for optimisation purposes.

Figure 2-5 indicates the potential use of the simulation tools in IDP. The holistic approach of IDP typically requires the use of tools capable of making overall analyses in the early part of the design phase and tools capable of detailed analyses later in the phase when decisions of design details are taken. The use of such tools is typically based on specific needs in different projects, and the illustration is therefore only an indicator of typical usage.

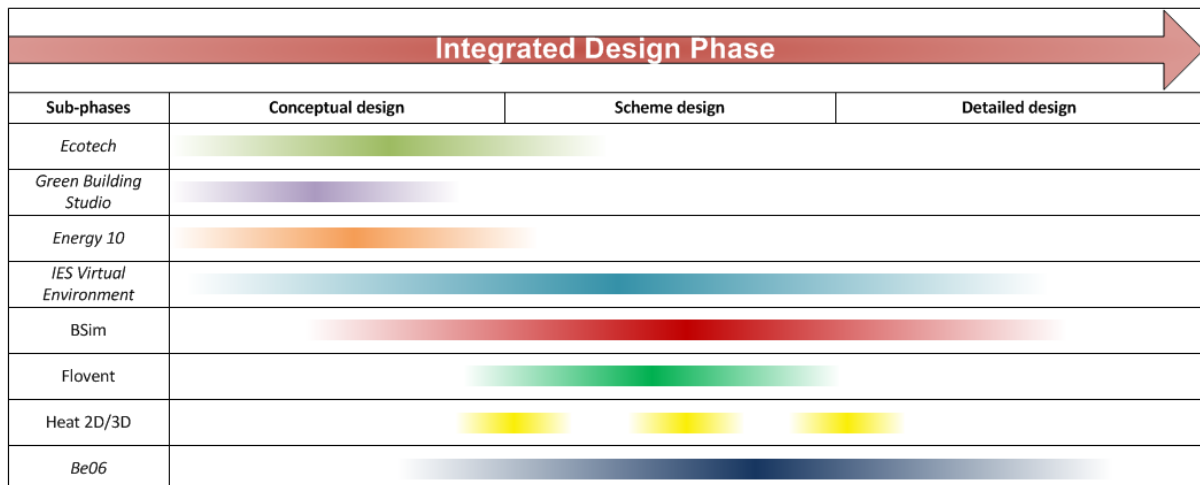


Figure 2-5. Potential use of simulation tools in IDP. Based on needs for tools capable of making overall analyses in the early part of the design phase and tools capable of detailed analyses later in the phase.

The use of simulation tools in IDP results in a process where shifting simulation tools are used through the design phase. Due to various capabilities of the tools, several tools could be used simultaneously.

2.2.5 Space of Solutions

Ongoing research at DTU¹ has led to the development of yet another simulation tool for the conceptual design stage: iDbuild. This simulation tool is intended to support a specific refined concept of the integrated design process. The concept intends to provide the design team with a basis for decision by creating a so called “space of solutions”. The space of solutions is intended as a set of boundary conditions for the design decisions which will ensure optimal comfort and energy performance [Svendsen et al. 2007]. In relation to this thesis, the concept is interesting because this is a way of working with iteration which can be closed in IDP right from the beginning as the boundary conditions can gradually be narrowed as the design progresses.

The hypothesis of the research at DTU is that optimal room design is the key to high building performance in relation to energy and comfort. In the conceptual design phase, the focus should therefore be upon generating a space of solutions based on the client requirements for the rooms which are now the design goals. When the room layouts have been defined, the next task is to combine the rooms to a proposal for the total building design. Followed by additional optimisation of the total building, this should then lead to a result which fulfils the design goals. Similar to the general approach of IDP, it is argued that this concept requires a skilled design facilitator to ensure that the design progresses as intended and within the space of solutions. The concept is illustrated in Figure 2-6.

¹ More information on the research at DTU can be found here:
<http://www.dtu.dk/centre/IRS/Current%20research/STP.aspx>

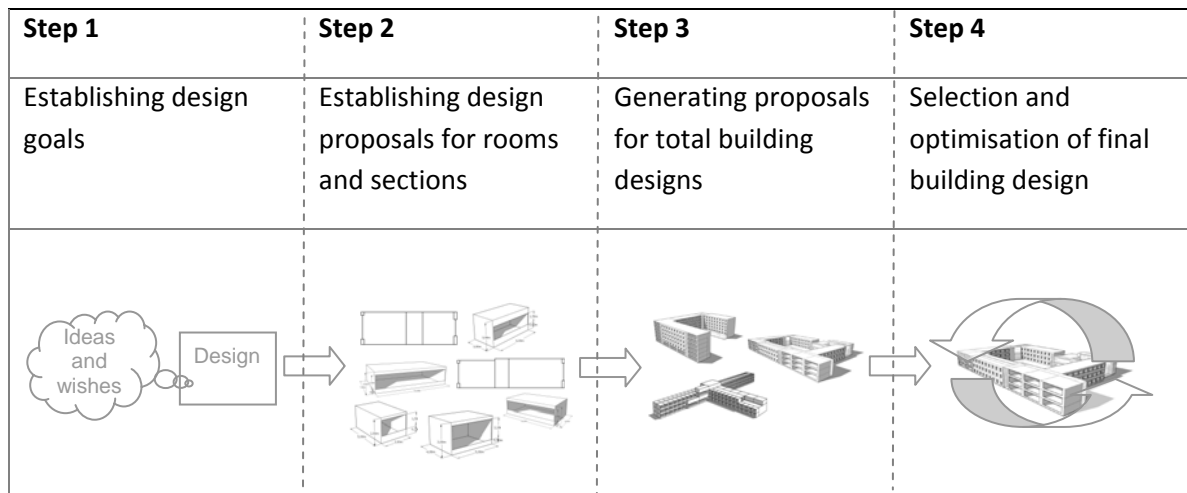


Figure 2-6. The total integrated design process [Svendsen et al. 2007].

The simulation tool iDbuild is intended to assist in identifying constellations of performance-decisive parameters which fulfil the design goals [Svendsen et al. 2007]. These performance-decisive parameters define the boundaries for the layout of the rooms including an optimal design of the attached façade. It is constraints for these parameters which constitute the space of solutions. A list of constraints for parameters believed by [Petersen & Svendsen 2007] to constitute a space of solutions in relation to this concept is listed in Appendix 1. As the scope of IDP is greater than energy and comfort performance targets, the space of solutions should consist of constraints for more parameters to include other areas of concern such as structural design, cost estimation or maintenance issues. When the design progresses, the space of solutions could be extended by further sets of constraints to define additional design boundaries as they apply. Ongoing debate of the importance of various parameters makes it difficult to unambiguously define a prioritised list of constraints in a space of solutions and it must therefore be up to the design facilitator to define the parameters in the space of solutions based on the design of the current project.

iDbuild is, similar to Ecotech, intended to analyse solar distribution, energy use and thermal performance in the conceptual design phase. iDbuild does, however, differ from other simulation tools by its ability to conduct parameter analysis which provide the designers with an improved basis for decision. An example of a parameter analysis can be found in Figure 2-7.

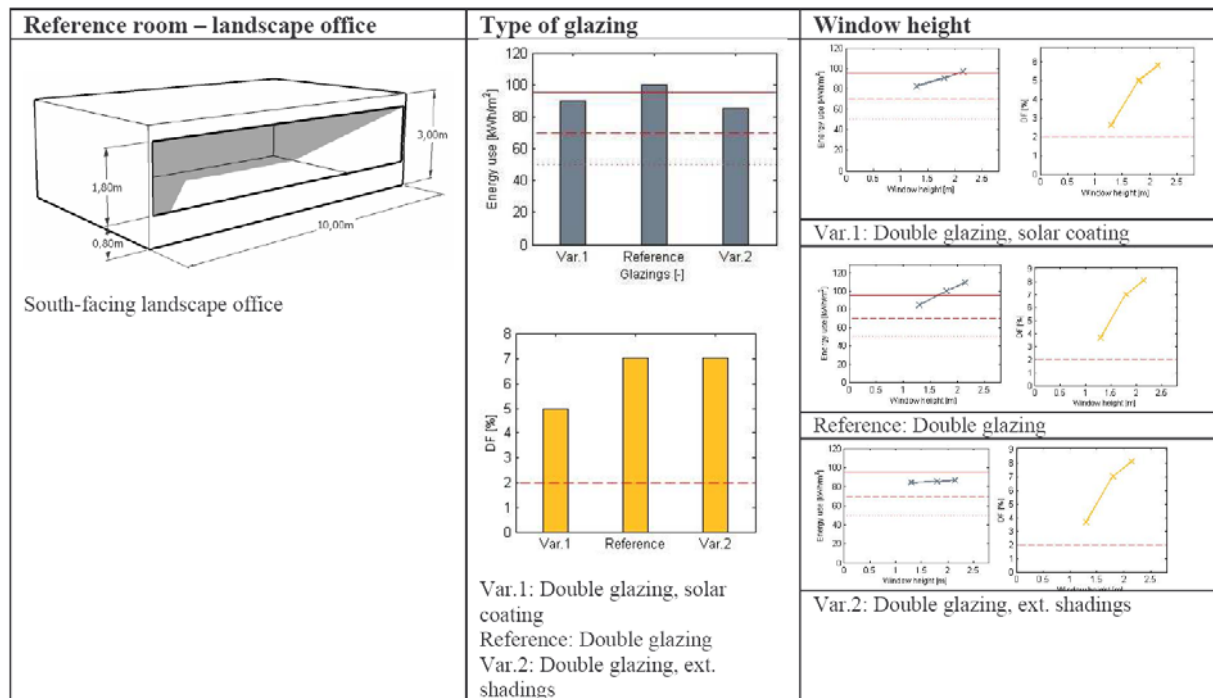


Figure 2-7. Results of parameter analysis in iDbuild [Petersen & Svendsen 2007]. In this case the window height has been analysed for three types of windows. On the right side, energy use (blue) and the daylight factors (yellow) are illustrated for each window analysed.

Such a parameter analysis is intended to allow the design team to decide on design boundaries based on an overall consideration to the energy and comfort performance. This is an important possibility in the holistic approach of IDP. From the results it is intended that the design team is to select one or more optimal room layouts with a fixed set of parameters and by combining an appropriate selection of these room layouts the total building can be created.

In [Svendsen et al. 2006] it is argued that such parameter analysis could also be used to create a more flexible space of solutions by setting up intervals in which constraints for parameters are able to vary without weakening the building performance. In relation to the example in Figure 2-7, this could for example result in an allowed interval of variation in window height. These would allow for the interval of the second variant of windows (including external shading) to be bigger than for the other window types. The change in height of this window is less sensitive to the change in energy use compared to the other windows, and it is the argument of this thesis that this should allow for more flexibility in the design. One of the main challenges for the concept of using a space of solutions is believed to be that the architect could be limited too much in creating architectural design if he or she is provided with a space of solutions with fixed constraints only. The use of constraints consisting of intervals for performance-decisive parameters is therefore preferred.

2.2.6 Sensitivity Analysis

One issue, with regard to the flexibility of parameters, is that iDbuild only allows for parameter analysis of one parameter at the time, such as window height. This only allows for one parameter to include a constraint with a flexible interval per room in the space of solutions because the variation

in more parameters will cause performance results which are unknown based on the analysis conducted. This parameter analysis is a type of sensitivity analysis which [Heiselberg & Brohus 2007] describes as a local sensitivity methods. Global sensitivity methods are another approach described where “output variability due to one design parameter is evaluated by varying all other design parameters as well” [Heiselberg & Brohus 2007]. One such global sensitivity method has been successfully used in a conceptual design phase by [Struck and Hensen 2007]. By combining local and global sensitivity analyses, they were able to illustrate both individual sensitivity and correlation of design parameters. Such results could potentially allow for the creation of varying intervals of more constraints in the space of solutions and hereby create the desired flexible approach to the decision-making process. The space of solutions could thereby consist of a variation of fixed and variable constraints for performance-decisive parameters which the design team will be bounded by during their design decisions.

2.2.7 Summary on Integrated Design

IDP is hereby identified as a concept where the typical iterative design process is broken down into closed iterations to ensure that the design goals are met. Design decisions in one iteration create milestones which form the framework for the following iterations. IDP uses a holistic approach supported by a range of simulation tools to ensure that the defined milestones will not cause deviations from design goals as the design progresses. The milestones could be a space of solutions consisting of constraints for a range of performance-decisive parameters in relation to e.g. a room layout. Such constraints can be narrowed as the design progresses and continuously be extended to cover more parameters and thereby allow for flexibility within well defined boundaries.

It is important to notice that since IDP is not intended to change the way information is shared, the procedure will most likely follow the traditional way of exchanging information primarily via documents and verbal communication.

2.3 BIM design

2.3.1 The Need for BIM Design

The AEC industry today is not only facing the challenge of creating buildings with increased performance. The industry has for many years suffered from a decrease in productivity both in Denmark and internationally. A comparison of growth rates for labour productivity² in Denmark from [OECD.Stat 2007] reveals that the construction industry experienced an average annual growth rate of -0.7 % between 1990 and 2005. In contradiction to this, the non-farming industry³ experienced an average annual growth rate of 2.5 % in the same period. The development is illustrated in Figure 2-8.

² According to [Regeringen 2003] labour productivity per unit labour input is an appropriate way to describe productivity in the construction industry.

³ The non-farming industry is assumed to consist of the industry and manufacturing companies.

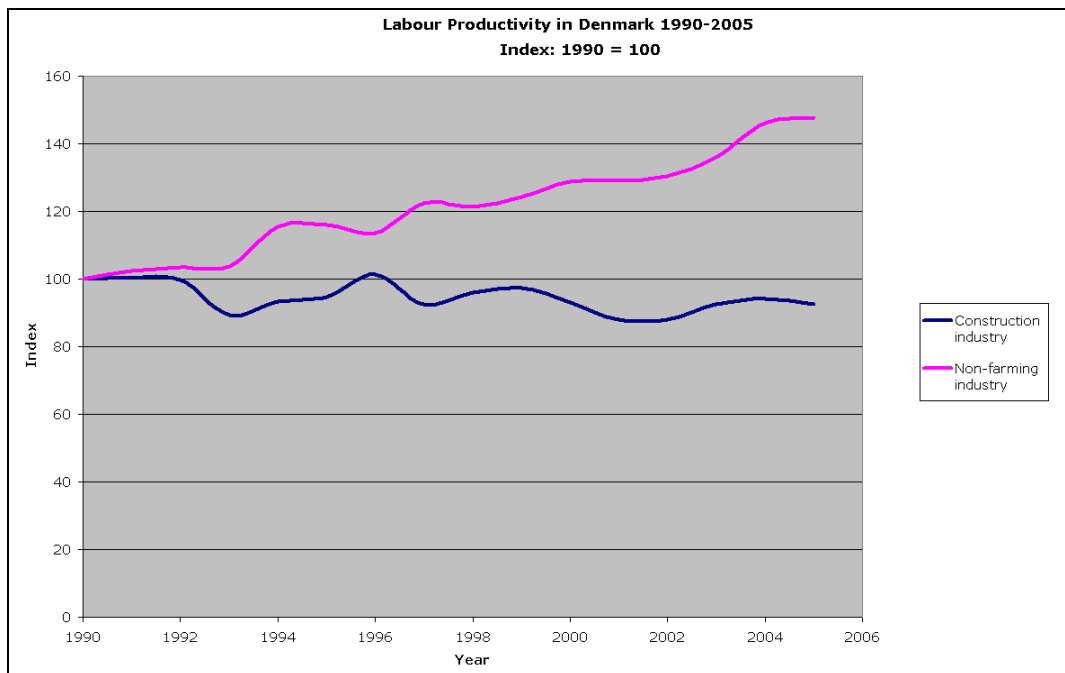


Figure 2-8. Labour Productivity per Unit Labour Input per Hour in construction and non-farming industry in Denmark based on information from [OECD.Stat 2007].

[NIBS 2007] describes a very similar development for productivity in the US with decreasing productivity in the AEC industry and a general increase in productivity in other industries. It therefore seems as if the productivity has stagnated in the AEC industry.

In order to improve the productivity of the AEC industry, focus has been attended to the value chain of the industry. The fragmented industry is characterised by a complex process and by having projects with temporary constellations of varying companies [DAC 2006]. One of the challenges of such temporary constellations is that it is complicated to share and maintain information in the value chain [DAC 2006]. This acknowledgment has been one of the primary drivers for the creation of Digital Construction⁴ (in Danish: Det Digitale Byggeri) in Denmark. The vision of this project is that data needs to be able to be stored, processed, copied, reused and extracted based on desired needs [translated from EBST 2006]. In the US, the creation of a National Building Information Modeling Standard is based in the same idea by intending to create “a framework and foundation to encourage the flow of information and interoperability between all phases of a facility’s life” [NIBS 2007]. The need for such initiatives is supported by [NIST 2004] which indicates that the lack of abilities to share information in the AEC industry annually generates a loss of \$15.8 billion in the US alone.

The ability to share information within the AEC industry is termed *interoperability* and defined by [NIST 2004] as “the ability to manage and communicate electronic product and project data between

⁴ More information on the project “Det Digitale Byggeri” can be found in [Trelidal & Johnsen 2005] and on www.DetDigitaleByggeri.dk. In spite of the intentions of the project, the ambitions so far have only led to the requirements of a 3D object orientated approach [bips 2006] which is focusing primarily on the geometry of objects and less on other properties.

collaborating firms' and within individual companies' design, construction, maintenance and business process systems." Within the AEC industry, one of the most widely accepted concepts of facilitating interoperability is through the use of a Building Information Model (BIM).

2.3.2 Concept of BIM Design

BIM has over the years been interpreted in many ways by various organisations and companies. [Erabuild 2008] has a list of four such interpretations but chooses to define BIM in its own way by: "BIM (Building Information Model) is an object-oriented, AEC-specific model – a digital representation of a building to facilitate exchange and interoperability of information in digital format. The model can be without geometry or with 2D or 3D representations. Integrated BIM, when you share information, requires open standards." This definition will also be used in the following for this thesis.

This definition of BIM implies that a BIM does not need a specific level of detail or to reach a certain maturity in the design process before it is valid. A BIM can therefore be created right at the earliest steps of the conceptual design phase and be used throughout the building lifecycle to facilitate information interoperability in the entire process.

[Bazjanac 2006] describes that BIM has a meaning both as a noun and as verb because BIM could also indicate Building Information Modelling which is the act of creating a Building Information Model. This thesis separates the two terms by saying that *BIM design* is the act of creating a Building Information Model in the design process.

The idea of BIM design is therefore to create a digital representation of a building consisting of objects. An object is a physical or imaginary item to which properties can be added [Tredal & Johnsen 2006]. Such properties could be information about geometry, placement, materials, performance data, relations etc. The concept of an object is illustrated in Figure 2-9.

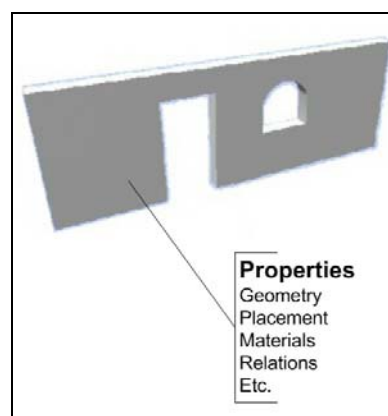


Figure 2-9. An object and some of its properties.

The use of BIM has facilitated an important change in the design of buildings compared to the traditional way of designing buildings in CAD software because space (room) objects and zones are now a fundamental part of the modelling process. [Marsh 2006] argues that the use of spaces and

zones “fundamentally challenge [...] the way CAD tools organise building information and how designers must generate their building models”. This change has, however, ensured that a BIM model is capable of including spaces which is of crucial importance when analysing energy and comfort performance as indicated in section 2.2.5 *Space of Solutions*. Systems and space boundaries are two other concepts used in BIM design, which are important in relation to analysing energy and comfort performance. Systems allow for several objects to be grouped and are used primarily in relation to building services. This allows for all objects in e.g. a piping or ventilation system to be identified as a whole. Space boundaries are used to create a connection between a space and the objects acting as its boundaries and are useful because such information is important for thermal simulation purposes. Space boundaries are typically generated automatically by the applications used to create the BIM.

To create a BIM, one will need a CAD tool which is capable of working with objects. [Erabuild 2008] mentions 27 dominating BIM supporting applications capable of creating objects in a BIM in architectural, structural, building services, construction and facility management domains. The scope of this thesis is within the architectural and HVAC domains where some of the following BIM applications are widely used:

BIM applications

- ArchiCAD – an architectural application capable of creating object oriented models contained in a database which holds all information about the model created.
- AutoCAD Architecture – a file based architectural application which is able to create object oriented models and store information of partial models in files which combined constitutes the whole model.
- Revit Architecture – an architectural application capable of creating object oriented models contained in a database which holds all information about the model created. Additionally Revit Architecture is capable of handling parametric constraints of geometry within and among objects.
- MagiCAD – a file based application for building services engineers which functions on top of AutoCAD Architecture and allows for the creation of a large amount of building services objects taken from manufacturer libraries. MagiCAD also allows for limited customisation of additional objects.
- AutoCAD MEP – a file based application for building services engineers which functions on top of AutoCAD Architecture. AutoCAD MEP allows for the creation of objects from a large amount of predefined building services components available. It also allows for the creation and customisation of additional objects.

The intention of creating a BIM is to be able to share the information available in the BIM with a large range of software applications used throughout the facility's life as stated in [NIBS 2007] and thereby create interoperability in a digital format. The BIM software mentioned above is thereby intended only to be a part of the applications expected to interact with the BIM. Other software used

for simulations, cost estimation, facilities management etc. are typically only used to extract information from the BIM and occasionally edit and/or supplement the object properties of the BIM. All software should, however, have equal abilities to interact with the BIM. The concept is illustrated in Figure 2-10. By improving the possibilities of interoperability, the intention is to increase the use of design and simulation tools throughout the design phase [Tredal & Johnsen 2005] and hereby achieving better quality of the final design.

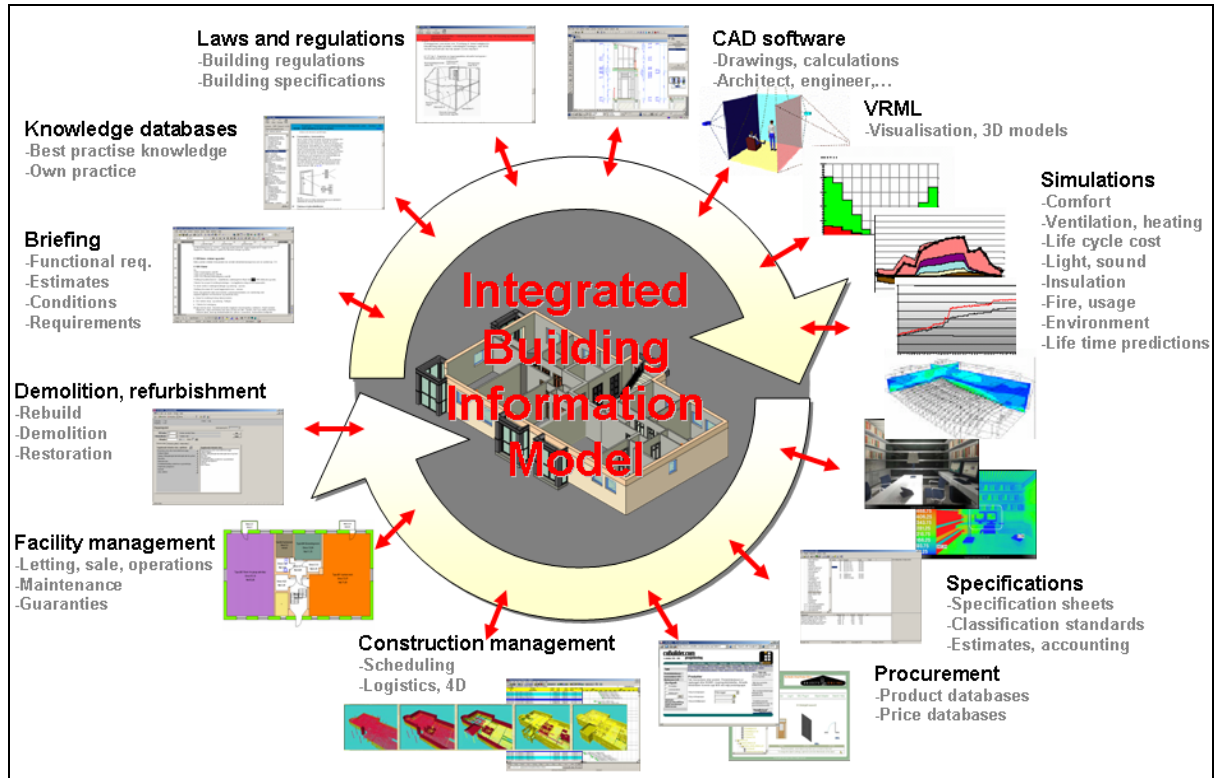


Figure 2-10. Concept of integrated BIM. Illustration originally made by Norwegian Building Research Institute, Olof Granlund, NBLN University of California and Stanford University.

The experience gained in using BIM design in actual construction projects vary both in terms of success and in the ability to use a BIM for more than just the exchange of 3D geometry. [Bazjanac 2005] and [Fischer & Gao 2005] describe how the use of BIM has generated improvements in terms of increased efficiency in the design and construction process and improved quality of final design. The improvements are primarily a result of increased interoperability which allow for an increased use of existing design tools and the inclusion of new advanced tools. [Bakkmoen et al. 2007] and [Haug 2007], however describe, that the use of cutting edge technology in BIM design has caused several complications and required that the level of ambitions had to be lowered for their projects to progress. The experience indicates that the use of BIM design should result in improvements to both the quality of the final design and efficiency in the design process. However, the lack of maturity of technical solutions can be a challenge in reaching this goal.

As stated in the definition, BIM is described as an integrated BIM when used for sharing information with other software. The definition also states that in such case, the information sharing should be via open standards. The main reason for such a requirement is that the alternative would be using proprietary standards owned by one software vendor. Such proprietary standards will limit equal

abilities to interact with the BIM by software from other vendors and is not useful for the concept of BIM design. The most widely recognised open standard for information exchange of a BIM is the IFC specifications [NIBS 2007].

2.3.3 IFC as the Shared Data Repository

The Industry Foundation Classes (IFC) specifications have, as previously described, been developed by the International Alliance for Interoperability and support the concept of object oriented building models⁵. The specifications cover areas such as architecture, building services, structural engineering and facility management. Objects in the BIM can be stored in an IFC file according to the specifications and accessed by any other software via interfaces capable of interpreting this open standard. IFC is thereby capable of acting as a shared data repository for the integrated BIM ensuring interoperability among various software. IFC files can either include the entire integrated BIM or partial models of for example architectural or HVAC objects.

Other open standards such as gbXML and CIS/2 exist for the sharing of information in a BIM⁶. These standards are, however, specifically intended to support information sharing in areas of energy simulations and structural design respectively. Although [Dong et al. 2007] argues that gbXML is easier to use and implement than IFC in a HVAC perspective, IFC has an advantage of being the only standard with a “scope of covering the whole lifecycle for buildings” [Erabuild 2008]. Although the scope of this thesis is within the range of capabilities of gbXML, the IFC format is selected as the standard for information sharing. This is primarily because IDP and BIM design are intended to cover the entire design phase across all domains. To define a methodology based on several standards would create weakened data integrity of the BIM, and IFC is therefore the only appropriate standard to use in this thesis.

The latest release of the IFC specifications is version 2x3 which was published in February 2006. It is this version which will be used as the basis for the thesis. The specifications are freely available in [IAI 2007a] and consist of the content illustrated in Figure 2-11. The content of the IFC specifications is separated into four layers, consisting of a core layer in the centre, an interoperability layer and domain layer in the top and a resource layer in the bottom. The part of the specification relevant for this thesis relates to the HVAC domain, the electrical domain, the building controls domain, the shared elements, the extensions, the kernel and the resources.

⁵ Details on the development of the IFC specifications can be found in [IAI 2000].

⁶ More information on the standards can be found at:

gbXML: www.gbxml.org/about.htm.

CIS/2: www.cis2.org.

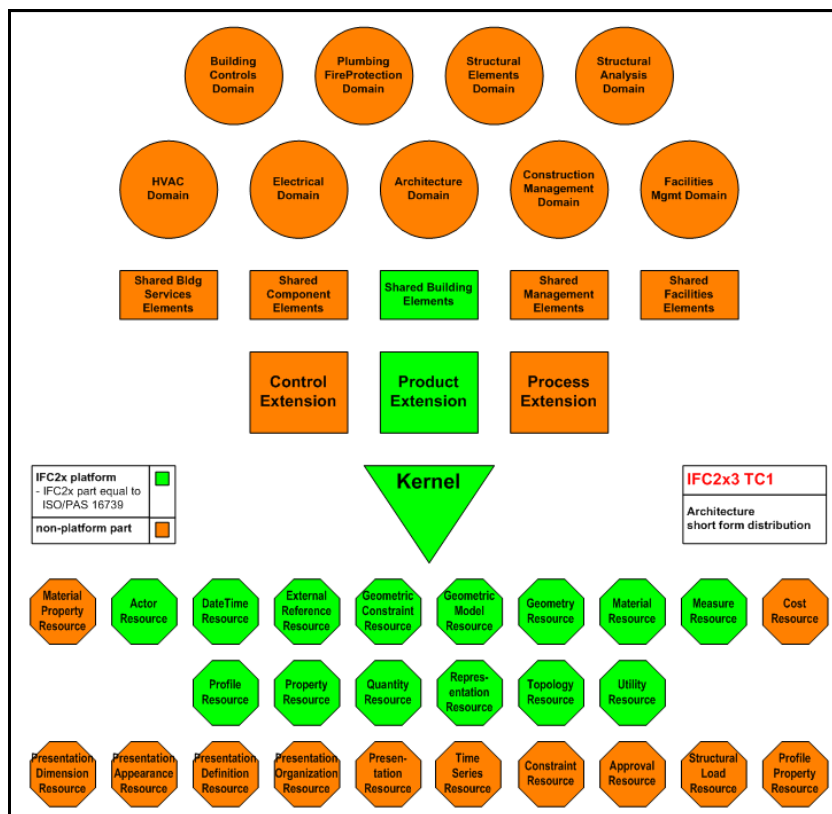


Figure 2-11. Composition of the IFC 2x3 specifications [IAI 2007a].

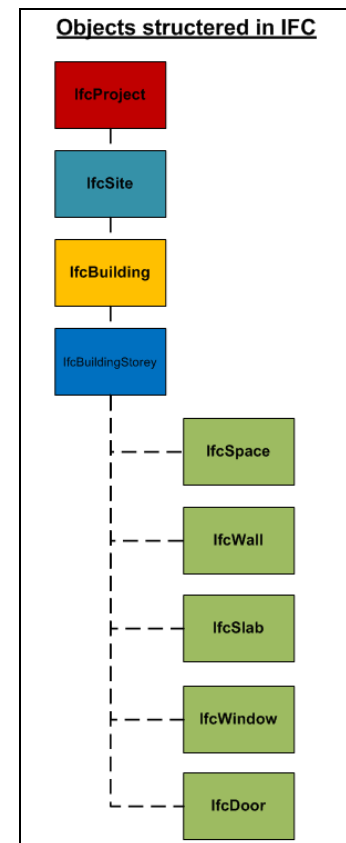


Figure 2-12. Hierarchical structure of typical objects defined in IFC.

Objects in IFC are defined as a subset of classes called *entities* represented by types such as IfcWall, IfcDoor and IfcWindow. All entities are organised in a hierarchical structure as illustrated in Figure 2-12. Properties of an object are either assigned to the entity or attached as attributes consisting of a new set of entities. The graphical modelling notation EXPRESS-G is used to identify classes, the data attributes of classes and the relationships that exist between classes in IFC [IAI 2007b]. The basis hierarchy of defining building elements in IFC based on the EXPRESS-G format is illustrated in Figure 2-13. The entity IfcRoot is a supertype for the entity IfcObject and contains fundamental properties for its subtypes. One such subtype is IfcObject which again is a supertype for IfcProduct and so on. This hierarchical structure characterises IFC as a top-down specification and is an important concept for the understanding of the composition of objects in IFC.

Other attributes such as relations between objects and type related properties are also defined as entities. Such attributes are, however, applied to the object by inverse attributes which is a different way of assigning attributes to one or more objects. This convention allows to encapsulate the object class definitions, which could be distributed without the relationship objects in valid sub (or partial) models [IAI 2004]. The composition of a wall object is illustrated in Figure 2-14. The figure illustrates how the current wall has a range of properties attached such as global unique ID (GUID), ownership, name, placement and representations. These attributes have been inherited from the wall's supertype. Additionally, the wall has five inverse assigned attributes which in this case relates to additional properties defined in property sets, material composition and a space boundary among others.

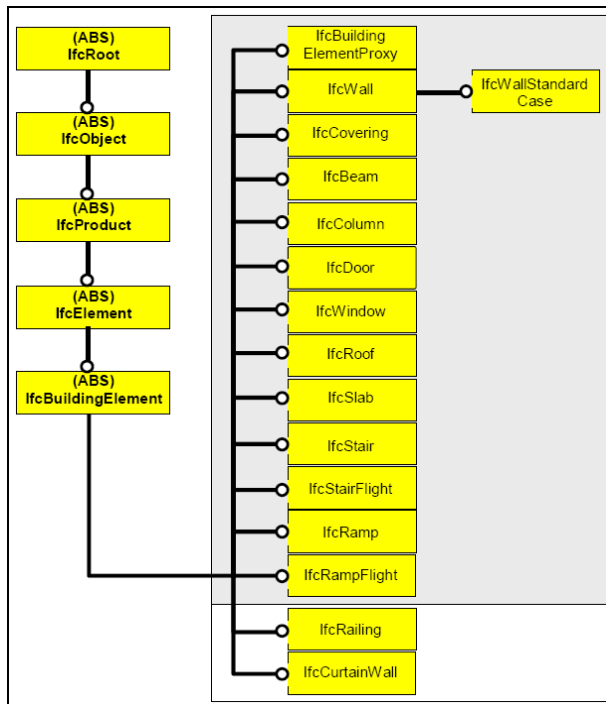


Figure 2-13. Hierarchy chart of building elements defined by EXPRESS-G notation [IAI 2004].

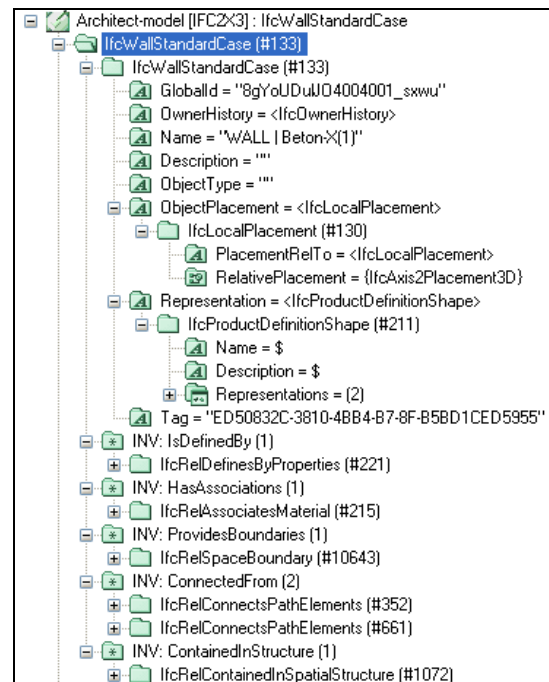


Figure 2-14. The composition of a wall object in IFC.

Today IFC data will typically be stored in IFC files which can be defined using either STEEP Part 21 or an ifcXML standard. The STEP Part 21 standard is the most widely used way of defining IFC data and refers to the STEP definition from which the IFC specifications originate. ifcXML is not supported by many tools, but allow for IFC data to be stored according to the XML specifications. Using either standard only varies the way the IFC data is stored, as the content of an IFC file will be identical and independently of the standard [Treldal & Johnsen 2005]. When IFC data is placed in a database, such as a model server described later, it will typically be stored according to the specifications of the database. The content should, however, be the same as if the data was stored in an IFC file. The placement of IFC data therefore only relates to the requirement for accessibility to the data and not the content.

Several development projects have extended the range of the IFC specifications over the years. In relation to this thesis, the most interesting of these projects is the Building Services project number 8 (BS-8)⁷ which aimed at supporting “the exchange of HVAC information between software currently used by the AEC/FM community (e.g., CAD) and various building services simulation tools” [IAI 2001]. Ongoing development is, however, continuing to improve the specification and to standardise the way objects and attributes are defined in IFC.

⁷ The current possibilities of IFC including the implementation of BS-8 will be investigated much further in chapter 3.

2.3.4 Information Delivery Manual and Model View Definitions

Two of the most important initiatives in the continuing development of IFC are the Information Delivery Manual (IDM) and the Model View Definitions (MVD).

The IDM project is intended to develop a set of exchange requirements throughout the building construction process in order to make sure that information is available when required in the process. This is done from the assumption that “if the information required is available when it is needed and the quality of information is satisfactory, the construction process will itself be significantly improved” [BuildingSMART Norway 2007]. Although the intention of this thesis was to include the work of the IDM, it has been decided not to do so because the development of the IDM has been considered not to have reached a proper stage of maturity. Energy analysis is an area where some development has been conducted, however, so far only the first out of four stages in the design process has been properly described.

Supplementing the work on defining the IFC standard, IAI has decided also to develop a range of Model View Definitions (MVD) which is intended as a way to document IFC based data exchange capabilities in software [IAI 2007c]. Three MVDs are interesting in relation to this thesis:

- **Extended coordination view**
General agreement on how geometry information is to be exchanged.
- **Architectural design to thermal simulation**
The central information in this view relates to space geometry, space boundaries, construction types and materials of common building elements.
- **Indoor climate simulation to HVAC design**
The central information in this view is the simulation results, i.e. the design values for spaces.

Only the Extended Coordination View has yet been finalised and implemented in software. So far 11 major BIM applications have been certified for this MVD. From the list of BIM application in section 2.3.2 *Concept of BIM Design* only AutoCAD MEP has not yet been certified for the extended coordination view. The development of the two other MVDs has only just started and results of these views are so far limited.

The IFC specifications are therefore seen as an open standard which should already be capable of acting as a shared data repository for the integrated BIM throughout the design process, although further improvements to the standard and greater consistency in the software implementation are in the making.

2.3.5 Centralised Data Repository – Model Server Concept

In relation to the concept of sharing information, the traditional way of exchanging IFC files is also undergoing interesting development. The traditional way of exchanging IFC files has typically required members of the design team to exchange partial models of the BIM containing the objects which they created and were responsible for. Combining the partial models would constitute the

complete BIM containing all objects and information available. Many have argued – among others [Erabuild 2008], [Kiviniemi 2005] and [Trelidal & Johnsen 2005] – that in order to facilitate effective information sharing among all parties during for example the design process, a centralised data repository defined using the IFC specifications is required as a way to provide access to the integrated BIM. Such a centralised data repository is defined as a model server consisting of a database where information is stored. The intention is to allow all information to flow via the model server and thereby optimise the flow in accordance with the needs of each design team member [Trelidal & Johnsen 2005]. By uploading partial models to the model server, each member contributes to the enrichment of the integrated BIM located on the model server, and all members of the design team then have the ability to extract the information they need directly from the model server. The concept is illustrated in Figure 2-15.

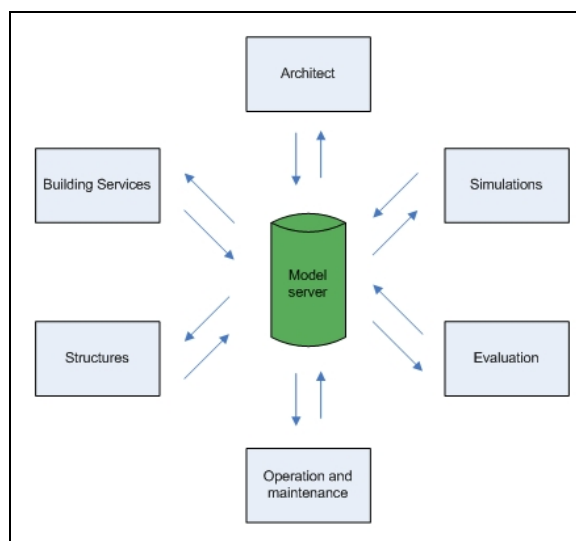


Figure 2-15. Information sharing using a model server.
Illustration originally made by [Trelidal & Johnsen 2005].

Several model server solutions based on the IFC specifications have been developed with similar capabilities. Two of the most widely used are EPM IFC Model server from EPM Technology and EMS Eurostep ModelServer for IFC from Eurostep. So far experience has shown that the technical capabilities in these solutions are not mature for professional use [Erabuild 2008]. Model servers have, however, been used in several test cases – [Bakkmoen et al. 2007] and [Haug 2007] among others – where the concept has been acknowledged although technical barriers have limited their use.

One of the advantages of using a model server is an improved capability to monitor and manage the information flow [Erabuild 2008]. Because new information to the BIM coming from design decisions is uploaded to the model server hereby instantly being reflected in the integrated BIM, the ability to track changes in a manageable environment is much greater than in the traditional file based exchange where the integrated BIM could be more loosely defined. As the database in a model server is additionally very flexible in its structure it could allow for several models to be stored on the same model server which could be used to e.g. maintain a history of the design development.

2.3.6 Code Compliance and BIM Evaluation

To ensure that the change in the design and the design in general comply with building codes and is consistent with the additional design solutions, new tools have been developed for evaluation of information available in the BIM. The leading tool in relation to such BIM evaluation is developed by Solibri in Finland and is called Solibri Model Checker (SMC). SMC includes a large range of parameterised rule sets capable of checking a BIM model defined in the IFC format for a large range of issues such as consistency between domains, modelling errors, accessibility, escape routes, code compliance and area distribution requirements. Other tools like NavisWorks from Autodesk also allow for checking of BIMs, however, SMC is far more advanced with its ability for parameterised checking. The layout of SMC is illustrated in Figure 2-16.

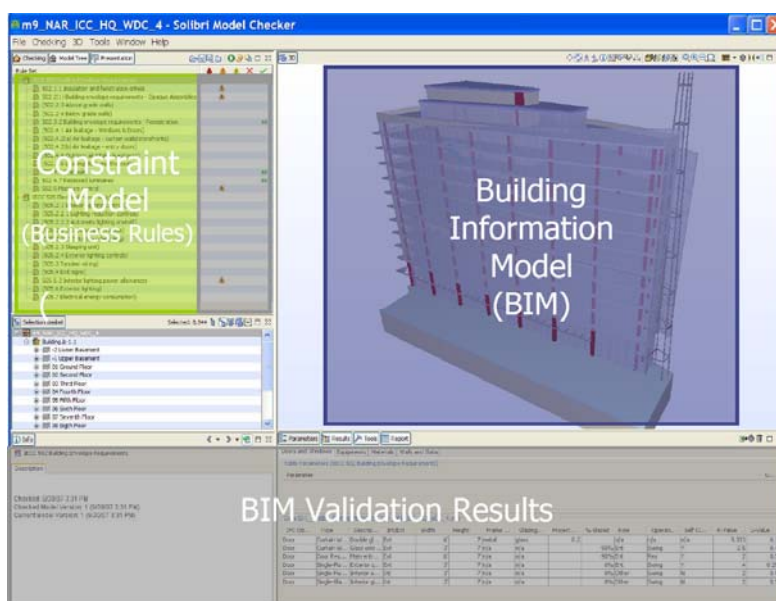


Figure 2-16. The layout of Solibri Model Checker with parameterised rule sets (constraint model) on the left and a BIM to be checked on the right [Kvarsvik et al. 2007].

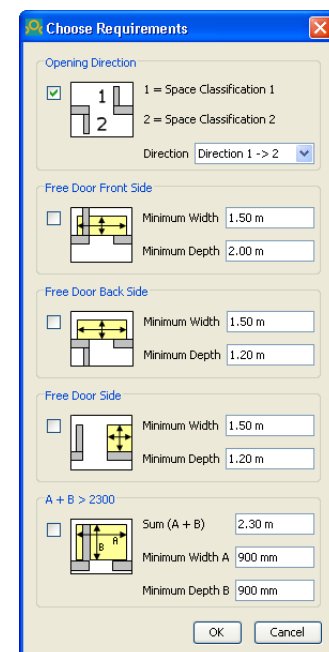


Figure 2-17. Example of parameterised rules, in this case for compliance with accessibility requirements.

The concept of SMC is interesting because it allows for continuous checking of the design in the BIM throughout the design process in relation to both consistency and requirements assuring that the design develops accordingly. The rule sets can be manually adjusted to the needs in each particular project within the framework of existing rule sets as illustrated in Figure 2-17. One such set of requirements that needs to be met is the compliance of the client requirements defined in the first phase of construction projects. In relation to client requirements, SMC includes possibilities to check if the spaces in the BIM comply with the requirements in relation to areas, heights, numbers and location.

The magnitude of client requirements is, however, much more comprehensive. [Kiviniemi 2005] has conducted significant research in the area of client requirements and the relation to BIM design. [Kiviniemi 2005] argues that only around 30 % of the requirements are numerical values, and that

descriptions and references constitute a much larger share. This limits the possibilities for automatic checking of the BIM in relation to client requirements although it is argued that the possibilities should be used to the extent possible.

2.3.7 Definition of a Client Requirements Model

[Kiviniemi 2005] has found that it is necessary to separate the client requirements and the design into two separate BIMs which include objects containing requirements and design information respectively. Linking the objects in the requirements model with the objects in the design model allows for client requirements to be an integrated part of the design process. [Kiviniemi 2005] argues that the separation of the two models is required because this allows for proper management of the requirements independently of the design. The concept is illustrated in Figure 2-18.

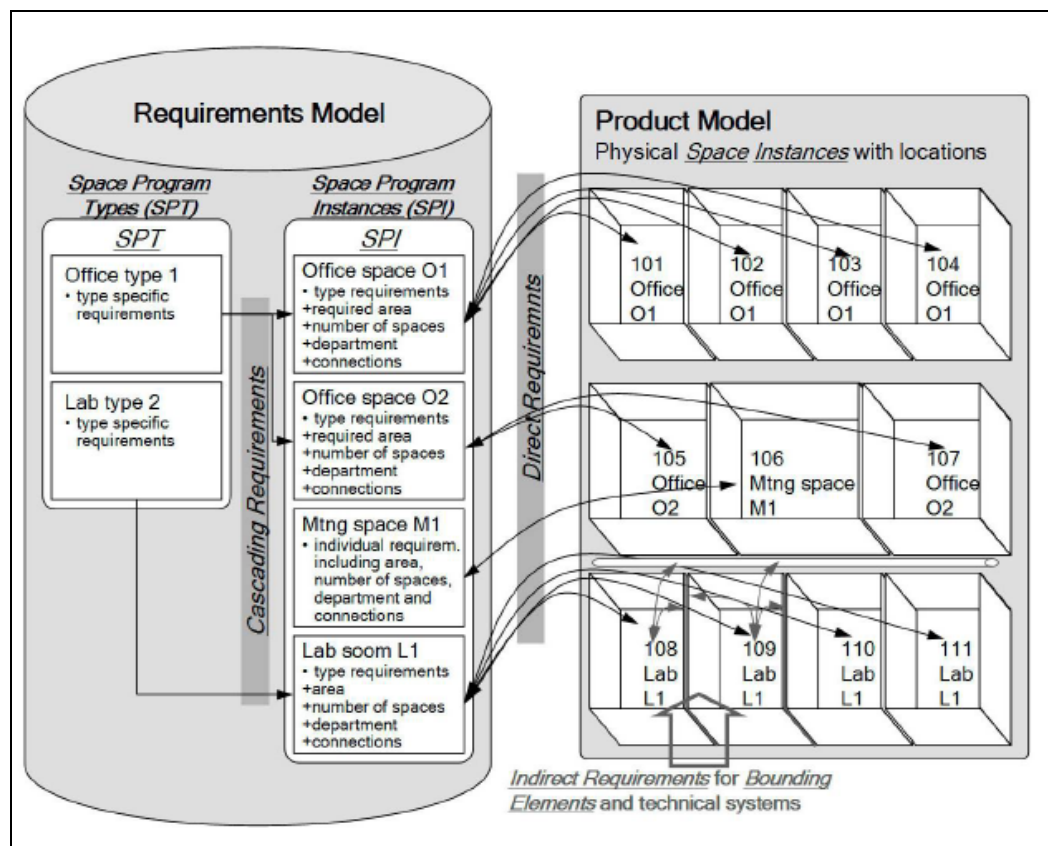


Figure 2-18. Concept for co-existence of a requirements model and a design model [Kiviniemi 2005]. The figure mentions concepts of SPT, SPI and cascading requirements which will not be included in this thesis.

The requirements model is intended to be defined in the IFC format similar to the design model. [Kiviniemi 2005] further argues that there is a need for both direct and indirect requirements because only limited requirements relate to for example a space itself whereas most of the requirements for a space relates to its boundaries and the systems supplying the space. In order to create the direct links, it is suggested to agree upon a simple naming convention such as space numbering for linking of spaces. It is argued that the use of a more complex identification such as unique ID numbers described in section 2.3.3 *IFC as the Shared Data Repository* for objects defined in

IFC is not useful, as it does not allow for the required flexibility in the design process when e.g. objects are deleted and replaced by others. In such a cases, it will be much easier to simply assign the name of the deleted item to the newly created, as opposed to a process where a new set of unique ID numbers has to be paired. Indirect requirements should be handled in the design model using the possibility to create relations between objects as mentioned in section 2.3.3 *IFC as the Shared Data Repository*.

[Kiviniemi 2005] argues that there is a current lack of a systematic plan for requirement management in IFC, and it is found that the IFC specifications must be extended with a new set of requirement objects if proper requirement definitions in IFC should be possible.

As mentioned previously, the requirement objects can contain requirements in terms of descriptions, numerical values or references to external documents. The requirement management is outside the scope of [Kiviniemi 2005], but it is described how EcoProp from VTT in Finland and the Design Intent Tool from Lawrence Berkeley National Laboratory in USA hold capabilities of recording client requirements in relation to sustainability and energy efficiency although none of these tools are capable of creating a link to the design model.

Also out of the scope of [Kiviniemi 2005] is the definition of how the requirements model can be used during the design process to ensure design compliance with the requirements. The possibility for automatic checking as described in section 2.3.6 *Code Compliance and BIM Evaluation* is suggested to the extent possible along with a more simple possibility of providing access to the requirements in the design tools. An example of such a possibility is illustrated in Figure 2-19 where a requirements management interface is created for ArchiCAD. Such a requirement management interface however only allows for the designer to access requirement information and is not used actively in the design process.

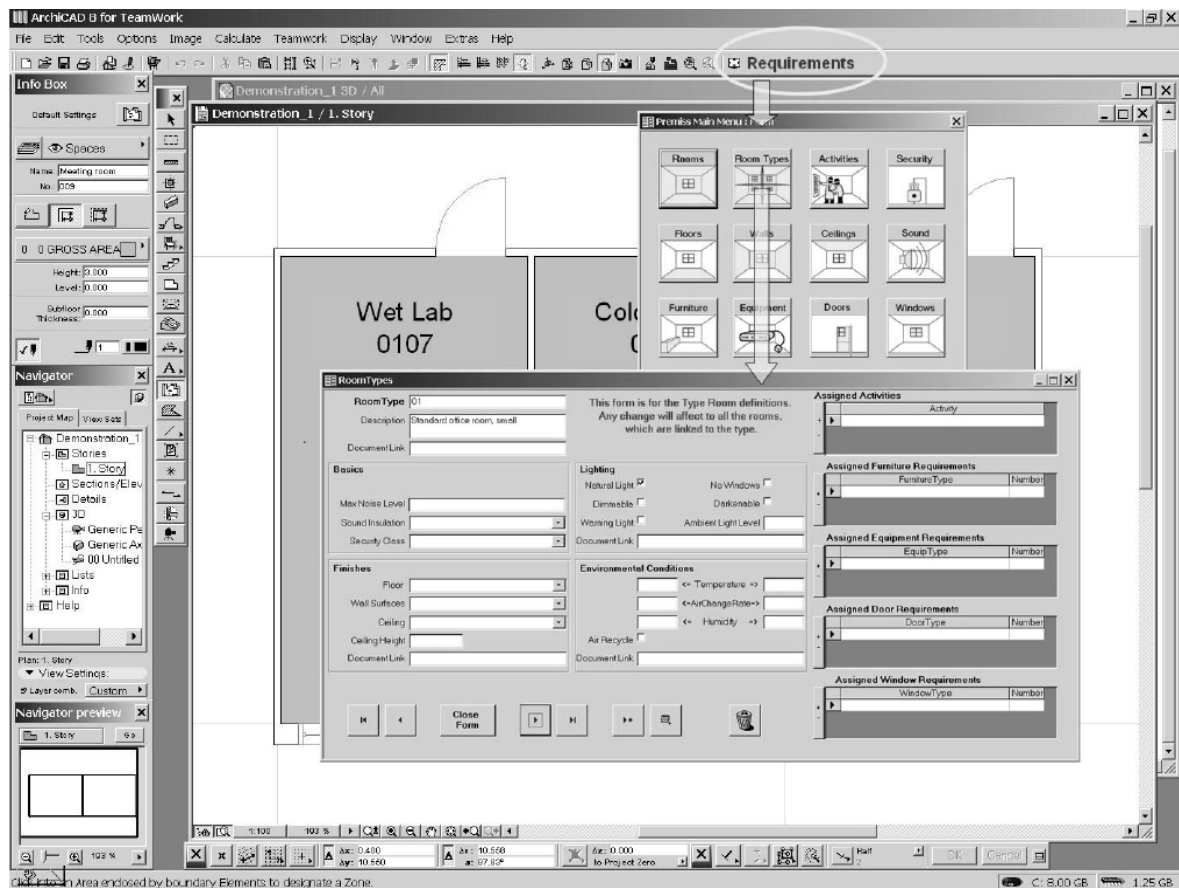


Figure 2-19. Suggested requirements management interface for ArchiCAD to the requirements model [Kiviniemi 2005].

From the literature study in this current thesis, only very few other suggestions of how to use requirements actively during the design phase have been identified. dRofus developed by Nosyko in Norway is an example of a database tool which intends to create a link between the requirements in the space program and the design models via IFC. So far the comparison is primarily relating to space areas whereas most other requirements are simply stored in the database for information purposes. Granlund in Finland has started the development of a tool called Roomex which is intended to manage several requirements of spaces and their boundaries and allow for these values to be compared to the current design models. So far only limited information is, however, available on this tool.

As the research in [Kiviniemi 2005] will be used further in the next chapter, it should be noted that the scope of this research was limited to architectural design; however, the linking to other domains was considered.

The possibilities of including requirements in BIM design as a linked requirements model is found to be an important opportunity for an improved ability to control the design process. Although a more comprehensive implementation of such possibilities is still to come, this is seen as an important opportunity to improve the decision-making process for the designers.

2.3.8 Summary on BIM Design

BIM design is found to be a concept of creating a Building Information Model containing as much information as possible of a construction project. The BIM should be stored in a shared data repository defined using open standards. This allows for all software to have equal abilities to interact with the BIM and creates full interoperability within software used e.g. by the design team. The IFC specifications are identified as the only valid open standard for this task. The use of a model server to manage the shared data repository is seen as an important way to improve the information flow and does further allow for new capabilities for management of information and changes during the design process. Linking a requirements model with the design model further allow for better possibilities to ensure that the design meets the requirements set by the client.

2.4 Possibilities to Improve IDP by Using BIM design

The previous sections described the concepts of integrated design and BIM design. It was found that there is a need for both concepts independently of each other due to requirements for better building performance and improved efficiency in the AEC industry respectively. Although issues still need to be addressed, both concepts have been used in actual construction projects and proved that they are capable of achieving the needs addressed. The approach of the two concepts to optimise the work processes in the design phase is different, and for this reason they could potentially support each other in order to improve the process even further. This is the hypothesis of this thesis, and in the following such potentials will be analysed. The primary aim of the thesis is to identify how IDP can be improved by using BIM design, and this will be the topic in the following.

2.4.1 Challenges for Successful Use of IDP

IDP fundamentally changes the design process by involving all members of a design team right from the beginning of the design process. By using a holistic approach, IDP further changes the design process by defining that the design decisions should gradually move towards an increasing level of detail and at the same time ensuring that previous decisions are respected throughout the process.

In order for these changes in the design process to result in improved building performance, it can be summarised from the literature study that IDP requires the following:

- Committed design team members with significant skills within their domain and abilities to co-operation.
- Support of the decision-making process by an extensive use of simulation tools.
- Efficient management and documentation of the design process to ensure that the integrated procedure of IDP proceeds as intended.

Skills of the design team members are a personal capability, however, the use of simulation software and the design management and documentation are also part of the concept of BIM design. These capabilities will be discussed further in the following.

2.4.2 Interoperability and Simulation Tools

Because simulation tools include various capabilities, it has been found that a range of simulation tools are likely to be used through IDP and that several tools could be used simultaneously. [Bazjanac 2007] describes in relation to interoperability that the current use of simulation tools is chaotic and that “no *commonly agreed to rules* exist for data collection, data transformation, verification of model quality or almost anything else”. It is further describes that the quality of models used in simulation tools depends on data and resources available at the given stage of the project and the knowledge and experience of the design team member responsible for the simulations. In [Bazjanac 2003] it is argued that “when such assumptions can be replaced by better data generated with other tools and promptly imported into simulation when needed, the quality of the resulting simulation can be significantly improved.” This concept supports the intentions of a holistic approach in IDP far better than the more sporadic use of simulation tools practised today. Such possibilities can, however, only be achieved by wide interoperability among simulation tools.

All information regarding building shape, composition, location and material selection used in simulation tools will most likely be derived from the current design documentation. It is therefore of great importance for the simulation tools to be capable of sharing information with the design tools containing the latest design documentation in order to ensure the best possible quality of the models used for simulations. This further requires that interoperability exists within both design and simulation tools. Interoperability within such a large range of design and simulation tools is one of the main objectives in the concept of BIM design and the possibilities of interoperability in BIM design is therefore identified as one of the main contributions to IDP.

2.4.3 Information Management and Control

The integrated procedure of IDP with its closed iterations requires effective management to ensure that the design progresses within the constraints of the performance-decisive parameters defined at milestones in e.g. a space of solutions. The management and checking of design compliance in relation to these constraints is therefore one of the main challenges for the project management in IDP. As the design progresses and more and more milestones have been reached, the amount of parameters to manage could raise significantly making it complicated for the project management to keep track of changes and to ensure proper design development in every detail of the building.

Being capable of managing constraints in the space of solutions effectively, being able to maintain a relation between the design and the space of solutions and being able to monitor and check the changes and development of the design would allow for optimal management of IDP. Such capabilities of being able to store, link and control building related requirements and design information are another main objective of BIM design. Although the literature study has identified

limited technical capabilities at present, the future possibilities of information management and control are found to be very useful. These possibilities should therefore also constitute a main contribution to IDP.

2.5 Summery of Possibilities

The previous sections have indicated that BIM design is capable of improving IDP in two predominant ways. Increased interoperability among design and simulation tools will first of all improve the decision-making process in IDP and secondly, the potential possibilities for information management and control should allow for improved abilities to ensure proper development of design.

It is important to notice that these improvements to IDP will not cause the process to change. The fundamental concept of IDP remains unchanged; however, the framework, the efficiency and the results should be significantly improved. The effect on the integrated procedure of IDP is illustrated in Figure 2-20. The figure illustrates that interoperability is intended to support the decision-making process within each iteration and that information management and control is intended to support continuous development of the design by ensuring that each iteration remains closed for the remaining of the design process.

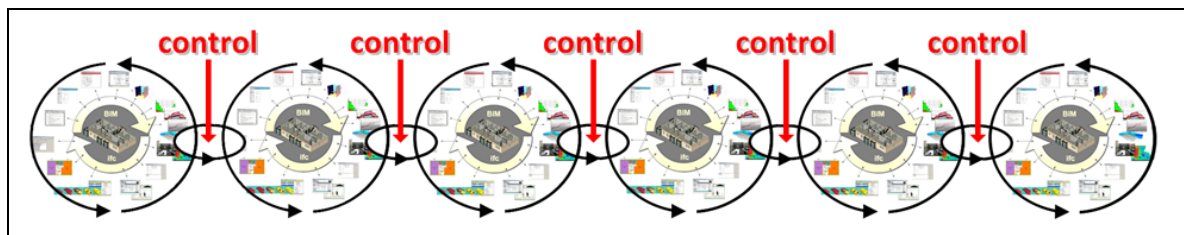


Figure 2-20. Illustration of how BIM design can improve IDP by interoperability for a better decision making process and information management for continuous development of the design.

To ensure proper possibilities to refer to the concept in the following chapters, it will be defined as *Integrated BIM Design* (IBD) which refers to the use of BIM design in IDP.

This chapter has indicated possible capabilities of BIM design to support IDP. The following chapter will continue the exploration of these capabilities; however, this will be done in particular relation to the possibilities of the IFC specifications to support the concept of IBD. Once possibilities of IFC have been identified, Chapter 5 will summarise the findings and try to define a common methodology for Integrated BIM Design.

Chapter 3

Capabilities of the IFC Specifications

3.1 Introduction

In order for the possibilities of IBD described in the previous chapter to have the intended effect, the following chapter will analyse the capabilities to support the process of IBD when using a shared data repository defined using the IFC specifications.

The previous chapter identified interoperability and information management and control as two main possibilities for the improvement of IDP by using BIM design. Because the two possibilities are relatively independent of each other, this chapter will analyse the two possibilities separately.

3.2 IFC Capabilities for Interoperability in IBD

As described in section 2.4.2 *Interoperability and Simulation Tools*, interoperability is required within both design and simulation tools. This creates a demand for information from all tools to be stored and exchanged via the shared data repository. The ability to store and exchange information within design tools using IFC has been evaluated in [Trelde & Johnsen 2005] which demonstrates several possibilities although a range of technical barriers are identified. The implementation of IFC version 2x3 in design tools during 2006 and the “Extended Coordination View” described in section 2.3.4 *Information Delivery Manual and Model View Definitions* seems to have further limited the technical barriers, although holes are still found [Erabuild 2008]. What remains unclear is, however, the possibilities to store and exchange information required for simulations tools. MagiCAD and recently also AutoCAD MEP hold capabilities to export some building services objects to an IFC file, with limited capabilities to include properties according to the IFC specifications. These properties primarily relate to the duct and pipe objects modelled, and information on control and performance capabilities of the systems are therefore limited. [Bazjanac 2004] describes that the development of an IFC interface allowing for such HVAC related information to be imported into an American simulation tool called EnergyPlus was the only attempt to use such information at that time. The IFC interface to EnergyPlus is not currently available and only limited information on this interface exists. The research in this thesis has not found any information indicating that other considerable attempts have been made to use such information. The ability for simulation tools to exchange information via IFC therefore remains unclarified.

Since the IFC specifications include a large range of possibilities to define both general design information and HVAC specific information as described in section 2.3.3 *IFC as the Shared Data Repository*, this thesis will conduct a detailed analysis of the current potentials of storing and exchanging information required for simulations tools in order to properly evaluate the current capabilities for interoperability in IBD.

To do so, five simulation tools used in Denmark and other Nordic countries have been selected for evaluation. By identifying the required input and output data of these simulation tools, it is the intention to document the required information which needs to be stored and exchanged. Followed by a comparison of the information which can be stored using the IFC specifications, the intention is hereby to demonstrate the current capabilities for interoperability among simulation tools.

3.2.1 Identification of Input and Output Data for Simulation Tools

The selection of simulation tools to be analysed has been based on their ability to support the scope of this thesis which is energy and comfort performance and by their ability to represent the variety of different simulations required in IBD. The five selected simulation tools are:

- *iDbuild* – described in section 2.2.5 *Space of Solutions*.
- *Riuska* – a Finnish thermal simulation tool similar to BSim although it is more simple in its functionality. Included because Riuska is capable of importing geometry from an IFC file and exporting limited simulation results back to the IFC file.
- *BSim* – described in section 2.2.4 *Use of Simulation Tools in IDP*.
- *Be06* – described in section 2.2.4 *Use of Simulation Tools in IDP*.
- *Flovent* – described in section 2.2.4 *Use of Simulation Tools in IDP*.

According to Figure 2-5 this selection of tools represents the use of simulation tools in various stages of the design process, and their specialised abilities to analyse energy, thermal and comfort performance should cover a large range of tasks in the decision-making process of IBD.

Because the simulation tools have a very large range of capabilities to simulate many different building types and designs, it has been decided to create a small demo project of an office building to limit the scope of this analysis to a model which should represent typical building design. The project was named MiniOffice and is illustrated in Figure 3-1 and Figure 3-2.

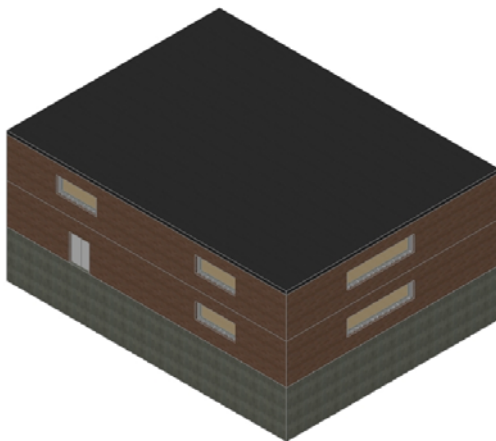


Figure 3-1. The MiniOffice project seen from the north west corner.

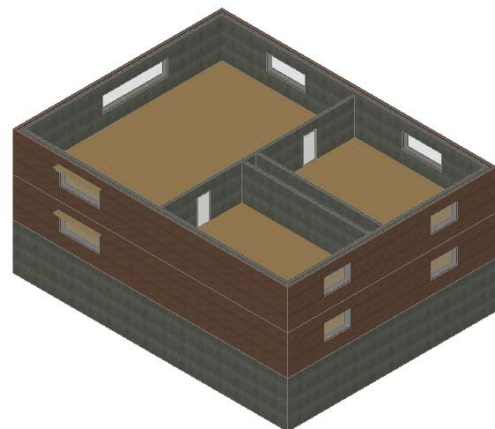


Figure 3-2. The MiniOffice project seen from the south east corner with no roof. The overhang of the windows in the southern façade is indicated in the left part of the figure.

The MiniOffice is a two storey building with a basement and includes five offices, a reception and a toilet. It has a total gross area of 332 m² and is supplied by VAV ventilation with both heating and cooling capabilities. To simplify the simulations, standard guideline values were used for occupancy, loads, water consumption etc. and construction types were selected as standard types from the material databases available in the simulation tools. To include shading possibilities for windows, small overhangs are installed above the windows in the southern façade. Based on this design, the

intention is to conduct a typical simulation in each of the five simulation tools of such a building and hereby identify all required input and output data.

All input data identified as required for the simulation of the MiniOffice has been documented in input/output data charts which can be found in Appendix 2-6 for the five simulation tools respectively. Additionally, other input data which could be required for similar building types is also included for information purposes. Most of the simulation tools are capable of generating a large amount of output data; however, only key information has been included in this analysis to maintain a reasonable scope. The output data has been selected based on an assessment of what information would typically be required to describe the results of the simulation and hereby what would be preferred to include in the BIM for further use.

In order to be able to compare the input and output data with the possibilities of the IFC specifications, the identified information has been organised in an object oriented tree structure similar to the one used in IFC which is illustrated in Figure 2-12. This means that all information has been attached to the objects to which they relate. In some cases the information has been broken down into further detailed information, if this is required by the simulation tool.

Although building services objects such as fans and boilers are contained in the same hierarchical structure as spaces and walls, it is found more appropriate to place these in another structure consisting of systems. Several of the simulation tools use systems to describe performance of e.g. the ventilation system for several spaces at the same time and therefore require that information can be attached to objects in such systems. As described in section 2.3.2 *Concept of BIM Design*, systems allow for grouping of objects and can therefore be used to identify all objects in systems such as ventilation, heating or lighting.

No units are included in relation to the identified information because the five simulation tools in many cases use different units for the same values. To keep the analysis simple it has been decided to leave these out because it typically only requires a simple transformation to achieve the correct unit. In all cases where a unit is required per area, the area size is also required, and values can therefore be derived in such cases as well.

Except for Flovent, all simulation tools use variations of schedules to describe activities in the room and operation of systems. Each analysis of the simulation tools does therefore include a description of how schedules are handled to be able to compare what information is required.

The input and output data identified typically consist of type specification or numerical values, however, in some cases the data can also be a 3D solid, a placement or a data set. It has been decided not to divide the data definitions into more details, because this concept is in accordance with the possibilities of attributes used in the IFC specifications. Attributes in IFC can, as described in section 2.3.3 *IFC as the Shared Data Repository*, consist of equivalent geometrical representations, placement information, data sets etc. and it should then be possible to match the attributes to the input and output data identified.

Any additional information required for the simulations which does not relate to the building directly such as simulation period and grid size has been listed under manual input, and is ignored in the later comparison with IFC.

The following sections will describe the input and output data analysis for each individual simulation tool. Each tool will briefly be described to create an understanding of how information is used in the tools.

3.2.1.1 Input/Output Data Analysis of iDbuild

As mentioned previously, iDbuild is capable of simulating energy and comfort performance of one room at a time. The input data requirements are therefore primarily related to information concerning geometry and composition of the space and its boundaries, loads in the space and fresh air, heating and cooling demands. Since no geometrical engine is implemented in iDbuild, all information is defined by numerical values. Some of the required input data is illustrated in Figure 3-3. iDbuild also calculates the overall energy consumption and this requires information on the average electricity consumptions for additional building services components in the building. Furthermore, iDbuild evaluates the indoor climate for which it requires information on the desired comfort level. Some of the results from the MiniOffice simulation in iDbuild are illustrated in Figure 3-4. iDbuild can also present information on the annual variation of most performance parameters calculated, such as air change rate, energy consumption, daylight and comfort level.

The 'Systems, Reference' dialog box contains several sections for input data:

- Heating & cooling:** Checkboxes for Heating and Cooling, setpoint temperatures (20 and 24 deg C), Extra insulation (0 W/m²), Internal loads (94 W), Min shading factor (1), and Shading control (Dynamic, No shading).
- Time in use:** Profiles (All year, Days, Hours, R17), Weeks (1:18 38:53), Days (1:5), Hours (8:17), and a note 'NBI 8:17 is defined as [R17]'. A list of systems (System 1, System 2, System 3) is shown with New, Update, and Remove buttons.
- Lighting:** General (Setpoint 200, Min power 2, Max power 6, Wlm2/100 lux 3), Task (500 lux, Wlm2, Wlm2, Wlm2/100 lux 1), and Control (Continuous, On-off).
- Ventilation:** Initiation (0.15 1/h), Ventilation (Ventilation, Passive ventilation), Control system with heat recovery (Settings), NBI Passive systems are yet to be implemented in iDbuild, Min air change (2.5 1/h), Max air change (5 1/h), Max venting rate (0 1/h), Heat exchanger efficiency (0.9), and Bypass.

Figure 3-3. Input data requirements for systems in iDbuild.

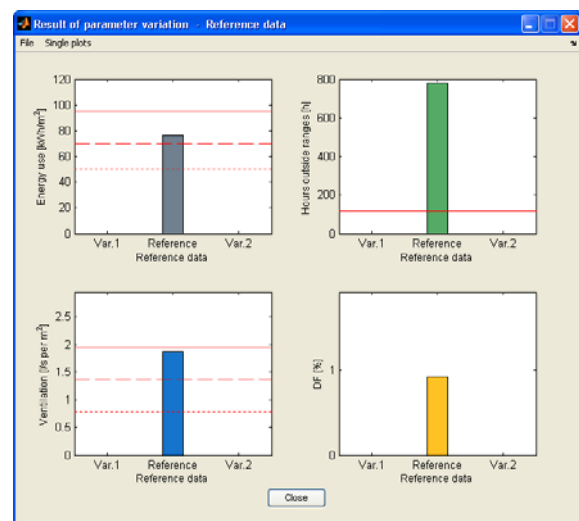


Figure 3-4. Results of selected average annual performance values from simulation in iDbuild.

As previously described, iDbuild is currently limited to simulations of only one façade with one window. Therefore the office on the second floor in the south east corner of the MiniOffice project was selected because this only has one window. For the simulation of this office in iDbuild a total of 42 required input data were identified for architectural objects and 18 required input data were identified for building services objects. The simulation generated several output data from which 8

key output data were selected. The complete chart of input and output data identified can be found in Appendix 2.

3.2.1.2 Input/Output Data Analysis of Riuska

Riuska is capable of simulating energy and comfort performance of a whole building. Although Riuska includes a graphical engine, all information relating the building design must be imported from an IFC file. This means that all geometry related information must be provided via an architectural BIM supporting application capable of creating an IFC file. Only geometry can currently be imported from the IFC file, and all other properties of the construction must therefore be defined manually. Riuska requires most of the thermal related properties such as loads and ventilation rates to be attached to each individual space. Only performance information for the ventilation system is applied on a system based level and linked to selected spaces. The layout of Riuska and load information required for a space in the MiniOffice project is illustrated in Figure 3-5. Results of a simulation in Riuska can be either annual variations of performance parameters related to a space or space group, or it can be monthly values of energy consumption, ventilation rates etc. The result of the annual energy consumption of the MiniOffice can be seen in Figure 3-6. Riuska is currently capable of exporting some simulation results back to the IFC file. These results include primarily minimum and maximum values of performance parameters for each space.

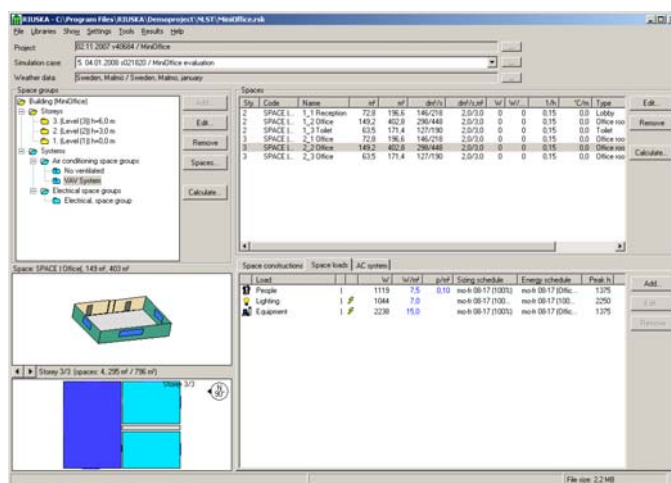


Figure 3-5. Layout of Riuska and list of loads attached to an office in the MiniOffice project.

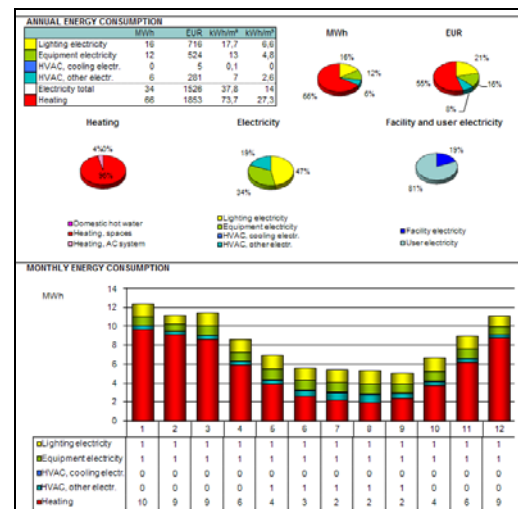


Figure 3-6. Energy consumption results from simulation in Riuska. The set point configuration in Riuska does not allow for the heating system to be turned off outside the heating season, hence the large energy consumption of heating.

For the MiniOffice project to be simulated in Riuska, it was required to model the project in AutoCAD Architecture and export it to an IFC file. Because Riuska requires space boundaries to create spaces, the IFC file further needed to be imported into ArchiCAD⁸ and exported to a new IFC file from here⁸.

⁸ ArchiCAD is currently the only architectural BIM supporting application which is able to export physical space boundaries in IFC files. All IFC files used in Riuska must therefore be created in ArchiCAD.

The simulation could then be conducted for the whole building by adding thermal and performance related information. In case of comfort level simulations it is further required to identify the position of a person in each space⁹. 49 input data were identified for architectural objects and 18 input data were identified for building services objects. From the simulation results 9 key output data were selected mostly as min/max values for space related information. The complete chart of input and output data identified can be found in Appendix 3.

3.2.1.3 Input/Output Data Analysis of BSim

BSim includes most of the capabilities of Riuska in terms of energy and thermal calculations and further includes several extra possibilities to define and control systems and loads in the building. Additionally BSim allow for advanced sun distribution and moisture load calculations, however these has been ignored in this case to maintain a reasonable scope of the analysis. BSim also includes a graphical engine where models have to be created manually as no import possibilities are yet present for 3D geometry. All construction related information is attached to the objects modelled. Spaces modelled in BSim needs to be organised in zones and it is in these zones that loads and system related information should be attached. One of the challenges in this concept is that several zones could be supplied by the same ventilation or heating system. In order to handle such issues, performance capabilities of each system must be split up and attached to several systems. In this analysis, it is assumed that each system only supplies one zone. This is because the described issue is seen as a simulation tool issue rather than an interoperability issue. The MiniOffice project defined in BSim including attached systems is illustrated in Figure 3-7. BSim holds some of the most extensive capabilities to provide output data from the simulation and all values can be described as annual variations. Some of these parameters describing indoor environment in the offices of the MiniOffice are illustrated in Figure 3-8.

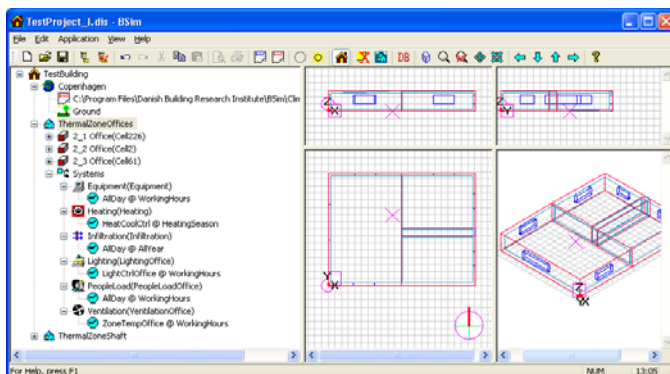


Figure 3-7. Layout of BSim with a list of loads and systems attached to a zone of offices on the second floor in the MiniOffice project.

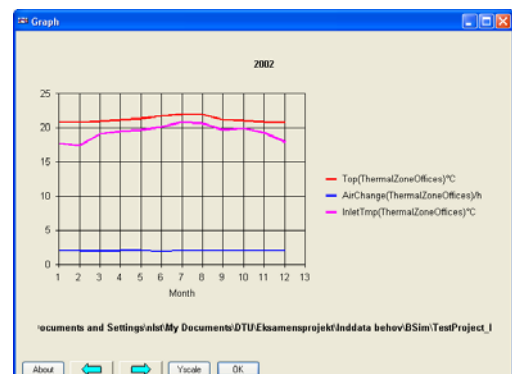


Figure 3-8. Simulated thermal conditions in the zone of offices. Here temperature, inlet temperature and air change rate is presented.

Because of modelling problems in BSim only the second floor of the MiniOffice project was simulated, however, concerns to information regarding constructions at ground level has been

⁹ Software errors in Riuska prevented simulations of comfort levels to be conducted, however, the functionality is available in the tool as present.

included in the identification of input data. For the simulation of the MiniOffice 45 input data were identified for architectural objects and 25 input data were identified for building services objects. From the large amount of output data, 7 key output data were selected consisting of annual variation values and annual summations. The complete chart of input and output data identified can be found in Appendix 4.

3.2.1.4 Input/Output Data Analysis of Be06

Be06 stands out from the rest of the simulation tools by being intended primarily for summation of energy consumptions. The tool, however, includes simple thermal simulations to estimate heating and cooling demands and various functionalities to estimate any additional energy consumptions. Be06 assumes the whole building to be one large room, and the required information on building design does therefore not need to be as detailed as in other simulation tools. Be06 does not include a graphical engine and all information therefore has to be defined by numerical values similar to iDbuild. Although Be06 requires a summarised value for the total gross area of the building, it is defined in this thesis that gross area will be required per space, so that unheated spaces (set point temp. below 15°C) can be handled separately. In order for Be06 to know the location of objects a b-factor is further used in the tool. The b-factor is used to describe the conditions on the outside of walls and windows and if objects such as boilers and pipes are placed inside or outside of the heated area. Because Be06 is intended to summarise all building related energy consumptions, the amount of required information relating to building services is significantly higher than for the other simulation tools. The layout of Be06 and some of the required input data for ventilation of the offices in the MiniOffice project is illustrated in Figure 3-9. The output of the simulation is primarily summarised energy consumptions on an annual or monthly basis. A presentation of the annual energy consumptions for the MiniOffice project is illustrated in Figure 3-10.

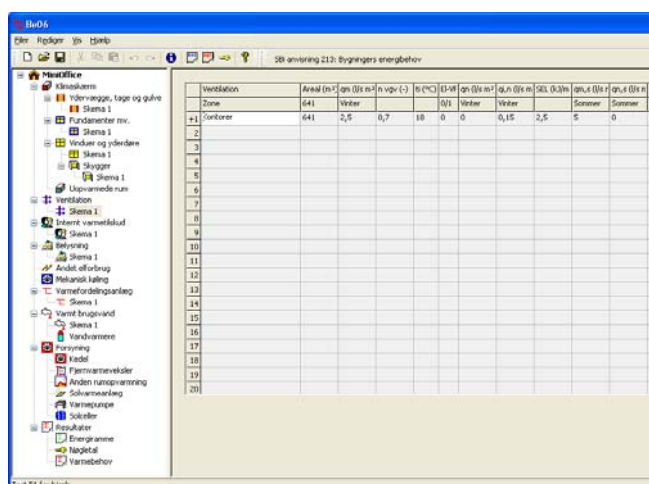


Figure 3-9. Layout of Be06 with a table for ventilation related information for the MiniOffice project.

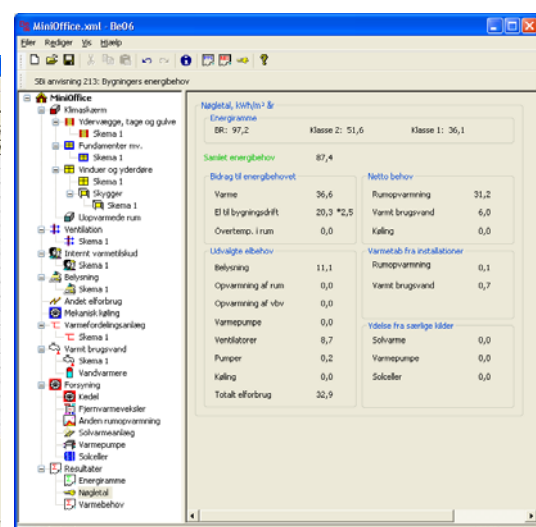


Figure 3-10. Calculated overall energy consumption (green) and key parameters of various energy contributions in the MiniOffice project.

Many of the information for building services objects relates to special cases of alternative energy sources or complex piping design which is not relevant for the simulation of the MiniOffice. The amount of input data is thereby limited in relation to the actual potential. For the simulation in Be06 a total of 36 required input data were identified for architectural objects and 37 required input data were identified for building services objects. The amount of output data is limited in Be06 but 4 key output data were selected. The output data consists of various annual energy consumptions. The complete chart of input and output data identified can be found in Appendix 5.

3.2.1.5 Input/Output Data Analysis of Flovent

Flovent holds advanced capabilities to simulate movements of e.g. air in a building by conducting a CFD simulation. To do so, Flovent needs information on the building design and any objects in the building that could affect the air movement by their geometry and/or thermal properties. Flovent includes a graphical engine for the creation of simulation models and is further capable of importing geometry from a range of 3D file formats. Import of objects including information is not currently possible. The layout for building models in Flovent is illustrated in Figure 3-11 and shows the construction of an office from the MiniOffice project. Because Flovent does not consider energy consumption, the primary information required from building services objects relates to the way they affect each space e.g. by a ventilation rate or heating load. A simulation in Flovent is a steady-state simulation of a specific day at a specific time, and all information required needs to represent the conditions in and around each space right in that moment. Usually this would be for the design days for heating and cooling design, as these are typically the most critical in relation to draught and proper air mixing. The results of a simulation in Flovent can be either thermal or comfort level properties in a specific point or distributions of temperature, air velocity, air quality etc. throughout each space. The temperature distribution for an office in the MiniOffice project can be seen in Figure 3-12.

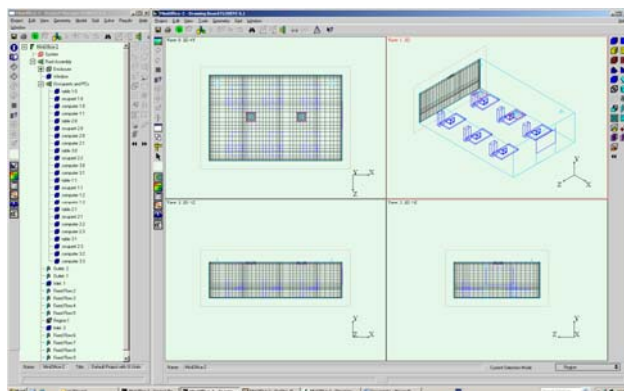


Figure 3-11. Modelling layout of Flovent. In the figure, the office on the second floor in the south east corner of the MiniOffice project has been modelled.

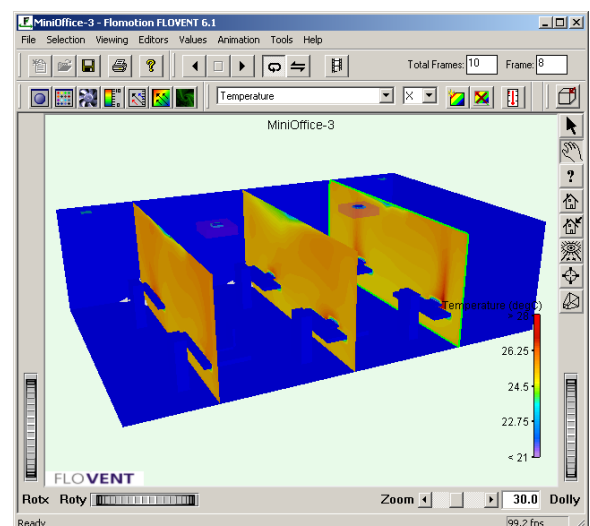


Figure 3-12. Results from the CFD simulation of temperature distribution in the office on the cooling design day.

In a project like the MiniOffice, Flovent would usually be used to simulate air movements in one office at the time. The same office used in section 3.2.1.1 *Input/Output Data Analysis of iDbuild* is therefore selected and by adding occupants, furniture and air terminal devices the simulation could be conducted. For the simulation of this office in Flovent a total of 30 required input data were identified for architectural objects and 15 required input data were identified for building services objects. From the output data 4 key output data were selected. The output data consisting of distributions of temperature etc. can be provided either in data sets or in visual presentations. As the data sets will include a very extensive range of values, it is not appropriate to store these in the shared data repository. Instead the data sets or visual presentations could be referred to by the BIM to maintain a link to the results. The output data specified in a point is just single values and could therefore be included. The complete chart of input and output data identified can be found in Appendix 6.

Although the input/output data charts will be used extensively in the following sections, it has been decided to keep these as appendixes as it will be too comprehensive to include them in the main thesis.

3.2.2 Capabilities of IFC to Contain Information Required for Simulation Tools

The previous sections have identified required input and output data for five simulation tools which could potential constitute the selection of simulation tools required to ensure proper energy and comfort performance in IBD. To ensure interoperability it must therefore be required that the IFC specifications can contain the input and output data identified. For this reason the following section will analyse the potentials of the IFC specifications in this relation.

Simulation tools in particular require a certain level of detail in the information provided in order to be able to run valid simulations. Especially in the early stages of the design process, this can be challenging because much information is still unknown. It is out of the scope of this thesis to identify how and when information can be generated during the design process. It is, however, necessary to have a basic understanding of how information is generated in order to understand how information should be stored in an object oriented data repository. The following section will therefore briefly describe how design information can be generated during IBD.

3.2.2.1 Generation of Information

[bips 2006] describes how the detail level of 3D object oriented models should gradually increase as the design progresses in order to follow the requirements developed under the Digital Construction project in Denmark described in section 2.3.1 *The Need for BIM Design*. Figure 3-13 illustrates this concept.

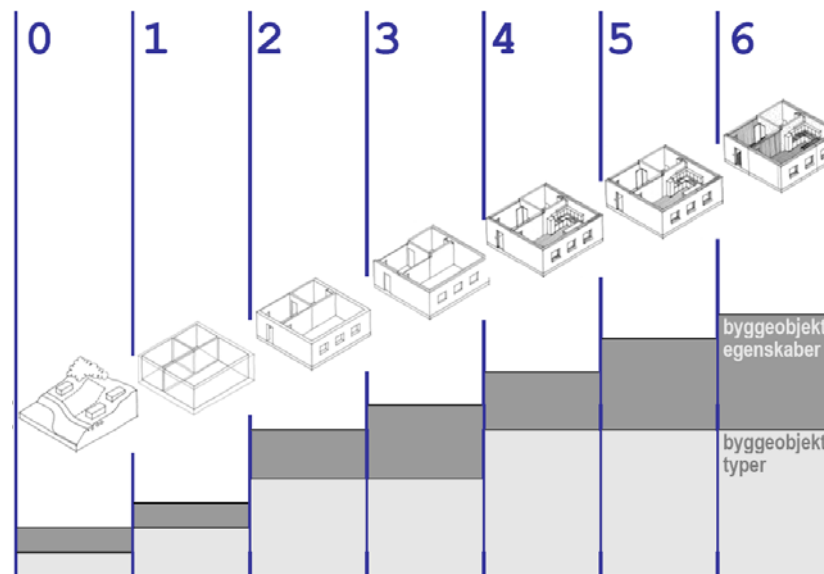


Figure 3-13. Required development of 3D object oriented model in the Digital Construction project [bips 2006]. The steps 0-6 describe the model development from the client requirements definition phase to the end of the construction phase.

Although this concept relates to a more traditional design process and only specifies limited requirements for the information in the models, it is important to notice that only a very limited amount of object types is present in the early stages of the design phase, and that all objects constituting a proper description of the building is not to be included before step 4 which is in the detailed design phase. The use of IDP and actual BIM design will most likely force a greater level of detail earlier in the design phase, however, it cannot be expected that all objects are present in the integrated BIM at the point when simulations are required in the early stages of the design phase.

As described in section 2.2.4 *Use of Simulation Tools in IDP*, simulation tools should be used right from the beginning of the design phase and potentially even before the architect starts modelling the building design. This further implies that simulation tools could be one of the initial tools for the creation and refining of much of the information related to both architectural and building services components based on the client requirements. If input data is not yet available in the BIM, the defined input data in a simulation tool should therefore also be part of the output data so that this information is available in the BIM for the remaining of the design process. The BIM supporting design tools mentioned in section 2.3.2 *Concept of BIM Design* along with tools such as various EEDSS described in section 2.2.4 *Use of Simulation Tools in IDP* will additionally be capable of supplementing and refining information in the BIM throughout the design process. Generating information in a BIM is therefore a concept of continuous enrichment of the BIM from various sources at various stages of the design phase.

The IDM and MVDs described in section 2.3.4 *Information Delivery Manual and Model View Definitions* aims at defining more precisely how and when information should be available in a BIM defined in IFC. As previously mentioned these projects have not yet finished, and the analysis of IFC capabilities in this thesis will therefore only be based on the descriptions currently available in the IFC specifications documented in [IAI 2007a]. Because the IFC specifications include possibilities to store identical information in several places and allow for a large range of complex ways to combine and link information, the analysis in this thesis can only describe some out of several possibilities to

store and exchange information. To maintain some sort of consistency in the defined data, it will be attempted to define overall directions of how to store information. In some cases the structure and content of the IFC specifications, however, requires rather complex relations in order for specific information to be defined.

3.2.2.2 Geometrical Representations and Placement in IFC

Geometry related information has been within the scope of the only operational MVD (the Extended Coordination View) and some consensus has been reached of how to represent geometry of various objects from a large range of opportunities. Furthermore [IAI 2004] includes detailed guidelines of how geometry should be defined. The most commonly used representations for 3D geometry in IFC is either a Swept Solid representation or a BREP representation. A swept solid defines an object by a defined area (*IfcArbitraryProfileDef*) in a plane and an extrusion direction and length. This concept is illustrated in Figure 3-14. A BREP representation is used to describe more complex geometry of e.g. a window by defining a set of surfaces which combined represent the window geometry. Furthermore, Boolean operations can be used to either subtract or add solids to the geometry. Some objects also include a 2D representation such as an axis or a contour. As the scope of this thesis is limited to use of simple geometry, it will for now be concluded that the geometrical representations available in IFC should be sufficient to define geometrical information required for the simulation tools.

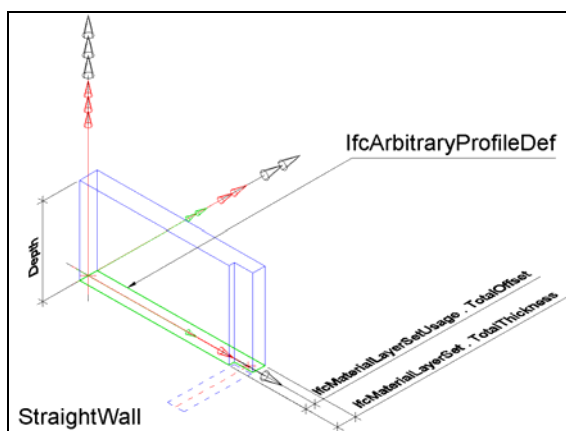


Figure 3-14. The definition of 3D geometry of a wall using a Swept Solid representation [IAI 2004].

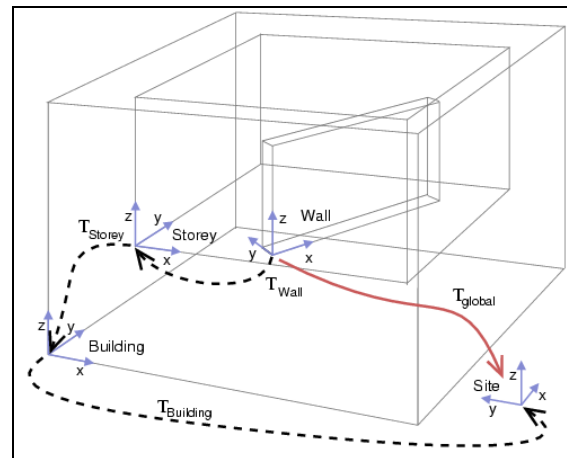


Figure 3-15. The concept of *IfcLocalPlacement* and its relations to other local coordinate systems [Mentzel 2003].

In order to define the exact placement of an object in IFC, a concept of *IfcLocalPlacement* is used. Each object will have its own local coordinate system to which all geometry definitions relate. The local coordinate system then relates to another local coordinate system of e.g. the storey which the object is located on. This local coordinate system can then relate to another local coordinate system of the building and so on. When the hierarchy of local coordinate system has reached the top, a world (global) coordinate system can be defined to relate the coordinates to such a system if required. The concept is illustrated in Figure 3-15. Combining the geometry representation and the placement of an object should allow for the simulation tools to retrieve all required information in relation to shape, size, placement and orientation

3.2.2.3 Definition of Possible Locations for Non-Geometry Information in IFC

The lack of proper implementation guidelines of how to define energy and thermal properties makes it uncertain how such information should be defined consistently because only the IFC specifications can be used as a reference. To ensure that information relating all objects can exist from the beginning of the design phase, this thesis propose two possibilities for storing information which are within the scope of the IFC specifications:

1. Information added to an object not necessarily defined with placement or geometry.
2. Summarised/aggregated information attached to objects to which they concern.

The definition of an object described in section 2.3.2 *Concept of BIM Design* implies that an object can exist without the need for defined geometry and an object consisting only of performance based information could therefore exist before the object was properly modelled in a design tool. In the early stages of the design phase, this could be relevant if a simulation tool requires information on e.g. fan performance or the output from a space heater and these objects have yet to be modelled in the design tools. Because such objects in large numbers can be complicated to manage, an alternative is to apply summarised information on e.g. performance or loads to the systems or spaces affected. Instead of modelling all objects contributing to the heat load in a space, their summarised load could e.g. simply be assigned to the space.

Such a concept potentially allow for all relevant information to be described when required in the BIM, however, as the design progresses this could result in the information being represented both via the objects and via the related spaces or systems. Duplicate data hereby exist with no assurance of what information is valid. It is out of the scope of this thesis to describe a proper solution to this issue. However, one option could be that information assigned to one or more building services objects would replace similar information assigned to a system or space.

Performance information in this concept can additionally be applied to both systems and spaces. In order to ensure that the same performance information is only applied to either the system or the space, this thesis further defines that performance information on a system, which affects a space, should be attached to the space and not to the system to the extent possible. This requires that e.g. the total air flow rate in a ventilation system must be calculated by summarising the air flow rates which the system provides to each space.

3.2.2.4 Use of Properties in IFC to Store Information

Some information required by the simulation tools is assigned directly to each individual object in IFC, however, most information required are defined in property sets. A property set consists of a list of specified properties to which values can be assigned if required. Property sets can be identified by the naming convention Pset_XXX, e.g. Pset_SpaceCommon or Pset_WallCommon. An example of a property set is illustrated in Figure 3-16. All building elements can have such property sets assigned by IfcRelDefinesByProperties or via a type definition of that object using IfcRelDefinedByType.

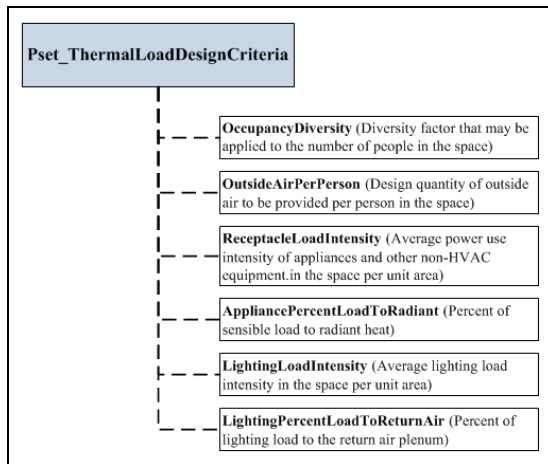


Figure 3-16. The ThermalLoadDesignCriteria property set and its defined content. Additionally the IFC specifications define that such a property set can be assigned to any *IfcSpatialStructureElement* such as a building or a space and further describe the type of values to assign to each property.

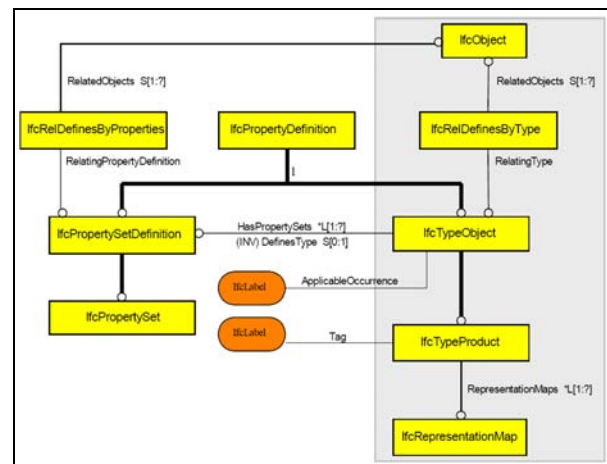


Figure 3-17. Assignment of property sets to an object either directly or via the type definition of the object in IFC [IAI 2004].

Although the IFC specifications allow for custom made property sets to be created, it has been decided that only predefined property sets are evaluated in this thesis due to the aim of assessing the current capabilities of IFC. Currently 317 property sets are defined which can be assigned to a selected range of the 653 entities in the current IFC specifications. Figure 3-17 illustrates how property sets can be assigned either directly to an object or via a type definition of that object. It should be noted that most properties mentioned in this section are assigned to entities via the supertype *IfcRelAssigns*. Such assignments could be *IfcRelDefinesByProperties* or *IfcRelDefinedByType* and they are characterised by assigning properties to the entities by inverse as described in section 2.3.3 *IFC as the Shared Data Repository*.

In relation to HVAC objects, the BS-8 project described in section 2.3.3 *IFC as the Shared Data Repository* has introduced a useful concept of describing such objects by their individual occurrence, by general type related information and by a performance history of how they perform. The concept is illustrated in Figure 3-18. Using this concept, it is possible to describe the placement of each component, the capabilities of this type of component and the performance of the component. Although termed “performance history”, the performance values can represent desired, simulated or measured values. One of the advantages of using the performance history is that simulation tools which only require a min or max value can derive this from the performance values while other tools that need more detailed descriptions of the performance variations can also derive such information. Property sets describing performance history are easily identified as their name end by “...PHistory” as for example “Pset_FanPHistory” for a fan. Property sets describing type related information include “Type” in the name such as “Pset_FanTypeCommon”.

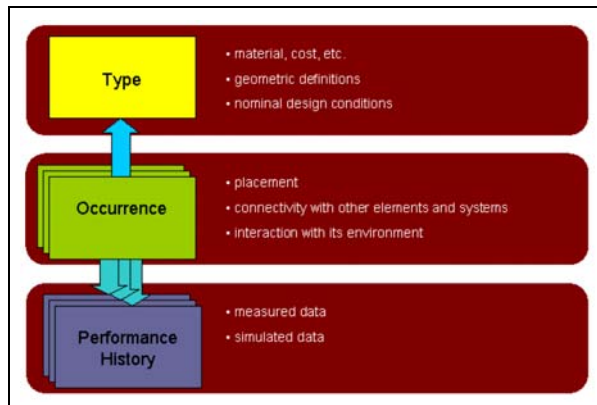


Figure 3-18. Concept of modelling HVAC objects in IFC [Bazjanac 2004, ©Jim Forester].

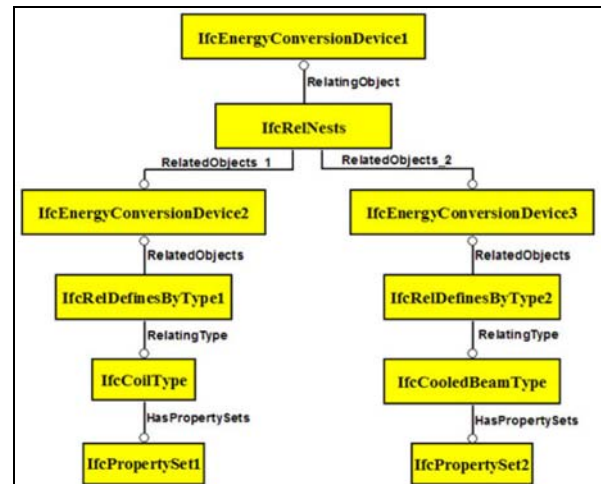


Figure 3-19. Concept of creating a container of objects to represent a shared object [Bazjanac 2004].

To define the performance history, the BS-8 project further introduced a concept of time series which enables the modelling of data that are defined dynamically and change in time [Bazjanac 2003]. Time series are able to define variations with a regular or irregular time step and can thereby be used to define variations in any performance parameter on e.g. an annual basis. Although there are some slight variations in the identified use of schedules in the simulation tools, they should all be able to relate to the concept of time series as values are precisely defined over time. A further description of time series can be found in Appendix 8.

Unfortunately the BS-8 project came to a conclusion that it was too complicated to define a common control definition for HVAC systems, and the possibilities in this regard therefore remains limited [Bazjanac 2003].

One of the advantages of IFC is that objects can be defined as a composition of other objects and their properties hereby allowing for a detailed description of the composed object. Such relationships between objects can be created either by nesting using `IfcRelNests`, if all objects are of the same class, or by aggregation using `IfcRelAggregates`, if the objects are of different classes within `IfcObject` [IAI 2004]. An example of such a composition is illustrated in Figure 3-19.

Another useful feature of relationships in IFC is the ability to link objects by geometrical relations. Such relations could be the relation between a space and its boundaries using `IfcRelSpaceBoundary` or between a building element and its location in the building using `IfcRelContainedInSpatialStructure`. This way, properties from geometrically related objects can be derived. In relation to building services components these can be grouped into systems as previously described. This is done using `IfcRelAssignsToGroup` which is capable of grouping all elements in a system and hereby allow for identification of all objects in the system. In addition to systems, it can be useful to define connections between the objects in e.g. a ventilation system so that the composition can be derived when for example the flow route must be analysed. Such connections are established using `IfcRelConnectsPorts` and `IfcRelConnectsPortToElement` by defining an `IfcDistributionPort` between all flow devices in each system.

Additionally, it is attached to these *IfcDistributionPorts* that properties of the fluid inside the duct or pipe must be defined using *IfcFluidFlowProperties* as illustrated in Figure 3-20. This property entity is assigned similar to other property entities using *IfcRelDefinesByProperties*.

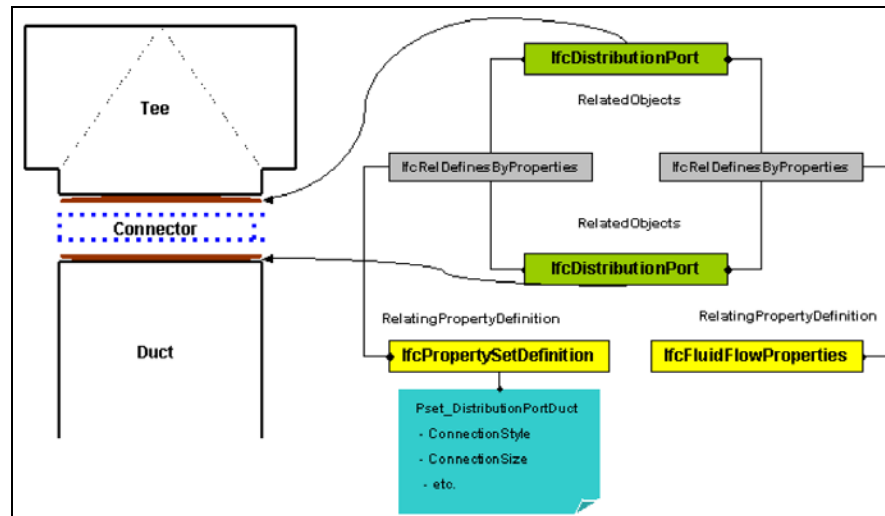


Figure 3-20. Concept of using *IfcDistributionPorts* and assigning fluid flow properties [IAI 2004].

All of the above properties and relationships must be defined by tools either holding this information already or having the capabilities to derive the information. It is a prerequisite for the following that such possibilities exist. Since only limited support for IFC is currently present in simulation and design tools, the following can only be a description of what could be possible if all of the capabilities of IFC could be handled by the tools.

3.2.3 Comparison of input and output data and IFC capabilities

Based on the possibilities for defining information in IFC, as described in the previous sections, it is now possible to compare the identified input and output data from the simulation tools with the possibilities of the IFC specifications to store this information.

The comparison has been conducted by examining the IFC specifications described in [IAI 2007a] and from here collecting all relevant attributes and relations identified to be capable of defining input and output data from the simulation tools. Due to the lack of proper implementation guidelines, as previously mentioned, the task of identifying relevant attributes and relations was found to be rather complex. The separation of the IFC specifications in different layers as illustrated in Figure 2-11 and the possibility of inverse assignment of attributes across layers make it impossible to derive a consistent hierarchy structure of the specifications. The inverse assignment of attributes allow for an infinite amount of possibilities to create relationships between entities and the specification only mentions some of the suggested possibilities for each entity. In order to identify relevant attributes and relations it was therefore necessary to go through each of the 317 property sets and 653 entities and to firstly identify relevant attributes. Secondly it was then required to identify whether possibilities existed to assign the relevant attributes to the desired object as either inherit from its supertypes or via an inverse assignment using relationships. This procedure was thorough, however,

it still does allow for potential possibilities to be overlooked although this is not believed to be the case in the following.

A chart with the complete collection of identified attributes can be found in Appendix 7. The collection of identified IFC attributes has been listed in accordance with the input and output data charts to allow for easy comparison. Based on the chart, the following sections will discuss how input and output data can be stored using the IFC specifications. Appendix 7 should be addressed by the reader in the following along with Appendix 2-6 as the following sections only intend to supplement information already available here. The sections will discuss similar objects combined to avoid too many repetitions and the capabilities to store output data in IFC will be discussed in the end to avoid confusion.

3.2.3.1 Input Data for Projects, Sites, Buildings and Stories

Name and location

All of the simulation tools require names for various objects and these are all available in IFC attached directly to the elements. A location is required primarily to identify an appropriate weather data file and for this purpose latitude and longitude can be defined for `IfcSite`. The property set `Pset_OutsideDesignCriteria` attached to `IfcBuilding`, however, defines an exact reference to a weather data file and is therefore preferred. This property set is also capable of defining the outdoor design temperatures for heating and cooling design required by Riuska and Flovent and it can be used to define the type of surroundings required by BSim, although another terminology is used for the type definitions. The attachment of a defined time zone to the building or site is not possible but iDbuild should be able to derive this information from the exact location defined for `IfcSite`.

Building type

Be06 requires information about the overall heating capacity of the building which cannot be assigned to the building, however, in section 3.2.3.3 *Input Data for Walls and Slabs* it will be described how such information can be derived from individual construction objects and this could be used for an evaluation of the overall capacity. Be06 further needs a type description of the building which couples type of occupancy and building shape. Since no precise definition of this exists in IFC, it is expected that such a type definition can only be derived in limited cases. The definition of type and behaviour of occupants required by Be06 will be discussed in section 3.2.3.2 *Input Data for Spaces*.

Orientation

Most of the simulation tools require information on the orientation of either the whole building or individual windows and doors. Such information can be derived from `IfcGeometricRepresentationContext` attached to `IfcProject` which describes the direction of north in relation to the Y-axis of the world coordinate system. Based on this information, the orientation of all objects can be derived as described in section 3.2.2.2 *Geometrical Representations and Placement in IFC*.

Terrain and ground

From IfcSite it is possible to retrieve the elevation of the site and BSim should therefore be able to calculate the angle to the horizon. Although IfcSite could potentially have material properties assigned, none of the material properties allow for a description of temperature variations in the ground as required by BSim. Such a value would additionally be difficult to apply, as the temperature varies in different levels.

3.2.3.2 Input Data for Spaces

Geometry and placement

All of the simulation tools require information about space geometry. Some only need either the gross or net area, while other requires information about the actual 3D shape. The Pset_SpaceCommon can hold both gross and net area information, while any other shape related information must be derived from either 2D or 3D representations assigned to IfcSpace. The placement of each space can be derived from the ObjectPlacement also assigned to IfcSpace as described in 3.2.2.2 *Geometrical Representations and Placement in IFC*.

Occupants

The required information for occupants in a space relates to their number, their activity level or load contribution and a schedule for their presence. The maximum number of occupants and occupancy type can be defined in Pset_SpaceCommon. The type description depends on a national standard but could allow for the activity level of the occupant to be derived although an exact value would be required for proper identification. Their clothing level required by iDbuild cannot be defined either but it could in some cases also be derived from the type description. Although the occupants can be described in more than five different property sets, none of these allow for a description of the schedule for their presence over time. For Riuska and iDbuild which require an occupant schedule e.g. to calculate their load contribution based on activity level, the possibilities are limited. Be06 and BSim require the max load and a schedule for the occupants and such values can be derived from IfcSpaceThermalLoadProperties which can represent the total load from occupants in a space as a time series. The time series could potentially allow for Riuska and iDbuild to derive the schedule for the occupants, however, this is more uncertain as the time series relate to load contribution and not presence of the occupants. Flovent additionally requires a 3D representation of the occupants. Usually occupants are not modeled in design tools and IFC does not include a possibility to define such a representation for occupants. Additionally Flovent and Riuska both need placement of occupants which is also not possible to define. Common practice today is, however, to identify the location of occupants by the placement of chairs in a space. As described in section 3.2.3.5 *Input Data for Furniture*, placement of chairs can be identified and the locations of occupants could thereby be derived.

Relation to systems, boundaries and furniture

For the simulation tools to know which spaces are supplied by each system, IfcRelServicesBuildings should be used to create a relationship between each space and each system. Most tools also need to know the relationship between a space and the construction objects bounding it. As previously

described this relationship is establish using `IfcRelSpaceBoundary`. Furthermore it can be required to identify the furniture in the space. This relation is established by a special relation either by `IfcRelContainedInSpatialStructure` or alternatively by `IfcRelReferencedInSpatialStructure`. The type of relationship depends on whether the object is already defined in one or more spatial structures such as a space or storey. This relationship can also be applied to building services components in a space and hereby allow for the identification of all objects which services each space.

Equipment

The total load from equipment and a schedule for its use is required by most simulation tools. As long as the information is attached to the space, it can be defined using `IfcSpaceThermalLoadProperties`. `IfcSpaceThermalLoadProperties` allow both for a definition of min/max values or a time series if the load is to be combined with a schedule. In case the actual equipment objects are to be defined, it should be done using an `IfcDistributionElement` to which a 3D representation and placement can be attached. This information is required by Flovent. Flovent further requires material properties, however, only emissivity of an assigned material can be defined via `IfcOpticalMaterialProperties`. The roughness is not currently part of the material properties in IFC. A further description on how to assign materials to objects can be found in section 3.2.3.3 *Input Data for Walls and Slabs*. The load of each object can be defined in `Pset_ElectricalDeviceCommon`, however, this is only a max value, and simulation tools which require a schedule will loose such information if the space information is replaced by object information.

Lighting

Similar to equipment, lighting loads can be defined by `IfcSpaceThermalLoadProperties` for each space and hereby include both load values and a schedule. The use of control and lighting fixture objects will be discussed in section 3.2.3.9 *Input Data for Lighting Systems*.

Set points and design temperatures

The use of set points for heating and cooling loads in spaces vary in the simulation tools. Some tools only require one set point for heating and cooling respectively and this could be derived from `Pset_SpaceThermalRequirements` although proper design values would be preferred. The use of sensors objects will, however, be required in case of different set points for ventilation and heating systems. This will be discussed in section 3.2.3.6 *Input Data for Ventilation systems* and 3.2.3.8 *Input Data for Heating and Domestic Hot Water Systems*. Be06 need a possibility to separate heated and unheated spaces and since this relates to a set point temperature below or above 15°C, the separation can be derived for the set point information. Design temperatures in the space required by Riuska can be defined in `Pset_SpaceThermalDesign`. iDbuild further requires definitions of comfort classes, required daylight factor and emission of air pollutants, however, these cannot currently be defined.

Infiltration

Although infiltration rate is required by most simulation tools, it is not possible to assign such information to the space. An infiltration rate can be assigned to windows and doors. However, since infiltration can also occur from e.g. joints in the construction, this is not found sufficient. The infiltration could potentially be found by subtracting the air change rate of the ventilation system by

the overall air change rate in each space, however this is found to be too uncertain as mixing between spaces could also occur.

Air flow rate

All simulation tools require information about the air flow rate in each space, typically as min and/or max values. To ensure that all tools can retrieve the required information and that a schedule can be derived from the definition, the air flow rate should be defined in `Pset_SpaceThermalPHistory`. In this property set, the air flow rate for both supply and exhaust air can be defined in time series by designed or simulated values. This way information can be derived as required by each simulation tool. The use of air terminal devices and definition of inlet temperature will be discussed in section *3.2.3.6 Input Data for Ventilation systems*.

3.2.3.3 Input Data for Walls and Slabs

It is possible to define walls by either `IfcWall` or `IfcWallStandardCase`. Only `IfcWallStandardCase` can contain several material layers and this class is therefore most widely used in design tools. This will also be the class selected in this analysis.

Roofs in IFC can either be defined as `IfcSlab` or `IfcRoof`. An `IfcRoof` must be composed by `IfcSlabs` using nesting in case any material properties are to be attached to the roof. For this reason only `IfcSlab` will be analysed in the following as any relevant roof structures are assumed to be composed by slabs.

Geometry and placement

Different levels of detail are required for geometry of the walls and slabs in the simulation tools. `IfcElementQuantity` can be used to define values such as areas and perimeter of the objects, however, since `IfcElementQuantity` is intended to follow a standard code for measurement which, at least in Denmark, has yet to be defined and implemented in design tools, it is found most reliable for the geometry related information to be derived individually from the geometry representations of the objects. Some simulation tools require information about the placement of the objects and such information is defined for both `IfcWallStandardCase` and `IfcSlab` along with the geometry representations.

Material properties

The composition of walls and slabs is defined by material layer sets. Materials and their properties are assigned to any object via `IfcRelAssociatesMaterial` and could consist of just one material, a list of materials or a material layer set. Via the Material Property Resource schema in the IFC specifications, a large range of properties can further be used to describe the materials. The assignment of materials to an object is illustrated in Figure 3-21 for a material layer set. The thickness of each layer is assigned to `IfcMaterialLayer`. The additional properties are assigned using `IfcThermalMaterialProperties` which allow for the definition of heating capacity, conductivity and density as required by most of the simulation tools. `iDbuild` and `Be06` only requires the U-value for each object and this can either be derived from the material properties or retrieved directly from `Pset_WallCommon` or `Pset_SlabCommon`. Similar to equipment, Flovent can retrieve information on

emissivity and not of roughness for the materials used. The requirement of Flovent for additional heat sources to be applied to the surfaces is not supported by IFC. Be06 further requires a value for the linear loss along typically the foundation, however, a placement for such a value is currently not available.

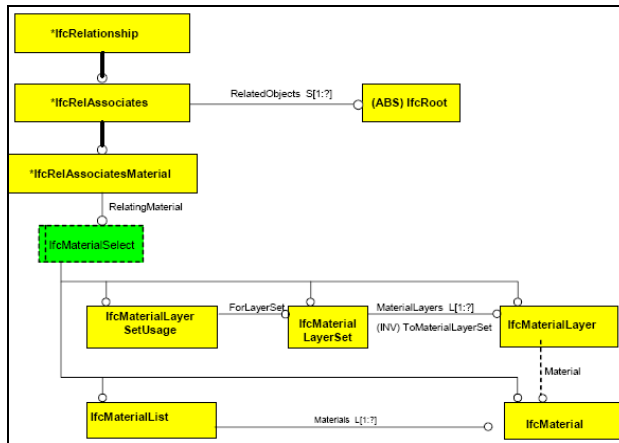


Figure 3-21. Assignment of a material layer set and materials to e.g. a wall or slab in IFC [IAI 2004].

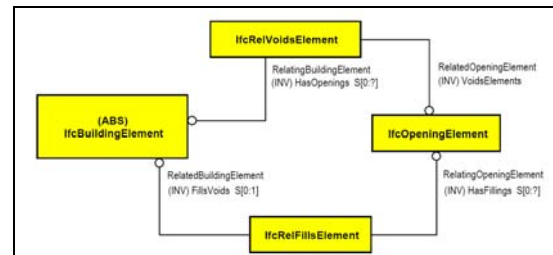


Figure 3-22. Concept of relating an opening element to both the object voided (e.g. a wall) and the object filling the opening (e.g. a window) [IAI 2004].

Relation to windows and doors

For the simulation tools to structure a proper model many of them need to know the relationship between typically walls and the windows and doors placed in the walls. Such a relationship must be established via a relationship between a wall and the opening element which voids the wall using **IfcRelVoidsElement** and afterwards a relationship between the opening and the window or door filling out this opening using **IfcRelFillsElement**. The concept is illustrated in Figure 3-22. All simulation tools are therefore capable of deriving such relationships when required.

Internal and external walls and slabs

Be06 and iDbuild require a possibility to distinguish between internal and external walls. This is possible using the **IsExternal** property defined in **Pset_WallCommon** or **Pset_SlabCommon** as this property is intended to separate internal and external walls and slabs. Since some external walls and slabs could be placed in the open, thereby not bounding any spaces, it is necessary to supplement this property with information on whether the wall or slab is acting as a boundary for one or more spaces and hereby being part of the building envelope or not. As described in 3.2.3.2 *Input Data for Spaces* this can be done via the **IfcRelSpaceBoundary**, which allow for the simulation tools to derive which external walls and slabs are part of the building envelope.

Conditions on other side

All simulation tools further require an ability to define the conditions on the side of either a wall or slab facing away from a space. A space on the other side can be identified via **IfcRelSpaceBoundary**, however, in case of the other side being either ground or outside air, it will require a further analysis of whether the placement of a wall or slab is below or above the terrain level. The ground can be represented by geometry attached to **IfcSite** as described in section 3.2.3.1 *Input Data for Projects, Sites, Buildings and Stories* and it should therefore be possible to derive the conditions at the outside of a wall or slab as the placement of each object is known. Be06 further requires information of a

design temperature which corresponds to the conditions on the other side. This information cannot be defined if the outside is below ground as such a value cannot be defined for the ground. In most cases the value will relate to standard values depending on the conditions and it should therefore be possible derived the information in some cases.

3.2.3.4 Input Data for Windows and Doors

Geometry and placement

Most simulation software requires information on area and slope for windows and doors. The area information can be derived directly from the properties OverallHeight and OverallWidth assigned to either IfcWindow or IfcDoor. The perimeter can also be derived from these properties, which is required by Be06. The slope and the defined shape of the window must however be retrieved from the 3D geometry representation also assigned to IfcWindow or IfcDoor. The placement can also be assigned to both objects. This is required by several simulation tools in order to place the windows and doors correctly in the walls. It is further required for Be06, iDbuild and Riuska in order to derive information on the recess of the windows. Since the windows and doors are most likely inserted into a wall or a roof, the ability to distinguish internal and external objects and to evaluate the conditions on the other side of windows and doors can be derived from their related wall or roof slab.

Material properties

Most material properties for windows and doors are defined in either property sets or property entities. The construction of windows and doors are defined by property entities containing lining (frame) and panel properties. Using these two property entities, it is possible to describe the construction of the window in very great detail. The concept of assigning the property entities to windows and doors is illustrated in Figure 3-23. It is too extensive to describe the way of deriving information on the fill percentage of a panel here, since it involves a complex structure of defining frames and panel areas but the possibility does exist and is described in great details in [IAI 2007a]. The frame width can be directly retrieved from the lining properties.

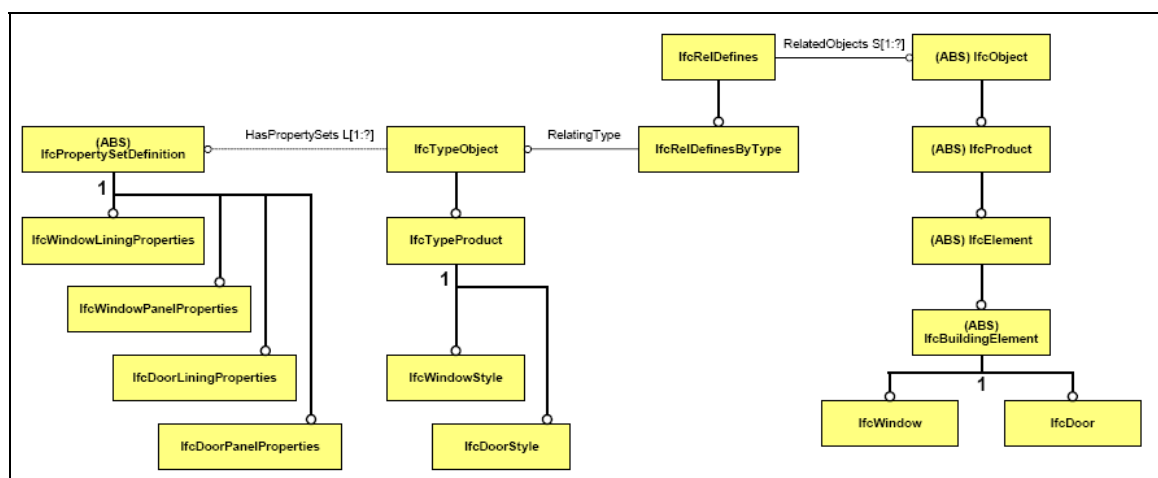


Figure 3-23. Assignment of lining and panel properties to IfcWindow and IfcDoor [IAI 2004].

Both property entities containing lining and panel properties can have materials assigned via `IfcRelAssociatesMaterial` and hereby allow for material properties to be assigned similar to walls and slabs. Further material properties for the glazing are defined in `Pset_DoorWindowGlazingType` and include U-value, g-value, light transmittance and direct solar transmittance. For Be06 which requires the overall U-value and a precise ratio of glass over frame, these values could also be defined in either `Pset_WindowCommon` or `Pset_DoorCommon`. The linear loss required from iDbuild, Be06 and BSim can currently not be defined in IFC similar to the case of walls and slabs.

Shading

Although most of the simulation tools require detailed description of the shading devices which affects windows and doors, limited possibilities exist in IFC to define such information. From an object oriented point of view the idea of applying detailed descriptions of the shape and placement of shading objects to either a window or door is not preferred although this is widely used by the simulation tools. Creating a relationship between each window or door and all the elements which generates shading is also complicated as this could involve a significant amount of relations as many parts of the building could be contributing to the shading for one or more windows. It must therefore be a task for the simulations tools to conduct an evaluation of potential shading elements and retrieve required information from here. As this is not currently possible in any of the simulation tools and poses several challenges, the possibility is seen as only a potential possibility. Since all potential shading devices can be modelled either as an `IfcSlab` or similar, it should be possible to retrieve the required information on shape and placement. iDbuild and Be06 further requires information on shading from surroundings, however, it cannot be expected that surroundings has been modelled, and the information cannot be included elsewhere in IFC.

3.2.3.5 Input Data for Furniture

Geometry and placement

The requirements for geometry and placement of furniture is primarily required by Flovent. The furniture should be defined as `IfcFurnishingElements` which can have both geometry representations and placement assigned.

Material Properties

As with all previously described elements, `IfcFurnishingElements` can also have material properties assigned. Flovent can hereby retrieve all material properties except information on roughness. iDbuild can additionally retrieve information on the heat capacity of the furniture from the material properties which is required.

Type

The type of furniture can be required in order to identify the locations of occupants as described in section 3.2.3.2 *Input Data for Spaces*. Assigning `IfcFurnitureType` to each object allow for distinguishing between e.g. chairs and desks and should therefore be used in such a case.

3.2.3.6 Input Data for Ventilation systems

As described in *3.2.2.3 Definition of Possible Locations for Non-Geometry Information in IFC*, all information that affects the performance of spaces should be attached to the space on not to the system to ensure consistency. The information assigned to systems should therefore only relate to the performance of the system such as information on power use and efficiency.

System type

The property set Pset_AirSideSystemInformation, which describes general information on ventilation systems, can be used to identify the type of system as required by Riuska and BSim. For a reasonable amount of details in the properties, all other information must however be derived from the specific components in the system.

Air flow rate and schedule

As previously described, the total air flow rate required in the ventilation system must be derived from the air flow rate in each space. The air flow rate for both supply and exhaust air is defined as time series in Pset_SpaceThermalPHistory and the schedule for the ventilation system can thereby be derived from these time series.

Air terminal devices

Using Pset_AirTerminalPHistory it is also possible to define the air flow rate from each air terminal device, and this information can replace the summarised air flow rate defined for the space in Pset_SpaceThermalPHistory. Flovent requires the existence of air terminal devices as many properties must be derived from here. The placement and geometry representation can be derived from the entity IfcFlowTerminal. The effective opening area and the flow pattern of the air terminal devices can be defined in Pset_AirTerminalTypeCommon. Flovent, however, needs different information on the flow pattern than what can be defined in IFC, although the type of the air terminal can be defined. The type of device (supply or exhaust) is required by all simulation tools to separate the flows and this information can be defined in Pset_FlowTerminalAirTerminal. BSim requires information about the air source of the ventilation air. There is no possibility to assign this information to e.g. the system; however, in case all components in the system are defined, an analysis of its composition could potentially reveal the source of the air.

Inlet temperature

Information on the inlet temperature in the ventilation system is required by most simulation tools. This information can, however, not be defined for the system. Although flow temperature information could be defined for a random IfcDistributionPort somewhere in the ventilation system as described in *section 3.2.2.4 Use of Properties in IFC to Store Information* this is believed to be too uncertain for deriving a correct value. In this case it is therefore required that air terminal devices are modelled, as the inlet temperature can be assigned to air terminal devices defined as time series in Pset_AirTerminalPHistory. An overall value for inlet temperature can therefore only sometimes be defined based on the air terminal devices in the system. Riuska additionally requires information on the corresponding outdoor temperatures for summer and winter. These values cannot be defined specifically in IFC and since they do not usually relate to the standard design temperatures, they cannot be derived precisely.

Set points and sensors

As described in section 3.2.3.2 *Input Data for Spaces*, the set points for heating and cooling can be derived from Pset_SpaceThermalRequirements. However, if different set points need to be given for the ventilation system and the heating system – as could be the case in BSim – it requires that sensors objects are used. For ventilation systems the Pset_SensorTypeTemperatureSensor can be used to define the set point for each sensor and by nesting several sensors, all set points for the system can be defined. In case sensors are defined, their set point should replace the set point information defined in Pset_SpaceThermalRequirements to ensure consistency.

Fans

The efficiency of the fans is required by BSim and Riuska and can be defined in Pset_FanPHistory as the overall efficiency. Furthermore, Riuska and BSim require information on the pressure rise over the fans and this value can be defined in Pset_FanTypeCommon. Riuska also requires information on the ratio between flow and power consumption. Both an curve for efficiency and the power consumption can be defined in Pset_FanPHistory and as the flow is already known, this ratio should be possible to derived. Additionally information on the type of fan control is required which can be derived from Pset_FanTypeCommon, although the naming convention is not identical. The temperature rise over the fan also required by Riuska cannot be defined as a property in IFC; however, this value is usually derived from the energy consumption of the fan. iDbuild and Be06 requires information on the overall power use of the fans and controls, however, as this would include a large range of objects it is decided that it should be able to define this total value for the entire system. Currently this is not possible in IFC.

Recovery unit

The efficiency of the recovery unit is also required by most simulation tools and this value can be defined in Pset_AirToAirHeatRecoveryPHist as the total efficiency. BSim requires a detailed description of the efficiency and as the total efficiency in Pset_AirToAirHeatRecoveryPHist is described as a time series it should be possible to derive the information from different times of the year from here. iDbuild requires information on a possible bypass in the recovery unit and to include such information it is required to include an IfcFlowController via aggregation as described in 3.2.2.4 *Use of Properties in IFC to Store Information*. By doing so, the recovery unit can be composed by a damper as well if required.

Heating coil

Riuska and BSim requires information on the available output from the heating coil and this can be defined in Pset_CoilTypeCommon. In case the heating coil is not part of a fluid flow system and instead supplied by electricity, this can be defined by assigning IfcEnergyProperties to the coil object and use this property entity to define the energy source which is required by Be06.

Cooling coil

Riuska and BSim also requires information on the available output from the cooling coil and this can equivalently to the heating coil be defined in Pset_CoilTypeCommon. From this property set information on the surface temperature can also be defined which is required by BSim. Be06 further

requires information on an extension factor relating to condensation. Such value can, however, not be defined in IFC.

3.2.3.7 Input Data for Cooling Systems

Chillers

Both iDbuild and Be06 requires information on the efficiency of the cooling system and this information must be assigned to a chiller object in the cooling system using Pset_ChillerTypeCommon.

3.2.3.8 Input Data for Heating and Domestic Hot Water Systems

Heat load

As long as space heaters in the spaces are not defined, the heat output information should be attached to the spaces similar to air flow rate. Although heating is not part of the defined list of sources to be used in IfcSpaceThermalLoadProperties, heating output in a space can be defined using this property entity. Both minimum and maximum values and a schedule can thereby be derived as the heating output is then defined as a time series.

Heating devices

When space heaters are defined, the heat output can be defined in Pset_SpaceHeaterPHistoryCommon as a time series and from here minimum and maximum values along with the schedule can also be derived. When the heat output is assigned to the space heater, it should replace the information assigned to the space. Flovent additionally requires similar properties for heating devices in a space as for walls and slabs. This relates to geometry representation, placement and surface properties. As heating devices should be defined as IfcEnergyConversionDevices, the required properties for geometry, placement and emissivity can be defined. Roughness can, similar to the other building elements, not be defined.

Set points

Similar to ventilation, the information on set points can be derived from Pset_SpaceThermalRequirements or defined by a sensor. When assigned to a sensor object the set point for heating systems can be defined in Pset_SensorTypeHeatSensor. BSim further requires the location of the sensor and this can be derived by the relationship between the sensor object and the space using IfcRelContainedInSpatialStructure. An outdoor design temperature and outdoor temperature limit is further required to control the heating systems in BSim, however, these cannot be defined in IFC and since this information does not necessarily relate to the standard design temperatures, they cannot be derived.

System type

Be06 requires information on the system layout type; however, such type information cannot be defined for the system. In case the whole system is represented in the BIM model, the information could be derived by analysing the composition of the system although this can be a complicated task.

Be06 and iDbuild additionally requires information on the energy source of the heating system. As described below, such information can be defined for the boilers in fluid flow systems and hereby derived for all objects in the system. In case a heating device is supplied by electricity, IfcEnergyProperties can be assigned to the specific instance to define the energy source.

Boilers and tanks

Be06 requires information on both boilers and district heating exchangers, however, as district heating exchangers are defined in the same way as boilers in IFC, only boilers are described in the following. Most of the required information from Be06 and iDbuild of performance of the heating system relates to the boilers and can therefore be assigned to one or more boiler objects. The efficiency and power consumption of each boiler can be defined in Pset_BoilerPHistory as time series and by assigning IfcEnergyProperties to the boiler, the energy source can also be defined. In case the boiler includes water storage capabilities, the capacity can be defined in Pset_BoilerTypeCommon. If the water is contained in a separate tank the capacity can be defined in Pset_TankTypeCommon. Information on the intended water temperature in the tank or boiler and the supply and return temperatures of the heating system required by Be06 can be defined by information on the outlet and inlet water temperature for the boiler in Pset_BoilerTypeCommon. This information is defined as possible ranges of temperatures; however, it is the maximum temperature for the outlet and the minimum temperature for the inlet which is required by Be06.

Domestic hot water consumption

Although the total water consumption can be defined in Pset_UtilityConsumption which is assigned to the building, it is not possible to define the share of the total water consumption which consists of hot water. It is not possible to define such a value for each water tap either, and the annual use of hot water in the building can therefore not be derived.

Pumps

Be06 also requires information on the power consumption of pumps in the heating systems. This information can be derived from Pset_PumpPHistory and is defined as a time series which further allows for the identification of a schedule. Additionally Be06 requires information about the share of the power consumption which is transferred to the water, however, such information cannot be defined in IFC.

3.2.3.9 Input Data for Lighting Systems

Loads and schedule

As described in 3.2.3.2 *Input Data for Spaces* the information on lighting load in each space can be defined using IfcSpaceThermalLoadProperties and since the information is defined as a time series, a schedule can also be derived. BSim, Be06 and iDbuild, however, requires a possibility to separate general and task lighting. As this separation cannot be defined on a system level, it requires the definitions of individual lighting fixtures. Unfortunately the definition of load from each light fixture cannot be defined as a time series, and any schedule information is thereby lost. In such a case, it will only be the max power of each lighting fixture that can be defined using

Pset_LightFixtureTypeCommon and this information should potentially replace the information defined in IfcSpaceThermalLoadProperties. It is also in this property set that information regarding the type of lighting can be defined for the simulation tools to derive whether the lighting fixture relates to general or task lighting.

Type and light level

BSim requires information about the lighting type and this information is defined as a type for one or more lamps within a lighting fixture using IfcLampType. The connection between a lamp object and a lighting fixture is established via an *IfcDistributionPort* as described in section 3.2.2.4 *Use of Properties in IFC to Store Information*. Be06 and iDbuild further requires information about the lighting level and this information is also assigned to each lamp using Pset_LampTypeCommon. iDbuild further requires a relation between the lighting level and the power use, however, this information can currently not be defined in IFC

Set points

The set point for lighting level can only be defined using sensors. The set point should be defined in Pset_SensorTypeLightSensor for each sensor. As the lighting fixtures can have different set points depending on their type, a sensor must be assigned either to systems grouping each type or to the individual lighting fixtures. BSim alternatively requires a temperature set point and in such a case the set point should be defined in Pset_SensorTypeTemperatureSensor. Be06 further requires a definition for the daylight factor; however, such information can currently not be defined in IFC.

3.2.3.10 Input Data for Electrical components

Load and schedule

Be06 and iDbuild finally require information on any additional energy consumptions from building services objects. As described in the previous sections such energy consumptions can be derived from all type of electrical components. It is, however, only some objects which can hold the information as a time series and annual consumptions can therefore not be derived precisely. Since the value cannot be assigned to e.g. the building as a summarised value either, such information cannot currently be defined.

3.2.3.11 Output Data

The output data from the simulation tools has been selected from a large range of possibilities as described in 3.2.1 *Identification of Input and Output Data for Simulation Tools*. The selected output data primarily consists of annual variations, annual summations, min/max values and from Flovent detail distributions of values in the spaces. As described in 3.2.1.5 *Input/Output Data Analysis of Flovent* it is required to create a reference to the data set of values or a visual presentation of the data from Flovent as the data amount would be too large to include in the BIM. Such a reference can be created by *IfcRelAssociatesDocument* and allow for the definition of an address to the data via *IfcDocumentReference*.

In order to allow for consistency with the defined locations of input data in IFC, it would be preferred if the output data could be defined in similar locations so that the output data can be used in further simulations. The previous sections aimed at defining performance information in time series and the same principle should therefore be applied for the output data.

Space temperature and air flow rate

The temperature and air flow rate variations simulated by iDbuild and BSim should be defined in Pset_SpaceThermalPHistory similar to the input data requirement. The property VentilationAirFlowRate and ExhaustAirFlowRate could be used simultaneously in case of diverse supply and exhaust rate. Riuska only defines a minimum and maximum value for the temperature and air flow rate and this can only be stored in Pset_SpaceThermalDesign as HeatingDesignAirflow and CoolingDesignAirflow respectively.

Daylight in spaces

iDbuild is capable of calculating the daylight level in each space, however, currently there is no possibilities to define this information in IFC.

Comfort level in spaces

iDbuild, Riuska and Flovent is capable of assessing the comfort level in each space, but currently this information cannot be defined in IFC either.

Energy consumption for spaces

iDbuild calculates various annual energy consumptions for each space; however, it is not possible to assign such information individually to each space.

Energy consumption for building

The energy consumptions for the entire building are calculated by Riuska, Be06 and BSim. Although they define several key output data, only heating energy consumption and electricity consumption can be defined. These can however be defined as time series which thereby support both monthly and annual values.

3.2.4 Identified Capabilities for Interoperability

The descriptions of capabilities have lead to a result of which information from the input and output data can be defined using the IFC specifications. For an overview of the possibilities, Appendix 9-13 contains the input and output data with colour codes to identify the capabilities of IFC in relation to each simulation tool. The information has been divided into four categories relating to the descriptions in the previous sections. These categories describe information that can be defined directly as an attribute or property value (green), information that can be derived as described in the previous sections (yellow), information that cannot currently be defined in IFC (red) and information which only sometimes is defined sufficiently (blue).

Table 3-1 summarises the findings for each simulation tool based on the total number of input and output data.

Table 3-1. Capabilities of IFC to handle input and output data. Values defined as a percentage of total number of input and output data.

	Information defined in IFC (green)	Information derived from IFC (yellow)	Information unable to be define in IFC (red)	Information sometimes defined in IFC (blue)
iDbuild	40 %	22 %	25 %	12 %
Riuska	47 %	25 %	14 %	13 %
BSim	50 %	29 %	12 %	9 %
Be06	37 %	29 %	22 %	13 %
Flovent	33 %	35 %	24 %	8 %
Average	42 %	28 %	19 %	11 %
	70 %		30 %	

As the previous sections have described, it requires a significant amount of IFC attributes to achieve the results presented in the table above. An average of 30 % of information can still not be defined or is only sometimes defined sufficiently, and capabilities for IFC to handle the required input and output data of the simulation tools are therefore currently not fully sufficient. Most of the information which cannot be defined in IFC simply relates to an attribute missing in a property set and as IFC allow for the creations of custom property sets, such information could be included if agreements were reached among simulation tool vendors.

In general, the lack of consistency in the definition of performance parameters is found most critical to the success of interoperability among simulation tools because the performance parameter constitutes the framework for all of the simulations. Information on performance parameters should preferably be assigned as time series; however, this is only the case for some of these parameters. Occupancy and load from individual lighting fixtures are examples of such values which at present are defined only as maximum values. They should, however, be defined as time series similar to air flow rate and heat load as information on variation and a schedule needs to be derived from all these attributes. As the possibilities of defining the information as time series exist in IFC, it is only a matter of changing the definition of e.g. occupancy and load from lighting fixtures to be defined as time series to achieve improved possibilities for interoperability.

It is out of the scope of this thesis to propose solutions to all missing properties in the current IFC specifications. However, a list has been created in Appendix 14 with corrections to property sets which have been found appropriate to implement in the IFC specifications based on the current analysis. Changing the IFC specifications to include these suggested improvements would allow for 51 additional input and output data to be defined and this would increase the total number of input and output data capable of being defined to an average of 85 %.

Information on linear loss along the edge of construction objects is a type of information which cannot currently be defined using the IFC specifications. It could be a challenge to include support for such information as it relates to the gap between objects and not to a specific object and it will

therefore be difficult to assign the information to objects consistently. Detailed information on control of the ventilation system, among others, was intentionally skipped in previous extensions of the IFC specifications as described in section 3.2.2.3 *Definition of Possible Locations for Non-Geometry Information in IFC* due to a lack of consensus of what information is needed. This is to some extent also identified in this analysis and a greater extent of consensus among simulation tools should therefore be reached before improvements can be implemented.

Another challenge identified is the way some information in the IFC specifications is required to be defined as type descriptions. In most cases the descriptions should be selected from a predefined list of types, however, in some cases the type descriptions should follow appropriate guidelines, such as in the case of occupancy type. As this allow for significant challenges in relation of consistent use of the descriptions, it would be preferred if agreements were reached so that a predefined list is available for all type descriptions.

Although shortcomings have been identified in the IFC specifications, it is important to notice that an average of 70 % of the required information can either be defined or derived from a data repository defined by use of the IFC specifications. This value should be significantly higher before interoperability at a desired level can be achieved. However, as described, slight modifications to the IFC specifications could improve the capabilities in important areas and allow for up 85 % of the information to be defined. Even though the possibilities are moderate, the current capabilities for interoperability could still be used to improve IBD to a reasonable extent.

Before interoperability can be achieved, implementation of IFC import and export capabilities in the simulation tools, however, further needs to be developed. As for now, Riuska is the only tool of the five selected, which holds limited capabilities for import and export information via IFC files. When an average of 70 % of the required information is identified to be possible to exchange via IFC, this should constitute increasing encouragement for the software developers to start the development of IFC interfaces for their simulation tools.

3.3 IFC Capabilities for Management and Control in IBD

The possibilities for improved interoperability was mainly capable to support the decision-making process by allowing for improved use of simulation tools and hereby an improved basis for decision. In order to ensure that all decisions are taken within the space of solutions defined, information management and control was identified as an additional capability in IBD to ensure an improved process.

In section 2.4.3 *Information Management and Control* it was identified that effective management in IBD would require three tasks:

1. Proper management of constraints in the space of solution
2. A relation between the design and the space of solutions
3. Monitoring and checking of the development in design

As described in section 2.2.7 *Summary on Integrated Design* most information would typically be shared via documents or verbal communication during IDP, and this is also believed to be the case for the content in a space of solutions. As previously described, the space of solutions could include a significant amount of constraints of parameters as the design progresses. This makes it hard to retrieve the correct information and even harder to check for continuous consistency between the design decisions and the constraints in the space of solutions when information is most likely spread across several documents. It would therefore be preferred if the space of solutions could be defined in a structure which would allow for easier access to the information. As a space of solutions consists of constraints for parameters of objects in the BIM, it could be possible to define the values in an object oriented environment.

3.3.1 Space of Solutions in a Requirements Model

Constraints in the space of solutions defined in an object oriented model would be very similar to the content of the client requirements model described in section 2.3.7 *Definition of a Client Requirements Model* since both the space of solutions and the client requirements is intended to define constraints/requirements for objects in the design model. A requirements model containing the space of solutions would therefore be an obvious solution to support the tasks mentioned for effective management in IBD. It would allow for proper management of the constraints in the space of solutions, and it could create a relation between the design model and the space of solutions similar to the one described for client requirements in Figure 2-18.

As constraints in the space of solutions is typically derived from simulation results, they are believed to consist of numerical values only – as oppose to the client requirements where numerical values could constitute as little as 30 % as described in section 2.3.6 *Code Compliance and BIM Evaluation*. Even though the space of solutions would preferably consist of both fixed and variable constraints as described in section 2.2.6 *Sensitivity Analysis*, the use of numerical values should allow for much improved capabilities to automatically check and monitor that the design complies with the space of solutions.

[Kiviniemi 2005] has already conducted a thorough analysis of the capabilities of IFC to support the definition of a requirements model and how to maintain a link between the requirements and the design model using IFC. It is concluded in his findings that it requires modifications to the IFC specifications to support both tasks. To support the task of storing requirement information, a new range of requirement objects must be defined as described in section 2.3.7 *Definition of a Client Requirements Model* and for linking the models, it is required to extend `IfcExternalReference` and `IfcRelAssociates`. `IfcExternalReference` should be extended so that it is possible to reference to other models and `IfcRelAssociates` should be extended so that objects in a design model can have the external model assigned. It will be too extensive to account for all of the extensions to IFC suggested by [Kiviniemi 2005] as the focus in the present analysis is primarily on current possibilities in IFC. The latest version of IFC (version 2x3) was released after the completion of the research in [Kiviniemi 2005]. However, the improvements in the suggested areas have not been implemented in this version of the IFC specifications.

As this thesis primarily intend to assess current capabilities of IFC, it must be concluded that currently a proper possibility for storing client requirements related information is limited. To ensure that the same limitations apply to a requirements model containing a space of solutions, the following sections will briefly analyse whether the prerequisites are the same for requirement models containing a space of solutions and client requirements.

3.3.1.1 Separation of Requirements and Design Model

It is argued in 2.3.7 *Definition of a Client Requirements Model* that a client requirements model should be separated from the design model. First of all, this should be done to maintain proper management of the requirements by ensuring that design and requirements can be created and maintained independently. It must further be ensured that the requirements can be applied to all design models as several design alternatives could coexist. Such a case is very likely in IBD when iterations are used to evaluate and identify optimal alternatives, and in this case it will be important that all design alternatives can be evaluated in relation to the space of solutions as illustrated in Figure 3-24.

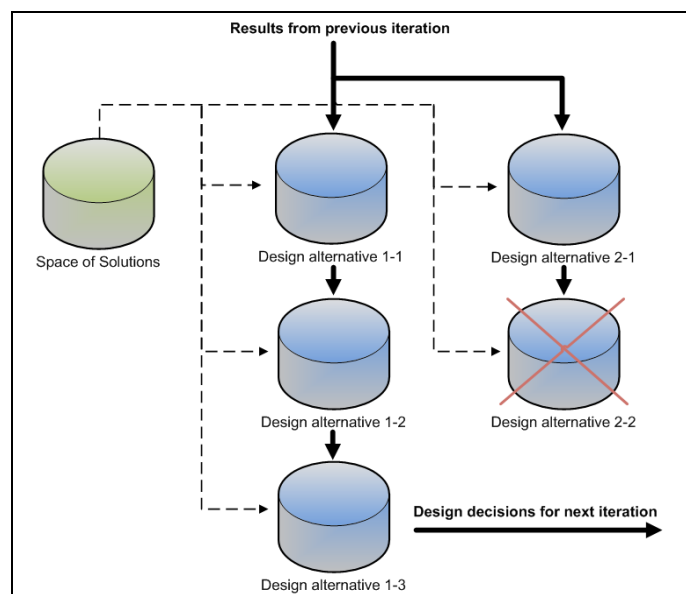


Figure 3-24. Evaluation of space of solutions in relation to design alternatives in IBD should be possible for all design alternatives. Illustration inspired by [Kiviniemi 2005].

For this reason it is found that the space of solutions should also be defined as a requirements model as the constraints should serve as boundary conditions for all design alternatives. This requires a possibility to link the requirements model with the design models, as it was the case for client requirements.

3.3.1.2 Direct and Indirect Linking

As described in section 2.3.7 *Definition of a Client Requirements Model* it is found, that in order to create a proper link between client requirements and the design model, both direct and indirect

linking is needed. Direct linking should be used between e.g. a space and the requirements for that space, however, as most requirements for a space typically relates to its boundaries or supply systems most requirements relates to other objects in the design model than the space itself. The linking of such requirements will therefore be indirect. The idea is illustrated in Figure 3-25.

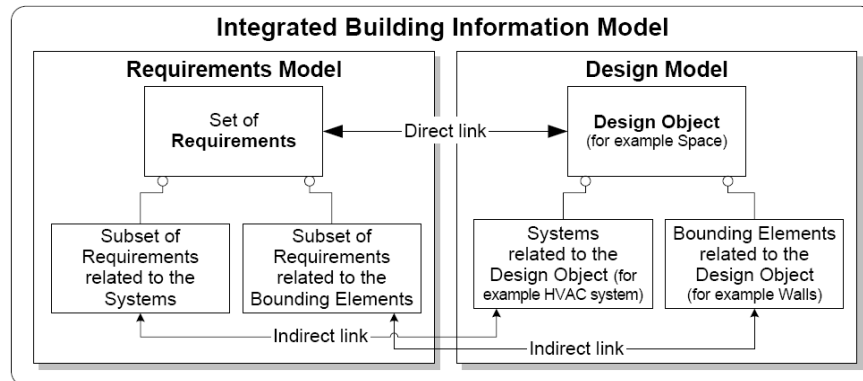


Figure 3-25. Concept of direct and indirect linking between requirements and design model [Kiviniemi 2005].

As described in section 2.2.5 *Space of Solutions*, a space of solutions can include many different parameters, however, the performance-decisive parameters suggested for a space are listed in Appendix 1. Out of the 18 parameters defined here, only 3 relates to the space object itself which indicates the demand for indirect linking. In case of design decisions focusing on the overall building shape instead of individual spaces, similar sets of parameters could be applied to the building envelope. A space of solutions for the building envelope would contain requirements for both walls and slabs but also for the windows and doors inserted, and this would also require the use of indirect linking. There is therefore a similar demand in the space of solutions to be able to use both direct and indirect linking for the ability to manage and apply requirements properly.

For the separation indicated in Figure 3-25 it should be noted that the integrated BIM is composed by both the requirements model and the design model, as all information in the models relate to the digital representation of the building.

3.3.1.3 Storing Constraints in IFC

The IFC specifications currently allow for constraints to be defined in three ways; using *IfcControl*, *IfcConstraint* or in property sets containing requirement assigned to individual objects.

[Kiviniemi 2005] uses *IfcControl* for the definition of his suggested requirements objects which includes a large range of new entities intended to define constraints for the objects and systems in the scope of his research. The scope of *IfcControl* is to define sets of requirements in specific entities which can then be assigned to related objects using *IfcRelAssignsToControl*. All of the requirements objects must be subtypes of *IfcControl* in this regard.

In case of the requirements model and the design model being separated, [Kiviniemi 2005] suggests that the definition for *IfcRelAssociates* is extended as previously described in section 3.3.1 *Space of Solutions in a Requirements Model*. This would allow for the linking between objects in the design

models with the requirements objects in the requirements model. An example of a suggested requirements object is illustrated in Figure 3-26 for an entity [Kiviniemi 2005] defines as *NewSpaceIndoorClimate*. In this requirements object, client requirements for indoor climate in a space can be defined.

```

ENTITY NewSpaceIndoorClimate;
  SUBTYPE OF (NewRequirement);
    MaxHvacNoiseLevel : OPTIONAL NewRequirementSound;
    MaxTemperature : OPTIONAL NewRequirementTemperature;
    MinTemperature : OPTIONAL NewRequirementTemperature;
    IndividualRoomTemperatureControl : OPTIONAL NewRequirementTemperature;
    MaxHumidity : OPTIONAL NewRequirementRatio;
    MinHumidity : OPTIONAL NewRequirementRatio;
    MaxAirVelocity : OPTIONAL NewRequirementAttribute;
    MinAirflowPerPerson : OPTIONAL NewRequirementAttribute;
    MinNoOccupancyAirChangeRate : OPTIONAL NewRequirementRatio;
    MaxFloorTemperature : OPTIONAL NewRequirementTemperature;
    MinFloorTemperature : OPTIONAL NewRequirementTemperature;
    TemporarilyVentilationControl : OPTIONAL NewRequirementRatio;
    AllowedTemporaryDeviation : OPTIONAL NewRequirementTemperature;
    MaxVerticalTemperatureDifference : OPTIONAL NewRequirementTemperature;
END_ENTITY;

```

Figure 3-26. Example of a requirements object for client requirements from [Kiviniemi 2005].

One of the disadvantages of using *IfcControl* is that it requires entities for each set of requirements that needs to be defined – such as the *NewSpaceIndoorClimate* suggested for climate related requirements.

The entity *IfcSpaceProgram* is one of the only current entities in the IFC specifications which is a subtype of *IfcControl*. *IfcSpaceProgram* can be used to define constraints to the area size of a space. It can further have a property set, *Pset_SpaceProgramCommon*, assigned in which descriptions of various systems and the occupants can be defined. Compared to the constraints for a space of solutions described in Appendix 1, none of these can be defined in *IfcSpaceProgram* or *Pset_SpaceProgramCommon*. For a space of solutions to be defined using *IfcControl*, it would therefore require that a new set of entities should be defined in IFC, similar to the one illustrated in Figure 3-26.

Alternatively *IfcConstraint* could be used, which allow for the assignment of constraints to any existing attributes in IFC via *IfcRelAssociatesConstraint*. The disadvantage of this concept is, however, that constraints would have to be assigned to the attributes in the design model, as this is where the attributes would exist. As described in 3.3.1.1 *Separation of Requirements and Design Model* there is a demand for several reasons to separate the requirements and design in different models, and the use of *IfcConstraint* is therefore not possible as the requirements model would not contain the object which the constraints needs to be assigned to. Furthermore, this concept would not allow for indirect linking as constraints would have to be uniquely assigned to affected attributes. Indirect linking is created via relationships and since relationships can change without affecting the object directly linked with the requirements no unique assignment is possible for indirect linking.

The last alternative for defining constraints in IFC is via property sets assigned directly to objects. Currently several requirements property sets allow for the assignment of constraints to spaces and

their supply systems. One such example is Pset_SpaceThermalRequirements in which requirements for thermal performance in a space can be defined. Such property sets, however, also needs to be assigned to the objects in the design model and as this solution therefore poses similar issues in regards to the separation of the requirements model and design model, this option is not possible either.

In order to define constraints for a space of solutions in IFC in an appropriate way, it must therefore be concluded that it requires the use of IfcControl similar to the case of client requirements. Before the constraints can be defined in IFC, it will require that a new set of “space of solution”-entities are defined, similar to the requirements objects suggested by [Kiviniemi 2005].

Once such an implementation has been done, it will further require the creation of a management system that would allow for the creation and management of the requirements model containing the space of solutions. As described in section 2.3.7 *Definition of a Client Requirements Model* some limited possibilities exist today to manage client requirements, however, this area should be extended much further before proper management during IBD can be efficient.

3.3.2 Capabilities of IFC for Compliance Checking

Proper management during IBD has additionally been identified to require one more task which is the possibility to ensure that the design decisions comply with the space of solutions. As described in section 2.3.6 *Code Compliance and BIM Evaluation* Solibri Model Checker is currently able to conduct compliance checking of BIMs defined in IFC based on parameterised rule sets. As described in section 3.3.1 *Space of Solutions in a Requirements Model*, it is expected that all constraints in a space of solutions will be defined as numerical values. In this case rule sets in Solibri Model Checker or similar tools could be used to represent the constraints in a space of solutions.

Assuming that the space of solutions in the future will be defined in a requirements model based on IFC, all constraints will exist in an object oriented environment which can be accessed and easily understood by other applications. A tool such as Solibri Model Checker could then via IFC retrieve information on constraints from the requirements model and the latest design from the design model as illustrated in Figure 3-27. Either direct or indirect linking will then allow Solibri Model Checker to conduct a check to see if all design information is compliant with the space of solutions.

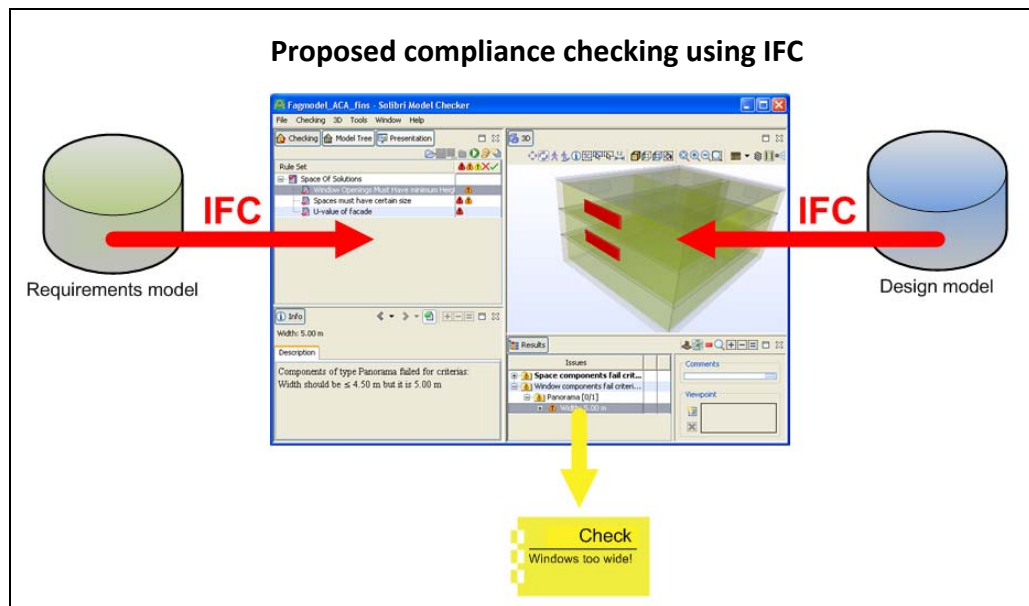


Figure 3-27. Potential automatic compliance checking using Solibri Model Checker as the engine for checking.

Currently, as a space of solutions cannot be defined in a requirements model based on IFC, it will require that the rule sets need to be manually created in Solibri Model Checker or other compliance checking tools. The import of the design model via IFC is already a possibility and this does therefore allow for semi-automatic checking of the design in a data repository defined in the IFC format although it must be manually controlled.

To allow for proper checking of the design, it will obviously require that the information that needs to be checked is available in the design model. As described in section 2.2.3 *Integrated Design Process* the checking of design should as a minimum be conducted in the transitions from one iteration to the next when design decisions have been taken from the basis of the results in the previous iteration.

At this stage it can be expected that the latest simulation results are available and potentially represented in the BIM. Although the content of the space of solutions can vary, at least for the scope of this thesis, it could be assumed that all constraints would relate to either input or output data of the simulation tools previously described. As identified in section 3.2.4 *Identified Capabilities for Interoperability*, only 70 % of the input and output data can be properly defined in IFC, and this additionally results in the possibilities for compliance checking to be limited to the same extent as interoperability between the simulation tools.

3.3.3 Identified Capabilities for Information Management and Control

The possibilities to currently use the IFC specifications for management and control in relation to a space of solutions are therefore found to be limited in several ways. First of all, the specifications for a requirements model which is able to define a space of solutions combined with the possibilities in IFC requires that a new set of entities are defined. These entities should be able to contain

constraints for a space of solutions. Secondly, several corrections must be included in the IFC specifications similar to the ones required by [Kiviniemi 2005] for his client requirements model for the possibility of linking the design and requirements models. Finally, the possibilities to define required information in a design model based on IFC must be improved to ensure that the design can be properly checked for compliance with the space of solutions. Additionally to the improvements to the IFC specifications, requirements management solutions have to be developed which are capable of managing the requirements model and maintaining the link between requirements and design.

For now, the possibilities therefore seems limited but as indicated in the previous section, possibilities does exist for semi-automatic checking of some design parameters in a design model based on IFC using Solibri Model Checker. To extend the possibilities further, it would require significant development of requirements management tools which so far seems to have received only little attention from the research community and software developers.

3.4 Summary of IFC Capabilities

The previous chapter identified that interoperability could support the decision-making process within each iteration in IBD and that information management and control could ensure proper development of the design.

This chapter evaluated the capabilities of IFC to support for such possibilities. It was identified that 70 % of the input and output data required for a selection of simulations tools could be defined using the IFC specifications. Although this means that 30 % of all information required still needs to be handled manually, it is seen as a reasonably high number taking into consideration that only very few simulation tools today include import and/or export capabilities for IFC.

In relation to information management and control the possibilities are even further limited. In order to define constraints in space of solutions using the IFC specifications, it will require a range of new extensions to the specifications in relation to overall demands for a requirements model and in particular for the definition of constraints. Additionally, the limited ability to define information in the design model allow for only partial capabilities for compliance checking.

Chapter 4

IFC Implementation

4.1 Introduction

The previous chapter identified several limitations in the IFC specifications before its capabilities could fully support IBD. Besides the contents in the IFC specifications another challenge for the use of IFC in a shared data repository was identified to be the lack of capabilities in the software to import and export IFC related information. The limited capabilities of IFC do not allow for management tools to interact with a shared data repository via IFC at present. However, simulation tools should already have reasonable opportunities to exchange information. In spite of this, Riuska is so far the only simulation tool of the five selected with capabilities to import and export limited information via an IFC file.

To further explore the possibilities for the simulation tools to interact with the integrated BIM defined using IFC, an IFC interface has been developed to demonstrate some of the current capabilities. It is the intention that the development of this interface can supplement the analysis of the IFC specifications to illustrate current capabilities of IFC.

Although the design team can usually freely select the simulation tools which they prefer to use, a tool like Be06 is mandatory in Denmark to prove the energy consumption of a building. Currently Be06 includes no possibilities to retrieve information from a BIM, which means that all information must be manually defined. Such limitations are in contradiction to the intentions of IBD where a tool like Be06 should be used extensively to ensure a proper basis for decision in each iteration. For this reason, the interface developed will be used to retrieving information required for Be06 which could currently be defined in a BIM.

To maintain a reasonable scope of the development of the IFC interface several limitations apply. The limitations will be mentioned along with the descriptions in the following sections. As the purpose of this chapter is primarily to evaluate the challenges for tools to interact with an IFC file, the descriptions of the software development will primarily relate to the methods used to capture and use IFC data. Descriptions of general software related functionality has not been included. The IFC interface is fully functional and the IFC interface is included on the CD enclosed with the thesis.

In order to interact with a BIM defined using the IFC specifications, it is useful to use a so called IFC toolbox which is capable of mapping applications and IFC files. The chapter will therefore start with an introduction to such a toolbox.

4.2 IFC Toolbox

As the use of model servers is still limited, the current possibilities to interact with a BIM must be via a data repository defined in an IFC file. To read and write to such an IFC file, an IFC toolbox can be used to import an IFC file and make the information in that file available in the software. Most toolboxes are further capable of exporting corrected information back to an IFC file.

[Erabuild 2008] mentions five such toolboxes. One toolbox on the list is the IFCsvr ActiveX Component¹⁰ developed by SECOM Co., Ltd and this toolbox has the advantage of being free to use and is additionally easy to implement into Visual Basic which is the language selected for the creation of the interface in this thesis. This toolbox is therefore selected and will be referred to as IFCsvr in the following.

Some of the possibilities of IFCsvr are described as [SECOM 2005]:

- IFC data import (STEP P21 file and ifcXML file)
- IFC data export (STEP P21 file and ifcXML file)
- IFC object searching
- IFC object updating
- IFC object creation
- IFC object deletion
- GUID generation

The main advantage of the toolbox is that it allows for a possibility to easily access the hierarchy structure of IFC via calls for either the value of an entity or the values of attributes for an entity.

To find out how to call required values, it is useful to explore the structure of the IFC file via an IFC browser such as the one illustrated in Figure 4-1. From here it can be derived that in order to find the name for a wall it will require a call for the value of the name attribute for that wall object. An example of how to make such a call in Visual Basic is illustrated in Figure 4-2.

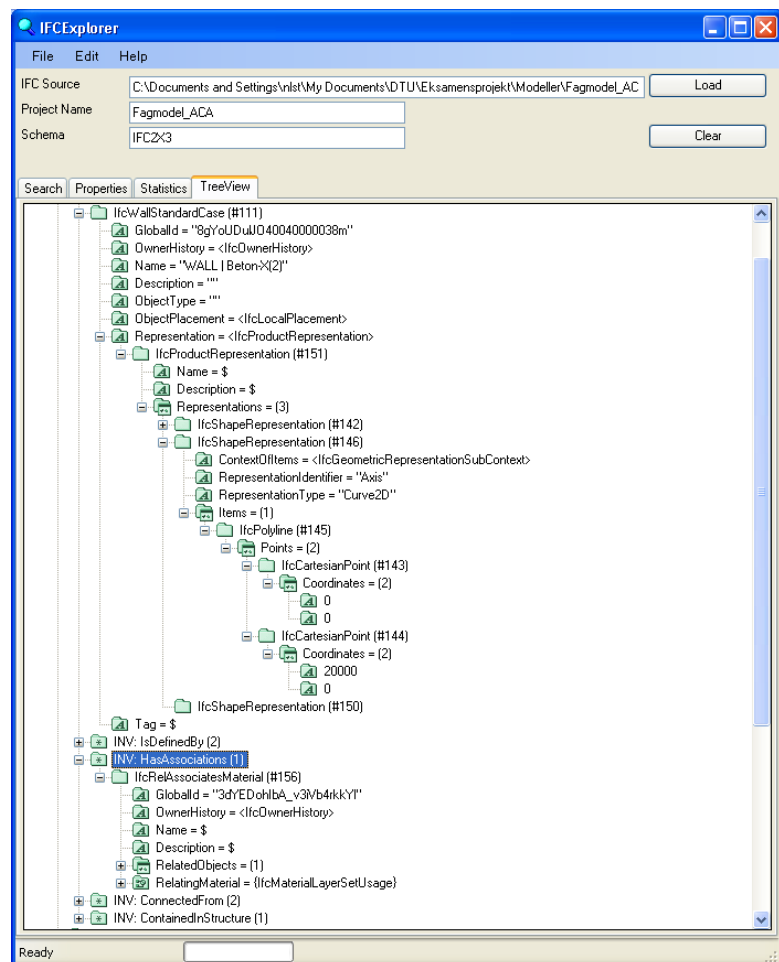


Figure 4-1. View of data structure for an IfcWallStandardCase in IFCExplorer. IFCExplorer also allows for the detection of inverse assigned attributes such as a Material Layer Set which can be seen in the bottom of the figure.

¹⁰ IFCsvr ActiveX Component can be downloaded along with many samples and the IFCExplorer at <http://groups.yahoo.com/group/ifcsvr-users/>.

```

Function FindWallNames()
    objIFCsvr = New IFCsvr.R300
    objDesign = objIFCsvr.OpenDesign("IfcFile.ifc")

    For Each objIfcWall In objDesign.FindObjects("IfcWall")
        WallName = objIfcWall.Attributes("Name").Value
        MsgBox("Wall name of current object is: " & WallName)
    Next objIfcWall

End Function

```

Figure 4-2. Example of Visual Basic script which will use the IFCsvr to display the name of each wall in the file "IfcFile.ifc"

IFCsvr handles the various assignments of attributes in four levels: *Entities*, *Entity*, *Attributes* and *Attribute*. Each object has one or more *Attributes* assigned and each *Attribute* consists of a set of *Entities*, one *Entity*, a set of *Attributes* or a specific value. The *IfcProductRepresentation* in Figure 4-1 e.g. consists of a set of *Attributes* which can be called either by their name such as in Figure 4-2 or by their number in the structure. The attribute "Representations" under *IfcProductRepresentation* consists of three *Entities* which can be called by their number.

In case the value for the first point in the 2D curve of the wall – also illustrated in Figure 4-1 – would be needed, it would require a call that will move through the structure like the following: Wall entity -> Representation -> Representations (2) -> Items (1) -> Points (1) -> Coordinates.

To retrieve values a long way down in the hierarchy structure can therefore require a significant amount of calls and can be further complicated if an attribute consists of a set of entities. In order to find the correct entity from the set of three *IfcShapeRepresentations* it would e.g. require an analysis of the content of each *IfcShapeRepresentation* to ensure that it is the representation containing the "Curve2D" which is selected.

Although it can be complicated, this method allow for all values of an entity and its attributes in IFC to be retrieved. In case the attributes are inversely assigned (using relationships) to an object, it will require a search through all entities in the IFC file, to check if any relationships are pointing to the object in question.

It is also possible to make corrections to any existing values of attributes and additionally create new attributes and entities. The corrections can then be saved by IFCsvr to a new IFC file. The use of IFCsvr thereby allows any application to import and export all IFC data available in an IFC file.

As IFCsvr only provides access to the structure of an IFC file, it requires that the information is defined according to the expected structure. As previously mentioned, many possibilities exist to define the same information in several ways in IFC. An example is that Revit and ArchiCAD exports the swept areas of walls and spaces as a set of points whereas Autocad Architecture currently exports the same swept areas as line segments. Although the same information can be derived from both constellations, it has been decided to primarily support only the way ArchiCAD exports IFC data. The main reason to select ArchiCAD is that this is the only application which currently exports information of the boundaries for each space using *IfcRelSpaceBoundary*. Revit also export space boundaries, however, they are only "virtual" which means that they do not include a relation to the actual object bounding the space. As it will be described later, proper space boundaries are required

for the IFC interface to work, and ArchiCAD must therefore be used to create such space boundaries for now. As both Revit and AutoCAD Architecture hold sufficient information about space boundaries in the applications, it is simply a matter of including this information in the IFC export functionality.

To further limit the scope of the development, it is assumed that all lengths are defined in millimetres. The correct measurement unit can be derived from a specification in `IfcUnit` included in all IFC files in case the interface would require such an extension.

4.3 IFC Interface for Be06

Be06 only requires key information of the building design and performance as input data and as previously described it is a mandatory tool to use in Denmark to ensure a proper energy consumption of new buildings. In section 1.1.4 *Identified capabilities for interoperability*, it was identified that around 65 % of the information required by Be06 could be defined using the IFC specifications. But the range of limitations in current IFC support in most design and simulation tools, does not allow for such a large amount of information to be defined in a BIM based on IFC. Some possibilities are, however, found to be achievable at the moment.

As Be06 requires an extensive amount of information, only some parameters have been selected, as the idea of the interface is primarily to demonstrate the potentials of IFC. The parameters have been selected from some of the current possibilities described in the following section. As building design today can be very complex, the interface is further limited to being capable of handling only spaces with straight surfaces, straight walls and plain slabs. Although Boolean operations can be used to modify the shape of the objects, this will further not be supported by the interface. Additionally, windows and doors are expected only to be inserted in the walls of the building envelope.

4.3.1 Identified Current Possibilities

To identify current possibilities for importing information defined in IFC to Be06, the possibilities of creating such information in a IFC file exported from MagiCAD, AutoCAD, Revit, ArchiCAD and Riuska have been analysed as these applications have been available for the research of this thesis. From Appendix 5 it is identified that Be06 primarily requires information of the building envelope, activities in the building and performance data of building services. As described in section 3.2 *IFC Capabilities for Interoperability in IBD* the possibilities to include performance data of building services is limited because the HVAC related BIM applications are primarily capable of defining information on the duct and piping objects modelled. From MagiCAD all properties are additionally exported using custom property sets and this is not desired because equivalent information cannot be derived from other potential HVAC applications. Properties from MagiCAD will therefore not be used in the interface. The potential information would additionally only be of limited use, as objects from MagiCAD do not yet include a relation to the spaces which they supply and it would therefore require extensive analyses of the BIM to identify which systems supply each space.

The architecturally BIM applications primarily have similar capabilities of supplementing existing objects with additional information. Such information could be the U-value of a window, which ArchiCAD is able to define according to the IFC specifications as illustrated in Figure 4-3. AutoCAD Architecture and Revit are also capable of assigning information to objects according to the IFC specifications although the property sets must be created manually. Beside the additional information, all applications are currently able to export the geometry, materials and type related information of objects to IFC. The possibilities for defining geometry are described in section 3.2.2.2 *Geometrical Representations and Placement in IFC*. From evaluation of the BIM applications, it has been identified that all applications use a Swept Solid representation to define the geometry of spaces with straight surfaces, straight walls and plain slabs. A 2D curve representation is further assigned to all wall objects. The possibilities to assign a material layer set to walls and slabs are described in section 3.2.3.3 *Input Data for Walls and Slabs* and currently all applications assign the material layer set directly to the objects instead of using a type definition. The possibilities for assigning information on the composition of windows and doors are described in section 3.2.3.4 *Input Data for Windows and Doors* and it has been identified that information on the lining and panel properties are assigned via type definitions to windows and door objects by the BIM applications. For both walls, slabs, windows and doors, property sets are assigned directly to each object.

Searching via correct assignments, all information of the geometry and composition required for architectural objects therefore seem to be available in BIMs currently defined in an IFC file.

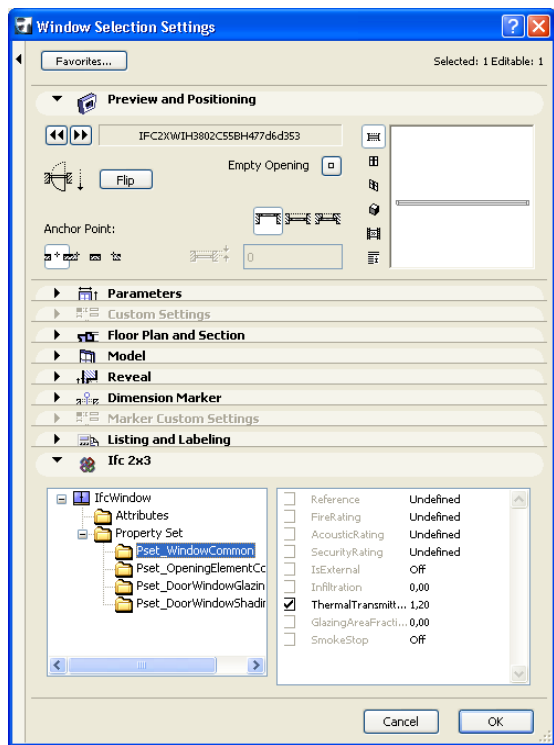


Figure 4-3. Possibilities in ArchiCAD to define information to be exported to an IFC file.

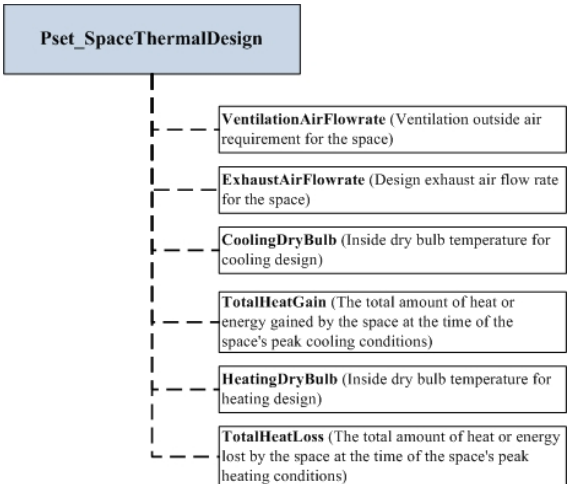


Figure 4-4. Property set and current information being exported from Riiska.

As described in section 3.2.1.2 *Input/Output Data Analysis of Riiska*, Riiska is currently capable of exporting key information on simulation results back to an IFC file. The information which is exported

from Riuska is illustrated in Figure 4-4. All values are defined in Pset_SpaceThermalDesign¹¹ which is assigned to each space object. Riuska additionally groups all spaces which are supplied by the same ventilation system into zones and this information is also exported back to the IFC file using IfcZone.

The identified possibilities for assigning information to a BIM defined in the IFC format is therefore limited to information relating architectural objects such as spaces, walls, windows, doors and slabs. Some thermal and performance related information can be defined for these objects, and the selected functionality of the interface will therefore focus on these possibilities.

4.3.2 Included Functionality

The current possibilities to retrieve required information to Be06 are limited to information about the building envelope and limited performance data for each space. As most thermally related information must be assigned manually to the objects in the BIM applications, it has been decided only to describe how some of the information can be retrieved from the IFC file. This has led to the decision to include the following functions in the IFC interface for Be06:

- Import of gross area of the building.
- Import of ventilation rates for spaces, preferably grouped into zones
- Import of areas, U-values, and b-factors (conditions on other side) of walls and slabs/roofs in the building envelope.
- Import of areas, orientations, U-values, glazing fraction and perimeters of windows and doors in the building envelope.
- Import of the perimeter of the foundation.

The required information by Be06 on areas of spaces, walls and slabs does not relate directly to the actual size of the objects. Instead it relates to the area for which it is believed that heat transmission occur. The measurement standard required by Be06 is described in Appendix 16. Although the aim of the development was to be able to handle most of such cases, it has been found to require too many complex geometrical analyses of the building design to derive the correct values for all cases. As the development primarily aims at exploring the capabilities of IFC, only limited support for the exact measurement standard required by Be06 will be included in the interface. The current capabilities for each type of object will be described separately in the following sections along with a description of the methods used to derive the required information.

¹¹ There is a software error in Riuska, which means that the property set is named "Pset_SpaceHvacInformation". This was the naming convention in IFC version 2x2, however, it changed in IFC 2x3 and the issue should be solved by Riuska. The tool created here, will support both naming conventions.

4.3.3 Development of Interface

Be06 does not allow for corrections to be made to the tool itself, however, Be06 allow for all required information to be imported via an XML file defined using a specifically defined structure. As Rambøll – the place where part of the development took place – and some other consultants already have small interfaces which can convert an Excel spread sheet to the correct XML file for Be06, it was decided to create the IFC interface so that it imports the information to Excel. Unfortunately an interface to convert an Excel spread sheet to the correct XML file for Be06 is not publicly available. However, the IFC import to Excel also allows for other spread sheet calculations to be made from the imported data and this is found useful to widen the potentials of the created IFC interface. Such additional spread sheet calculations could be heat loss calculations used by many consultants to estimate required capacities of the heating system. The programming language used to create scripts in Excel is called Visual Basic for Applications (VBA) and this will be the language used for the creation of the IFC interface. The entire source code created for the interface is available in Appendix 15. In the description of the methods used, all names in *Italic* will refer to functions defined in Appendix 15.

The starting point of the interface is illustrated in Figure 4-5 and makes it possible to identify the IFC file which holds the information required. It further requires a value for the level of the ground as this value is needed to identification the placement of the building. For now it is therefore assumed that the terrain is a plain surface and that the ground level is therefore identical along the entire perimeter of the building. The interface is activated by clicking “Run”.

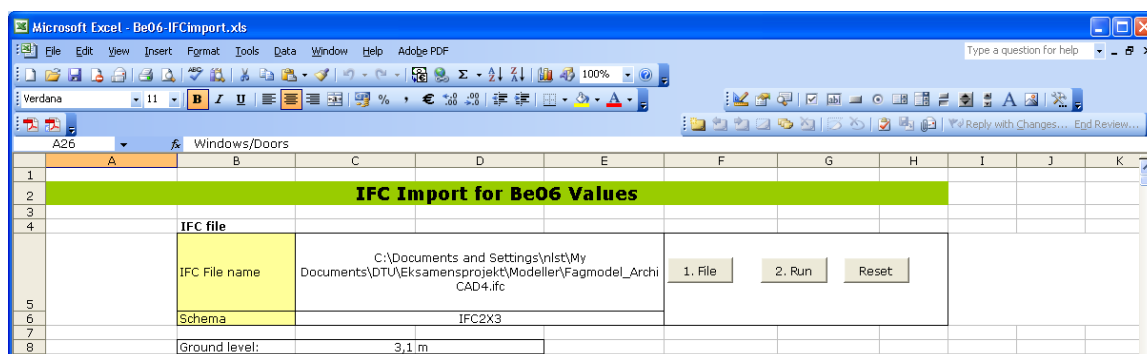


Figure 4-5. Starting point for IFC interface for Be06.

4.3.3.1 Transposing Local Coordinates to Global Coordinates

Before the required information of objects can be retrieved by the IFC interface, it is required to create a function which is capable of handling the placement of the objects so that the interface can identify their exact location in the BIM. As described in section 3.2.2.2 *Geometrical Representations and Placement in IFC*, all objects are defined in a local coordinate system using *IfcLocalPlacement*, which consists of a hierarchical structure of local placements. If derived, this structure can be used to identify the actual (global) coordinates defined for an object via the hierarchical structure. In order to identify the placement of objects in relation to each other, it is required to derive the global coordinates as coordinates in different local coordinate systems cannot be compared. This way the placement of objects can be identified in the same global coordinate system and values can hereby

be compared. In the current interface, this is required to derive the location of walls and slabs among others.

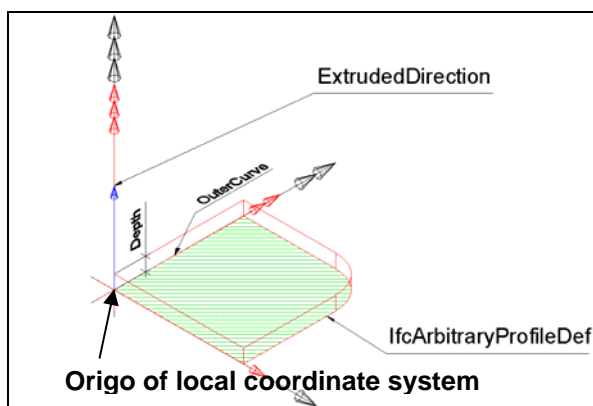


Figure 4-6. Geometrical definition of a slab using a Swept Solid representation. The local coordinate system for the objects is indicated with its axes [IAI 2004].

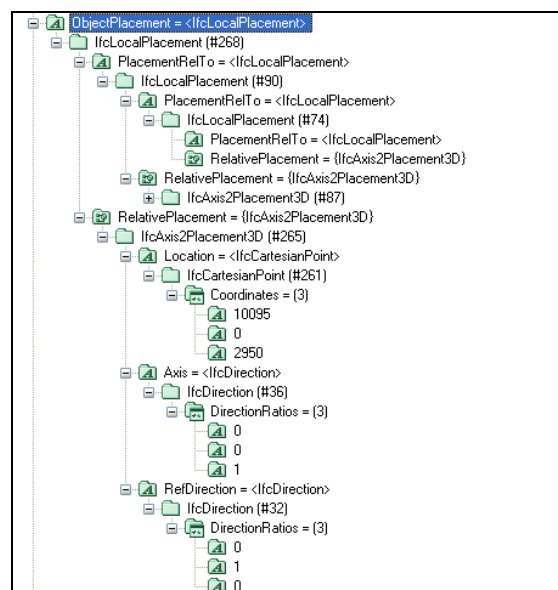


Figure 4-7. The definition of the placement for an object with the hierarchical structure of IfcLocalPlacement.

Each local coordinate system is defined by an IfcAxis2Placement3D which contains information of the relative placement (Location), the direction of the Z-axis (Axis) and the direction of the X-axis (RefDirection). The location of any coordinates in a geometry representation will refer to this coordinate system as illustrated in Figure 4-6. From the object placement defined by IfcLocalPlacement as illustrated in Figure 4-7, the local coordinate system can be transposed to a relative placement (PlacementRelTo) by the IfcAxis2Placement3D defined for the relating local coordinate system.

In order to transpose the coordinates in a geometry representation to the global coordinate system, it is therefore required to transpose each point by all the relative placements in the hierarchical structure of IfcLocalPlacement one relative placement at the time, as it was previously illustrated in Figure 3-15. To do so in the IFC interface, the function *GetLocal2Global* was created. This function is based on a script created by SECOM Co., Ltd¹², however, during the development it was found that the original script was not able to transpose individual point in the direction of the Z-axis, and the script therefore need to be corrected.

The current functionality of the function allows for one or more coordinates to be transposed by all IfcAxis2Placement3D identified to relate to the current coordinates. For each transposition the function firstly rotate the coordinates to comply with the new orientation of the Z-axis. Secondly the

¹² The script was part of a project created to convert IFC objects to surfaces in a game engine. The project was named "IFCsvr MAP for QuakeII Converter" and the source code is available together with the IFCsvr toolbox at <http://groups.yahoo.com/group/ifcsvr-users/>.

coordinates are rotated to comply with the new orientation of the X-axis and finally the coordinates are moved according to the new location. An example of the functionality is illustrated in Figure 4-8 where a point is first rotated according to the directions to the Z-axis (Axis) and then rotated according to the directions to the X-axis (RefDirection). Finally the point is moved according to the placement (Location) of the new local coordinate system. A description of an `IfcAxis2Placement3D` that would cause this transposition is also illustrated in the figure.

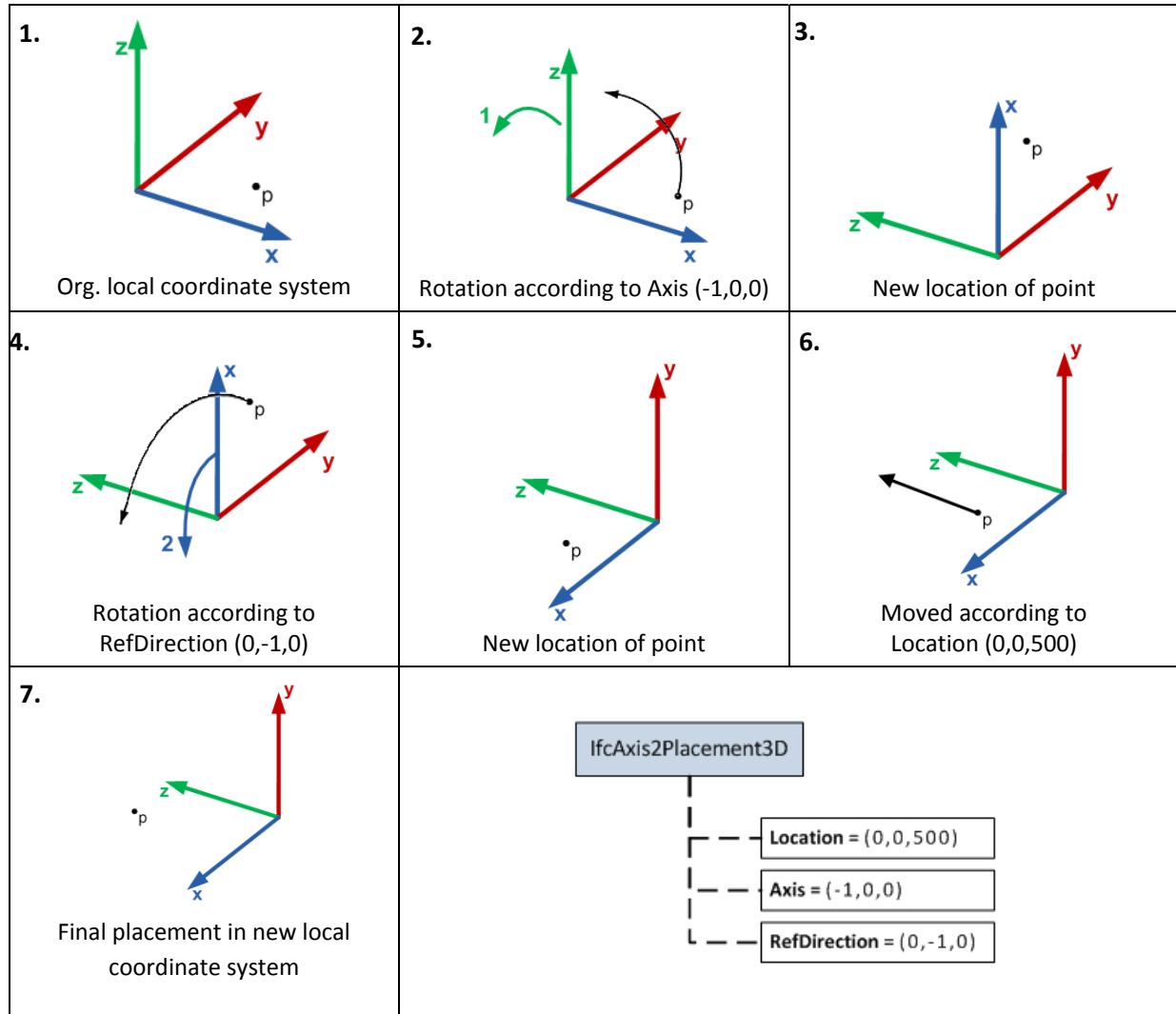


Figure 4-8. Example of transposition caused by the `IfcAxis2Placement3D` illustrated in the lower right corner. It requires three steps for such a transposition.

As the rotation of points occurs around the one axis which is fixed in each step of the process, this concept will only allow for transpositions as long as the directions of the axis in `IfcAxis2Placement3D` are perpendicular or parallel to existing axes. If e.g. the coordinate system was turned 45° this function will not work. This limits the IFC interface to be able to only support buildings with right angles. It would require the use of more complex matrix transformation as described in [Mentzel 2003] to support more transpositions; however, it is believed to be out of the scope this thesis to develop such solutions.

Using the function *GetLocal2Global*, the IFC interface is then capable of transposing points in a right angled building to the global coordinate system and hereby allow for comparison between placements of objects.

4.3.3.2 Retrieving Information on Spaces

Be06 requires information of the heated gross area of the building and this could be derived by either finding the outer perimeter of the building in a plane or by summarising the gross area of all heated spaces. It would be complicated to define the correct area using the building perimeter as the perimeter can vary by each storey and because this method does not account for potential unheated areas in the building. Summarising the areas of all spaces would allow for a possibility to account for variation in the building design and a possibility to exclude unheated spaces. The gross area of each space can currently be derived in two ways: Either by calculating the area based on the geometry representations of spaces and walls available in the BIM, or the area value could be taken directly from the property set Pset_SpaceCommon as described in section 3.2.3.2 *Input Data for Spaces*. For the values to be taken directly from Pset_SpaceCommon, it will require that the values have previously defined here by one of the BIM applications.

AutoCAD Architecture currently has capabilities of doing so. The method of how to assign information of gross and net area to spaces is described in Appendix 17 as it requires special actions to insure that the values are exported along to the IFC file. Revit and ArchiCAD also have capabilities to handle gross areas of spaces. However, at least for ArchiCAD it would require a significant amount of manual setup to ensure that these values are exported to the IFC file. For this reason the interface includes two solutions. The first solution will estimate the gross area based on the net area of each space and the second solution will use the defined value for gross area – if this exists in Pset_SpaceCommon – for each space. As information of heating set point is not expected to be defined in the BIM at present, all spaces are assumed to be heated and the gross area of all spaces will therefore be included in the summation.

It is the function *GetSpaces* that retrieve information of spaces from the IFC file. It does so by analysing each space defined in the model one at the time and then prints the values for each space to a sheet in the Excel workbook as illustrated in Figure 4-9. The spaces are sorted in groups which constitutes the zones that Riuska or another tool might have defined.

Zone name:	Gross area [m2]:	Net area [m2]:	Net volume [m3]:	Max air flow rate [l/s/m2]:	Min air flow rate [l/s/m2]:	Zone name:
Meeting rooms - VAV	1.175,78	1.085,13	3.852,20	4,61	1,35	Toilets/Kitchens - C
Space name	Gross area [m2]:	Net area [m2]:	Net Volume [m3]:	Max air flow rate [l/s]:	Min air flow rate [l/s]:	Space name
SPACE Meeting rooms(1) - Space (20)	28,47	25,74	91,37	129	39	SPACE Kitchen(1) - S
SPACE Meeting rooms(2) - Space (36)	28,47	25,74	91,37	129	39	SPACE Kitchen(2) - S
SPACE Meeting rooms(3) - Space (37)	79,04	73,92	262,43	369	111	SPACE Toilets(1) - S
SPACE Meeting rooms(4) - Space (58)	58,91	55,08	195,55	275	83	SPACE Toilets(2) - S
SPACE Meeting rooms(5) - Space (8)	39,76	36,47	129,47	182	55	SPACE Toilets(3) - S
SPACE Meeting rooms(6) - Space (29)	28,47	25,82	91,67	129	39	SPACE Toilets(4) - S
SPACE Meeting rooms(7) - Space (32)	28,47	25,82	91,67	129	39	SPACE Toilets(5) - S
SPACE Meeting rooms(1) - Space (9)	28,47	25,84	91,73	129	39	SPACE Toilets(6) - S
SPACE Meeting rooms(2) - Space (11)	28,47	25,95	92,13	129	39	SPACE Toilets(7) - S
SPACE Meeting rooms(3) - Space (21)	28,48	25,82	91,67	129	39	SPACE Toilets(8) - S
SPACE Meeting rooms(4) - Space (24)	28,47	25,82	91,67	129	39	SPACE Toilets(9) - S
SPACE Meeting rooms(5) - 103	39,76	36,61	129,98	182	55	SPACE Kitchen(1) - S
SPACE Meeting rooms(6) - 107	39,27	36,72	130,36	182	55	SPACE Kitchen(2) - S
SPACE Meeting rooms(7) - 111	39,27	36,61	129,95	182	55	SPACE Toilets(1) - S
SPACE Meeting rooms(8) - 119	79,54	73,85	262,16	369	111	SPACE Toilets(2) - S
SPACE Meeting rooms(9) - Space (55)	28,47	25,82	91,67	129	39	SPACE Toilets(3) - S
SPACE Meeting - Space (56)	39,28	36,78	130,56	184	55	SPACE Toilets(4) - S
SPACE Meeting - Space (57)	39,30	36,64	130,08	183	55	SPACE Toilets(5) - S
SPACE Meeting rooms(1) - Space (7)	39,76	36,47	129,47	182	55	SPACE Toilets(6) - S
SPACE Meeting rooms(2) - Space (8)	39,27	36,46	129,45	182	55	SPACE Toilets(7) - S
SPACE Meeting rooms(3) - Space (14)	28,47	25,93	92,05	130	4	SPACE Toilets(8) - S
SPACE Meeting rooms(4) - Space (15)	28,47	25,82	91,67	129	39	SPACE Toilets(9) - S
SPACE Meeting rooms(5) - Space (17)	28,47	25,84	91,73	129	39	SPACE Kitchen(1) - S
SPACE Meeting rooms(6) - Space (30)	28,47	25,95	92,13	129	39	SPACE Kitchen(2) - S
SPACE Meeting rooms(7) - Space (33)	28,03	25,81	91,61	129	39	SPACE Toilets(1) - S
SPACE Meeting rooms(8) - Space (34)	39,29	36,53	129,67	183	55	SPACE Toilets(2) - S
SPACE Meeting rooms(9) - Space (43)	79,03	73,84	262,12	369	111	SPACE Toilets(3) - S
SPACE Meeting - Space (46)	28,47	25,82	91,67	129	39	SPACE Toilets(4) - S
SPACE Meeting - Space (56)	39,28	36,37	129,10	182	55	SPACE Toilets(5) - S
SPACE Meeting - Space (57)	59,40	55,22	196,03	275	83	SPACE Toilets(6) - S

Figure 4-9. Imported information about spaces in the IFC interface.

The *GetSpaces* function starts by searching through all instances of *IfcRelAssignsToGroup* to see if the current space is assigned to any zones referring to the ventilation system. If a zone is identified, the interface will group the space with other spaces relating to the same zone.

The function *GetSpaces* then calculates the net area of each space to be able to estimate the gross area. It does this by browsing through the hierarchical structure of properties assigned to each space and from here it identifies the geometrical representation defined as a “SweptSolid”. Such a representation is illustrated in Figure 4-10. From this representation the IFC interface retrieves information of the swept area defining the bottom of each space. The area is either defined by its width and length or by an arbitrary profile which is a polyline defined by points. In case a space has one or more columns or other spaces enclosed within the space, the swept area will include a description of inner curves which define the shape of any voids such as the case in Figure 4-11.

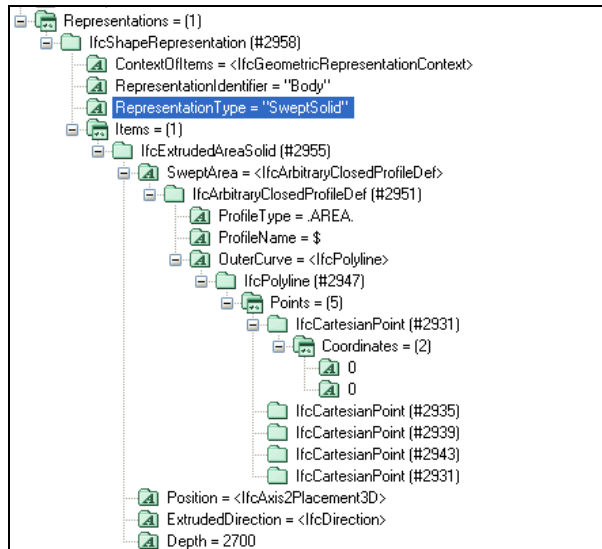


Figure 4-10. Information defined for a SweptSolid representation for a space. This representation does not include voids.

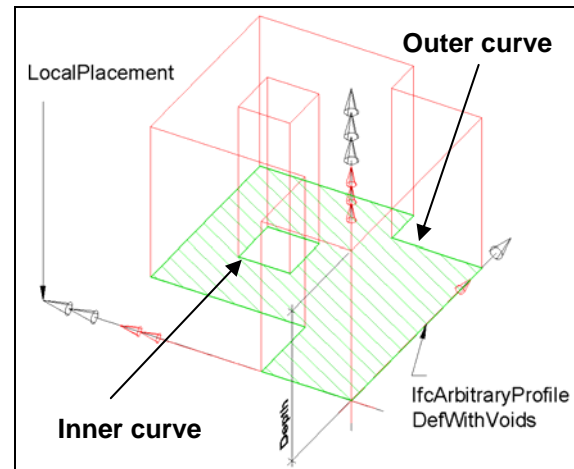


Figure 4-11. The swept area of a space which includes an inner curve to define the void [IAI 2004].

As the swept area represents the net area of the space, this area can now be calculated. In case the area is defined by an arbitrary profile, the area is calculated from the points of the curve based on Equation 4-1. In case the swept area includes any voids, their area will be subtracted from the net area of the space.

$$A = \frac{1}{2} \left(\begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix} + \begin{vmatrix} x_2 & x_3 \\ y_2 & y_3 \end{vmatrix} + \dots + \begin{vmatrix} x_n & x_1 \\ y_n & y_1 \end{vmatrix} \right)$$

Equation 4-1. Area, A, of a non-self-intersecting polyline [Weisstein 2004].
(x_1, y_1), (x_2, y_2), ..., (x_n, y_n) constitutes points on the polyline and each matrix denotes a determinant.

To find the gross area of each space it would require that the net area is added with the plan area of any external wall boundaries and half the plan area of any internal wall boundaries. Although the space boundaries can be identified via *IfcRelSpaceBoundary*, it has been found to be a complex task to identify the correct area to add. Each space can have several different walls which are bounding the same side of the space (e.g. on top of each other), and it will therefore require a thorough analysis to identify which walls are bounding the space at different locations along its perimeter. As such an analysis can be conducted by AutoCAD Architecture automatically, the IFC interface will simply add 10 % to the net area to estimate the gross area if information is not available to limit the scope of the development. The correct value to add to the net area varies depends on the building design but 10 % is found to represent a typical office building such as the building described in Chapter 5. The MiniOffice previously described has a 12 % difference between net and gross area as a reference. In case the gross area is estimated it will be marked with a comment in the Excel sheet.

Searching through all instances of *IfcRelDefinesByProperties*, the *GetSpaces* function will then try to find if any *Pset_SpaceCommon* property set is assigned to the space in question. If it finds such a

property set, it will try to retrieve values for the gross and net area. In case they exist, these values will then replace the calculated values.

From the *SweptSolid* representation the height (Depth in Figure 4-10) of the space can further be retrieved, and for information purposes the volume of each space is therefore calculated based on the net areas and included in the imported information.

Finally the *GetSpaces* function searches through all instances of *IfcRelDefinesByProperties* to see if the *Pset_SpaceThermalDesign* property set is assigned to the space. It first looks for the attributes “CoolingDesignAirflow” and “HeatingDesignAirflow” which would give the maximum and minimum air flow rate directly. If it does not find these values, it will look for the attribute “VentilationAirFlowrate” which Riuska uses to define the maximum air flow rate for the space. In case it finds a value for “VentilationAirFlowrate”, the function will calculate the minimum air flow rate as 30 % of the maximum, which is believed to be a common value for the difference between min and max.

The function could look for other required information such as set points or occupancy type, and this should be done similar to the search for air flow rates.

4.3.3.3 Retrieving Information on Walls

The measurement standard which applies for wall areas in Be06 is defined in Appendix 16. For walls this means that the height of the wall depends on the conditions of the space on the inside of the wall when deciding on how much of the external wall to include. Even though unheated spaces are ignored, it would still require a complex geometrical analysis of the relation between spaces, roof constructions and the wall geometry to identify the precise height to include. For this IFC interface, it has been decided to include only the actual height of each wall as this typically is close to the required height. The width of the wall is defined as the distance between corners on the external side of the façade. The width of the wall can be derived from the length of the 2D axis (curve) which represents each wall. The placement of the axis within the wall can, however, vary depending on the modelling procedure used, and it will therefore only be in some cases that the axis represents the external length of the wall. As both AutoCAD Architecture and ArchiCAD includes possibilities to define that the axis should follow the outer perimeter of the walls, the IFC interface will for now simply use the length of the 2D axis. The placement of the 2D axis within the wall is defined in the material layer set properties assigned to the wall and the exact placement can therefore be derived in future extensions of the interface.

It is the function *GetWalls* which retrieve information required for the walls. Before it retrieves information of the area this function will search for a *Pset_WallCommon* property set assigned to the current wall to identify if the wall is internal or external. This information is defined in the attribute “IsExternal” which is either true or false, if it exists. The information can be set by ArchiCAD either automatically by layer names or manually for each object. The information is used to identify the external walls to be analysed. Unless a function – described later – identifies that the wall is not bounding any spaces and therefore placed outside the building envelope, it will be assumed that

heat is transmitted through the entire surface of the wall. From the Pset_WallCommon the interface additionally tries to retrieve information on the U-value of the wall if this value is present in the attribute "ThermalTransmittance".

The function then calls another function, *GetWallDirAndPlace*, which returns the orientation of the wall and the level of the bottom of the wall. The orientation functionality of the function will be described later as it does not directly relate to the walls. To find the level of the bottom of the wall the function simply selects the first point of the 2D axis of the wall and transposes this point into global coordinates using the *GetLocal2Global* function previously mentioned. The Z-value of the point is then derived as the location of the bottom of the wall. The value is used to identify the placement of the wall in relation to the ground level defined at the starting point of the interface. This is needed to derive the correct b-factor of the wall. Parts of the wall placed more than 2 meters under ground will have a b-factor of 0.7 where as the rest of the wall will have a b-factor of 1.0¹³. In case parts of a wall are identified to have different b-factors the interface will split the wall in two and assign correct b-factors to each part. The function *GetWallDirAndPlace* additionally keeps track of the wall with the lowest bottom elevation as this is required later when estimating the outer perimeter for the foundation.

The function *GetWalls* will then try to group similar wall types. The grouping is based on the naming of the material composition of the wall defined in *IfcMaterialLayerSet*. The way to assign the material layer set to the wall was described in section 3.2.3.3 *Input Data for Walls and Slabs* and the *GetWalls* function therefore looks for any material layer sets assigned to the wall using *IfcRelAssociatesMaterial*. It is assumed that walls with the same material layer set have the same U-value and walls will therefore be grouped according to these names.

Microsoft Excel - Be06-IFCimport.xls

File Edit View Insert Format Tools Data Window Help Adobe PDF

Type a question for help

Verdana 9 B I U

M57

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

Walls

Wall type: Total area: [m2] U - Value [W/m2K]: b-factor:

Beton-X_1 39,71 0,30 1,0

Wall name Comment Width Height Area

WALL | Beton-X(4) Orientation: South 20,90 1,90 39,71

Wall type: Total area: [m2] U - Value [W/m2K]: b-factor:

Mursten-108 Isolering-187 Bet_1 398,34 0,26 1,0

Wall name Comment Width Height Area

WALL | Mursten-108 Isolering-187 B Orientation: West 15,89 3,00 47,67

WINDOW | Panorama(1) -6,06

WALL | Mursten-108 Isolering-187 B Orientation: North 20,89 3,00 62,67

DOOR | Dobbeltddr(1) -3,02

WINDOW | Panorama(2) -3,64

WALL | Mursten-108 Isolering-187 B Orientation: East 15,89 3,00 47,67

WINDOW | Panorama(3) -2,42

WINDOW | Panorama(4) -2,42

WALL | Mursten-108 Isolering-187 B Orientation: South 20,89 3,00 62,67

WINDOW | Panorama(5) -3,64

WALL | Mursten-108 Isolering-187 B Orientation: West 15,89 3,00 47,67

WINDOW | Panorama(1) -6,06

WALL | Mursten-108 Isolering-187 B Orientation: North 20,89 3,00 62,67

WINDOW | Panorama(2) -3,64

WINDOW | Panorama(3) -3,64

WALL | Mursten-108 Isolering-187 B Orientation: East 15,89 3,00 47,67

WINDOW | Panorama(4) -2,42

WINDOW | Panorama(5) -2,42

WALL | Mursten-108 Isolering-187 B Orientation: South 20,89 3,00 62,67

WINDOW | Panorama(6) -3,64

Summation Spaces Walls WinDoors Slabs

Ready

NUM

Figure 4-12. Imported information about walls in the IFC interface.

¹³ The outside design temperature must also be corrected in Be06 in case the wall is more than 2 meters under ground, however, this has been excluded in the interface.

The function then retrieves information of the wall height from its “SweptSolid” representation similar to the function for the spaces. The width of the wall is derived from the length of the 2D axis also representing each wall. The points of the axis are defined as it was illustrated in Figure 4-1. From the height and width, the function calculates the area of each wall. All values are printed to a sheet in the Excel workbook as illustrated in Figure 4-12.

As the area of the walls is reduced if any windows and doors are inserted, the function *GetWalls* will then call another function called *findRelatedWinDoors*. This function will be described later, however, it returns information on the area of any windows or doors inserted into the current wall and this information is therefore subtracted from the area of the wall.

Although the areas of the walls are not fully compliant with the requirements of Be06, the interface has managed to retrieve most information required for walls.

4.3.3.4 Retrieving Information of Orientation

Information of the orientation of windows and doors are required by Be06. As the walls include a 2D axis for their representation, it has been found that the orientation is easier to derive from here as the windows and doors are only defined by BREP representations consisting of surfaces. As it is only windows and doors inserted in the walls that are included, this information of orientation should be sufficient.

The information of True North as described in section 3.2.3.1 *Input Data for Projects, Sites, Buildings and Stories* is available in the IFC files from the BIM applications, however, it is not possible to adjust the value in the BIM applications and the IFC interface therefore simply assumes that the direction of the Y-axis in the global coordinate system (0,1,0) represents the direction of north.

It is the function *GetWallDirAndPlace* previously mentioned, which derives the orientation. To do so the function starts by finding the angle between the wall axis and the direction of north using Equation 4-2. To ensure that the direction of the wall can be compared, the function *GetLocal2Global* is used to transpose the coordinates of the wall axis to the global coordinate system before the angle is calculated.

$$\det(\vec{a}, \vec{b}) = \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix} = |\vec{a}| |\vec{b}| \sin(v)$$

Equation 4-2. Determinant of the vectors a (x₁,y₁,z₁) and b(x₂,y₂,z₂) where v is the angle from a to b.

Knowing this information, it will then require information about which side of the wall axis is turning to the outside to derive the orientation of the outer wall surface. This information is not currently available in the IFC files from the BIM applications, however, ArchiCAD export information of the space boundaries. Unfortunately there is no attributes which defines at which side the space boundary applies and this must therefore be derived. It has, however, been identified that each *IfcRelSpaceBoundary* defined by ArchiCAD includes a surface geometry which represents the area of the boundary between e.g. a space and a wall. The surface geometry is defined by a polyline and this

line is characterised by always being defined in the same direction making the normal vector of the surface pointing away from the space. The information available on the surface is illustrated in Figure 4-13.

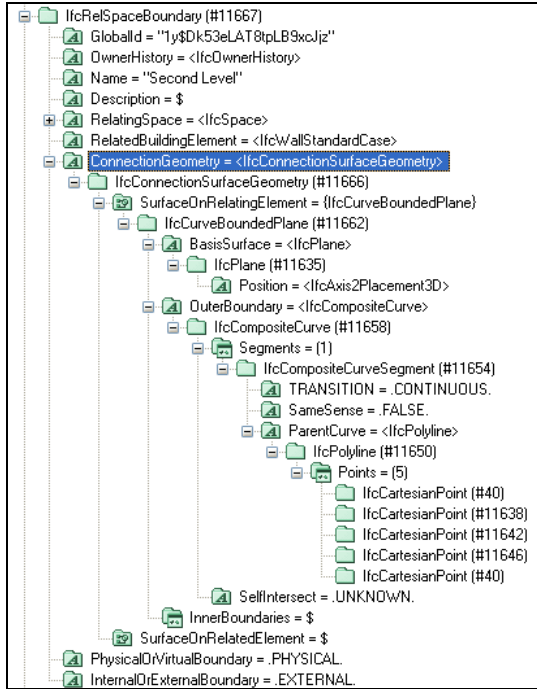


Figure 4-13. Information available on the surface geometry of a space boundary in IFC.

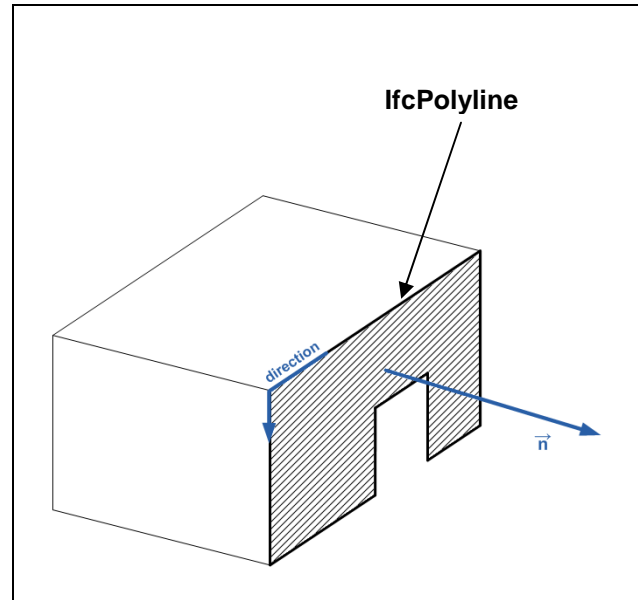


Figure 4-14. Normal of the surface of a space boundary.

The coordinates of a normal for a surface defined by two vectors can be found using Equation 4-3.

$$\vec{n} = \vec{a} \times \vec{b} = \begin{pmatrix} y_1 & y_2 \\ z_1 & z_2 \\ x_1 & x_2 \\ y_1 & y_2 \end{pmatrix}$$

Equation 4-3. Coordinates of a normal, n , defined as the cross product of two vectors $a(x_1, y_1, z_1)$ and $b(x_2, y_2, z_2)$ [Uddannelsesstyrelsen 1999].

As the polyline representing the surface can be concave in some cases as illustrated in Figure 4-14, it is required to find the normal representing the entire surface and not just the normal represented by two point of the polyline. To do so, the *GetWallDirAndPlace* function calculates the normal of the surface using Equation 4-4.

$$\vec{n}_{surface} = \begin{pmatrix} y_1 & y_2 \\ z_1 & z_2 \\ z_1 & z_2 \\ x_1 & x_2 \\ y_1 & y_2 \end{pmatrix} + \begin{pmatrix} y_2 & y_3 \\ z_2 & z_3 \\ z_2 & z_3 \\ x_2 & x_3 \\ y_2 & y_3 \end{pmatrix} + \dots + \begin{pmatrix} y_n & y_1 \\ z_n & z_1 \\ z_n & z_1 \\ x_n & x_1 \\ y_n & y_1 \end{pmatrix}$$

Equation 4-4. Normal representing the entire surface define by a polyline with coordinates $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$.

Using Equation 4-2 the *GetWallDirAndPlace* function then finds the angle between the normal of the space and the wall axis. As the surface of the boundary is defined according to the local coordinate system of the space, the points of the surface are transposed to the global coordinate system before the normal is calculated so that the normal and wall axis can be compared. The angle is derived so that it will be defined in the interval -180° to $+180^\circ$ and based on this information, it can be determined on which side of the wall the space is placed, as illustrated in Figure 4-15.

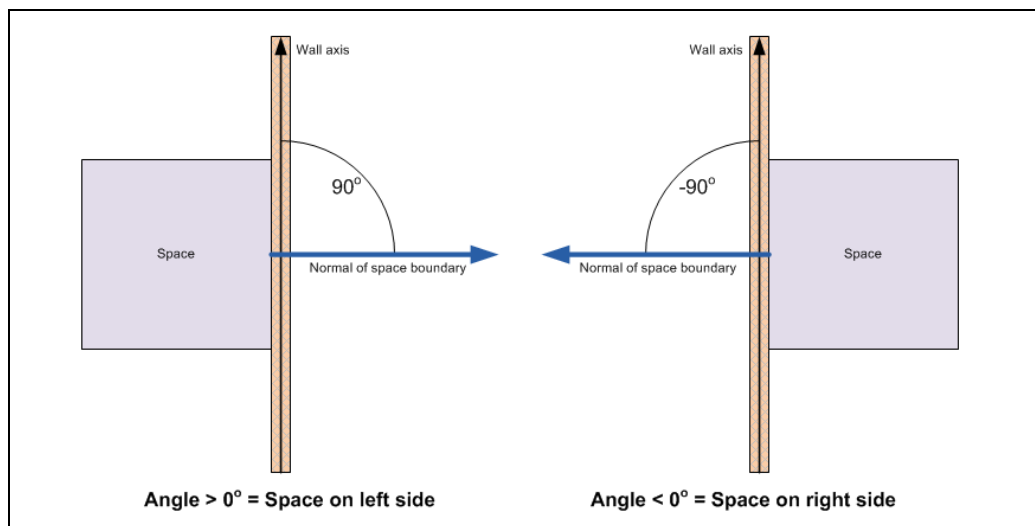


Figure 4-15. Determination of the placement of spaces along the wall axis.

Combining information of which side the spaces are placed with the angle between the wall axis and the direction of north, the function can then determine the orientation of the outside surface of the wall. If the wall axis is e.g. pointing in the direction of either north or south, the outside of the wall will be facing either east or west. The limitations in the interface only allow for right angles in the building and the function *GetWallDirAndPlace* therefore only categorise the orientations in north, east, south and west. This could easily be extended to include the orientations north east, south east etc. which can also be used in Be06.

The information of the placement of spaces is additionally used to sort out any walls which have not been defined to bound any spaces and which are therefore assumed to be placed outside of the building envelope.

4.3.3.5 Retrieving Information on Windows and Doors

The information of windows and doors is retrieved by the function *findRelatedWinDoors*. As previously described it is used to find information on the windows and doors placed in the walls in the building envelope. From the wall information the function already knows the orientation of the window or door as described above and this information will therefore be used directly.

The relations between windows, doors and walls are described in section 3.2.3.3 *Input Data for Walls and Slabs* which indicates that the relation is assigned via an opening object. The function *findRelatedWinDoors* therefore starts by searching through instances of *IfcRelVoidsElement* to find openings in the wall. For each opening, the function then searches through instances of *IfcRelFillsElement* to find any windows or doors which fills the opening.

For each window or door identified the function then looks for any *Pset_WindowCommon* or *Pset_DoorCommon* assigned to the object and tries to retrieve the U-value from the attribute "ThermalTransmittance". As windows and doors always have a type definition assigned using *IfcRelDefinesByType*, the function then retrieves the name of the window or door type. This information is used to group similar objects and it is thereby assumed the similar objects have the same U-value.

From the attributes of the object, the width and height can be derived directly in "OverallWidth" and "OverallHeight" and this information is used by *GetWallDirAndPlace* to calculate the area and perimeter of each window or door. As described in section 3.2.3.4 *Input Data for Windows and Doors*, *IfcWindowLiningProperties* includes information of the frame width of the windows in the attribute "LiningThickness" as illustrated in Figure 4-16. For windows, the function therefore finds the lining properties via the assigned type and calculates the area of the frame based on the width and height of the window. Using information of the overall area and the frame area, the glazing fraction is calculated. This is based on an assumption that the windows imported only include a frame along the perimeter. It will require further analyses to derive other compositions of windows. However, this is believed to be outside the scope of this development. All identified information is then printed to a sheet in the Excel workbook as illustrated in Figure 4-17.

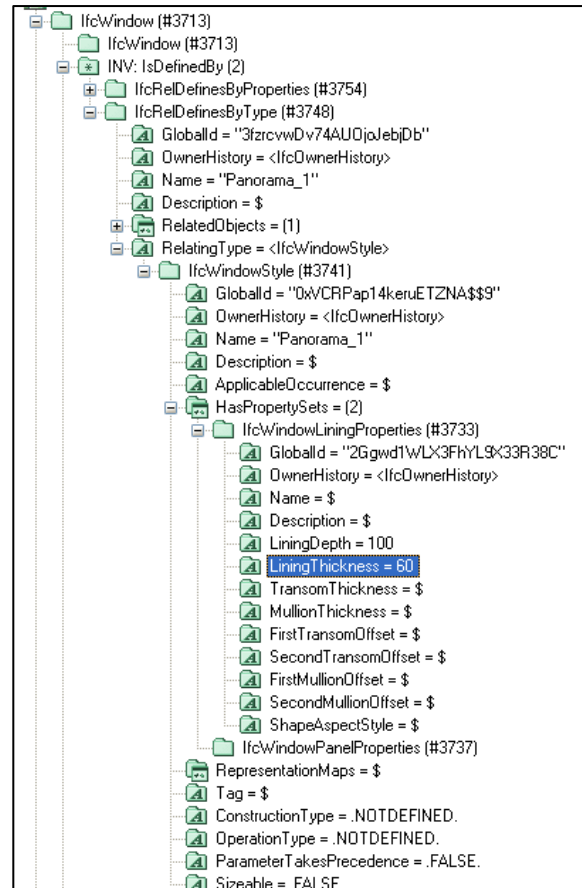


Figure 4-16. Definition of lining (frame) thickness of a window in an IFC file.

Name	Width [m]	Height [m]	Perimeter [m]	Area [m2]	U-Value [W/m2K]	Glazing fraction [%]	Total perimeter [m]
WINDOW -205.02_1(1)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_1(2)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_1(3)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_1(4)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_1(5)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(1)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(2)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(3)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(4)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(5)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(6)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(7)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(8)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(9)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(10)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(11)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(12)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(13)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(14)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(15)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(16)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(17)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(18)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(19)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(20)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(21)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84
WINDOW -205.02_2(22)	1,10	2,19	6,58	2,41	1,20	0,87	1,302,84

Figure 4-17. Imported information about windows and doors in the IFC interface.

Be06 needs further information than what have been imported by the IFC interface. The g-value required for glazing could be retrieved similar to the U-value if available in the BIM. As described in section 3.2.3.4 *Input Data for Windows and Doors*, information about shading devices is more complex to derive as this would require an extensive geometrical analysis to identify potential shading objects. The information should be available in the BIM, and the information could therefore potentially be derived.

4.3.3.6 Retrieving Information on Slabs/Roofs

Be06 requires similar information on slabs as it does for walls in the building envelope. As described in section 3.2.3.3 *Input Data for Walls and Slabs*, the IFC specifications does not allow for information on e.g. the U-value of a roof to be assigned to a roof object. In case it is the roof itself which includes the insulation, it must then be modelled as a slab. The IFC interface will therefore only retrieve information on slabs which are selected as external using the “IsExternal” attribute in Pset_SapceCommon. This attribute is expected to be assigned to any external slab objects. This way to identify the external roof slab is in accordance with the measurement standard for Be06 described in Appendix 16. Here it is defined that the area of the roof slab is the area of the insulated part of a roof construction.

It is the function *GetSlabs* which retrieves information of the slabs one by one. As described, the function will start by searching for any Pset_SapceCommon assigned to the objects, and from here retrieve information of the slab being either internal or external. The slab will only be included if it is identified to be external. From Pset_SapceCommon, the function also tries to retrieve information on the U-value of the slab from the attribute “ThermalTransmittance”.

The function then calls another function, *GetSlabPlaceAreaPerimeter*, which returns the placement, area and perimeter of the slab. As described in Appendix 16 the area information required by Be06

varies depending on the location and the conditions in the spaces. If the outside of the slab is below ground and the slab bounds a heated space, it is the net area which is required. If the outside surface is above ground, the area relates to the outside part of the slab expected to contribute to the heat transmittance. Similar to the area of walls, it would require complex geometrical analyses to identify the correct area for all type of building designs, and for this interface it is therefore decided only to retrieve the net area of the slab. This will be the correct value for slabs below ground and more or less accurate for slabs above ground, depending on the modelling technique and the shape of e.g. the roof. The net area of the slab is calculated in the same way as the area of a space using Equation 4-1 since plain slabs are also defined by a SweptSolid representation.

The *GetSlabPlaceAreaPerimeter* function then evaluates whether the bottom of the slab is below or above the ground level by taking the first point of the swept area and transposing its coordinates to the global coordinate system. The Z-value is then compared to the defined level for the ground similar to the walls.

Be06 also requires information of the length of the perimeter along the foundation to calculate the linear loss. As the foundation objects are not found to be handled well by the architectural BIM applications and the placement of external walls among each other is complex to derive, this analysis is conducted based on the perimeter of the ground slab. It is the external perimeter of the foundation which is required and the thickness of the external walls must therefore be added to the perimeter of the ground slab. As in previous cases it would require a comprehensive analysis to identify the exact walls along the perimeter, and for the IFC interface it is therefore decided to simply use the thickness of the lowest wall identified in the building. This wall was already selected in the *GetWallDirAndPlace* function. The *GetSlabPlaceAreaPerimeter* function will retrieve the thickness of this wall and use it as an estimate for the thickness along the entire perimeter of the ground slab. The thickness of the wall must be derived by summarising the thicknesses of each material in the material layer set of the wall, and the material layer set is therefore identified via *IfcRelAssociatesMaterial*.

The limitations of the interface only allow right angles in the building and the shape of the slab perimeter will therefore only have right angles in the corners. As the corners can have either a convex or concave shape as illustrated in Figure 4-18, the function will therefore analyse the shape of the corner at the end of each line segment. If a corner is identified to be convex, the function will add two times the wall thickness to the line segment and if a corner is concave the function will subtract one times the wall thickness of the line segment. This way the net perimeter is corrected to represent the gross perimeter required. For now, this functionality only works when the points at the perimeter are defined anticlockwise.

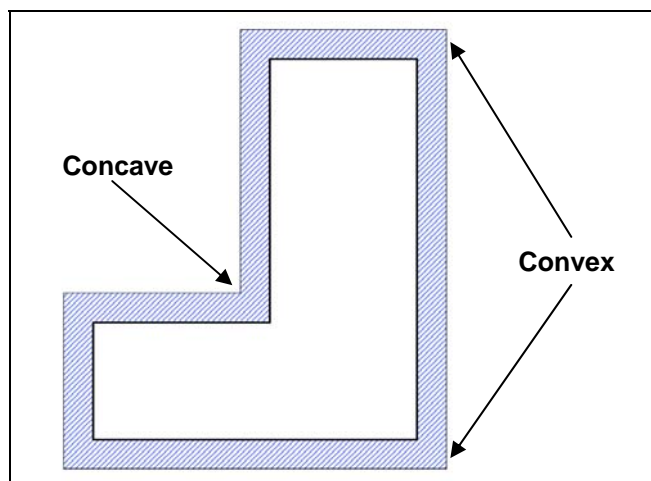


Figure 4-18. Possible shape of the ground slab with external walls along the perimeter.

The function does further not take into account if internal walls penetrate the ground slab and hereby split the ground slab into several slabs. Such functionality could be included although it would require a more comprehensive analysis of the construction. As described in section 3.2.3.3 *Input Data for Walls and Slabs* the thermal value for linear loss cannot be defined in IFC, and the length of the perimeter is therefore the only information which can currently be retrieved for the foundation.

Based on the information on the placement of the slab, the *GetSlabs* function finalises the by defining a corresponding b-factor to each slab object and then prints all information to a sheet in the Excel workbook as illustrated in Figure 4-19.

Slab type:	Total net area [m2]:	U - Value [W/m2K]:	b-factor:	Gross perimeter [m]:	Slab type:	Total net area [m2]:	U - Value [W/m2K]:	b-factor:
Foundation	1.389,02	0,30	0,7	234,27	Flat roof	1.449,49	0,22	1,00
Slab name	Placement	Area [m2]	Perimeter [m]		Slab name	Placement	Area [m2]	Perimeter [m]
FoundationSlab1	BelowGround	1.389,02	234,27		RoofSlab1	AboveGround	1.449,49	

Figure 4-19. Imported information about slabs in the IFC interface.

4.3.4 Summarising Information

All key information which has been imported is summarised in a sheet in the Excel workbook as illustrated in Figure 4-20. The content illustrated here, is what have been imported for the model described in the following chapter.

SUMMATION OF IFC IMPORT:							
Spaces							
Zone name:	Gross area [m2]:	Net area [m2]:	Net volumen [m3]:	Max air flow rate [l/s/m2]:	Min air flow rate [l/s/m2]:		
Small offices - VAV	1,258,37	1,117,07	3965,60	2,93	0,88		
Toilets/Kitchens - CAV	222,85	200,95	713,39	3,56	1,01		
Meeting rooms - VAV	1,175,78	1,085,13	3852,20	4,61	1,35		
Laboratories - VAV	275,48	253,10	898,50	3,01	0,42		
Unknown	198,99	178,92	1047,39	0,00	0,00		
Vent. other - CAV	2,412,12	2,230,30	8331,51	0,92	0,28		
Walls							
Wall type:	Total area [m2]:	U-value [W/mK]:	b-factor:				
-A205_2	1,254,08	0,29	1,00				
-A205_1	310,30	0,23	1,00				
-A205_3_2	625,53	0,19	1,00				
-A205_3_2	19,19	0,19	0,70				
-A205_3_1	37,48	0,26	1,00				
-A205_3_1	1,25	0,26	0,70				
-A205_3	88,99	0,29	1,00				
-A205_3	2,83	0,29	0,70				
Windows/Doors							
Window/Door type:	Number:	Orientation:	Area [m2]:	U-value [W/mK]:	Glazing fraction [%]:	Perimeter [m]:	
205.02_3	37	n	1,75	1,20	0,84	221,26	
205.02_1	190	n	2,41	1,20	0,87	1250,20	
205.02_3	27	s	1,75	1,20	0,84	161,46	
205.02_1	198	s	2,41	1,20	0,87	1302,84	
205.03_1	1	s	4,29	1,10		8,29	
205.03_3	1	n	3,21	1,10		7,23	
205.03_3	1	ø	3,21	1,10		7,23	
Slabs							
Wall type:	Net area [m2]:	U-value [W/mK]:	b-factor:	Gross perimeter [m]:			
Flat roof	1,449,49	0,27	1,00				
Foundation	1,389,02	0,30	0,70	234,27			

Figure 4-20. Summation of information imported by the IFC interface for Be06.

Using a script capable of converting this information to an XML file according to the correct specification would therefore allow for the information to be used directly in Be06.

4.3.5 Suggested Future Development of Interface

Several limitations apply to the IFC interface and future development will be required for the interface to support more types of building design and to be capable of more exact calculations. Several of the limitations relate to the challenges of retrieving correct information according to the measurement standards defined for Be06. One example is the required areas of objects which can span across several other objects as well. The placement among adjacent objects must therefore be derived by geometrical analysis which can be complicated. To overcome some of these challenges a different approach to the calculations was identified during the development. This could be to use the spaces as the basis for the retrieval of all information. The advantage of using spaces is that they

include relations to all boundaries with an exact definition of how the surfaces align. The placement of boundaries can therefore be retrieved from here, and because the external space boundaries constitute the building envelope, all required information should be available via the boundaries. Time constraints did not allow for this possibility to be explored further, as the framework for such a solution would require several new functions to be included.

Several other limitations could be eliminated by developing a proper calculation method for the transposition of coordinates from a local to a global coordinate system. As the programming language Visual Basic for Applications (VBA) does not handle calculations of matrices very well, such solutions has not yet been created for this environment.

4.4 Identified IFC Capabilities

Although limitations apply to the interface, the example of creating an IFC interface for Be06 still demonstrates interesting possibilities. Using the interface, 11 out of the 36 architectural related input data identified for Be06 in Appendix 5 can now be retrieved directly from an IFC file created with BIM applications currently available. More effort in the development of the interface could improve the quality and number of input data further and although full interoperability cannot be achieved due to limitations in the IFC specifications and BIM applications, the interface would allow for a much improved use of Be06 with a stronger link to the current BIM. This should improve the possibilities to ensure a proper basis for decision in IBD.

The use of an IFC toolbox allow for all IFC data to be retrieved from the IFC file in an organised hierarchical structure, and it is thereby possible for any application to access information from an IFC file. The challenge in this respect is to know where to find the correct information because the current IFC implementation in the BIM applications is not consistent. This is possible because the IFC specifications allow for a large range of opportunities to define the same information in different attributes. To be able to develop proper IFC interfaces to applications, the need for MVDs described in section 2.3.4 *Information Delivery Manual and Model View Definitions* is found to be of great importance to successful implementation of IFC in the applications required in IBD. This way it would be clearly identified where information would be defined in the IFC file, and all applications would be able to precisely retrieve the same data every time.

One of the main challenges for the successful use of IFC data in Be06 is the possibilities to derive the correct information on geometry. This issue is, however, not believed to be an issue specifically relating the IFC specifications but a more general challenge of translating 3D object oriented information to the requirements of Be06. The measurement standard used by Be06 is clearly intended for manual measuring of 2D drawings, and as the AEC industry is moving towards an extensive use of 3D in general, such standards should be adjusted to support the new possibilities of a much more precise definition of the building design in 3D.

Chapter 5

Methodology Description

5.1 Introduction

In the previous chapters it has been identified how BIM design could potentially improve IDP by interoperability and information management and control. Further more it has been analysed how the capabilities of a data repository defined using the IFC specifications could support such tasks. The use of BIM design in IDP has been defined as a concept of IBD and to define this concept more accurately, the following sections will use the knowledge gathered to define a methodology of IBD.

As many limitations have been identified for the full concept of IBD to be possible, the methodology can only be a description of how IBD could preferably work in future construction projects. To place the methodology in a current perspective, the description will be followed by examples of how IBD could function with the possibilities currently available.

5.2 Iterations and Design Model

The idea of IDP was to change the decision-making process of the design phase into an integrated procedure which should ensure a steady development in the project without the need for redoing previous iterations. Such a process requires access to the latest project information and extensive possibilities to evaluate different design alternatives properly. Using an integrated BIM, the latest information would be easily accessible for all members of the design team and simulation tools could be used extensively – based on the information available – to provide a proper basis for decision. The integrated BIM should be stored in a data repository defined using the IFC specifications as this could allow for proper interoperability among all design and simulation tools used. Based on these findings, the first part of the methodology will be to include a design model (a part of the integrated BIM) in the decision-making process of IBD to ensure proper information sharing. This part of the methodology is illustrated in Figure 5-1.

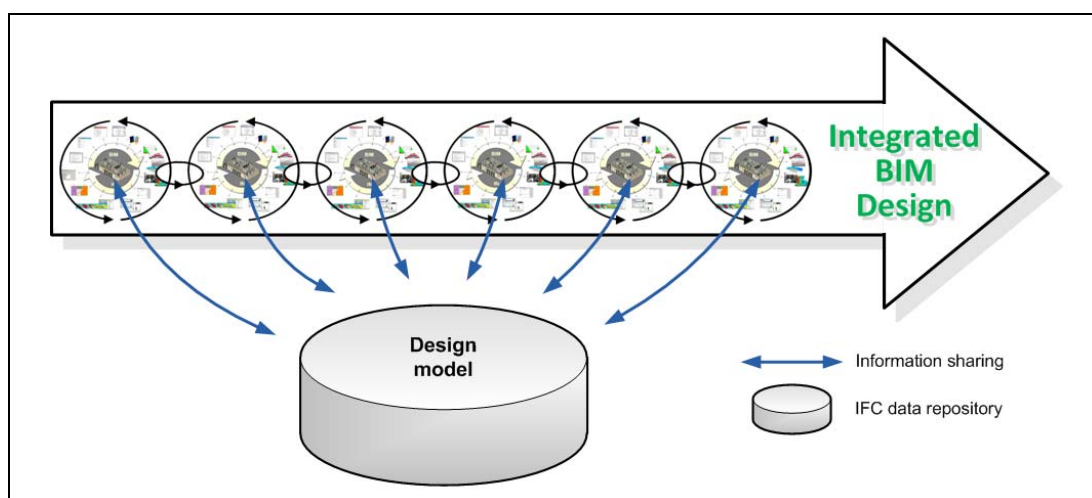


Figure 5-1. Information for the decision-making process of IBD provided by a design model.

Figure 5-1 demonstrates how the design model should constitute the main source for design related information in the integrated procedure. For this reason, it must be the task for the design team to enrich the design model with the latest project information to insure an up-to-date model which would represent all information available. Simulation results and design decisions must therefore also be stored in the design model to ensure that information is available for the following iterations. The information exchange between the design model and the activities in the decision-making process is therefore a two-way communication.

The identification of input and output data required for simulation tools indicated that the amount of information which needs to be shared could be very large. The use of time series could e.g. contribute significantly to the data amount if annual variations were included on a daily basis. The data repository should therefore preferably be located on a model server as described in 2.3.5 *Centralised Data Repository – Model Server Concept* because this would ensure an adjusted and reduced data flow among design and simulation tools. The model server also has the advantage of being capable of store more data than just the current design model. This could be used to document a history of the design development which would be useful in case questions in relation to the design development arise.

As described in section 3.3.1.1 *Separation of Requirements and Design Model* several design alternatives could exist simultaneously. In such a case all alternatives should be stored in similar data repositories – preferably on the same model server – and this would allow for easy accessibility to all alternatives which should insure improved possibility to evaluate different design alternatives with an increasing amount of simulations.

5.3 Requirements Model and Control

To insure proper management of the space of solutions in IBD it was identified that the constraints in a space of solutions should be defined in a requirements model separated from the design information. Such a requirements model should be defined similar to the client requirements model suggested by [Kiviniemi 2005] which is a data repository also defined by use of the IFC specifications. As both the client requirements and space of solutions is intended to form the framework for the design development and includes similar content, it would be appropriate to combine the two solutions in a shared requirements model.

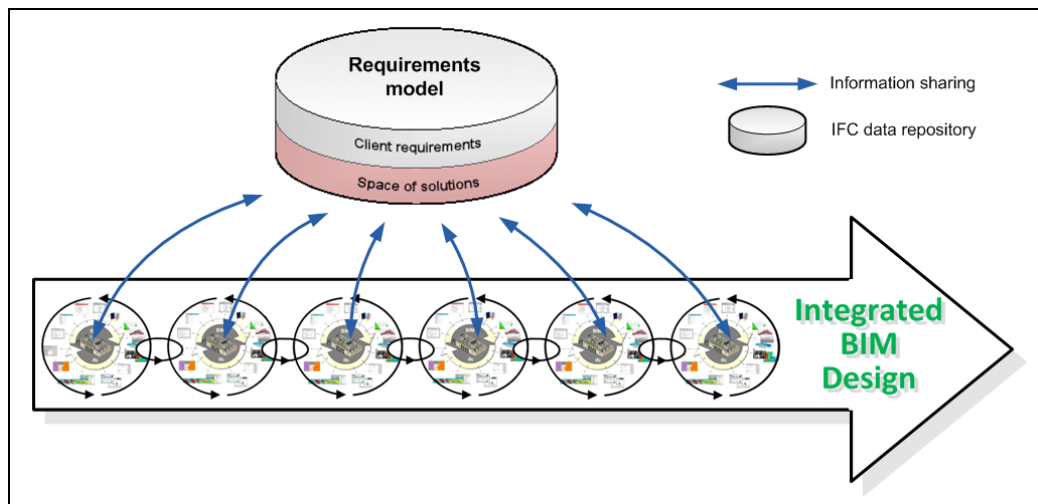


Figure 5-2. Information sharing for a requirements model in the decision-making process in IBD.

Based on these findings, the second part of the methodology will be to include a requirements model (also a part of the integrated BIM) in the decision-making process of IBD to ensure that decisions are taken within the framework defined. This part of the methodology is illustrated in Figure 5-2.

The way to provide information to the design team during the decision-making process could be via an interface as previously illustrated in Figure 2-19 where an example indicates how the information from the requirements model is accessible directly in ArchiCAD. This way the design team would always have access to latest information on the requirements and thereby be able to make decisions in accordance to this. As the design progresses, further simulations would allow for the constraints in the space of solutions to be tightened and extended to cover more parameters, and the requirements model must therefore be continuously updated to include latest decisions on the framework for the design development. This means that the information exchange between the requirements model and the activities in the decision-making process is also two-way communication.

Since the requirement objects are defined using the IFC specifications they would include an attribute describing ownership information as indicated in section 2.3.3 *IFC as the Shared Data Repository*. This ownership information could be used to separate client requirements from constraints in a space of solutions and thereby allow for the two to be defined in a shared data repository. Similar to the design model, it would also be appropriate to place the requirements model on a model server to be able to adjust the data flow to the specific activity in the decision-making process and to have an ability to maintain a history of the development in the requirements model.

The requirements model further allow for possibilities for automatic checking of design compliance. This is an activity which stretches across the decision-making process in IBD to evaluate the design model at various stages in the process and this will be described in the following section where the two parts of the methodology are combined.

5.4 Methodology of IBD

Combining the two parts of the methodology constitutes the definition of IBD for this thesis. The combination is illustrated in Figure 5-3.

The methodology has been supplemented with an indication of potential checking for design compliance which should be used as a minimum in the transitions between iterations. Here the latest design decisions would be reflected in the design model and these can be compared to the latest requirements defined in the requirements model.

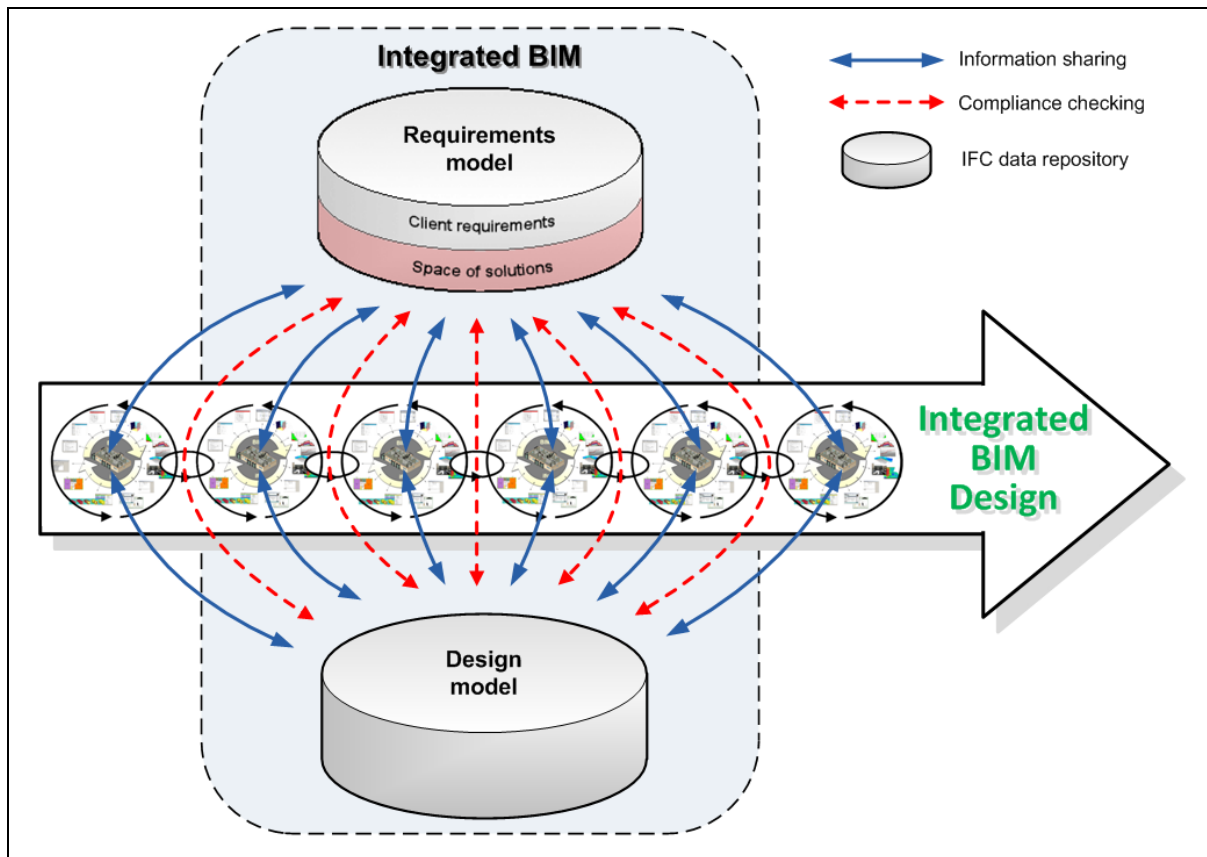


Figure 5-3. A methodology to define Integrated BIM Design.

The combination of the requirements model and the design model constitute the integrated BIM, and as indicated in Figure 5-3, IBD requires extensive interaction with this integrated BIM for the process to progress as intended. To improve IDP by using BIM design this is, however, believed to be how BIM design can contribute most effectively to ensure that the design progresses with the best possible basis for decision and in a properly controlled environment.

As documented in the previous chapters, many limitations apply to both the capabilities of the IFC specifications to allow for the definition of all required information and to the management, design and simulation tools to be capable of interacting with the integrated BIM via IFC. Some possibilities are, however, identified and to illustrate the idea of the methodology of IBD, the following section will illustrate how a design process could currently progress using the methodology of IBD.

5.5 Use of Methodology in Current Design

To demonstrate how IBD could be used in a current design situation, an existing building has been selected as the basis for the demonstration. The building is a 3 storey office building with a basement located at DTU. It is called building 118 and has a gross area of 5.500 m². It primarily consists of small offices, meeting rooms, classrooms, storage rooms and a few other rooms such as toilets and kitchens. The building is a long rectangular block with windows facing north and south. This building has been selected because a 3D object oriented model of the building has already been created by students at DTU using AutoCAD Architecture 2008. The model is illustrated in Figure 5-4. The model includes spaces, walls, slabs, columns, windows and doors.

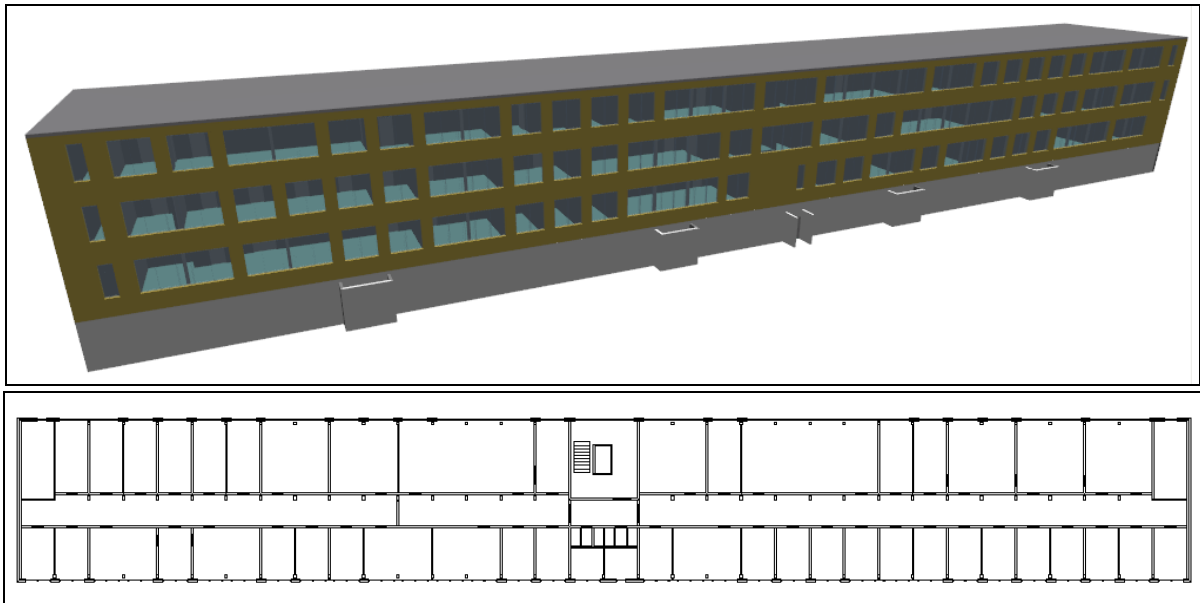


Figure 5-4. Model of Building 118 at DTU created in AutoCAD Architecture 2008. Plain drawing illustrates the layout on the first floor of the building. North would be upwards on this drawing.

To demonstrate how a decision-making process of IBD could be used to design such a building, four steps have been selected as illustrated in Figure 5-5. These follow the procedure described in 2.2.5 *Space of Solutions* which includes room design, total building composition and optimisation of building design. The four steps selected only constitute some of the design steps required for building design and only limited part of each step will be described in the following.

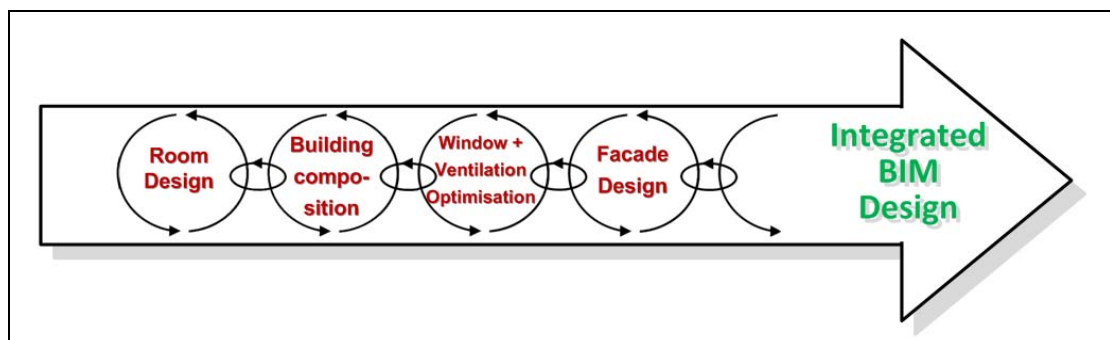


Figure 5-5. Four selected design steps in IBD.

The demonstration will use iDbuild for the initial room design and AutoCAD Architecture for additional design of spaces and the total building. Riuska and Be06 will be used in the optimisation of the building. To conduct compliance checking within the process, Solibri Model Checker will be used. All information will be stored in IFC files to the extent possible. All additional information is assumed to be defined in documents as it would be the case in a traditional design process.

All information which will be included via property sets in the design model is created in AutoCAD Architecture similar to the description in Appendix 17. Riuska creates its property set automatically.

To limit the scope of the demonstration, focus in the following will primarily be on the design of the small offices in the building. As only limited information can be defined in IFC and even less information can be included for compliance checking only a few constraints will be included in the requirements model.

5.5.1 Definition of Client Requirements

Before the design phase can begin, the client requirements must be defined. The client requirements will be based on the existing design of the small offices in building 118. Here the client has identified that he or she needed 79 single cell offices and that 18 of these needed to have a size of approximately 18 m² whereas the additional 61 only needed to have a size of approximately 12 m². To simplify the demonstration the small offices have been selected to be placed along the northern façade and the small offices with an area of 12 m² are placed at the southern façade. This results in a set of client requirements as illustrated in Figure 5-6.

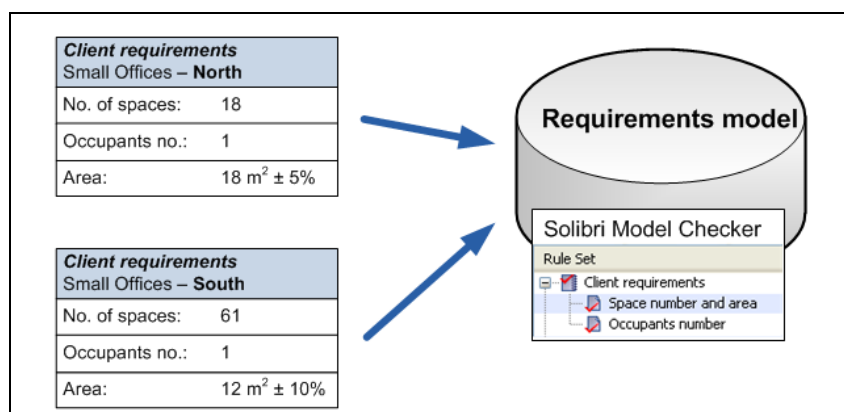


Figure 5-6. Defining client requirements for small offices in building 118.

As client requirements cannot currently be stored in an actual requirements model, the requirements are defined as a rule set in Solibri Model Checker. How the requirements are defined in Solibri Model Checker is described in Appendix 18. The rule sets must therefore constitute the requirements model for now.

5.5.2 Room Design

Based on the client requirements, the design team can now start the design process by creating optimal room design for the offices. To do so, several simulations are conducted in iDbuild. For this demonstration the focus will be on optimisation of the U-value of the closed part of the façade and the height of the windows.

Many other parameters should be evaluated to properly identify optimal room design, however, for now the original shape of the offices and standard values for performance are used for the additional parameters. This means that the rooms orientated to the south will have the dimensions 2.9 x 4.4 x 3.0 meter and the rooms orientated to the north will have the dimensions 2.9 x 6.2 x 3.0 meter.

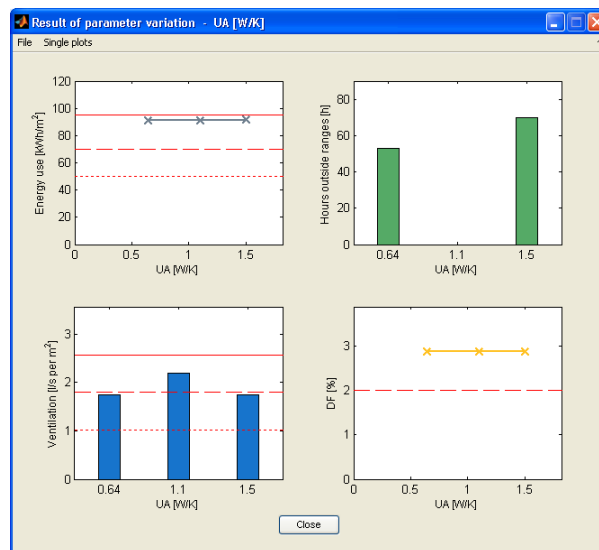


Figure 5-7. Parameter variation in iDbuild of U-value in façade for small offices along the southern façade. The variation is from 0.15 W/m²K to 0.35 W/m²K.

Based on a simulation in iDbuild, the parameter variation of the U-values of the façade can be created as illustrated in Figure 5-7. This indicates that the energy use in the room is hardly affected by changing the U-value of the closed part of the façade. Based on analyses of the results, it is decided that as long as the U-value is below 0.25 W/m²K the performance will be acceptable.

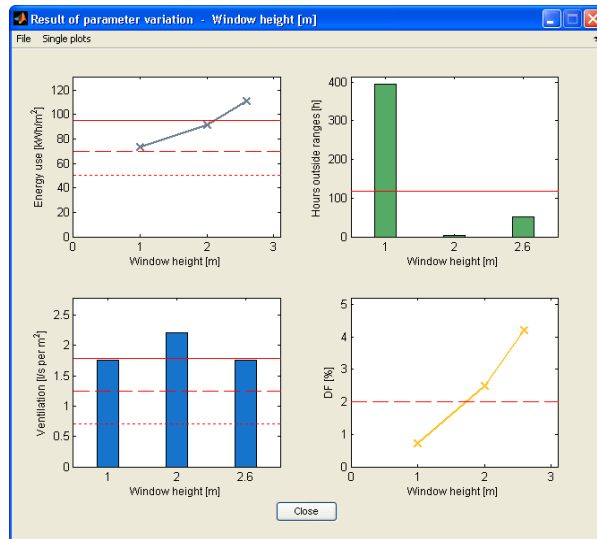


Figure 5-8. Parameter variation of window height of offices along the southern facade. The variation is from 1.0 to 2.5 m.

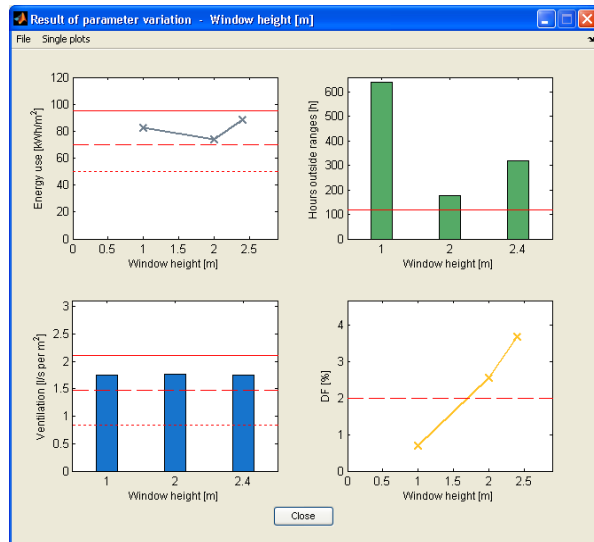


Figure 5-10. Parameter variation of window height of offices along the northern facade. The variation is from 1.0 to 2.5 m.

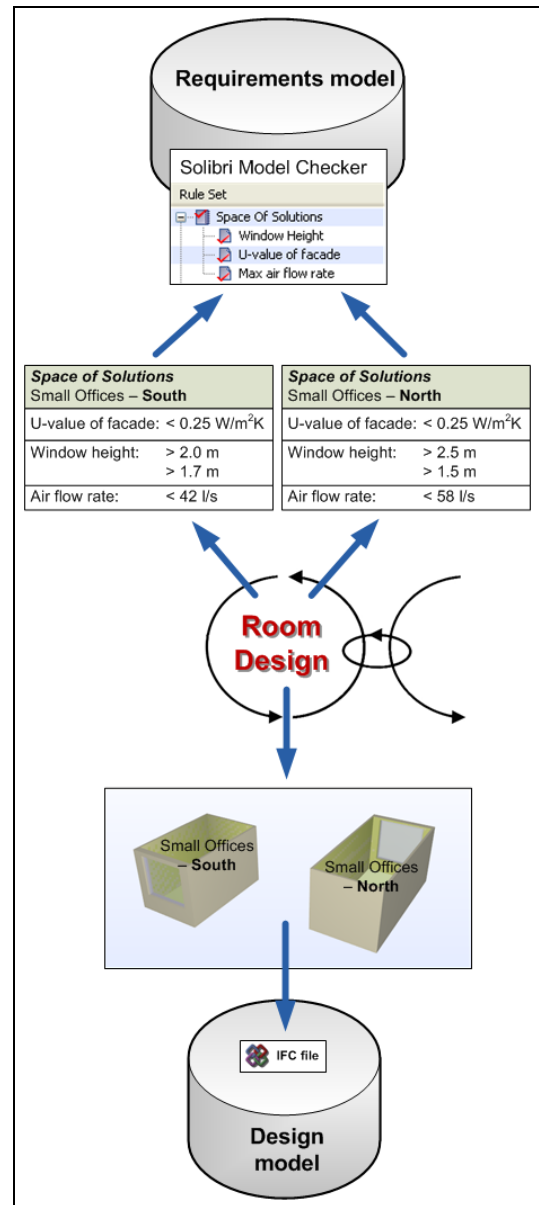


Figure 5-9. The first iteration in IBD for optimisation of room design.

Other simulations in iDbuild have resulted in parameter variation of the window height for the two office types. The results are illustrated in Figure 5-8 and Figure 5-10 for the southern and northern orientation respectively. The red line in the graph for energy use defines the energy frame allowed for buildings according to the Danish Building Code. For this reason, the results indicate that the windows orientated to the south must have a height less than 2.0 meters to ensure that each room would not contribute to exceeding the energy frame. The windows orientated to the north can have a height of up to 2.5 meters without exceeding the limit for energy use. The graph for DF indicates the daylight factor in the room, and this should be around or above 2% for offices. For this reason the windows at both orientations should not be lower than 1.7 meter.

Similar simulations in iDbuild indicates that an air change rate of maximum 4 h^{-1} is appropriate to insure proper indoor environment in the rooms with a reasonable consumption of energy. The basis for this decision will not be illustrated but this value should be the maximum allow¹⁴.

Based on these simulations, three constraints for performance-decisive parameters have been defined for the rooms and these should be added to the requirements model to constitute part of the space of solutions. Similar to the client requirements they can currently only be defined in a rule set in Solibri Model Checker. How these constraints are defined in a rule set is also described in Appendix 18. Based on the simulation results the design team can also generate the initial room layouts of the rooms in e.g. AutoCAD Architecture to start working on the building design. These should be stored in the design model, which for now would be an IFC file. These activities could constitute the first iteration in IBD as illustrated in Figure 5-9 where both the requirements model and the design model are enriched with information.

5.5.3 Building Composition

Based on the layout of the individual rooms, the design team can now compose the building based on these layouts and layouts of additional rooms required by the client for the building. The room layouts can be retrieved from the design model (the IFC file) and combined to compose the total building in an appropriate way. Once the design team has created an appropriate design, the information should be stored in the design model consisting of an IFC file. This could constitute an example of the second iteration in IBD as illustrated in Figure 5-11.

Before the design process continues, the design model should be checked for compliance with the requirements to ensure that the design team has composed a building which meets the requirements. This is done using Solibri Model Checker where both the client requirements and the space of solutions are defined. By importing the latest design model, represented in an IFC file to Solibri Model Checker, such a check could reveal if e.g. the design team has created a building with only 58 small offices orientated to

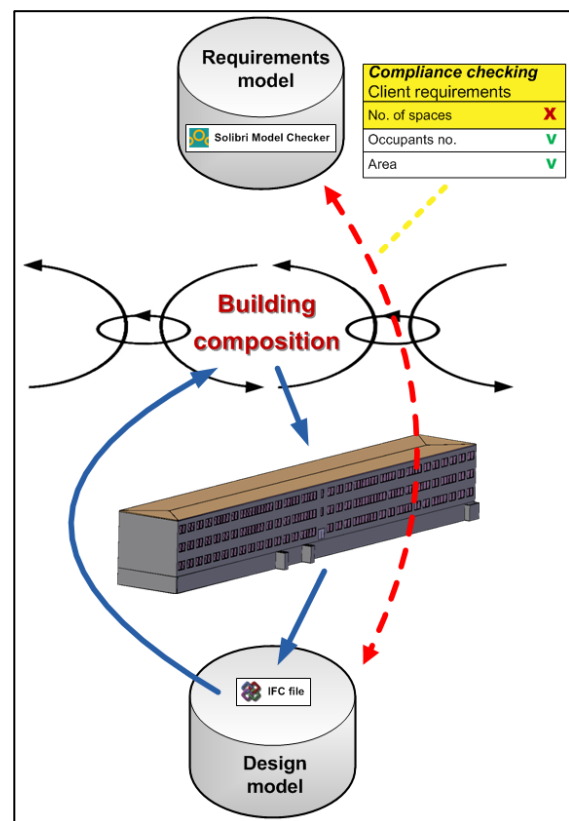


Figure 5-11. Second iteration in IBD for composition of building.

¹⁴ The air change rate must be converted to air flow rate (l/s) to allow for comparison with the values later defined by Riiska.

the south. This would be discovered by Solibri Model Checker as illustrated in Figure 5-12. The design team must therefore redo the iteration until all requirements are met.

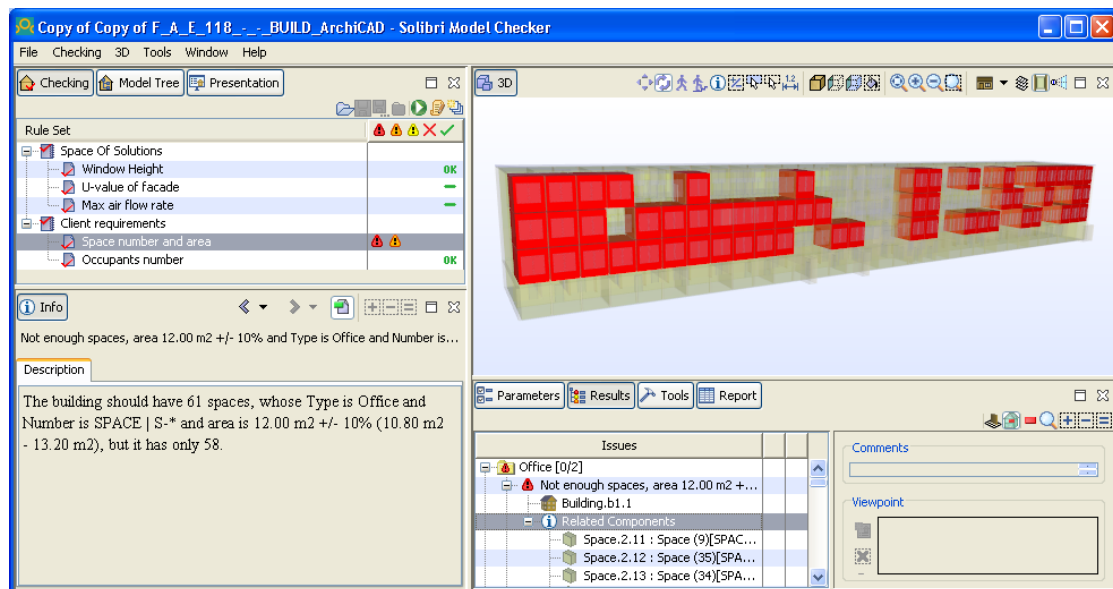


Figure 5-12. Compliance checking in Solibri Model Checker. Here it is identified that there is less spaces than required for the office type orientated to the south.

5.5.4 Optimisation of Windows and Ventilation

The individual optimisation of rooms can lead to different constraints for e.g. the window height for different type of spaces. This might lead to different window heights in the façade and as this could be undesirable from an architectural point of view, it would be appropriate to adjust the window height to be identical for the entire façade. To identify the optimal height, it requires energy simulations of the whole building. Such simulation can be done by Riuska, by creating several design alternatives from the current design model. As the small offices in the southern façade is found to have the smallest allowed interval for window height in their space of solution, two design alternatives are created in AutoCAD Architecture from the current design model: One where all windows are 2 meters high and one where all windows are 1.7 meters high. These design alternatives can be exported as IFC files¹⁵ which can then be imported into Riuska as illustrated in Figure 5-13.

¹⁵ Before IFC files from AutoCAD Architecture can be imported into Riuska they must be imported into ArchiCAD and exported from here again. This is required as Riuska also needs information of space boundaries which only ArchiCAD can currently export to IFC.

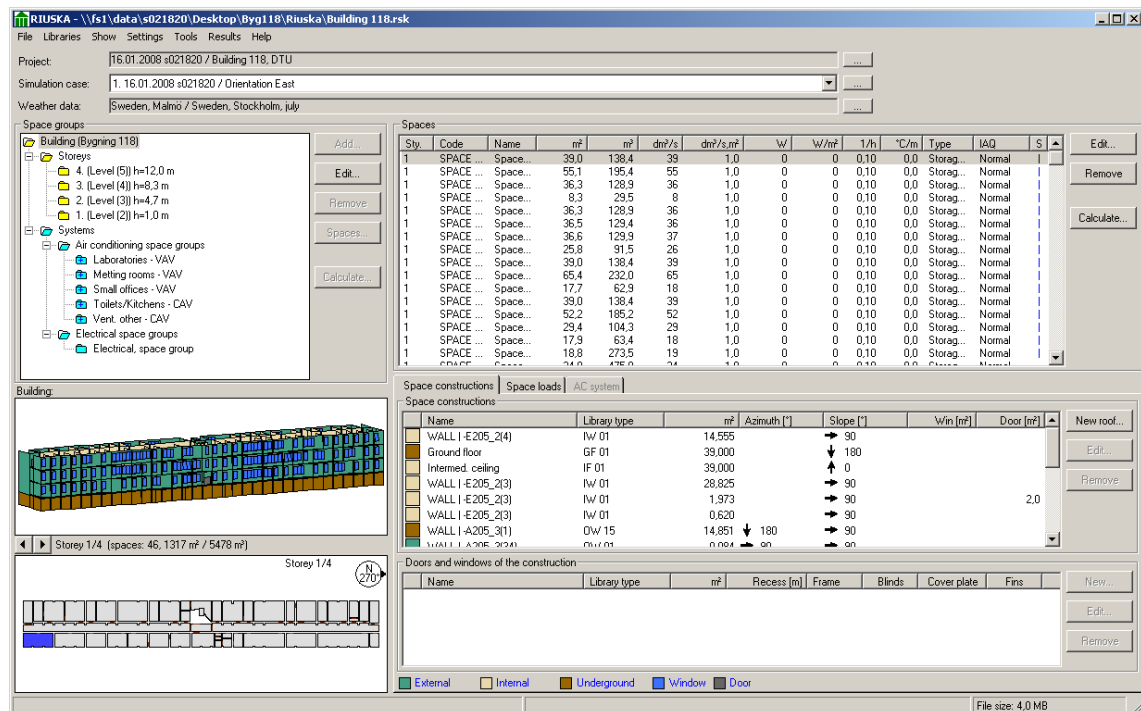


Figure 5-13. One design alternative imported into Riuska and supplemented with thermal and performance related information.

For now Riuska only imports information of geometry, and additional information of thermal properties and performance of HVAC systems must therefore be manually defined. Riuska includes functionality to import different design alternatives and reuse all manually defined information for imported objects identified to be similar in the two cases. In the current situation, the comparison of the design alternatives is completed by only defining manual information once in Riuska. The results of the two simulations are illustrated in Figure 5-14. This indicates that the low window height of 1.7 meter results in the lowest energy consumption for the whole building. For this reason it is decided to select the low window height for all windows in the facade.

Using Riuska, the ventilation system in the building could also be optimized. Such optimisation requires a large range of considerations in regards to thermal comfort in each space, the size of the system, the control type and potential capacities. A description of such an optimisation has been omitted and instead it is only assumed that such an optimisation has been conducted. The results of the Riuska simulation can be exported to the design model which would now consist of the IFC file with the design alternative including the low window height.

In the calculation of energy consumption Riuska only includes some consumptions required to assess the energy frame of the building according to the Danish Building Code. To ensure that the total building meets the requirements for the energy frame, it is required also to use Be06. Using the IFC interface developed in Chapter 4, the import of limited information can be done directly from the IFC file containing the design alternative and the simulation results as illustrated in Figure 5-15. Supplementing the information with guideline values for HVAC performance in the building, Be06 calculates an energy frame of 89.4 kWh/m²/year which is below the required limit of 95.4

kWh/m²/year for such a building. The design alternative is therefore accepted and the iteration can be completed.

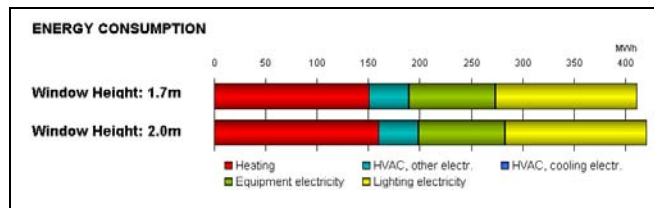


Figure 5-14. Comparison of simulation results in Risika of energy consumption from building with window height of 1.7 m and 2.0 m.

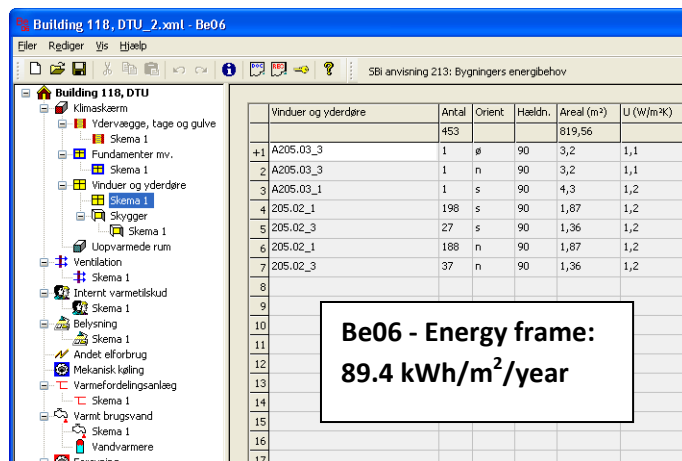


Figure 5-15. Import of design alternative with low window height in Be06 via IFC interface and the resulting energy frame calculated by Be06. The figure shows information imported for windows and doors in the building.

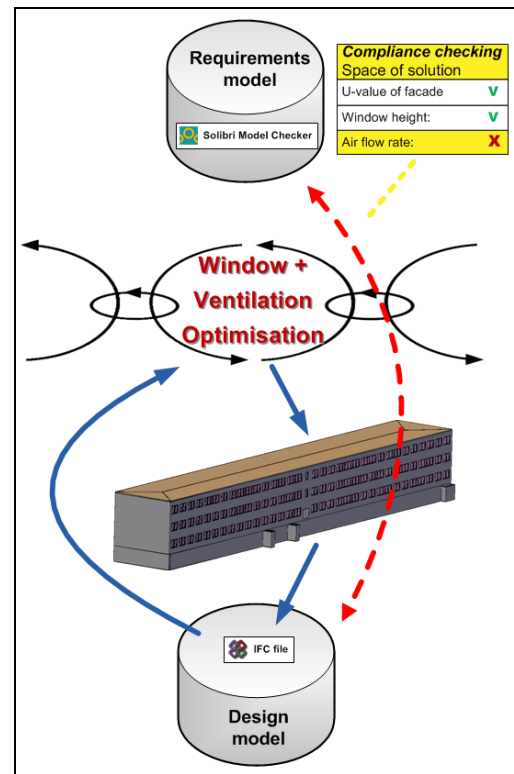


Figure 5-16. Third iteration in IBD for optimisation of window height and ventilation system.

Before the design process moves on to the next iteration, compliance checking should be conducted. Such a check could e.g. discover if the optimisation of the ventilation system has created a supply air flow rate to any of the rooms which is outside the constraints of the space of solutions as the case in Figure 5-17. To prevent problems with draught, these air flow rates should be adjusted before the design process continues.

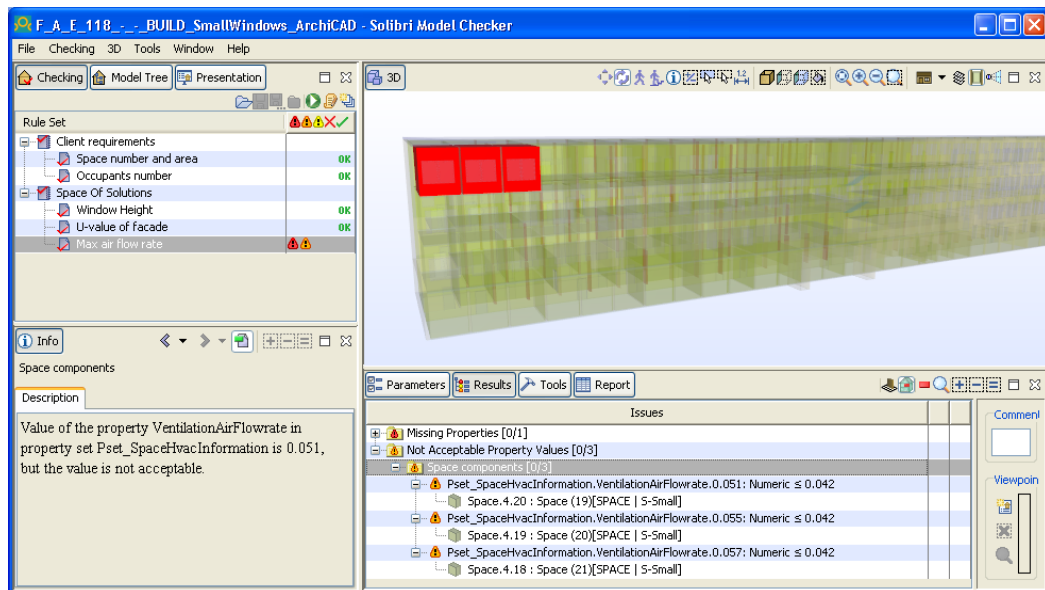


Figure 5-17. Compliance checking in Solibri Model Checker. Here it is identified that the air flow rate for three rooms facing south are above the required limit of 42 l/s.

The third iteration is illustrated in Figure 5-16 and it demonstrates how information are shared from the design model to form the basis for the iteration, and how the design model is then enriched with information on the selected design alternative including results from the simulations. This information can then be used for compliance checking and only when the design team has ensured that all requirements are met, the design process can continue with the next iteration.

5.5.5 Façade Design

Many further iterations could be required before the design process would be completed. One such iteration could be the detailed design of the façade. The architect would have requirements in relation to the look of the façade and the structural engineer could have requirements in regards to the load bearing capabilities. The design team must therefore try to develop solutions for the façade which satisfy all requirements. Some of these requirements could already be defined in the space of solutions such as constraints for the U-value which was defined in the first iteration. In the decision-making process of selecting solutions for the façade, this constraint should therefore be included as a reference. In case the design team should select solutions which would result in the U-value of the closed part of the façade being too high, this would be detected when the design team tries to move on to the next iteration and compliance checking is carried out. Such a check is illustrated in Figure 5-18 where the design model is found to include walls which do not comply with the space of solution.

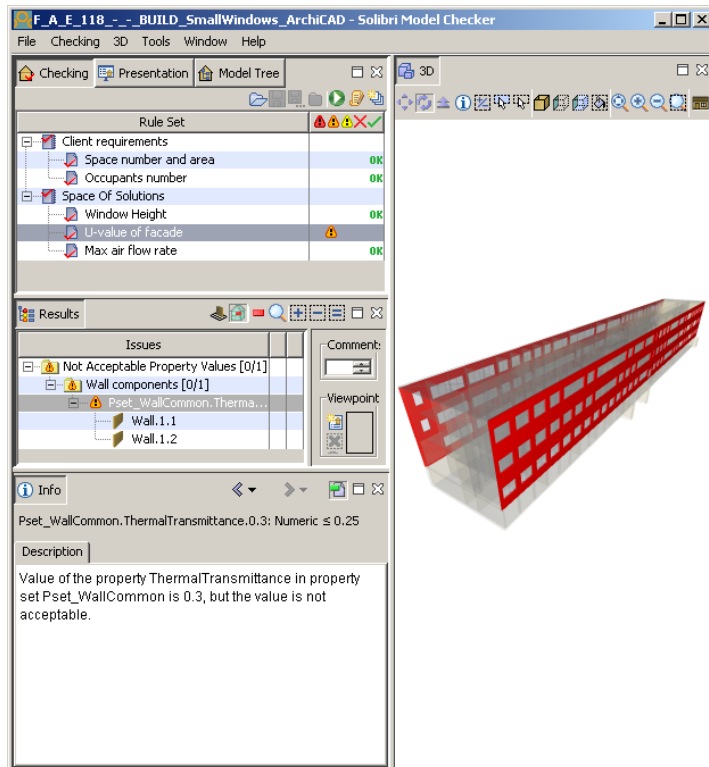


Figure 5-18. Compliance checking in Solibri Model Checker.
Here it is identified that some walls in the façade does not fulfil the requirements in the space of solutions.

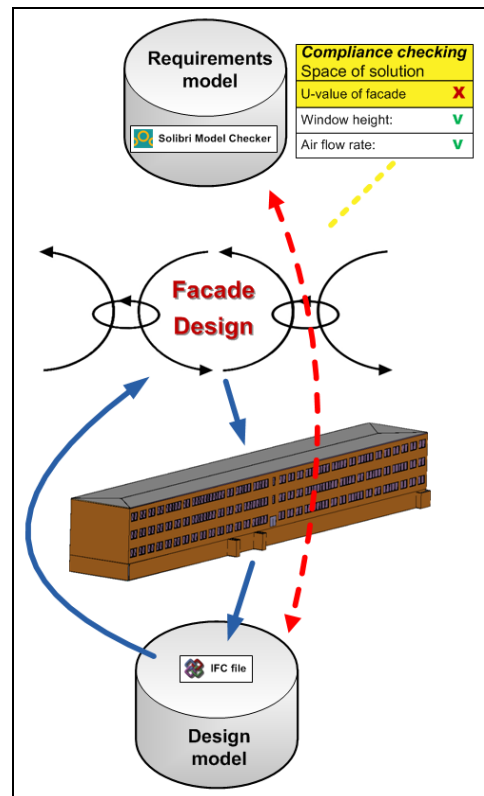


Figure 5-19. Fourth iteration in IBD for detailed façade design.

In any future iteration it is therefore assured that the latest information in the design model will also follow the requirements defined in the space of solutions as illustrated in Figure 5-19. Furthermore the requirements model could also be refined or extended with additional constraints for the space of solutions.

5.6 Summary of Potentials in the Methodology for IDB

The demonstration of some current possibilities of using IBD is restricted to very few parameters and possibilities which indicated that the methodology is far from being possible to use by its full potential in current design. For now limited information from the requirements model can be stored in rule sets in Solibri Model Checker and different versions of the design model can be stored in individual IFC files. This concept is far from the idea of interacting with an integrated BIM. However, it allows for a few restricted activities intended for the design process in IBD to be possible using current design and simulation tools.

In spite of the limitations, the activities described are believed to illustrate several interesting potentials of using IBD. The abilities demonstrated in the third iteration with fast evaluation of several design alternatives and the use of two simulation tools to take several aspects into account at the same time illustrates the possibilities to generate a proper basis for decision. Although this is possible in traditional design with manual input of information, the ability to retrieve much information directly from the design model now insures that the starting point for the simulations

includes consistent and latest information and that the preparation time for the simulations can be significantly reduced.

The continuous compliance checking of the design further more indicates the advantages of defining information in the integrated BIM. Although only few constraints can be included for now, it would still be a comprehensive and uncertain task to conduct such checking manually by the project management. With the six constraints in the demonstration it would require 474 manual checks to insure that all information complies in the 79 small offices alone.

If the intention is to ensure that the design develops as illustrated in the demonstration, it therefore seems as a necessary demand that all requirements and design information are defined in the integrated BIM for a proper development of design in IBD. If the development of IFC interfaces and new management tools is hopefully accelerated in the near future, the methodology of IBD would then allow for a better basis for decision and extended possibilities for control of the decision-making process in IDP.

Chapter 6

Discussion

6.1 Introduction

The previous chapter defined a methodology for IBD and demonstrated some limited possibilities to use part of the methodology in a current design phase. The methodology is, however, found to be far from possible to use by its full potential with the technology currently available. In order to encourage future development that would allow for IBD to be used as intended in construction projects, the following sections will describe the effect believed to be caused by using IBD and suggestions as to how the technology barriers can be reduced.

6.2 Consequences of Implementing IBD

In the literature study, IDP was identified as a concept of reorganising the decision-making process to ensure an improved result using existing methods and tools by process integration. Correspondingly, BIM design was identified as a concept of making the existing process more efficient by data integration. A proper combination of the two concepts should therefore create an efficient design process which generates an improved result. This is what the methodology of IBD is believed to do by combining an integrated BIM with the integrated procedure in the decision-making process. The effect of implementing IBD should therefore contribute positively to the design process. However, compared to a traditional design process, IBD significantly changes the work methods of the design team and it will therefore involve several challenges to implement the methodology.

6.2.1 Complexity of Information Sharing

From the description in the previous chapter, the methodology of IBD could seem complex with the large extent of interaction with the integrated BIM required to ensure the use of the latest information. This should, however, be seen in contradiction to a manual process of IDP, where the same amount of information sharing would be required if the equivalent basis for decision was to be achieved. In a manual process, a large amount of information is typically shared via verbal communication, drawings and based on various documents such as minutes from meetings and a large range of individual specifications of different components. The way information is shared is thereby dispersed by many communication channels which make it a less visible part of the process. In reality, information sharing during a manual process is most likely much more complex than the methodology of IBD. In IBD information sharing is simply emphasised because this is believed to be of great importance for the improvements of the decision-making process. The collection of information in a well structured and easily accessible integrated BIM should further contribute significantly to the efficiency and consistency of information sharing. The effect of implementing IBD would therefore only contribute to more efficient information sharing as opposed to making the process more complex.

6.2.2 Effort of Implementation

The idea of using BIM design in IDP is that the fundamental concept of IDP remains unchanged. The intention is instead that BIM design should improve the framework, the efficiency and the results of the process. Implementing IBD is therefore believed to require the same effort as if the concepts of IDP and BIM design were implemented separately within the consultants companies and design teams.

The implementation of IDP will result in new work methods for the design team as many members will be involved from the beginning of the design phase. This will require that the design team is able to co-operate more closely and the outcome of the process will be a shared product more than individual contributions. This creates challenges in relation to both contractual issues and legal issues. It can e.g. be difficult to decide who to blame if any mistakes are found later in the construction phase when several members have contributed to the same design solutions. It will furthermore require a new fee structure as each member of the design team is now involved in a much larger part of the design phase making it unclear who is contributing to various parts of the design. It will be out of the scope of this thesis to find proper solutions to these issues, however, it will most likely require a new approach where responsibility is shared among the design team members and fees are based on the product more than on the individual contributions to the design.

The implementation of BIM design will affect the work methods for each design team member as all information now has to be defined in the BIM. This requires enhancements of qualifications to handle BIM applications by each member of the design team. Furthermore, it will require significant investments in new technology as a range of design and simulation tools is required along with new platforms to store and maintain the data repositories.

For the design team members to understand how to act during the decision-making process in IBD it will further require additional training when combining the two concepts of IDP and BIM design. Implementation of IBD therefore requires a significant effort to ensure proper education, adjustment of agreements and investments in technology.

6.2.3 Ensuring Development and Quality in the Design Process

Another challenge of IBD could be that the amount of requirements which apply to the development of the design could simply impose so many restrictions, that it would prevent a steady development of the design. Fire requirements would e.g. typically require that doors should open outwards whereas accessibility requirements would require that doors should open inwards. Such issues will demand that some requirements must be compromised by proper judgment for an alternative solution. Similar contradictions could apply to several types of requirements and the methodology of IBD should therefore include some sort of flexibility or prioritising to allow for the design to develop. As it would be complicated to make predefined prioritisation of all requirements it should be up to skilled design team members to make such judgement. The advantage of IBD is, however, that any deviations can be clearly identified and the overview by the project management should therefore be intact to ensure that the deviations will not cause the design to move away from the design goals.

The methodology also poses challenges by requiring that the latest information is always reflected in the integrated BIM. If e.g. the composition of a wall is changed, it must be ensured that the U-value for the wall and the heating requirements for the spaces on the inside are updated along with all other parameters affected. This must be done to reflect the changes before any of these values are used for further assessments of performance. A possibility could be to impose constraints to all values so that if one value changes, all affected values will be required to be updated before they can be used for further assessments. Such interrelated constraints are, however, believed to be far too comprehensive to define and manage. A change in window height would for example cause changes to anything from the comfort level to the performance of the ventilation system and it therefore calls for a proper simulation tool to handle such changes. To ensure proper quality of the information in the integrated BIM it must therefore require well defined procedures to define how changes should be handled and when information should be valid.

The above considerations indicate that the implementation of IBD not only depends on technology improvements, but that implementation is additionally met by other challenges which it is equally important to address. Attention should therefore be directed at developing solutions to these challenges as well as at technology improvements. The increased focus on meeting the other challenges should also encourage developers of technology to continue on their part.

In relation to the methodology of IBD, the developers of technology are primarily facing challenges in relation to interoperability and requirements management. This will be discussed in the following sections.

6.3 Achieving Proper Interoperability

The main challenges to achieve interoperability among simulation tools using the IFC format have previously been identified to be caused by limitations in both the IFC specifications and the IFC support by design and simulation tools. As some challenges are easier to overcome than others, the following discussion distinguishes between improvements within a short and longer time frame of 0-1 year and 1-5 years respectively.

6.3.1 Improvements within a Short Time Frame

Although full interoperability cannot be achieved until the IFC specifications are fully capable of defining all information required, up to 70 % of the information required by simulation tools has been identified to currently be possible to define. Implementing the suggested extensions to the property sets described in Appendix 14 would further increase the average amount of information which can be defined to 85 %. As all suggestions in Appendix 14 only relate to property sets which can easily be adjusted in the IFC specifications, this would be an appropriate step to take within a short time frame to allow for improved interoperability.

The current support for IFC by the design and simulation tools is far from capable of handling such a large amount of information as most of the information exchanged relates only to geometry and limited material properties. Within a short time frame focus should therefore be to improve the IFC implementation in design and simulation tools so that more of the information available can be exchanged. It would e.g. be obvious for a tool like Riuska to be able to import information about the material properties and in this way avoid the large amount of manual work for selection of such information for each individual object. Other simulation tools with no current IFC support should start by supporting import for geometry as all information is typically defined in relation to a geometric model. Material properties could also be supported as these are currently available from all design tools. These improvements would not allow for IBD to be implemented as intended, however, it could improve the use of simulation tools in the decision-making process considerably and thereby contribute to an improved basis for decision.

Both the analysis of IFC capabilities and the development of the IFC interface for Be06 indicate that there is, however, several more fundamental challenges which should be addressed before a thorough and consistent implementation can be completed. This will require a longer time frame to be accomplished.

6.3.2 Improvements within a Longer Time Frame

Even though up to 85 % of the required input and output data for simulation tools would be possible to define within a short period of time, the decision of how and where information should be defined to ensure consistency in the implementation of IFC still poses a significant challenge. This thesis has suggested an approach which assigns information to either the spaces or to individual devices in the building services systems to the extent possible. Because much information could also be assigned to the systems or in other property sets not included in this thesis, there is an imminent risk that implementation will not be consistent if individual software developers would create in-depth IFC implementation based solely on the IFC specifications. As previously mentioned, this calls for proper guidance of how to implement IFC support in design and simulation tools.

First of all there is a need to agree upon which information to define in the BIM. Riuska e.g. requires information about the temperature increase over each fan in the ventilation system whereas BSim calculates this value based on information on the power consumption of the fan. To ensure that only a reasonable amount of information will be defined in the BIM, an agreement must be reached of which information to define in the BIM and which information is expected to be calculated by the simulation tools. It is the goal of the Information Delivery Manual project to define which information to include in the BIM at various stages of the process and the completion of this project will therefore be a prerequisite for ensuring improvements for interoperability. Based on these definitions it must then be required that the IFC specifications are updated to allow for all selected information to be defined.

Secondly, agreements need to be made upon how the selected information should be defined consistently using the IFC specifications. This is the goal for the Model View Definitions also being developed at the moment, and completion of these definitions is therefore also a prerequisite for

ensuring improvements of interoperability. At present the Model View Definitions only include concepts for exchanging information from architectural design tools to simulation tools and from simulation tools to HVAC design tools. Information exchange among simulation tools is not part of the current scope and this needs to be included for the definitions to be useful for the implementation of IBD.

The creation of the IFC interface for Be06 indicated that an agreement on how to use IFC is, however, not the only challenge for successful interoperability among simulation tools. It was also identified that it can be complicated to correctly map information available in the BIM with information required by the simulation tools. Be06 in particular but also most other simulation tools are designed from the basis that the majority of information today is manually derived by a design team member. The way of measuring gross area sizes in Be06 and the way to define window shading in most tools by assigning shading geometry to each window are examples of cases where information would not be defined correspondingly in a BIM but would be simple to derive manually from a drawing.

For successful interoperability among simulation tools it would improve the consistency in the use of information if the simulation tools were adjusted to use more of the information directly available in the BIM. The BIM is intended to constitute a digital representation of the actual building and it should therefore allow for more accurate simulations if the tools were adjusted to a BIM orientated approach. Be06 could e.g. be adjusted to handle a more precise representation of the building envelope as it would be defined in an object oriented model and all tools could be able to use the actual geometry of the building instead of simplified representations of e.g. window shading. This would allow for improved consistency in the way information is used in the tools and it would be simpler to define the correct information using IFC.

From an ideal point of view, interoperability would be even further improved if the tools were adjusted so that their internal structure would be similar to the hierarchical structure of the IFC specifications. This way the complex mapping of information defined in IFC with information required in individual simulation tools could be avoided, and the consistency of information would be much easier to maintain.

The use of a model server was previously identified as an appropriate way to handle the information sharing. For this reason the tools should furthermore be adjusted to be able to interact with the design model via such a model server. At present, no common interface to the available model servers exists, and this further poses a challenge for proper implementation of IFC support.

All of these improvements to the IFC specifications, the IFC support in design and simulation tools and the internal handling of information in individual tools will require a significant effort to be developed and implemented. The improvements are, however, necessary to allow for proper interoperability during IBD. Even with a considerable effort, the development and implementation will most likely take several years and proper interoperability via IFC will not be possible until then.

6.4 Allowing for Requirements Management

A significant effort has also been identified to be needed before the requirements in a space of solutions can be defined and managed properly. Similar to the discussion on interoperability, the following discussion distinguishes between improvements within a short and longer time frame of 0-1 year and 1-5 years respectively.

6.4.1 Improvements within a Short Time Frame

As one of the main barriers for including constraints in a space of solutions was previously identified to be the very limited capabilities in the IFC specifications, a possibility within a short time frame would be to define the requirements in another platform which could be more freely adjusted. This could be in a compliance checking tool like Solibri Model Checker as it was done in the demonstration in Chapter 5. The possibilities in Solibri Model Checker are for now limited as only predefined rules can be created. However, as indicated in Appendix 18, Solibri Model Checker holds capabilities to support both direct and indirect linking between requirements and design models defined using IFC. Minor extensions to the functionality in Solibri Model Checker would therefore allow for considerable possibilities to link constraints in a space of solutions with a design model within a short time frame.

Using this procedure would, however, lock the space of solutions to be accessible in Solibri Model Checker only and information sharing of requirements is thereby limited. A proper implementation of IBD should include a requirements model and this call for several additional improvements which will demand a longer time frame to be developed.

6.4.2 Improvements within a Longer Time Frame

Chapter 3 identified a range of extensions required in the IFC specifications before a space of solutions can be defined. Most of the extensions are required because the concept of IFC only allows for constraints from the space of solutions to be defined in unique attributes representing e.g. window height or U-value of a wall. To define precisely which extensions are required in the IFC specifications will therefore require that the content of a space of solutions can be precisely defined. So far the content in a space of solutions is more loosely defined depending on how the design of each individual project progresses. Before extensions can be made to the IFC specifications it would therefore require that a research study identifies precisely what the appropriate content of a space of solutions would be. Only then the IFC specifications can be extended to allow for a space of solutions to be defined. To ensure that information on constraints in a space of solutions is defined consistently in the IFC specifications, implementation guidelines should also be created for these new extensions when they are implemented in the standard.

Based on these implementation guidelines, new tools additionally need to be developed which include capabilities to manage the requirements and similar to the solution suggested for the design model, the interaction with the requirements model should preferably be via a model server.

Compliance checking tools like Solibri Model Checker and other tools which need to retrieve information from the requirements model would additionally be required to extend the capabilities of their IFC interfaces so that they could also import information on requirements.

Even with a considerable development effort in this part of the methodology of IBD, it will most likely also in this case take several years before requirements management can be implemented and proper use of IBD can commence.

6.5 Reasons for Slow Development

As mentioned in Chapter 1 research and development in the area of integration – data integration in particular – within the AEC industry has been ongoing for more than 15 years. Even so, the possibilities to implement only parts of IBD are still found to be limited. This indicates that it could be a significant challenge to ensure that the required development will take place within a reasonable time frame.

Integration is by its nature required to stretch across domains in the AEC industry. The fragmented composition of companies in the industry consisting of either architects, engineers or contractors seem to be a challenge for the AEC industry to develop such solutions which stretch across domains.

The advantage of using IBD was primarily identified to be improved building performance. As IBD should also lead to a more efficient design process, the current negative productivity development in the AEC industry could potentially be changed by using IBD. In turn, this could potentially result in a reduced cost for building construction projects. The effect of using IBD – and other types of integrated procedures – therefore seems highly beneficial for the clients, and it would be appropriate if they took a leading role by requiring that the design teams should start using integrated procedures.

Most clients only have something built once or twice which is not optimal for long-term development, however several professional clients such as developers, large housing associations and governments represent a significant amount of the projects in the industry, and they are believed to be able to contribute significantly to the development if they would start demanding increased integration. A challenge for the clients is, however, that they typically have limited knowledge of the current development in the industry, and their ability to demand the most appropriate solutions for future projects is thereby inadequate. For this reason the client representative – typically a consultant – also has a significant responsibility to ensure that the intentions of the client results in appropriate demands for proper integration.

The Danish project Digital Construction is an example in which a large client, the Danish Government, sets new demands that force both organisations and individual companies in the industry to agree upon new work methods and standards. Unfortunately the results of this project do not reach far enough to support the concept of IBD. However, similar activities for the development and digitisation of the AEC industry is initiated in Finland, Norway and the US, among others, where the aim is to initiate concepts more closely related to actual BIM design. Along with the initiatives

previously described to improve the IFC specifications and their use, this is believed to finally drive the development forward with a faster pace than what has been the case for the last 15 years. Development specifically directed towards the methodology of IBD would still be required; however, the starting point for the development will be much improved due to the ongoing activities.

Furthermore, it should be mentioned that the information contained in the integrated BIM would additionally allow for several other process to be improved. The large amount of information gathered in the BIM could for example be used in the operation phase to ensure that the building is controlled as intended by the designers. In case of future renovation projects, the information would also constitute valuable knowledge that could allow for a much better basis for the expansion of existing systems and solutions. The use of a BIM will also contribute considerable to many other areas such as construction management and cost estimation.

In sum, the advantages of using the methodology of IBD are likely to outweigh the significant effort required for the implementation to be possible. For a fast implementation the development should, however, be pushed forward by clients who could in turn benefit considerably from its results.

Chapter 7 Conclusion

The aim of this thesis was to identify how an integrated design process can be improved by using BIM design and further how the capabilities of the IFC specifications could support this improved process. The result of the thesis is a methodology of Integrated BIM Design which constitutes a proposal of how the concepts of integrated design and BIM design could be combined.

Based on the research in this thesis, it has been identified that such a methodology should consist of a decision-making process which follows an integrated procedure to ensure that the design goals are met. To support the decision-making process the project related information should be defined in an integrated BIM. This integrated BIM should be separated into a design model and a requirements model which allow for appropriate information sharing for an improved basis of decision and abilities to control the design development by client requirements and a space of solutions. It was furthermore identified that both models should be stored in data repositories defined using the IFC specifications.

A range of challenges has been identified which needs to be overcome before the methodology can be implemented. These challenges relate to contractual issues, education of design team members, technical barriers and considerable new investments. Furthermore attention needs to be directed to an agreement of how to ensure that information available in the integrated BIM is valid and to find an appropriate balance between setting requirements for the design and at the same time allowing for a steady development.

The range of technical barriers to overcome before the methodology can be implemented is identified to be considerable. An analysis of the capabilities of the IFC specifications identified that 70 % of the input and output data required by the five simulation tools can currently be defined using the IFC specifications and that modifications to the specifications is therefore needed. The concepts for requirements definition in the IFC specifications further require that a range of extensions needs to be implemented before constraints in a space of solutions can be defined. Beside improvements to the IFC specifications, it will also require improvements for IFC support of design and simulation tools and development of new solutions for requirements management to ensure proper abilities for interaction with the integrated BIM.

It is believed that if effort is put into improving the technical possibilities, a solution could be reached within a short timeframe that would allow for the methodology of Integrated BIM Design to be used to some extent. Several limitations would, however, apply and proper integration would not be possible unless more significant effort is put into development for a longer time frame. This should ensure that all required information can be defined and that proper interaction with the integrated BIM would be possible.

To support the implementation of IFC, implementation guidelines need to be extended to cover more areas such as information sharing among simulation tools. This is believed to be of great importance because the current IFC specifications are found to be far from consistent in their possibilities to unambiguously define information. To improve the implementation further it would be appropriate if the simulation tools would adjust their internal model structure to be more object oriented. In this way the complex mapping of information between the integrated BIM and the simulation tools could be avoided.

Several challenges therefore exist and significant effort needs to be put into development before implementation of the methodology of Integrated BIM Design can be completed. However, the development can be based on the implementation of BIM design, IFC, and integrated design which is currently taking place in the AEC industry, and the starting point is therefore solidly placed within the present development in the industry.

The literature study identified that the integrated design process as a concept of process integration could successfully re-organise the decision-making process to create an improved result in the design phase. BIM design as a concept of data integration was identified to be able to improve the decision-making process by allowing for efficient information sharing and control. A combination of the two concepts was therefore identified to be an efficient process with an improved result. This demonstrates that a combination of process integration and data integration creates significant advantages when complete integration is achieved.

Although a solution for integration was suggested 15 years ago by [Augenbroe 1992] to achieve improved building performance, and although the inclusion of requirements in data models was suggested even before then by [Gielingh 1988], the use of these opportunities in current design is still lacking. The suggested methodology of Integrated BIM Design is contrary to these suggestions based on concepts for process and data integration already being implemented in the AEC industry. As the advantages of combining the concepts can be highly beneficial for the clients, this should be a call for clients to put more pressure on the industry to ensure that the development is pushed even further to allow for complete integration using a concept such as Integrated BIM Design. In this way the goal of complete integration could hopefully be reached in the AEC industry within a reasonable number of years.

References

- [Augenbroe 1992] *COMBINE: Project Overview*, G. L. M. Augenbroe, Delft University of Technology, 1992. **Included on CD**
- [Bakkmoen et al. 2007] *IFC-projekt Nye Ahus – Evalueringsrapport*, K. I. Bakkmoen et al., C.F. Møller, 2007.
- [Bazjanac 2007] *Impact of the U.S. National Building Information Model Standard (Nbims) on Building Energy Performance Simulation*, V. Bazjanac, Lawrence Berkeley National Laboratory, proceedings at: Building Simulation 2007, Beijing, 2007. **Included on CD**
- [Bazjanac 2006] *Software Interoperability for Building Energy Performance in the United States*, V. Bazjanac, Lawrence Berkeley National Laboratory, presentation at: Building Life Cycle Management Utilising Building Information Models Workshop, IRUSE, Ireland, 2006.
- [Bazjanac 2005] *Building Information Modeling for the e-Lab at LBNL*, V. Bazjanac, Lawrence Berkeley National Laboratory, 2005.
- [Bazjanac 2004] *Ifc Hvac Interface to Energyplus – a Case of Expanded Interoperability for Energy Simulation*, V. Bazjanac, Lawrence Berkeley National Laboratory, proceedings at: SimBuild 2004, Colorado, 2004.
- [Bazjanac 2003] *Improving Building Energy Performance Simulation With Software Interoperability*, V. Bazjanac, Lawrence Berkeley National Laboratory, proceedings at: Building Simulation 2003, Eindhoven, 2003.
- [bips 2006] *3D arbejdsmetode*, bips for Det Digitale Byggeri, 2006.
- [BR95 2007] *Bygningsreglement 1995 inkl. tillæg 1-15*, Erhvervs- og Byggestyrelsen, website reviewed Dec. 2007: <http://www.ebst.dk/br08.dk/BR95/1/54/0>, 2007.
- [BuildingSMART Norway 2007] *Information Delivery Manual - Guide to Components and Development Method*, BuildingSMART Norway, 2007.
- [DAC 2006] *Brugerbehov og brugerdreven innovation i byggeriet – en statusrapport*, Dansk Arkitektur Center for Erhvervs- og Byggestyrelsen, 2006.
- [DOE 2007] *Building Energy Software Tools Directory*, U.S. Department of Energy, website reviewed Dec. 2007: http://www.eere.energy.gov/buildings/tools_directory/, 2007

- [Dong et al. 2007] *A comparative study of the IFC and gbXML informational infrastructures for data exchange in computational design support environments*, B. Dong, K.P. Lam, Y.C. Huang, G. M. Dobbs, Carnegie Mellon University & United Technologies Research Centre, proceedings at: Building Simulation 2007, Beijing, 2007. **Included on CD**
- [DS 418] *DS 418: Calculation of heat loss from buildings*, Dansk Standard, 6. edition, 2003.
- [EBST 2006] *Det Digitale Byggeri (report)*, Erhvervs- og Byggestyrelsen, 2006.
- [EPBD 2003] *Energy Performance of Buildings Directive - Directive 2002/91/EC*, The European Parliament and Council on energy efficiency of buildings, website reviewed Dec. 2007: <http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32002L0091:EN:HTML>, 2003.
- [Erabuild 2008] *Review of the Development and Implementation of IFC compatible BIM*, A. Kiviniemi et al., Erabuild, 2008.
- [Fischer & Gao 2005] *Case Studies on the Implementation & Benefits of 3D and 4D CAD*, M. Fischer & J. Gao, Stanford University, website reviewed Dec. 2007: <http://www.stanford.edu/~gaoju/3D4DFramework/>, 2005.
- [Forester 2001] *Information Modeling*, J. Forester, Marinsoft Inc./AEC3 Ltd., presentation at: 2001 ASHRAE Annual Meeting, 2001.
- [Gielingh 1988] *General AEC Reference Model (GARM)*, W. Gielingh, IBBC-TNO, Construction Informatics Digital Library, paper w78, 1988.
- [Haug 2007] *Experiences in development and use of a digital Building Information Model (BIM) according to IFC standards from the building project of Tromsø University College (HITOS) after completed Full Conceptual Design Phase*, Diderik Haug et al., The Norwegian Agency of Public Construction and Property, 2007.
- [Heiselberg & Brohus 2007] *Application of Sensitivity Analysis in Design of Low Energy Office Buildings – DRAFT*, P. Heiselberg & H. Brohus, Aalborg University, 2007. **Included on CD**
- [Hestnes 2007] *Integrated Design - a must for low energy buildings*, A. G. Hestnes, Norwegian University of Science and Technology, presentation at: Seminar on Integrated Design of Low-Energy Buildings, DTU, 2007. **Included on CD**
- [Hietanen 2002] *BLIS Review: IMSrv*, Jiri Hietanen, BLIS, 2002.

- [IAI 2007a] *IFC2x2 Edition 3 Technical Corrigendum 1*, International Alliance for Interoperability, website reviewed Dec. 2007: <http://www.iai-tech.org/ifc/IFC2x3/TC1/html/>, 2007.
- [IAI 2007b] *International Alliance for Interoperability*, website reviewed Dec. 2007: <http://www.iai-international.org>, 2007.
- [IAI 2007c] *The Model View Definition site*, IAI, website reviewed Dec. 2007: <http://www.blis-project.org/IAI-MVD>, 2007.
- [IAI 2004] *Model Implementation Guide – Version 1.7*, T. Liebich, IAI Model Support Group, 2004.
- [IAI 2001] *BS-8 HVAC Extension Schemata for Modeling and Simulation*, IAI, website reviewed Dec. 2007: http://ce.vtt.fi/iaifcprojects/ShowProjectInfo.jsp?project_id=3&status_id=3, 2001.
- [IAI 2000] *IFC Technical Guide - Release 2x*, T. Liebich & J. Wix, IAI Model Support Group, 2000.
- [Ibbs et al. 2001] *Project Change Management System*, C. W. Ibbs, C. K. Wong & Y. H. Kwak, Journal Of Management In Engineering, July version, 2001.
- [Khemlani 2005] *The CIS/2 Format: Another AEC Interoperability*, L. Khemlani, AECbytes, website reviewed Dec. 2007: <http://www.aecbytes.com/buildingthefuture/2005/CIS2format.html>, 2005.
- [Kiviniemi 2005] *Requirements Management Interface to Building Product Models*, A. Kiviniemi, Ph.D. thesis, Stanford University, 2005.
- [Kvarsvik et al. 2007] *Realizing Industry Use of IFC BIMs*, O. K. Kvarsvik, R. See, J. Wix & J. Hietanen, presentation at: IAI ITM Meeting, Brisbane, 2007. **Included on CD**
- [Larsson 2003] *Solar Low Energy Buildings and the Integrated Design Process - An Introduction*, N. Larsson, IEA, 2003.
- [Löhnert et al. 2003] *Integrated Design Process – Task 23*, G. Löhnert, A. Dalkowski & W. Sutter, International Energy Agency, 2003.
- [Marsh 2006] *CAD Geometry vs Performance Analysis*, A. J. Marsh, Square One, Natural Frequency issue 3, 2006.
- [Marsh 1997] *Performance Analysis and Conceptual Design*, A. J. Marsh, Ph.D. thesis, University of Western Australia, 1997.

- [Mentzel 2003] *Framework zur Bearbeitung des IFC Produktdatenmodells am Beispiel der Wandsegmentierung*, J. Mentzel, FH Karlsruhe, website reviewed Dec. 2007: <http://jan.islandofwaiting.de/dipl>, 2003.
- [Motawa et al. 2007] *An integrated system for change management in construction*, I.A. Motawa, C.J. Anumba, S. Lee & F. Peña-Mora, Automation in Construction, no. 16, 2007.
- [NIBS 2007] *National Building Information Modeling Standard*, National Institute of Building Sciences, 2007.
- [NIST 2004] *Cost Analysis of Inadequate Interoperability in the U.S. Capital Facilities Industry*, Michael P. Gallaher et al., National Institute of Standards and Technology, 2004.
- [OECD.Stat 2007] OECD.Stat (Query: Labour Productivity per Unit Labour Input), OECD, website reviewed Dec. 2007: <http://stats.oecd.org/wbos/default.aspx?queryname=346&querytype=vi>
[ew](http://stats.oecd.org/wbos/default.aspx?queryname=346&querytype=vi), 2007.
- [Oracle 2006] *Oracle announces innovative Collaborative Building Information Management initiative for the construction industry*, Oracle, Press release, 2006.
- [PAR & F.R.I 2002] *Ydelsesbeskrivelser – Byggeri og Planlægning*, Praktiserende Arkitekters Råd & Foreningen af Rådgivende Ingeniører, 2002
- [Petersen & Svendsen 2007] *Method of Integrated design*, S. Petersen & S. Svendsen, Byg.DTU, 2007.
Included on CD
- [Poel 2002] *Integrated Design Process in Practice*, B. Poel, G. de Vries & G. van Cruchten, International Energy Agency, 2002.
- [Regeringen 2003] *Staten som bygherre - vækst og effektivisering i byggeriet*, Regeringen 2003, Erhvervs- & Byggestyrelsen, 2003.
- [Rizos 2007] *Next generation energy simulation tools: Coupling 3D sketching with energy simulation tools*, I. Rizos, Master thesis, University of Strathclyde, 2007.
- [SECOM 2005] *IFCsvr ActiveX Component Object Reference*, SECOM Co., Ltd Intelligent Systems Lab, 2005. **Included on CD**
- [Square One 2007] *ECOTECT: An Overview*, Square One, website reviewed Dec. 2007: <http://www.squ1.com/ecotect>, 2007

- [Struck & Hensen 2007] *On Supporting Design Decisions in Conceptual Design Addressing Specification Uncertainties Using Performance Simulation*, C Struck & J. Hensen, Technische Universiteit Eindhoven, proceedings at: Building Simulation 2007, Beijing, 2007.
- [Svendsen et al. 2007] *Integrated design of sustainable buildings*, S. Svendsen, S. Petersen & C. A. Hviid, BYG.DTU, 2007.
- [Svendsen et al. 2006] *Løsningen til de nye energikrav*, S. Svendsen, S. Petersen & C. A. Hviid, BYG.DTU, Arkitekten nr. 1, 2006. **Included on CD**
- [Transport- og Energiministeriet 2005] *Handlingsplan for en fornyet energispareindsats*, Transport- og Energiministeriet, 2007
- [Trelidal & Johnsen 2005] *3D projektering hos rådgiver*, N. Trelidal & J. S. Johnsen, Bachelor's thesis, BYG.DTU, 2005. **Included on CD**
- [Uddannelsesstyrelsen 1999] *Matematisk formelsamling*, Uddannelsesstyrelsen, 1999.
- [Weisstein 2004] *Polygon Area*, E. W. Weisstein, MathWorld--A Wolfram Web Resource, website reviewed Jan. 2008: <http://mathworld.wolfram.com/PolygonArea.html>, 2004.

List of Tables

Table 1-1. Terms and definitions used in this thesis, partly based on [Erabuild 2008]. VI

Table 3-1. Capabilities of IFC to handle input and output data. Values defined as a percentage of total number of input and output data. 63

Table of Figures

<i>Figure 2-1. Traditional involvement of members in a design team during the design phase.</i>	8
<i>Figure 2-2. Work processes of members in a design team in an integrated design phase.</i>	9
<i>Figure 2-3. The linear procedure, the iterative procedure and the integrated procedure. Illustration inspired by [Löhnert et al. 2003] and [Kiviniemi 2005].</i>	10
<i>Figure 2-4. Iterations as a provider for problem-orientated analyses of design alternatives and optimisation [Löhnert et al. 2003].</i>	11
<i>Figure 2-5. Potential use of simulation tools in IDP. Based on needs for tools capable of making overall analyses in the early part of the design phase and tools capable of detailed analyses later in the phase.</i>	14
<i>Figure 2-6. The total integrated design process [Svendsen et al. 2007].</i>	15
<i>Figure 2-7. Results of parameter analysis in iDbuild [Petersen & Svendsen 2007]. In this case the window height has been analysed for three types of windows. On the right side, energy use (blue) and the daylight factors (yellow) are illustrated for each window analysed.</i>	16
<i>Figure 2-8. Labour Productivity per Unit Labour Input per Hour in construction and non-farming industry in Denmark based on information from [OECD.Stat 2007].</i>	18
<i>Figure 2-9. An object and some of its properties.</i>	19
<i>Figure 2-10. Concept of integrated BIM. Illustration originally made by Norwegian Building Research Institute, Olof Granlund, NBLN University of California and Stanford University.</i>	21
<i>Figure 2-11. Composition of the IFC 2x3 specifications [IAI 2007a].</i>	23
<i>Figure 2-12. Hierarchical structure of typical objects defined in IFC.</i>	23
<i>Figure 2-13. Hierarchy chart of building elements defined by EXPRESS-G notation [IAI 2004].</i>	24
<i>Figure 2-14. The composition of a wall object in IFC.</i>	24
<i>Figure 2-15. Information sharing using a model server. Illustration originally made by [Trelidal & Johnsen 2005].</i>	26
<i>Figure 2-16. The layout of Solibri Model Checker with parameterised rule sets (constraint model) on the left and a BIM to be checked on the right [Kvarsvik et al. 2007].</i>	27
<i>Figure 2-17. Example of parameterised rules, in this case for compliance with accessibility requirements.</i>	27
<i>Figure 2-18. Concept for co-existence of a requirements model and a design model [Kiviniemi 2005]. The figure mentions concepts of SPT, SPI and cascading requirements which will not be included in this thesis.</i>	28
<i>Figure 2-19. Suggested requirements management interface for ArchiCAD to the requirements model [Kiviniemi 2005].</i>	30
<i>Figure 2-20. Illustration of how BIM design can improve IDP by interoperability for a better decision making process and information management for continuous development of the design.</i>	33
<i>Figure 3-1. The MiniOffice project seen from the north west corner.</i>	36
<i>Figure 3-2. The MiniOffice project seen from the south east corner with no roof. The overhang of the windows in the southern façade is indicated in the left part of the figure.</i>	36
<i>Figure 3-3. Input data requirements for systems in iDbuild.</i>	38
<i>Figure 3-4. Results of selected average annual performance values from simulation in iDbuild.</i>	38

<i>Figure 3-5. Layout of Riuska and list of loads attached to an office in the MiniOffice project.</i>	<i>39</i>
<i>Figure 3-6. Energy consumption results from simulation in Riuska. The set point configuration in Riuska does not allow for the heating system to be turned off outside the heating season, hence the large energy consumption of heating.</i>	<i>39</i>
<i>Figure 3-7. Layout of BSim with a list of loads and systems attached to a zone of offices on the second floor in the MiniOffice project.</i>	<i>40</i>
<i>Figure 3-8. Simulated thermal conditions in the zone of offices. Here temperature, inlet temperature and air change rate is presented.</i>	<i>40</i>
<i>Figure 3-9. Layout of Be06 with a table for ventilation related information for the MiniOffice project.</i>	<i>41</i>
<i>Figure 3-10. Calculated overall energy consumption (green) and key parameters of various energy contributions in the MiniOffice project.</i>	<i>41</i>
<i>Figure 3-11. Modelling layout of Flovent. In the figure, the office on the second floor in the south east corner of the MiniOffice project has been modelled.</i>	<i>42</i>
<i>Figure 3-12. Results from the CFD simulation of temperature distribution in the office on the cooling design day.</i>	<i>42</i>
<i>Figure 3-13. Required development of 3D object oriented model in the Digital Construction project [bips 2006]. The steps 0-6 describe the model development from the client requirements definition phase to the end of the construction phase.</i>	<i>44</i>
<i>Figure 3-14. The definition of 3D geometry of a wall using a Swept Solid representation [IAI 2004]. .</i>	<i>45</i>
<i>Figure 3-15. The concept of IfcLocalPlacement and its relations to other local coordinate systems [Mentzel 2003].</i>	<i>45</i>
<i>Figure 3-16. The ThermalLoadDesignCriteria property set and its defined content. Additionally the IFC specifications define that such a property set can be assigned to any IfcSpatialStructureElement such as a building or a space and further describe the type of values to assign to each property.</i>	<i>47</i>
<i>Figure 3-17. Assignment of property sets to an object either directly or via the type definition of the object in IFC [IAI 2004].</i>	<i>47</i>
<i>Figure 3-18. Concept of modelling HVAC objects in IFC [Bazjanac 2004, ©Jim Forester].</i>	<i>48</i>
<i>Figure 3-19. Concept of creating a container of objects to represent a shared object [Bazjanac 2004].</i>	<i>48</i>
<i>Figure 3-20. Concept of using IfcDistributionPorts and assigning fluid flow properties.</i>	<i>49</i>
<i>Figure 3-21. Assignment of a material layer set and materials to e.g. a wall or slab in IFC [IAI 2004].</i>	<i>54</i>
<i>Figure 3-22. Concept of relating an opening element to both the object voided (e.g. a wall) and the object filling the opening (e.g. a window) [IAI 2004].</i>	<i>54</i>
<i>Figure 3-23. Assignment of lining and panel properties to IfcWindow and IfcDoor [IAI 2004].</i>	<i>55</i>
<i>Figure 3-24. Evaluation of space of solutions in relation to design alternatives in IBD should be possible for all design alternatives. Illustration inspired by [Kiviniemi 2005].</i>	<i>66</i>
<i>Figure 3-25. Concept of direct and indirect linking between requirements and design model [Kiviniemi 2005].</i>	<i>67</i>
<i>Figure 3-26. Example of a requirements object for client requirements from [Kiviniemi 2005].</i>	<i>68</i>
<i>Figure 3-27. Potential automatic compliance checking using Solibri Model Checker as the engine for checking.</i>	<i>70</i>

<i>Figure 4-2. Example of Visual Basic script which will use the IFCsvr to display the name of each wall in the file "IfcFile.ifc"</i>	<i>75</i>
<i>Figure 4-3. Possibilities in ArchiCAD to define information to be exported to an IFC file.</i>	<i>77</i>
<i>Figure 4-4. Property set and current information being exported from Riuska.</i>	<i>77</i>
<i>Figure 4-5. Starting point for IFC interface for Be06.</i>	<i>79</i>
<i>Figure 4-6. Geometrical definition of a slab using a Swept Solid representation. The local coordinate system for the objects is indicated with its axes [IAI 2004].....</i>	<i>80</i>
<i>Figure 4-7. The definition of the placement for an object with the hierarchical structure of IfcLocalPlacement.</i>	<i>80</i>
<i>Figure 4-8. Example of transposition caused by the IfcAxis2Placement3D illustrated in the lower right corner. It requires three steps for such a transposition.....</i>	<i>81</i>
<i>Figure 4-9. Imported information about spaces in the IFC interface.</i>	<i>83</i>
<i>Figure 4-10. Information defined for a SweptSolid representation for a space. This representation does not include voids.....</i>	<i>84</i>
<i>Figure 4-11. The swept area of a space which includes an inner curve to define the void [IAI 2004]... ..</i>	<i>84</i>
<i>Figure 4-12. Imported information about walls in the IFC interface.....</i>	<i>86</i>
<i>Figure 4-13. Information available on the surface geometry of a space boundary in IFC.</i>	<i>88</i>
<i>Figure 4-14. Normal of the surface of a space boundary.....</i>	<i>88</i>
<i>Figure 4-15. Determination of the placement of spaces along the wall axis.</i>	<i>89</i>
<i>Figure 4-17. Imported information about windows and doors in the IFC interface.....</i>	<i>91</i>
<i>Figure 4-18. Possible shape of the ground slab with external walls along the perimeter.</i>	<i>93</i>
<i>Figure 4-19. Imported information about slabs in the IFC interface.</i>	<i>93</i>
<i>Figure 4-20. Summation of information imported by the IFC interface for Be06.</i>	<i>94</i>
<i>Figure 5-1. Information for the decision-making process of IBD provided by a design model.</i>	<i>97</i>
<i>Figure 5-2. Information sharing for a requirements model in the decision-making process in IBD.....</i>	<i>99</i>
<i>Figure 5-3. A methodology to define Integrated BIM Design.</i>	<i>100</i>
<i>Figure 5-4. Model of Building 118 at DTU created in AutoCAD Architecture 2008. Plain drawing illustrates the layout on the first floor of the building. North would be upwards on this drawing....</i>	<i>101</i>
<i>Figure 5-5. Four selected design steps in IBD.</i>	<i>101</i>
<i>Figure 5-6. Defining client requirements for small offices in building 118.....</i>	<i>102</i>
<i>Figure 5-7. Parameter variation in iDbuild of U-value in facade for small offices along the southern facade. The variation is from 0.15 W/m²K to 0.35 W/m²K.</i>	<i>103</i>
<i>Figure 5-8. Parameter variation of window height of offices along the southern facade. The variation is from 1.0 to 2.5 m.</i>	<i>104</i>
<i>Figure 5-9. The first iteration in IBD for optimisation of room design.</i>	<i>104</i>
<i>Figure 5-10. Parameter variation of window height of offices along the northern facade. The variation is from 1.0 to 2.5 m.</i>	<i>104</i>
<i>Figure 5-12. Compliance checking in Solibri Model Checker. Here it is identified that there is less spaces than required for the office type orientated to the south.</i>	<i>106</i>
<i>Figure 5-13. One design alternative imported into Riuska and supplemented with thermal and performance related information.</i>	<i>107</i>
<i>Figure 5-14. Comparison of simulation results in Riuska of energy consumption from building with window height of 1.7 m and 2.0 m.....</i>	<i>108</i>

<i>Figure 5-15. Import of design alternative with low window height in Be06 via IFC interface and the resulting energy frame calculated by Be06. The figure shows information imported for windows and doors in the building.....</i>	<i>108</i>
<i>Figure 5-16. Third iteration in IBD for optimisation of window height and ventilation system.....</i>	<i>108</i>
<i>Figure 5-17. Compliance checking in Solibri Model Checker. Here it is identified that the air flow rate for three rooms facing south are above the required limit of 42 l/s.</i>	<i>109</i>
<i>Figure 5-18. Compliance checking in Solibri Model Checker. Here it is identified that some walls in the façade does not fulfil the requirements in the space of solutions.....</i>	<i>110</i>
<i>Figure 5-19. Fourth iteration in IBD for detailed façade design.</i>	<i>110</i>

Appendices

Appendix 1	Performance-decisive parameters of a room
Appendix 2	Input/output data chart - iDbuild
Appendix 3	Input/output data chart - Riuska
Appendix 4	Input/output data chart - BSim
Appendix 5	Input/output data chart - Be06
Appendix 6	Input/output data chart - Flovent
Appendix 7	Identified IFC attributes for simulation tools
Appendix 8	Definition of time series
Appendix 9	IFC capabilities - iDbuild
Appendix 10	IFC capabilities - Riuska
Appendix 11	IFC capabilities - BSim
Appendix 12	IFC capabilities - Be06
Appendix 13	IFC capabilities - Flovent
Appendix 14	Identified shortcomings of the IFC Specifications
Appendix 15	Source code for IFC Interface for Be06
Appendix 16	Measurements standard for Be06
Appendix 17	AutoCAD Architecture 2008 and gross + net area of spaces
Appendix 18	Creation of Rule Set in Solibri Model Checker

Appendix 1

Performance-decisive parameters of a room

Space of Solutions is a concept currently being developed at DTU. The hypothesis is that by starting the design process by optimising the layout and façade of rooms the aim of better building performance can be achieved. A set of performance-decisive parameters have been identified in this regard which are intended to constitute part of the space of solutions for the remaining design process. The parameters are listed in Table 1.

Table 1. Performance-decisive parameters of a room in a building [Petersen & Svendsen 2007].

Geometry	Comments
Room depth	Internal measure
Room width	Internal measure
Room height	Internal measure
Overhang and side fins	
Building components	Comments
Window	Defined by U-value, g and LT, and the placement and size of the window
U-value of opaque constructions	
Thermal capacity of constructions	The thermal mass of the building itself.
Thermal capacity of interior	Other thermal mass besides the building itself, like furniture.
Systems	Comments
Internal loads	People load and load from equipment.
Lighting	Internal heat load due to lighting.
Ventilation	Infiltration, natural/mechanical air change, heat exchanger efficiency.
Controls	Comments
Set points summer	Depend on design goals in terms of indoor environment.
Set points winter	Depend on design goals in terms of indoor environment.
Energy data	Comments
COP heating	Coefficient of performance of the heating system
COP cooling	Coefficient of performance of the cooling system
Solar water heating	Amount of kWh/year heat generated from solar power
Photovoltaic	Amount of kWh/year electricity generated from solar power
SFP	Specific fan power for mech. ventilation system
Installations	Energy use for e.g. pumps and other building services

Appendix 2 – Input/output data chart – iDbuild

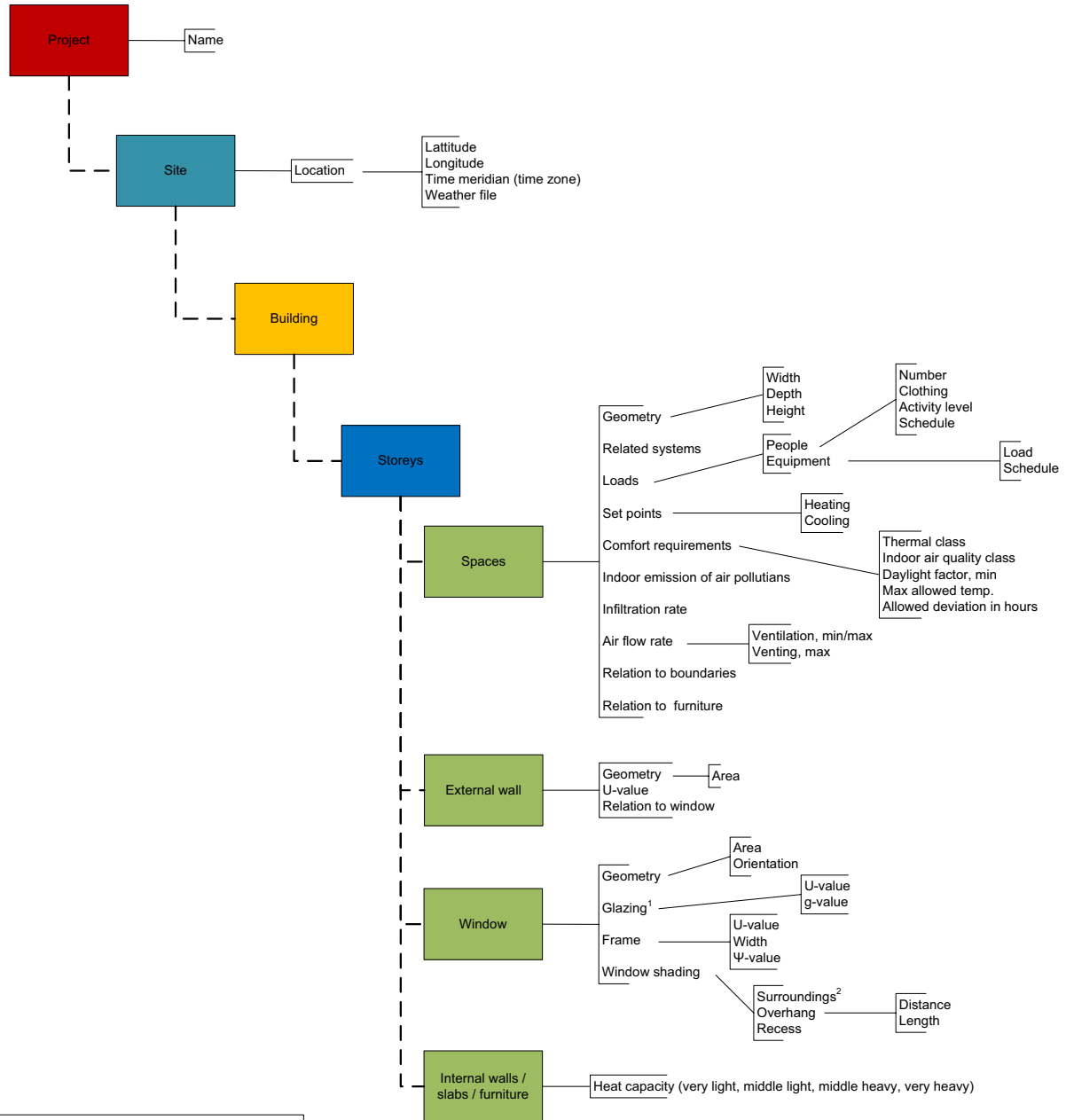
iDbuild is only capable of simulating one room with one side facing the facade in which only one window can be placed. The indata/outdata analysis will analyse such a situation.

Signatures

Required input data for MiniOffice

Additional input data not used in MiniOffice

Architectural objects



Schedule setup in iDbuild

Controlled in three levels:
Weeks in a year - Days in a week - Hours in a Day
on/off - on/off - on/off

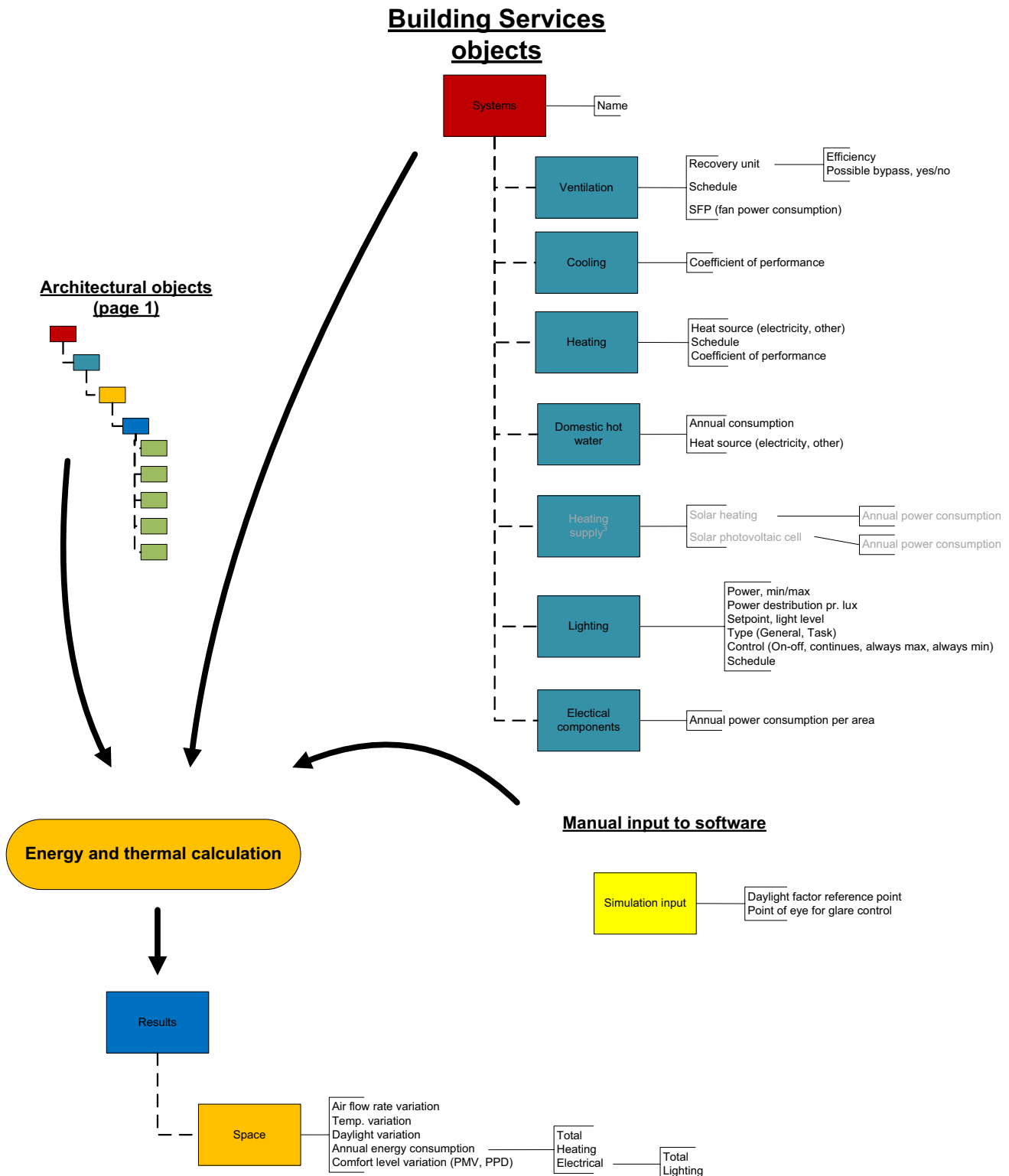
¹The amount of glazing properties is very comprehensive and hence only key properties are included.
²Shading from surroundings is described by a complex relation between azimuth and elevation and is hence left out in this case.

Appendix 2 – Input/output data chart – iDbuild

Signatures

Required input data for MiniOffice

Additional input data not used in MiniOffice



²It is required to distinguish between installed and movable lighting. In this case, it is assumed that all general lighting is installed.

³The properties need for the heating supply components are to detailed for the scope of this thesis. Hence only key properties have been selected.

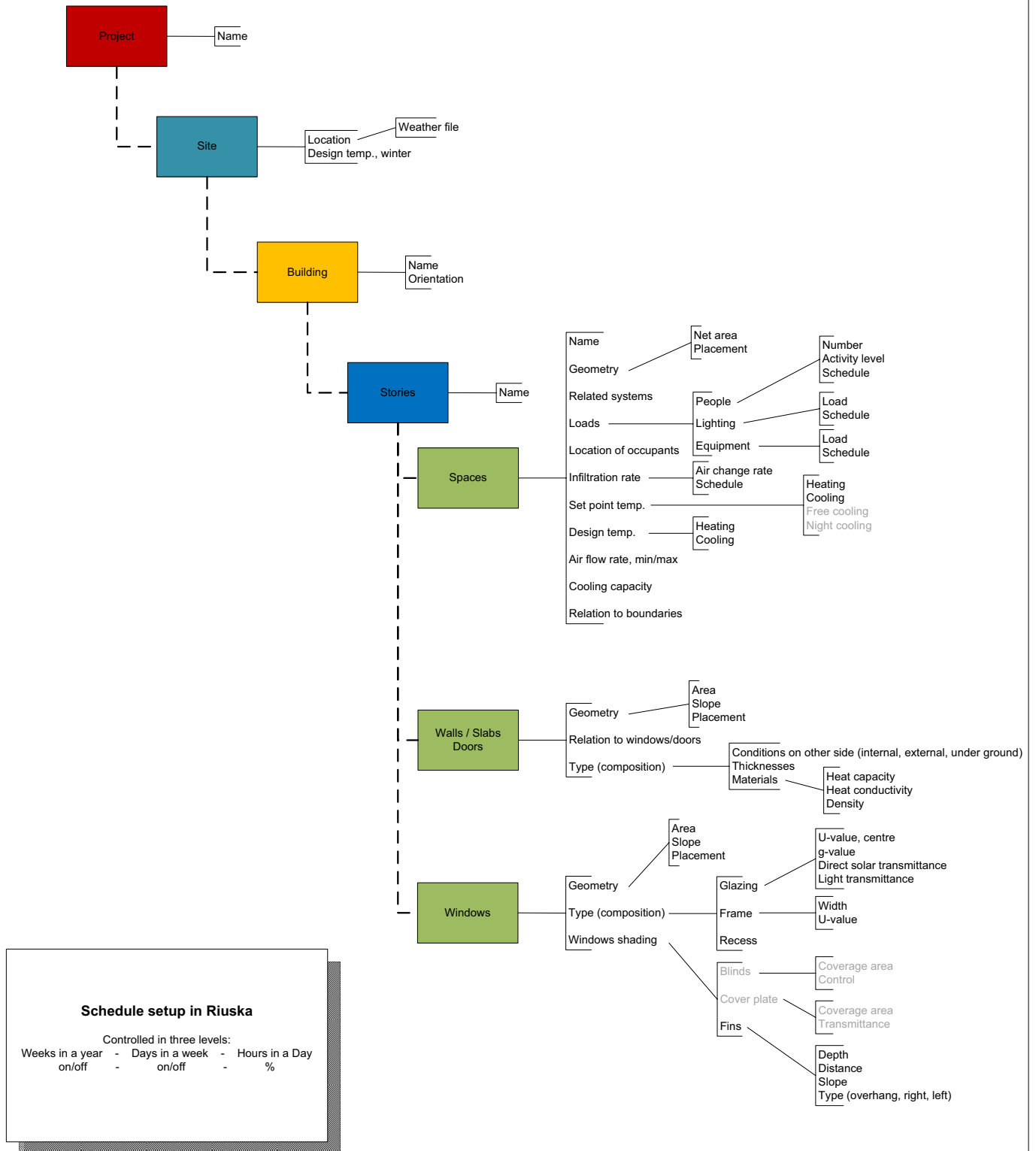
Appendix 3 - Input/output data chart - Riuska

Signatures

Required input data for MiniOffice

Additional input data not used in MiniOffice

Architectural objects

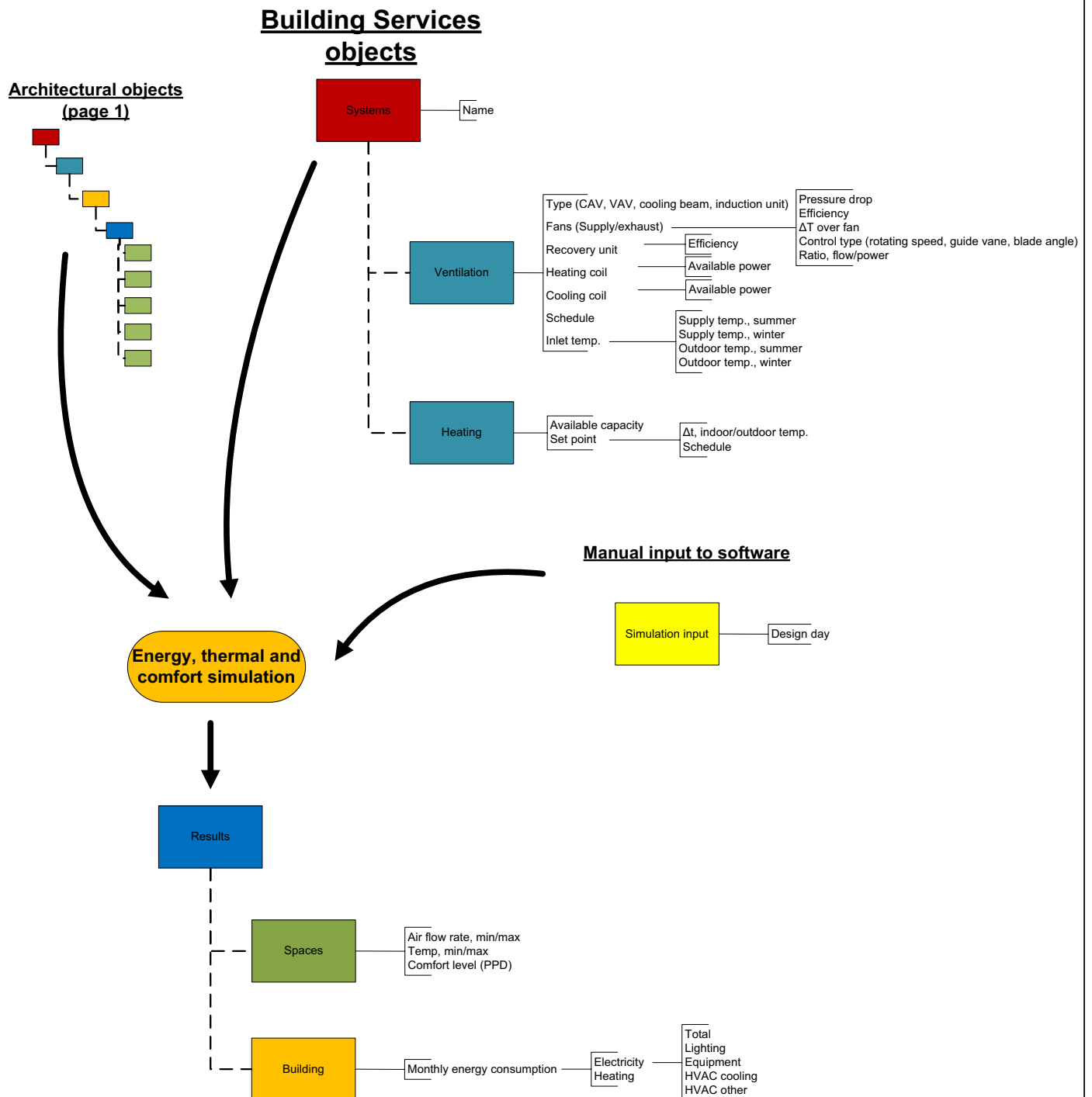


Appendix 3 - Input/output data chart - Riiska

Signatures

Required input data for MiniOffice

Additional input data not used in MiniOffice



Appendix 4 - Input/output data chart - BSim

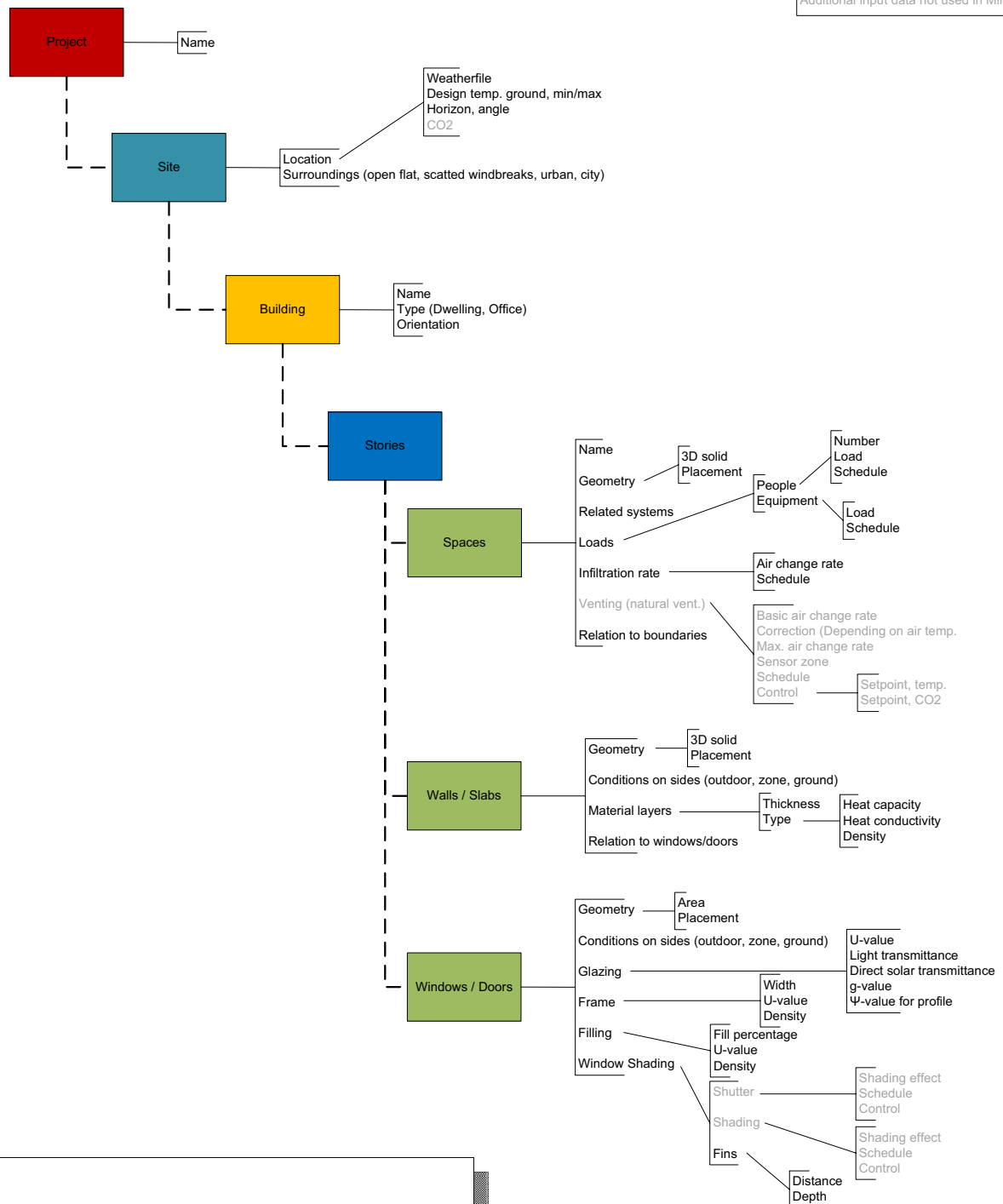
Moisture calculation not included. Sun distributions calculation not included.
Values considered relevant to the calculation engine of BSim only has been left out. This could e.g. be the ratio of how much heat is distributed via convection and radiation from heat sources.

Signitures

Required input data for MiniOffice

Additional input data not used in MiniOffice

Architectural objects



Schedule setup in BSim

Controlled in five levels:

Months in a year - Weeks in a year - Days in a week - Hours in a Day - Hours in a Day
on/off - on/off - on/off - on/off - %

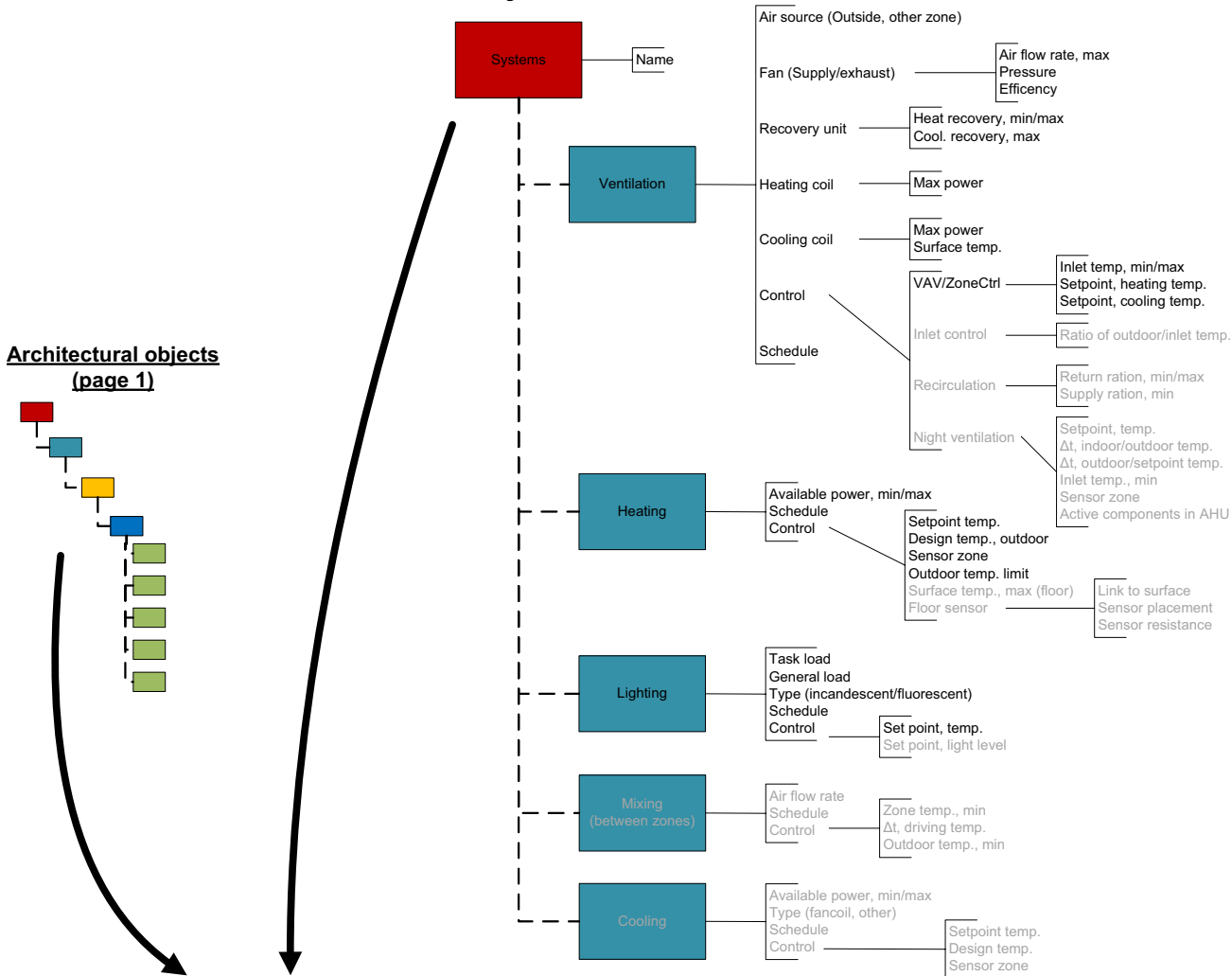
Appendix 4 - Input/output data chart - BSim

Signitures

Required input data for MiniOffice

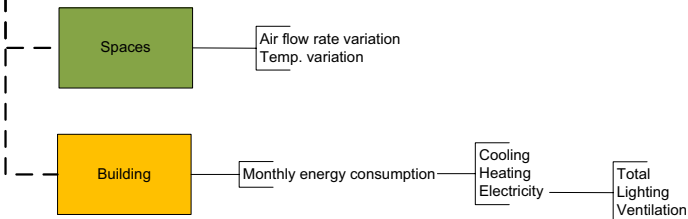
Additional input data not used in MiniOffice

Building Services objects

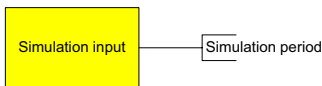


Energy and thermal simulation

Results



Manual input to software



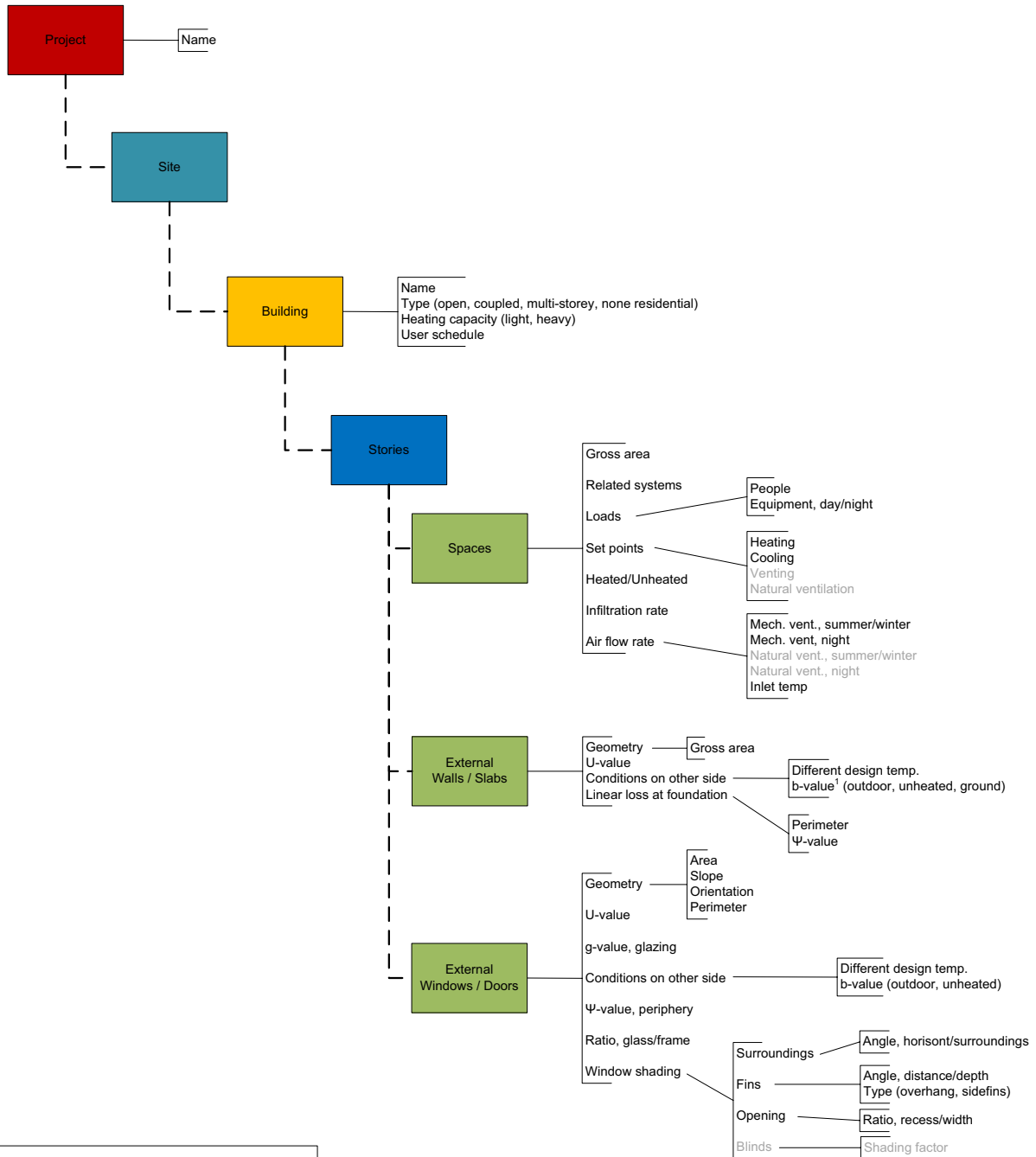
Appendix 5 - Input/output data chart – Be06

Signitures

Required input data for MiniOffice

Additional input data not used in MiniOffice

Architectural objects



Schedule setup in Be06

Controlled in two levels:
Hours in a week - Hours in a Day
no. - no.

Or

Controlled by:
Day / Night
Summer / Winter

¹In case of floor heating, the b-value depends on the temp. in the floorpipes, the indoor temp., and the outdoor temp. This has been discarded in this case.

Appendix 5 - Input/output data chart – Be06

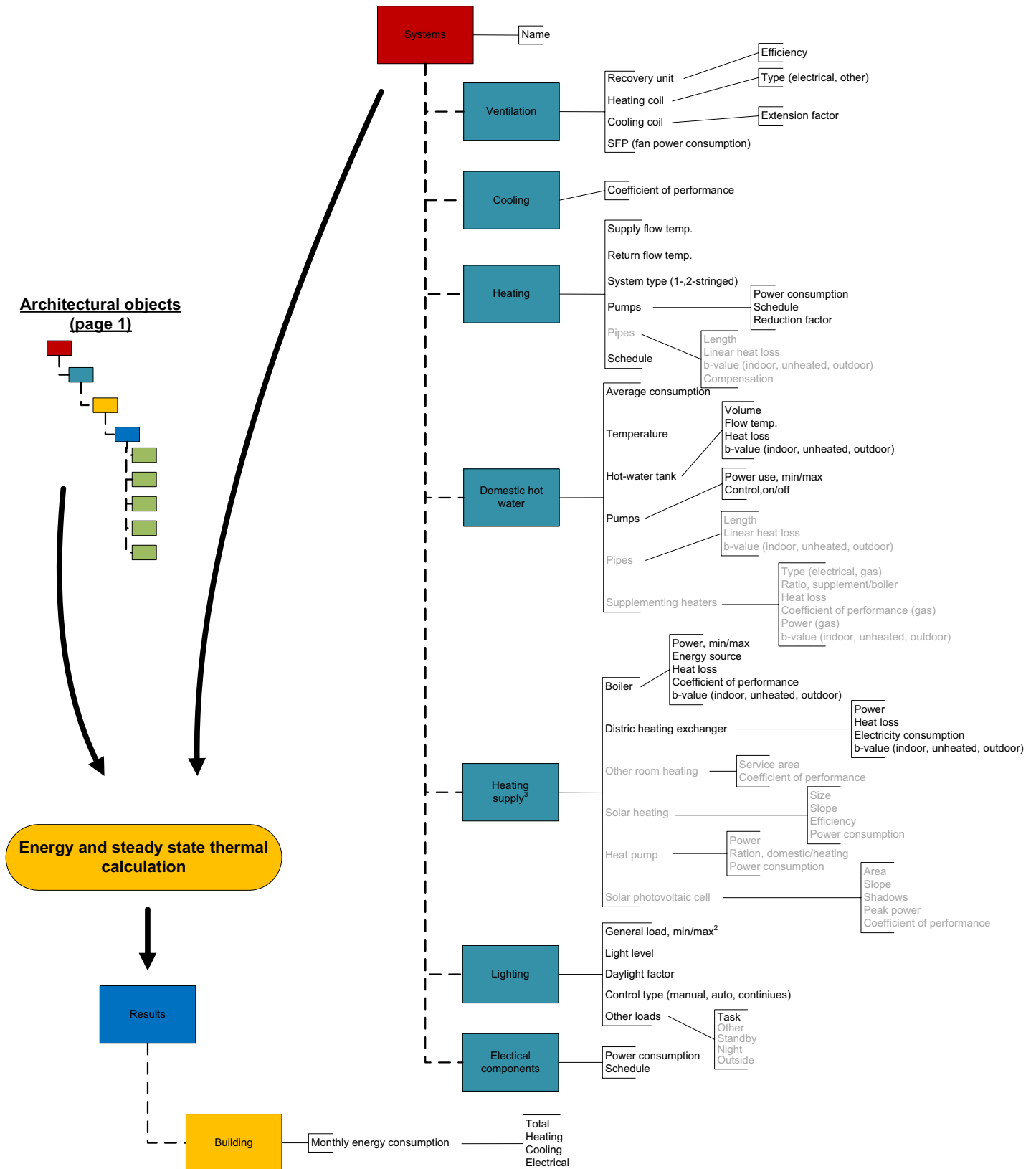
Signitures

Required input data for MiniOffice

Additional input data not used in MiniOffice

Building Services objects

Architectural objects (page 1)

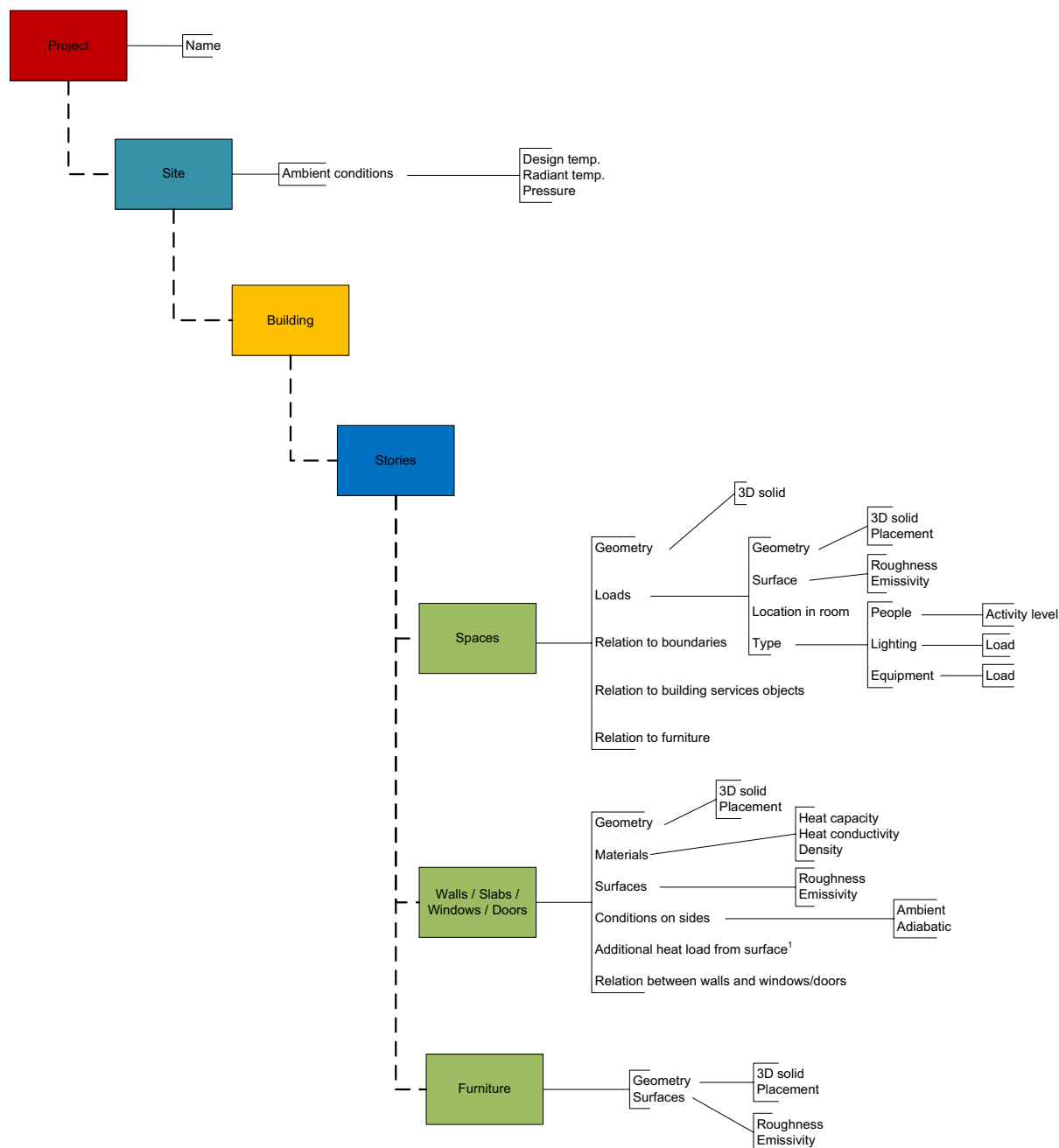


²It is required to distinguish between installed and movable lighting. In this case, it is assumed that all general lighting is installed.
³The properties need for the heating supply components are detailed for the scope of this thesis. Hence only key properties have been selected.

Appendix 6 - Input/output data chart - Flovent

Single room CFD simulation considered for design day calculation.
Simulation of pollution sources has not been included in this chart.

Architectural objects



¹To limit calculation time of CFD simulations, solar distribution is typical applied to surfaces as an area of the surface with an additional heat load. Flovent does allow for the solar distribution to be calculated, but this has been discarded in this case.

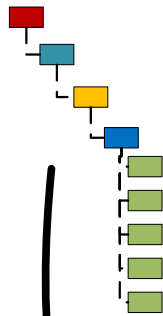
Appendix 6 - Input/output data chart - Flovent

Signatures

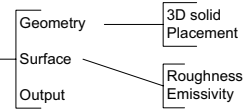
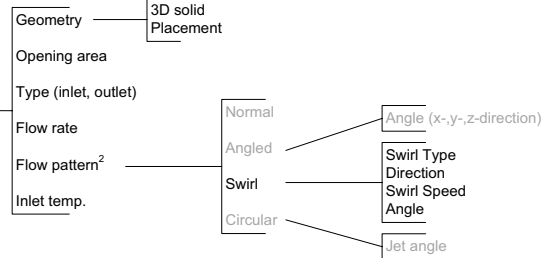
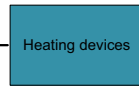
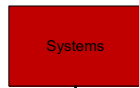
Required input data for MiniOffice

Additional input data not used in MiniOffice

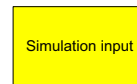
Architectural objects (page 1)



Building Services objects

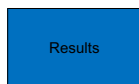


Manual input to software



Grid size in room
Required iterations

Computational Fluid
Dynamics simulation

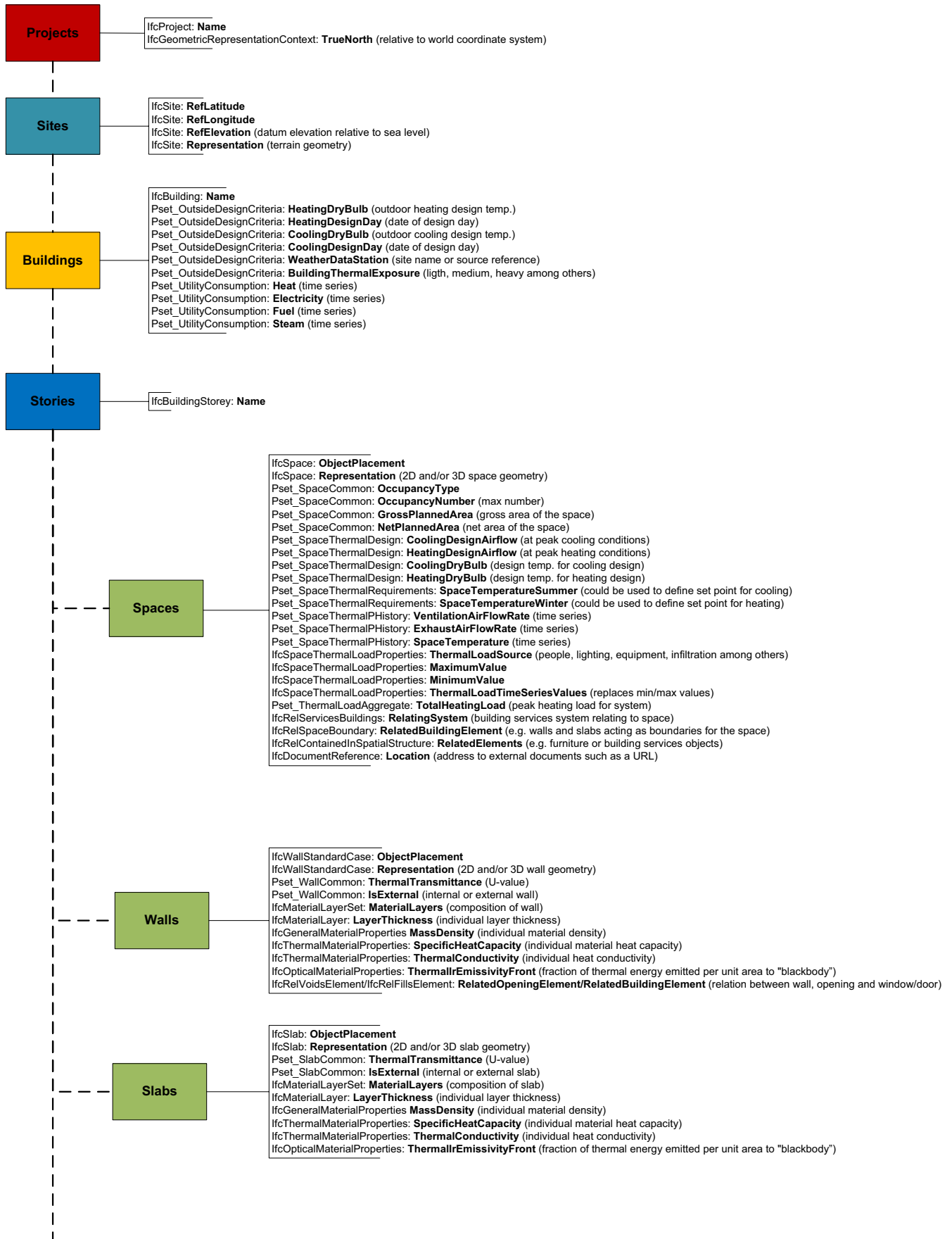


Reference to temp. distribution
Reference to air velocity distribution
Reference to air quality distribution
Comfort level in point (PMV, PPD)

²Flovent allows for the the selection of four simple representations of air terminal devices. In case of other or more complex devices, the devices has to be modelled manually. This requires a detailed description of the flow pattern provided by the supplier.

Appendix 7 – Identified IFC attributes for simulation tools

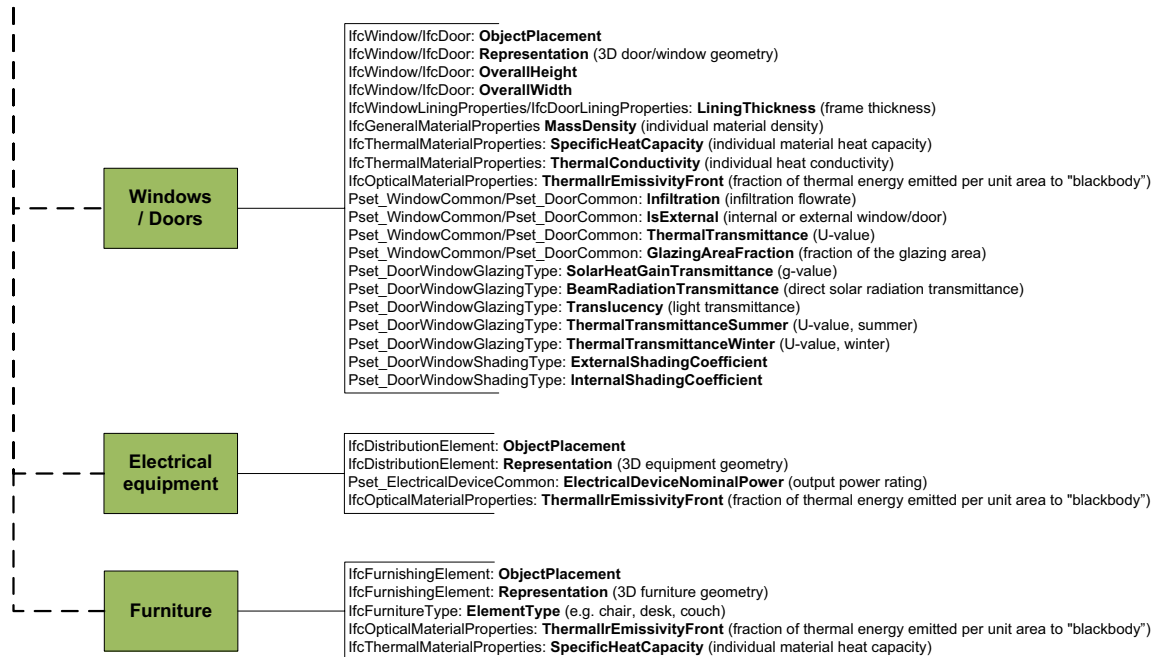
Architectural objects



continues on page 2

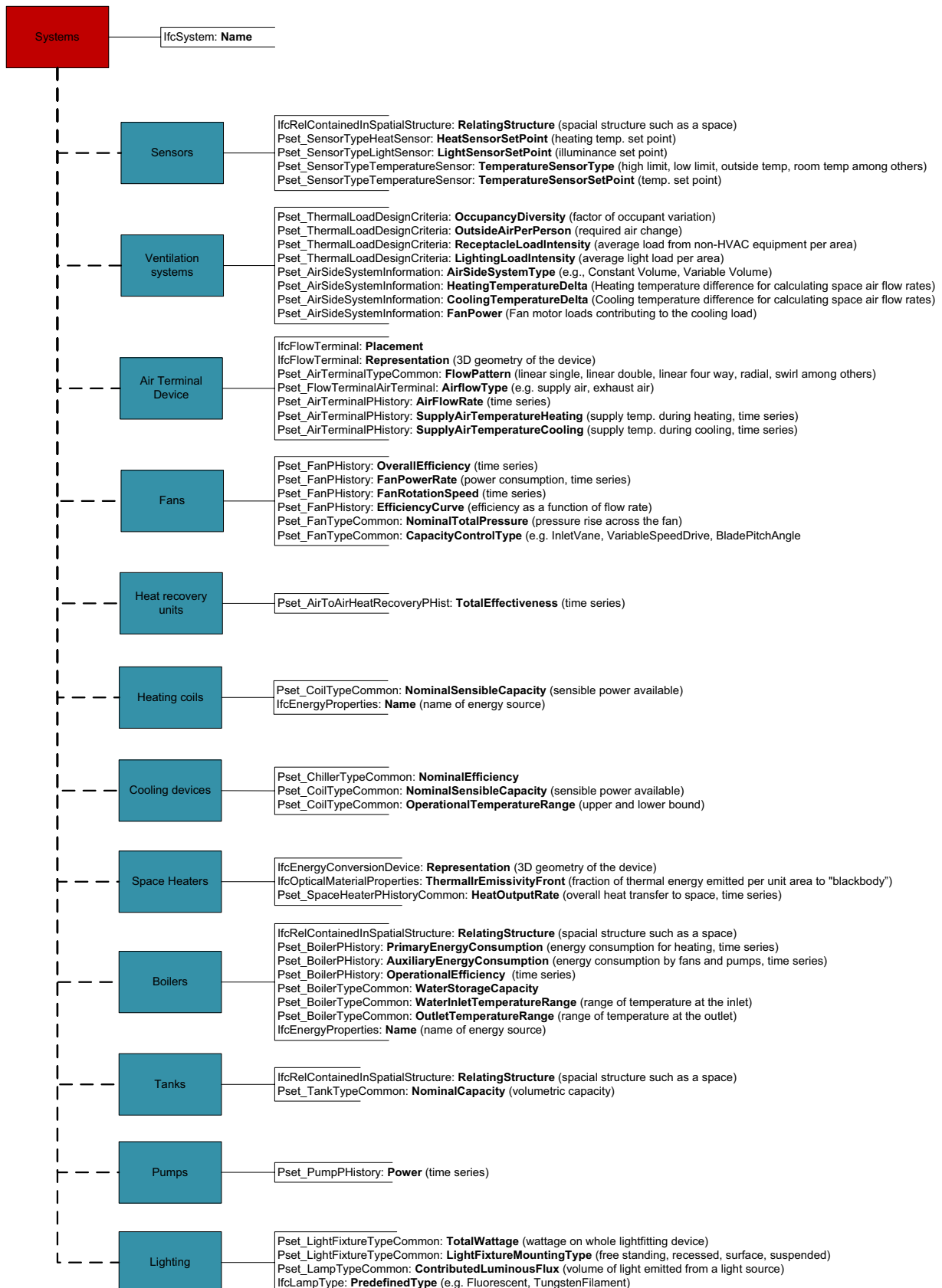
Appendix 7 – Identified IFC attributes for simulation tools

Architectural objects (continued)



Appendix 7 – Identified IFC attributes for simulation tools

Building Services objects



Appendix 8

Definition of time series

The definition of a time series in IFC as described by [IAI 2007a] is: “A time series is a set of a time-stamped data entries. It allows a natural association of data collected over intervals of time. Time series can be regular or irregular. In regular time series data arrive predictably at predefined intervals. In irregular time series some or all time stamps do not follow a repetitive pattern and unpredictable bursts of data may arrive at unspecified points in time.” Time series are hereby able to capture the variation in performance parameter which are required by the simulation tools analysed in this thesis.

A time series is among others defined by a start and end time, a definition of the origin of the data (e.g. design, simulated, measured) and a regular or irregular time series containing a list of values and a description of the time to which the values relate. The entire definition of a time series is illustrated in Figure 1.

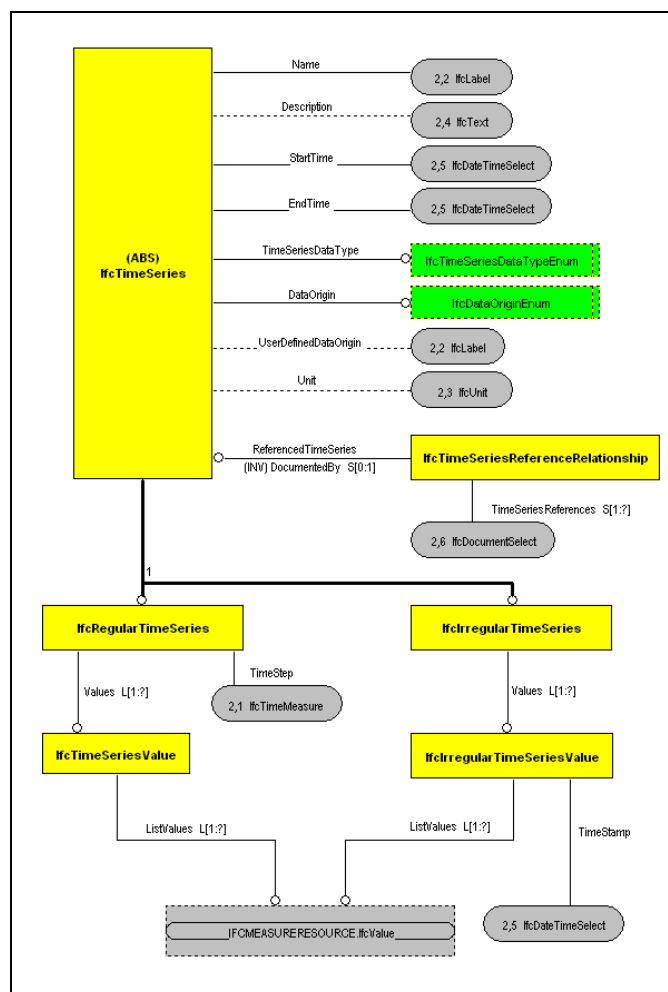


Figure 1. Definition of time series [IAI 2007a].

Some examples of time series are illustrated below to demonstrate how values can be defined.

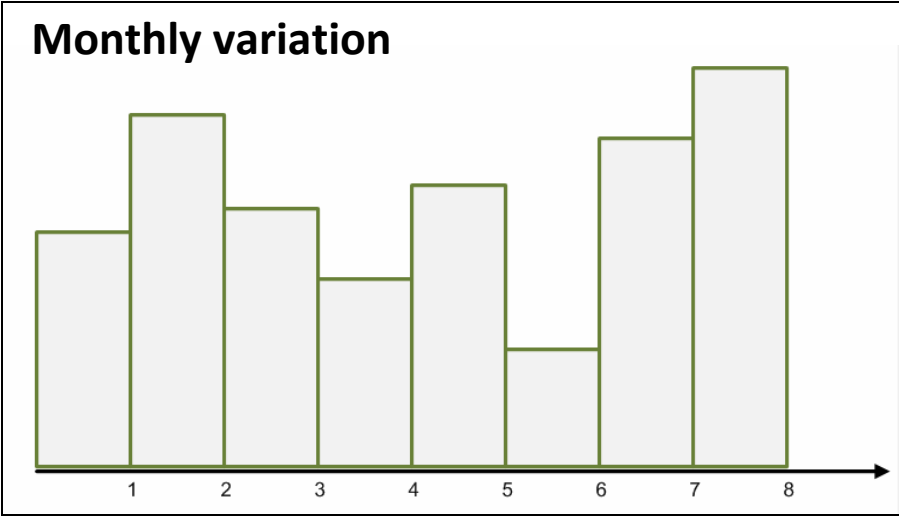


Figure 2. 8 regular time steps to represent e.g. a monthly values.

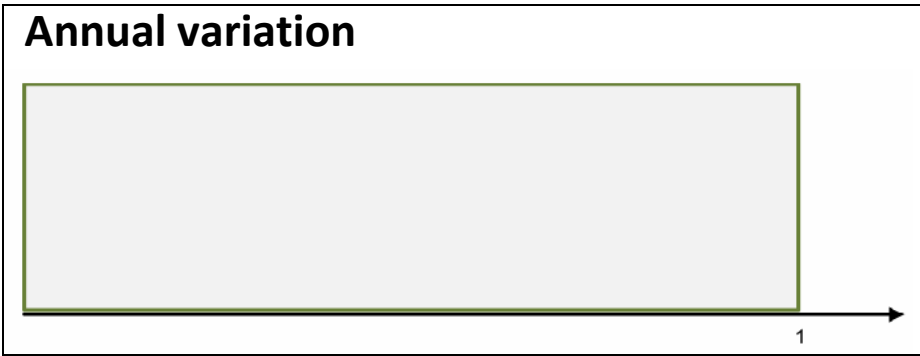


Figure 3. 1 time step to represent e.g. an annual value.

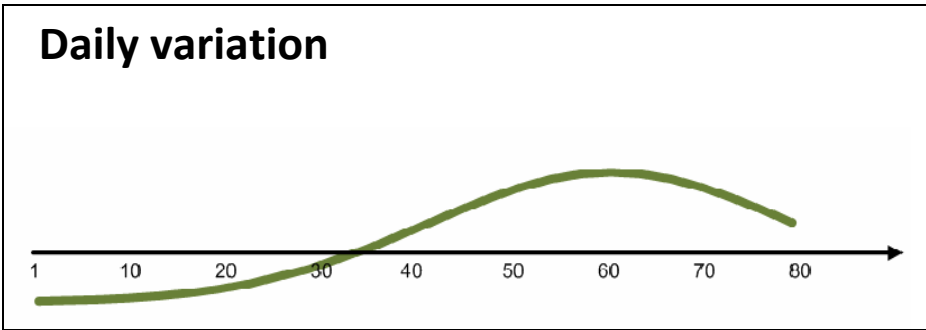


Figure 4. Figure 5. Many time step to represent e.g. daily values.

Appendix 9 – IFC capabilities – iDbuild

iDbuild is only capable of simulating one room with one side facing the facade in which only one window can be placed. The indata/outdata analysis will analyse such a situation.

Signatures

Information defined in IFC

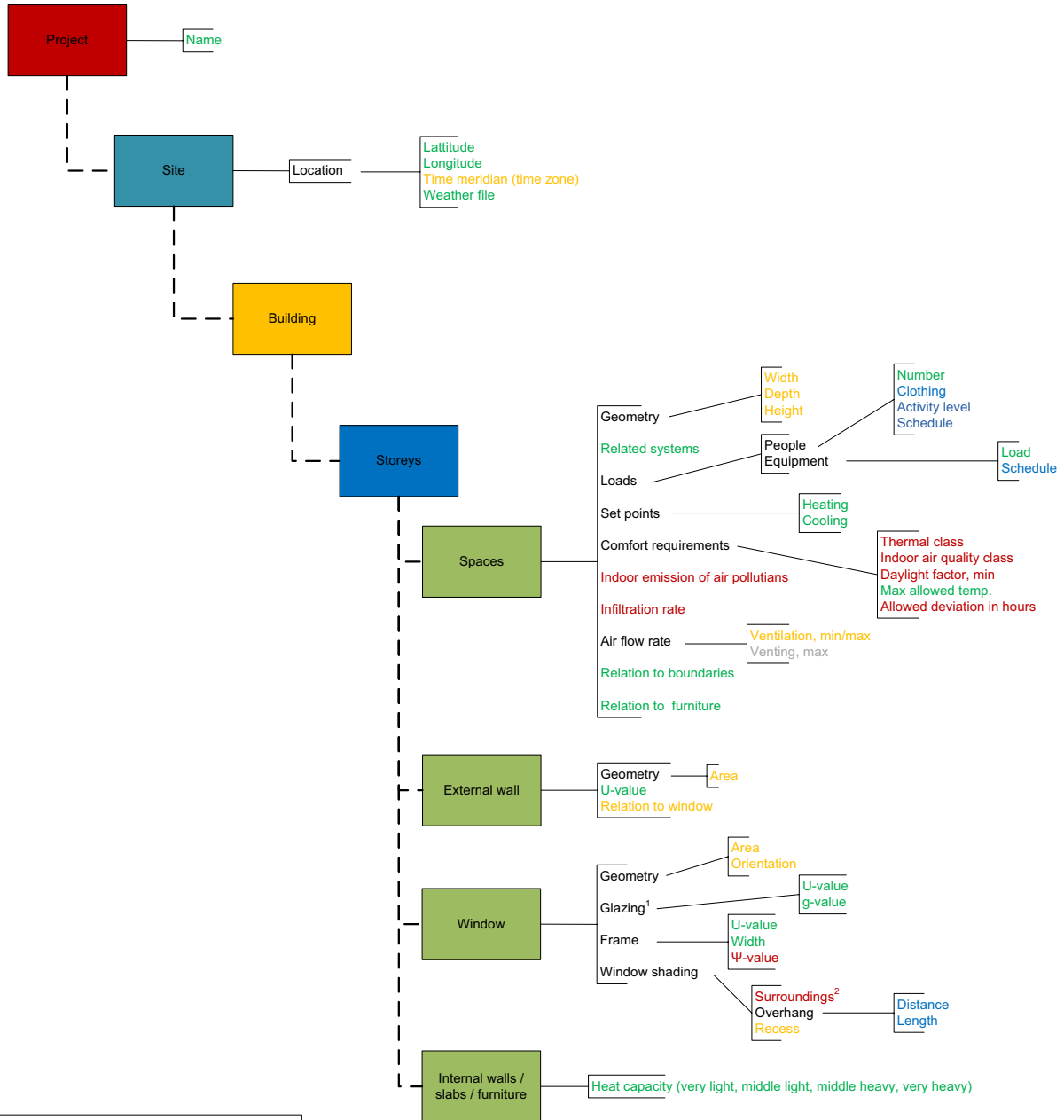
Information derived from IFC

Information unable to be define in IFC

Information sometimes defined in IFC

Ignored information

Architectural objects



Schedule setup in iDbuild

Controlled in three levels:
 Weeks in a year - Days in a week - Hours in a Day
 on/off - on/off - on/off

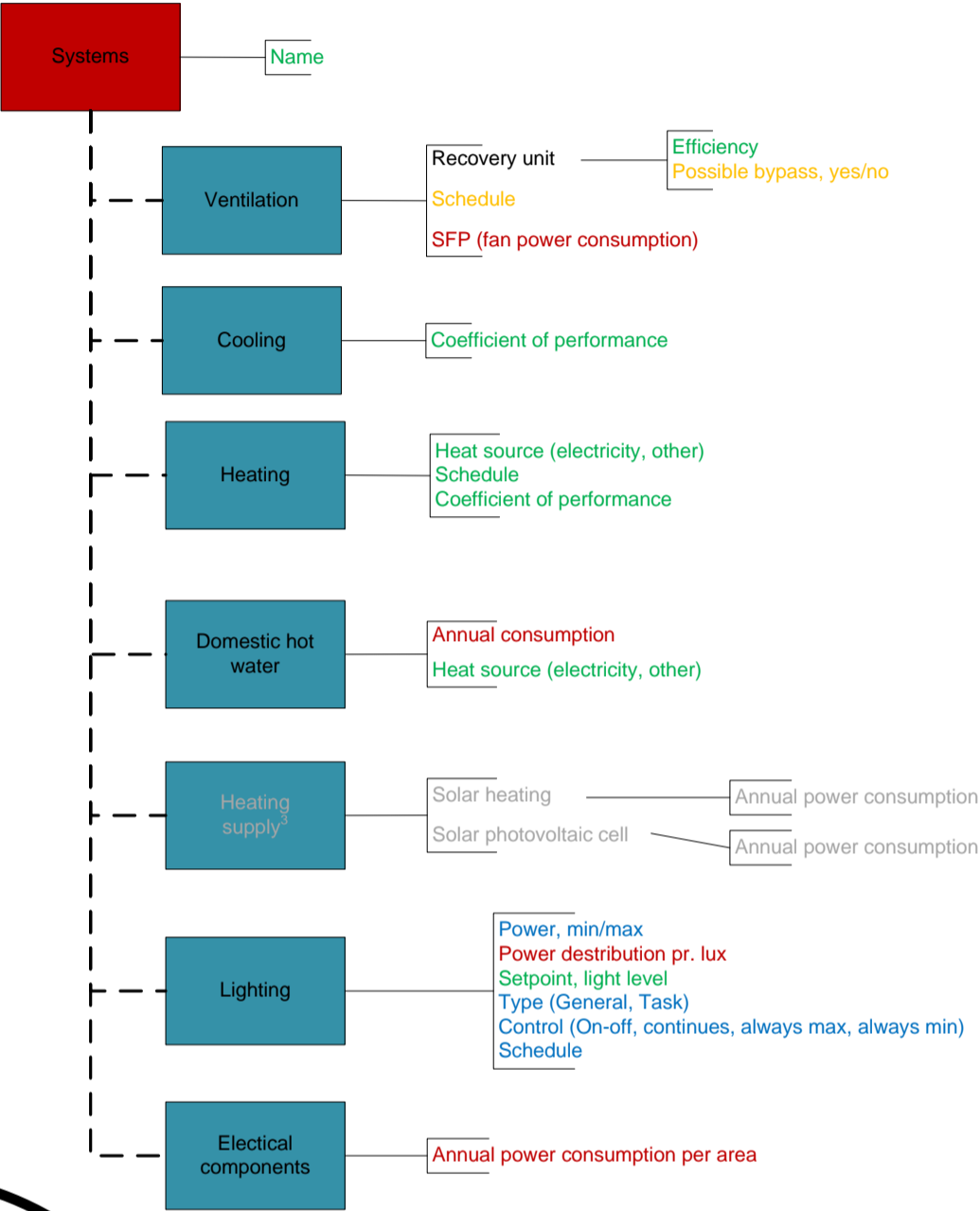
¹The amount of glazing properties is very comprehensive and hence only key properties are included.

²Shading from surroundings is described by a complex relation between azimuth and elevation and is hence left out in this case.

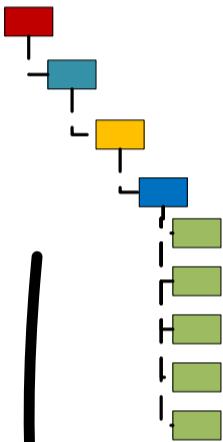
Appendix 9 – IFC capabilities - iDbuild

Signatures
Information defined in IFC
Information derived from IFC
Information unable to be define in IFC
Information sometimes defined in IFC
Ignored information

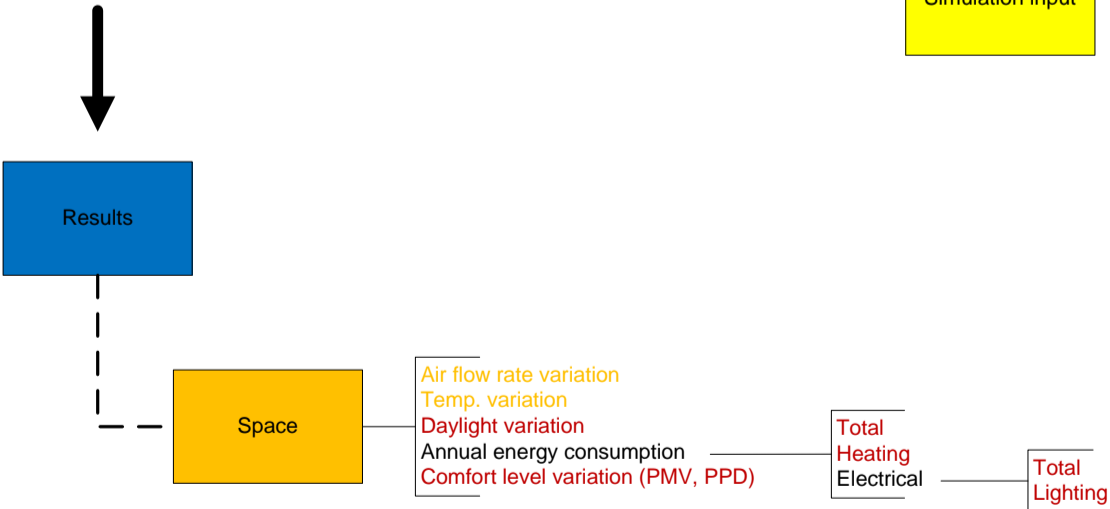
Building Services objects



Architectural objects (page 1)



Energy and thermal calculation



Manual input to software



²It is required to distinguish between installed and movable lighting. In this case, it is assumed that all general lighting is installed.

³The properties need for the heating supply components are to detailed for the scope of this thesis. Hence only key properties have been selected.

Appendix 10 – IFC capabilities – Riuska

Architectural objects

Signatures

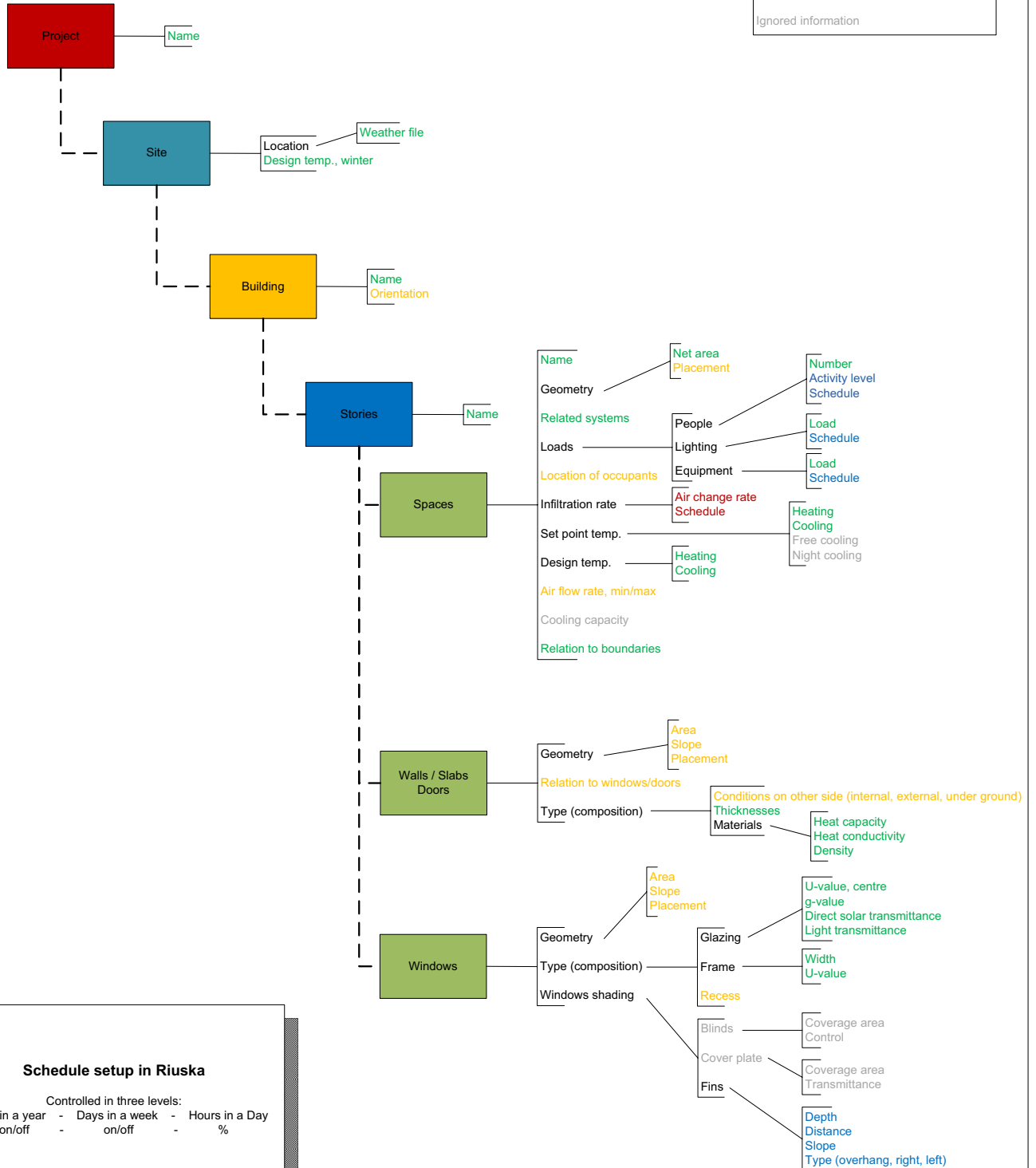
Information defined in IFC

Information derived from IFC

Information unable to be define in IFC

Information sometimes defined in IFC

Ignored information



Schedule setup in Riuska

Controlled in three levels:
 Weeks in a year - Days in a week - Hours in a Day
 on/off - on/off - %

Appendix 10 – IFC capabilities – Riuska

Signatures

Information defined in IFC

Information derived from IFC

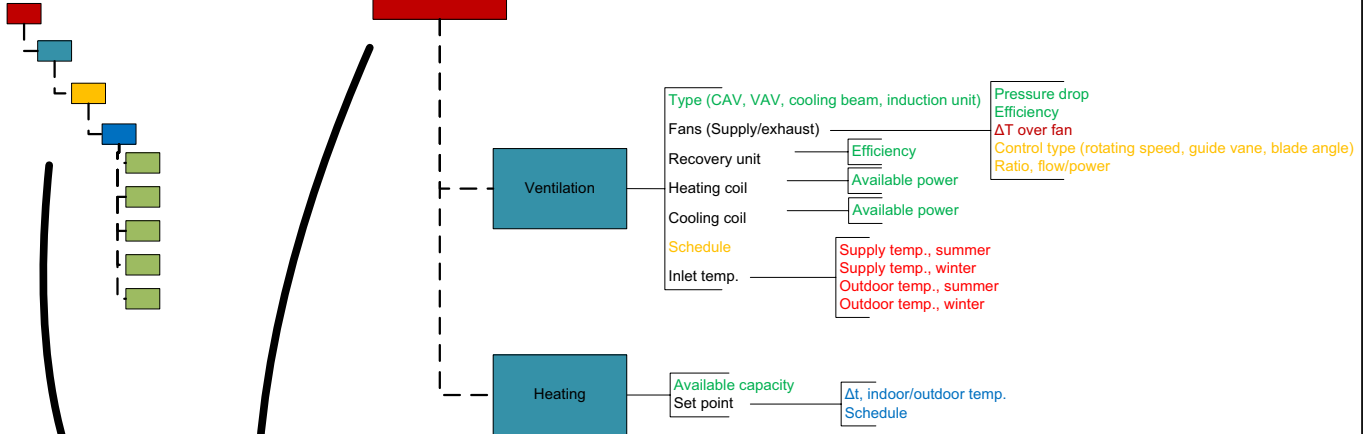
Information unable to be define in IFC

Information sometimes defined in IFC

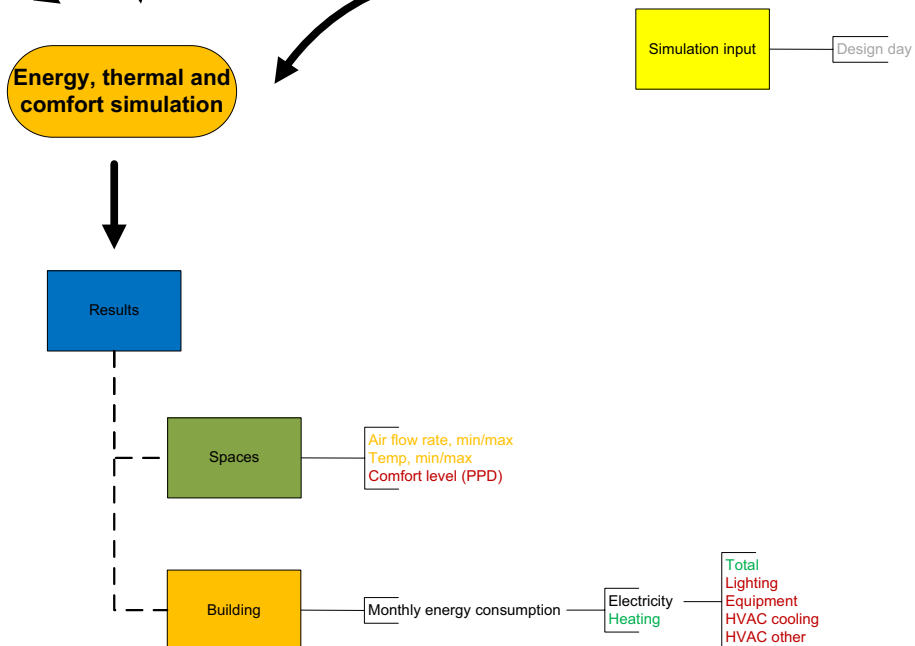
Ignored information

Architectural objects (page 1)

Building Services objects



Manual input to software



Appendix 11 – IFC capabilities – BSim

Moisture calculation not included. Sun distributions calculation not included.
Values considered relevant to the calculation engine of BSim only has been left out. This could e.g. be the ratio of how much heat is distributed via convection and radiation from heat sources.

Signatures

Information defined in IFC

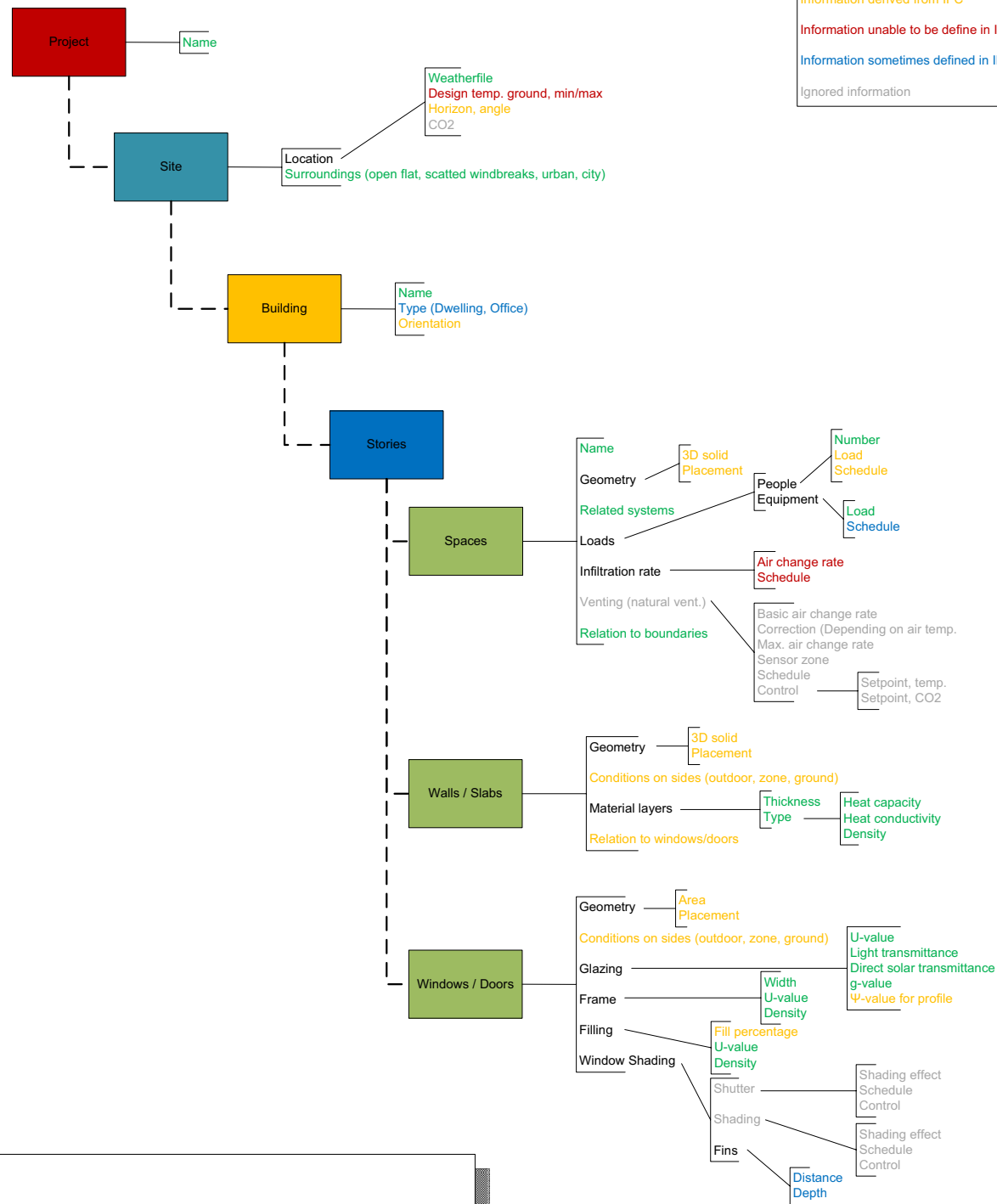
Information derived from IFC

Information unable to be define in IFC

Information sometimes defined in IFC

Ignored information

Architectural objects



Schedule setup in BSim

Controlled in five levels:
Months in a year - Weeks in a year - Days in a week - Hours in a Day - Hours in a Day
on/off - on/off - on/off - on/off - %

Appendix 11 – IFC capabilities – BSim

Signatures

Information defined in IFC

Information derived from IFC

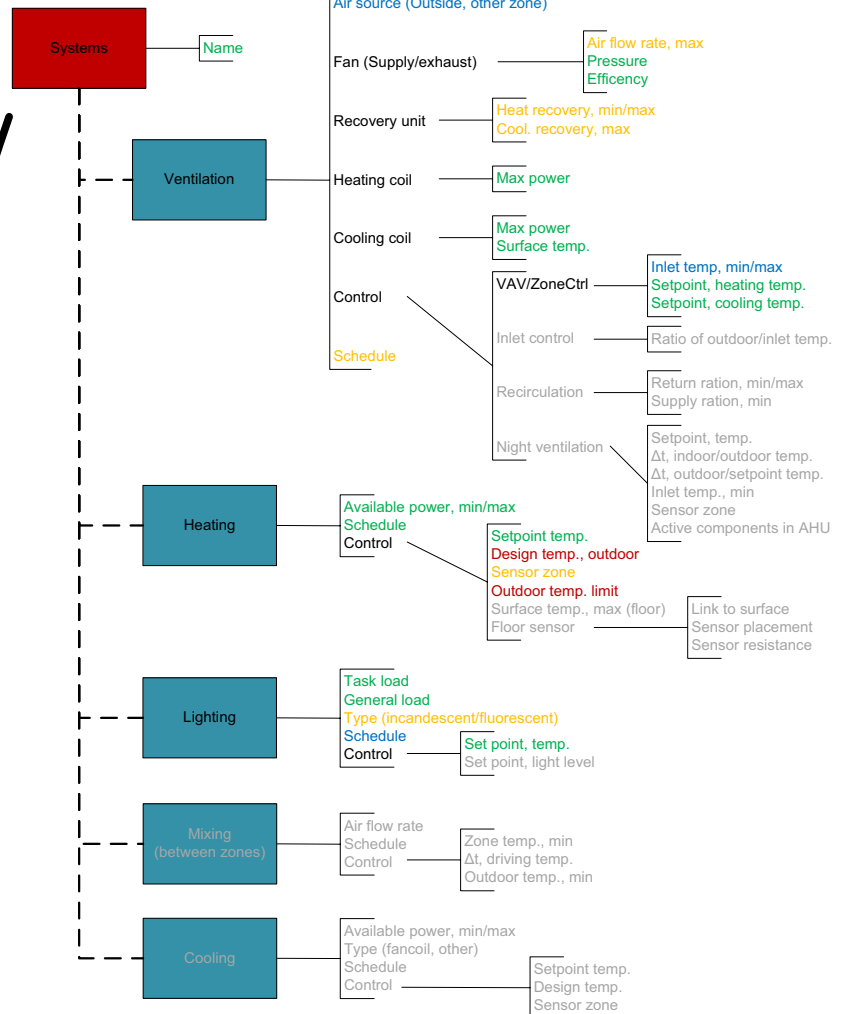
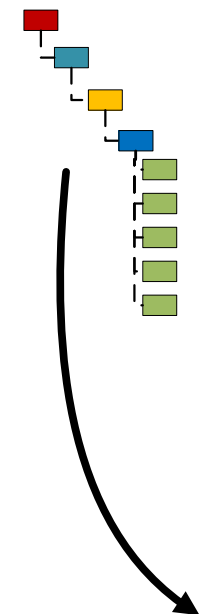
Information unable to be define in IFC

Information sometimes defined in IFC

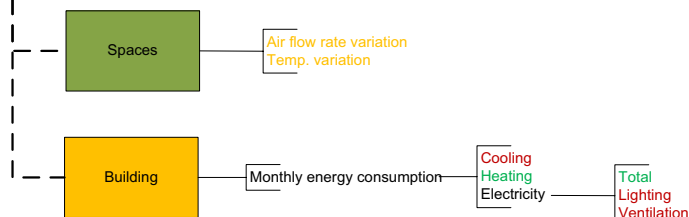
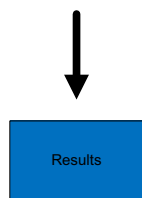
Ignored information

Building Services objects

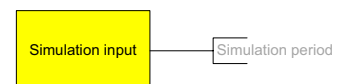
Architectural objects (page 1)



Energy and thermal simulation



Manual input to software



Appendix 12 – IFC capabilities – Be06

Architectural objects

Signatures

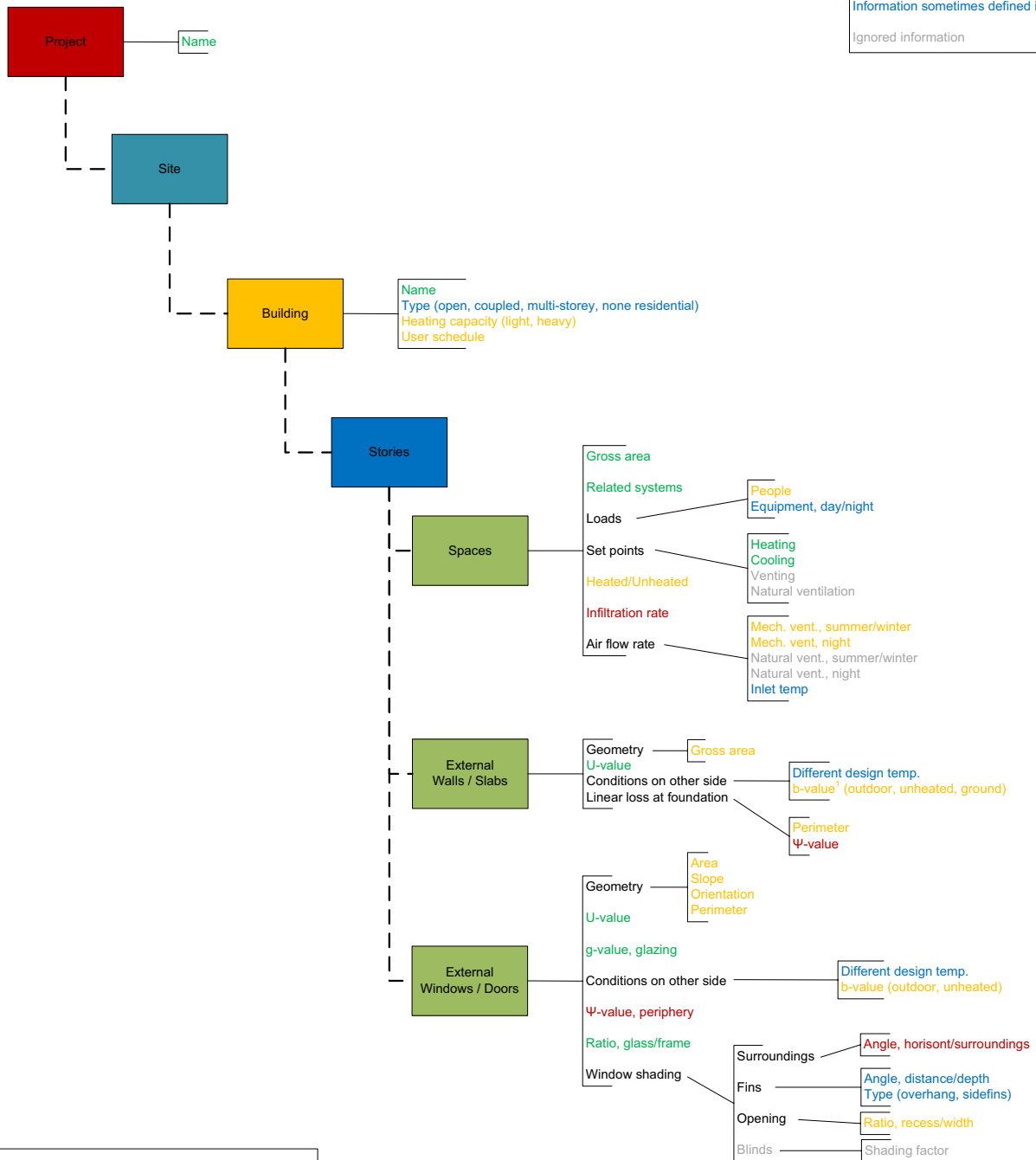
Information defined in IFC

Information derived from IFC

Information unable to be define in IFC

Information sometimes defined in IFC

Ignored information



Schedule setup in Be06

Controlled in two levels:
Hours in a week - Hours in a Day
no. - no.

Or

Controlled by:
Day / Night
Summer / Winter

¹In case of floor heating, the b-value depends on the temp. in the floorpipes, the indoor temp., and the outdoor temp. This has been discarded in this case.

Appendix 12 – IFC capabilities – Be06

Signatures

Information defined in IFC

Information derived from IFC

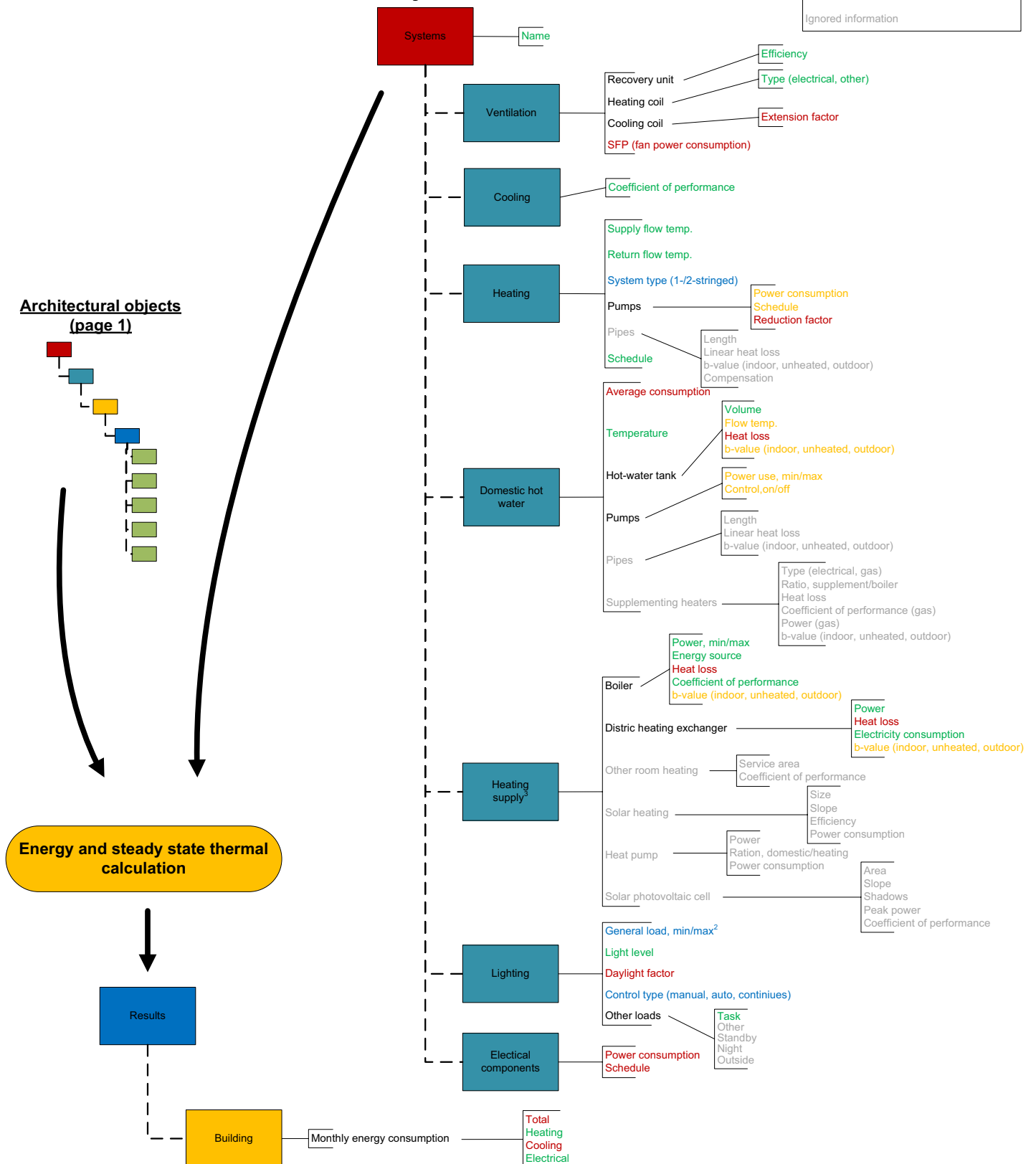
Information unable to be define in IFC

Information sometimes defined in IFC

Ignored information

Building Services objects

Architectural objects (page 1)



²It is required to distinguish between installed and movable lighting. In this case, it is assumed that all general lighting is installed.

³The properties need for the heating supply components are to detailed for the scope of this thesis. Hence only key properties have been selected.

Appendix 13 – IFC capabilities – Flovent

Single room CFD simulation considered for design day calculation.
Simulation of pollution sources has not been included in this schema.

Signatures

Information defined in IFC

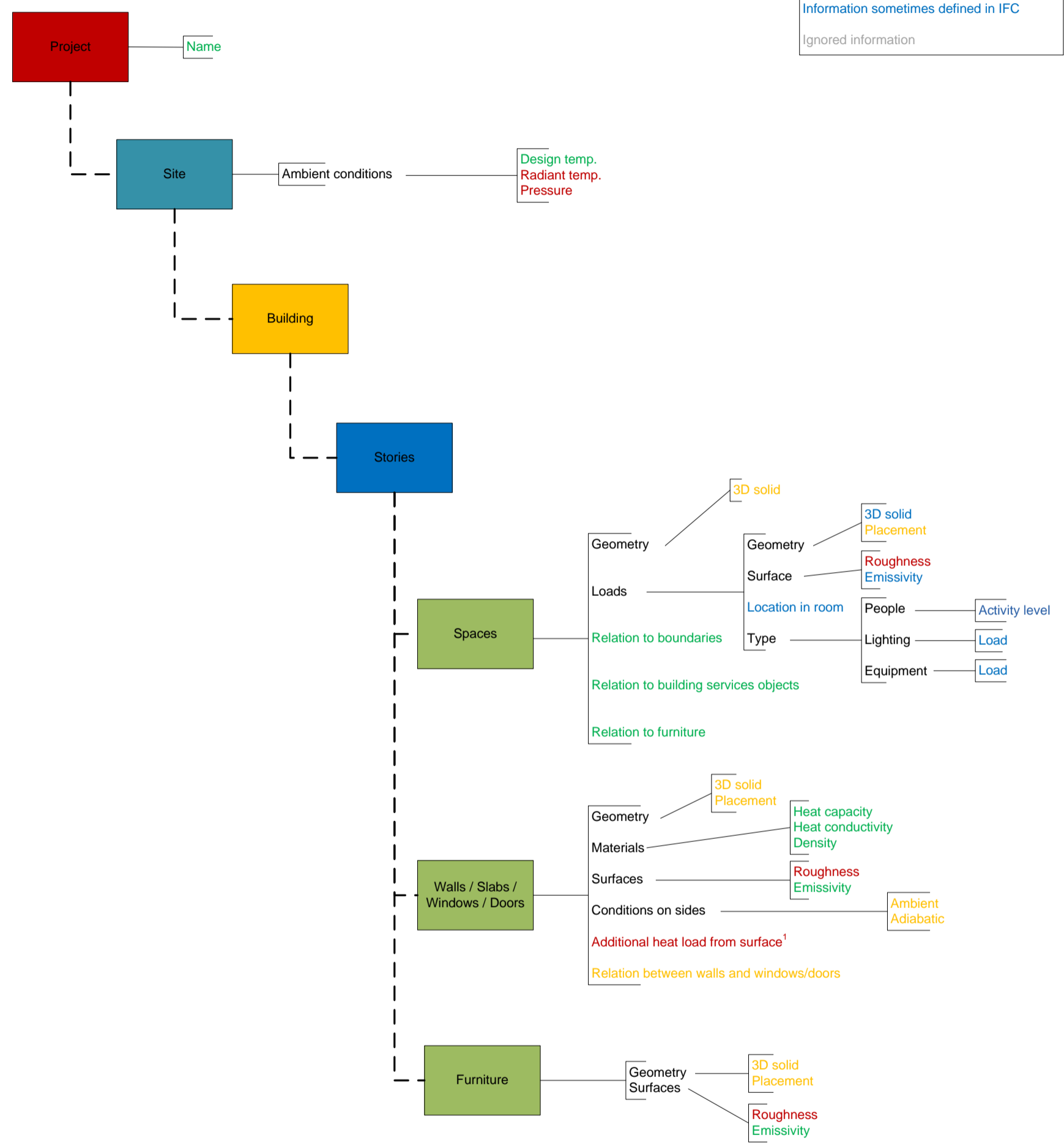
Information derived from IFC

Information unable to be define in IFC

Information sometimes defined in IFC

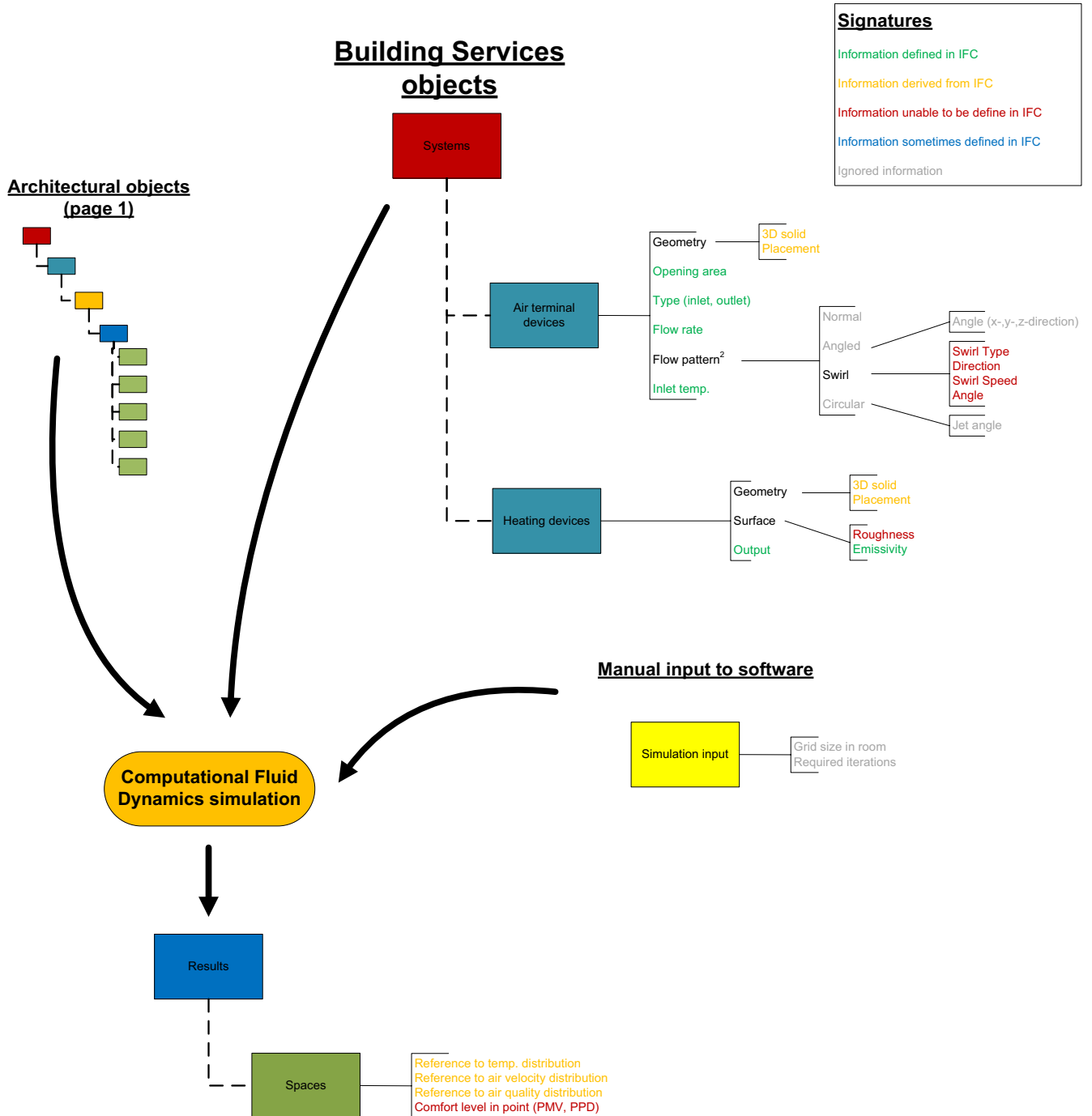
Ignored information

Architechtrual objects



¹To limit calculation time of CFD simulations, solar distribution is typical applied to surfaces as an area of the surface with an additional heat load. Flovent does allow for the solar distribution to be calculated, but this has been discarded in this case.

Appendix 13 – IFC capabilities – Flovent



²Flovent allows for the the selection of four simple representations of air terminal devices. In case of other or more complex devices, the devices has to be modelled manually. This requires a detailed description of the flow pattern provided by the supplier.

Appendix 14

Identified shortcomings of the IFC Specifications

The following list contains suggested modifications to property sets which have been identified as appropriate to implement in the IFC specifications. These were identified during the analysis of capabilities in IFC to support the input and output data required by the five simulation tools iDbuild, Riuska, BSim, Be06 and Flovent. The list is not complete, however, it indicates the inadequacies identified as most obvious to correct for the support for full interoperability within the simulation tools. The suggestions have not been listed in prioritised order.

1. Correction so that *OccupancyNumber* in Pset_SpaceCommon are defined as a time series. To allow for a schedule of the occupants to be derived.
2. Inclusion of *OccupancyActivity* and *OccupancyClothing* in Pset_SpaceThermalDesign. To allow for activity level and clothing level of occupant to be defined.
3. Inclusion of *OverallInfiltrationAirFlowRate* defined as a time series in Pset_SpaceThermalPHistory. To allow for the infiltration rate to be identified.
4. Inclusion of *VentilationInletTemp* to Pset_SpaceThermalPHistory as a time series. To allow for the definition of inlet temperature to be defined without the need for an air terminal device.
5. Inclusion of *HeatLoad* defined as a time series in Pset_SpaceThermalPHistory. To allow for the definition of heat output to be defined without the need for a space heater.
6. Include *ComforLevelPMV* and *ComforLevelPPD* as values in Pset_SpaceThermalPHistory. To allow for the definition of a comfort level for occupants in the space.
7. Creation of a Pset_LightFixturePHistory, which include *EffectiveWattage* as a time series. To allow for a schedule and min/max value to be derived for lighting load.
8. Creation of a Pset_ElectricalDevicePHistory, which include *EffectivePower* as a time series. To allow for a schedule and min/max value to be derived for equipment load.
9. Possibility to define comfort classes in Pset_SpaceThermalRequirements. To allow for comfort evaluation.
10. Inclusion of *Roughness as a property* in IfcGeneralMaterialProperties. To be used for CFD simulations.
11. Inclusion of *SFP* (total power consumption) to Pset_AirSideSystemInformation.
12. Inclusion of a *HotWaterRatio* in Pset_UtilityConsumption to define the amount of hot water used out of the total water consumption.

13. Inclusion of *TotalEnergyConsumption* defined as a time series in Pset_UtilityConsumption. To define the total energy consumption of the building.
14. Inclusion of *Cooling* defined as a time series in Pset_UtilityConsumption. To define the total energy consumption for cooling of the building.
15. Inclusion of *Lighting* defined as a time series in Pset_UtilityConsumption. To define the total energy consumption for lighting of the building.
16. Possibility to assign Pset_UtilityConsumption to spaces.
17. Inclusion of *TotalHeatLoss* defined as a time series in Pset_BoilerPHistory. To allow for the heat loss of boilers to be defined.

Appendix 15

This appendix contains the source code for the IFC interface for Be06. This first page contains the code for the first sheet in the Excel workbook which activates the functions defined in the modules on the following pages.

```
' =====
'
'  **Be06 - IFC import**
'
'  Script created by Niels Treldal, s021820, Technical University of Denmark
'
'  This script uses the following AxtiveX Component which is freeware:
'
'  IFCsvr ActiveX Component, Copyright c 1999, 2005 SECOM Co., Ltd. All Rights Reserved
'  SECOM Co., Ltd. Intelligent Systems Lab.
'  Yoshinobu Adachi
'  E-Mail: yo-adachi@secom.co.jp
' =====

Private Sub CommandButton1_Click()
    Dim vTemp As Object

    vTemp = Application.GetOpenFilename("IFC file (*.ifc),*.ifc")
    If Not vTemp = False Then
        ActiveSheet.Range("C5").Value = vTemp
    End If
End Sub

Private Sub CommandButton2_Click()

    ResetLayout.ResetContent()
    ObjImport.StartImport()

End Sub

Private Sub CommandButton3_Click()
    ResetLayout.ResetContent()
End Sub

Private Sub CommandButton4_Click()
    FindNormal2()
End Sub
```

Appendix 15 - Module "GeoLib"

```
' =====
'
' START OF MODULE "GeoLib"
'
' Script created by Niels Trelldal, s021820, Technical University of Denmark
'
'
' Based on the following:
' IFCsvr/Tk Geometric Library module
'
'   (c) 1998 - 2000
'   SECOM Co., Ltd. Intelligent Systems Lab
'   Yoshinobu Adachi
'
'   E-Mail: adachi@ai.isl.secom.co.jp
'   Tel:    0422-76-2103
'           SC Center 4047
'
' =====
'
' -----
' Global Setting
' -----

Option Explicit On

Public Pi As Double

' -----
'   rotZ
'
'   P2(x2, y2, z2) = rot_by_Zaxis(P1(x1, y1, z1), theta)
'
' -----

Public Function rotZ(ByVal x1 As Double, ByVal y1 As Double, ByVal z1 As Double, _
    ByVal Theta As Double, _
    ByVal x2 As Double, ByVal y2 As Double, ByVal z2 As Double)
    Dim cosT As Double
    Dim sinT As Double

    cosT = Cos(Theta)
    sinT = Sin(Theta)

    x2 = x1 * cosT - y1 * sinT
    y2 = x1 * sinT + y1 * cosT
    z2 = z1
End Function

' -----
'   rotX
'
'   P2(x2, y2, z2) = rot_by_Xaxis(P1(x1, y1, z1), theta)
'
' -----

Public Function rotX(ByVal x1 As Double, ByVal y1 As Double, ByVal z1 As Double, _
    ByVal Theta As Double, _
    ByVal x2 As Double, ByVal y2 As Double, ByVal z2 As Double)
    Dim cosT As Double
    Dim sinT As Double

    cosT = Cos(Theta)
    sinT = Sin(Theta)
```

Appendix 15 - Module "GeoLib"

```

y2 = y1 * cosT - z1 * sinT
z2 = y1 * sinT + z1 * cosT
x2 = x1

```

End Function

```

' -----
' rotY
'
' P2(x2, y2, z2) = rot_by_Yaxis(P1(x1, y1, z1), theta)
' -----

Public Function rotY(ByVal x1 As Double, ByVal y1 As Double, ByVal z1 As Double, ↵
    _
    ByVal Theta As Double, _
    ByVal x2 As Double, ByVal y2 As Double, ByVal z2 As Double)
    Dim cosT As Double
    Dim sinT As Double

    cosT = Cos(Theta)
    sinT = Sin(Theta)

    x2 = x1 * cosT - z1 * sinT
    z2 = x1 * sinT + z1 * cosT
    y2 = y1

```

End Function

```

' -----
' getangle
'
' theta = vec1(x1, y1, z1) and vec2(x2, y2, z2)
'
'      pi/2
'      |
'      pi---+--- 0
'      |
'      3pi/2
'
' Yoshinobu Adachi (adachi@ai.isl.secom.co.jp)
' 2000/05/01 Ver 1.0
' -----

Public Function getangle(ByVal x1 As Double, ByVal y1 As Double, ByVal z1 As ↵
    Double, _
    ByVal x2 As Double, ByVal y2 As Double, ByVal z2 As Double) As Object
    Dim cosT, sinT As Double
    Dim d1 As Double
    Dim d2 As Double
    Dim d3 As Double
    Dim d4 As Double
    Dim bPositive As Boolean
    Dim tempAngle(3) As Double

    ' cosT = vec1.vec2 / (sqr(vec1.vec1)*sqr(vec2.vec2))

    ' Find angle in XY plane
    bPositive = True

    d1 = x1 * y2 - x2 * y1
    d2 = x1 * x1 + y1 * y1
    d3 = x2 * x2 + y2 * y2

    If d2 = 0 Or d3 = 0 Then

```

Appendix 15 - Module "GeoLib"

```
        sinT = 0
Else
    sinT = d1 / (Sqr(d2) * Sqr(d3))
End If

d4 = Excel.WorksheetFunction.Asin(sinT)

If Round(d4, 2) = 0 Then
    d1 = x1 * x2 + y2 * y1

    If d2 = 0 Or d3 = 0 Then
        sinT = 0
    Else
        sinT = d1 / (Sqr(d2) * Sqr(d3))
    End If
    d4 = Excel.WorksheetFunction.Asin(sinT)

    If d4 < 0 Then
        d4 = Pi
    Else
        d4 = 0
    End If

ElseIf d4 < 0 Then d4 = d4 + 2 * Pi
End If

If d4 < 0 Or d4 > 2 * Pi Then MsgBox("Vinkel er gal", vbCritical)

tempAngle(0) = d4

' Find angle in YZ plane
bPositive = True

d1 = y1 * z2 - y2 * z1
d2 = y1 * y1 + z1 * z1
d3 = y2 * y2 + z2 * z2

If d2 = 0 Or d3 = 0 Then
    sinT = 0
Else
    sinT = d1 / (Sqr(d2) * Sqr(d3))
End If

d4 = Excel.WorksheetFunction.Asin(sinT)

If Round(d4, 2) = 0 Then
    d1 = y1 * y2 + z2 * z1

    If d2 = 0 Or d3 = 0 Then
        sinT = 0
    Else
        sinT = d1 / (Sqr(d2) * Sqr(d3))
    End If
    d4 = Excel.WorksheetFunction.Asin(sinT)

    If d4 < 0 Then
        d4 = Pi
    Else
        d4 = 0
    End If

ElseIf d4 < 0 Then d4 = d4 + 2 * Pi
End If

If d4 < 0 Or d4 > 2 * Pi Then MsgBox("Vinkel er gal", vbCritical)

tempAngle(1) = d4
```

Appendix 15 - Module "GeoLib"

```
' Find angle in XZ plane
bPositive = True

d1 = x1 * z2 - x2 * z1
d2 = x1 * x1 + z1 * z1
d3 = x2 * x2 + z2 * z2

If d2 = 0 Or d3 = 0 Then
    sinT = 0
Else
    sinT = d1 / (Sqr(d2) * Sqr(d3))
End If

d4 = Excel.WorksheetFunction.Asin(sinT)

If Round(d4, 2) = 0 Then
    d1 = x1 * x2 + z2 * z1

    If d2 = 0 Or d3 = 0 Then
        sinT = 0
    Else
        sinT = d1 / (Sqr(d2) * Sqr(d3))
    End If
    d4 = Excel.WorksheetFunction.Asin(sinT)

    If d4 < 0 Then
        d4 = Pi
    Else
        d4 = 0
    End If

ElseIf d4 < 0 Then d4 = d4 + 2 * Pi
End If

tempAngle(2) = d4

getangle = tempAngle

End Function

' -----
' End of module "GeoLib"
' -----
```

Appendix 15 - Module "Local2Global"

```

' =====
'
' START OF MODULE "Local2Global"
'
'
' IFCsvr LocalPlacement -> GlobalPlacement
' Written by: Niels Treldal, s021820, Technical University of Denmark
'
' Based on:
' IFCsvr MAP for QuakeII Converter module
'
' (c) 1998 - 2000
' SECOM Co., Ltd. Intelligent Systems Lab
' Yoshinobu Adachi
'
' E-Mail: adachi@ai.isl.secom.co.jp
' Tel:      0422-76-2103
'           SC Center 4047
'
' =====
'
' -----
' Function to transpose local coordinates to global coordinates
' -----

Public Function GetLocal2Global(ByVal coors As Object, ByVal noCoors As Integer, ✓
    ByVal objIfcAxis2Placement3D As IFCsvr.Entity, ByVal objIfcLocalPlacement As ✓
    IFCsvr.Entity)

    Dim i As Integer

    Dim dCoordX As Double, dCoordY As Double, dCoordZ As Double
    Dim dDirX As Double, dDirY As Double, dDirZ As Double
    Dim dAxisX As Double, dAxisY As Double, dAxisZ As Double

    ' Either coors or objIfcAxis2Placement3D can be left blank when calling the ✓
    ' function and will instead be handled here

    If noCoors = 0 Then
        ' If no coors defined, they will be defined as the coordinates of ✓
        IfcAxis2Placement3D
        ' Get values from IfcAxis2Placement3D
        FgetIfcAxis2Placement3D(objIfcAxis2Placement3D, dCoordX, dCoordY, dCoordZ ✓
        , dAxisX, dAxisY, dAxisZ, dDirX, dDirY, dDirZ)
        coors(0, 0) = dCoordX
        coors(0, 1) = dCoordY
        coors(0, 2) = dCoordZ
        Debug.Print("coors defined")
        noCoors = 1
    ElseIf Not objIfcAxis2Placement3D Is Nothing Then
        ' If IfcAxis2Placement3D IS defined, the coors will be transposed to this ✓
        ' settings first.
        ' Get values from IfcAxis2Placement3D
        FgetIfcAxis2Placement3D(objIfcAxis2Placement3D, dCoordX, dCoordY, dCoordZ ✓
        , dAxisX, dAxisY, dAxisZ, dDirX, dDirY, dDirZ)

        ' coors being transposed
        procRotatePoints(coors, noCoors, dCoordX, dCoordY, dCoordZ, dAxisX, ✓
        dAxisY, dAxisZ, dDirX, dDirY, dDirZ)
    End If

    ' Now transpose the coors by every RelativePlacement defined
    Do While True
        ' Define new IfcAxis2Placement3D
        objIfcAxis2Placement3D = objIfcLocalPlacement.Attributes( ✓
        "RelativePlacement").Value
    End Do
End Function

```

Appendix 15 - Module "Local2Global"

```

' Get values from IfcAxis2Placement3D
FgetIfcAxis2Placement3D(objIfcAxis2Placement3D, dCoordX, dCoordY, dCoordZ, _
' -
dAxisX, dAxisY, dAxisZ, dDirX, dDirY, dDirZ)

' coors being transposed
procRotatePoints(coors, noCoors, dCoordX, dCoordY, dCoordZ, dAxisX, dAxisY, dAxisZ, dDirX, dDirY, dDirZ)

'If this was the second last RelativePlacement, the loop will stop
If objIfcLocalPlacement.Attributes("PlacementRelTo").IsNull Then
    Exit Do
End If

' Else the loop continues with next PlacementRelTo
objIfcLocalPlacement = objIfcLocalPlacement.Attributes("PlacementRelTo").Value
Loop

' The last RelativePlacement will be used
objIfcAxis2Placement3D = objIfcLocalPlacement.Attributes("RelativePlacement").Value

' Get values from IfcAxis2Placement3D
FgetIfcAxis2Placement3D(objIfcAxis2Placement3D, dCoordX, dCoordY, dCoordZ, _
dAxisX, dAxisY, dAxisZ, dDirX, dDirY, dDirZ)

' coors being transposed
procRotatePoints(coors, noCoors, dCoordX, dCoordY, dCoordZ, dAxisX, dAxisY, dAxisZ, dDirX, dDirY, dDirZ)

'Return the transposed coors
GetLocal2Global = coors

End Function

' -----
' procRotatePoints
'
'
'
' Yoshinobu Adachi (adachi@ai.isl.secom.co.jp)
' 2000/05/15 Ver 1.0
' -----

Public Function procRotatePoints(ByVal p As Object, ByVal noCoors As Integer, _
ByVal dLoc_X As Double, ByVal dLoc_Y As Double, ByVal dLoc_Z As Double, _
ByVal dAxis_X As Double, ByVal dAxis_Y As Double, ByVal dAxis_Z As Double, _
ByVal dRef_X As Double, ByVal dRef_Y As Double, ByVal dRef_Z As Double)

    Dim i As Integer
    Dim tempAngle As Object
    Dim dAngle(3) As Double
    Dim dx As Double, dy As Double, dz As Double
    Dim x1 As Double, y1 As Double, z1 As Double
    Dim x2 As Double, y2 As Double, z2 As Double
    Dim bRotX As Boolean, bRotY As Boolean, bRotZ As Boolean
    Dim ux1 As Double, uy1 As Double, uz1 As Double
    Dim ux2 As Double, uy2 As Double, uz2 As Double

    Pi = Excel.WorksheetFunction.Pi()

    bRotX = False
    bRotY = False
    bRotZ = False

```

Appendix 15 - Module "Local2Global"

```
ux1 = 1.0#
uy1 = 0.0#
uz1 = 0.0#

' Rotation of Axis
dx = dAxis_X + 0.0#
dy = dAxis_Y + 0.0#
dz = dAxis_Z + 1.0#

tempAngle = GeoLib.getangle(0, 0, 1, dAxis_X, dAxis_Y, dAxis_Z)
dAngle(0) = tempAngle(0)
dAngle(1) = tempAngle(1)
dAngle(2) = tempAngle(2)

If Round(dx, 2) = 0 Then
    bRotX = True
ElseIf Round(dy, 2) = 0 Then
    bRotY = True

End If

If bRotZ Then GeoLib.rotZ(ux1, uy1, uz1, dAngle(0), ux2, uy2, uz2)
If bRotX Then GeoLib.rotX(ux1, uy1, uz1, dAngle(1), ux2, uy2, uz2)
If bRotY Then GeoLib.rotY(ux1, uy1, uz1, dAngle(2), ux2, uy2, uz2)

For i = 0 To noCoors - 1
    x1 = p(i, 0)
    y1 = p(i, 1)
    z1 = p(i, 2)
    If bRotZ Then GeoLib.rotZ(x1, y1, z1, dAngle(0), x2, y2, z2)
    If bRotX Then GeoLib.rotX(x1, y1, z1, dAngle(1), x2, y2, z2)
    If bRotY Then GeoLib.rotY(x1, y1, z1, dAngle(2), x2, y2, z2)
    p(i, 0) = x2
    p(i, 1) = y2
    p(i, 2) = z2
Next i

bRotX = False
bRotY = False
bRotZ = False

' Rotation of Ref
dx = Abs(ux2) + Abs(dRef_X)
dy = Abs(uy2) + Abs(dRef_Y)
dz = Abs(uz2) + Abs(dRef_Z)

tempAngle = GeoLib.getangle(ux2, uy2, uz2, dRef_X, dRef_Y, dRef_Z)
dAngle(0) = tempAngle(0)
dAngle(1) = tempAngle(1)
dAngle(2) = tempAngle(2)

If Int(dx) = 0 Then
    bRotX = True
ElseIf Int(dy) = 0 And dAxis_Z = 0 Then
    bRotY = True
ElseIf Int(dz) = 0 Then
    bRotZ = True
End If

For i = 0 To noCoors - 1
    x1 = p(i, 0)
    y1 = p(i, 1)
    z1 = p(i, 2)
    If bRotZ Then GeoLib.rotZ(x1, y1, z1, dAngle(0), x2, y2, z2)
    If bRotX Then GeoLib.rotX(x1, y1, z1, dAngle(1), x2, y2, z2)
    If bRotY Then GeoLib.rotY(x1, y1, z1, dAngle(2), x2, y2, z2)
    p(i, 0) = x2
```

Appendix 15 - Module "Local2Global"

```
        p(i, 1) = y2
        p(i, 2) = z2
    Next i

    ' Local Position
    Debug.Print("[Position]")
    For i = 0 To noCoors - 1
        x1 = p(i, 0)
        y1 = p(i, 1)
        z1 = p(i, 2)
        x2 = x1 + dLoc_X
        y2 = y1 + dLoc_Y
        z2 = z1 + dLoc_Z
        p(i, 0) = x2
        p(i, 1) = y2
        p(i, 2) = z2
    Next i

End Function

' -----
' Function to retrieve values from IfcAxis2Placement3D
'
' Original from:
' Yoshinobu Adachi (adachi@ai.isl.secom.co.jp)
' 2000/05/15 Ver 1.0
' -----

Public Function FgetIfcAxis2Placement3D(ByVal objPtr As IFCsvr.Entity, _
ByVal dCoordX As Double, ByVal dCoordY As Double, ByVal dCoordZ As Double, _
ByVal dAxisX As Double, ByVal dAxisY As Double, ByVal dAxisZ As Double, _
ByVal dDirX As Double, ByVal dDirY As Double, ByVal dDirZ As Double)

    On Error Resume Next

    Dim objLocation As IFCsvr.Entity
    Dim objAxis As IFCsvr.Entity
    Dim objRefDirection As IFCsvr.Entity
    Dim vCoord As Object
    Dim vAxis As Object
    Dim vDir As Object

    dCoordX = 0 : dCoordY = 0 : dCoordZ = 0
    dAxisX = 0 : dAxisY = 0 : dAxisZ = 0
    dDirX = 0 : dDirY = 0 : dDirZ = 0

    objLocation = objPtr.Attributes("Location").Value
    objAxis = objPtr.Attributes("Axis").Value
    objRefDirection = objPtr.Attributes("RefDirection").Value

    vCoord = objLocation.Attributes("Coordinates").Value
    dCoordX = vCoord(0)
    dCoordY = vCoord(1)
    dCoordZ = vCoord(2)

    If objAxis Is Nothing Then
        dAxisX = 0
        dAxisY = 0
        dAxisZ = 1
    Else
        vAxis = objAxis.Attributes("DirectionRatios").Value
        dAxisX = vAxis(0)
        dAxisY = vAxis(1)
        dAxisZ = vAxis(2)
    End If

    If objRefDirection Is Nothing Then
        dDirX = 1
```

Appendix 15 - Module "Local2Global"

```
        dDirY = 0
        dDirZ = 0
    Else
        vDir = objRefDirection.Attributes("DirectionRatios").Value
        dDirX = vDir(0)
        dDirY = vDir(1)
        dDirZ = vDir(2)
    End If
End Function

' -----
'   End of module "Local2Global"
' -----
```

Appendix 15 - Module "ObjImport"

```
' =====
'
' START OF MODULE "ObjImport"
'
'
' Script created by Niels Treldal, s021820, Technical University of Denmark
'
' =====
'
' -----
' Global Setting
' -----

Option Explicit On
Public objIFCsvr As IFCsvr.R300
Public objDesign As IFCsvr.Design

Public strStyleName(20, 2) As String
Public noStyleNames, noActiveStyle As Integer
Public noWinDoorsInStyle(20) As Integer
Public rSumSpaces, rSumWalls, rSumWinDoors, rSumSlabs As Range
Public lowestWall(2) As Single

' -----
' Starts the import of IFC data
' Imports data from spaces, walls, windows, doors and slabs
' -----

Public Function StartImport()
    Dim filename As String
    Dim strSchemaName As String
    Dim ArchiCAD_OK As Boolean
    Dim objIfcApplication As IFCsvr.Entity

    noStyleNames = 0
    noActiveStyle = 0
    Erase noWinDoorsInStyle
    Erase strStyleName

    'Get IFC file
    filename = ActiveSheet.Range("C5").Value

    objIFCsvr = New IFCsvr.R300
    If objIFCsvr Is Nothing Then
        MsgBox("IFCsvr is not installed.")
        Exit Function
    End If

    objDesign = objIFCsvr.OpenDesign(filename)
    If objDesign Is Nothing Then
        MsgBox("IFC file not found.")
        objDesign = Nothing
        objIFCsvr = Nothing
        Exit Function
    End If

    ' Check that the IFC file schema is 2x3 which this script is created for
    strSchemaName = UCase(objDesign.SchemaName)
    ActiveSheet.Range("C6").Value = strSchemaName

    If Not strSchemaName Like "IFC2X3" Then
        MsgBox("IFC Schema not match. The IFC Schema needs to be 2x3 for the script to work!", vbCritical)
        objDesign = Nothing
        objIFCsvr = Nothing
        Exit Function
    End If
```

Appendix 15 - Module "ObjImport"

```

'Check if file is created using ArchiCAD as one of the applications. This
script needs the IFC file to be created in ArchiCAD due to SpaceBoundaries.
ArchiCAD_OK = False
For Each objIfcApplication In objDesign.FindObjects("IfcApplication")
    If objIfcApplication.Attributes("ApplicationIdentifier").Value =
"ArchiCAD" Then
        ArchiCAD_OK = True
    End If
Next objIfcApplication

If Not ArchiCAD_OK Then
    MsgBox("IFC not created using ArchiCAD. This script needs the IFC file to
be created in ArchiCAD due to missing SpaceBoundaries from other software.",
vbCritical)
    objDesign = Nothing
    objIFCsvr = Nothing
    Exit Function
End If

' Set values of where to write summation
rSumSpaces = Worksheets("Summation").Range("A16")
rSumWalls = Worksheets("Summation").Range("A20")
rSumWinDoors = Worksheets("Summation").Range("A24")
rSumSlabs = Worksheets("Summation").Range("A28")

' Start import of values by calling each functions

GetSpaces()

GetWalls()

GetSlabs()

'The import has finished

Worksheets("Summation").Activate()
objDesign = Nothing
objIFCsvr = Nothing

'MsgBox "End"

End Function

' -----
' Get Spaces
' -----

Private Sub GetSpaces()
    Dim objIfcSpace, objIfcShapeRepresentation, objIfcExtrudedAreaSolid,
objInnerCurve As IFCsvr.Entity
    Dim objIfcRelDefinesByProperties, objIfcPropertySingleValue As IFCsvr.Entity
    Dim objAtt As Object
    Dim varAtt As Object
    Dim r1, r2 As Excel.Range
    Dim strPropertySetName As String
    Dim strPropertyName As String
    Dim strPropertyType As String
    Dim sngLength As Single
    Dim sngWidth As Single
    Dim sngHeight As Single
    Dim sngGrossArea, sngNetArea As Single
    Dim sngNetAreaTemp As Single
    Dim sngNetVolumen As Single
    Dim strACRmin, strACRmax As String
    Dim sngACRmin, sngACRmax As Single
    Dim ACRminEstimated As Boolean
    Dim intX(2) As Single

```

Appendix 15 - Module "ObjImport"

```

Dim intY(2) As Single
Dim runIt(2) As Integer
Dim tempZoneName, strZoneName(20) As String
Dim noZoneNames, noActiveZone As Integer
Dim noSpacesInZone(20) As Integer
Dim ZoneNameFound As Boolean
Dim foundACR As Boolean

Worksheets("Spaces").Activate()
Range("A2").Value = "Spaces"
r1 = ActiveSheet.Range("A5")

noZoneNames = 0

' Analyse each Space
For Each objIfcSpace In objDesign.FindObjects("IfcSpace")

    'Find VAV zone name, if any
    ZoneNameFound = False
    For Each objIfcRelDefinesByProperties In objIfcSpace.GetUsedIn(
        "IfcRelAssignsToGroup", "RelatedObjects")
        With objIfcRelDefinesByProperties.Attributes("RelatingGroup")
            If .Value.Type = "IfcZone" Then
                If .Value.Attributes("ObjectType").Value = "AHU service area"
Then
                    tempZoneName = .Value.Attributes("Description").Value
                    ZoneNameFound = True
                End If
            End If
        End With
    Next objIfcRelDefinesByProperties

    If Not ZoneNameFound Then
        tempZoneName = "Unknown"
    End If

    ' Find out if space is in a existing or new Zone
    noActiveZone = 0
    Do
        If tempZoneName = strZoneName(noActiveZone) Then
            ' Space is in zone with others
            noSpacesInZone(noActiveZone) = noSpacesInZone(noActiveZone) + 1
            Exit Do
        ElseIf noActiveZone = noZoneNames Then
            ' Space is in new zone
            strZoneName(noActiveZone) = tempZoneName
            noZoneNames = noZoneNames + 1

            noSpacesInZone(noActiveZone) = 1

            rSumSpaces.EntireRow.Insert()
            rSumSpaces = rSumSpaces.Offset(-1, 0)
            rSumSpaces.Offset(0, 0).EntireRow.Font.Bold = False

            r1.Offset(0, noActiveZone * 7).Value = "Zone name:"
            r1.Offset(1, noActiveZone * 7).Value = tempZoneName
            rSumSpaces.Offset(0, 0).Formula = "=Spaces!" & r1.Offset(1,
noActiveZone * 7).Address
            rSumSpaces.Offset(0, 0).HorizontalAlignment = xlLeft

            r1.Offset(0, 1 + noActiveZone * 7).Value = "Gross area [m2]:"
            r1.Offset(1, 1 + noActiveZone * 7).NumberFormat = "#,##0.00"
            rSumSpaces.Offset(0, 1).Formula = "=Spaces!" & r1.Offset(1, 1 +
noActiveZone * 7).Address
            r1.Offset(0, 2 + noActiveZone * 7).Value = "Net area [m2]:"
            r1.Offset(1, 2 + noActiveZone * 7).NumberFormat = "#,##0.00"
            rSumSpaces.Offset(0, 2).Formula = "=Spaces!" & r1.Offset(1, 2 +
noActiveZone * 7).Address
            r1.Offset(0, 3 + noActiveZone * 7).Value = "Net volumen [m3]:"
            r1.Offset(1, 3 + noActiveZone * 7).NumberFormat = "#,##0.00"

```

Appendix 15 - Module "ObjImport"

```

        rSumSpaces.Offset(0, 3).Formula = "=Spaces!" & r1.Offset(1, 3 + noActiveZone * 7).Address
        r1.Offset(0, 4 + noActiveZone * 7).Value = "Max air flow rate [l/s/m2]:"
        r1.Offset(1, 4 + noActiveZone * 7).NumberFormat = "#,##0.00"
        rSumSpaces.Offset(0, 4).Formula = "=Spaces!" & r1.Offset(1, 4 + noActiveZone * 7).Address
        r1.Offset(0, 5 + noActiveZone * 7).Value = "Min air flow rate [l/s/m2]:"
        r1.Offset(1, 5 + noActiveZone * 7).NumberFormat = "#,##0.00"
        rSumSpaces.Offset(0, 5).Formula = "=Spaces!" & r1.Offset(1, 5 + noActiveZone * 7).Address

        r1.Offset(3, noActiveZone * 7).Value = "Space name"
        r1.Offset(3, 1 + noActiveZone * 7).Value = "Gross area [m2]:"
        r1.Offset(3, 2 + noActiveZone * 7).Value = "Net area [m2]:"
        r1.Offset(3, 3 + noActiveZone * 7).Value = "Net Volumen [m3]:"
        r1.Offset(3, 4 + noActiveZone * 7).Value = "Max air flow rate [l/s]:"
        r1.Offset(3, 5 + noActiveZone * 7).Value = "Min air flow rate [l/s]:"

        Exit Do
    End If
    noActiveZone = noActiveZone + 1
    Loop While noActiveZone <= noZoneNames

    ' Find space name
    strPropertyName = objIfcSpace.Attributes("Name").Value & " - " & objIfcSpace.Attributes("LongName").Value

    ' Reset all values
    sngLength = 0
    sngWidth = 0
    sngHeight = 0
    sngGrossArea = 0
    sngNetArea = 0
    sngNetVolumen = 0
    strACRmax = "0"
    strACRmin = "0"
    sngACRmax = 0
    sngACRmin = 0
    ACRminEstimated = False
    foundACR = False

    ' Calculate the net area of each space
    For Each objIfcShapeRepresentation In objIfcSpace.Attributes("Representation").Value.FindObjects("IfcShapeRepresentation")

        If objIfcShapeRepresentation.Attributes("RepresentationType").Value = "SweptSolid" Then

            objIfcExtrudedAreaSolid = objIfcShapeRepresentation.Attributes("Items").Value.Item("IfcExtrudedAreaSolid")
            sngHeight = objIfcExtrudedAreaSolid.Attributes("Depth").Value
            sngHeight = sngHeight / 1000

            strPropertyType = objIfcExtrudedAreaSolid.Attributes("SweptArea").Value.Type

            If strPropertyType = "IfcRectangleProfileDef" Then
                ' Space area calculated by length and width
                sngLength = objIfcExtrudedAreaSolid.Attributes("SweptArea").Value.Attributes("XDim").Value
                sngLength = sngLength / 1000
                sngWidth = objIfcExtrudedAreaSolid.Attributes("SweptArea").Value.Attributes("YDim").Value
            End If
        End If
    End For

```

Appendix 15 - Module "ObjImport"

```

sngWidth = sngWidth / 1000
sngNetArea = sngWidth * sngLength
sngNetVolumen = sngNetArea * sngHeight

ElseIf strPropertyType = "IfcArbitraryClosedProfileDef" Or
strPropertyType = "IfcArbitraryProfileDefWithVoids" Then
    ' Space area of outer perimiter calculated
    runIt(1) = 0
    For Each objAtt In objIfcExtrudedAreaSolid.Attributes(
"SweptArea").Value.Attributes("OuterCurve").Value.Attributes("Points").Value
        runIt(0) = 0
        For Each varAtt In objAtt.Attributes(1).Value
            If runIt(0) = 0 Then
                intX(runIt(1)) = varAtt / 1000
                runIt(0) = 1
            Else
                intY(runIt(1)) = varAtt / 1000
                runIt(0) = 0
            End If
        Next varAtt

        'Using the method difined here to calculate the area:
        ' http://mathworld.wolfram.com/PolygonArea.html
        If runIt(1) = 0 Then
            runIt(1) = 1
        Else
            sngNetArea = sngNetArea + intX(0) * intY(1) - intX(1)
* intY(0)

            intX(0) = intX(1)
            intY(0) = intY(1)
        End If
    Next objAtt

    sngNetArea = sngNetArea / 2

    ' Calculate any inner areas which needs to be subtracted
    If strPropertyType = "IfcArbitraryProfileDefWithVoids" Then
        For Each objInnerCurve In objIfcExtrudedAreaSolid.
Attributes("SweptArea").Value.Attributes("InnerCurves").Value
            runIt(1) = 0
            sngNetAreaTemp = 0

            For Each objAtt In objInnerCurve.Attributes("Points")
.Value
                runIt(0) = 0
                For Each varAtt In objAtt.Attributes(1).Value

                    If runIt(0) = 0 Then
                        intX(runIt(1)) = varAtt / 1000
                        runIt(0) = 1
                    Else
                        intY(runIt(1)) = varAtt / 1000
                        runIt(0) = 0
                    End If

                Next varAtt

                'Using the method difined here to calculate the
area:
                ' http://mathworld.wolfram.com/PolygonArea.html
                If runIt(1) = 0 Then
                    runIt(1) = 1
                Else
                    sngNetAreaTemp = sngNetAreaTemp + intX(0) *
intY(1) - intX(1) * intY(0)

                    intX(0) = intX(1)
                    intY(0) = intY(1)
                End If
            End If
        Next objInnerCurve
    End If
End If

```

Appendix 15 - Module "ObjImport"

```

Next objAtt

sngNetAreaTemp = sngNetAreaTemp / 2
sngNetArea = sngNetArea - Abs(sngNetAreaTemp)

Next objInnerCurve
End If

sngNetVolumen = sngNetArea * sngHeight
End If
End If

Next objIfcShapeRepresentation

' Find gross and net area if defined in property set "Pset_SpaceCommon"
For Each objIfcRelDefinesByProperties In objIfcSpace.GetUsedIn(
    "IfcRelDefinesByProperties", "RelatedObjects")
    With objIfcRelDefinesByProperties.Attributes(
        "RelatingPropertyDefinition")
        If .Value.Type = "IfcPropertySet" Then
            If .Value.Attributes("Name").Value = "Pset_SpaceCommon" Then
                For Each objIfcPropertySingleValue In .Value.Attributes(
                    "HasProperties").Value
                    If objIfcPropertySingleValue.Attributes("Name").Value =
                        "GrossPlannedArea" Then
                        sngGrossArea = objIfcPropertySingleValue.
                            Attributes("NominalValue").Value
                        ElseIf objIfcPropertySingleValue.Attributes("Name").
                            Value = "NetPlannedArea" Then
                        sngNetArea = objIfcPropertySingleValue.Attributes(
                            "NominalValue").Value
                        'Recalculate Net Volumen
                        sngNetVolumen = sngNetArea * sngHeight
                    End If
                Next objIfcPropertySingleValue
            End If
        End With
    Next objIfcRelDefinesByProperties

' Find air change rate if defined in property set
"Pset_SpaceHvacInformation" (IFC2x2) or "Pset_SpaceThermalDesign" (IFC2x3)
For Each objIfcRelDefinesByProperties In objIfcSpace.GetUsedIn(
    "IfcRelDefinesByProperties", "RelatedObjects")
    With objIfcRelDefinesByProperties.Attributes(
        "RelatingPropertyDefinition")
        If .Value.Type = "IfcPropertySet" Then
            If .Value.Attributes("Name").Value =
                "Pset_SpaceHvacInformation" Or .Value.Attributes("Name").Value =
                "Pset_SpaceThermalDesign" Then
                For Each objIfcPropertySingleValue In .Value.Attributes(
                    "HasProperties").Value
                    ' CoolingDesignAirflow is available
                    If objIfcPropertySingleValue.Attributes("Name").Value =
                        "CoolingDesignAirflow" Then
                        strACRmax = objIfcPropertySingleValue.Attributes(
                            "NominalValue").Value
                        foundACR = True
                    End If
                    ' HeatingDesignAirflow is available
                    If objIfcPropertySingleValue.Attributes("Name").Value =
                        "HeatingDesignAirflow" Then
                        strACRmin = objIfcPropertySingleValue.Attributes(
                            "NominalValue").Value
                        foundACR = True
                    End If
                    ' If only VentilationAirFlowrate is available values

```

Appendix 15 - Module "ObjImport"

```

are derived
    If objIfcPropertySingleValue.Attributes("Name").Value = "VentilationAirFlowrate" Then
        If foundACR = False Then
            ' Issues with "." and "," requires the transformation between single and string values
            strACRmax = objIfcPropertySingleValue.Attributes("NominalValue").Value
            strACRmax = WorksheetFunction.Substitute(strACRmax, ",", ".")
            sngACRmax = strACRmax
            sngACRmax = sngACRmax / 1000
            sngACRmin = sngACRmax * 0.3
            strACRmin = sngACRmin
            strACRmin = WorksheetFunction.Substitute(strACRmin, ",", ".")
            ACRminEstimated = True
            foundACR = True
        End If
    End If
Next objIfcPropertySingleValue
End If
End With
Next objIfcRelDefinesByProperties

' Insert information found for space
r1.Offset(3 + noSpacesInZone(noActiveZone), noActiveZone * 7).Value = strPropertyName

If sngGrossArea = 0 And sngNetArea <> 0 Then
    With r1.Offset(3 + noSpacesInZone(noActiveZone), 1 + noActiveZone * 7)
        .AddComment()
        .Comment.Visible = False
        .Comment.Text(Text:="Gross area property for space is not found in property set. The value is estimated as 'calculated net area' * 1,1.")
    End With
    sngGrossArea = sngNetArea * 1.1
End If

r1.Offset(3 + noSpacesInZone(noActiveZone), 1 + noActiveZone * 7).Value = sngGrossArea
r1.Offset(3 + noSpacesInZone(noActiveZone), 1 + noActiveZone * 7).NumberFormat = "#,##0.00"
r1.Offset(1, 1 + noActiveZone * 7).Formula = "=SUM(" & r1.Offset(4, 1 + noActiveZone * 7).Address & ":" & _
    r1.Offset(3 + noSpacesInZone(noActiveZone), 1 + noActiveZone * 7).Address & ")"

r1.Offset(3 + noSpacesInZone(noActiveZone), 2 + noActiveZone * 7).Value = sngNetArea
r1.Offset(3 + noSpacesInZone(noActiveZone), 2 + noActiveZone * 7).NumberFormat = "#,##0.00"
r1.Offset(1, 2 + noActiveZone * 7).Formula = "=SUM(" & r1.Offset(4, 2 + noActiveZone * 7).Address & ":" & _
    r1.Offset(3 + noSpacesInZone(noActiveZone), 2 + noActiveZone * 7).Address & ")"

r1.Offset(3 + noSpacesInZone(noActiveZone), 3 + noActiveZone * 7).Value = sngNetVolumen
r1.Offset(3 + noSpacesInZone(noActiveZone), 3 + noActiveZone * 7).NumberFormat = "#,##0.00"
r1.Offset(1, 3 + noActiveZone * 7).Formula = "=SUM(" & r1.Offset(4, 3 + noActiveZone * 7).Address & ":" & _
    r1.Offset(3 + noSpacesInZone(noActiveZone), 3 + noActiveZone * 7).Address & ")"

If foundACR = True Then

```

Appendix 15 - Module "ObjImport"

```

        r1.Offset(3 + noSpacesInZone(noActiveZone), 4 + noActiveZone * 7).
Formula = "=" & strACRmax & "*1000"
        r1.Offset(3 + noSpacesInZone(noActiveZone), 4 + noActiveZone * 7).
NumberFormat = "#,##0"
        r1.Offset(1, 4 + noActiveZone * 7).Formula = "=SUM(" & r1.Offset(4, 4 +
+ noActiveZone * 7).Address & ":" & _
        r1.Offset(3 + noSpacesInZone(noActiveZone), 4 +
noActiveZone * 7).Address & ")/" & r1.Offset(1, 1 + noActiveZone * 7).Address

        If ACRminEstimated = True Then
            With r1.Offset(3 + noSpacesInZone(noActiveZone), 5 + noActiveZone
* 7)
                .AddComment()
                .Comment.Visible = False
                .Comment.Text(Text:="Value for min air change rate was not
found. The value is estimated as 'max air change rate' * 30%.")
            End With
        End If
        r1.Offset(3 + noSpacesInZone(noActiveZone), 5 + noActiveZone * 7).
Formula = "=" & strACRmin & "*1000"
        r1.Offset(3 + noSpacesInZone(noActiveZone), 5 + noActiveZone * 7).
NumberFormat = "#,##0"
        r1.Offset(1, 5 + noActiveZone * 7).Formula = "=SUM(" & r1.Offset(4, 5 +
+ noActiveZone * 7).Address & ":" & _
        r1.Offset(3 + noSpacesInZone(noActiveZone), 5 +
noActiveZone * 7).Address & ")/" & r1.Offset(1, 1 + noActiveZone * 7).Address
    End If

Next objIfcSpace

ActiveSheet.Columns("A:BA").AutoFit()

End Sub

' -----
' Get Walls
' -----

Private Sub GetWalls()
    Dim IsWallExternal As Boolean
    Dim uvalueWall As Single

    Dim objIfcWall, objIfcShapeRepresentation, objIfcRelDefinesByProperties,
objIfcPropertySingleValue As IFCsvr.Entity
    Dim objRelAssociatesMaterial As IFCsvr.Entities
    Dim strLayerSetName(20, 2) As String
    Dim noWallsInLayerSet(20) As Integer
    Dim tempLayerSetName As String
    Dim noLayerSetNames, noActiveLayerSet As Integer
    Dim objAtt As Object
    Dim varAtt As Object
    Dim r1, rWinDoors, rTemp As Excel.Range
    Dim strPropertySetName As String
    Dim strPropertyType As String
    Dim strPropertyName As String
    Dim strWidth As Single
    Dim strHeight As Single
    Dim strArea As Single
    Dim intX(2) As Single
    Dim intY(2) As Single
    Dim runIt(2) As Integer
    Dim direction As String
    Dim wallDepth As Single
    Dim SplitWall, SplitDone As Boolean
    Dim wallPlacement As String

    Worksheets("Walls").Activate()

```

Appendix 15 - Module "ObjImport"

```

Range("A2").Value = "Walls"
r1 = ActiveSheet.Range("A5")

'Reset values
lowestWall(0) = 0
lowestWall(1) = 1000 + Worksheets("Summation").Range("C8").Value
noLayerSetNames = 0

' Find information for each wall
For Each objIfcWall In objDesign.FindObjects("IfcWall")
    IsWallExternal = False
    uvalueWall = 0.0#
    SplitWall = False
    SplitDone = True
    wallPlacement = "AboveGround"

    For Each objIfcRelDefinesByProperties In objIfcWall.GetUsedIn(
        "IfcRelDefinesByProperties", "RelatedObjects")

        ' Find U-value and if wall is internal or external from values in
        Pset_WallCommon
        With objIfcRelDefinesByProperties.Attributes(
            "RelatingPropertyDefinition")

            If .Value.Type = "IfcPropertySet" Then
                If .Value.Attributes("Name").Value = "Pset_WallCommon" Then
                    For Each objIfcPropertySingleValue In .Value.Attributes(
                        "HasProperties").Value
                        If objIfcPropertySingleValue.Attributes("Name").Value =
                            "ThermalTransmittance" Then
                                uvalueWall = objIfcPropertySingleValue.Attributes(
                                    "NominalValue").Value
                                ElseIf objIfcPropertySingleValue.Attributes("Name").
                                    Value = "IsExternal" Then
                                    IsWallExternal = objIfcPropertySingleValue.
                                        Attributes("NominalValue").Value
                                End If
                            Next objIfcPropertySingleValue
                        End If
                    End If
                End With

            Next objIfcRelDefinesByProperties

            ' Only external walls will be analysed to save time
            If IsWallExternal Then

                ' Find the direction and placement of the wall
                WallDirection.GetWallDirAndPlace(objIfcWall, direction, wallDepth)

                ' If wall is part of the building envelope continue
                If direction <> "Internal" And direction <> "Outside" Then

                    ' If some part of the wall is below 2 meters of ground, the wall
                    must be split

                    MsgBox "wallDepth: " & wallDepth
                    If wallDepth < 0 Then
                        wallPlacement = "BelowGround"
                        SplitWall = True
                        SplitDone = False
                    End If

                    ' Do one or to times to potentially split wall in two
                    Do
                        'If SplitWall Then MsgBox "komnu"

                        If SplitDone = True Then
                            SplitWall = False
                            wallPlacement = "AboveGround"

```

Appendix 15 - Module "ObjImport"

```

End If

' Find layer name of material composition
objRelAssociatesMaterial = objIfcWall.GetUsedIn(
    "IfcRelAssociatesMaterial", "RelatedObjects")
tempLayerSetName = objRelAssociatesMaterial.Item(1).
Attributes("RelatingMaterial") _
.Value.Attributes("ForLayerSet").Value.Attributes("LayerSetName").Value()

' Find out if the wall composition is new or exists
noActiveLayerSet = 0
Do
    If tempLayerSetName = strLayerSetName(noActiveLayerSet,
1) And wallPlacement = strLayerSetName(noActiveLayerSet, 2) Then
        ' Wall composition exists, current wall will be added
        noWallsInLayerSet(noActiveLayerSet) =
noWallsInLayerSet(noActiveLayerSet) + 1
        Exit Do

    ElseIf noActiveLayerSet = noLayerSetNames Then
        ' Wall composition is new and will be added
        strLayerSetName(noActiveLayerSet, 1) =
tempLayerSetName
        strLayerSetName(noActiveLayerSet, 2) = wallPlacement
        noLayerSetNames = noLayerSetNames + 1

        noWallsInLayerSet(noActiveLayerSet) = 1

        ' Setup composition information in Excel
        rSumWalls.EntireRow.Insert()
        rSumWalls = rSumWalls.Offset(-1, 0)
        rSumWalls.Offset(0, 0).EntireRow.Font.Bold = False

        r1.Offset(0, noActiveLayerSet * 6).Value = "Wall type"
        r1.Offset(1, noActiveLayerSet * 6).Value =
tempLayerSetName
        rSumWalls.Offset(0, 0).Formula = "=Walls!" & r1.
Offset(1, noActiveLayerSet * 6).Address
        rSumWalls.Offset(0, 0).HorizontalAlignment = xlLeft

        r1.Offset(0, 1 + noActiveLayerSet * 6).Value = "Total"
        r1.Offset(1, 1 + noActiveLayerSet * 6).NumberFormat =
"#,##0.00"
        rSumWalls.Offset(0, 1).Formula = "=Walls!" & r1.
Offset(1, 1 + noActiveLayerSet * 6).Address

        r1.Offset(0, 2 + noActiveLayerSet * 6).Value = "U -
valueWall"
        r1.Offset(1, 2 + noActiveLayerSet * 6).Value =
        r1.Offset(1, 2 + noActiveLayerSet * 6).NumberFormat =
"#,##0.00"
        rSumWalls.Offset(0, 2).Formula = "=Walls!" & r1.
Offset(1, 2 + noActiveLayerSet * 6).Address

        r1.Offset(0, 3 + noActiveLayerSet * 6).Value = "b-
factor:"
        r1.Offset(1, 3 + noActiveLayerSet * 6).Value =
        uvalueWall

        If SplitWall = True Then
            r1.Offset(1, 3 + noActiveLayerSet * 6).Value = 0.
7

        Else
            r1.Offset(1, 3 + noActiveLayerSet * 6).Value = 1
        End If
        r1.Offset(1, 3 + noActiveLayerSet * 6).NumberFormat =
"#,##0.0"

```

Appendix 15 - Module "ObjImport"

```

        rSumWalls.Offset(0, 3).Formula = "=Walls!" & r1.
Offset(1, 3 + noActiveLayerSet * 6).Address

        r1.Offset(3, noActiveLayerSet * 6).Value = "Wall name"
"
        r1.Offset(3, 1 + noActiveLayerSet * 6).Value =
"Comment"
        r1.Offset(3, 2 + noActiveLayerSet * 6).Value = "Width"
[m]"
        r1.Offset(3, 3 + noActiveLayerSet * 6).Value =
"Height [m]"
        r1.Offset(3, 4 + noActiveLayerSet * 6).Value = "Area"
[m2]"

        Exit Do
    End If
    noActiveLayerSet = noActiveLayerSet + 1
Loop While noActiveLayerSet <= noLayerSetNames

' Find wall name
strPropertyName = objIfcWall.Attributes("Name").Value

' Reset values
strPropertyType = ""
strWidth = 0
strHeight = 0
strArea = 0

' Find wall height and wall length
For Each objIfcShapeRepresentation In objIfcWall.Attributes(
"Representation").Value.Attributes("Representations").Value.FindObjects(
"IfcShapeRepresentation")

    ' Find wall height from swept solid profile
    If objIfcShapeRepresentation.Attributes(
"RepresentationType").Value = "SweptSolid" Then

        strHeight = objIfcShapeRepresentation.Attributes(
"Items").Value.Item("IfcExtrudedAreaSolid").Attributes("Depth").Value
        strHeight = strHeight / 1000

        If SplitWall = True Then
            strHeight = -wallDepth
        Else
            strHeight = strHeight + wallDepth
            If strHeight < 0 Then strHeight = 0
        End If

        ' Find wall width from length of 2D axis
        ElseIf objIfcShapeRepresentation.Attributes(
"RepresentationType").Value = "Curve2D" Then

            runIt(1) = 0
            For Each objAtt In objIfcShapeRepresentation.
Attributes("Items").Value.Item("IfcPolyline").Attributes("Points").Value

                runIt(0) = 0
                For Each varAtt In objAtt.Attributes(1).Value
                    If runIt(0) = 0 Then
                        intX(runIt(1)) = varAtt / 1000
                        runIt(0) = 1
                    Else
                        intY(runIt(1)) = varAtt / 1000
                        runIt(0) = 0
                    End If
                Next varAtt
                runIt(1) = 1
            Next objAtt

            strWidth = Sqr((intX(0) - intX(1)) ^ 2 + (intY(0) -
intY(1)) ^ 2)

```

Appendix 15 - Module "ObjImport"

```

        End If

    Next objIfcShapeRepresentation

    ' Calculate wall area
    strArea = strWidth * strHeight

    ' Write values to Excel
    r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet),
noActiveLayerSet * 6).Value = strPropertyName
    r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet), 1 +
noActiveLayerSet * 6).Value = "Orientation: " & direction

    r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet), 2 +
noActiveLayerSet * 6).Value = strWidth
    r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet), 2 +
noActiveLayerSet * 6).NumberFormat = "#,##0.00"

    r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet), 3 +
noActiveLayerSet * 6).Value = strHeight
    r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet), 3 +
noActiveLayerSet * 6).NumberFormat = "#,##0.00"

    r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet), 4 +
noActiveLayerSet * 6).Value = strArea
    r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet), 4 +
noActiveLayerSet * 6).NumberFormat = "#,##0.00"

    r1.Offset(1, 1 + noActiveLayerSet * 6).Formula = "=SUM(" & r1.
.Offset(4, 4 + noActiveLayerSet * 6).Address & ":" & _
        r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet),
, 4 + noActiveLayerSet * 6).Address & ")"

    ' Find windows and doors inserted in walls above ground
    If SplitWall = False Then
        rWinDoors = findRelatedWinDoors(objIfcWall, direction)
        Worksheets("Walls").Activate()

        ' If any windows/doors inserted, subtract their area from
wall area
        If Not rWinDoors Is Nothing Then
            For Each rTemp In rWinDoors.Rows
                noWallsInLayerSet(noActiveLayerSet) =
noWallsInLayerSet(noActiveLayerSet) + 1
                r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet),
, 1 + noActiveLayerSet * 6).Formula = "=WinDoors!" & rTemp.Cells(1, 1).
Address
                r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet),
, 4 + noActiveLayerSet * 6).Formula = "--WinDoors!" & rTemp.Cells(1, 5).
Address
            Next rTemp
        End If

        r1.Offset(1, 1 + noActiveLayerSet * 6).Formula = "=SUM(" &
& r1.Offset(4, 4 + noActiveLayerSet * 6).Address & ":" & _
            r1.Offset(3 + noWallsInLayerSet(noActiveLayerSet),
, 4 + noActiveLayerSet * 6).Address & ")"

    End If
    SplitDone = True
    Loop While SplitWall = True

End If

End If
Next objIfcWall

```

Appendix 15 - Module "ObjImport"

```

ActiveSheet.Columns("A:BA").AutoFit()

End Sub

' -----
' Find windows and doors inserted in wall
' -----

Public Function findRelatedWinDoors(ByVal objIfcWall As IFCsvr.Entity, ByVal direction As String) As Range
    Dim objIfcRelVoidsElement, objIfcWinDoors, objIfcOpening,
    objIfcRelFillsElement As IFCsvr.Entity
    Dim objIfcRelDefinesByProperties, objIfcPropertySingleValue As IFCsvr.Entity
    Dim objRelDefinesByType, objIfcRelDefinesByType As IFCsvr.Entities
    Dim tempStyleName As String
    Dim r1, rWinDoor As Excel.Range
    Dim strPropertyName As String
    Dim strWidth As Single
    Dim strHeight As Single
    Dim strPerimeter, strArea As Single
    Dim uvalueWinDoor As Single
    Dim LiningThickness As Single
    Dim LiningFound As Boolean
    Dim glazingFraction As Single

    Worksheets("WinDoors").Activate()
    Range("A2").Value = "Windows/Doors"
    r1 = ActiveSheet.Range("A5")

    findRelatedWinDoors = Nothing

    ' The function GetUsedIn does not work for IfcRelVoidsElement, hence all
    IfcRelVoidsElement has to been evaluated
    For Each objIfcRelVoidsElement In objDesign.FindObjects("IfcRelVoidsElement")

        ' Check if VoidsElement relates to the wall in question
        If objIfcRelVoidsElement.Attributes("RelatingBuildingElement").Value.
        P21ID = objIfcWall.P21ID Then

            objIfcOpening = objIfcRelVoidsElement.Attributes(
            "RelatedOpeningElement").Value

            ' The function GetUsedIn does not work for IfcRelFillsElement, hence
            all IfcRelFillsElement has to been evaluated
            For Each objIfcRelFillsElement In objDesign.FindObjects(
            "IfcRelFillsElement")

                ' Check if VoidsElement relates to the wall in question
                If objIfcRelFillsElement.Attributes("RelatingOpeningElement").
                Value.P21ID = objIfcOpening.P21ID Then

                    objIfcWinDoors = objIfcRelFillsElement.Attributes(
                    "RelatedBuildingElement").Value

                    If objIfcWinDoors.Type <> "IfcWindow" And objIfcWinDoors.Type
                    <> "IfcDoor" Then
                        MsgBox("Opening no. #" & objIfcOpening.P21ID & " in wall
                        " & objIfcWall.Attributes("Name").Value & " is not a window or door and hence
                        not supported")
                        GoTo NextWinDoor
                    End If

                    ' Find U-value property, if any, for windows and doors

```

Appendix 15 - Module "ObjImport"

```

        uvalueWinDoor = 0.0#
        For Each objIfcRelDefinesByProperties In objIfcWinDoors.
GetUsedIn("IfcRelDefinesByProperties", "RelatedObjects")
            With objIfcRelDefinesByProperties.Attributes(
"RelatingPropertyDefinition")
                If .Value.Type = "IfcPropertySet" Then
                    If .Value.Attributes("Name").Value =
"Pset_WindowCommon" Or .Value.Attributes("Name").Value = "Pset_DoorCommon"
Then
                        For Each objIfcPropertySingleValue In .Value.
Attributes("HasProperties").Value
                            If objIfcPropertySingleValue.Attributes(
"Name").Value = "ThermalTransmittance" Then
                                uvalueWinDoor =
objIfcPropertySingleValue.Attributes("NominalValue").Value
                                End If
                            Next objIfcPropertySingleValue
                        End If
                    End If
                End With
            Next objIfcRelDefinesByProperties

            ' Find type name of window/door
            objIfcRelDefinesByType = objIfcWinDoors.GetUsedIn(
"IfcRelDefinesByType", "RelatedObjects")
            tempStyleName = objIfcRelDefinesByType.Item(1).Attributes(
"RelatingType").Value.Attributes("Name").Value

            ' Check if type is new or existing
            noActiveStyle = 0
            Do
                If tempStyleName = strStyleName(noActiveStyle, 1) And
direction = strStyleName(noActiveStyle, 2) Then
                    ' Existing type, window/door added
                    noWinDoorsInStyle(noActiveStyle) = noWinDoorsInStyle
(noActiveStyle) + 1
                Exit Do
            ElseIf noActiveStyle = noStyleNames Then
                ' New type, type added
                strStyleName(noActiveStyle, 1) = tempStyleName
                strStyleName(noActiveStyle, 2) = direction
                noStyleNames = noStyleNames + 1

                noWinDoorsInStyle(noActiveStyle) = 1

                ' Setup type information in Excel
                rSumWinDoors.EntireRow.Insert()
                rSumWinDoors = rSumWinDoors.Offset(-1, 0)
                rSumWinDoors.Offset(0, 0).EntireRow.Font.Bold = False

                If objIfcWinDoors.Type = "IfcWindow" Then
                    r1.Offset(0, noActiveStyle * 8).Value = "Window
type:"
                ElseIf objIfcWinDoors.Type = "IfcDoor" Then
                    r1.Offset(0, noActiveStyle * 8).Value = "Door
type:"
                End If

                r1.Offset(1, noActiveStyle * 8).Value = tempStyleName
                rSumWinDoors.Offset(0, 0).Formula = "=WinDoors!" & r1.
.Offset(1, noActiveStyle * 8).Address
                rSumWinDoors.Offset(0, 0).HorizontalAlignment =
xlLeft

                r1.Offset(0, 1 + noActiveStyle * 8).Value = "Number:"
                r1.Offset(1, 1 + noActiveStyle * 8).NumberFormat = "#
,##0"

                rSumWinDoors.Offset(0, 1).Formula = "=WinDoors!" & r1.
.Offset(1, 1 + noActiveStyle * 8).Address
                rSumWinDoors.Offset(0, 1).NumberFormat = "#,##0"

```

Appendix 15 - Module "ObjImport"

```

' Find orientation of window/door based on direction ✓
of wall
    r1.Offset(0, 2 + noActiveStyle * 8).Value = ✓
"Orientation:"
    With r1.Offset(1, 2 + noActiveStyle * 8)
        Select Case direction
            Case "North"
                .Value = "n"
            Case "East"
                .Value = "ø"
            Case "South"
                .Value = "s"
            Case "West"
                .Value = "v"
        End Select
    End With
    rSumWinDoors.Offset(0, 2).Formula = "=WinDoors!" & r1✓
.Offset(1, 2 + noActiveStyle * 8).Address

    r1.Offset(0, 3 + noActiveStyle * 8).Value = "Area ✓
[m2]:"
    r1.Offset(1, 3 + noActiveStyle * 8).NumberFormat = "#✓
,##0.00"
    rSumWinDoors.Offset(0, 3).Formula = "=WinDoors!" & r1✓
.Offset(1, 3 + noActiveStyle * 8).Address

    r1.Offset(0, 4 + noActiveStyle * 8).Value = "U -Value✓
[W/m2K]:"
    r1.Offset(1, 4 + noActiveStyle * 8).Value = ✓
uvalueWinDoor
    rSumWinDoors.Offset(0, 4).Formula = "=WinDoors!" & r1✓
.Offset(1, 4 + noActiveStyle * 8).Address
    r1.Offset(1, 4 + noActiveStyle * 8).NumberFormat = "#✓
,##0.00"

    r1.Offset(0, 5 + noActiveStyle * 8).Value = "Glazing ✓
fraction [%]:"
    r1.Offset(1, 5 + noActiveStyle * 8).NumberFormat = "#✓
,##0.00"
    If objIfcWinDoors.Type = "IfcWindow" Then
        rSumWinDoors.Offset(0, 5).Formula = "=WinDoors!" ✓
& r1.Offset(1, 5 + noActiveStyle * 8).Address
    End If

    r1.Offset(0, 6 + noActiveStyle * 8).Value = "Total ✓
perimeter [m]:"
    r1.Offset(1, 6 + noActiveStyle * 8).NumberFormat = "#✓
,##0.00"
    rSumWinDoors.Offset(0, 6).Formula = "=WinDoors!" & r1✓
.Offset(1, 6 + noActiveStyle * 8).Address

    r1.Offset(3, noActiveStyle * 8).Value = "Name"
    r1.Offset(3, 1 + noActiveStyle * 8).Value = "Width ✓
[m]"
    r1.Offset(3, 2 + noActiveStyle * 8).Value = "Height ✓
[m]"
    r1.Offset(3, 3 + noActiveStyle * 8).Value = ✓
"Perimeter [m]"
    r1.Offset(3, 4 + noActiveStyle * 8).Value = "Area ✓
[m2]"
    r1.Offset(3, 5 + noActiveStyle * 8).Value = "Glazing ✓
fraction [%]:"

    Exit Do
End If
noActiveStyle = noActiveStyle + 1
Loop While noActiveStyle <= noStyleNames

' Find window/door name

```

Appendix 15 - Module "ObjImport"

```

strPropertyName = objIfcWinDoors.Attributes("Name").Value
' Find window/door width
strWidth = objIfcWinDoors.Attributes("OverallWidth").Value
strWidth = strWidth / 1000
' Find window/door height
strHeight = objIfcWinDoors.Attributes("OverallHeight").Value
strHeight = strHeight / 1000
' Calculate perimeter of window/door opening
strPerimeter = 2 * strWidth + 2 * strHeight
strArea = strWidth * strHeight

'Find Lining Thickness of windows from style property in order to calculate Glazing fraction
glazingFraction = 0
For Each objIfcRelDefinesByProperties In
objIfcRelDefinesByType.Item(1).Attributes("RelatingType").Value.Attributes(
"HasPropertySets").Value
    If objIfcRelDefinesByProperties.Type =
"IfcWindowLiningProperties" Then
        LiningThickness = objIfcRelDefinesByProperties.
Attributes("LiningThickness").Value
        LiningThickness = LiningThickness / 1000
        LiningThickness = LiningThickness * 2 * strHeight +
LiningThickness * 2 * (strWidth - 2 * LiningThickness)
        glazingFraction = (strArea - LiningThickness) /
strArea
    End If
Next objIfcRelDefinesByProperties

' Write values to Excel

r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), noActiveStyle
* 8).Value = strPropertyName

r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 1 +
noActiveStyle * 8).Value = strWidth
r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 1 +
noActiveStyle * 8).NumberFormat = "#,##0.00"

r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 2 +
noActiveStyle * 8).Value = strHeight
r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 2 +
noActiveStyle * 8).NumberFormat = "#,##0.00"

r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 3 +
noActiveStyle * 8).Value = strPerimeter
r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 3 +
noActiveStyle * 8).NumberFormat = "#,##0.00"
r1.Offset(1, 6 + noActiveStyle * 8) = "=SUM(" & r1.Offset(4,
3 + noActiveStyle * 8).Address _
& ":" & r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 3 + noActiveStyle * 8).
Address & ")"
r1.Offset(1, 1 + noActiveStyle * 8).Value = "=" &
noWinDoorsInStyle(noActiveStyle)

r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 4 +
noActiveStyle * 8).Value = strArea
r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 4 +
noActiveStyle * 8).NumberFormat = "#,##0.00"
r1.Offset(1, 3 + noActiveStyle * 8).Formula = "=" & r1.Offset
(3 + noWinDoorsInStyle(noActiveStyle), 4 + noActiveStyle * 8).Address

If objIfcWinDoors.Type = "IfcWindow" Then
    r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 5 +
noActiveStyle * 8).Value = glazingFraction
    r1.Offset(3 + noWinDoorsInStyle(noActiveStyle), 5 +
noActiveStyle * 8).NumberFormat = "#,##0.00"
    r1.Offset(1, 5 + noActiveStyle * 8).Value =
glazingFraction

```

Appendix 15 - Module "ObjImport"

```

        End If

        rWinDoor = r1.Offset(3 + noWinDoorsInStyle(noActiveStyle),
noActiveStyle * 8)

        If Not findRelatedWinDoors Is Nothing Then
            findRelatedWinDoors = Union(findRelatedWinDoors,
rWinDoor)
        Else
            findRelatedWinDoors = rWinDoor
        End If

    End If
Next objIfcRelFillsElement

End If

NextWinDoor:

Next objIfcRelVoidsElement

ActiveSheet.Columns("A:BA").AutoFit()

End Function

' -----
' Get Slabs
' -----

Private Sub GetSlabs()
    Dim IsSlabExternal As Boolean
    Dim uvalueSlab As Single

    Dim objIfcSlab, objIfcShapeRepresentation, objIfcRelDefinesByProperties,
objIfcPropertySingleValue As IFCsvr.Entity
    Dim objRelAssociatesMaterial As IFCsvr.Entities
    Dim strLayerSetName(20, 2) As String
    Dim noSlabsInLayerSet(20) As Integer
    Dim tempLayerSetName As String
    Dim noLayerSetNames, noActiveLayerSet As Integer
    Dim objAtt As Object
    Dim varAtt As Object
    Dim r1, rWinDoors, rTemp As Excel.Range
    Dim strPropertyName As String
    Dim strPropertyType As String
    Dim strPropertyType As String
    Dim strWidth As Single
    Dim strHeight As Single
    Dim strArea As Single
    Dim intX(2) As Single
    Dim intY(2) As Single
    Dim runIt(2) As Integer
    Dim placement As String
    Dim slabArea As Single
    Dim slabPerimeter As Single

    Worksheets("Slabs").Activate()
    Range("A2").Value = "Slabs"
    r1 = ActiveSheet.Range("A5")

    noLayerSetNames = 0

    ' Find information for each slab
    For Each objIfcSlab In objDesign.FindObjects("IfcSlab")
        IsSlabExternal = False

```

Appendix 15 - Module "ObjImport"

```

uvalueSlab = 0.0#

' Find U-value and if slab is internal or external
For Each objIfcRelDefinesByProperties In objIfcSlab.GetUsedIn(
    "IfcRelDefinesByProperties", "RelatedObjects")
    With objIfcRelDefinesByProperties.Attributes(
        "RelatingPropertyDefinition")
        If .Value.Type = "IfcPropertySet" Then
            If .Value.Attributes("Name").Value = "Pset_SlabCommon" Then
                For Each objIfcPropertySingleValue In .Value.Attributes(
                    "HasProperties").Value
                    If objIfcPropertySingleValue.Attributes("Name").Value
                        = "ThermalTransmittance" Then
                        uvalueSlab = objIfcPropertySingleValue.Attributes(
                            "NominalValue").Value
                        ElseIf objIfcPropertySingleValue.Attributes("Name").
                            Value = "IsExternal" Then
                            IsSlabExternal = objIfcPropertySingleValue.
                                Attributes("NominalValue").Value
                        End If
                    Next objIfcPropertySingleValue
                End If
            End If
        End With

    Next objIfcRelDefinesByProperties

' Only external slabs will be analysed to save time
If IsSlabExternal Then

    ' Find placement of slab using "GetSlabPlaceAreaPerimeter" function
    SlabPlacement.GetSlabPlaceAreaPerimeter(objIfcSlab, placement,
        slabArea, slabPerimeter)

    ' Find layer name of material composition
    objRelAssociatesMaterial = objIfcSlab.GetUsedIn(
        "IfcRelAssociatesMaterial", "RelatedObjects")
    tempLayerSetName = objRelAssociatesMaterial.Item(1).Attributes(
        "RelatingMaterial") _
        .Value.Attributes("ForLayerSet").
        Value.Attributes("LayerSetName").Value()

    ' Find out if slab composition exists or new must be created.
    ' This analysis accounts for the placement of the slab
    noActiveLayerSet = 0
    Do
        If tempLayerSetName = strLayerSetName(noActiveLayerSet, 1) And
            placement = strLayerSetName(noActiveLayerSet, 2) Then
            ' Slab composition exists, current slab will be added
            noSlabsInLayerSet(noActiveLayerSet) = noSlabsInLayerSet
            (noActiveLayerSet) + 1
            Exit Do
        ElseIf noActiveLayerSet = noLayerSetNames Then
            ' Slab composition is new and will be added
            strLayerSetName(noActiveLayerSet, 1) = tempLayerSetName
            strLayerSetName(noActiveLayerSet, 2) = placement
            noLayerSetNames = noLayerSetNames + 1

            noSlabsInLayerSet(noActiveLayerSet) = 1

            ' Setup composition information in Excel
            rSumSlabs.EntireRow.Insert()
            rSumSlabs = rSumSlabs.Offset(-1, 0)
            rSumSlabs.Offset(0, 0).EntireRow.Font.Bold = False

            r1.Offset(0, noActiveLayerSet * 6).Value = "Slab type:"
            r1.Offset(1, noActiveLayerSet * 6).Value = tempLayerSetName
            rSumSlabs.Offset(0, 0).Formula = "=Slabs!" & r1.Offset(1,
                noActiveLayerSet * 6).Address

```

Appendix 15 - Module "ObjImport"

```

        rSumSlabs.Offset(0, 0).HorizontalAlignment = xlLeft

        r1.Offset(0, 1 + noActiveLayerSet * 6).Value = "Total net      ✓
area [m2]:"
        r1.Offset(1, 1 + noActiveLayerSet * 6).NumberFormat = "#,##0.✓
00"
        rSumSlabs.Offset(0, 1).Formula = "=Slabs!" & r1.Offset(1, 1 +✓
noActiveLayerSet * 6).Address

        r1.Offset(0, 2 + noActiveLayerSet * 6).Value = "U -Value [W/ ✓
m2K]:"
        r1.Offset(1, 2 + noActiveLayerSet * 6).Value = uvalueSlab
        r1.Offset(1, 2 + noActiveLayerSet * 6).NumberFormat = "#,##0.✓
00"
        rSumSlabs.Offset(0, 2).Formula = "=Slabs!" & r1.Offset(1, 2 +✓
noActiveLayerSet * 6).Address

        r1.Offset(0, 3 + noActiveLayerSet * 6).Value = "b-factor:"
        Select Case placement
            Case "BelowGround"
                r1.Offset(1, 3 + noActiveLayerSet * 6).Value = "0.7"
            Case "AboveGround"
                r1.Offset(1, 3 + noActiveLayerSet * 6).Value = "1"
        End Select
        rSumSlabs.Offset(0, 3).Formula = "=Slabs!" & r1.Offset(1, 3 +✓
noActiveLayerSet * 6).Address

        If placement = "BelowGround" Then
            r1.Offset(0, 4 + noActiveLayerSet * 6).Value = "Gross      ✓
perimeter [m]:"
            r1.Offset(1, 4 + noActiveLayerSet * 6).NumberFormat = "#,✓
##0.00"
            rSumSlabs.Offset(0, 4).Formula = "=Slabs!" & r1.Offset(1,✓
4 + noActiveLayerSet * 6).Address
        End If

        r1.Offset(3, noActiveLayerSet * 6).Value = "Slab name"
        r1.Offset(3, 1 + noActiveLayerSet * 6).Value = "Placement"
        r1.Offset(3, 2 + noActiveLayerSet * 6).Value = "Area [m2]"
        r1.Offset(3, 3 + noActiveLayerSet * 6).Value = "Perimeter [m]✓
"

        Exit Do
    End If
    noActiveLayerSet = noActiveLayerSet + 1
    Loop While noActiveLayerSet <= noLayerSetNames

    ' Find name of slab
    strPropertyName = objIfcSlab.Attributes("Name").Value

    ' Write information on slab to Excel

    r1.Offset(3 + noSlabsInLayerSet(noActiveLayerSet), noActiveLayerSet *✓
6).Value = strPropertyName
    r1.Offset(3 + noSlabsInLayerSet(noActiveLayerSet), 1 +
noActiveLayerSet * 6).Value = placement

    r1.Offset(3 + noSlabsInLayerSet(noActiveLayerSet), 2 +
noActiveLayerSet * 6).Value = slabArea
    r1.Offset(3 + noSlabsInLayerSet(noActiveLayerSet), 2 +
noActiveLayerSet * 6).NumberFormat = "#,##0.00"
    r1.Offset(1, 1 + noActiveLayerSet * 6).Formula = "=SUM(" & r1.Offset ✓
(4, 2 + noActiveLayerSet * 6).Address & ":" & _
        r1.Offset(3 + noSlabsInLayerSet(noActiveLayerSet), ✓
2 + noActiveLayerSet * 6).Address & ")"

    If placement = "BelowGround" Then
        r1.Offset(3 + noSlabsInLayerSet(noActiveLayerSet), 3 +
noActiveLayerSet * 6).Value = slabPerimeter
        r1.Offset(3 + noSlabsInLayerSet(noActiveLayerSet), 3 +

```

Appendix 15 - Module "ObjImport"

```
noActiveLayerSet * 6).NumberFormat = "#,##0.00"  
    r1.Offset(1, 4 + noActiveLayerSet * 6).Formula = "=SUM(" & r1.  
Offset(4, 3 + noActiveLayerSet * 6).Address & ":" & _  
    r1.Offset(3 + noSlabsInLayerSet  
(noActiveLayerSet), 3 + noActiveLayerSet * 6).Address & ")"  
    End If  
  
    End If  
Next objIfcSlab  
  
ActiveSheet.Columns("A:BA").AutoFit()  
  
End Sub  
  
' -----  
' End of module "ObjImport"  
' -----
```

Appendix 15 - Module "ResetLayout"

```
' =====
'
' START OF MODULE "ResetLayout"
'
'
' Script created by Niels Treldal, s021820, Technical University of Denmark
' =====
'
' -----
' Function to reset layout
' -----

Public Sub ResetContent()
    Worksheets("Spaces").Activate()
    Range("A1:BB400").Activate()
    Selection.ClearContents()
    Selection.ClearComments()
    Range("A1").Activate()

    Worksheets("Walls").Activate()
    Range("A1:BB400").Activate()
    Selection.ClearContents()
    Range("A1").Activate()

    Worksheets("WinDoors").Activate()
    Range("A1:BB400").Activate()
    Selection.ClearContents()
    Range("A1").Activate()

    Worksheets("Slabs").Activate()
    Range("A1:BB400").Activate()
    Selection.ClearContents()
    Range("A1").Activate()

    Worksheets("Summation").Activate()
    Range("A12:G150").Select()
    Selection.ClearContents()
    Selection.HorizontalAlignment = xlCenter
    Selection.Interior.ColorIndex = xlNone
    Selection.Font.Bold = False
    Selection.Font.Size = 9

    Range("A14").Select()
    ActiveCell.FormulaR1C1 = "Spaces"
    With Selection.Interior
        .ColorIndex = 4
        .Pattern = xlSolid
    End With
    Selection.Font.Bold = True
    Selection.Font.Size = 11
    With Range("A15")
        .EntireRow.Font.Bold = True
        .Value = "Zone name:"
        .Offset(0, 1).Value = "Gross area [m2]:"
        .Offset(0, 2).Value = "Net area [m2]:"
        .Offset(0, 3).Value = "Net volumen [m3]:"
        .Offset(0, 4).Value = "Max air flow rate [l/s/m2]:"
        .Offset(0, 5).Value = "Min air flow rate [l/s/m2]:"
    End With

    Range("A18").Select()
    ActiveCell.FormulaR1C1 = "Walls"
    With Selection.Interior
        .ColorIndex = 33
        .Pattern = xlSolid
    End With
    Selection.Font.Bold = True
```

Appendix 15 - Module "ResetLayout"

```
Selection.Font.Size = 11
With Range("A19")
    .EntireRow.Font.Bold = True
    .Value = "Wall type:"
    .Offset(0, 1).Value = "Total area [m2]:"
    .Offset(0, 2).Value = "U-value [W/mK]:"
    .Offset(0, 3).Value = "b-factor:"
End With

Range("A22").Select()
ActiveCell.FormulaR1C1 = "Windows/Doors"
With Selection.Interior
    .ColorIndex = 44
    .Pattern = xlSolid
End With
Selection.Font.Bold = True
Selection.Font.Size = 11
With Range("A23")
    .EntireRow.Font.Bold = True
    .Value = "Window/Door type:"
    .Offset(0, 1).Value = "Number:"
    .Offset(0, 2).Value = "Orientation:"
    .Offset(0, 3).Value = "Area [m2]:"
    .Offset(0, 4).Value = "U-value [W/mK]:"
    .Offset(0, 5).Value = "Glazing fraction [%]:"
    .Offset(0, 6).Value = "Perimeter [m]:"
End With

Range("A26").Select()
ActiveCell.FormulaR1C1 = "Slabs"
With Selection.Interior
    .ColorIndex = 3
    .Pattern = xlSolid
End With
Selection.Font.Bold = True
Selection.Font.Size = 11
With Range("A27")
    .EntireRow.Font.Bold = True
    .Value = "Wall type:"
    .Offset(0, 1).Value = "Net area [m2]:"
    .Offset(0, 2).Value = "U-value [W/mK]:"
    .Offset(0, 3).Value = "b-factor:"
    .Offset(0, 4).Value = "Gross perimeter [m]:"
End With

End Sub
```

```
' -----
'   End of module "ResetLayout"
' -----
```

Appendix 15 - Module "SlabPlacement"

```

' =====
' START OF MODULE "SlabPlacement"
'
'
' Script created by Niels Treldal, s021820, Technical University of Denmark
' =====
' -----
' Function to find area and placement of slabs
' -----

Public Function GetSlabPlaceAreaPerimeter(ByVal objIfcSlab As Object, ByVal placement As String, ByVal slabArea As Single, ByVal slabPerimeter As Single)
    Dim objIfcShapeRepresentation, objIfcExtrudedAreaSolid As IFCsvr.Entity
    Dim objIfcAxis2Placement3D As IFCsvr.Entity
    Dim objRelAssociatesMaterial, objIfcWall, objMaterial As IFCsvr.Entity
    Dim objAtt As Object
    Dim coorsSlab(100, 3), tempCoors2(1, 3) As Single
    Dim noCoors, i As Integer
    Dim vecSlab(2, 2) As Single
    Dim vecLength As Single
    Dim tempCoors As Object
    Dim groundLevel, slabLevel, foundationThickness As Single
    Dim closedCorner As Boolean

    ' Find slab area and check if bottom of slab is below or above ground level
    For Each objIfcShapeRepresentation In objIfcSlab.Attributes("Representation").Value.Attributes("Representations").Value.FindObjects("IfcShapeRepresentation")

        If objIfcShapeRepresentation.Attributes("RepresentationType").Value = "SweptSolid" Then

            objIfcExtrudedAreaSolid = objIfcShapeRepresentation.Attributes("Items").Value.Item("IfcExtrudedAreaSolid")
            If objIfcExtrudedAreaSolid.Attributes("SweptArea").Value.Type = "IfcArbitraryClosedProfileDef" Then

                ' Find net area from perimeter of slab
                slabArea = 0
                i = 0
                For Each objAtt In objIfcExtrudedAreaSolid.Attributes("SweptArea").Value.Attributes("OuterCurve").Value.Attributes("Points").Value
                    tempCoors = objAtt.Attributes(1).Value
                    coorsSlab(i, 0) = tempCoors(0) / 1000
                    coorsSlab(i, 1) = tempCoors(1) / 1000
                    coorsSlab(i, 2) = 0
                    i = i + 1
                Next objAtt
                noCoors = i

                'Using the method difined here to calculate the area:
                ' http://mathworld.wolfram.com/PolygonArea.html
                sngArea = 0
                For i = 0 To noCoors - 2
                    slabArea = slabArea + coorsSlab(i, 0) * coorsSlab(i + 1, 1) - coorsSlab(i + 1, 0) * coorsSlab(i, 1)
                Next
                slabArea = Abs(slabArea / 2)

                ' Transform the first set of local coors to global coors to find elevation of slab
                tempCoors2(0, 0) = coorsSlab(0, 0) * 1000
                tempCoors2(0, 1) = coorsSlab(0, 1) * 1000
                tempCoors2(0, 2) = coorsSlab(0, 2)
                Local2Global.GetLocal2Global(tempCoors2, 1,

```

Appendix 15 - Module "SlabPlacement"

```

objIfcExtrudedAreaSolid.Attributes("Position").Value, objIfcSlab.Attributes(
"ObjectPlacement").Value)

' Find ground level
groundLevel = Worksheets("Summation").Range("C8").Value

' Find out of slab is over or under ground
slabLevel = tempCoors2(0, 2) / 1000
If slabLevel <= groundLevel Then
    placement = "BelowGround"
Else
    placement = "AboveGround"
End If

' Find length of gross perimeter
If placement = "BelowGround" Then
    ' Starts by calculating the thickness of the lowest wall in
the building
    objIfcWall = objDesign.FindObjectByP21Id(lowestWall(0))
    For Each objRelAssociatesMaterial In objIfcWall.GetUsedIn(
"IfcRelAssociatesMaterial", "RelatedObjects")
        For Each objMaterial In objRelAssociatesMaterial.
Attributes("RelatingMaterial").Value _
Attributes("ForLayerSet").Value.
Attributes("MaterialLayers").Value
            foundationThickness = objMaterial.Attributes(
"LayerThickness").Value / 1000
        Next objMaterial
    Next objRelAssociatesMaterial

    ' The calculates the length of the perimeter
    slabPerimeter = 0
    closedCorner = False
    For i = 0 To noCoors - 2
        ' Find current and next vector of perimeter
        vecSlab(0, 0) = coorsSlab(i + 1, 0) - coorsSlab(i, 0)
        vecSlab(0, 1) = coorsSlab(i + 1, 1) - coorsSlab(i, 1)

        If i = noCoors - 2 Then
            vecSlab(1, 0) = coorsSlab(i - noCoors + 3, 0) -
coorsSlab(i - noCoors + 2, 0)
            vecSlab(1, 1) = coorsSlab(i - noCoors + 3, 1) -
coorsSlab(i - noCoors + 2, 1)
        Else
            vecSlab(1, 0) = coorsSlab(i + 2, 0) - coorsSlab(i + 1
, 0)
            vecSlab(1, 1) = coorsSlab(i + 2, 1) - coorsSlab(i + 1
, 1)
        End If

        ' Check if primiter includes a closed corner and add 2 *
wall thickness
        If vecSlab(0, 0) > 0 And vecSlab(1, 1) > 0 Then
            closedCorner = True
        End If

        If vecSlab(0, 1) > 0 And vecSlab(1, 0) < 0 Then
            closedCorner = True
        End If

        If vecSlab(0, 0) < 0 And vecSlab(1, 1) < 0 Then
            closedCorner = True
        End If

        If vecSlab(0, 1) < 0 And vecSlab(1, 0) > 0 Then
            closedCorner = True
        End If
    Next i
    slabPerimeter = slabPerimeter + 2 * foundationThickness * closedCorner
End If

```

Appendix 15 - Module "SlabPlacement"

```

        'MsgBox "i: " & i & " closedCorner: " & closedCorner
        If closedCorner = True Then
            slabPerimeter = slabPerimeter + 2 *
foundationThickness
        End If

        ' Check if primiter includes a open corner and subtract 1
* wall thickness
        closedCorner = True
        If vecSlab(0, 0) > 0 And vecSlab(1, 1) < 0 Then
            closedCorner = False
        End If

        If vecSlab(0, 1) > 0 And vecSlab(1, 0) > 0 Then
            closedCorner = False
        End If

        If vecSlab(0, 0) < 0 And vecSlab(1, 1) > 0 Then
            closedCorner = False
        End If

        If vecSlab(0, 1) < 0 And vecSlab(1, 0) < 0 Then
            closedCorner = False
        End If

        If closedCorner = False Then
            slabPerimeter = slabPerimeter - foundationThickness
        End If

        ' Add net length of perimeter
        vecLength = Sqr(vecSlab(0, 0) ^ 2 + vecSlab(0, 1) ^ 2)
        slabPerimeter = slabPerimeter + vecLength
        closedCorner = False
    Next
End If

Else
    ' If slab included holes, it can not be calculated for now
    MsgBox("Slab '" & objIfcSlab.Attributes("Name").Value & "'
include holes, which are not currently supported", vbCritical)
    placement = "Unknown"
    slabArea = 0
    Exit Function
End If

Else
    ' If slab is defined by another geometry representation, it is most
likely not a plain slab and then not supported
    MsgBox("Slab '" & objIfcSlab.Attributes("Name").Value & "' is not a
plain slab. Only plain surfaces are supported for now in this script",
vbCritical)
    placement = "Unknown"
    slabArea = 0
    Exit Function
End If
Next objIfcShapeRepresentation

End Function

' -----
'   End of module "SlabPlacement"
' -----

```

Appendix 15 - Module "WallDirection"

```

' =====
'
' START OF MODULE "WallDirection"
'
'
' Script created by Niels Trelldal, s021820, Technical University of Denmark
' =====

' -----
' Function to find direction and placement of walls
' -----

Public Function GetWallDirAndPlace(ByVal objIfcWall As Object, ByVal direction As String, ByVal wallDepth As Single)
    Dim coorsWall(3, 3), coorsSpace(20, 3) As Double
    Dim BoundaryV_1(3), BoundaryV_2(3) As Double
    Dim BoundaryN As Object
    Dim WallV_X, WallV_Y, BoundaryN_X, BoundaryN_Y As Double
    Dim noCoors As Integer
    Dim lengthWall, lengthBoundaryN As Double
    Dim tempCoors As Object
    Dim objAtt As Object
    Dim objIfcShapeRepresentation, objIfcPolyline, objIfcLocalPlacement As Object
    Dim objIfcAxis2Placement3D, objIfcRelSpaceBoundary, objIfcSpace, objEntity As Object
    Dim angleWall, angleBoundaryN, Theta As Double
    Dim i As Integer
    Dim SpaceOnRightSide, SpaceOnLeftSide As Boolean
    Dim strPropertyType As String
    Dim Xmax, Xmin, Ymax, Ymin As Double
    Dim groundLevel, wallLevel As Single

    'Debug
    Dim tael As Integer
    tael = 0

    ' Find wall axis
    For Each objIfcShapeRepresentation In objIfcWall.Attributes("Representation").Value.Attributes("Representations").Value.FindObjects("IfcShapeRepresentation")
        If objIfcShapeRepresentation.Attributes("RepresentationType").Value = "Curve2D" Then
            i = 0
            For Each objAtt In objIfcShapeRepresentation.Attributes("Items").Value.Item("IfcPolyline").Attributes("Points").Value
                tempCoors = objAtt.Attributes(1).Value
                coorsWall(i, 0) = tempCoors(0)
                coorsWall(i, 1) = tempCoors(1)
                coorsWall(i, 2) = 0
                i = i + 1
            Next objAtt
            noCoors = 2
        End If
    Next objIfcShapeRepresentation

    ' Transform the local coors to global coors
    objIfcAxis2Placement3D = Nothing
    Local2Global.GetLocal2Global(coorsWall, noCoors, objIfcAxis2Placement3D, objIfcWall.Attributes("ObjectPlacement").Value)

    ' Find wall placement above or under ground
    ' Find ground level
    groundLevel = Worksheets("Summation").Range("C8").Value

```

Appendix 15 - Module "WallDirection"

```

' Find out if wall is below or above 2 meters under ground
wallLevel = coorsWall(0, 2) / 1000
If wallLevel <= groundLevel - 2.0# Then
    ' If wall is underground the depth UNDER 2 meters is defined.
    wallDepth = wallLevel + 2.0# - groundLevel
Else
    wallDepth = 0
End If

' Identify the lowest wall in the building by is P21ID for use to calculate slab perimeter
If wallLevel < lowestWall(1) Then
    lowestWall(0) = objIfcWall.P21ID
    lowestWall(1) = wallLevel
End If

' Find angle between north (0,1) and wall axis
' Define wall axis vector
WallV_X = coorsWall(1, 0) - coorsWall(0, 0)
WallV_Y = coorsWall(1, 1) - coorsWall(0, 1)

' Debug
Debug.Print("coorsWall0: " & coorsWall(0, 0) & " , " & coorsWall(0, 1))
Debug.Print("coorsWall1: " & coorsWall(0, 1) & " , " & coorsWall(1, 1))

lengthWall = Sqr(WallV_X ^ 2 + WallV_Y ^ 2)

' Determinant between unit vector for north (0,1,0) and wall axis is used to find angle
' det(WallV,NorthV) = lengthWall*lengthNorth*sin(angleWall)
angleWall = Excel.WorksheetFunction.Asin(WallV_X / lengthWall)

' Correct values to be in range -Pi to +Pi
If angleWall < 0 And WallV_Y < 0 Then
    angleWall = -Pi - angleWall
ElseIf angleWall <= 0 And WallV_Y > 0 Then
    angleWall = Pi - angleWall
End If

' Convert from radian to degrees
angleWall = angleWall * 180 / Pi

' ** Find relativ location af spaces by analysing space boundaries **
' Reset values
SpaceOnRightSide = False
SpaceOnLeftSide = False

' The function GetUsedIn does not work for IfcRelSpaceBoundary, hence all IfcRelSpaceBoundary has to been evaluated
For Each objIfcRelSpaceBoundary In objDesign.FindObjects("IfcRelSpaceBoundary")
    ' Check if SpaceBoundary relates to BuildingElement or VirtualElement
    If Not objIfcRelSpaceBoundary.Attributes("RelatedBuildingElement").IsNull Then
        ' Check if SpaceBoundary relates to the wall in question
        If objIfcRelSpaceBoundary.Attributes("RelatedBuildingElement").Value.P21ID = objIfcWall.P21ID Then
            tael = tael + 1

            ' Define related information of related space
            objIfcSpace = objIfcRelSpaceBoundary.Attributes("RelatingSpace").Value
            objIfcLocalPlacement = objIfcSpace.Attributes("ObjectPlacement").Value
            objIfcAxis2Placement3D = objIfcRelSpaceBoundary.Attributes("ConnectionGeometry")._

```

Appendix 15 - Module "WallDirection"

```

        Value.Attributes("SurfaceOnRelatingElement").Value.Attributes(
"BasisSurface")._
        Value.Attributes("Position").Value

        ' Define surface of space boundary
        objIfcPolyline = objIfcRelSpaceBoundary.Attributes(
"ConnectionGeometry")._
        Value.Attributes("SurfaceOnRelatingElement").Value.Attributes(
"OuterBoundary")._
        Value.Attributes("Segments").Value.Item(1).Attributes(
"ParentCurve").Value

        ' Capture coordinates of surface points
        For i = 1 To objIfcPolyline.Attributes("Points").Value.Count
            tempCoors = objIfcPolyline.Attributes("Points").Value.Item(i)
        .Attributes(1).Value
            coorsSpace(i - 1, 0) = tempCoors(0)
            coorsSpace(i - 1, 1) = tempCoors(1)
            coorsSpace(i - 1, 2) = tempCoors(2)
        Next
        noCoors = objIfcPolyline.Attributes("Points").Value.Count

        ' Debug
        Debug.Print("-----T&L: " & tael)
        Debug.Print(" objIfcRelSpaceBoundary.p2lid: " &
objIfcRelSpaceBoundary.P2lID & " objIfcWall.p2lid: " & objIfcWall.P2lID & "
objIfcSpace.p2lid: " & objIfcSpace.P2lID)
        For i = 0 To noCoors - 1
            Debug.Print("coorsSpace no:" & i & " " & Round(coorsSpace(i,
0), 3) & " , " & Round(coorsSpace(i, 1), 3) & " , " & Round(coorsSpace(i, 2),
3))
        Next

        ' Convert points from local to global coors
        Local2Global.GetLocal2Global(coorsSpace, noCoors,
objIfcAxis2Placement3D, objIfcLocalPlacement)

        ' Debug
        For i = 0 To noCoors - 1
            Debug.Print("coorsSpaceG no:" & i & " " & Round(coorsSpace(i,
0), 3) & " , " & Round(coorsSpace(i, 1), 3) & " , " & Round(coorsSpace(i, 2),
, 3))
        Next

        ' Find the normal of the boundary surface
        BoundaryN = FindNormal(coorsSpace, noCoors)

        ' Debug
        For i = 0 To 2
            Debug.Print("BoundaryV_1(" & i & ") " & BoundaryV_1(i))
        Next
        For i = 0 To 2
            Debug.Print("BoundaryV_2(" & i & ") " & BoundaryV_2(i))
        Next

        ' Find length of boundary normal
        lengthBoundaryN = Sqr(BoundaryN(0) ^ 2 + BoundaryN(1) ^ 2)

        ' Debug
        Debug.Print("BoundaryN: " & BoundaryN(0) & " , " & BoundaryN(1))
        Debug.Print("WallV: " & WallV_X & " , " & WallV_Y)
        Debug.Print("lengthBoundaryN: " & lengthBoundaryN)
        Debug.Print("lengthWall: " & lengthWall)
        Debug.Print("WallID: " & objIfcWall.P2lID & " WallV: " & WallV_X
& " , " & WallV_Y)
        Debug.Print("SpaceID: " & objIfcSpace.P2lID & " angleBoundaryN: "
& angleBoundaryN)

        ' Determinant between Boundary normal and wall axis is used finde

```

Appendix 15 - Module "WallDirection"

```

angle between the two
' det(BoundaryN,WallV) = (BoundaryN(0) * WallV_Y - BoundaryN(1) *
WallV_X)
angleBoundaryN = (BoundaryN(0) * WallV_Y - BoundaryN(1) *
WallV_X) / (lengthWall * lengthBoundaryN)
angleBoundaryN = Excel.WorksheetFunction.Asin(angleBoundaryN)

' Correct values to be in range -Pi to +Pi
If angleBoundaryN < 0 And BoundaryN(1) < 0 Then
    angleBoundaryN = -Pi - angleBoundaryN
ElseIf angleBoundaryN <= 0 And BoundaryN(1) < 0 Then
    angleBoundaryN = Pi - angleBoundaryN
End If

' Convert from radian to degrees
angleBoundaryN = angleBoundaryN * 180 / Pi

' Debug
Debug.Print("WallID: " & objIfcWall.P21ID & " WallV: " & WallV_X
& " , " & WallV_Y)
Debug.Print("BoundaryN: " & BoundaryN(0) & " , " & BoundaryN(1))
Debug.Print("SpaceID: " & objIfcSpace.P21ID & " angleBoundaryN: "
& angleBoundaryN)

' Find out which side space is on
If 0 < angleBoundaryN And angleBoundaryN < 180 Then
    SpaceOnLeftSide = True
    Debug.Print("SpaceOnLeftSide")
ElseIf -180 < angleBoundaryN And angleBoundaryN < 0 Then
    SpaceOnRightSide = True
    Debug.Print("SpaceOnRightSide")
Else
    'Location of wall relative to the space is unsure and ignored
    'MsgBox "Location of space #" & objIfcSpace.P21ID & " relativ
to wall #" & objIfcWall.P21ID & " could not be found", vbCritical
End If

End If
End If

Next objIfcRelSpaceBoundary

' From information of spaces location, orientation of wall can be found
If SpaceOnRightSide = True And SpaceOnLeftSide = True Then
    direction = "Internal"
ElseIf SpaceOnRightSide = False And SpaceOnLeftSide = False Then
    direction = "Outside"
Else
    Debug.Print("angleWall: " & angleWall)
    If -45 < angleWall And angleWall <= 45 Then
        Debug.Print("1 angleWall")
        If SpaceOnRightSide Then
            direction = "West"
        Else
            direction = "East"
        End If
    ElseIf 45 < angleWall And angleWall <= 135 Then
        Debug.Print("5 angleWall")
        If SpaceOnRightSide Then
            direction = "North"
        Else
            direction = "South"
        End If
    ElseIf 135 < angleWall And angleWall <= 180 Then
        Debug.Print("2 angleWall")
        If SpaceOnRightSide Then
            direction = "East"
        Else
            direction = "West"
        End If
    End If
End If

```

Appendix 15 - Module "WallDirection"

```

        End If
    ElseIf -180 <= angleWall And angleWall <= -135 Then
        Debug.Print("3 angleWall")
        If SpaceOnRightSide Then
            direction = "East"
        Else
            direction = "West"
        End If
    ElseIf -135 < angleWall And angleWall <= -45 Then
        Debug.Print("4 angleWall")
        If SpaceOnRightSide Then
            direction = "South"
        Else
            direction = "North"
        End If
    Else
        MsgBox("Direction of wall #" & objIfcWall.P21ID & " could not be found. Angle of wall is: " & angleWall, vbCritical)
        direction = "Unknown"
    End If
End If

Debug.Print("direction: " & direction)

End Function

' -----
' Function to find normal of a surface defined by points
' -----

Public Function FindNormal(ByVal coors As Object, ByVal noCoors As Integer) As Object
    Dim TempArray(3), Vec1(3), Vec2(3) As Double
    Dim i As Integer

    ' Find normal of surface defined by the coors
    ' n = n1 + n2 + n3... where nX represents the normal at each corner of the surface
    For i = 0 To noCoors - 2
        TempArray(0) = TempArray(0) + coors(i, 1) * coors(i + 1, 2) - coors(i, 2) * coors(i + 1, 1)
        TempArray(1) = TempArray(1) + coors(i, 2) * coors(i + 1, 0) - coors(i, 0) * coors(i + 1, 2)
        TempArray(2) = TempArray(2) + coors(i, 0) * coors(i + 1, 1) - coors(i, 1) * coors(i + 1, 0)
    Next

    TempArray(0) = TempArray(0) + coors(i, 1) * coors(i - noCoors + 1, 2) - coors(i, 2) * coors(i - noCoors + 1, 1)
    TempArray(1) = TempArray(1) + coors(i, 2) * coors(i - noCoors + 1, 0) - coors(i, 0) * coors(i - noCoors + 1, 2)
    TempArray(2) = TempArray(2) + coors(i, 0) * coors(i - noCoors + 1, 1) - coors(i, 1) * coors(i - noCoors + 1, 0)

    ' Returns normal
    FindNormal = TempArray

End Function

' -----
' End of module "WallDirection"
' -----

```

Appendix 16

Measurements standard for Be06

The required values for the areas of components in the building envelope required by Be06 are defined in [DS 418]. This states that the area vary depending on the placement of the insulation in the construction, and whether the building has a heated or unheated basement. The correct measurements to define the areas of components are defined in Figure 1 and Figure 2.

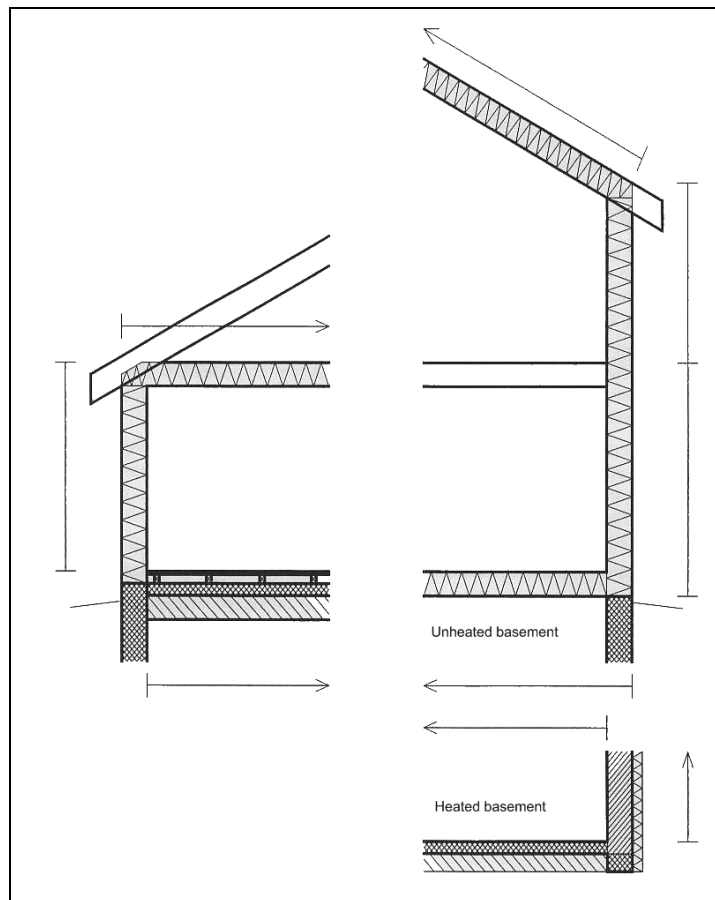


Figure 1. Definitions in a section of areas with heat loss in a building according to [DS 418].

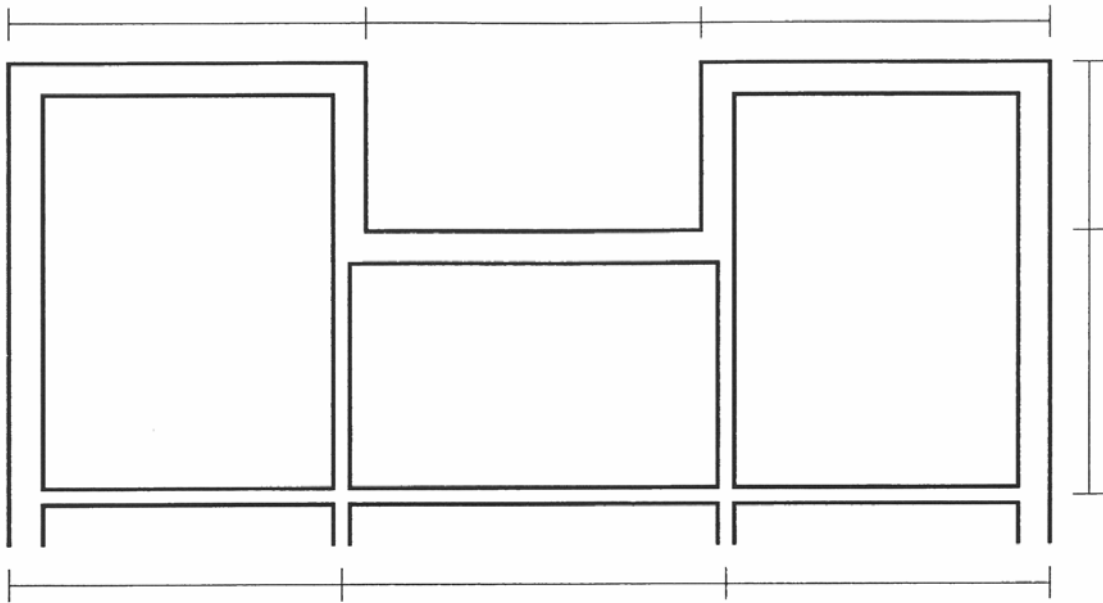


Figure 2. Definitions in a plan of areas with heat loss in a building according to [DS 418].

Be06 further requires the gross area of the heated area in the building. This requires that the area of each space is defined as the area of the space including the area of external walls and half of the area of internal walls. This is the same measurements principle as is Figure 2.

This means that the actual size of the geometry assigned to an object in a BIM would only rarely represent the area required by Be06.

Appendix 17

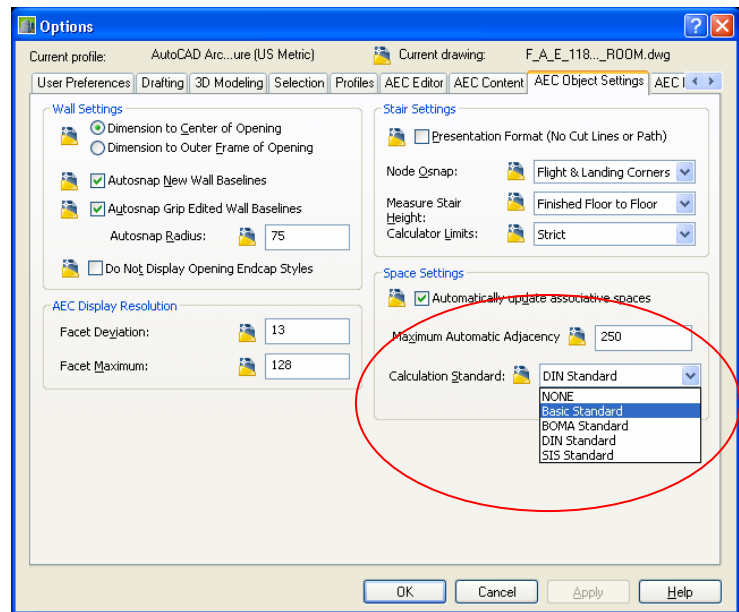
AutoCAD Architecture 2008 and gross + net area of spaces

In the latest editions of BIM applications, the ability to calculate gross and net area of spaces has been much improved. This appendix will go through the steps required for AutoCAD Architecture 2008 (ACA) to calculate the gross and net area and how to get ACA to store these values in the correct property set in an IFC file.

1. Set calculation standard

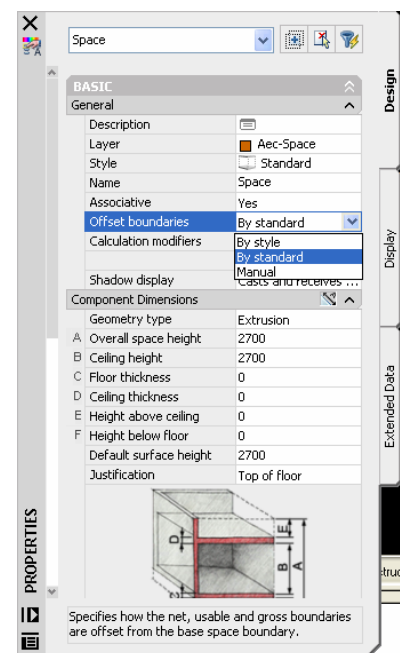
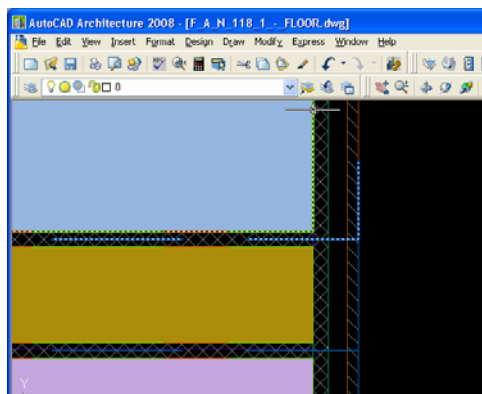
For ACA to know how the areas should be calculated, a standard has to be set. This is done under Format – Options – AEC Object Settings:

Select the “Basic Standard” to follow common guidelines for gross area calculations.



2. Set spaces to follow this standard

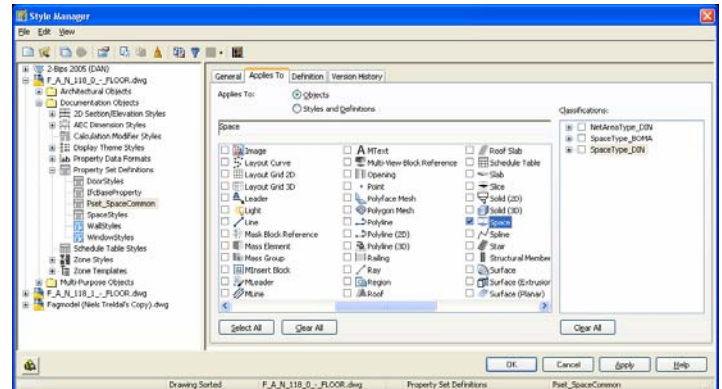
In the properties for each space select “By standard” from the options under “Offset boundaries”. The space boundaries will then be adjusted to follow both internal wall lines (net area) and external and centre wall lines for external and internal walls respectively (gross area):



3. Define IFC property set

The property set in IFC for spaces in which to store gross area and net area is called “Pset_SpaceCommon”.

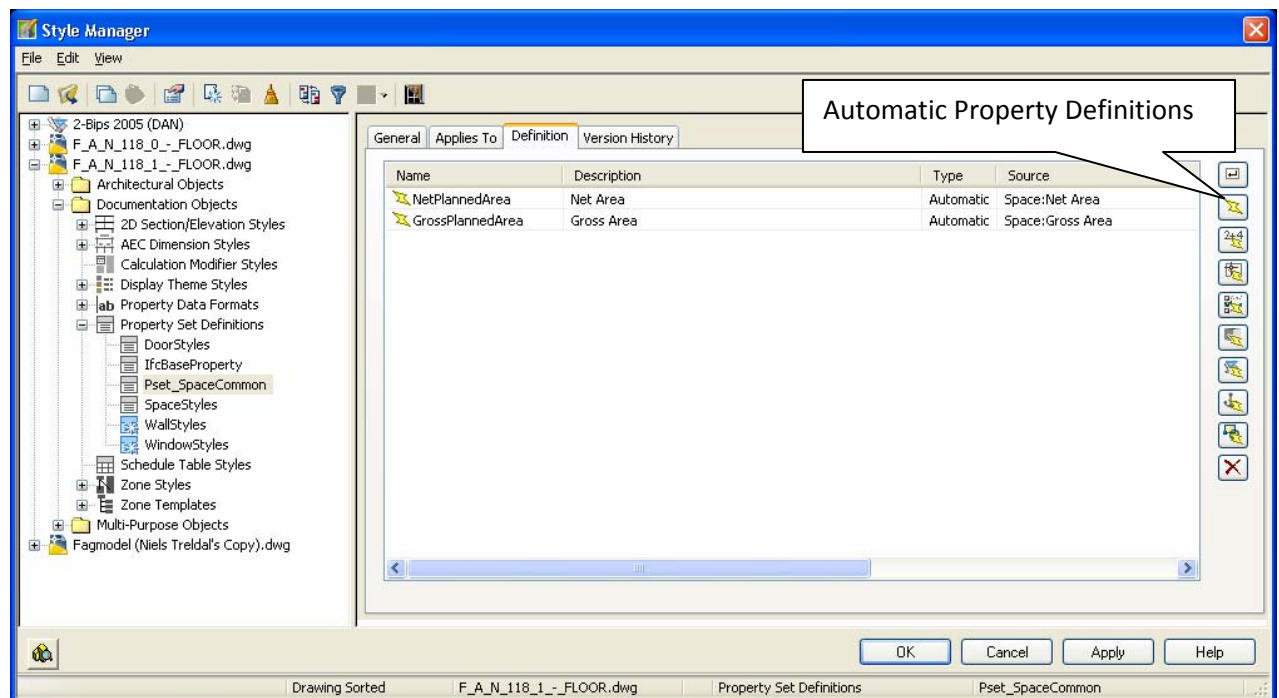
Create a new property set with this name under Format – Style Manager – Documentation Objects – Property Set Definitions. Then apply this property set to space objects under the tab “Applies To”.



4. Define IFC properties

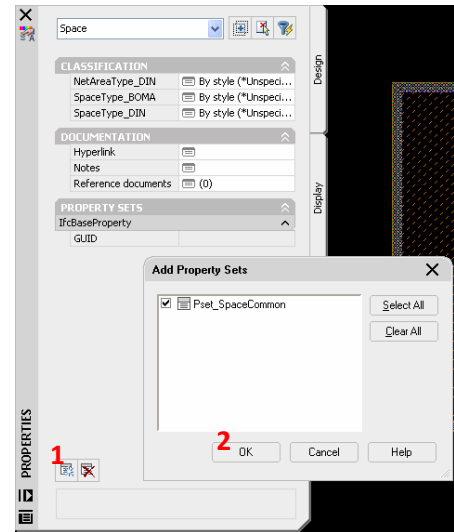
The properties, that needs to be created is called “GrossPlannedArea” and “NetPlannedArea” according to the IFC specifications.

Start by creating two Automatic Property Definitions of GrossArea and NetArea. Then rename these to automatic properties to “GrossPlannedArea” and “NetPlannedArea” respectively.



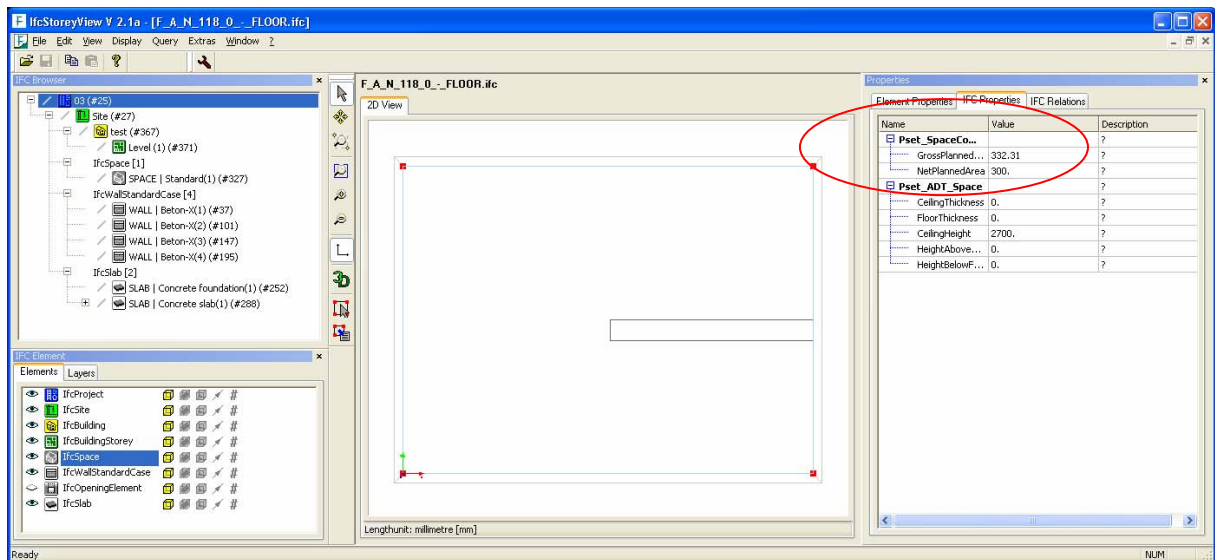
5. Assign property set to spaces

In the properties for each space select the Extended Data and click on the “Add Property Set” button. Now select the “Pset_SpaceCommon” property set. Do so for all required spaces.



This property set will then be exported along with the spaces during IFC export.

An IFC file containing a space with the Pset_SpaceCommon property set viewed in IfcStoreyViewer:



Appendix 18

Creation of Rule Set in Solibri Model Checker

The demonstration in Chapter 5 uses Solibri Model Checker (SMC) to conduct compliance checking for the example of IBD. In order for SMC to be used for compliance checking, new rule sets must be created which reflects the constraints in the requirements model. Only numerical values or type related information can be checked, so requirements defined as descriptions can not be included.

SMC only allow for a library of predefined rule sets to be used. The latest version of SMC is version 4.2 and the rule sets available in this version relates to object validation, consistency control, fire and accessibility code compliance, model comparison and space program validation.

To currently represent client requirements and a space of solution by rule sets will therefore require that this is done based on existing rule sets. From the existing rule sets, possibilities to evaluate individual values in property sets, dimensions of windows and space layouts have been identified as some of the only possibilities useful for defining client requirements and a space of solution in regards to the demands in the demonstration in Chapter 5.

The limitations primarily relates to the demand for indirect linking between e.g. requirements for a space and its related systems and boundaries. SMC does not currently include possibilities to conduct checking on individually selected systems and boundaries via relationships to the space. Combined with the limited possibilities to define required information in the design model this only allow for very few parameters to be checked.

The identified possibilities which were found useful for the demonstration in Chapter 5 are:

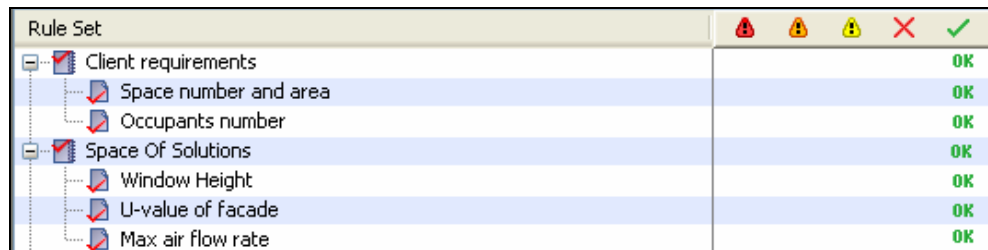
Client requirements

- Check for correct amount of spaces of certain type.
- Check for required area for spaces of certain type.
- Check for correct amount of occupants in each space.

Space of Solutions

- Check for correct U-value of façade wall of certain type.
- Check for correct height of windows of certain type.
- Check for maximum ventilation rate in each space.

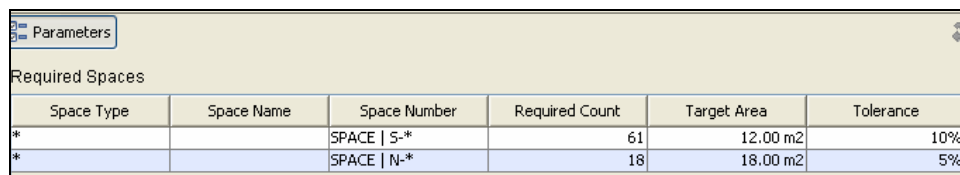
This has led to the creation of two rule sets in SMC as illustrated in Figure 1 where the individual constraints are defined in a total of five rules.



Rule Set		
Client requirements		OK
Space number and area		OK
Occupants number		OK
Space Of Solutions		OK
Window Height		OK
U-value of facade		OK
Max air flow rate		OK

Figure 1. Rule sets in SMC representing the requirements model with client requirements and a space of solutions.

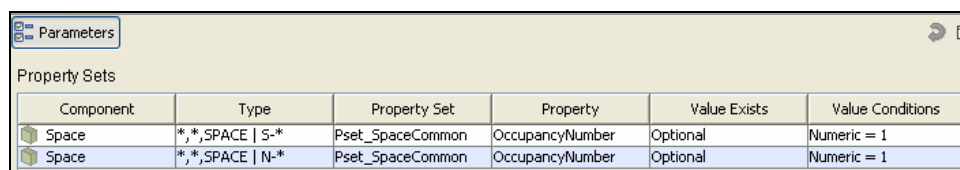
To identify the small offices from the demonstration in Chapter 5, their type identification has been supplemented with a prefix of “N-” or “S-” for identification of northern and southern orientated spaces respectively. The first rule, “Space number and area”, is therefore capable of checking that the required number of spaces are present in the BIM and that each space is within the target area required by the client requirements. Each space is identified by its type (in SMC “Space Number”). The rule is illustrated in Figure 2.



Parameters					
Required Spaces					
Space Type	Space Name	Space Number	Required Count	Target Area	Tolerance
*		SPACE S-*	61	12.00 m2	10%
*		SPACE N-*	18	18.00 m2	5%

Figure 2. Parameters for rule “Space number and area”.

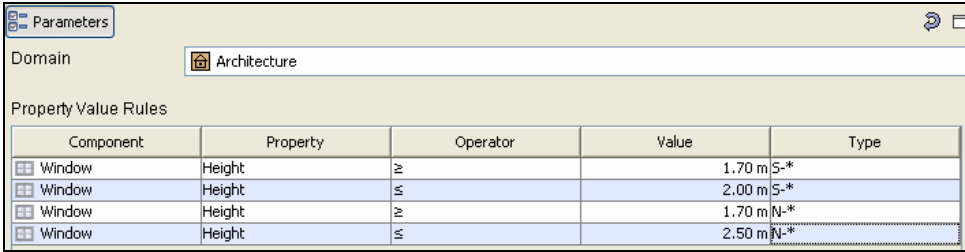
To ensure that each space is intended for the correct number of occupants, the second rule “Occupants number” evaluates the attribute “OccupancyNumber” in the property set “Pset_SpaceCommon” to check that only one occupant is intended in each of the small offices as required in the client requirements. The rule is illustrated in Figure 3.



Parameters					
Property Sets					
Component	Type	Property Set	Property	Value Exists	Value Conditions
Space	*,*,SPACE S-*	Pset_SpaceCommon	OccupancyNumber	Optional	Numeric = 1
Space	*,*,SPACE N-*	Pset_SpaceCommon	OccupancyNumber	Optional	Numeric = 1

Figure 3. Parameters for rule “Occupants number”.

Because the possibilities for indirect checking is not currently possible, it was decided to rename the window types in the north and south side of the façade with a prefix of “N-” or “S-” respectively. SMC will hence not check for compliance with individual space requirements but only check that the requirements are acceptable for all windows in either the north or south side of the façade. This is done in the third rule, “Window height”, illustrated in Figure 4.

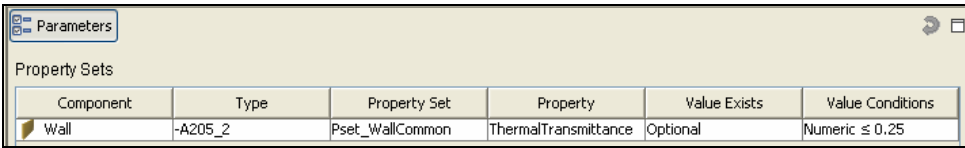


The Parameters window for the 'Window height' rule is shown. It has a 'Domain' of 'Architecture'. The 'Property Value Rules' table is as follows:

Component	Property	Operator	Value	Type
Window	Height	≥	1.70 m	S-*
Window	Height	≤	2.00 m	S-*
Window	Height	≥	1.70 m	N-*
Window	Height	≤	2.50 m	N-*

Figure 4. Parameters for rule "Window height".

The wall part of the façade on the northern and southern side of the building is created by its own type, "-A205_2" and the fourth rule "U-value of facade" will hence check this type of wall to ensure that the U-value is below the required value. This is done by evaluating the attribute "ThermalTransmittance" in the property set "Pset_WallCommon" as illustrated in Figure 5.

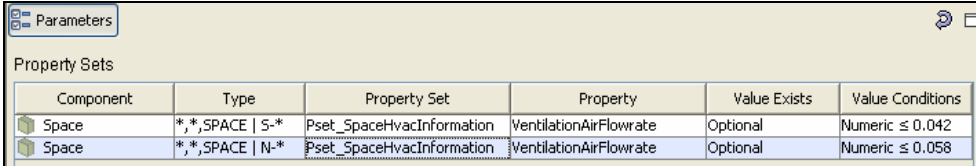


The Parameters window for the 'U-value of facade' rule is shown. The 'Property Sets' table is as follows:

Component	Type	Property Set	Property	Value Exists	Value Conditions
Wall	-A205_2	Pset_WallCommon	ThermalTransmittance	Optional	Numeric ≤ 0.25

Figure 5. Parameters for rule "U-value of facade".

Finally, the fifth rule, "Max air flow rate", checks that the air flow rate is below the maximum required. As this information is only provided by Riiska in the attribute "VentilationAirFlowRate" in the property set Pset_SpaceHvacInformation, the rule will check for the value here. The value is defined with the unit m³/s and values must hence be in accordance with this unit.



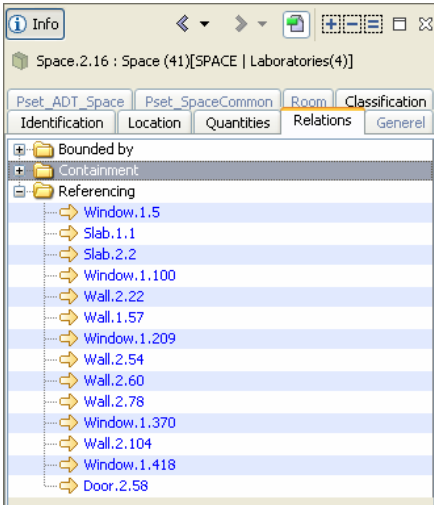
The Parameters window for the 'Max air flow rate' rule is shown. The 'Property Sets' table is as follows:

Component	Type	Property Set	Property	Value Exists	Value Conditions
Space	*,SPACE S-*	Pset_SpaceHvacInformation	VentilationAirFlowrate	Optional	Numeric ≤ 0.042
Space	*,SPACE N-*	Pset_SpaceHvacInformation	VentilationAirFlowrate	Optional	Numeric ≤ 0.058

Figure 6. Parameters for rule "Max air flow rate".

Several other parameters could be checked using SMC, however, in relation to the demonstration in Chapter 5, the listed rules are identified as some of the only ones appropriate to use.

As illustrated in Figure 7, SMC has exact information of all boundaries of e.g. a space, and to implement a possibility to allow for rules to represent indirect linking should hence be possible. This would allow for proper handling of constraints in a space of solutions.



The SMC Info window for 'Space.2.16 : Space (41)[SPACE | Laboratories(4)]' is shown. The 'Relations' tab is active, showing a list of boundaries under 'Referencing'.

Boundary
Window.1.5
Slab.1.1
Slab.2.2
Window.1.100
Wall.2.22
Wall.1.57
Window.1.209
Wall.2.54
Wall.2.60
Wall.2.78
Window.1.370
Wall.2.104
Window.1.418
Door.2.58

Figure 7. Information available in SMC for relations to space boundaries for each space boundary.

www.byg.dtu.dk

Department of Civil Engineering (BYG•DTU)
Technical University of Denmark
Brovej, Building 118
DK-2800 Kgs. Lyngby
Denmark

Tel: (+45) 45 25 17 00
Fax: (+45) 45 88 32 82
E-mail: byg@byg.dtu.dk