Towards a Model-Based System Specification and Verification in Integrated AEC-Projects

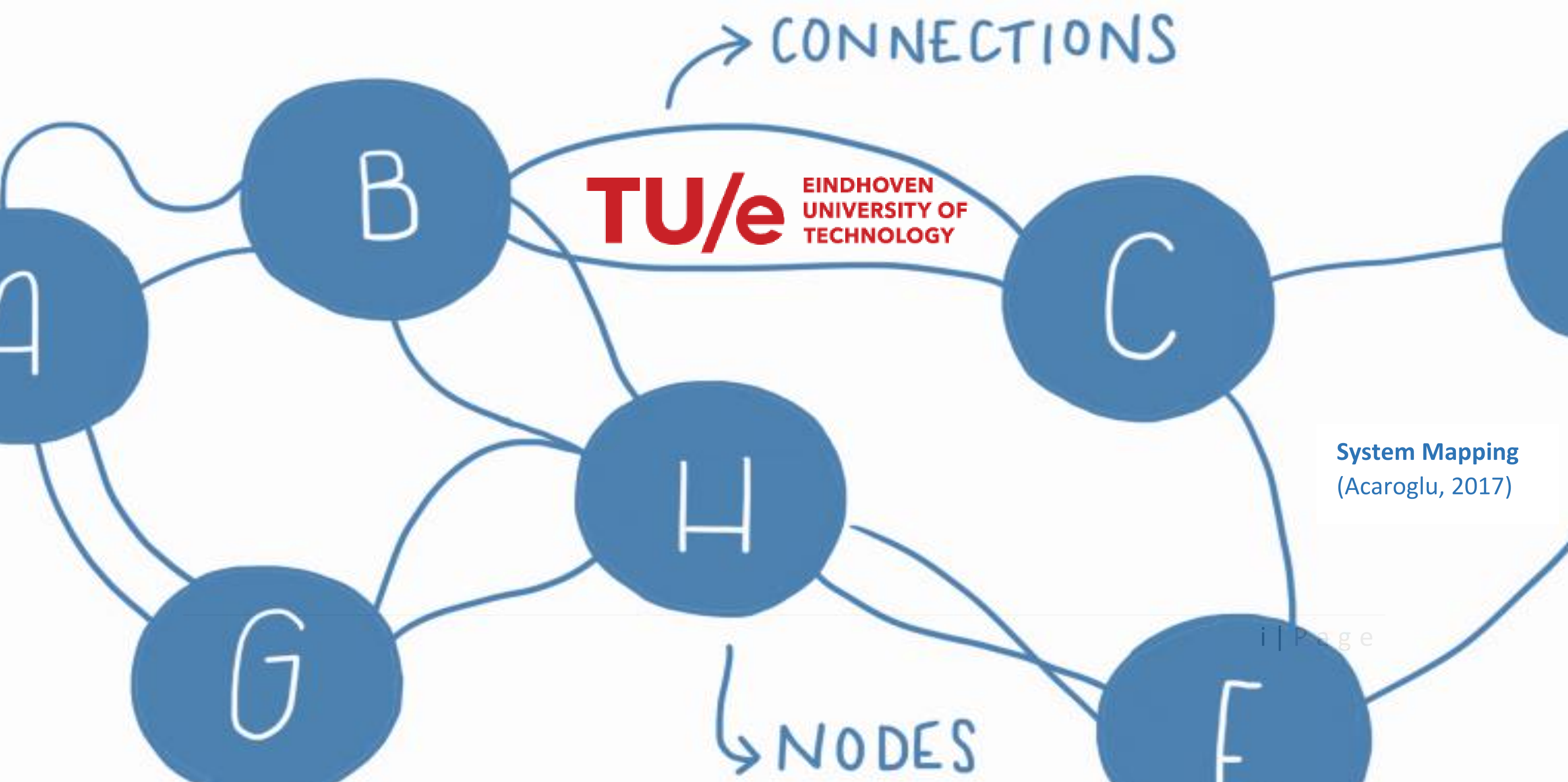
Integrating SysML and BIM as Initial Step

Master's Thesis

Michael J Granville

Date: 23 August 2019

Eindhoven University of Technology
Construction Management & Engineering (CME)
2018-2019



System Mapping
(Acaroglu, 2017)

Towards a Model-Based System Specification and Verification in Integrated AEC-Projects

Integrating SysML and BIM as Initial Step

Colophon

Title : **Towards a Model-Based System Specification and Verification in Integrated AEC-Projects**
Subtitle: Integrating SysML and BIM as Initial Step

Date : 23 August 2019
Date Presentation : 30 August 2019

Author : Michael Junior Granville
Student Number : 0978077
E-mail : granville.mj@gmail.com
University : Eindhoven University of Technology
Master Program : Construction Management & Engineering
Chair : Information Systems in the Built Environment

Graduation Committee

Chairman : **Prof.dr.ir. B. (Bauke) de Vries**

Full Professor, Department of the Built Environment, Information Systems in the Built Environment

1st Supervisor : **ir. B. (Bob) van Thiel**

University Lecturer, Department of the Built Environment, Information Systems in the Built Environment

2nd Supervisor : **dr.ir. I. (Ion) Barosan**

Assistant Professor, Department of Mathematics and Computer Science, Model Driven Software Engineering

2018-2019

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Preface

Herewith, I present to you my master's thesis comprising the integration between two powerful approaches, namely: SysML and BIM to improve the requirements management in integrated AEC-projects. The idea behind this ambitious plan emerged from my deeply and lovely interest in buildings and the technology that define them. Focusing on the requirements as basis for their development; I was convinced that if we desire buildings that perform perfectly, the focus must lie on their definition and performance. Initially, it was very difficult to define a research plan that overcomes my goal, because of the complexity of the two approaches. At first hand, SysML – originated from the system engineering – has less popularity and/or totally unknown in the AEC-industry. On the other hand, BIM its full implementation is still in progress. However, due to my determination, grit, and wiliness to learn; finally, I managed to set-up a plan where this master's thesis evinces these efforts, and so on to fulfill the graduation of the Construction Management and Engineering (CME) program at the Eindhoven University of Technology (TU/e).

For this incredible result, I would like to thank some persons for their excellent guidance and support during the process. First of all, I gracefully would like to thank professor de Vries for his advise going further with the idea despite the complexity of the topic. Furthermore, my gratefully thanks to my first supervisor Bob van Thiel, who helped me to set-up a good plan and supported me during its execution resulting in an excellent product. Those meetings were very helpful and instructive. Thereby, my respectfully and admirably thanks to my second supervisor Mr. Ion Barosan. Despite his busy agenda; he was always able to supervise and help me with the execution of my plan, and to become acquainted with SysML.

In addition, my lovely thanks to Masoud Delghandi – my best friend; who suggested me with some priceless advises, and helped me to arrange some interviewers for my practical framework. In this case, we arranged to interview Mr. Steven Knol and Mr. Anton Wubben from the multidisciplinary engineering firm: Royal HaskoningDHV. For this reason, my appreciatively thanks to their valuable input in my research. Additionally, my gratitude thanks to Stijn van Schaijk for providing me with a BIM-model to validate my model, and Mr. Gert Regterschot for its valuable guidance. Not to miss my lovely thanks to my colleagues and best friends: Mark van der Harm, Tarik Azaaj, Tanju Demir for the valuable and inspirational coffee-breaks.

Last but not least, I expressively would like to thank my lovely family as my brothers, cousin, father, and uncle for their support, patience, and confidence in me; especially my mother and aunt for reasoning me the importance of education in forming the human person.

Finally, I hope that every reader will enjoy reading and learning from this thesis report.

Michael J Granville
Amsterdam, August 2019

Table of contents

Summary

Nowadays, the Architecture, Engineering, and Construction (AEC) industry is facing an increasing complexity of building designs due to high-level client specifications and sustainability turning them into a complex task. Substantially, this complexity is a result of the quantity and interdependence of components that these designs embrace. For this reason, understanding them and the complexity that they aim, have been identified as being a crucial part of effective building design, whereby a greater knowledge of their nature can point out the areas where the need for improved management is greatest. In this case, it can be traced back to which extent project requirements are properly identified, understood, and implemented during the project development process. Unfortunately, several researches pointed out project failures because of managing project requirements not properly, which is justified by the lack of requirements identification, traceability, and inadequate requirement-management frameworks in the development process. As a consequence, misinterpretation of project complexity to several project failures such as exceedances of cost and schedule, and poor project performance.

In response to these implications, several researchers proposed various solutions, nevertheless, they really do not address the complexity of project requirements regarding structuring and interrelationships, and the building as a complex system – which is composed of different components. Therefore, this research proposes a more systems approach to overcome these limitations, by exploring the integration of System Modeling Language (SysML) and Building Information Modeling (BIM), to improve the systematic identification, capturing, and verification of functional requirements in integrated AEC-projects. This proposition follows previous works that transfer the system modeling approach to the building design domain to tackle project complexity.

For this purpose, the following research approach is defined consisting of five consistent phases: (a) phase 1 covers the second part of the literature research focusing on the subjects associated with the research objective, resulting in the creation of the theoretical framework for the model development in phase 3; (b) phase 2 – building the practical framework for the model development based on an interview approach – comprises the acquirement process of additional facts regarding the subjects of interest from a practical perspective; (c) phase 3 consists of the development process of the integration model including a model design and implementation through a case making the model feasible for a practical execution; (d) phase 4 follows with the validation process of the integration model based on a case study evaluating its practical implications; and (e) phase 5 completes the research approach with a reflection process through discussions, conclusions, and recommendations.

The literature research explores the subjects associated with the research objective such as *integrated AEC-projects*, *functional requirements*, *systems modeling (SysML)*, *OpenBIM-standard IFC*, and *SysML-BIM integration*. At first hand, integrated AEC-projects comprise an approach in which aspects such *product*, *process*, *human and organizations* are integrated to ensure a successful configuration. As a basis of this integration is the functional requirements outlining the ideas and necessities of the clients. Proceeding with system modeling – as part of the system engineering process – can be implemented through the Systems Modeling Language (SysML); allowing a graphical mapping of requirements, structure, and behavior to present a complete description of the system under development. Accordingly, SysML provides a comprehensive set of diagrams and constructs for modeling many common aspects of this system, and ensuring interoperability to several disciplines.

On the other hand, BIM can be seen as the process of creating and using digital models for specification, design, construction, and/or operation of construction projects. On the basis of this process lies the OpenBIM-standard IFC ensuring for: (a) a set of standard concepts such as geometry,

relations, processes, material performances, fabrication and other properties; (b) a consistent model-based communication among all stakeholders covering the entire building lifecycle; and (c) allowing data to be exchanged between products by different domains through standard data-model languages.

In addition, the theoretical framework is complemented with additional facts about the subjects of interest. Focused on the following subjects: integrated AEC-projects, functional requirements and OpenBIM-standard IFC; practical information was collected through an interview-session arranged at a multidisciplinary engineering company. From this practical framework, it can be seen that integrated AEC-projects face an involvement of several clients and other stakeholders with different specialized knowledge. Thereby, these kinds of projects include processes where clients' requirements are processed and transforming in high-valuable and future-proof buildings incorporated with sustainable principles. Nevertheless, these projects bring about some challenges regarding the way how to manage them, since they are still bounded by traditional design approaches and contracts making integration difficult in general. Furthermore, these challenges are also a result of the functional requirements – sometimes – not being adequately clear and measurable expressed due to the absence of a consistent methodology. In this case, the OpenBIM-standard IFC can be implemented through an integrated process to effectively contribute to their quantifiability and verifiability during the project development process.

Subsequently, based on the theoretical and practical framework, the SysML-BIM Integration model is created; therefore, contributing to the systematic identification, capturing, and verification of functional requirements in integrated AEC-projects. In this integrated approach, SysML establishes a visual modeling approach to support specification, design, verification, and validation of complex systems through its diagram taxonomy. Whereby, functional requirements can be model based on diagrams and models; consequently, overcoming the communication ambiguity among stakeholders, and the complexity of requirements regarding structuring, interrelationships, and traceability. At the other side, BIM – through its open-standard – provides fundamental concepts that contribute to an explicit specification of the functional requirements in the briefing phase, and their consistent validation in the design process.

However, the research approach has brought some limitations restricting the implementation of the integration model in some cases. For instance, the research has focused only on measurable functional requirements excluding other project requirements as non-functional requirements as well as architectural and esthetical requirements. Regarding the SysML Diagram Taxonomy, the research covered only the Use Case Diagram, Requirement Diagram and Block Definition Diagram excluding the other ones. Another limitation is the inclusion of ifcOWL in the integration model excluding the formal one (ifcEXPRESS) and the alternative one (ifcXML). Lastly, the limitation of the IFC-to-SysML Transformation Tool to automatically extract exclusively one door-object and two key properties.

Finally, this research recommends the introduction of SysML as a standard language in the building design domain to improve communication and requirements management. For this purpose, project participants have to be conversant with it through additional courses and training. The next recommendation comprises the duration-extension of the briefing phase by including sufficient resources, as well as a practical tool and consistent methodology to implement SysML. The last recommendation suggests the development of a software/tool or a plug-in to enable a practical transformation of IFC-to-SysML automatically. To conclude, future researches should focus on overcoming the limitations mentioned before.

Samenvatting

De hedendaagse AEC-industrie wordt geconfronteerd met een toename van complexiteit van gebouwontwerpen vanwege de hoog ambitieuze klanten- en duurzaamheidseisen met als gevolg een ingewikkeld proces. Wezenlijk is deze complexiteit het resultaat van het grote aantal en de onderlinge afhankelijkheid van componenten die deze ontwerpen omvatten. Om deze reden is het aankaarten van deze componenten en de complexiteit die ze beogen onderkend als een cruciaal onderdeel voor een effectief gebouwontwerp. Een betere opvatting van hun oorspronkelijkheid kan verwijzen naar de gebieden waar een betere beheersing noodzakelijk is. In dit geval is het afhankelijk van in hoeverre projecteisen naar behoren zijn geïdentificeerd, geïnterpreteerd en geïmplementeerd tijdens het projectontwikkelingsproces. Helaas wijzen verschillende onderzoekers erop dat de tekortkomingen in huidige projecten zijn te wijten aan de beperkte beheersing van projecteisen. Dit wordt verklaard door de onvoldoende identificatie en traceerbaarheid van klanteneisen, en de daarbij inadequate kaders voor eisenbeheersing in het ontwikkelingsproces. Dat resulteert in een verkeerde interpretatie van projectcomplexiteit en alle mogelijke projectimperfecties als kosten- en tijdsoverschrijdingen en afwijkende projectprestaties.

Om die reden suggereerden diverse onderzoekers allerlei oplossingen. Desondanks adresseren ze de complexiteit van projecteisen met betrekking tot hun organisatie en onderlinge relaties niet, en evenmin zien ze het gebouw niet als een complex systeem dat uit allerlei componenten is opgebouwd. Daarom streeft dit onderzoek naar een meer systematische benadering om deze limitaties te hanteren. Met als doel de integratie tussen System Modeling Language (SysML) en Building Information Modeling (BIM) te onderzoeken, en de systematische identificatie, vastlegging, en verificatie van functionele eisen in geïntegreerde AEC-projecten te verbeteren. Dit voorstel vloeit voort uit voorgaande onderzoeken die de systeemmodellering toepassen op de gebouwensector met als doel de projectcomplexiteit te tackelen.

De onderzoeksbenadering die is opgesteld, bestaat uit de volgende vijf consistente fasen: (a) fase 1 beslaat het tweede deel van het literatuuronderzoek gericht op de onderwerpen geassocieerd met de onderzoeksdoelstelling, en het opstellen van het theoretisch kader voor de modelontwikkeling in fase 3; (b) fase 2 – invullen van het praktisch kader voor de modelontwikkeling op basis van een interviewmethode – bestaand uit een data-collectieplan van additionele aspecten vanuit een praktisch perspectief; (c) fase 3 omvat het ontwikkelingsproces van het integratiemodel bestaande uit een modelontwerp, en zijn implementatie op basis van een praktijkvoorbeeld zodat het model praktisch en operationeel inzetbaar is; (d) fase 4 bestaat uit het validatieproces van het integratiemodel aan de hand van een praktijksituatie om de praktische implicaties te evalueren; en (e) fase 5 rondt de onderzoeksbenadering af met een reflectie ondersteund door discussies, uitmondend in conclusies en aanbevelingen.

Het literatuuronderzoek onderzoekt de onderwerpen gerelateerd aan de onderzoeksdoelstelling, namelijk: *geïntegreerde AEC-projecten*, *functionele eisen*, *systeemmodellering (SysML)*, *OpenBIM-standaard IFC* en *SysML-BIM integratie*. Uitgaand van dit onderzoek omvatten geïntegreerde AEC-projecten een benadering waarin aspecten als *product, proces, humaan* en *organisaties* geïntegreerd zijn om een succesvolle configuratie te waarborgen. Aan de basis van deze integratie staan de functionele eisen die de ideeën en behoeften van de klanten weergeven. De systeemmodellering – als onderdeel van het system-engineering proces – kan worden geïmplementeerd aan de hand van SysML zodat een grafische samenstelling van eisen, structuur, en gedrag in een complete beschrijving van het ontwikkelde systeem gepresenteerd wordt.

Hiervoor levert SysML een uitgebreide set van diagrammen en constructen om veel van die algemene aspecten van het systeem te modelleren, en bovendien interoperabiliteit met verscheidene disciplines.

Aan de andere kant kan BIM worden gezien als een proces waarin digitale modellen worden gerealiseerd en gebruikt om bouwprojecten te specificeren, ontwerpen, bouwen en/of beheren. Ten grondslag aan dit proces ligt de OpenBIM-standaard IFC met het faciliteren van: (a) een set van standaard concepten als geometrie, relaties, processen, materiaalprestaties, fabricatiekenmerken en andere kenmerken; (b) een consistent modelgebaseerde communicatie onder alle stakeholders waarbij de gehele levenscyclus van het bouwproject wordt omarmd; en (c) data-uitwisseling tussen producten in verschillende domeinen gebaseerd op standaard data-modelleertalen.

Bovendien is het theoretisch kader uitgebreid met additionele gegevens over de elementaire onderzoeksonderwerpen met de focus gericht op onderwerpen als: geïntegreerde AEC-projecten, functionele eisen en OpenBIM-standaard IFC; praktische informatie werd verzameld in een interviewsessie gehouden bij een multidisciplinair ingenieursbureau. Uitgaand van dit praktisch kader is duidelijk dat geïntegreerde AEC-projecten te maken hebben met een betrokkenheid van diverse klanten en andere stakeholders elk met hun eigen specialisatie. Daarenboven bevatten deze projecten processen waarin klanteneisen worden verwerkt en getransformeerd in hoog kwalitatieve en toekomstgerichte gebouwen die voldoen aan allerhande duurzaamheidsprincipes.

Daarnaast brengen deze projecten enige uitdagingen met zich mee omtrent de manier waarop ze dienen te worden gemanaged, doordat ze nog steeds met traditionele ontwerpmethodes en contracten uitgevoerd worden en hierdoor integratie in het algemeen wordt bemoeilijkt. Deze uitdagingen zijn het resultaat van de functionele eisen die – in sommige gevallen – niet genoeg duidelijk en/of meetbaar uitgedrukt worden door het ontbreken van een consistente methodologie. In dit kader kan de OpenBIM-standaard IFC gesteund door een geïntegreerd proces worden ingezet, om effectief bij te dragen aan hun meetbaarheid en verifieerbaarheid tijdens het projectontwikkelingsproces.

Daaropvolgend volgt de ontwikkeling van het SysML-BIM Integratiemodel dat is ontwikkeld op basis van de ervaring met het theoretisch en praktisch kader. Samenvattend heeft het model als doel het verbeteren van de systematische identificatie, vastlegging, en verificatie van functionele eisen in geïntegreerde AEC-projecten. In dit verband vestigt SysML een visuele modelleerbenadering om specificatie, ontwerp, verificatie en validatie van complexe systemen te ondersteunen aan de hand van zijn diagram taxonomie. Op basis van diagrammen en modellen kunnen functionele eisen worden gemodelleerd die de dubbelzinnigie communicatie tussen stakeholders kan vermijden, evenals de complexiteit van eisen omtrent hun organisatie, onderlinge afhankelijkheid en traceerbaarheid inzichtelijker te maken. Aan de andere kant verschaft BIM – via zijn open-standaard – fundamentele concepten die bijdragen aan een expliciete specificatie van functionele eisen in de briefing fase en aan hun validatie in de ontwerpfase.

Hoewel er enkele limitaties zijn aan de onderzoeksbenadering heeft de implementatie van het integratiemodel in mijn onderzoek nog niet het volledige potentieel zichtbaar gemaakt. Zo richtte het onderzoek zich alleen op meetbare functionele eisen met uitsluiting van niet-functionele eisen evenals architectonische en esthetische eisen. Het onderzoek op de SysML Diagram Taxonomy is met name gericht op de diagrammen: Use Case Diagram, Requirement Diagram, en Block Definition Diagram. Een andere afbakening is het opnemen van de ifcOWL-taal in het integratiemodel met uitzondering van de formele taal (ifcEXPRESS) en de alternatieve (ifcXML). De laatste limitatie beslaat de beperking van de IFC-to-SysML Transformation Tool.

Tenslotte beveelt dit onderzoek de introductie van SysML als standaardtaal aan in de gebouwensector met als doel de communicatie en eisenbeheersing te optimaliseren. Hiervoor dienen betrokken partijen bekend te worden met de methodologie aan de hand van cursussen en trainingen. Een verdere aanbeveling verlengt de duur van de briefing fase met daarin voldoende beschikbare middelen, alsmede een praktische methodologie om SysML te implementeren. De laatste aanbeveling behelst de ontwikkeling van een software/tool of plug-in om een praktische en automatische transformatie van IFC-to-SysML te bevorderen. Toekomstig onderzoek dient zich te richten op het nader bestuderen van de eerder genoemde limitaties.

Abstract

This research project overcomes the increasing complexity of building designs that the AEC-industry are facing at this moment. Substantially, this complexity is a result of the quantity and interdependence of components that these designs embrace, which demands an approach in which these components and the complexity that they aim can be interpreted pointing out the areas of consideration. Basically, this approach can be determined depending to which extent project requirements are properly identified, understood, and implemented during the project development process. Lamentedly, the preliminary literature study shows a lack of requirements identification, traceability, and inadequate requirements-management frameworks in this process. As a consequence, misinterpretation of project complexity to several project failures such as exceedances of cost and schedule, and poor project performance. Therefore, this research proposes a more systems approach to deal with these limitations, by exploring the integration of System Modeling Language (SysML) and Building Information Modeling (BIM), to improve the systematic identification, capturing, and verification of functional requirements in integrated AEC-projects.

For this purpose, a research approach was defined consisting of the following five consistent phases: (a) phase 1: Theoretical Framework; (b) phase 2: Practical Framework; (c) phase 3: Model Development; (d) phase 4: Model Validation; and (e) phase 5: Conclusion. Accordingly, the SysML-to-BIM Integration Model was developed comprising an integrated process where SysML and BIM are linked. At first hand, SysML establishes a visual modeling approach to support specification, design, verification, and validation of complex systems through its diagram taxonomy. Whereby, functional requirements can be model based on diagrams and models; therefore, overwhelming the communication ambiguity among stakeholders, and the complexity of requirements regarding their structuring, interrelationships, and traceability. On the other hand, BIM-through its open-standard – provides fundamental concepts that contribute to an explicit specification of functional requirements in the briefing phase, and a consistent validation of them in the design process. Finally, this research contributes to the improvement of collaboration among project stakeholders for developing high-performance buildings.

List of Abbreviations/Glossary

AEC	: Architecture Engineering Construction
MBSE	: Model Based Systems Engineering
INCOSE	: International Council on Systems Engineering
OMG	: Object Management Group
SysML	: System Modeling Language
UML	: Unified Modeling Language
PSM	: Parametric Systems Modeling
RDF	: Resource Description Framework
RDFS	: Resource Description Framework Schema
OWL	: Web Ontology Language
BDD	: Block Definition Diagram
DSL	: Domain Specific Language
DSM	: Domain Specific Modeling
API	: Application Programming Interface
QVT	: Query/View/Transformation language
BIM	: Building Information Modeling
DBB	: Design Bid Build
IPD	: Integrated Project Delivery
CDE	: Common Data Environment
ICT	: Information and Communications Technology
IT	: Information Technology
CAD	: Computer-Aided Design
IFC	: Industry Foundation Classes
GUID	: Globally Unique Identifier
STEP	: Standard for the Technical Exchange of Product Model Data
ISO	: International Standards Organization
XML	: Extensible Mark-Up Language
HTML	: HyperText Mark-Up Language
WWW	: World Wide Web
HVAC	: Heating, Ventilation and Air Cooling
URI	: Universal Resource Identifiers
URN	: Uniform Resource Name
SDE	: Systems Development Environment
MOF	: Meta Object Facility

List of tables

List of figures

1. Introduction

Nowadays, the Architecture, Engineering, and Construction (AEC) industry is facing an increasing complexity of building designs due to high-level client specifications and sustainability turning them into a complex task (Geyer, 2012). This phenomenon drastically increases when realizing complex and ambitious construction projects in fragmented processes that consist of quantity and interdependence of components (Froese, 2010). These complexions reflect the characteristics of project complexity described by Davies & Mackenzie (2014), who indicated that complexity can be determined as a system in terms of the number and variety of components, and interdependencies among them. Focusing on the AEC-industry, its process comprises: (a) multiple phases of the construction project lifecycle; (b) the involvement of multidisciplinary teams, including owners, consultants, engineers, contractors, subcontractors and suppliers; and (c) the use of heterogeneous software and hardware system/tools. As a result, substantiating the well-known fragmentation and multidisciplinary interdependencies within the industry explicitly. In addition, the process includes the increasing technical complexity of the system – in this case the building – involving quantity subsystems and relations defining the system structure and influencing its behavior (Baundains, et al., 2014).

Simultaneously, understanding these aspects and the complexity that they aim have been identified as being a crucial component of effective building design. Whereby a greater knowledge of their nature can point to the areas where the need for improved management is greatest (Geyer, 2012; Baundains, et al., 2014). Basically, the nature of this complexity and its evolution can be traced back to which extent project requirements are properly identified, understood and implemented during the project development process (Jallow, Demian, Baldwin, & Chimay, 2014). Obviously, defining a clear requirement means that progress can be measured and areas needing attention can be identified according to Yu & Chan (2010). As a result, project complexity can be understood and managed. Unfortunately, several literatures pointed out project failures because of managing project requirements not properly, which is justified by the lack of requirements identification, traceability, and inadequate requirement-management frameworks during the development process according to Yu & Chan (2010), Jansson, Schade, & Olofsson (2013), and Pegoraro & Carisio de Paula (2017). In response to these implications, this research aims an improved management of project requirements to ensure a manageable project complexity; as a consequence; the project is on track when it comes to meet the expectations.

1.1. Problem definition

Taking into consideration the complexity of construction projects and project requirements that contribute to its evolution; a clear understanding of their nature is a prerequisite to project success. As mentioned before, the evolution and understandability of project complexity can be traced back to how project requirements are properly identified, understood and implemented during the project development process. Lamentably, the preliminary literature study indicated a lack of identification, traceability, management of clients' requirements, and inadequate requirement-management frameworks in the process. As a result, misunderstanding of project complexity to several failures like exceedances of cost and schedule, and poor project performance according to Yu A. T., Shen, Kelly, & Hunter (2007), Yu & Chan (2010), and Jallow A. K., Demian, Baldwin, & Chimay (2014). Despite the several solutions proposed by Kamara & Anumba (2000), Shen, LI, Chung, & Hui (2004), Kiviniemi, Fischer, Bazjanac, & Paulson (2004), Jansson, Schade, & Olofsson (2013), and Stancheva (2017) to overcome those issues. They really do not address the complexity of project requirements regarding structuring and interrelationships, and the building as a complex system that is composed of different interdependent components (Geyer, 2012; Moonen, 2016). Furthermore, some of those solutions focus only on one specific case making design integration difficult (Geyer, 2012; Stancheva, 2017).

On the other hand, the rising utilization of integrated contracts like ''Design & Build'' in building designs. Those solutions do not cover the possible implications concerned, which require an explicit release of complete requirement specifications, and the involvement of a multidisciplinary team early in the process (Spek, 2012; Pels, Beek, & Otter, 2013; Moonen, 2016).

Therefore, a more systems approach is required to deal with those aspects. In this case, Systems Engineering (SE) and its modeling approach can play an important key role. In accordance, several researches have indicated the application of SE in construction projects to tackle those issues, although it has received a little acceptance to date according to Pels, Beek, & Otter (2013), Baundains, et al. (2014) and Stancheva (2017). Accordingly, this is due to the lack of client's policies to enforce it, and the fact that building projects are still primarily bounded by traditional contracts. For instance, Yahiaoui, Sahraoui, Hensen, & Brouwer (2006) indicated that using SE is the right way in analysing and resolving a problem in developing a system, which covers a broad set of processes and methods for modelling and analyzing interactions among requirements, subsystems, components and constraints (Geyer, 2012; Baundains, et al., 2014).

In the field of requirements, SE provides models and simulations to enable more depth and analysis of systems requirements early in the design (Pels, Beek, & Otter, 2013). Besides, it brings a uniform process structure with a strong emphasis on completeness and traceability of requirements (Locatelli, Mancini, & Romano, 2014). Definitely, it is a good first step towards BIM implementation according to Pels, Beek, & Otter (2013). For these reasons, applying systems engineering and its modeling approach in building design promises a way to improve building performance substantially (Geyer, 2012; Locatelli, Mancini, & Romano, 2014).

Supplementally, to provide an approach that complements the problem statement defined above, this research suggests the integration of system modeling and BIM in order to improve the systematic identification, capturing and verification of project requirements during the development process. In this approach, the study will be limited to functional requirements/specifications that are set-up in integrated construction projects. This proposition follows Geyer's (2012) work that transfers the system modeling approach to the building design domain. System modeling, which is an important focus of SE, is a modeling approach based on the standard modeling language SysML to tackle project complexity (Locatelli, Mancini, & Romano, 2014). At the other side, BIM as an integrated approach to cover the entire development process of the building using digital models.

In a combined approach, SysML can be implemented within a model-based systems engineering (MBSE) framework, to address the interactions among requirements, subsystems, components, constraints, and subsequently capture them in a system model during the briefing process. This framework is based on three pillars including SysML as language, a methodology, and tool. After the system model is created, this model can proceed in the design process as a reference point for discussing the design development and validation. Integrally, the BIM-IFC standard, which is an open and standardized data model, will play a key role by enriching the SysML model with valuable building information, whereafter requirements can consistently be checked and verified. This standard fundamentally provides a background structure and relations between components that partially meet system modeling requirements (Geyer, 2012). For this purpose, the following research questions are defined.

1.2. Research questions

1.2.1. Main-research question

"How can an integration between SysML and BIM be created to improve the systematic identification, capturing, and verification of functional requirements in integrated AEC-projects?"

1.2.2. Sub-research questions

1. **What are the characteristics of integrated project environments and functional requirements?**
2. **How are functional requirements identified, captured, and verified in integrated project environments?**
3. **What are the characteristics of SysML and the OpenBIM-standard IFC?**
4. **How can SysML and the OpenBIM-standard IFC contribute to a better consistency of functional requirements during the briefing and design process?**
5. **How can SysML and the OpenBIM standard IFC be linked to create interoperability for exchanging information?**

1.3. Research design

This section describes the research design and research model (see Figure 1.1) defined to cover the research objectives and questions, which are arranged based on the following five phases. Phase 1 started with the second part of the literature leading to the development of the theoretical framework. The focus lied on the subjects associated with the research questions, namely: Integrated AEC-projects, Functional Requirements, Systems Modeling (SysML), OpenBIM-IFC, and SysML-BIM integration. Phase 2 proceeded with the acquirement of additional information regarding the subjects of interest, although from a practical perspective. In this case, the subjects of interest were Integrated AEC-projects, Functional Requirements, and OpenBIM-standard IFC. For this purpose, a procedure of defining interview guidelines, data collection, data analysis and interpretation was defined; consequently, this phase was executed through two interview-sessions with two experts from the field forming therefore the practical framework. These sessions are convened at a multidisciplinary engineering company. After the theoretical framework along with the practical framework were formed, Phase 3 could initiate with the development of the SysML-BIM Integration model comprising the model design, and model implementation through a use case for a practical application. Subsequently, Phase 4 proceeded with the model validation based on a case study evaluating its practical implications. The case study comprised a residential building project. Finally, Phase 5 completed the approach with the discussions, conclusions and recommendations describing the limitations and benefits of the model developed.

1.4. Research relevance

The research contributes to the improvement of the collaboration among project stakeholders to create high-performance buildings as a result meeting the users' expectations. Unfortunately, there are several examples in which a poor collaboration has led to poor buildings whereby the users, who finally bear the bunt of this lack. Back in the days, a building could be specified, designed, and built by one person, the so-called "Master Builder". Nowadays, the building is became a complex system demanding an integrated approach defining product, process, human and organizations to overcome this system complexity. In addition, this research follows previous works that use the ifcOWL as standard to expressed the building for valuable interoperable advantages to overwhelmed the limitations of ifcEXPRESS en ifcXML regarding difficulties in adaptation and semantic interoperability. In this case, this research shows the potential of ifcOWL to automatically extract partial graphs and adapt it into a new graph to cover possible use cases; subsequently, transform it into another standard language, namely: the MOF-XMI standard.

1.5. Reading guide

The thesis is organized as follow: Part 1 Theoretical Framework comprises the literature study conducted, which covers the following subjects of interest: Chapter 2 Integrated AEC-projects, Chapter 3 Model-Based Systems Engineering, and Chapter 4 Building Information Modeling (BIM); Part 2 Practical Framework (Chapter 5) consists of the practical view about the key subjects;

Part 3 Model Development (Chapter 6) includes the model design, implementation, and prototype implementation; Part 4 Model Validation (Chapter 7) illustrates the evaluation of the model through a case study; Part 5 Conclusion (Chapter 8) completes the thesis with the conclusions, recommendations and future research.

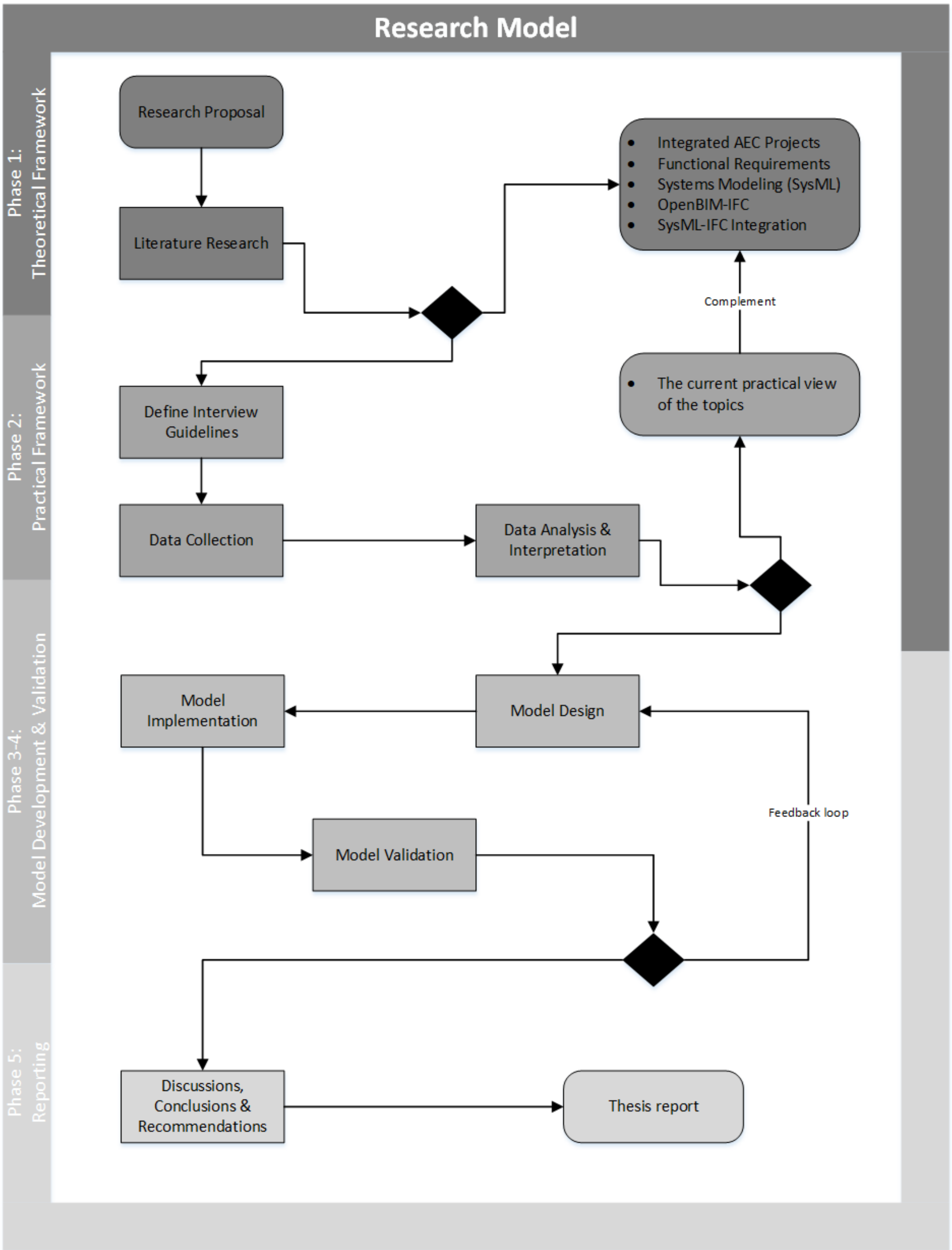


Figure 1.1. The research model

1 Theoretical Framework

2. Integrated AEC-projects

Integrated AEC-projects concern the approach in which aspects such as product, process, human and organizations apparently in some cases unrelated, are integrated in a way to allow synergistic benefits to be realized successfully. In this emergent approach: The product aspect, which refers to the artifact "building" as a physical asset created in these kinds of projects, aims to provide valuable services as delivering, healthcare, education, retail, and homes to the society (Winch, 2010). These services are inherently the results of the building functions and corresponding performance indicators defined, applicated, and performed through the specification (briefing), production (design and construct), and utilization (usage) of these buildings respectively. At different hierarchical level of the decomposition, building functions are being associated and influenced by building subsystems, and their functional interactions (Baundains, et al., 2014).

On the other hand, the process aspect deals with several phases in which the development of the building goes through to meet expectations (Yahiaoui, Sahraoui, Hensen, & Brouwer, 2006; Pels, Beek, & Otter, 2013). The process, which follows the same approach as the building lifecycle described before: Initiates firstly with the briefing phase, wherein the client's needs are refined, and defined in comprehensive requirements, and captured in a building specification format as a baseline for the design. In accordance, the design phase proceeds with the transformation of these requirements into design solutions, whereby building components are designed and integrated, and frequently verified through a sequence of sub-phases defined for this purpose. Thereafter, the construction phase follows with the realization of the building components and their integration based on the design specifications defined in the previous phase. As in the previous phases, verification and validation are usually performed to be consisted with the specifications.

As last phase, the operation and maintenance phase covers the operational state of the building that aims to deliver a required service, and the maintenance state that sustains the delivering of this service efficiently (Yahiaoui, Sahraoui, Hensen, & Brouwer, 2006; Pels, Beek, & Otter, 2013). Finally, the human and organization aspect comprises the involving of several parties with different roles, responsibilities, goals and objectives. As a consequence, complicating – in some cases – the decision-making within collaboration/projects according to Ruijven van (2018). Despite this fact, in integrated environments, they collaborate together sharing a definite goal and mission in a particular process to achieve the larger objectives of the projects as a whole (Hoeve ten, 2018; Ruijven van, 2018).

Additionally, in a recent research about project delivering complex systems; Ruijven van (2018) also acknowledged the importance of those aspects since issues arise through the mixing of product and process, and both processes and products are defined and realized by enterprises (human and organizations). In his dissertation, he indicated the substantial distinguishing and definition of these aspects within a complex system of systems, ensuring interoperability and coherence between them, based on the required interactions within and between them. In the context of integrated AEC-projects, these required interactions between those aspects are basically defined by the inputs at the start of the project, which in this case are the functional requirements/specifications outlining the ideas and necessities of the stakeholders.

2.1. Functional requirements

Functional requirements or specifications are often associated with integrated projects according to Spek (2012). However, alike to common technical specifications, functional specifications are pertaining to all kinds of projects despite the project or collaboration approach used (SBR, 2006; Spek, 2012). On the other hand, the association can be explained by the fact that functional specifications comprise a lifecycle-oriented and an integrated approach, in which client, users, and other interested parties are centrally involved, whereby their needs are converted in performances reflecting the desired system or service without a given solution (SBR, 2006). Accordingly, it can be related to the term "Performance Approach" according to SBR (2006), which is the practice of thinking and working in terms of ends rather than means. It concerned with what a building or a building product is required to do, and not by prescribing how it is to be constructed. An example of functional specifying is illustrated in Table 2.1.

Requirement
The accommodation must provide the opportunity to have meetings with a group of 25 persons in difference setups (theater and round table meeting setup)

Performances	
Space requirement	3 m^2 per person
Space ratio	Length : Width 1,5 : 1
Ventilation	Min. 30 m^3 fresh air per person/hour
Air temperature	19°C < T < 21°C
Background noise (as a consequence of external sources)	Max. 35 dB(A)
Reverberation time	0,8 – 1,0 sec.
Lighting level on table top	Max. 500 lux.

Table 2.1. An example of a functional requirement with its corresponding specifications (SBR, 2006)

3. Model-Based Systems Engineering

Model-Based Systems Engineering (MBSE) is a formalized application of modeling to support the system engineering process (Friendenthal, Moore, & Steiner, 2015; Cardoso, 2007). Its utilization in this process is to support the activities like systems requirements, design, analysis, verification, and validation, which take place beginning in the conceptual design phase, continuing throughout development, and later lifecycle phases (Friendenthal, Moore, & Steiner, 2015). The MBSE-approach emphasizes the use of models to perform the systems engineering activities that have traditionally been performed using a document-based approach (Czarnecki & Helsen, 2006). Despite the many advantages of this traditional approach, it has some fundamental limitations according to Friendenthal, Moore, & Steiner (2015), and Valdes (2016). Unfortunately, the completeness, consistency, relationships and relationships between requirements, design, engineering analysis and test information, are difficult to access because the information is spread across several documents. Contrary to this approach, MBSE integrates this information to address multiple aspects of the system under development in a cohesive manner, rather than dealing with a disparate collection of individual models. This integration enables the understanding of the system from multiple perspectives and to ensure interaction across the different perspectives. For this purpose, MBSE uses its primary artifact, which is the ''system model''. This model formally represents all aspects of a systems engineering problem, which enhances specification and design quality; reuse of system specifications and design artifacts; and communication among the development teams (Johnson, Paredis J.J., & Burkhart, 2008). In addition, the model can be integrated with other analysis and design models to represent other aspects of the system (Friendenthal, Moore, & Steiner, 2015). The following paragraphs will cover the key aspects of the system model and its implementation in the process for an effective MBSE application.

3.1. The system model as a primary artifact of MBSE

As the primary artifact of the MBSE-approach, the system model enables the design of a system that specifies its requirements and meets its overall objectives (Friendenthal, Moore, & Steiner, 2015). It provides a mechanism to specify and integrate subsystems and component designs into a system model, and maintain traceability between the system and components requirements. Considering the increasing complexity of contemporary systems engineering projects, as they are handled by geographically distributed design teams, contained by the objective of multiple stakeholders, and inundated by large quantities of information. The system model can be used to represent an information hub addressing the interdisciplinary dependencies; consequently, supporting a holistic modeling approach, and exchanging of project information between different views (Geyer, 2012; Valdes, 2016). The definition and evolution of this model is based on three main pillars: (a) the language: for representing the system being developed; (b) the method: that defines the activities and artifacts, and (c) the tool: to implement the modeling language and method (see Figure 3.1) (Barosan, 2017).

The language, which is a key part of the model development, is based on the System Modeling Language (SysML) standard. SysML was adopted by the Object Management Group (OMG) in 2006 and is supported by leading organizations from the system engineering industry – including International Council on Systems Engineering (INCOSE) (Weilkiens, 2008). SysML extends UML, which was originally specified as a modeling language to support general-purpose software modeling. It was developed in response to the shortcomings of UML with regard to systems engineering. Accordingly, several researchers suggested the application of SysML to model complex systems for an effective MBSE implementation. For instance, Geyer (2012) transfers SysML to the building design domain and develops a method called Parametric Systems Modeling (PSM) for multidisciplinary design optimisation and design collaboration. Locatelli, Mancini, & Romano (2014) indicated that the implementation of SysML allows an open-approach facilitating the communication among involved organizations, and therefore improving the project governance. In addition, these researchers concluded that SysML includes diagrams and models that reduce likelihood of miscommunication, and provides a standard and comprehensive paradigm for system specification (Geyer, 2012; Locatelli, Mancini, & Romano, 2014). Lastly, Shah, Kerzhner, Schaefer, & Paredis (2010) described SysML as a well-suited language for defining high-level relationships that exist between requirements, structure, and behaviour.

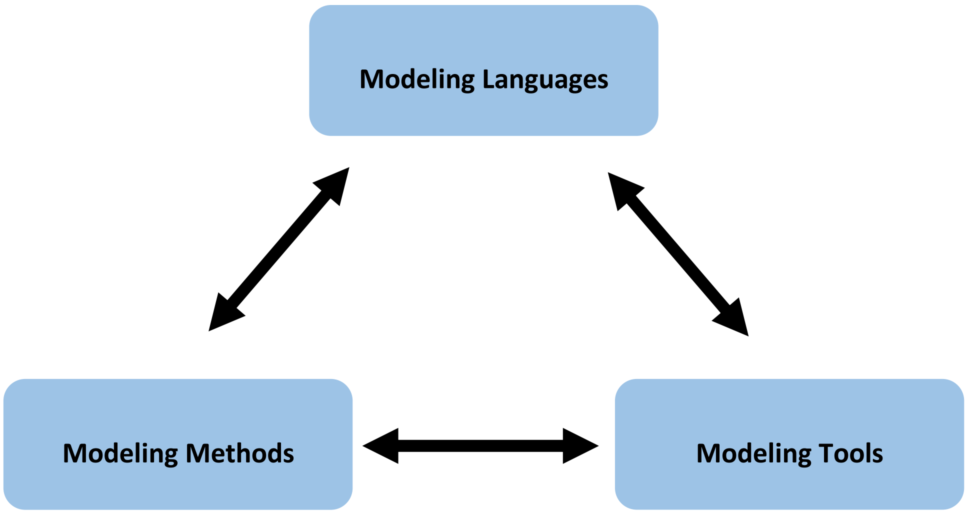


Figure 3.1. The Three Pillars of the MBSE-approach (Barosan, 2017)

Contrary to the SysML-approach, Nassar & Austin (2013) recommended a MBSE-approach including RDF graphs as language instead of SysML. In this work, RDF graphs are used to model: individual design requirements; graphs of requirements; characteristics of individual components; and graphs of design components. Basically, these graphs together with the Web Ontology Language (OWL) form a formal logical language to represent knowledge. Unfortunately, they have a limited adoption in systems engineering, because of the absence of a consistent graphical notation like the SysML language provides. Accordingly, Jenkins & Rouquette (2012) indicated the combination of both in each way to benefit from the attractive graphical notation of SysML, and the formal reasoning of RDF/OWL.

3.2. Introduction to Systems Modeling Language (SysML)

System Modeling Language (SysML) is a visual modeling language that provides a comprehensive set of diagrams and constructs for modeling many common aspects of systems engineering problems. Basically, this modeling-approach allows a graphical mapping of requirements, structure, and behaviour, to provide a complete description of the system under development – including its components and environment (Geyer, 2012). While the diagrams are showing the several views of the system to understand its aspects both individually and together, the description consists in an integrated system model to support analysis, specification, design, verification and validation of complex systems (Jenkins & Rouquette, 2012; Friendenthal, Moore, & Steiner, 2015). SysML provides nine diagrams as shown in Figure 3.2, and can represent the following aspects of a system, components, and other entities: (a) structural composition, interconnection, and classification; (b) flow-based, message-based, and state-based behaviour; (c) constraints on the physical and performance properties; (d) allocations between behaviour, structure and constraints; and (e) requirements and their relationships to other requirements, design elements and test cases (Friendenthal, Moore, & Steiner, 2015).

The following sections cover the most relevant diagrams according to the research objective, namely: the SysML diagram and notation in general, Use Case Diagram, Requirement Diagram, and Block Definition Diagram – also well-known as the BDD.

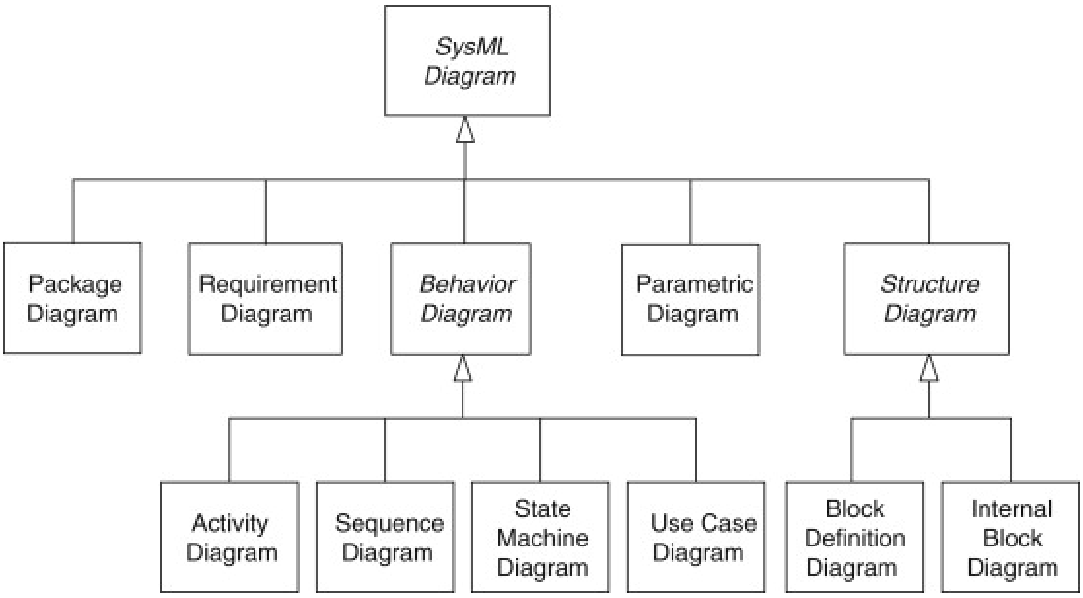


Figure 3.2. The SysML Diagram Taxonomy (Friendenthal, Moore, & Steiner, 2015)

3.2.1. SysML diagram and notation

The SysML diagrams and notations provide a mechanism to present different views of the model for a specific purpose. While these diagrams and notations are based on the UML diagrams and notations, they do not include the fundamental aspects of UML because of its lacking in satisfying the requirements for modeling systems. For this purpose, SysML included modifications to other UML diagrams such as class diagram, composite diagram, and activity diagram, and additionally added two new diagrams for requirements and parametrics (Friendenthal, Moore, & Steiner, 2015). This finally led to the composition of the SysML diagram taxonomy as shown in Figure 3.2.

Figure 3.3. The nodes (a); and the paths connecting the nodes (b) (Friendenthal, Moore, & Steiner, 2015)

In addition to this diagram taxonomy, SysML also supports tabular, matrix, and tree views for the model. Regarding the understandability and implementation of the diagrams and notations for modeling systems, each diagram shown in the taxonomy must have a “diagram frame” that encloses the diagram content according to Friendenthal, Moore, & Steiner (2015). The diagram frame corresponds to the model element that provides the context for the diagram content. The frame structure is rectangle with a diagram header containing information in the top left corner of the frame. The content area includes the diagram elements that present the model of interest (see Appendix I). Concerning the SysML diagrams, they can be composed of nodes and paths as depicted in Figure 3.3(a) and 3.3(b). The nodes are diagram elements that generally appear as shapes, such as rectangles, ovals or other polygons with text labels. Additionally, the nodes may contain text strings and/or graphical symbols that may correspond to other model elements (see Figure 3.3(a)). Regarding the paths – also known as edges – are diagram elements that generally appear as lines that have additional adornments such as arrowheads and text strings. The lines are implemented in different style and have several ends depending on the modeling concept they represent (see Figure 3.3(b)) (Friendenthal, Moore, & Steiner, 2015)

3.2.2. The Use Case Diagram

The Use Case Diagram describes the relationships between the system under consideration, with its use cases and actors. The system represents the subject being developed and provides several functionalities to its users. These functionalities are mapped and modeled using use cases to depict how the users use the system to achieve their goals. In this situation, a use case may cover one or more scenarios that correspond to how the system interacts with its users under different circumstances. The users are described by actors, which are representing the role of a human, an organization, or any external systems that participate in the use of the system. The actors may interact directly with the system or indirectly through other actors (Friendenthal, Moore, & Steiner, 2015).

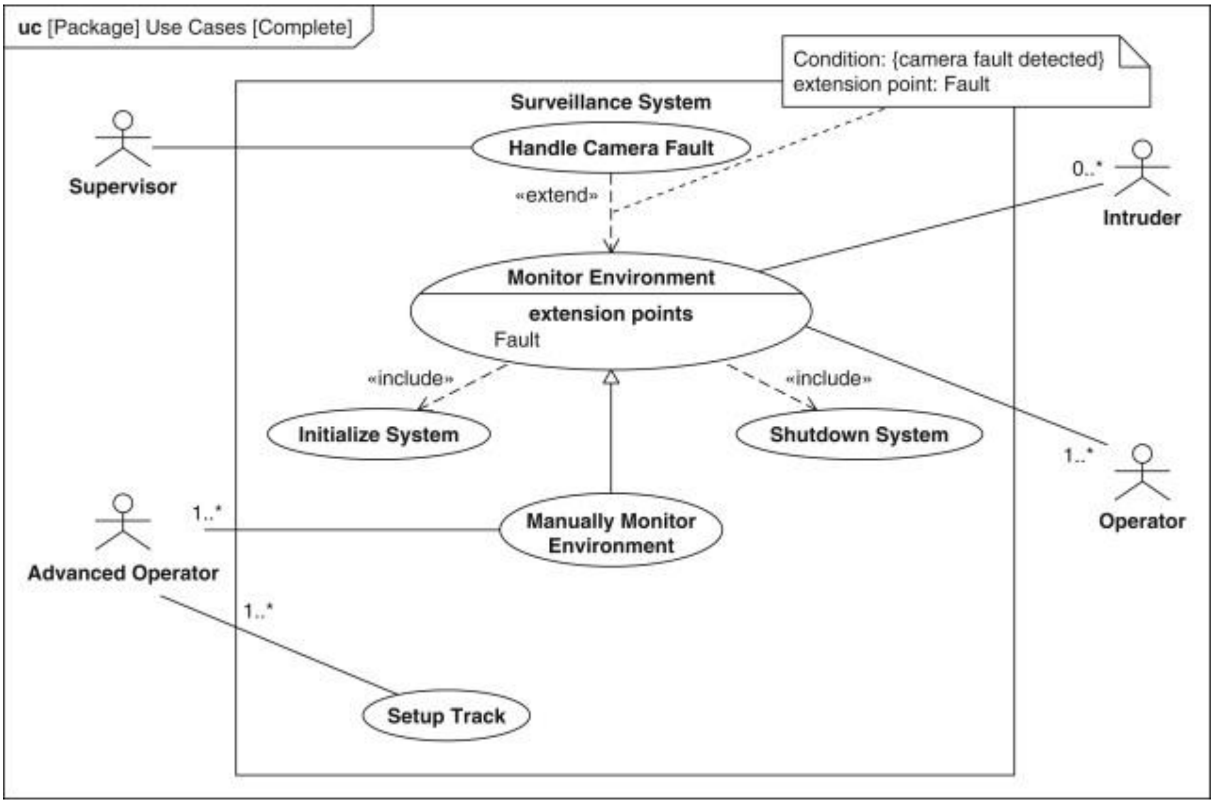


Figure 3.4. A Use Case Diagram depicting a surveillance system (Friendenthal, Moore, & Steiner, 2015)

The Use Case Diagram shown in Figure 3.4 is represented the system by a block including its name centered at the top. In this case, the name of the system of interest is *Surveillance System.* The use cases are shown as ovals with the use case names inside them. Actors are illustrated either as a stick figure with the actor's name underneath or as a rectangle containing the actor's name below the keyword <<actor>> depending on the tool or method being used. Regarding the interaction between the system and the actors is determined by the association between the actors and use cases using a standard association notation. This association also includes the multiplicity notation (0..* or 1..*) describing the number of instances of the use case in which the actor or actors can be involved at any one time (Friendenthal, Moore, & Steiner, 2015).

Besides, the relations that use cases are normally used, use cases can also be related to other cases by the following relationships: *classification, inclusion, and extension*. Classification of use cases can be applied using the standard SysML generalization relationships. Figure 3.4 shows an example of how the use case *Monitor Environment* is further classified in both the *Manually Monitor Environment* and *Automatically Monitor Environment* use case. Inclusions or extensions for use cases can be applied using dashed lines with an open arrowhead at the included and extended ends, respectively. While inclusion relationships allow one use case – referred to as the base use case – to include functionality of another use case; the extension relationships allow extending of functionalities that are not considered part of the base use case functionality. Furthermore, Interaction, Activities, and/or State Machines Diagrams can be used to model detail information of use cases (Friendenthal, Moore, & Steiner, 2015).

3.2.3. The Requirement Diagram

The Requirement Diagram graphically depicts the hierarchy of requirements defining the system or component specification. This way of modeling requirements significantly improves requirements management throughout the system lifecycle, by enabling rigorous traceability between text-based requirements, and the model elements that represent system design, analysis, implementation, and test cases. The capturing of a requirement in SysML is represented by a stereotype <<requirement>> including compartments for a name, an id and a text string describing the text-based requirements (see Figure 3.5). In addition, the requirements can be customized by adding properties such as *verification method*, *verification status*, *criticality*, *risk*, and *requirements category* (Friendenthal, Moore, & Steiner, 2015).

<<requirement>> **Operating Environment**
Id= “S1” **Text=** “The system shall be capable of detecting intruders 24 hours per day, 7 days per week, under all weather conditions”

Figure 3.5. SysML Requirement (Friendenthal, Moore, & Steiner, 2015)

Once the requirements are captured, SysML requirement provides relationships that can be used for defining a requirement structure, wherein the interactions between requirements to other requirements and models elements are manifested. Friendenthal, Moore, & Steiner (2015) distinguished the following relationships: *satisfy, verify, refine, derive, copy, trace, and containment*. While the relationships derive and copy can only relate one requirement to another, the relationships satisfy, verify, refine, and trace relate requirements to other model elements. Furthermore, their implementation can be depicted through the following notations: *direct notation, compartment notation, callout notation,* and *rationale.* The direct notation depicts a direct relationship including a dashed arrow with the name of the relation displayed as a keyword, e.g., <<satisfy>>, <<verify>>, <<refine>>, <<deriveReqt>>, <<copy>>, and <<trace>> as shown in Appendix I. Contrary to this notation, compartment notation can be used by including a compartment in the requirement block as shows in the Requirement Diagram example in Appendix I. Focusing on this diagram, it can be seen that the requirement "All Weather Operation" includes three additional compartments describing the next relations: *derived, verifiedBy, and refineBy*. On the other hand, callout notation is depicted as a note symbol graphically connected to a model element. The callout symbol references the model at the other end of the relationships. Finally, rationale is expressed using a note symbol with the keyword <<rationale>> (see Appendix I). The text in the note symbol can either provide the rationale directly or reference an external document of another part of the model such as a parametric diagram (Friendenthal, Moore, & Steiner, 2015).

3.2.4. The Block Definition Diagram

The Block Definition Diagram can be used to model and represent the structure of systems in terms of their features, hierarchy, and interconnection. For describing these structures, the diagram uses ''blocks'' as fundamental modular unit. Based on blocks, several systems, components, component interconnections, or item that flows through the system can be defined and represented. In addition, external and conceptual entities, or logical abstractions can also be constructed. The block is characterized as a rectangle that is segmented into a series of compartments. For defining the block, block features can be used. Friendenthal, Moore, & Steiner (2015) and Barosan (2017) classified these features as structural and behavioural features. At the top of the block symbol appears the name compartment which is the only mandatory compartment. If desired, the optional keyword <<block>> may be used, preceding the name compartment. The block features as, parts, operations, value properties, and ports can be presented in other compartments of the block symbol. These compartments have labels that indicate the kind of feature they contain. These labels are depicted in lower case italics, are plural, and include space between words (see Figure 3.6).

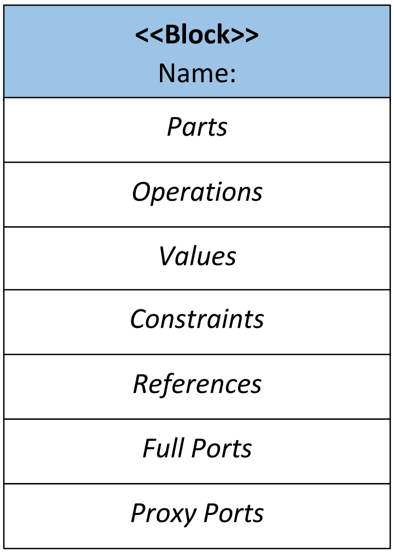


Figure 3.6. The block with its mandatory and block features

3.2.4.1. Structural features

Structural features can be seen as properties that define the characteristics of a block. Friendenthal, Moore, & Steiner (2015) and Barosan (2017) define the following categories of the properties. These are namely *part properties*, *reference properties*, *value properties*, *constraints*, *ports* and *flows*.

Part properties

Part properties describe the composition relationships between blocks. This relationship is also called a ''whole-part'' relationship. The part describes an instance or instances of a block in the context of its composite block. The potential number of instances is specified by the multiplicity of the part, which is defined as: (a) a lower bound (minimum number of instances), and (b) an upper bound (maximum of number of instances). On a given block definition diagram, the part can be shown either in a parts compartment or as composite association. A part is a feature of a block, and as such can be listed in a separate parts compartment within a block (see Figure 3.6). The parts compartment is labeled with the keyword parts and contains one entry for each part in the block. The composite association is shown as a line between two blocks, with a block diamond adornment on the whole end, and an open arrowhead on the part end (see Appendix I) (Friendenthal, Moore, & Steiner, 2015).

Reference properties

Reference properties describe the logical hierarchy that references blocks that are part of other composition hierarchies. Like part properties, reference properties can be listed in a separate compartment within block or as reference associations. The references compartment is headed by the keyword references and contains one entry for each reference property in the block (see figure 3.6). The reference association is represented as a line between two blocks, with an open arrowhead on the end of the association pointing from the owner of the reference property to the type that is referenced. There is no arrowhead on the end of the association that owns the reference property. Simultaneously, one end of a reference association may be represented by a white diamond. SysML assigns the same meaning to the association whether the white diamond is present or not. Furthermore, if the reference association is bidirectional, then there are no arrowheads on either end. Multiplicities on the ends of reference associations have the same form as for composite associations (see Appendix I) (Friendenthal, Moore, & Steiner, 2015).

Value properties

Value properties are used to model the quantitative characteristics of a block, such as its weight, speed, position or velocity. They can have multiplicity and are shown in a compartment of their owning block similar to other properties. The values compartment has the label *values* (see Figure 3.7(a)). Besides, they are based on a value type that specifies the range of valid values the property can take when describing an instance of its owning block. Value types are used to describe the values for quantities. In this case, value properties such as *total weight* and *component weight* might be typed by a value type called *kilograms (kg)*, which value can be any real number greater than or equal to 0 (see Figure 3.7(b)). A value type can be described the data structure for representing a quantity and specifies its allowable set of values. Friendenthal, Moore, & Steiner (2015) defined the following categories of value type: (a) a primitive type, which supports the definition scalar values like, *Integer, String, Boolean, and Real*; (b) an enumeration, defines a set named values called literals like colors or days of the week; and, (c) a structured type, that represents a specification of a data structure that includes more than one data element, each of which is represented by a value property.

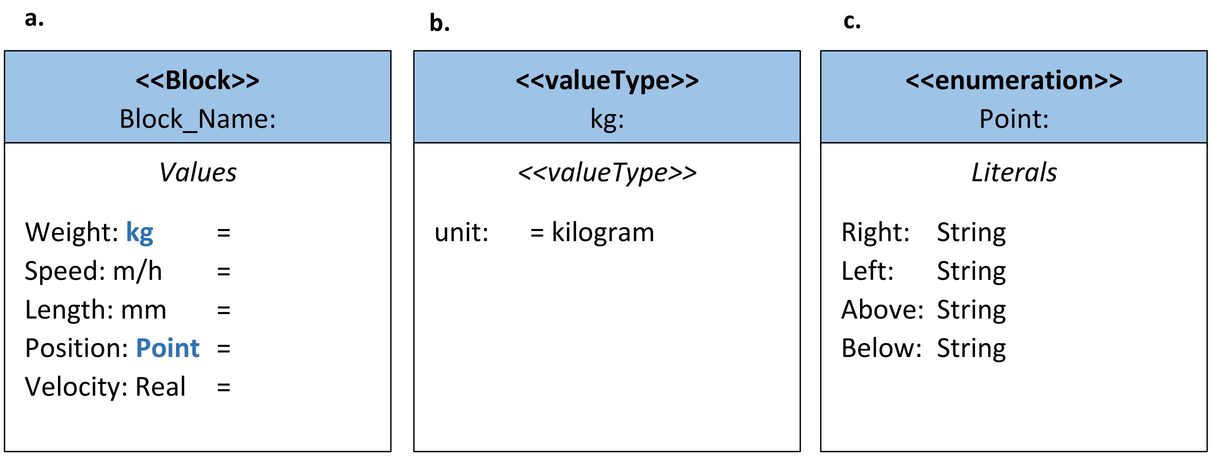


Figure 3.7. A block including value properties (a); Value type (b); Enumeration (c) (Friendenthal, Moore, & Steiner, 2015)

In the Block Definition Diagram, the value type is represented by a box symbol with a solid boundary. The name compartment has the keyword <<valueType>> preceding its name (see Figure 3.7(b)). The enumeration symbol has a single compartment, labeled *literals,* listing all the literals of the enumeration and the keyword <<enumeration>> preceding its name in the name compartment (see Figure 3.7(c)). The structured type symbol has a single compartment labeled values that list the nested value properties of the value type, using the same compartment notation as shown for other value properties (Friendenthal, Moore, & Steiner, 2015).

Constraints

Constraints can be used to restraint the properties of a block. They represent a mathematical relationship (an equation or inequality) that is imposed on a set of value properties. The constraint can be shown in a special compartment labeled *constraints*. A constraint can also be shown as a note symbol attached to the model element(s) it constraints, with the text of the constraint shown in the body of the note (Friendenthal, Moore, & Steiner, 2015).

Ports and Flows

The interactions between different parts of a system can further be specified through ports and flows, which provide an abstract view of the interaction. Ports represent an access point on the boundary of a block and on the boundary of any part or reference typed by that block. According to Friendenthal, Moore, & Steiner (2015), SysML distinguishes two kinds of ports called *full ports* (<<full>>) and *proxy* (<<proxy>>) ports. While a full port is equivalent to a part on the boundary of the parent block that is made available as an access point to and from the block. A proxy port does not constitute a part of its parent block, but instead provides external access to and from the features of its parent block or the block's parts without modifying its inputs or outputs.

On the other hand, flows may be physical in nature. Friendenthal, Moore, & Steiner (2015) described flows as water flowing in and out the pump, and electrical power flowing in the same pump. In SysML, these flows are defined as items, which is the general term to define things that flow. These items can be modeled as blocks, value types, or signals. When they are modelled as blocks, they typically include value properties that describe characteristics of the flows, such as temperature and pressure for a block that represents flowing water for instance. In addition, a flow may also be simplified to represent just a quantifiable property, in which case the item can be represented as a value type instead of a block, or as signals, which may be used to control the behaviour of a part that is target of the signal flow. Lastly, flow properties can be included in a block compartment to specify what flow in or out the block. Each flow property has a name, type multiplicity, and direction. The flow properties of a block are shown in the compartment labeled as *flow properties* (Friendenthal, Moore, & Steiner, 2015).

3.2.4.2. Behavioural features

Behavioural features define how blocks interact with their environment or modify their own state. These features can be subdivided into the following types: *operations and receptions*. Operation is typically triggered by a synchronous request (i.e., when the requester waits for a response), and may be triggered by an asynchronous request (i.e., when the requester does not wait). Each operation defines a set of parameters that describes the arguments passed in with the request, or passed back out once a request has been handled, or both. On the other hand, reception may only be triggered by an asynchronous request. It is associated with a signal that defines a set of attributes that represent the content of the message; the parameters of the reception must be the same as the attributes of the associated signal. Furthermore, operations and receptions are shown in a separate compartment of the block labeled as *operations* and are described by their signature. While the signature for an operation is a combination of its name along with parameters and optional return type. The signature for a reception is a combination of its name and a list of parameters (Friendenthal, Moore, & Steiner, 2015).

3.2.4.3. Classification and generalization

Both structural and behavioural features can be organized into a classification hierarchy for facilitating reuse of features. These features, which can be defined as classifiers according to Friendenthal, Moore, & Steiner (2015), can be described as being more general or more specialized than others in the classification hierarchy. In this hierarchy, a more specialized classifier (subclass) will inherit the common features from the general classifier (superclass), and may contain additional features that unique to it. This relationship, which is called generalization, is represented by a line between two classifiers with a hollow triangular arrowhead on the superclass end of the relationship (see Appendix I).

3.3. Domain-specific language through SysML profiling

SysML can be customized to define domain-specific semantics through domain-specific language (DSL) (Shah, Kerzhner, Schaefer, & Paredis, 2010; Valdes, 2016). In this case, several DSLs can be defined in SysML and consequently create a bridge to multiple other domains, whereby programmer productivity can be improved, and effective communication among domain experts be enhanced (Kooralla, 2018). Defining these DSLs, several researchers have manifested their preferences to SysML extensible constructs such as "profiles". For instance, Johnson, Paredis J.J., & Burkhart (2008) used in their research SysML stereotypes to abstract and capture the specialized semantics of a particular application domain, to allow integration of models and simulations of continuous dynamics into SysML. Stereotypes are used to extend SysML constructs to allow a good balance between converting some implicit Modelica semantics into SysML semantics. In addition, Shah, Kerzhner, Schaefer, & Paredis (2010) also pointed out their preferences to SysML profiles and thereby meta-models to define domain-specific description of different disciplines and languages, to create and integrate multiple views of embedded systems in SysML. In their investigation, formal definition of the domains involved in the system are defined through meta-models, and SysML profiles were created to enable DSL.

Additionally, SysML profile is adopted by Valdes (2016) to represent domain-specific bodies of knowledge, to integrate SysML with the CAD tool Siemens NX. In his research, the key aspects of the data structure of this specific CAD package are represented in a profile. As a result, allowing the SysML-CAD integration for: (a) automation of low level and highly tasks within the domain; (b) integration of application and data sets; (c) documentation and report generation; and (d) simplification and standardization of more complex processes. In another approach, Kooralla (2018) used SysML profile to model an offshore system, to define its general components and interact with databases. In this approach, the general components of the system such as the requirements, mechanical system, electrical system, hydraulic system, pneumatic system, and control system, are defined through SysML profiles, and accordingly integrated with Microsoft Excel, MS Visio and MagicDraw (SysML modeler).

3.3.1. SysML profile

A SysML profile provides constructs that extend and add new capabilities to the modeling language itself. It is a special kind of package containing a set of ''stereotypes'' and supporting definitions. Its use serves for defining a set of concepts that support a new domain, or a set of concepts that add new information to a model in a domain that is already supported. More complex profiles may contain either sub-profiles or sub-packages that further divide the overall domain into subsets of related domain concepts. Its depiction can be shown on a package diagram with ''Profile'' as the model element that corresponds to the diagram frame. Figure 3.8 and 3.9 show two domain-specific profiles that were created including the stereotypes defining its concepts.

On the other hand, the stereotypes are used to create the profiles and new model elements from existing ones, including detailed attributes suitable for domain-specific applications. For instance, one or more meta-classes in a reference metamodel can be extended by stereotypes (see Figure 3.8), or an existing stereotype can be specialized in other stereotypes (see Figure 3.9). A stereotype is represented by a rectangle with the keyword <<stereotype>> centered at the top, followed by the name of the stereotype. The extension relationship is depicted as a line with a filled triangle at the meta-class end, and the generalization relationship is depicted as a line with a hollow triangle at the general end (Friendenthal, Moore, & Steiner, 2015).

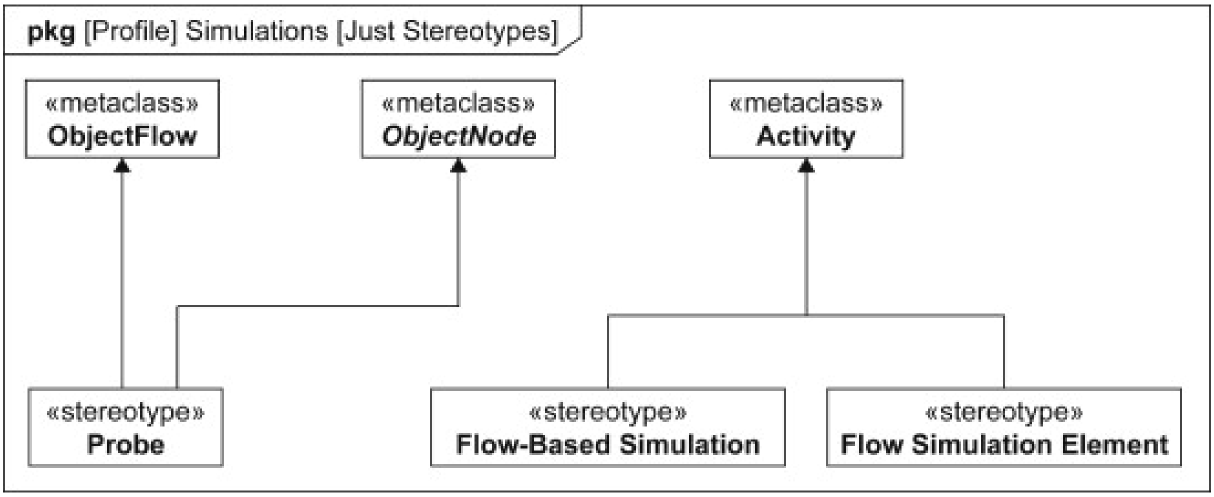


Figure 3.8. A SysML profile for a Flow-Based Simulation (Friendenthal, Moore, & Steiner, 2015)

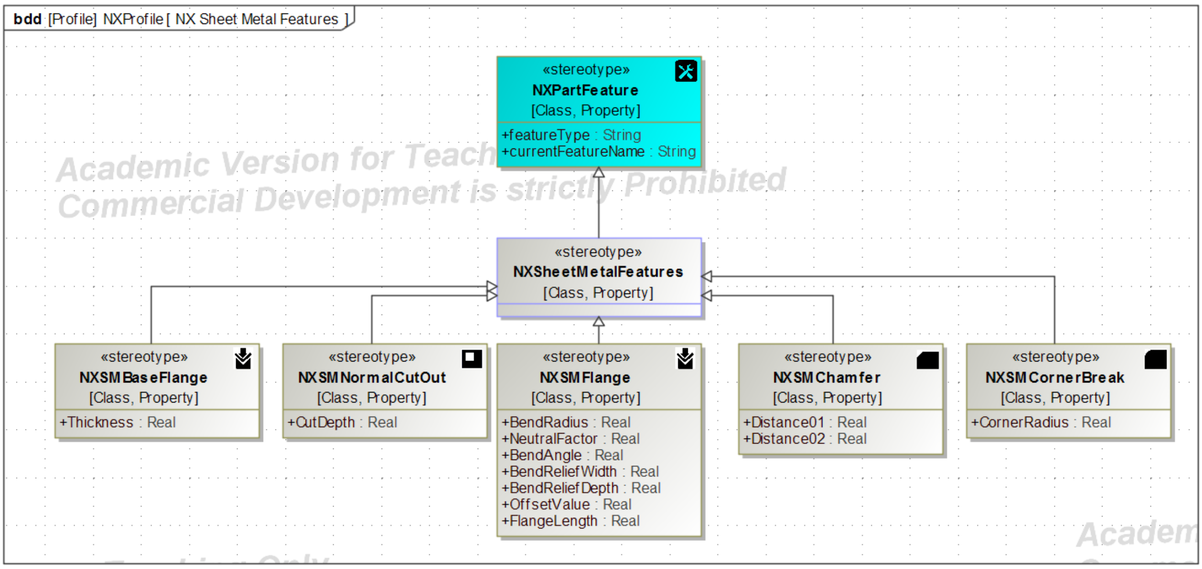


Figure 3.9. A SysML profile for an extension of the Siemens NXSheetMetal (Valdes, 2016)

3.3.2. SysML viewpoints

SysML models contain comprehensive descriptive information about the system under development and its environment. This information can be presented in diagrams and tables. Because of substantial amount of information that can be included, it is important to be able to customize the presentation of this information to support a diverse set of consumer. In this case, SysML provides viewpoints that can specify both the process-use to generate the artifacts, and how the artifacts should be presented to the stakeholders. This information may include figures, tables, plots, entire documents, presentation slides, or videos (Friendenthal, Moore, & Steiner, 2015).

3.4. Integrating SysML into SDE(s) through model transformation

As mentioned before, the SysML model can represent multiple aspects of a system under development at a fairly abstract level. Basically, it specifies the system and its components down to some level of the system hierarchy (Friendenthal, Moore, & Steiner, 2015). Thereby, it can be used with other models that represent more detailed aspects of the system or may represent other aspects of the system not addressed by the SysML model. Simultaneously, managing this large number of models in different languages, and developed by different stakeholders, can pose problems including communication on ambiguity and inconsistency of information (Johnson, Paredis J.J., & Burkhart, 2008). Therefore, integrating the SysML model with other models and structured data developed by other engineering disciplines is very important and essential to ensure cohesive model-based solutions. For this reason, the relationship between data in the different models must be maintained in order to reduce inconsistencies and improve design integrity and quality according to Friendenthal, Moore, & Steiner (2015). Accordingly, it can be achieved by ensuring a data interface between the system modeling tool and other systems development environments (SDEs) (see Figure 3.10). Typically, this connection can be created by using data exchange mechanisms as stated by Friendenthal, Moore, & Steiner (2015) additionally. They summarized the following key mechanisms: (a) manual exchange, which involves re-entering the data from one tool into another tool; (b) file-based exchange using proprietary file format or standard exchange format (e.g., XMI); (c) interaction-based exchange using Application Programming Interface (API); or (d) model transformation (see Figure 3.11).

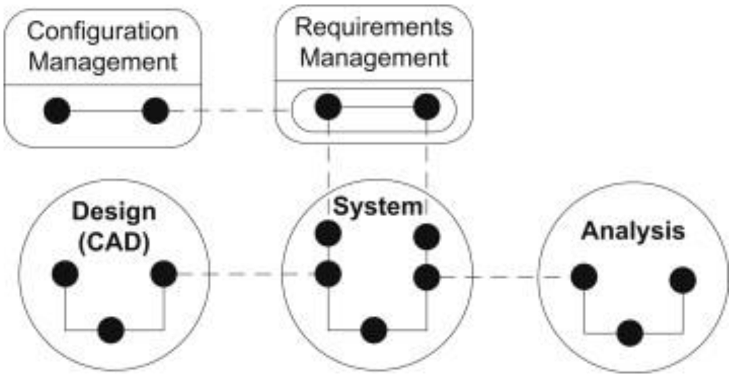


Figure 3.10. Variety of repositories and models including the integration between them (Friendenthal, Moore, & Steiner, 2015)

Considering the first two mechanisms, Shah, Kerzhner, Schaefer, & Paredis (2010) pointed out their concerns about them. They indicated that in such cases where models cannot be completely converted using interchange standards, maintaining consistency is usually left to users resulting in significant non-value-added efforts and potential error. Regarding the third one, Friendenthal, Moore, & Steiner (2015) expressed their concerns about the absence of a standard for API-based exchange, which reflects the using of point-to-point applications to facilitate exchange, since each tool has its own API. Contrary to those mechanisms, model transformation is very suitable when two sides of the exchange are different and transformation is necessary to support the data exchange. They stated that the area of model transformation is increasingly important as standard since model-based approaches and domain-specific languages are becoming more prevalent. Accordingly, several researchers included model-transformation approaches in their research to integrate especially SysML with other system environments.

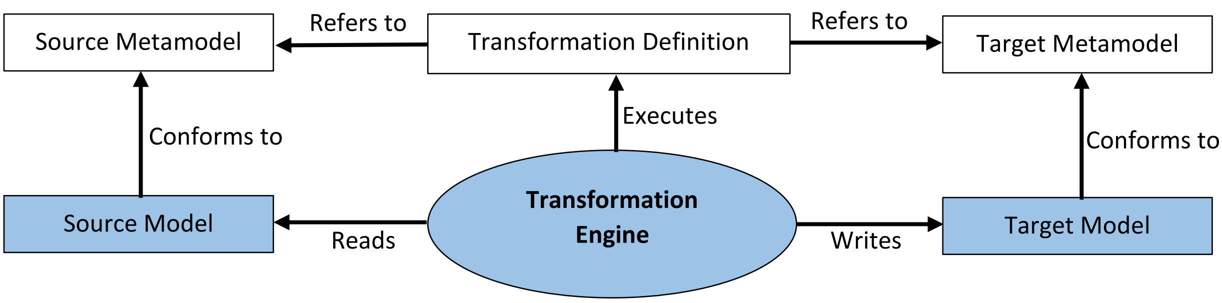


Figure 3.11. Basic concepts of model transformation (Czarnecki & Helsen, 2006)

For example, Johnson, Paredis J.J., & Burkhart (2008) used model transformation to integrate SysML and Modelica. In this example, the model transformation approaches Triple Graph Grammars (TGG) and graph transformation rules are implemented, to maintain a bidirectional mapping between the SysML constructs and the corresponding Modelica models. Shah, Kerzhner, Schaefer, & Paredis (2010) applicated model transformation in their research to generate domain-specific views in SysML, from the domain tools EPLAN and Modelica. Contrary to the previous approach, story diagrams are here used to visually define rules for the transformation between different views. These rules can be compared to the rules used in the TGG approach according to them. However, for a bidirectional transformation, additional rules in the story diagrams were required in each direction of the transformation.

In a different case, Jenkins & Rouquette (2012) used model transformation to integrate SysML and the Web Ontology Language (OWL). In this approach, the Query/View/Transformation (QVT) language is implemented to transform profiled SysML models into OWL ontologies and vice versa. Finally, Valdes (2016) used model transformation to integrate SysML and the Siemens NX CAD tool. In this approach, an innovative model-to-model transformation methodology is created and implemented, that programmatically integrated two different data structures by recreating the meta-model of the CAD application through a graph-based representation in SysML.

4. Building Information Modeling (BIM)

A long-time the AEC- industry was facing low productivity, backwardness, and wasted effort (LU, Fung, Peng, Liang, & Rowlinson, 2015; Borrmann, König, Koch, & Beetz, 2018). In response to these issues, the industry is nowadays facing a technological and digital transition due to the adoption of Building Information Modeling (BIM) in the project development process. Simultaneously, digitalization has transformed a wide range of industrial sectors, resulting in tremendous increase in productivity, product quality and product variety. Comparatively, the automotive industry has undergone the transition to digitized, model-based product development and manufacturing allowed it to achieve significant efficiency gains (Borrmann, König, Koch, & Beetz, 2018).

Contrarily, the AEC industry cannot realize the big transition to fully digitized model-based working procedures in one go for productivity improvement. Primarily, because of the well-known problems of fragmentation and discontinuity in the industry, which reflects on a fatally flawed system; under the current design, bid, build (DBB) system according to LU, Fung, Peng, Liang, & Rowlinson (2015). Professionals such as architects, engineers, surveyors, and contractors are separately contracted to perform a parcel of the work. Despite their involvement in the same project, they have competing interests, they do not necessarily interact throughout the project lifecycle, as a result that they do not always work together efficiently.

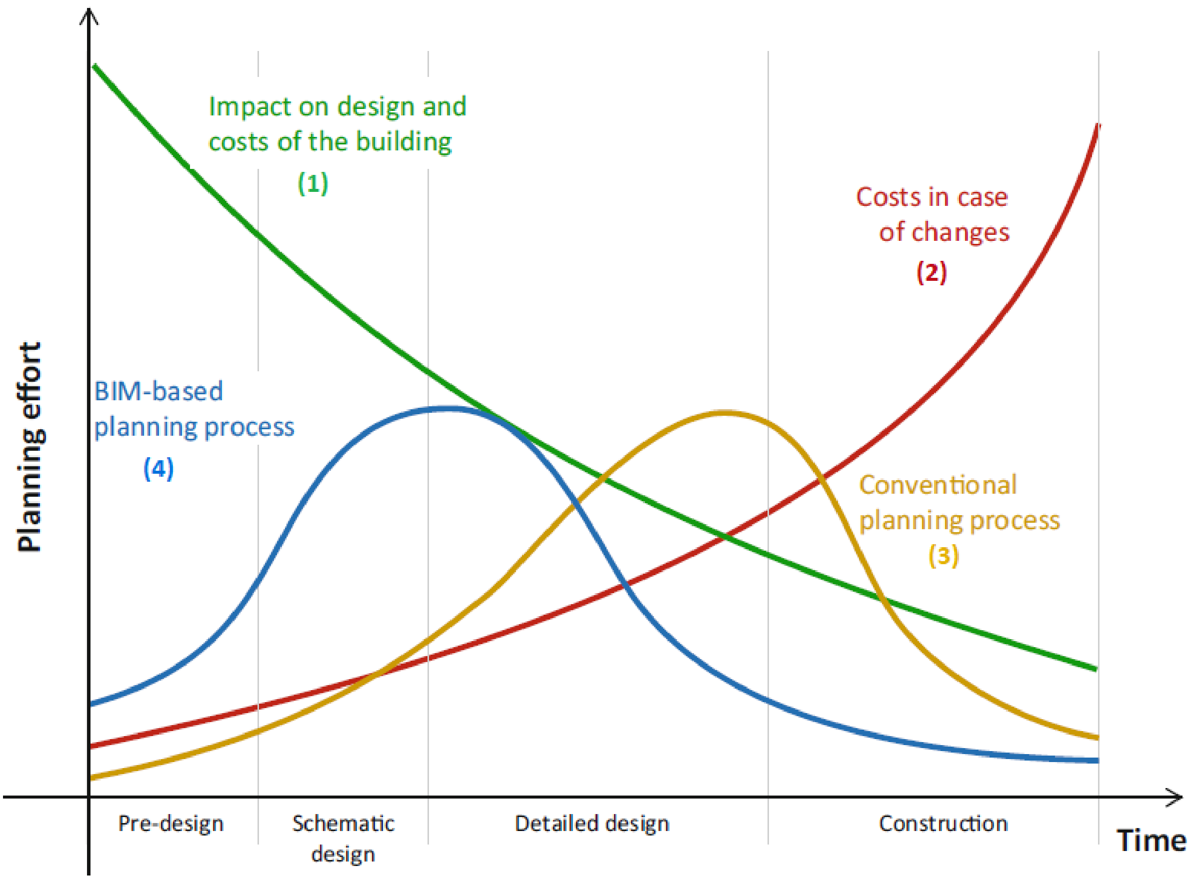


Figure 4.1. The Macleamy Time-Effort distribution graph (Borrmann, König, Koch, & Beetz, 2018)

In accordance, LU, Fung, Peng, Liang, & Rowlinson (2015) acknowledged the implementation of BIM in AEC-processes to address those inefficiencies. Additionally, they argued that BIM does not fit in current prevailing procurement systems like DBB quite well, and emphasized the need for new processes for BIM, such as integrated project delivery (IPD). Amplifying their theory, they illustrated the MacLeamy

Time-Effort distribution graph (see Figure 4.1), which shows two types of planning processes compared to their cost impacts (Construction User Roundtable, 2004; LU, Fung, Peng, Liang, & Rowlinson, 2015). Curve 1 indicates the ability to impact costs and functional as a project progresses. Curve 2 shows the cost of design change, the later the change, the more the cost it incurs. Curve 3 illustrates the traditional AEC-processes in the DBB procurement system, which involve the investment of separate effort by designers and contractors in construction documentation and management, which reflect a less ability to effect change and minimize the potential cost of design changes. Lastly, curve 4 shows the preferred process, in this case, BIM-enabled processes, encourage more effort (e.g. collaboration and open information sharing), which depict the critical concept of earliest possible decision-making to maximize the ability to effect change and minimize the potential cost of design changes.

Furthermore, the use of digital tools and information in the industry along the entire process chain fall significantly behind other domains. For instance, all too often, valuable information is predominantly handed over in the form of drawings either as physical printed plots on paper or in a digital but limited format (Borrmann, König, Koch, & Beetz, 2018). For this reason, a more appropriate approach is to introduce the new technology and the accompanying changes in processes step by step according to Borrmann, König, Koch, & Beetz (2018). Accordingly, they illustrated the BIM Maturity Model developed by the United Kingdom BIM Task Group, which defines four discrete levels of BIM implementation (see Figure 4.2). The levels are as follow defined according to them:

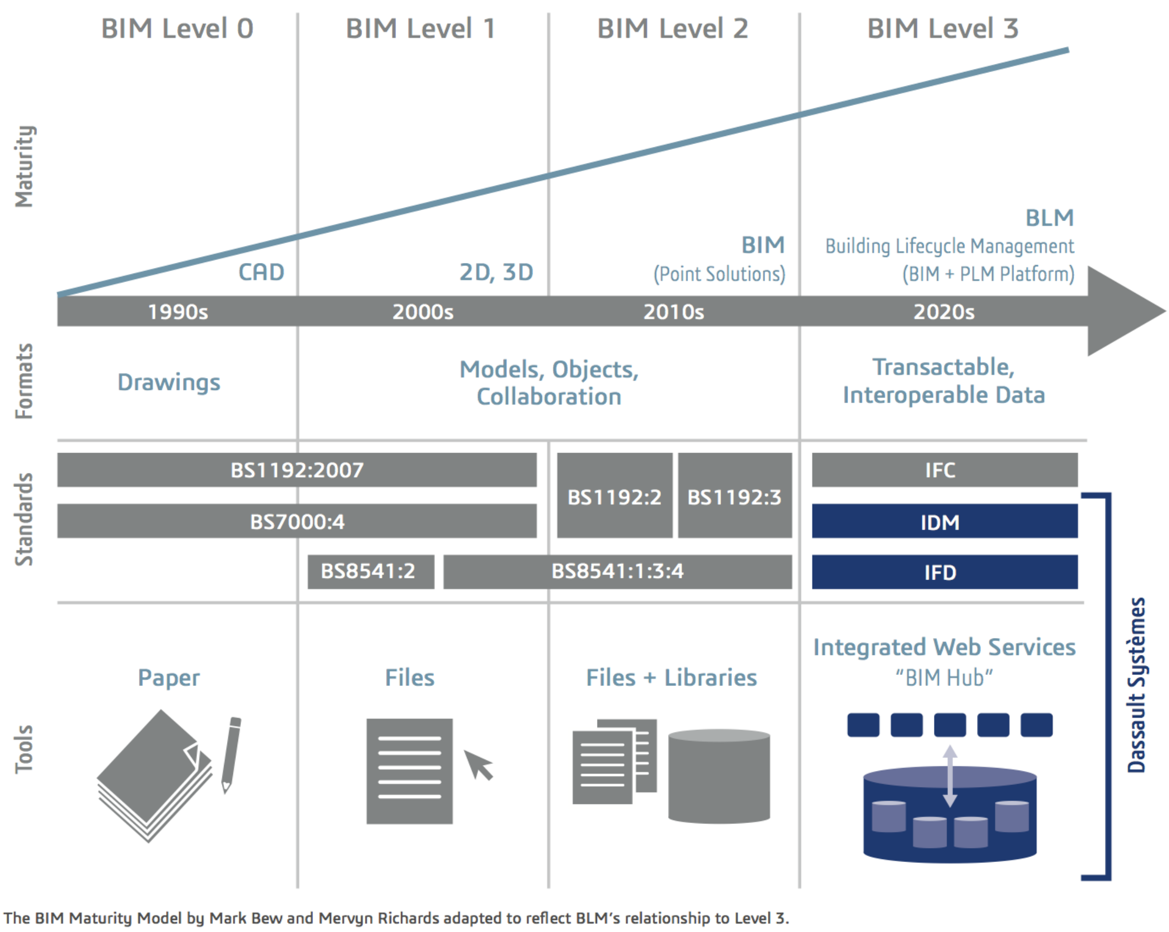


Figure 4.2. The BIM Maturity Model (Akio, 2017)

- **Level 0** describes conventional working practice based on 2D CAD whereby the exchange of information is paper-based drawings;
- **Level 1** comprises the partial 3D modeling of the facility (mostly complex geometries) while most of the design is still realized by means of 2D drawings. The exchange of information is realized through individual files, and a central project platform is not employed;
- **Level 2** is defined by the use of BIM software product for authoring digital models, whereby each of the various disciplines involved in the process, develops its own model. Whereafter each discipline model is brought together and checked for clashes or other discrepancies. Mutual consistency is achieved during the process by periodic coordination sessions or meetings. On this level, 2D drawings, which are derived from the models, are still circulating. The exchange of information is still realized on the basis of files (native formats) or open standards (not demand on this level), and managed on a central platform called a common data environment (CDE);
- **Level 3**, which is targeted for the future, is based on the concept of fully integrated BIM. The implementation is based on BIG Open BIM, i.e. ISO standards are employed for data exchange and process description, and deeply integrated digital models are used throughout the entire lifecycle. In order to maintain project data continuously and consistently over the lifecycle of the buildings, cloud services are used.

4.1. Introduction to Building Information Modeling

Building Information Modeling (BIM) can be defined as the process of creating and using digital models for design, construction, and/or operation of projects (Pels, Beek, & Otter, 2013). Supplementary, additional literatures have defined the definition explicitly in order to explain BIM better. For instance, Ilhan (2014) indicated that BIM is used both as *noun* (Building Information Model) and *verb* (Building Information Modeling). As noun, which is an unambiguously defined digital representation of the physical and functional characteristics of a facility. As verb, which is any process used to create, manage, derive and communicate information among stakeholders at various levels, using models created by different project participants at different times for different purposes. Concurrently, she indicated that BIM can also be defined in three dimensions: (a) as *product* (Building Information Model), which is a structured data set describing a building; (b) as *process* (Building Information Modeling), which is the act of creating a Building Information Model; and (c) as *system* (Building Information Management), which comprises the business work and communication structure that increase quality and efficiency throughout the lifecycle of the construction process (Ilhan, 2014). Figure 4.3 illustrates the integrated approach of BIM embracing the entire lifecycle.

Considering BIM as a process, using this approach in construction projects has a positive impact on project costs, schedule and building performance (Murguia, Brioso, Ruiz-Conejo, & Fernandez, 2017). It can be seen as a digital and managerial tool that is used to improve project visualization, information flow, system federation, planning, costing, and in general any prediction that relates to projects goals (Murguia, Brioso, Ruiz-Conejo, & Fernandez, 2017). In addition, it intents the organization and integration of all phases of the construction process, i.e. the integration and promotion of collaborative work by all the design disciplines involved in the design phase in a more effective and efficient way (Pels, Beek, & Otter, 2013; Maia, Mêda, & Freitas, 2015).

The implementation of BIM as a product, is executed through the employment of three-dimensional geometry and object-oriented CAD-model. This model has a full range of integrated value-adding services tailored for various project functions including building analysis (e.g. thermal, energy, lighting, and acoustic), structural engineering, quantity estimation, cost analysis, constructability analysis, 4D scheduling and facility management (see Figure 4.4) (Shen & K.H. Chua, 2011; Maia, Mêda, & Freitas, 2015). Furthermore, this three-dimensional model, which can be characterized as Building Information

Model, is intelligent, data-rich, inherently parametric, and object-based representation of the facility being designed and constructed (Deshpande, Azhar, & Amireddy, 2014).

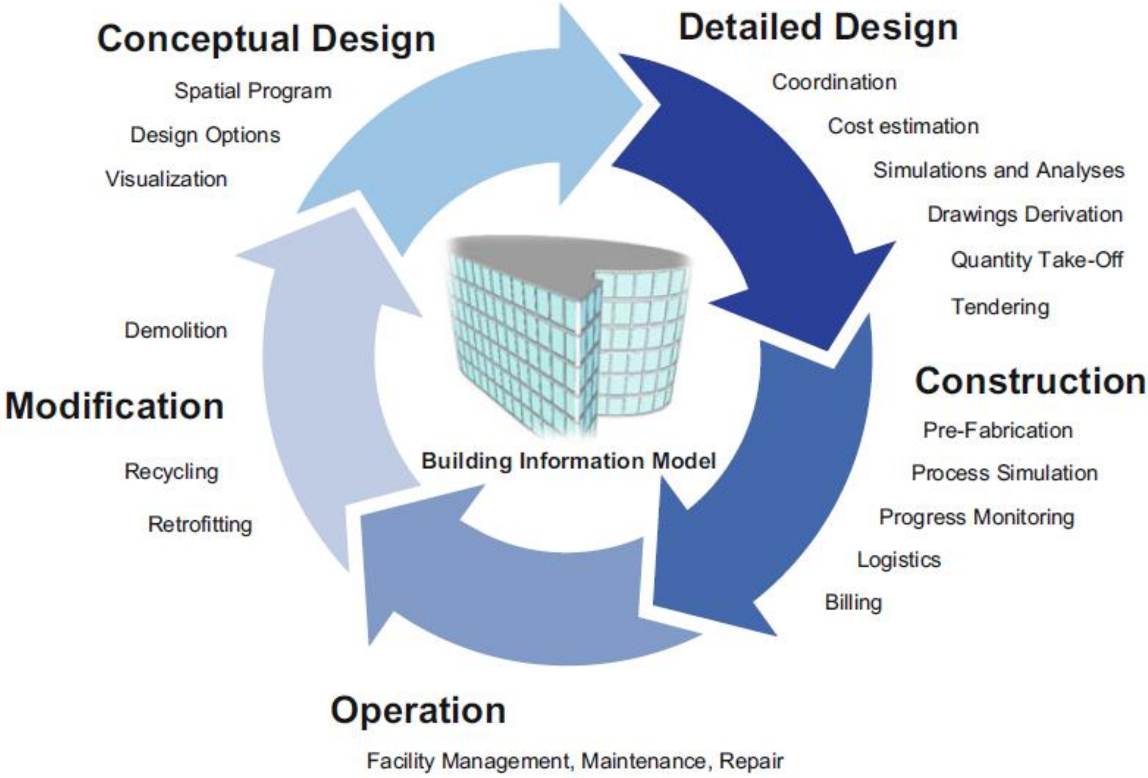


Figure 4.3. The integrated approach of BIM embracing the entire building lifecycle (Borrmann, König, Koch, & Beetz, 2018)

On the one hand, the intelligence ability is because of the information that can be introduced in the three-dimensional virtual model, which includes both graphic information (drawings) and non-graphic information (specification, schedules, and other data) (Maia, Mêda, & Freitas, 2015). BIM models are conceptualized as centralized, interconnected data stores which potentially provide a very context-rich platform for the capture, storage, and dissemination of design and construction information about architectural, structural, mechanical electrical plumbing (MEP), and heating ventilation air-conditioning (HVAC) systems (Deshpande, Azhar, & Amireddy, 2014). On the other hand, these models consist of semantics objects, that combined a parametrized 3D geometry representation with additional descriptive properties and their relationships to other elements in the model (see Figure 4.5) (Borrmann, König, Koch, & Beetz, 2018)

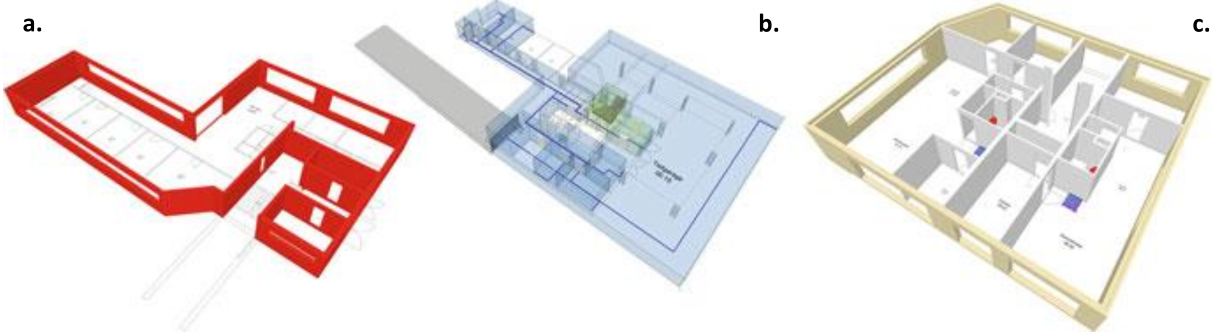


Figure 4.4. BIM Models for various project functions like, fire escape routing (a); accessibility & circulation (b); fire-safety related building components (c) (Borrmann, König, Koch, & Beetz, 2018)

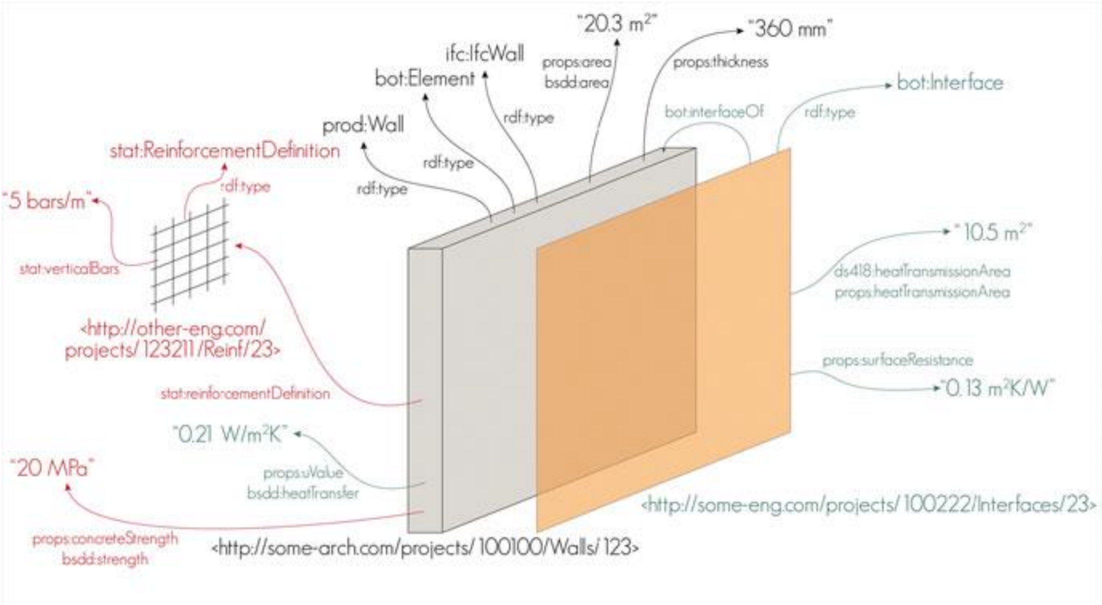


Figure 4.5. Geometry representation of a wall-element including its structure and properties (Borrmann, König, Koch, & Beetz, 2018)

4.2. Interoperability in the AEC-industry: "BIG Open BIM"

As shown in Figure 4.6(a), the AEC-industry is characterized by a highly fragmented process with a numerous different and independent participants using a wide range of software tools to perform their tasks (Beetz, 2009; Kiviniemi & Laakse, 2012; Borrmann, König, Koch, & Beetz, 2018; Zhang, et al., 2011). Unfortunately, most of them use proprietary formats for information exchange with tools that are still " islands of automation" according to Borrmann, König, Koch, & Beetz (2018). It means that these tools have or only limited support for data exchange between the separate applications, making interoperability of integrated building information within the industry difficult. Consequently, data and information that already exists in digital form need to be re-entered manually, which is laborious, and prone to introducing delays, new errors, and costs due to redundant data entry, redundant IT systems and IT staff, and inefficient business processes (Kiviniemi & Laakse, 2012).

In this situation, interoperability and/or an interoperability standard between information systems offers potential for considerable saving and financial gain according to Kiviniemi & Laakse (2012). Interoperability can be seen as the ability to pass data between applications, and for multiple applications to jointly contribute the work at hand (see Figure 4.6(b)). This approach eliminates the need to manually copy data already generated in another application resulting in a loss-free exchange of data between software products by different vendors (Eastman, Teicholz, Sacks, & Liston, 2011).

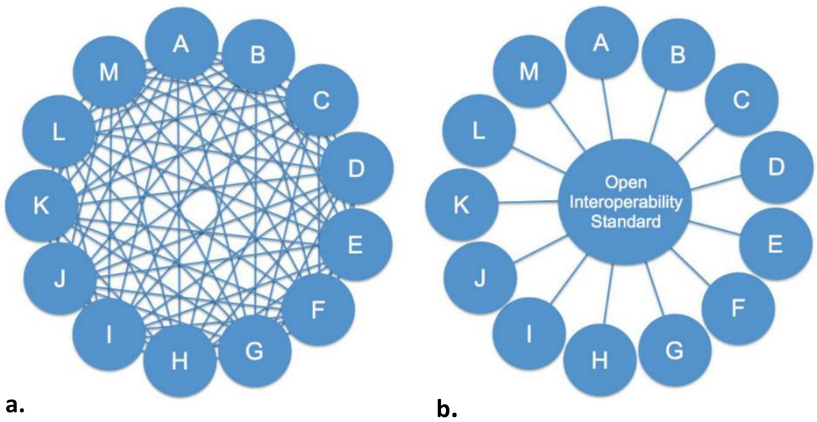


Figure 4.6. Exchange of information between project participants (a); an Open Interoperability Standard for data interoperability between project participants (b) (Kiviniemi & Laakse, 2012)

In this case, the Industry Foundation Classes (IFC) as an open interoperability standard (see Figure 4.6(b)), can provide a comprehensive, standardized data format for a vendor-neutral exchange of digital building models (Borrmann, König, Koch, & Beetz, 2018). It is developed by the international organization BuildingSMART, which has dedicated many years to the development of this standard in response to the claim of the industry for a neutral data exchange format. Representing both geometry and semantic structure of building models; it is an essential basis for the establishment of the BIG Open BIM goal according to Kiviniemi & Laakse (2012), which involves a consistent model-based communication between all stakeholders and across the entire lifecycle of a facility (BIG BIM), and a utilization of open vendor-neutral data formats to allow data be exchanged between products by different software vendors (Open BIM) (Borrmann, König, Koch, & Beetz, 2018). In conclude, it has the potential to bridge the connection between stakeholders and project phases in a fragmented project environment as the AEC-industry.

4.3. The Industry Foundation Class (IFC) standard

The literature has shown that there are several opinions about a common interpretation or meaning about the Industry Foundation Class (IFC) standard. For instance, Maia, Mêda, & Freitas (2015) seen the standard as a repository of data. Venugopal, Eastman, & Teizer (2015) described it as a data-model schema, and Eastman, Teicholz, Sacks, & Liston (2011) seen it merely as a schema. Despite these different descriptions, they definitely share the same opinion about the main function of the standard and its benefits. Whereby, it can be acknowledged as a neutral open and object-based standard that provides a data-model schema for interoperability of information throughout the AEC-FM project lifecycle: From feasibility and planning, through design (including analysis and simulation), construction, to occupancy and building operation. For this purpose, the standard provides geometry, relations, processes, material performance, fabrication and other properties based on the ISO-STEP EXPRESS language and concepts (Venugopal, Eastman, & Teizer, 2015). These characteristics form the conceptual organization of the standard, which define its structure that is composed via a four-layer architecture. This architecture structure consists of: *Resource layer*, *Core layer*, *Interoperability layer,* and *Domain layer* (see Figure 4.7) (Ilhan, 2014).

4.3.1. The IFC four-layer architecture

Figure 4.7 shows the four-layer architecture of IFC. The structure, which is both extensive and complex, is composed of several layers in order to improve its maintainability and extensibility (Borrmann, König, Koch, & Beetz, 2018). Each layer involves a number of classes (entities) that are intended to define or reference other layers for particular purposes. The following paragraphs give a description about them briefly.

The Resource layer provides general classes that can be used to commonly define any of the other higher-level layers. Contrary to the layers above, these classes do not derive from *ifcRoot* and therefore have no identity of their own. Furthermore, they describe items like *geometry*, *material*, *measurement actors*, *roles*, *property*, *presentation,* and *cost* (Eastman, Teicholz, Sacks, & Liston, 2011; Ilhan, 2014).

The Core layer contains the most elementary classes, which define the basic structures, key relationships and general concepts. These classes can be referenced by all layers above and re-used and defined by precisely classes in the upper layers. Which makes the classes in this layer and above supertype of *ifcRoot*, which are provided with a unique identification, a name, description, and change control information (BuildingSMART, 1996-2006). In addition, the layer is composed of a kernel and three specific extensions. The kernel comprises the basic abstract classes and the three extensions which consist of *product extension*, *process extension* and *control extension*. The product extension schema describes the physical and spatial objects of a building and their respective relationships.

The process extension schema comprises classes for describing processes and operations. It provides a basic means for defining dependencies between process elements for linking them with resources. The control extension defines the basic classes for control objects as well as possibilities for allocating these objects to physical and spatial objects (Borrmann, König, Koch, & Beetz, 2018).

The Interoperability layer, which is also known as the shared layer, represents an interoperability layer between the core and the domain-specific schemes. This layer provides modular classes that are derived from classes in the Core Layer, which can be used by more than one domain model (Ilhan, 2014).

The Domain layer encloses highly specialized classes that only apply to a particular domain. According to Borrmann, König, Koch, & Beetz (2018), they form the leaf nodes in the hierarchy of inheritance. The classes in this latter cannot be referenced by another layer or by another domain-specific schema.

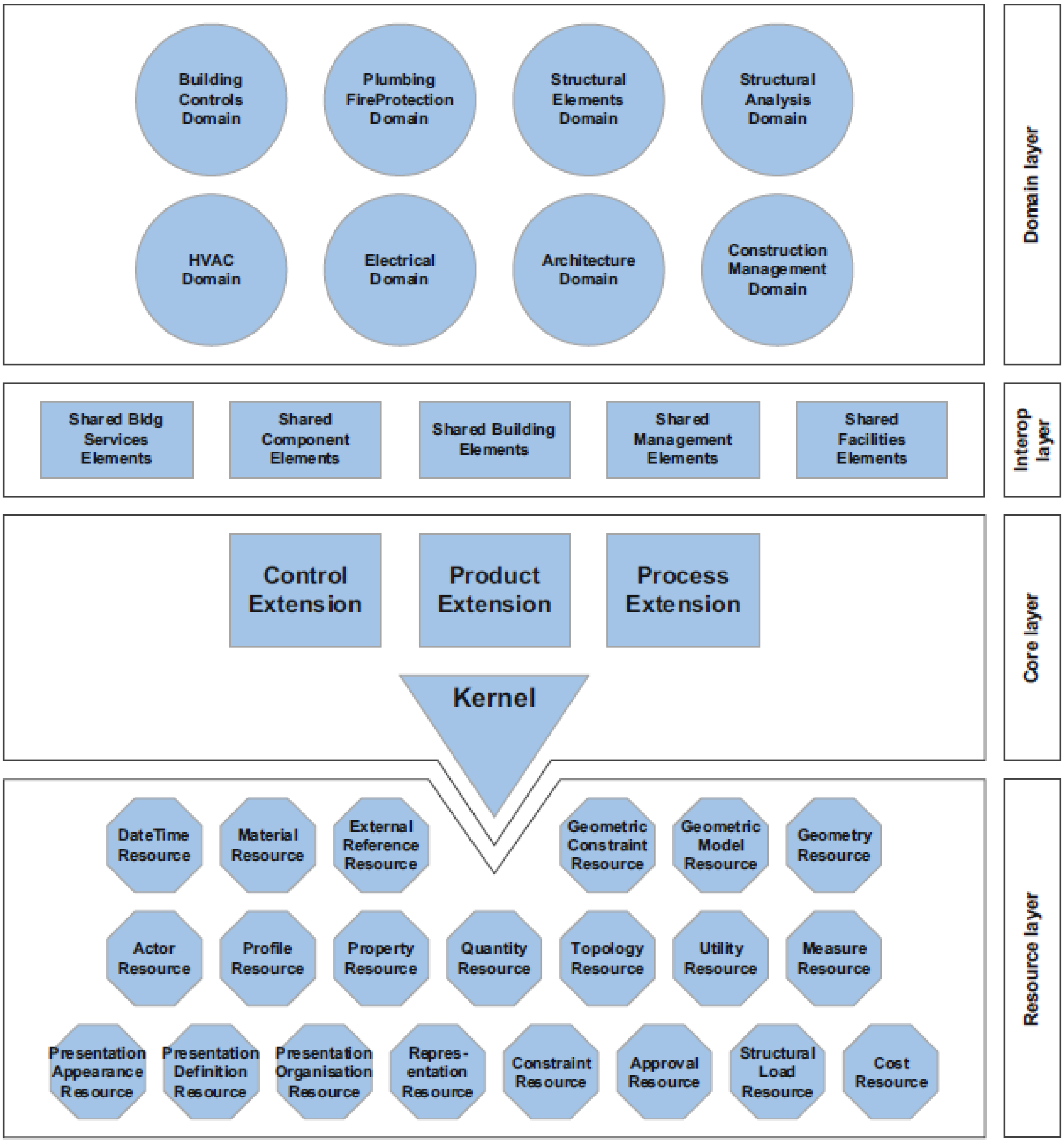


Figure 4.7. The IFC four-layer architecture (Borrmann, König, Koch, & Beetz, 2018)

4.3.2. Inheritance hierarchy

The IFC-standard is besides an extensible data model also an objected-oriented one (Eastman, Teicholz, Sacks, & Liston, 2011). In objected-oriented data models, inheritance hierarchy plays a crucial role, which is also the case for the IFC-standard (Borrmann, König, Koch, & Beetz, 2018). Inheritance hierarchy defines specialization and generalization relationships and therefore which attributes of which classes can be inherited by other classes. Basically, it follows a semantic approach. Figure 4.8 shows part of the IFC-model showing the most important classes of the inheritance hierarchy.

As can be seen in this figure, the class *ifcRoot* is the starting point and root of the IFC inheritance tree. As mentioned before, all classes, with the exception of those in the Resource layer, must derive directly or indirectly from *ifcRoot*. This class provides basic functionality for uniquely identifying an object using a Globally Unique Identifier (GUID), for describing ownership and the origin of an object, and to map the history of changes made to it. In addition, every object can be given a name and description (Borrmann, König, Koch, & Beetz, 2018). Directly from *ifcRoot,* the next level of the inheritance level follows the classes *IfcObjectDefinition, IfcPropertyDefinition,* and *IfcRelationship.* The following sections will briefly give a description of the key characteristics of these classes.

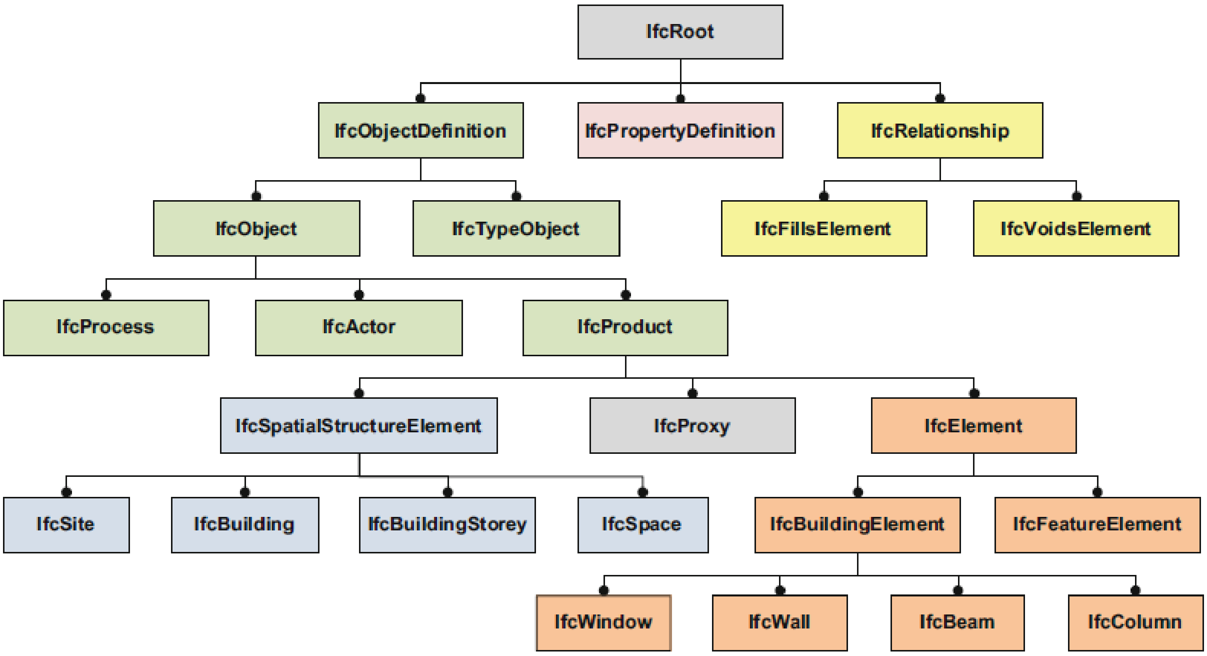


Figure 4.8. The IFC inheritance hierarchy including the key classes (Borrmann, König, Koch, & Beetz, 2018)

4.3.2.1. IfcObjectDefinition

The class *ifcObjectDefinition* is an abstract superclass for all classes that represent physical objects (e.g. building elements), spatial objects (e.g. openings and spaces) or conceptual elements (e.g. processes, costs, etc.). It also includes definitions for describing others involved in the building project according to Borrmann, König, Koch, & Beetz (2018). These classes are represented by the three subclasses of *ifcObjectDefinition,* namely (a) *IfcObject*: which represents an individual object as part of a building project; (b) *IfcTypeObject:* object type*,* and (c) *IfcContext:* which is the generalization of a project context in which, objects, type objects, property sets, and properties are defined (BuildingSMART, 1996-2006).

4.3.2.2. IfcPropertyDefinition

The class *ifcPropertyDefinition* defines those properties of an object that are not already part of the IFC data-model. These properties, better known as *property sets* or *P-sets*, are sets of properties that are used together to define material, a particular type of performance, and contextual properties such as wind, geological, or weather data. In addition, they are collected P-sets for many building objects such as common roof, wall, window glazing, window, and beam reinforcements, and many properties associated with different material behaviours such as for thermal material, products of combustion, mechanical properties, fuels, concrete, reinforcing, and others (Borrmann, König, Koch, & Beetz, 2018). Figure 4.9 shows an example of properties assigned to a door.

On the other hand, there are also key characteristics of objects, which can be defined directly within a schema of the IFC-model with the help of attributes in an entity definition. For instance, standard doors and windows with their absolute height and width can be specified by the attributes *OverallHeight* and *OverallWidth* according to Borrmann, König, Koch, & Beetz (2018) and BuildingSMART (1996-2006).

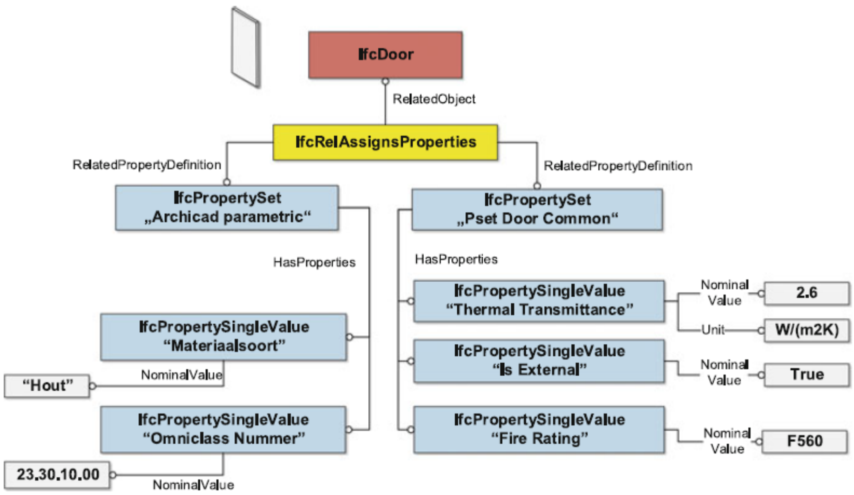


Figure 4.9. Properties assigned to ifcDoor (Borrmann, König, Koch, & Beetz, 2018)

4.3.2.3. IfcRelationship

The class *IfcRelationship* and its subclasses describe objectified relationships. These kinds of relationships are an important part and a powerful function of the IFC data-model according to Borrmann, König, Koch, & Beetz (2018). The reason is due to its ability to describe relationships, along with the semantic classification of objects, which is a fundamental aspect of an ''intelligent'' building information model that not only records building elements as isolated bodies; however, it highlights their function and interaction with other objects. The relations are not formed by direct association but instead with the help of a special intermediary object that represents the relationship itself. An example is illustrated in Figure 4.10, where the principle of objected relationships using the example of a wall, opening, and window is drawn. In addition, Figure 4.11 shows the inheritance tree of the object relationship, where the element *IfcRelationship* the root forms and every relationship can have an informal description that details the precise purpose for using this relationship (Borrmann, König, Koch, & Beetz, 2018).

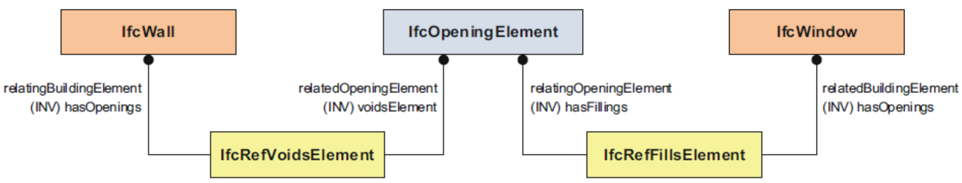


Figure 4.10. Intermediary objects representing relationships (Borrmann, König, Koch, & Beetz, 2018)

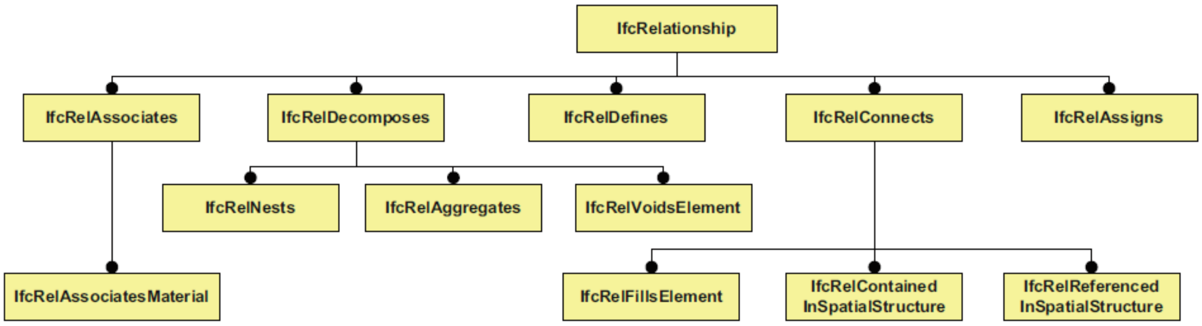


Figure 4.11. The inheritance tree of the object relationship (Borrmann, König, Koch, & Beetz, 2018)

4.3.3. The formal data modeling language of IFC: *ifcEXPRESS*

The data modeling language EXPRESS is a declarative and representational language, in which one can define object-oriented data models (Borrmann, König, Koch, & Beetz, 2018). This language is defined based on the STEP library definitions developed by the ISO. STEP is a comprehensive ISO standard that provides digital data information, representation, and exchange. It provides a mechanism that is capable of describing product data throughout the lifecycle of a product independent from any particular system (Ilhan, 2014). This standard was developed to address the advance data exchange issues of upcoming object models at that time, which led to the developing of the EXPRESS-language as main product of the STEP-standard (Eastman, Teicholz, Sacks, & Liston, 2011). Accordingly, EXPRESS has become the central mechanism to support modeling of products across a broad range of industries namely: mechanical and electrical, process plants, shipbuilding, process plans, furniture, finite elements models, and others, as well as buildings and bridges (Eastman, Teicholz, Sacks, & Liston, 2011; Ilhan, 2014).

In addition to the EXPRESS textual notation (see Appendix II), which is a machine-readable language and excellent for computational use, although difficult for human use; a graphical display (see Appendix II) version of the language – called EXPRESS-G – was developed. Simultaneously, the EXPRESS-I was developed to provide means of displaying instantiations of EXPRESS data-elements (Schenk & Wilson, 1994). Representing and defining the IFC data-model, the EXPRESS-schema enables a description of construction-related information through a network of entities, attributes, simple and select types, enumerations, collections and relations (Pauwels, et al., 2011). These language elements are used to build a specific EXPRESS-schema according to Schenk & Wilson (1994). Appendix II covers the key characteristics of these elements briefly.

Despite the fact that IFC is currently considered one of the most appropriate schema for improving information exchange and interoperability in the AEC industry for: automatic building cost calculation, simulation of a four-dimensional building schedule, escape route analysis, fire and evacuation simulation and others. Its deployed language has some limitations regarding limited-expression range, difficulties in portioning the information, and multiple descriptions of the same information according to (Pauwels, et al., 2011). Venugopal, Eastman, & Teizer (2015) acknowledged the ambiguous and implicit nature of the schema, and indicated the need for a formal and definition of IFC-concepts for machine readability and definitely interoperability. Lastly, Pauwels, Zhang, & Lee (2017) address the number of challenges in using the schema, namely: the heterogeneous IFC translation and binding processes; limitation in quick adaption of the schema; and the difficulty in extending the schema. In this case, they investigated whether semantic web technologies might provide alternative solutions.

4.3.4. The Extensible Mark-Up Language (XML) representation for IFC: *ifcXML*

IfcXML is an industry-standard for IFC building models based on the global Extensible Mark-Up Language (XML) (Cheng, Kumar, & Law, 2002; Zhang, et al., 2011; Ilhan, 2014). The standard is proposed by the BuildingSMART organization in addition to the ifcEXPRESS, in order to provide homogeneity and interoperability between the IFC data-Model and XML-based applications (Ilhan, 2014; Pauwels, Zhang, & Lee, 2017). The standard is automatically created from a transformation using XML representation of the EXPRESS schema and data (Cheng, Kumar, & Law, 2002), which ensures that both ifcEXPRESS and ifcXML handle the same data consistently, and the data files between them can be converted bi-directionally according to Ilhan (2014). The translation takes place in two steps: (a) the ifcEXPRESS source is translated into the raw ifcXML schema, which enables the exchange of IFC-data in XML-formats; (b) the raw schema is further optimized into more native XML-schema (Cheng, Kumar, & Law, 2002).

IfcXML does not only deal with geometry and product data; however it also supports lifecycle project information, including architecture, HVAC, construction and facility management. Some researchers have recognized these performances and shown the potential of using ifcXML as knowledge format. For instance, Cheng, Kumar, & Law (2002) used ifcXML as knowledge format for building a question answering (QA) system for the project management domain. The benefits of ifcXML in this research were evident. They concluded that ifcXML provides many of the terms and relationships commonly used in project management applications. Its XML-based schema is easy for querying and transforming on the internet. Zhang, et al. (2011) adopted ifcXML through an implementation of an application to evaluate the usability of standard schemas in the AEC-domain. Their main goal was developing a novel technology to support the task of extracting all manners of construction information from their native applications, and integrating them to more effectively support project participants in making critical decisions for large construction projects. According to them, ifcXML accordingly provides the most complete representation of the concepts in the domain ontology.

On the other hand, the researchers pointed out some concerns about the standard. For example, Cheng, Kumar, & Law (2002) have concern about the transformation, namely that the translation between the two languages is not a one-to-one mapping, consequently, information can be lost during the translation process. However, this concern had not hindered the objective of their research. Besides, the system can easily accommodate new terms and relationships as they became available in the ifcXML schema according to them. Lastly, Zhang, et al. (2011) indicated the complexity of the standard representing information about the building in concern, which would be exceedingly large and difficult not only to navigate although to query as well – see Appendix II.

4.3.4.1. The Extensible Mark-Up Language (XML)

The Extensible Mark-Up Language (XML) is an accepted data-format standard for data interchange on the web allowing structuring of data on the web (Cheng, Kumar, & Law, 2002; Cardoso, 2007).

It supports multiple handling of schemas. Some of these schemas are published and public, while others are proprietary (Eastman, Teicholz, Sacks, & Liston, 2011). According to them, each of these different XML-schema defines its own entities, attributes and relations, and rules. Due to their expressiveness and extensibility, they support exchange of many types of data between applications. For example, each of this different XML schema defines its own entities, attributes and relations, and rules as further indicated by them. Furthermore, users can define their own custom tags and structures and therefore a data representation that is tailored to their needs (Zhang, et al., 2011). In addition, XML provides query languages as XQL, XML-QL, LOREL, XPATH, and XQuery in order to query information from XML-files (Cheng, Kumar, & Law, 2002). For these reasons, XML is very popular for information exchange between application in the construction and manufacturing industry as well as in the business world (Eastman, Teicholz, Sacks, & Liston, 2011).

XML is an extension to the HyperText Mark-Up Language (HTML) – the base language of the web – to complement the problems regarding data exchange and integration, which have been impossible to solve using HTML according to Cardoso (2007). Despite the syntactic interoperability and integration of XML, it has some limitations regarding semantic interoperability. The data interchange and structuring on the web are without communicating the meaning of the data according to him. Fundamentally, XML aims at document structure and imposes no common interpretation of data, as a result, that is no way to recognize the semantics of particular domain. It has a weak data model incapable of capturing semantics, relationships, or constraints. Regarding the automated interoperability of systems, human involvement is always required. Nevertheless these limitations, it is part of the set of technologies that constitute the foundations of the semantic web (Cardoso, 2007).

4.3.5. The Ontology Web Language (OWL) representation for IFC: *ifcOWL*

The IFC data-model, which is defined as an EXPRESS-schema and translated into an XML-schema, is also obtainable as an RDF-schema and an OWL-ontology, to provide valuable interoperable advantages to the AEC-industry (Pauwels, Zhang, & Lee, 2017). The RDF/OWL-standard is a modeling language to describe any kind of information, not limited to the World Wide Web (WWW) according to Pauwels, et al. (2011). In this way, information implicitly becomes interoperable between different environments, whether these environments be web pages, complete software environments, or other ones. The standard is part of the Semantic Web domain developed by the World Wide Web Consortium (W3C), to extend the current web in which information is given well-defined meaning, for enabling computers and people to work in cooperation perfectly (Berners-Lee, Hendler, & Lassila, 2001).

Within the AEC-industry, the Semantic Web has received an increased attention for its application here, for introducing semantics into building information modeling by means of using Semantic Web technologies and techniques (Venugopal, Eastman, & Teizer, 2015; Pauwels, Zhang, & Lee, 2017). According to Venugopal, Eastman, & Teizer (2015), these semantics are necessary for reasoning tools, to check consistency, subsumption relationships and querying. Considering the lacking of formal semantics of IFC and its EXPRESS-schema, those efforts have led to the joint international standardization of *ifcOWL*: the Ontology Web Language representation of the Industry Foundation Classes under the umbrella of the BuildingSMART standardization organization (Borrmann, König, Koch, & Beetz, 2018).

This standardization, which is an integration between the semantic web technologies and IFC-standard, will allow to: (a) keeps using the well-establishing IFC-standard for representing construction data; (b) exploit the enablers of semantic web technologies in terms of data distribution, extensibility of the data-model, querying, and reasoning; and (c) re-use general-purpose software implementation for data storage, consistency, checking and knowledge (Pauwels & Terkaj, 2016). The following sections will cover the key aspects of this standardization briefly.

4.3.5.1. Semantic Web technologies

The Semantic Web was introduced in the World Wide Web as an extension on the traditional Web for making the processing of Web information by computer possibly (Cardoso, 2007). As it is acknowledged by Cardoso (2007), most of the information on the web is designed only for humans consumption instead to additionally allow their interpretation by computers. This traditional web, also known as the web of documents, encodes information by means of natural language, pictures, videos, and others, which make the meaning of information implicit or hidden in the context (Stancheva, 2017). Unfortunately, this information is only understandable by humans regardless of its ambiguity.

However to be correctly interpreted and understood by machine, the information has to be additionally described according to Stancheva (2017). Otherwise, an unable and inefficiently cooperation between computers and people as a result.

In response to this inefficiency, new internet business model required organizations to search for solutions to enable deep interoperability and integration between their systems and applications (Cardoso, 2007). This effort led to an emergent solution, which was defining the information on the web using semantics and ontologies in a way that could improve interoperability and integration. In this way, the Semantic Web was made through incremental changes to the Web, by bringing machines-readable descriptions to the data and documents already on the web (Cardoso, 2007).

Simultaneously, the Semantic Web can be defined as a semantic network in which information or domains are represented and combining as directed labeled graphs (Pauwels, Zhang, & Lee, 2017). This network consists of inter-linked data available in a standard format, reachable and manageable by automated tools (Venugopal, Eastman, & Teizer, 2015). The graphs are composed of nodes that represent concepts or objects in the world, and arcs that represent the logical relations between two of these concepts or objects (Pauwels, et al., 2011; Pauwels, Zhang, & Lee, 2017). To represent this structure easily, the Semantic Web uses the Resource Description Framework (RDF) standard as language, and for an improved semantic structure the RDF vocabularies or ontologies. These technologies together with other ones shown in figure 4.11 form the Semantic Web stack of integrated technologies to support Semantic Web and Linked Data. The following paragraphs cover the most relevant ones.

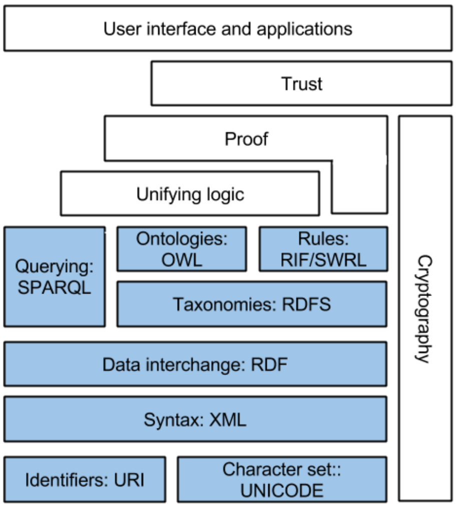


Figure 4.11. The Semantic Web Stack (Alsharif, 2013)

The Resource Description Framework (RDF) standard

On the top of the universal resource identifiers (URI), uniform resource name (URN) and XML as shown in the Semantic Web stack of integrated technologies figure (see Figure 4.11), lies the RDF-language. This language is developed for describing any web and resource in a machine way while exchanging information (Venugopal, Eastman, & Teizer, 2015). While XML was developed for enhancing integration of applications and systems; it is was not sufficient since the data exchanged lacked an explicit description of its meaning (Cardoso, 2007), which means that the integration of applications must also include a semantic integration. This situation led to the development of RDF to allow a richer integration and interoperability of data among communities and domains according to him.

RDF strives to add formal definition to resources by providing a data-model based on the ''subject-predict-object'', commonly known as the RDF ''triple'' as shown in Figure 4.12(a). A set of triples can be seen as statements containing concepts or objects in the world and their relations, resulting in an RDF-graph (Pauwels, et al., 2011). Each concept and relation are uniquely defined by a URI, making the RDF-graph explicitly labeled as further illustrated by them. The resulting graph can be converted into a textual representation that follows a specific syntax. Figure 4.12(b) shows an example of such triple, which depicts a subject: Window X; with a property: overall height; and a value: 2100 mm.

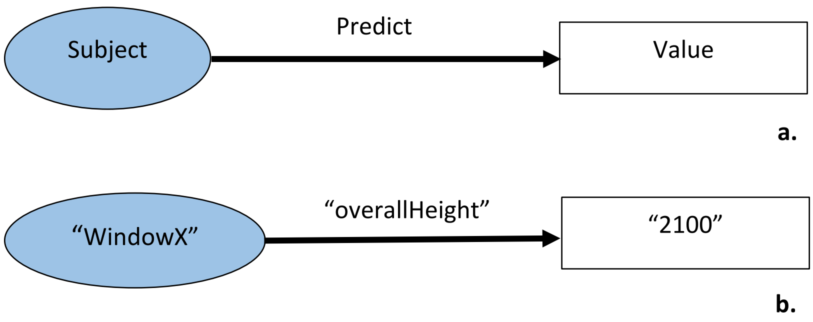


Figure 4.12. RDF triple: "subject-predict-object" (a); an example of such triple named "Window X overallHeight 2100" (b) (Cardoso, 2007)

The Resource Description Framework Schema (RDFS)

As mentioned before, the RDF-graph can be given an improved semantic structure using RDF vocabularies or ontologies. The most important basic elements to describe such ontologies are available in the RDF-schema (RDFS) vocabulary according to (Pauwels & Terkaj, 2016). It provides the RDF -graph with a type of system specification in order to build object models from which the actual data are referenced, and things can semantically be defined (Cardoso, 2007). This specification consists of classes, subclasses, comments and data types to allow users to define resources (Cardoso, 2007; Pauwels & Terkaj, 2016; Pauwels, Zhang, & Lee, 2017).

Additionally, Cardoso (2007) declared that a class can be viewed as a structure of similar things and inheritance is allowed. Resources can be allowed to be defined as instances of one or more classes. This allows classes to be organized in a hierarchical way. A property can be viewed as an attribute of a class. RDFS properties may inherit from other properties, and domain and range constraints can be applied to focus their use. A domain constraint is used to limit what class or classes a specific property may have, and a range constraint is used to its possible values.

The Web Ontology Language (OWL)

On the top of the RDFS lies the Web Ontology Language (OWL). The OWL is a vocabulary extension of the RDF-concepts for facilitating better machine interoperability according to Cardoso (2007). It allows to make more complex RDF statements, such as cardinality restrictions, type restrictions and complex class expressions (Pauwels, Zhang, & Lee, 2017). Furthermore, it is the formal ontology developed for the Semantic Web to define a more powerful language to describe semantics. On the internet, OWL is the most prominent mark-up language for publishing and sharing data using ontologies (Cardoso, 2007).

4.3.5.2. IfcOWL

As described before, modeling constructs employed in the specification of the IFC-model definition defined in the EXPRESS language are translated into equivalents from the RDF(S) and OWL modeling vocabularies, resulting in an OWL meta-model for buildings (Borrmann, König, Koch, & Beetz, 2018). Most of the initiatives to formalize IFC in an ontology, have been motivated with the aim of providing a semantically rich and platform-independent framework, which can support the integration of software tools, and exchange of data in a knowledge-based system both human-readable and usable by machines (Beetz, 2009; Beetz, Leeuwen van, & Vries de, 2009; Pauwels, Zhang, & Lee, 2017).

Accordingly, Pauwels & Terkaj (2016) indicated based on a number of use cases observed that the focus of the developments does not really lie in the replacement of existing technologies, although in the combination of building information with relevant information in other domains. Considering for instance some of the developments notably in the field of project requirements and building performance checking, Pauwels, et al. (2011) presented a semantic rule-checking environment for building design and construction based on an acoustic performance test case. They briefly shown how the limitations of the current IFC-schema can be overcome when deploying Semantic Web languages (*ifcOWL*). In this case, building information defined as EXPRESS-schema was transformed into an OWL-ontology and a range of rule sets developed in N3Logic. Both are then added to a knowledge-base to perform rule-checks. The logic-based graph structure of the Semantic Web enables the design and implementation of significantly improved rule-checking environments.

Remaining on this field of building performance checking, Moonen (2016) investigated the implications of automated verification of client-specific requirements using rule checking techniques and Semantic Web standards. In this approach, he developed an open and reusable requirements checker system. According to him, the usage of the Semantic Web makes automated verification possibly, whereby knowledge about building elements can be captured in an easy and retraceable way. Focusing on another use case, Stancheva (2017) investigated how the management of structural engineering requirement management can be improved. As a result, she developed a tool using Semantic Web and Linked Data principles, which enables the connection between structural engineering requirements and BIM-model components. As in Pauwels, et al. (2011), she also transformed building information and requirements into an OWL ontology, to possibly link, reuse, and retrieve desired information through SPARQL queries.

To conclude, the ifcOWL ontology is proposed and maintained as a second alternative schema according to Beetz, Leeuwen van, & Vries de (2009). The focus of its formalization lies in the conversion from the ifcEXPRESS-language to OWL-ontology directly (Pauwels & Terkaj, 2016). In this transformation, the network of entities, attributes, simple and selects types, enumerations, collections and relations of the EXPRESS schema, are translated in classes, object properties, data type properties, domains, ranges of the RDF(S) and OWL schema. An example illustrated in Beetz, Leeuwen van, & Vries de (2009) can be seen in Appendix II, wherein the transformation of entities into classes and subclasses are shown. In this example, the entities *IfcElement*, *IfcBuildingElement*, *IfcDoor* are defined as *owl:Class*. Whereby, the hierarchical relations between them are defined as *rdfs:subClassOf and owl:disjointWith*.

2 Practical Framework

5. Practical Framework: The practical view

The practical framework complements the theoretical Framework with additional facts about the subjects of interest associated with the research objectives. From a practical perspective, additional information regarding the subjects of interest, namely: integrated AEC-projects, functional requirements, and OpenBIM-standard IFC is collected based on an interview-approach. For this purpose, the following procedure was arranged: (a) defining interview guidelines; (b) data collection; and (c) data analysis, interpretation, and results. The following sections describe the procedure and the results further.

5.1. Interview guidelines and data collection

The interview questions included in Appendix III aim to complement and give answers to the sub-questions associated with the research objective. Basically, the theoretical framework provides an overview of valuable insights into the characteristics of the subjects under investigation. However, a practical point of view from experts from the field regarding the subjects of interest mentioned before is required. Whereby, their implications in practical contexts can be analyzed and understood. Therefore, the interview questions deal with their characteristics including the possible relationships between them, their impact on each other, or their practical problems and benefits.

Concerning the data collection, information is collected through two interview-sessions. The results are included in Appendix III. Both sessions are convened at a renominated engineering company focusing on one source of information. In this case, the engineering firm Royal HaskoningDHV is approached due to its multidisciplinary point of view. Two experts in the field of project management and BIM are interviewed. Table 5.1 shows their names, and roles in the company.

Interview id	Name	Role	Company
1	Mr. Steven Knol	BIM Manager	Royal HaskoningDHV
2	Mr. Anton Wubben	Senior Project Manager	Royal HaskoningDHV

Table 5.1. Interview-table depicting the experts interviewed

5.2. Interview results

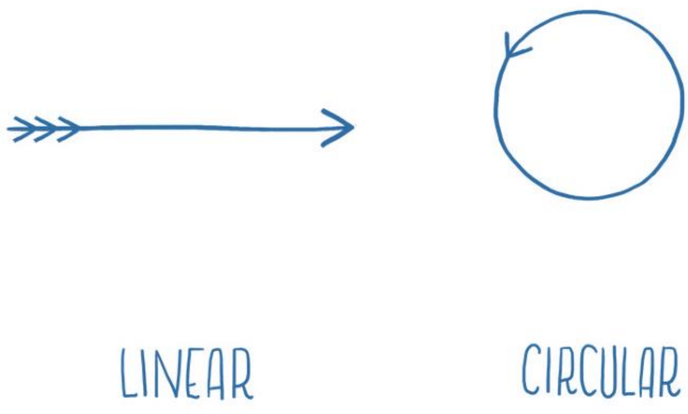
Form a practical view focusing on the Dutch industry, integrated AEC-projects can be defined as an approach in which several clients and stakeholders with specialized knowledge are involved – in other words: multidisciplinary projects. Furthermore, these kinds of projects comprise processes where clients' requirements are identified, captured, and should be met; resulting in high-value and future-proof buildings incorporated with sustainable principles. On the other hand, these projects bring some challenges regarding the way how to manage them properly and consistently, since they are still bounded by traditional design approaches and contracts making integration in general difficult.

In addition, these challenges are also a result of the functional requirements outlining the ideas and necessities of the clients and buildings users. These requirements can be defined as a description denoting the performances that a product or service has to deliver to its users. For this reason, they have to be defined as clear and precise to cope and face project complexity. Besides, depicted in a manner that they are quantifiable, verifiable, adaptable, interrelated, and objective instead of subjective. Unfortunately, sometimes they are not clear and measurable due to the absence of a scientific and/or consistent methodology. For instance, their specification and verification differ per project or is depending on the client, design team, size and complexity of the project. While in most of the projects, these requirements are manually specified and verified in spreadsheets as Microsoft

Excel, in other ones; they are processed in pragmatic requirements tools/databases – as Relatics, Briefbuilder or dRofus – that in most cases are linked to design models.

To conclude, BIM can be seen as a way to collaborate and manage information throughout the building lifecycle. It is not only referred to as a 3D-model or ICT; however it embraces the entire building lifecycle, where information is better defined, faster and smarter. It comprises a single source of information ensuring collaboration between project partners based on clear agreements. Besides, the OpenBIM-standard IFC is an end-product not very dynamic, although, a good product to use for sharing and snapshots. Concerning the consistency of functional requirements, the OpenBIM-standard IFC can be implemented through an integrated process to contribute to their quantifiability and verifiability during the development process. Despite the utilization of native standards as Revit, the Open-BIM-standard IFC is widely accepted due to its lifecycle orientation, transparency and/or possibility to automatically be checked in model checkers as Solibri Model Checker.

3 Model Development



Interconnectedness: Linear to Circular (Acaroglu, 2017)

6. Model development

The theoretical and practical framework as a basis for its development: Model Development comprises the development of the SysML-BIM Integration Model to improve the systematic identification, capturing, and verification of functional requirements in integrated AEC-projects. Accordingly, this section embraces: (a) the model design (see Figure 6.1): indicating how SysML and BIM can be integrated; (b) the model implementation: depicting its implementation through a use case: *IfcDoor,* for a practical application; and (c) the prototype implementation providing an automatic model-to-model transformation.

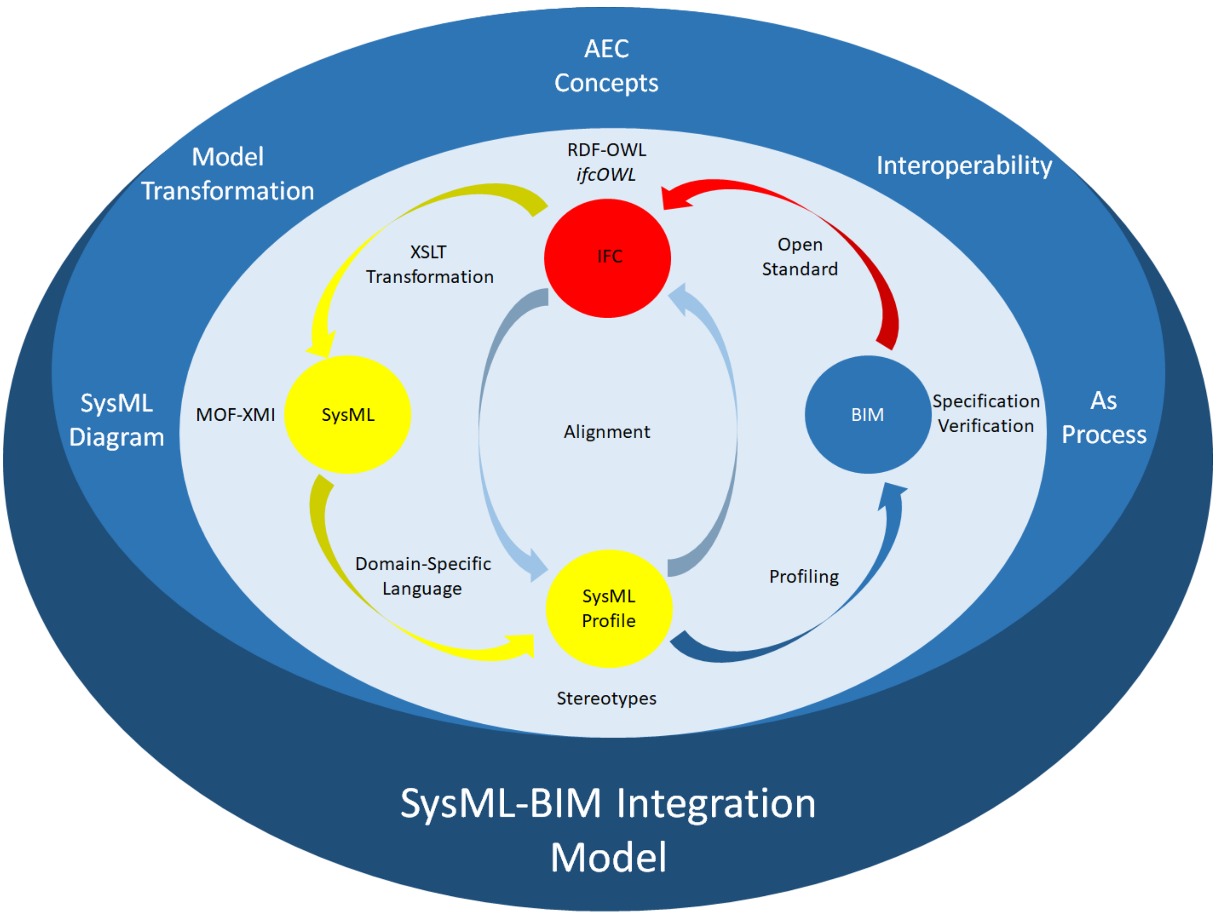


Figure 6.1. Model design: SysML-BIM Integration Model

6.1. Model design

The model design depicts the integration approach in which SysML and BIM can be integrated to provide an integrated and a model-based framework for specifying and verifying complex integrated AEC-projects. The framework considers an integration on a process-oriented level comprising the whole lifecycle of the project, whereby interoperability provides the ability to collaborate together regardless of the domain-specific differences; an explicit definition of AEC-concepts; a model-based collaboration trough model transformations; and communication improvement using diagrams and models. Sections 6.1.1. till 6.1.3. cover the composition of the model through three key parts, namely: Part 1: Interdependency; Part 2: Interoperability; and Part 3: Integration.

6.1.1. Part 1: Interdependency

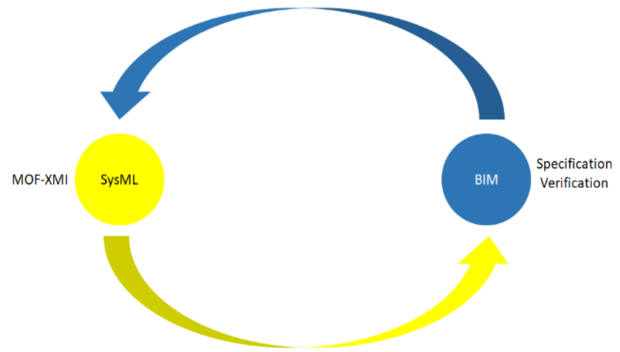


Figure 6.2. SysML and BIM as two interdependent approaches

The first part of the model composition depicts the two key components SysML and BIM as two interdependent approaches that contribute to the consistency of functional requirements during the briefing and design process. First of all, SysML as a visual modeling language provides a comprehensive set of diagrams and constructs for modeling their structure and their interactions with other aspects of the system being developed – in this case the building. In this modeling approach, diagrams as the Use Case Diagram can be implemented to map and model system functionalities based on use cases to depict how users use the system for achieving their goals. Simultaneously, the Requirements Diagram can be used to graphically depicts the hierarchy of requirements defining the system and its components specifications. Complementarily, the Block Definition Diagram can be adopted to model and represent the system structure in terms of its features, hierarchy, and interconnections. In addition, SysML – as graphical notation – improves communication between project participant and reduce likelihood of miscommunication. Furthermore, due to its origin from UML, it can be expressed as MOF-XMI meta-model or other ones ensuring interoperability among computer systems. On the other hand, BIM as process, provides an integrated approach to specify, design, build and operate buildings using digital models. Within this approach and other levels, the creation, management, and communication of information among stakeholders are fundamentally important. Whereby models are created by project participants and used at different times for several use cases. For instance, one of these use cases is the specification and verification of functional requirements during the entire building lifecycle. However, the research is limited to the briefing and design phase.

6.1.2. Part 2: Interoperability

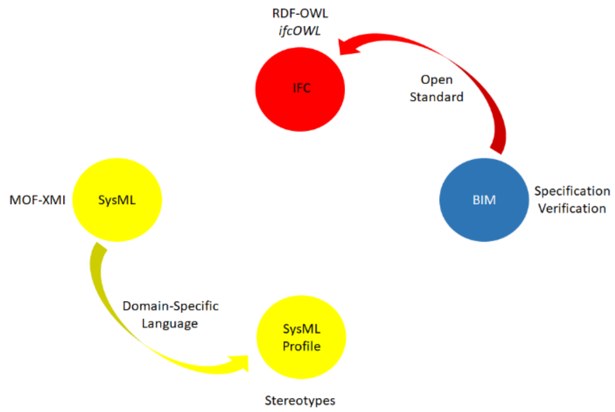


Figure 6.3. The interoperability constructs of SysML and BIM

The second part of the composition illustrates how the two approaches, in this case, SysML and BIM can interoperate with other domain-specific approaches. In the case of SysML, a domain-specific language can be developed by creating SysML profiles, which provide constructs that extend and add new capabilities to the modeling language itself. In accordance, stereotypes can be used to create the profiles and new model elements from existing ones, including detailed attributes suitable for domain-specific applications. In the case of BIM, the Industry Foundation Classes (IFC) can be used as an open interoperability standard for interoperability between several domain approaches. The IFC-standard represents both geometry, process, material performances, fabrication, and other properties. In other words, it provides the whole semantic structure of building models, which is an essential basis for the achievement of the BIG Open BIM goal. In this model, Its language deployment is based on the semantic web technology RDF-OWL for a semantic interoperability. This approach – well-known as ifcOWL – is developed to complement the limitations of the formal and alternative IFC-schema.

6.1.3. Part 3: Integration

The third part of the composition shows how the integration between SysML and BIM is definitely implemented. As shown in Figure 6.4., the missed connections have been included to link the components together in order to complete the circle in a connected circle. Focusing on the interaction between SysML profile and BIM (see Figure 6.4(a)); this link is created through the *Profiling-link*, which is established to overcome the use cases demands from the BIM-process. In this research, the use cases are limited to the specification and verification of functional requirements. SysML profile and stereotypes – using the IFC-concepts as a basis – can be used to define and create new concepts to perform these use cases in the SysML-environment. At the other side, the interaction between IFC and SysML (see Figure 6.4(b)) is created by an *XSLT Transformation-link*. This link can be executed through an ''RDF(S)-MOF bridge'' as defined by Gaševic, Djuric, & Devedžic (2006), which allows a transformation of RDF(S) concepts to MOF concepts and vice versa according to them.

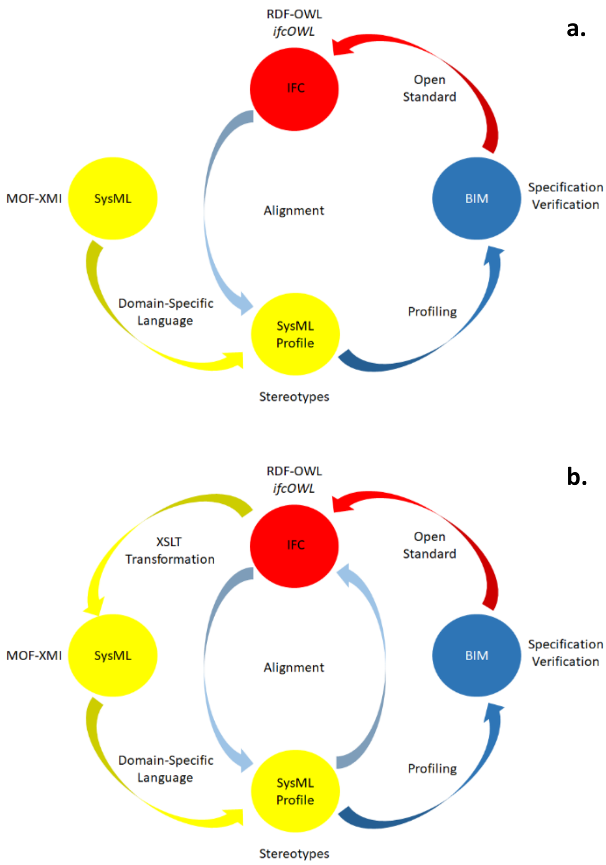


Figure 6.4. The integration parts: Profiling-Link (a); XSLT Transformation-Link (b)

This interaction – using the SysML profile as a basis – is crucial for a model-to-model collaboration maintaining a consistent relationship between data from IFC to SysML.

6.2. Model implementation

As mentioned before, the implementation of the model is executed through a specific use case for a practical application and evaluation. This use case, named IfcDoor, focuses on one specific object from the AEC-process: the door-object. The door as one of the fundamental elements in buildings can be defined as a building element that is predominately used to provide controlled access for people and goods according to BuildingSMART (1996-2006). The implementation covers the main parts of the integration as shown in Figure 6.4 and described in section 6.1.3., namely: (a) the link between SysML Profile and BIM through profiling with IFC as input; and (b) the link between IFC and SysML through an XSLT Transformation with SysML Profile as input.

6.2.1. SysML profiling

Most of the current SysML modeling tools provide the capability to create SysML profiles for a specific domain. In this research, Cameo Systems Modeler (Academic Version) is embraced as a modeling tool. The first step to the integration is possible by profiling concepts from the BIM-process. Considering the door-object as a use case, a Profile Diagram named Building Element is used to create the door class: IfcDoor (see Figure 6.5). In this case, the stereotype block is used and extended through a ''generalization relationship'' for the development of the stereotype IfcDoor – as stated by Friendenthal, Moore, & Steiner (2015). In addition, the stereotype includes compartments, which aims to accommodate the characteristics of the class (IfcDoor). In this example, the compartment includes the properties that are adopted from the IFC schema specification defined by BuildingSMART (1996-2006) (See Appendix IV) – the alignment with IFC according the model design.

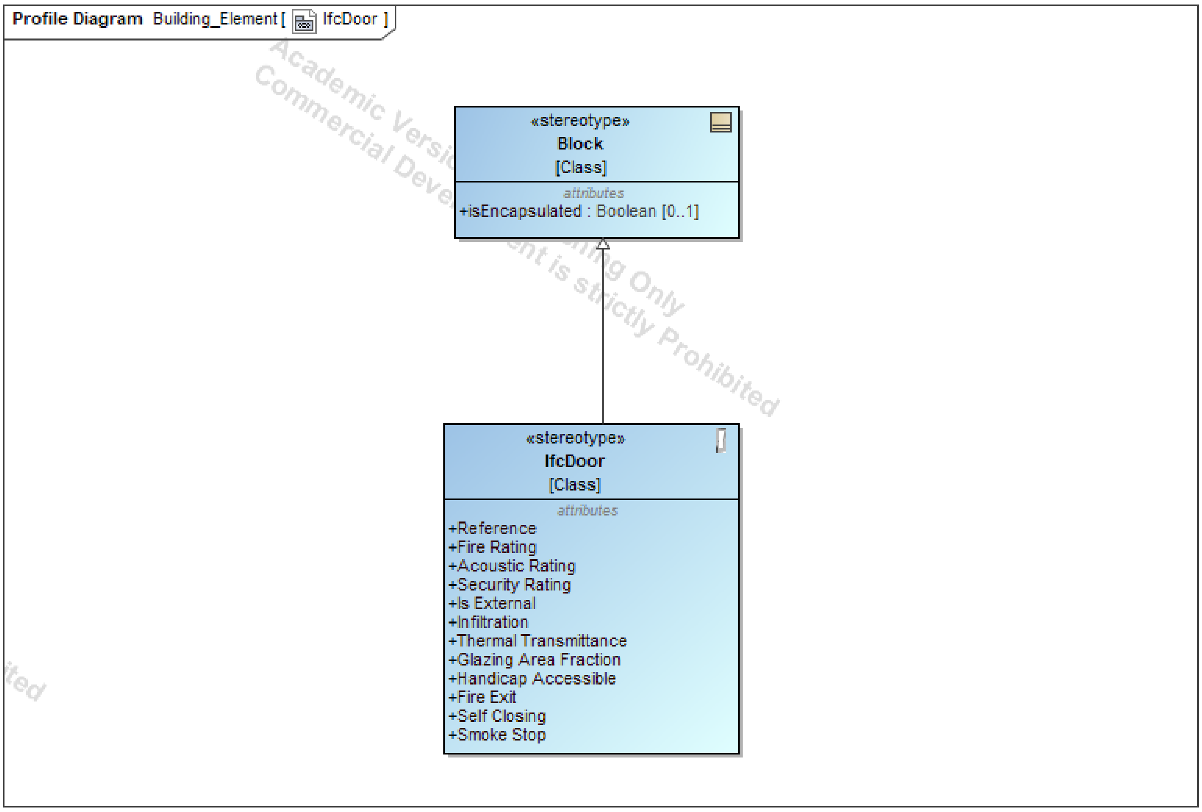


Figure 6.5. The Profile Diagram defining the stereotype IfcDoor

As a result, Figure 6.6 depicts a BDD named Building Element showing the door-object captured in a stereotype <<IfcDoor>> defined earlier. The upper stereotype shows the class IfcDoor, whereby the below is following by a generalization relationship illustrating the IfcDoor_id and its characteristics as the potential instances of this class. Focusing on this initial link to the integration process, this part illustrates the ability to define and model domain-specific objects from the BIM process, to perform use cases as specification and verification of functional requirements in BIM-processes.

6.2.2. Model transformation

The second part of the integration process is practicable based on a model-transformation approach as described in section 6.1.3. and shown Figure 6.4(b). Accordingly, the transformation approach depicted in Figure 6.7 is developed, which is composed based on the basics concepts of model transformation proposed by Czarnecki & Helsen (2006) (see Figure 3.11). As described earlier, model transformation is highly suitable when two sides of the exchange are different and transformation is necessary to support data exchange, which is the case in this research. As shown in Figure 3.11 and remodel in Figure 6.7, the transformation from IFC to SysML can be seen as a *Transformation Engine* reading a source model (IFC-model) and writing it into a target model (SysML BDD). The source model used in this test case is the well-known Duplex House. The house is expressed conform to the proposed OWL representation for the IFC-schema: ifcOWL.

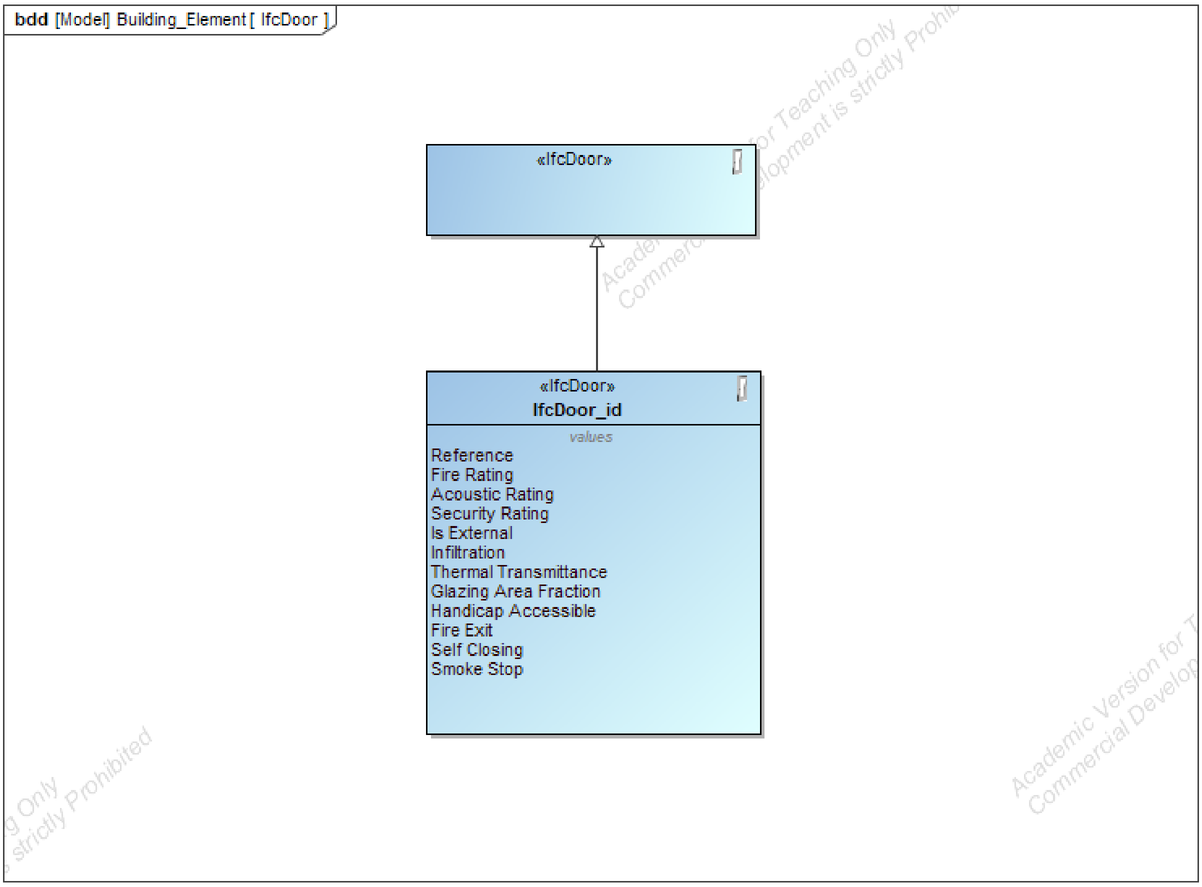


Figure 6.6. BDD showing the stereotype <<IfcDoor>> and its potential instances (IfcDoor_id)

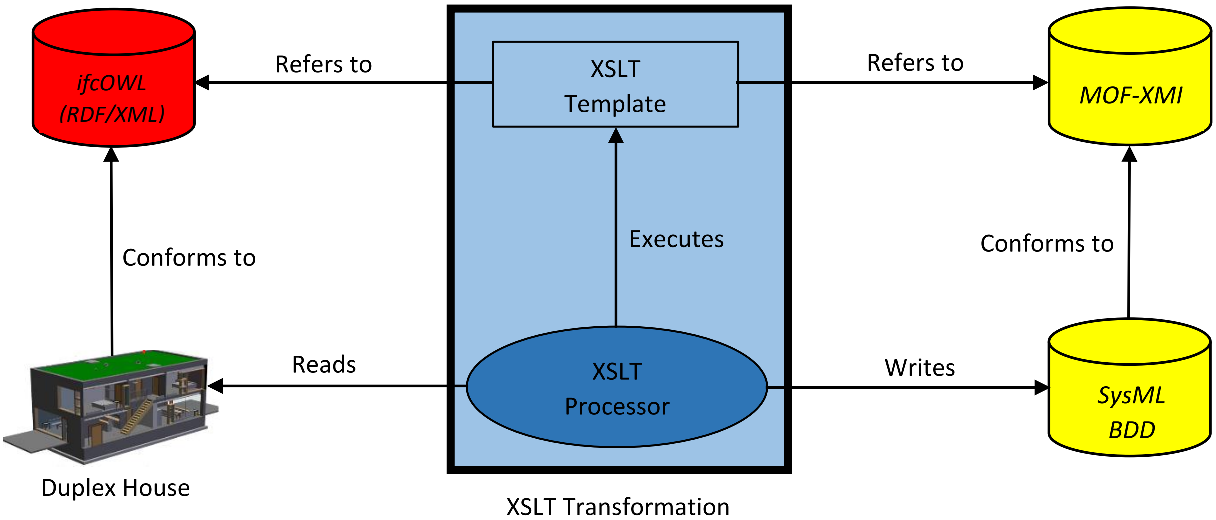


Figure 6.7. The model-transformation approach

The ifcOWL representation is created and then formatted into a TURTLE (.ttl) format, based on an *IFC to RDF-Desktop* tool (Appendix I) advised by mister Pieter Pauwel. On the other side, the target model is the SysML BDD, which can be expressed into a metamodel standard: MOF-XMI. Most of the SysML tools are provided with this interoperable standard. Regarding the transformation, it is based on an XSLT transformation, in which the *Tranformation Engine* is represented by an XSLT processor, and the *Transformation Definition* is captured in an XSLT template.

Finally, due to the extent of the schema and to meet the use case, this research covers the transformation of a random ifcDoor-object including two of its properties. The following sections comprise the extraction and creation of the ifcDoor graph, and its conversion to a SysML BDD through an XSLT transformation.

6.2.2.1. The ifcDoor graph

As described, the conversion only focuses on one random ifcDoor-object with two of its properties. Therefore, the first step was the extraction of the graphical relation in which the ifcDoor object is related to its properties. For this purpose, SPARQL is used to query this graph, whereafter a new partial graph could be extracted and created. As stated by Beetz, Leeuwen, & Vries de (2007), SPARQL queries are defined using the features “CONSTRUCT”, “WHERE”, “LIMIT 2” (See Appendix IV) to extract and create a new graph from the basic one. As illustrated in Figure 6.8, the SPARQL tool Apache Jena Fuseki is used to define the queries and create a partial graph that is shown there.

As can been seen in this figure, the instance IfcDoor_21821 which is a type of the IfcDoor class, is extracted in conjunction with its two properties named Reference and Fire Rating. The properties are linked to the door-object through an inheritance relationship, which makes the graph extremely large and consequently making the transformation to MOF-XMI highly complicated. For this reason, this research suggested another approach than the one applicated by Beetz, Leeuwen, & Vries de (2007), which establishes a development of a more simple and understandable graph for easy navigation through queries and transformation.

The basis for the development lies on the input from SysML profile to IFC as shown in Figure 6.4, in which the SysML profile represents the door-object and in a direct compartmentally association its properties as shown in Figure 6.9(c). Fundamentally, it can be seen as a framework in which new graphs can be created for easy transformations for instance between IFC and SysML. Therefore, it is adopted in order to create the graph shown in Figure6.9(d), whereby the properties are directly connected to the IfcDoor class making the graph more compacter, smaller and consistently suitable for integration than the one shown in Figure 6.8(c) Similarly to the previous approach, SPARQL queries are also defined in order to extract the door-object and create the new graph. For this purpose, features as “CONSTRUCT”, “WHERE”, and “LIMIT 2” are used.

6.2.2.2. The XSLT transformation

XSLT is used to transform the ifcDoor graph expressed in a TURTLE (.ttl) format to the SysML BDD. However, in advance of the transformation, the graph is expressed into an RDF/XML format. Which expresses the RDF-graph as an XML document making it more compatible with the MOF-XMI standard, since XMI is part of the XML-based standard for a meta-meta model, meta-model, and model sharing. In addition, making XSLT and its template and processor perfectly suitable for the transformation.

Focusing on the transformation approach depicted in Figure 6.7, the ifcDoor graph represents a partial graph from the entirety that defines the Duplex House. The graph expressed as RDF/XML is shown in Appendix IV describing the key components of the door-object and its characteristics, namely: its properties. Continuing with the XSLT-approach, the XSLT-template defines the rules about how the transformation will execute. These set of rules defined in an XSL-syntax as shown in Appendix IV, comprise firstly marked in red, the link with the RDF/XML file as input for data extraction and conversion; secondly marked in green, the XMI-namespace defining the output-file as an XMI-standard; thirdly marked as bold, the key components and properties of the MOF-structure in which the input-file (RDF/XML) has to be transformed and capture in an output-file (MOF-XMI); and finally marked in blue, the queries defined through the XPATH-language to extract the door-object and its properties from the input-file, and included in the output-file.

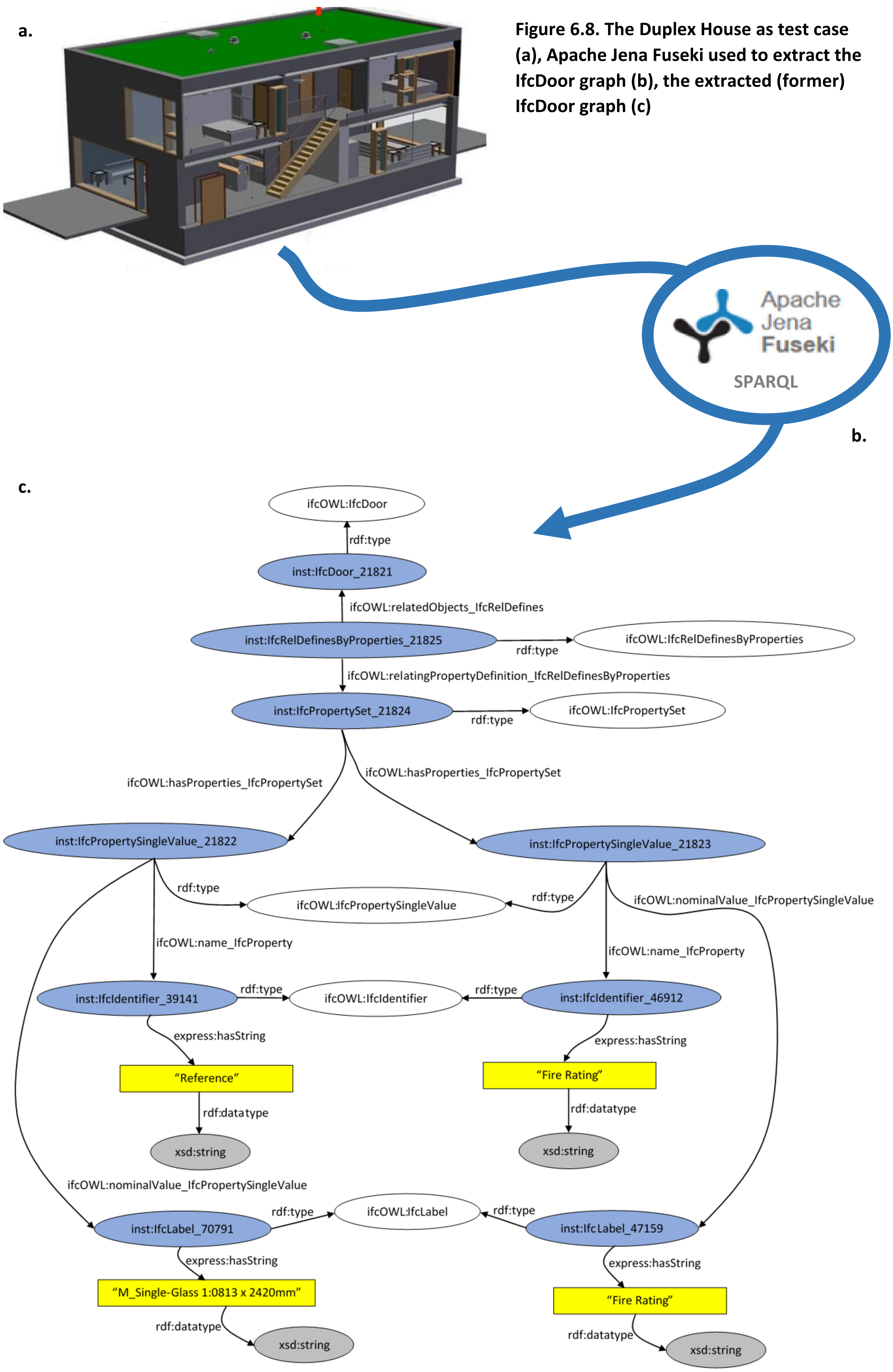


Figure 6.8. The Duplex House as test case (a), Apache Jena Fuseki used to extract the IfcDoor graph (b), the extracted (former) IfcDoor graph (c)

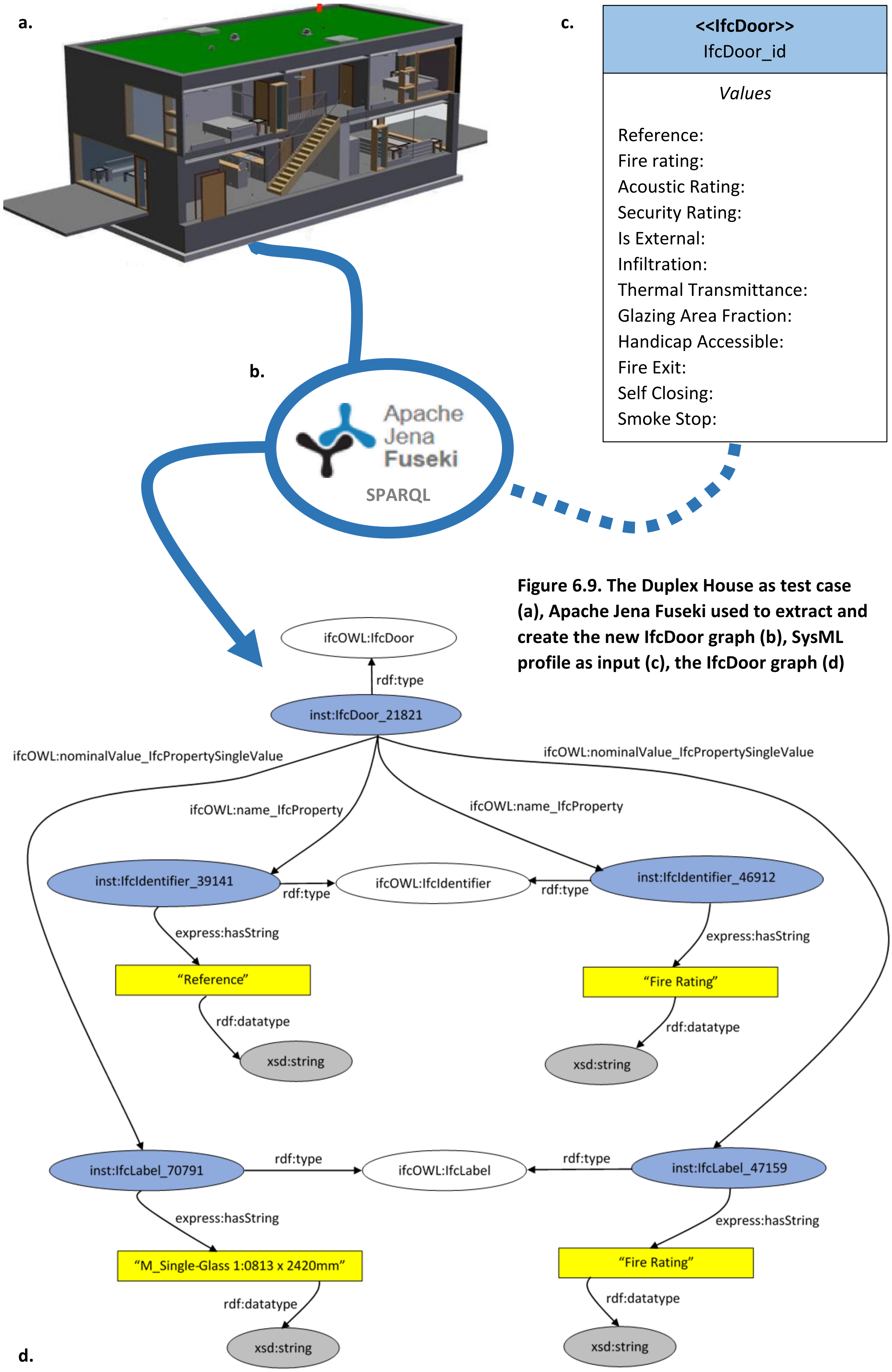


Figure 6.9. The Duplex House as test case (a), Apache Jena Fuseki used to extract and create the new IfcDoor graph (b), SysML profile as input (c), the IfcDoor graph (d)

Subsequently, the transformation can be executed. For this operation, the XSLT processor is used. As defined in Figure 6.7, the processor executes the transformation definition defined in the template using the RDF/XML file as input, and transformed into the MOF-XMI output-file as illustrated in Appendix IV. As can be seen in the appendix, the output-file is structured through a MOF-XMI syntax comprising the door-object and its properties. Furthermore, this file can be imported into each SysML tools that read MOF-XMI files. As an example, the output-file is imported in the Cameo Systems Modeler, where prior the profile was developed for the door-object. The result is depicted in Figure 6.10 showing a block diagram comprising the object IfcDoor_21821 and its two properties as defined earlier in section 6.2.2.1. Finally, it can be applicated to a specific-domain use case, in this case to the stereotype <<IfcDoor>> as shown in 6.11 and 6.12.

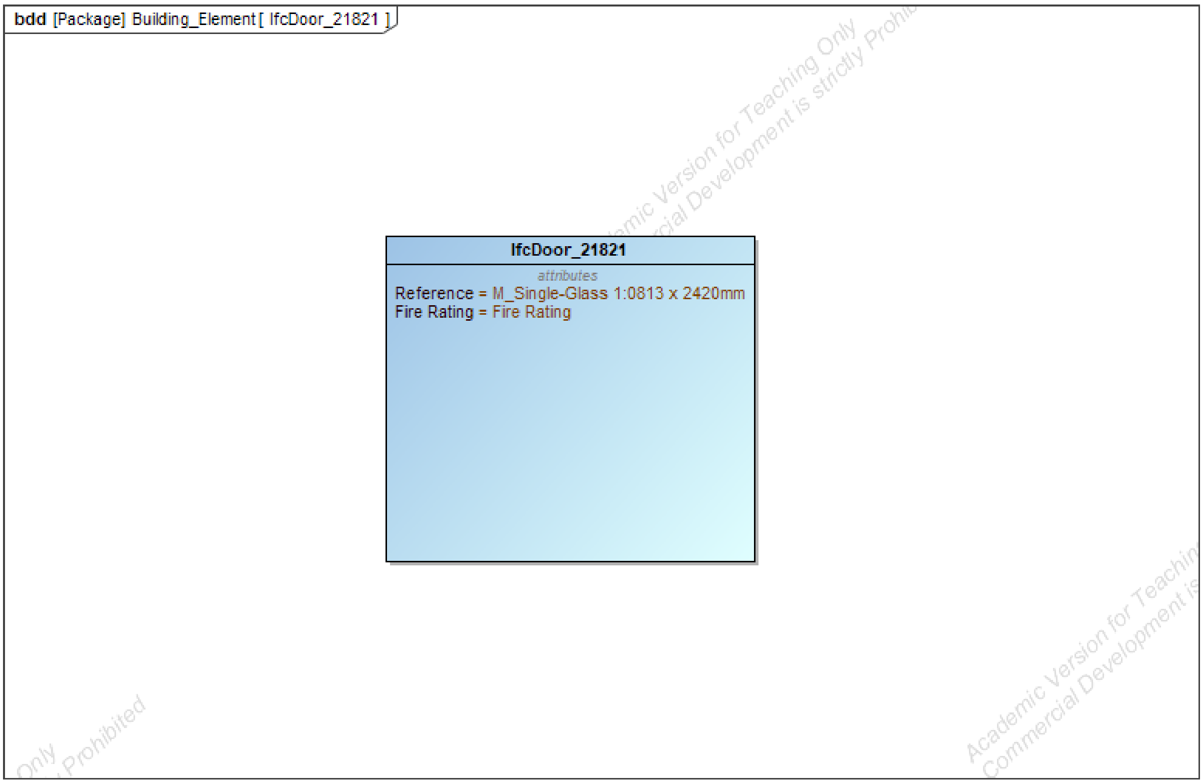


Figure 6.10. The BDD showing the imported IfcDoor_21821

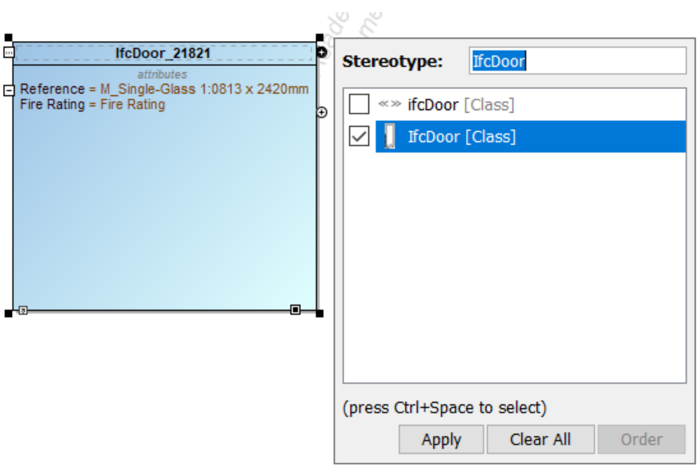


Figure 6.11. Applying IfcDoor_21821 to a specific stereotype; <<IfcDoor>>

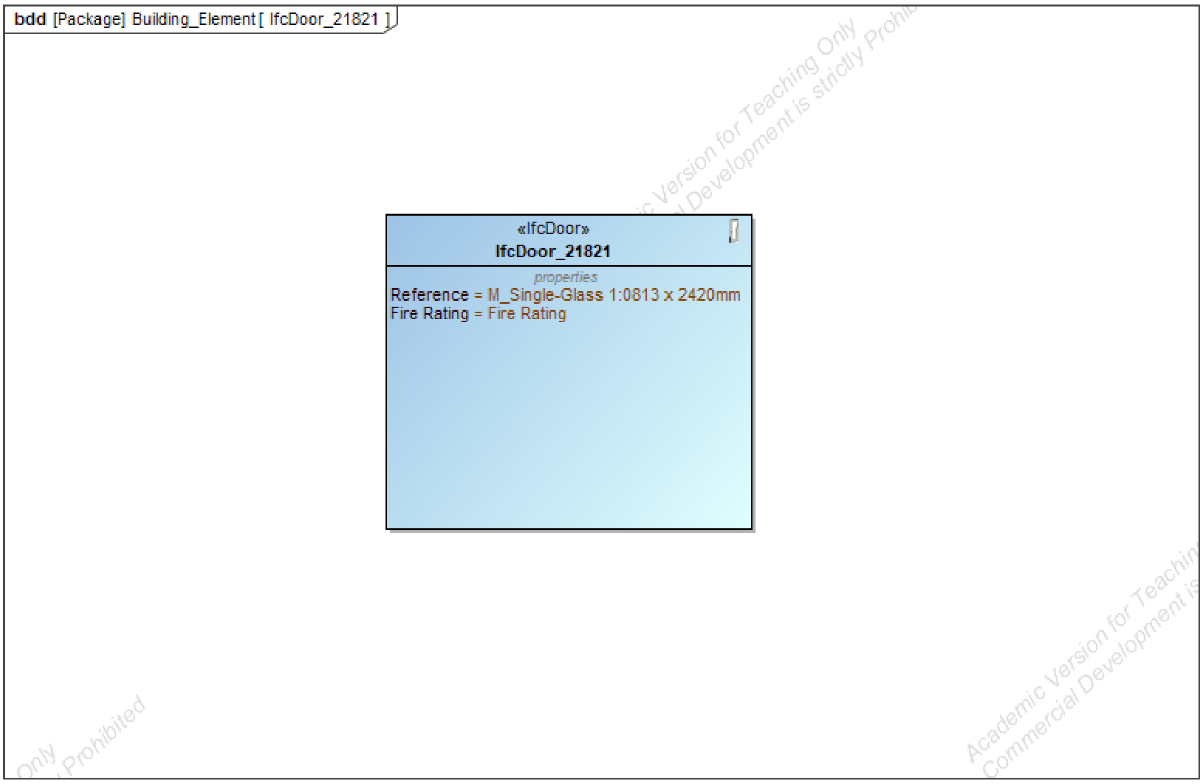


Figure 6.12. The BDD showing the imported IfcDoor_21821 as the stereotype <<IfcDoor>>

6.3. Prototype implementation

For an automated transformation, the IFC-to-SysML Transformation Tool is developed. The development comprises three main parts, namely: Part 1 Graphical User Interface (GUI) (see Figure 6.13); Part 2 extraction IfcDoor graph including its properties, and creation of a new IfcDoor graph expressed in RDF/XML **[IfcDoor]**; and Part 3 transformation of the IfcDoor graph into MOF-XMI **[Transform]**. Appendix V illustrates the Java-Code defined for this development.

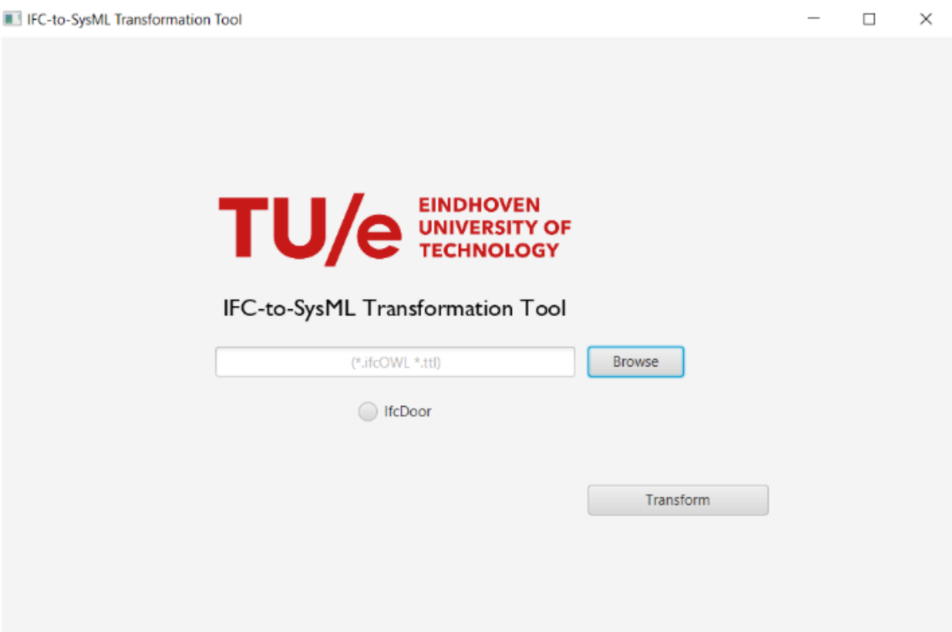


Figure 6.13. The Graphical User Interface (GUI)

4 Model Validation

7. Model validation

This section covers the validation process of the SysML-BIM Integration Model developed in section 6, which aims to evaluate the practical implications of the model within a real-world context: a case study. As defined before, the model establishes an integration between SysML and BIM in order to improve the specification and verification process of functional requirements in integrated AEC-projects. As a real-world example, the new in developing residence building project Sixty5 in Eindhoven is adopted as case study. As in most of the projects, the development process initiates with the creation of the well-known brief by the clients describing what are their needs and the users' expectations for the new building. From this brief, a functional description of a component is derived and used as input in the creation of the system model for this case study. Despite the model encompasses a whole lifecycle-approach, the validation process comprises only the briefing and design phase. The following sections denote the practical implementation of the model and its evaluation further in detail.

7.1. Case Study: Project Sixty5 in Eindhoven

The residence building project Sixty5 is another development regarding the revitalization of the railway zone of Strijp S in Eindhoven according to Diederendirrix (2019). The project is commissioned by Spoorzone BV and consisting of 105 luxury apartments varying from 65m^2 to 217m^2. Besides the apartments, the building also comprises a commercial space (Diederendirrix, 2019). It will be built by the constructor Stam + De Koning Bouw BV. The functional description derived and/or defined for the case study is that the luxury apartments have to provide comfort and safe living environment to its habitants. The focus is finally on one apartment of the residence project: *Apartment 4-B*.

a **b**

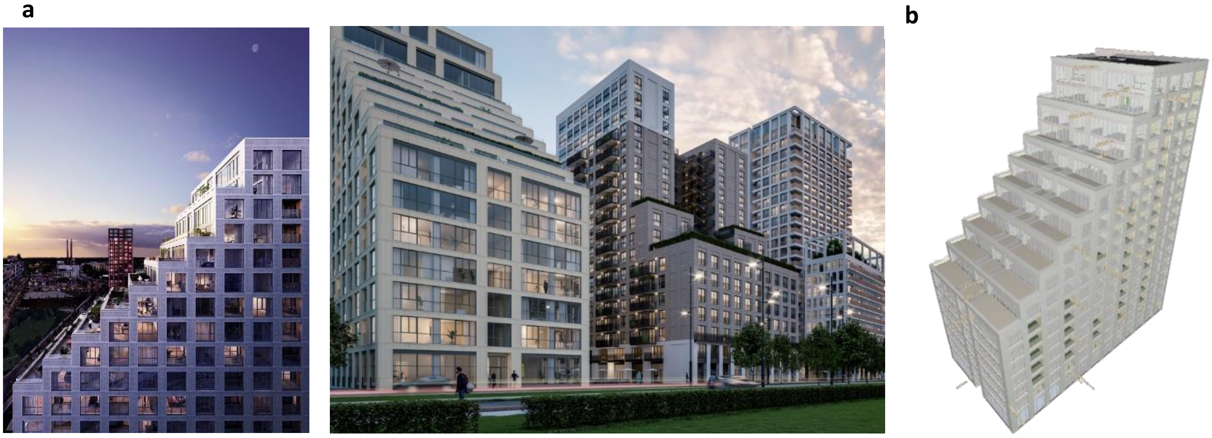


Figure 7.1. The Residence building project Sixty5 (Diederendirrix, 2019) **(a); The BIM-model (b)**

7.2. The process model

The process model depicted in Figure 7.2 gives an overview of how the developed SysML-BIM integration model can be used in the context of the residential building project Sixty5. In a collaborative approach, a team consisted of system engineers and designers collaborate together through a random share point server during the briefing and design phase of the project. As basis for this collaboration is the brief forming the starting point in the development of the system model for Apartment 4-B (SM.010). This model, which its development is assigned to the system engineer team, comprises the system specification of Apartment 4-B replacing the brief explicitly. After its creation process, the system model is then distributed through the share point server to the design team forming the basis to proceed with the Design Development (DM.010) by the team. The design is based on the integrated approach of BIM. The integrated BIM-model as result from this approach is then shared with the system engineer team through the server to initiate Design Verification (SM.020).

In this stage, the design is checked to ensure that it meets with the system model specification. At a positive result, the construction phase can start. The following paragraphs cover the main components of the process model briefly.

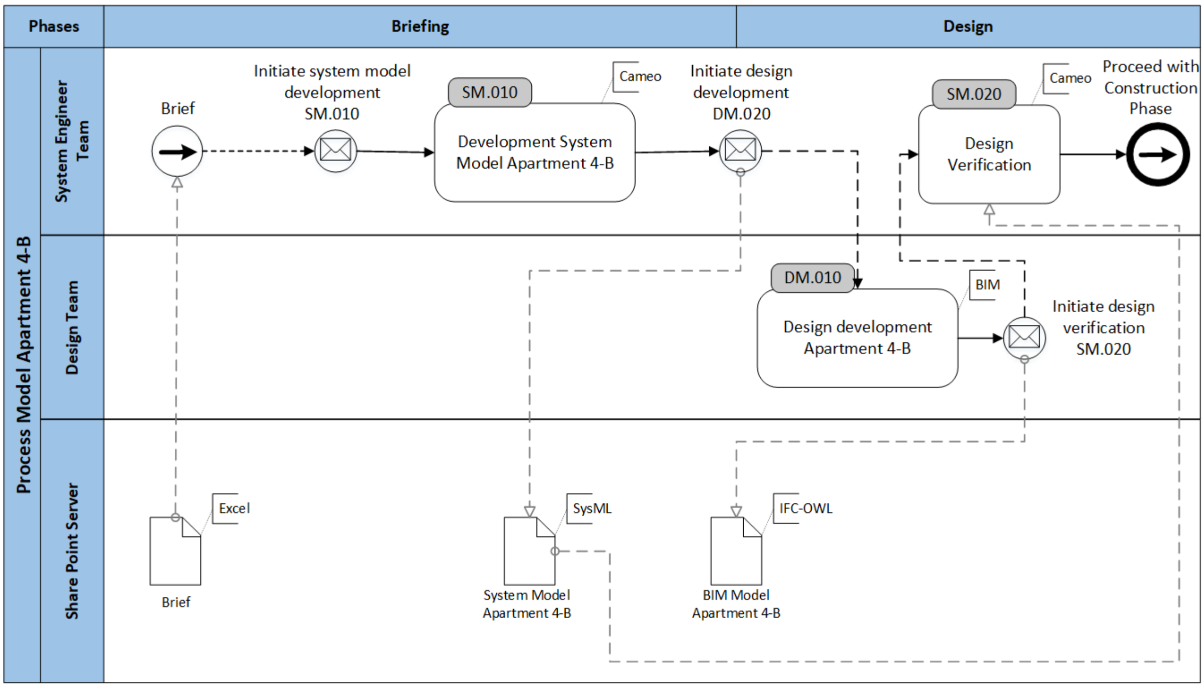


Figure 7.2. The process model

7.2.1. The System Model SM.010

As described earlier, the system model is the primary artifact of the MBSE-approach to enable the development of the system of interest in meeting its overall objectives. The development of this model is based on three pillars namely, (a) the modeling language, which in this case SysML is adopted as language; (b) the modeling method, whereby the Parametric System Modeling (PSM) methodology (see Appendix VI) from Geyer (2012) is embraced, however, only the Use Case Diagram (uc), Requirement Diagram (req), and Block Definition Diagram (bdd) are selected to meet the research objective; and (c) the modeling tool, whereby the SysML tool Cameo Systems Modeler Academic Version is chosen as tool.

Considering the SysML-BIM Integration Model described in section 6, the composition process of the system model initiates with defining a domain-specific language based on BIM through profiling. As shown in Appendix VI, the profile diagram provides this function of definition defining the BIM-IFC concepts based on blocks, to perform use cases as specification and verification of functional requirements. From this basis, the system model development – containing the Use Case Diagram: system functionalities; Requirement Diagram: system requirement; and Block Definition Diagram: system structure – is started (see Figure 7.3-7.4-7.5-7.8).

7.2.1.1. Use Case Diagram: system functionalities

The Use Case diagram aims to provide an illustration of the system under development and its interactions with use cases and actors. In this case, the diagram represents one of the apartments from the residence building project Sixty5 as a block (Figure 7.4). Within the block, the use cases are shown as ovals including their names inside them depicting the functionalities that the system has to provide to its inhabitants. This prerequisite is according to the functional requirement defined earlier, which indicates that the apartment has to provide a comfort and safe living environment to its habitants.

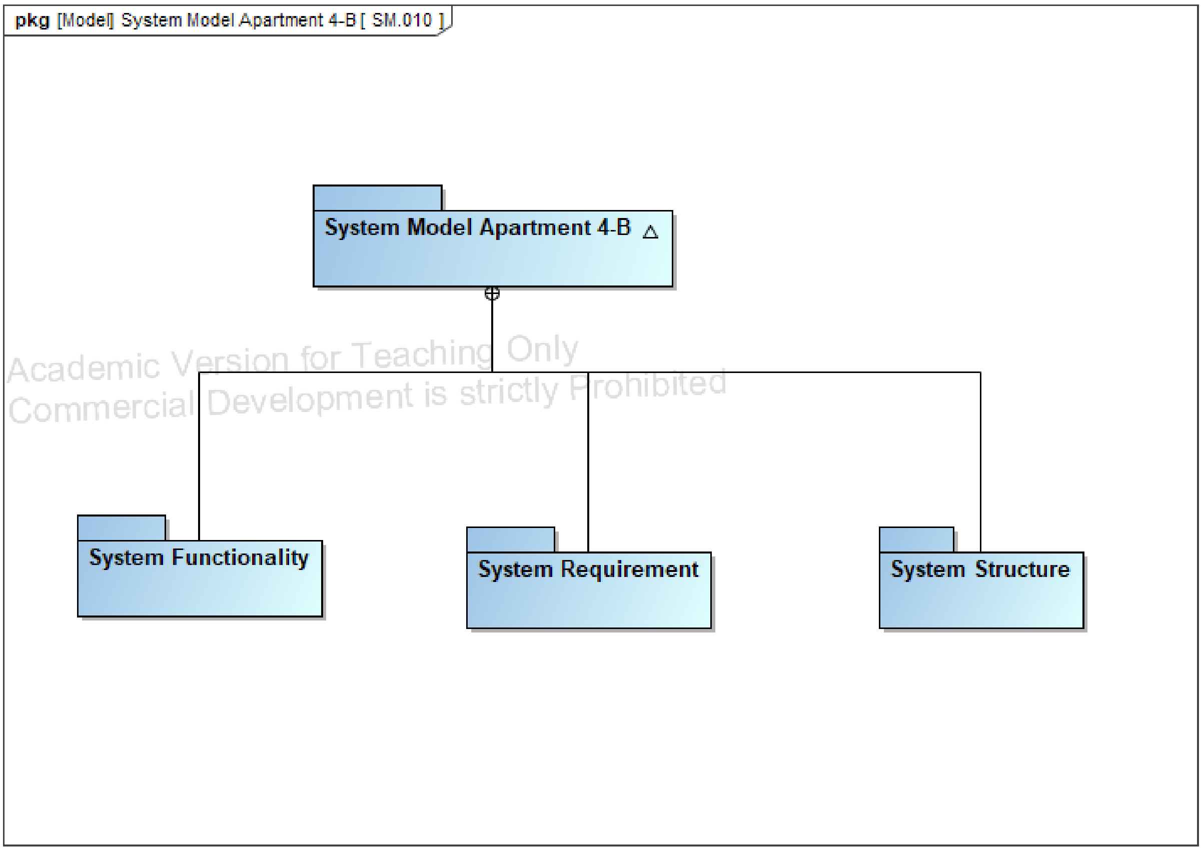


Figure 7.3. The System Model containing the specifications of Apartment 4-B

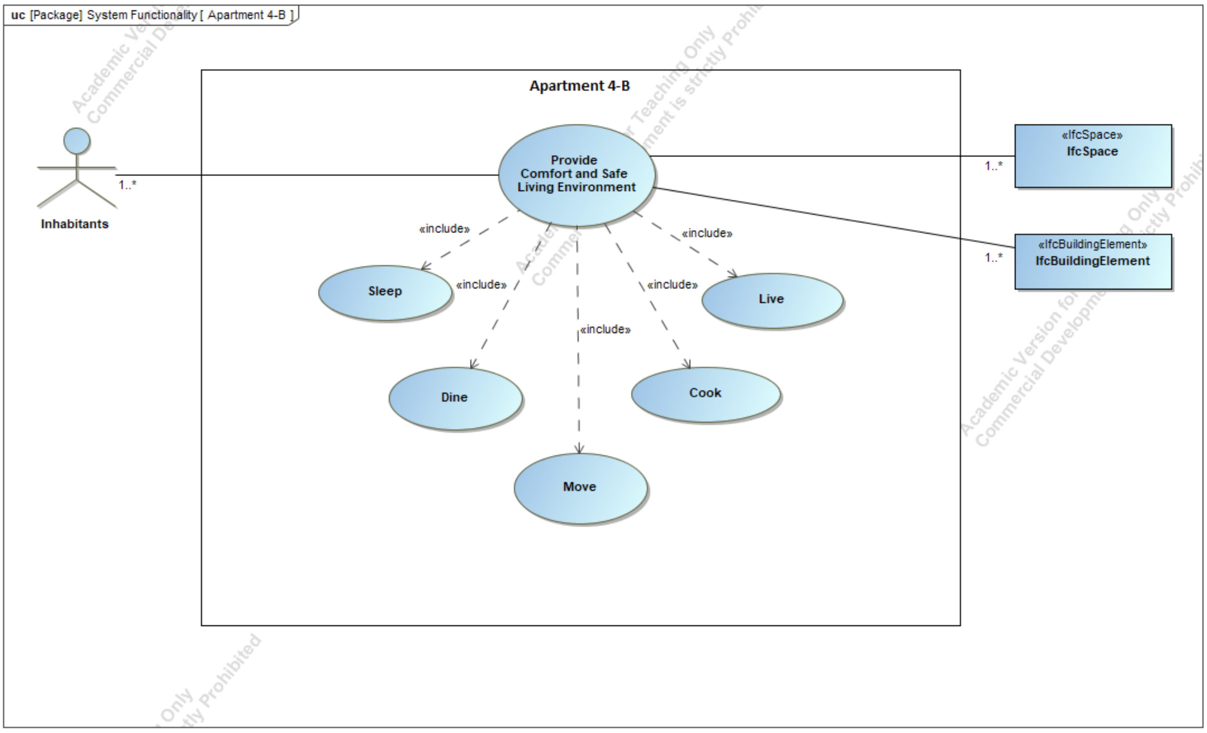


Figure 7.4. The Use Case Diagram representing the functionalities of Apartment 4-B

Besides, this condition includes other functionalities that the system also has to provide in order to facilitate the performing of the base condition. Through an “inclusion relationship”, the use cases Sleep, Dine, Live, Cook, and Move are connected to this base condition.

Outside the system boundary, the users of the system are situated and they are described by actors, which may represent the role of a human, an organization or any external system participating in the use of the system. In this example, the actors are referring to the inhabitants and building components interacting with the system through standard associations connected to the use cases. The association between the habitants and use cases shows the relevance of the functionalities to the habitants in order to ensure a cohesive interaction with the system. The multiplicity [1..*] referring to “one to many”, denotes the number of actors involved in the use cases. On the other hand, the association between the building components and use cases illustrates the key role that the components play in the realization of the system functionalities. The multiplicity [1..*] referring to “one to many”, denotes the number of components involved in the use cases.

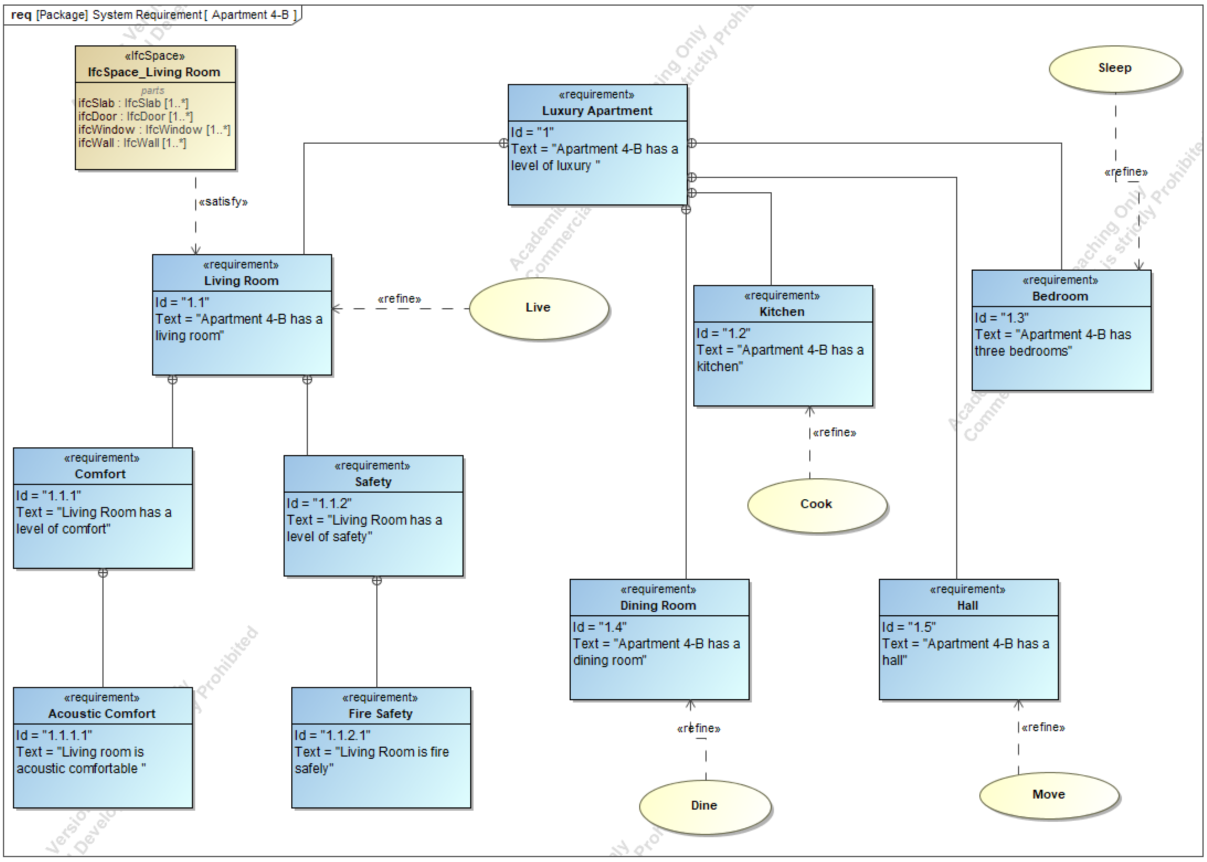


Figure 7.5. The Requirement Diagram representing the system requirement of Apartment 4-B

7.2.1.2. Requirement Diagram: system requirement

The Requirements Diagram graphically depicts the hierarchy of requirements defining the system or component specification as shown in Figure 7.5. After requirements are defined, they can be captured in the SysML requirement stereotype <<requirement>> and related to other requirements or elements. As can be seen in the figure, each requirement includes predefined properties for a unique identifier, and for a text string indicating a capability or condition that must be satisfied by a function that a system performs, or a performance condition a system must achieve. Based on a containment relationship the requirements are correspondingly related to each other forming a hierarchy of requirements specifying the system at different abstractions.

At the highest level of the hierarchy, the requirement Luxury Apartment is situated prescribing that "Apartment 4-B has a level of luxury" referring to the system under consideration as captured in the Use Case Diagram (see Figure 7.4).

At second level of the hierarchy, the requirements Living Room, Dining Room, Kitchen, Bedroom and Hall are following indicating the components desired in Apartment 4-B. Besides, they are also clarifying the use cases specified in the Use Case Diagram by a "refine relationship". Focusing on the Living Room as subject of interest, this requirement is related to the requirements comfort and safety referring to the required comfort and safety living environment that the apartment has to provide to its inhabitants. The target lies on a Living Room that is acoustic comfortable and fire safely. In addition, that same requirement is related to the subsystem Living Room, which is coming from the Block Definition Diagram that represents the structure of Apartment 4-B (see Figure 7.8). Through a "satisfy relationship", the subsystem satisfies the requirement in meeting with the capability or condition that the requirement requires. Lastly, the results can be captured in a satisfy requirement matrix and requirement containment map for a better understanding and an overview of the requirements (see Figure 7.6 and 7.7).

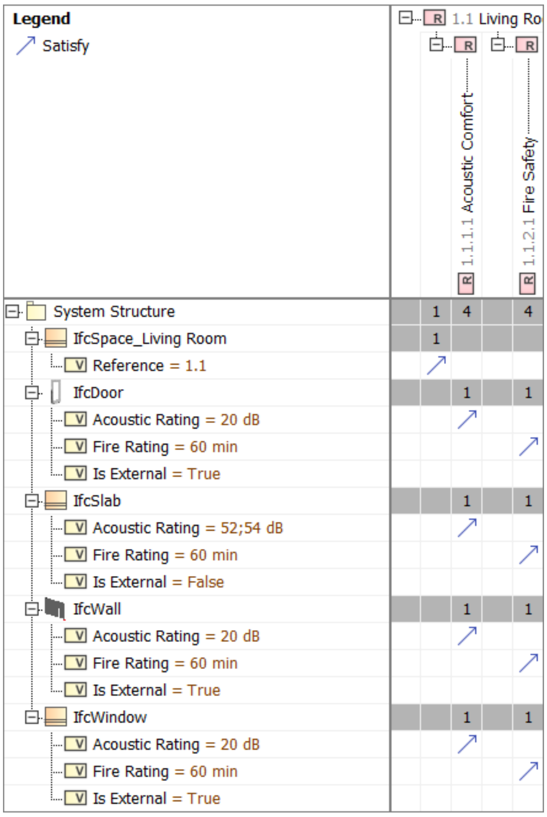


Figure 7.6. Satisfy requirement matrix (SRM)

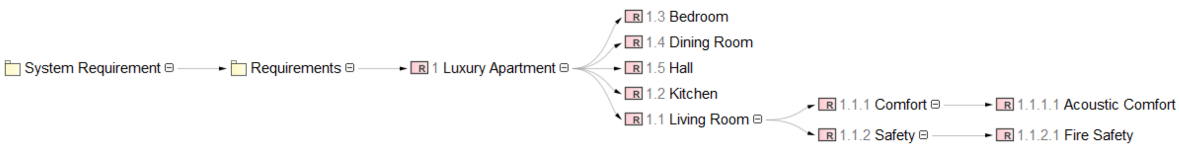


Figure 7.7. Requirement containment map (RCM)

7.2.1.3. Block Definition Diagram: system structure

The Block Definition Diagram (BDD) represents the system structure in terms of its features, hierarchy, and interconnection. Through blocks as a fundamental modular unit, several systems, components, and component interconnections can be defined and represented. In the case of a BDD for the Living Room as shown in Figure 7.8, it is represented as a subsystem defining the system Apartment 4-B through a composition of several components. These components are captured in blocks including their attributes, and are interacting through specific associations and multiplicities (See Appendix I). The following ones are forming part of that whole: IfcSpace_Living Room, IfcDoor, IfcWindow, IfcWall, and IfcSlab.

The IfcSpace_Living Room is associated with IfcDoor, IfcSlab, IfcWindow, IfcWall based on a composite association defining the importance of a whole-part relation to perform as a subsystem in this situation. Which means that at an absence of these relations can lead to unsatisfied requirements. Furthermore, each block has attributes indicating the performance that the blocks have to provide in order to connectedly meet the requirements regarding acoustic comfort and fire safety as defined in section 7.2.1.2. The number of elements that are possibly related with the IfcSpace_Living Room is incorporated in the multiplicity [1..*], which indicates a relation with one to many elements.

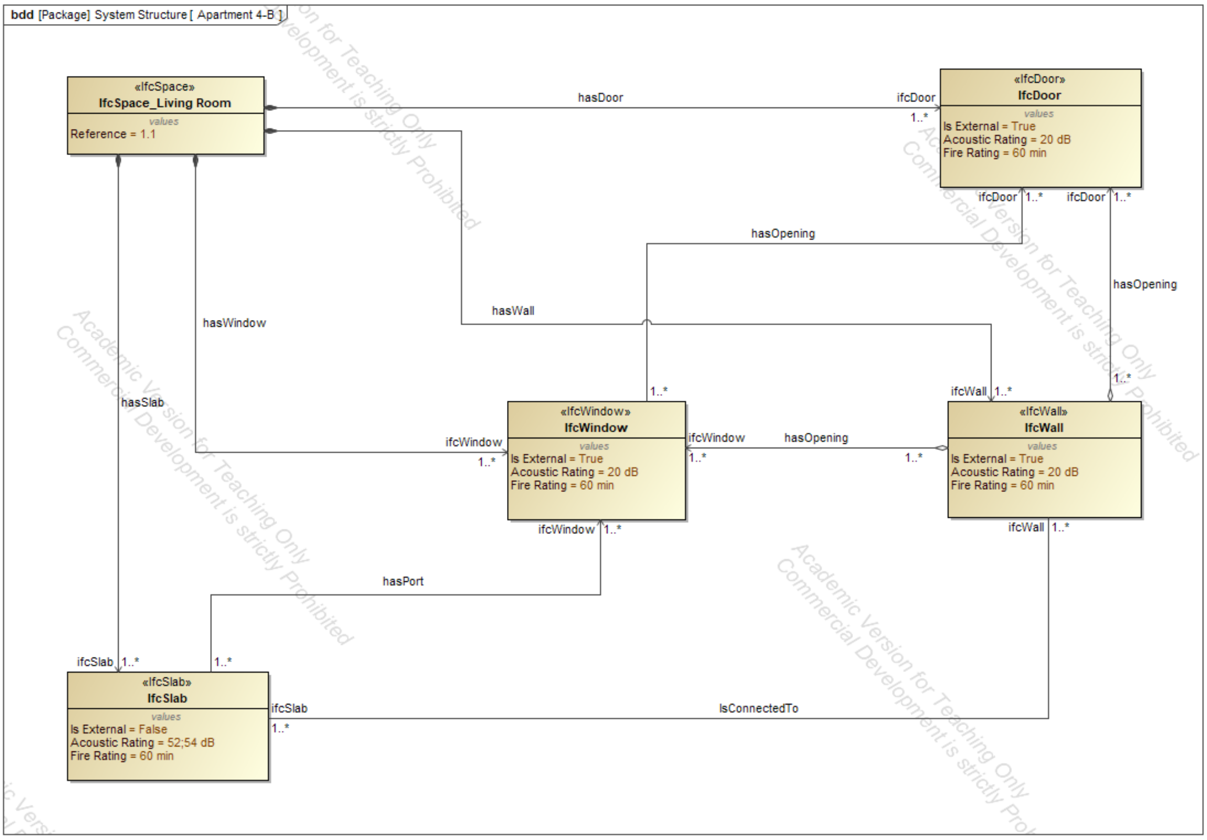


Figure 7.8. The Block Definition Diagram representing the subsystem structure of Apartment 4-B

Complementary, the interaction between IfcWall and IfcDoor is realized through an aggregation association defining IfcDoor as part of the IfcWall without making the IfcDoor part of the IfcWall composition. This same definition is applicable to the relation between IfcWall and IfcWindow. Regarding the relation between IfcSlab and IfcWall is based on a simple association which represents a bidirectional access to meet some purposes across the connection. In this interaction, the bidirectional relation between IfcSlab and IfcWall can be explained by the fact that a wall can be connected to a slab, or otherwise the slab to a wall. Lastly, the interaction between IfcSlab with IfcWindow, and IfcWindow with IfcDoor are based on a navigable association. This association is representing by a unidirectional access comprising a connection with an open arrowhead referencing a property on only one end. In the case of IfcSlab, this provides ports where IfcWindow can be installed, whereby at the same time the latter has openings wherein IfcDoor can be placed. As at the composite association, the same multiplicities [1..*] are adopted in these connections.

7.2.2. The Design Model DM.010

The design model is created based on the principles of BIM. For this case study, the constructor Stam + De Koning Bouw BV provided the research with the BIM-model of project Sixty5 (see Figure 7.1(b)). Figure 7.9 shows a fragment of its façade in an IFC-viewer showing for instance the properties of door 4.67 (in the viewer as green marked). These properties in this model are important to verify if the model/design meets the specifications defined in the system model. In order to perform this check, the BIM-model can be transformed to SysML (MOF-file) through the prototype developed in section 6.3, and then imported into the SysML environment to enable Design Verification (SM.020) as described by the process model.

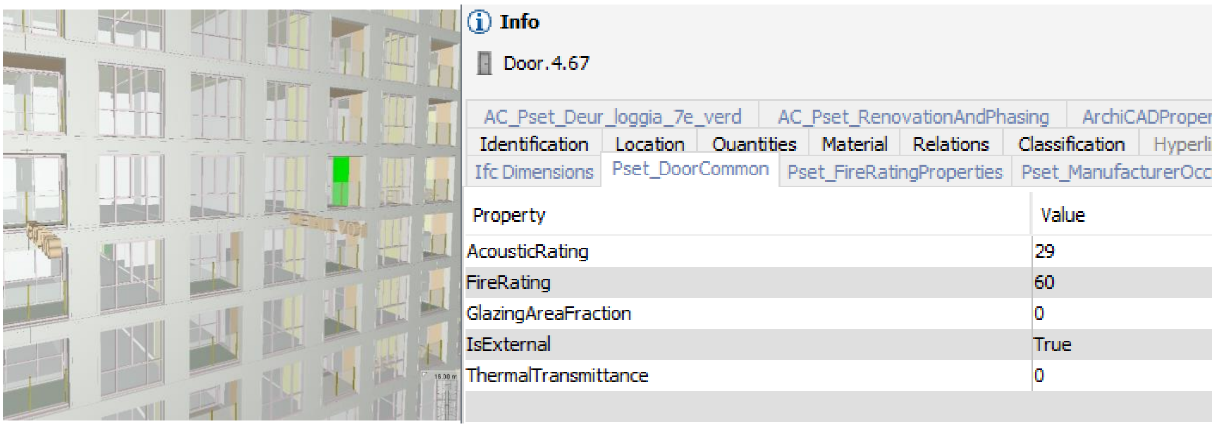


Figure 7.9. A fragment of the door-object and its properties in Solibri Model Viewer

7.2.3. The Design Verification SM.020

Proceeding with the next task namely the design verification in Cameo Systems Modeler, the prototype developed in section 6.3 can be used to transform the BIM-model to SysML (MOF-file). Before this transformation is executed, the BIM-model expressed ifcEXPRESS has to be converted into the ifcOWL standard for better data integration and interoperability. This conversion is realized through the *IFC to RDF-Desktop tool* shown in Appendix IV.

Subsequently, the transformation from IFC to SysML can be executed by using the prototype. As defined before, the prototype is limited to extract one random door-object including two of its key properties. This door-object and the two properties are besides shown in Figure 7.9. Therefore, due to the limitations of the prototype some assumptions are made. Considering the subsystem defined in Figure 7.8 during the briefing phase. It is assumed that the objects IfcSpace_Living Room, IfcSlab, IfcWindow, and IfcWall are already extracted from the BIM-model, whereby the object IfcDoor is missed in order to complete the subsystem for design verification. After the transformation is executed, the MOF-file is imported into the Cameo Systems Modeler tool wherein the subsystem is further composed and verified against the requirements. The result of this transformation is depicted in Figure 7.10, in which a block marked in blue is representing the object IfcDoor_2039251 and its properties Acoustic Rating=29, and Fire Rating=60 completing the subsystem in the BDD. In accordance, this subsystem can be compared and checked against the one defined earlier (see Figure 7.8), to ensure that both are similarly consistent, and so on satisfying and verifying the requirements.

However, comparing the two diagrams (Figure 7.9 and 7.10) against each other, Figure 7.10 differs in some aspects from the original one depicted in Figure 7.9. The reason for this deviation is by the fact that Figure 7.9 denotes the specification as-required, and Figure 7.10 denotes the design as-designed. In this comparison, the specification indicates that the IfcSpace_Living Room has one too many external doors (multiplicity 1..*), while in the design it has become one door. Despite this variation, the design meets the specification since the room is provided with an external door for accessibility to the balcony. Another difference can be seen regarding the Acoustic Rating specified for IfcDoor, which indicates a value of 20 dB the minimum according to the building regulations. The design shows a value of 29 representing a higher value than the specification meeting this condition. As last one, the variation in relation between IfcWall and IfcDoor_2039251. The specification comprises a relation based on a reference composition due to the assumption that IfcDoor is part of the IfcWall through its opening. Which is not the case according to the design. The design depicted a navigable association referencing the installation of the door to the wall through a port as shown in Figure 7.10 and by the IFC-viewer in Figure 7.9.

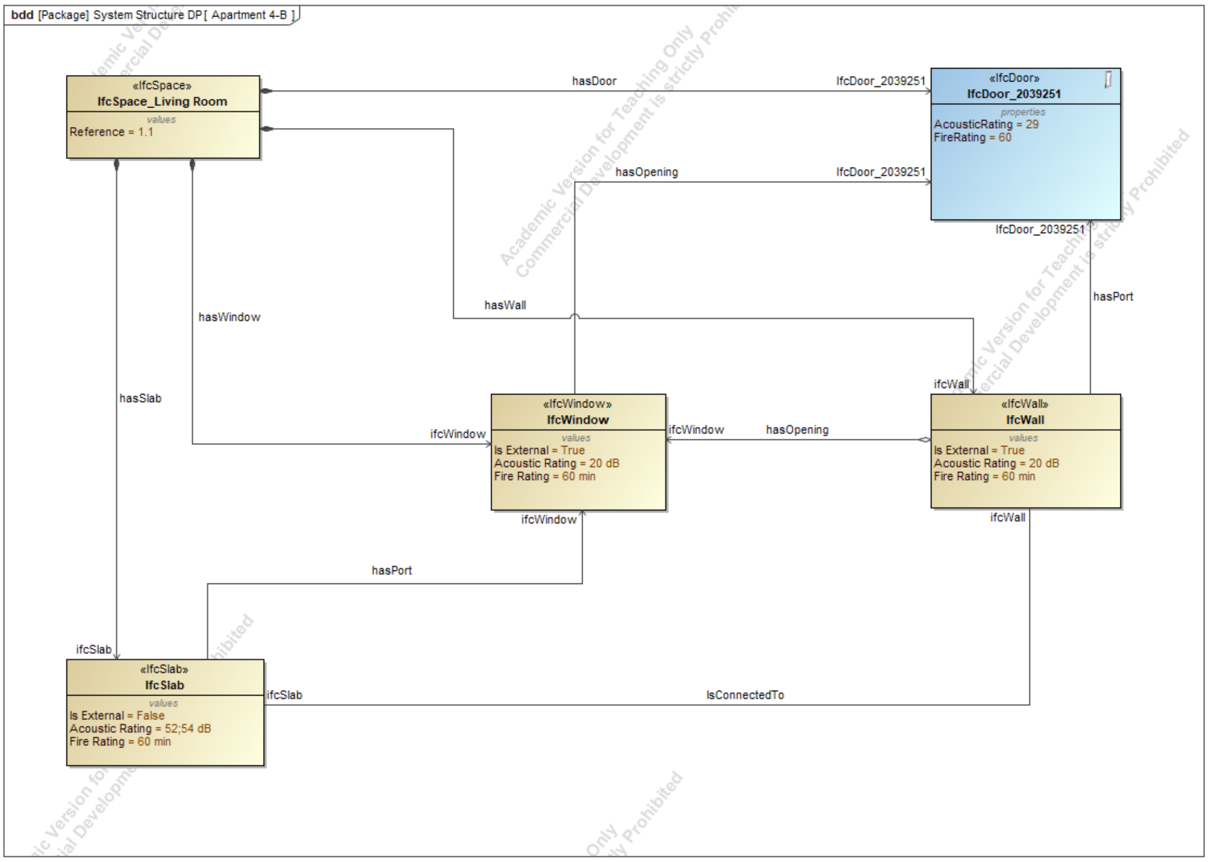


Figure 7.10. The Block Definition Diagram representing the subsystem structure of Apartment 4-B based on the design model

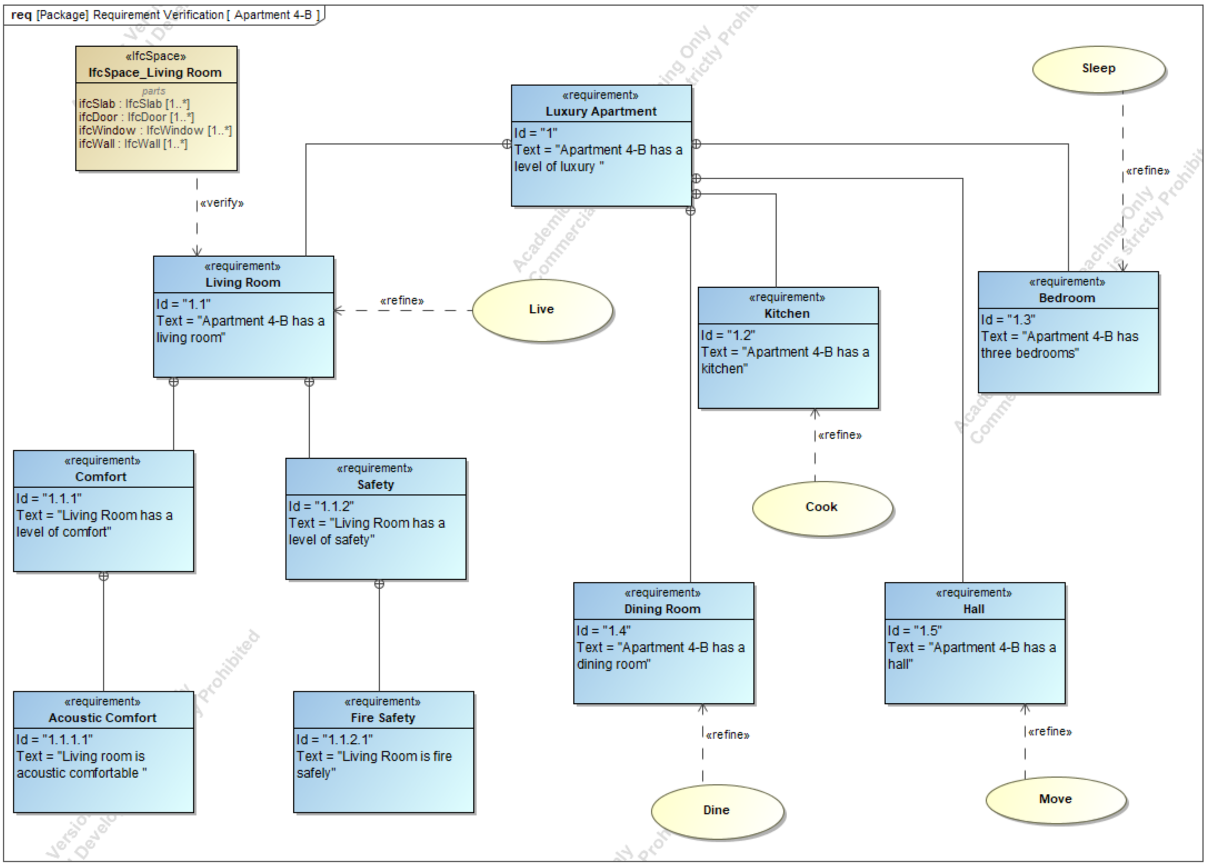


Figure 7.11. The Requirement Diagram representing the system requirements of Apartment 4-B

Nevertheless, the deviation of the design from the specification in this example, it is not directly affecting the fulfillment of the system requirement defined in section 7.1.2.2. By the fact that the connection between those objects is undisturbedly existing despite the distinction in associations. On the other hand, this deviation could have an impact on the requirements regarding architectural and esthetical conditions of the residence building, which is not part of the validation scope.

In conclude, it can be seen that the design meets the requirements specified earlier in the briefing phase. The satisfy relation between the requirements and IfcSpace_Living Room can be replaced by the verify relationship (see Figure 7.11). Accordingly, the results can be captured in a verify requirement matrix (see Figure 7.12) for better visualization and consistency of them composing a new system model that comprises the building and design specification to proceed with the construction phase.

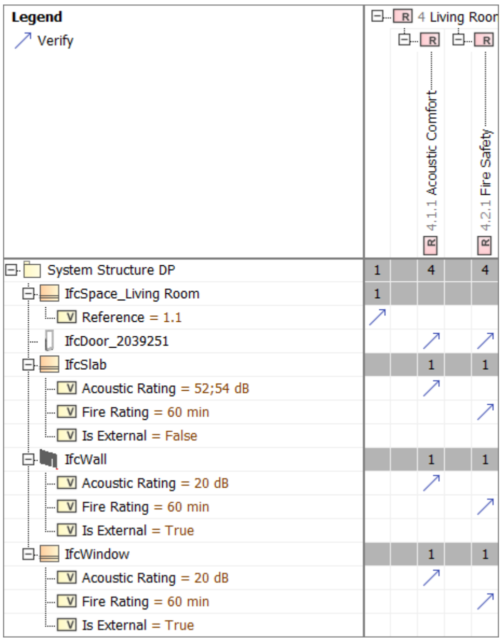


Figure 7.12. Verify requirement matrix (VRM)

5 Conclusion

8. Conclusions and recommendations

8.1. Conclusion

Considering the increasing complexity of construction projects whereby project requirements contributing to its evolution. This research has explored the integration of SysML and BIM to improve the systematic identification, capturing, and verification of functional requirements in integrated AEC-projects to cope with project complexity. As a result, the creation of a theoretical and practical framework leading to the development of the SysML-BIM Integration model depicted in Figure 6.1, and its evaluation through the case study: Project Sixty5 in Eindhoven, the Netherlands. In accordance, the conclusion covers these parts through a discussion of the sub-questions to the main-question following by the limitations, recommendations, and future work.

What are the characteristics of integrated project environments and functional requirements?

In this research, integrated project environments can be characterized as an approach concerning the integration of aspects as product, process, human and organizations to allow synergistic benefits to be realized successfully. In this interactive approach, product is referring to the building as a complex system including variety of independent components; where process is embracing the whole lifecycle of this system; lastly, human and organizations are comprising the involvement of many stakeholders in this process. On the basis of this integration lies the functional requirements outlining the ideas and necessities of the stakeholders involved reflecting a lifecycle-oriented and an integrated angle.

How are functional requirements identified, captured and verified in integrated project environments?

Posterior to the functional requirements are defined and documented by the client, the process of identification, capturing and verification of these requirements follows, and will be executed by the multidisciplinary team in charge. The procedure to execute this process differs per project, and it is depending on the client, design team, size and complexity of the project. In most projects, those requirements are manually specified and verified in spreadsheets. While in other ones, they are specified and verified in pragmatic requirement tools/databases that in most cases are linked to design models.

What are the characteristics of SysML and the OpenBIM-standard IFC?

SysML can be characterized as a powerful visual modeling language that provides a comprehensive set of diagrams and constructs; allowing a graphical mapping of requirements, structure, and behavior. Accordingly, it provides a complete description of the system of interest in an integrated system model including its key components and environment based on diagrams and models. On the other hand, the OpenBIM-standard IFC can be defined as an open interoperability standard for the establishment of the Big Open BIM goal, by performing vendor-neutral exchange of digital building models throughout the entire AEC-FM project-lifecycle. For this purpose, the standard features concepts as geometry, relations, processes, material performances, fabrication, and other properties.

How can SysML and the OpenBIM-standard IFC contribute to a better consistency of functional requirements during the briefing and design process?

Focusing on SysML, the SysML Diagram Taxonomy (see Figure 3.2) includes nine diagrams that can be used to represent aspects of a system, components and/or other entities. Regarding the functional requirements, the SysML Requirements Diagram can be used in the briefing phase to model these kinds of requirements resulting in a hierarchy of requirements, and defining what the system is required to accomplish and provide to its users. Within this hierarchy, all the requirements are captured in different abstractions, whereby their definitions and aims are described more explicit. At the same time, they are closely related by several kinds of relationships showing the interrelationship between them in a coherent view.

Furthermore, the Requirement Diagram as a whole or the requirements individually can be related to other SysML diagrams as the Package Diagram, Behavior Diagram, Parametric Diagram, or Structure Diagram, to additionally describe their features more explicitly and/or how they can be satisfied. Consequently, forming a system specification model to implement in the design process as reference point for discussing the design development and validation.

Supplementary, the OpenBIM-standard IFC as a data model, provides several concepts from the AEC-industry that can be captured in the SysML Structure Diagram or others. In the case of the functional requirements, the concepts associated with the building structure can be captured in the SysML Block Definition Diagram, and applicated in the briefing phase to assist in the understandability and definition of these requirements. Whereby for instance requirements could timely be satisfied, adapted or new requirements be derived to subsequently proceed with the design supervision and validation in the design phase.

How can SysML and the OpenBIM-standard IFC be linked to create interoperability for exchanging information?

In response to the integration, SysML can be customized to define domain-specific semantics and create a bridge to multiple domains as well as a link with the OpenBIM-standard IFC. For the former one, SysML can be extended by so-called profiles providing constructs to define a domain-specific language representing different disciplines of choice; therefore, improving productivity and communication among them. Profiles are special packages containing a set of supporting definitions and stereotypes that serve for concept definition to potential domains. For the latter one, the bridge to multiple domains is feasible through key data exchange mechanisms ensuring data interface between them.

At the side of the OpenBIM-standard IFC as an interoperable data-model, this model is expressed based on the EXPRESS language including digital data information, representation, and exchange. In addition, the standard is obtainable as: (a) XML in order to ensure homogeneity and interoperability between the data-model and XML-based applications; and (b) RDF and OWL to add valuable interoperable advantages to the AEC industry.

How can an integration between SysML and BIM be created to improve the systematic identification, capturing and verification of functional requirements in integrated AEC-projects?

Given the discussion described before, an integration between SysML and BIM can be created by using their interoperability components and features to interrelate them circularly instead of linearly; since a direct one-to-one integration is impractical because of their differences in implementation. At the side of SysML, a link with BIM can be created by using its constructs as the SysML profile and stereotypes. Oppositely, the link with SysML is achievable based on BIM its open interoperability standard IFC following by a model transformation approach.

Focusing on the initial link, the SysML profile and stereotypes can be used to define concepts originated from BIM, and subsequently, create a domain-specific language (DSL) that meets the domain purpose. In this connection, the alignment with the IFC-standard is essential since the standard comprises the concepts associated with the BIM process. On the opposite side, the link is viable through an XSLT-transformation whereby the open-standard expressed in RDF-OWL can be transformed into the MOF-

XMI standard. Similar as in the previous link, the alignment with the SysML Profile in this part of the interaction is fundamental.

In this integrated approach, SysML can be defined as a visual modeling language that supports specification, design, verification, and validation of complex systems through its diagram taxonomy. Underlying the aspects that define integrated AEC-projects, and the functional requirements as a basis to their distinguishing, definition and integration; demand a coherent and comprehensive set of specifications comprising the conditions of how to manage those projects. For this purpose, the SysML Diagram Taxonomy can be implemented to represent this set of specifications in a system model consequently replacing the brief explicitly. For instance, as can be seen in the case study conducted in section 7, a system model is created including a number of facets and diagrams specifying Apartment 4-B. In this model, the Use Case Diagram is used to model the System Functionality of Apartment 4-B denoting the functionalities that the system has to provide to its users, and how these can be achieved by the system objects.

In accordance, requirements can be derived from these functionalities and be associated with them. Otherwise, requirements that have already been defined can also be associated with these functionalities refining them more briefly. In response to this approach, the Requirement Diagram can then be adopted to model the requirements, and where needed connect them to the Use Case Diagram; forming the system model specification for Apartment 4-B. Accordingly, the Block Definition Diagram BDD can be implemented to model the apartment structure and definitely relates it to the Requirement Diagram depicting how the requirements – in interaction with the functionalities – will be satisfied completing the system model in the briefing phase.

Regarding the latter one is where the link between SysML to BIM is of great importance, wherein SysML is extended using profiles to meet the creation of the system structure in the BDD. Focusing on the IFC-standard as input to this creation, SysML profiles and the corresponding stereotypes can be used to define concepts originated from the standard covering the BIM process to great extent. In this definition process, objects as IfcSpace, IfcDoor, IfcWall, IfcSlab, IfcWindow and their interrelated attributes and relations can be defined in SysML to fit several purposes. Consequently, they can be model in the BDD to represent the system structure and be connected to other diagrams such as the Requirement Diagram as well as depicted in the case study (see Figure 7.5).

Proceeding with the design phase, the system model – depicting the system specification through the three diagrams – can be used as a reference point for the design development and verification. In this phase is the design development based on the BIM principles, which can be defined as an integrated approach to specify, design, build, and operate buildings using digital models through the open interoperability standard IFC. For this last part is where the link between BIM to SysML is very essential, in which the IFC standard expressed in RDF-OWL is transformed into the MOF-XMI standard. Taking into account the SysML profile as basis to this transformation, IFC-concepts can be transformed based on an XSLT-transformation approach, and imported in a SysML tool in order to capture, check and verify if the design is complying with the system model specification defined earlier. Accordingly, the analyses and/or results from this phase can then be captured in the system model extending this one and consequently proceed with the construction phase. These results can be shown or shared in various manner, namely: model viewpoints, or tabular matrices and tree views as illustrated in the case study.

To conclude, the SysML-BIM Integration Model yields an integrated and a model-based approach in which functional requirements are identified, captured and structured based on diagrams and models instead of documents. As a consequence, coping with their complexity regarding structuring, interrelationships, and traceability making them well-arranged to manage the process of development; in addition, improving the communication process between project participants. Expressed based on SysML as language, the requirements are becoming machine-readable ones and accessible by everyone involved in the process.

On the other hand, the association with the integrated design process corresponding to BIM as a process, ensures a comprehensive specification model and its consistent interaction with the design model; therefore, conserving a constant validation process. Additionally, it supports an earlier definition of the system model specification reflecting the building as-required as well as the interdisciplinary interactions meeting the perquisites of complex systems, and lifecycle-oriented integrated building projects. On the basis of this interaction lies the IFC open standard providing the concepts associated with the AEC-industry to support interoperability with other domains along with use cases as specification and verification of requirements. Expressed as RDF-OWL to enable deep flexibility, interoperability and integration overcoming the limitations of EXPRESS and XML. This standard can be transformed applying the model-transformation approach leading to a model-to-model collaboration. For this purpose, the IFC-to-SysML Transformation Tool in Appendix V is developed to automate the transformation; therefore, contributing to this emergent way of collaborating.

Nevertheless, the research approach has brought some limitations due to the time-frame and objectives in which the research is conducted. As a result, limiting the implementation of the IFC-BIM Integration model in some cases. The following section covers some of them briefly.

8.2. Limitations

Beginning with the theoretical framework, the research has focused only on measurable functional requirements while excluding other project requirements as non-functional requirements as well as architectural and esthetical requirements. Project requirements are in general very complex and broad in categories, which demands a more systematic and comprehensive approach in which they all can be covered. Therefore, this research is limited to functional requirements.

Another aspect is the SysML diagrams adopted to identify and capture functional requirements into a system model. For this purpose, the Use Case Diagram, Requirements Diagram, and Block Definition Diagram are used instead of the seven from the PSM-methodology according to Geyer (2012). This exclusion is due to the level of abstraction in which the requirements are specified during the case study to meet the research objective. A more level of detail will demand the inclusion of other diagrams to meet this target. As can be seen in Appendix VI, the PSM-methodology is developed to model sustainable building design using seven diagrams.

About the IFC-standard and the several languages – EXPRESS, XML, RDF-OWL – in which it is expressed. The integration model covers only the RDF-OWL standard due to its valuable interoperable advantages to the AEC-industry as shown in section 6. As a result, overcoming the limitations of: (a) EXPRESS regarding its limited expression range, and difficulties in quick adaption or extension of the schema; and (b) XML concerning the semantic interoperability, and the complexity that arises when representing information of a building.

Finally, the last limitation concerns the IFC-to-SysML Transformation Tool developed to extract data from the IFC data-model and transform it into SysML. As can be seen, the tool is limited to automatically extract exclusively one door-object and two of its properties to cover the use case and case study. However, it proves the feasibility to extract objects including their related components, which can be adapted and transformed to achieve a target.

8.3. Recommendations

As concluded, integrated AEC-projects are becoming more complicated as a result of the high-level expectations of clients and building users regarding comfort and sustainability. The traditional and/or ad-hoc approaches to cope with this complexity are not effective enough; consequently, misinterpretation of this phenomenon resulting in exceeding of project cost and schedule, or poor project performance. Focusing on the functional requirements as a basis to support these projects; demands an integrated approach in which these requirements are identified and captured in a system model specification instead of documents; whereby including a link with the design model for a consistent validation process. For this purpose, the SysML-BIM Integration model can be implemented. The following sentences depict somewhat recommendations for an efficient and effective application.

The first recommendation is regarding the introduction of SysML as a standard language in the building design domain. SysML is a system modeling language that is well-known and applicated by system engineering disciplines as astronautics, aeronautics, automotive, and software design. In the AEC-industry, it has less popularity and/or totally unknown by others. For a proper implementation in the building design, project members firstly have to become conversant with it. The way to achieve this purpose is to prepare them through additional courses and training showing them the potentials of this language, and how it can be used as standard to improve requirements management and communication among them.

Alternatively, this introduction and application of the SysML in the building domain could be limited to specialized individuals or team, who their core task will be to ensure a systematic description and supervision of the system based on SysML. In response to this prerequisite, disciplines as a system engineer with acquirable knowledge about complex systems, systems modeling, and building design can be appointed. In accordance, requirement-management tools/databases can be used to communicate and distribute the information to project members, since SysML is computer-executable.

The second recommendation comprises the specification process in the briefing phase. In most of the projects, this phase is not properly executed or too short in time. As a consequence, requirements are not enough specified or achieved during the development process resulting in project failures. Implementing SysML to overcome these limitations, demands an approach in which the phase is included with sufficient resources to define the boundaries wherein the project will be developed. Accordingly, the model-based approach through SysML requires a tool to model the system components, and a systematic method of how to model these components to meet the project development process.

The third and last recommendation comprises the validation process during the design phase. After the system specification model is created and the design development is executed through several stages, it can frequently be checked against the specification model for validation. This verification process takes place in a SysML tool by using its corresponding constructs and functions. For this purpose, the design model is transformed into SysML and imported into the tool, whereby the comparison between models can take place and the consistency between them be checked. This process can visually be executed or be automated for an efficient and effective execution. Regarding the transformation, a software/tool can be implemented as the prototype developed in this research showing the benefits of an automated transformation. However, a connection with the tool through an import-function or a plug-in is also feasible and seen as optimal solutions.

8.3.1. Future research

Taken into consideration the works from Geyer (2012), Baundains et al. (2014), and Valdes (2016); thereby the conclusions, limitations and recommendations of this research. The future research approaches should focus on the further implementation of SysML as system modeling language in integrated AEC-projects, to cope with project complexity. Subsequently, overcoming the limitations of mentioned before, the following propositions are made.

The first proposition suggests a research into the SysML diagrams that are excluded from this research. This approach should provide an overview of how these diagrams will additionally be used to specify buildings more clear and precise. For instance, coping with the complexity of non-functional requirements as well as architectural and esthetical requirements. Besides, this proposition should cover an approach in which a methodology can be developed showing how the diagrams could strategically be implemented to specify buildings.

The second proposition advises the extension of the prototype developed to contribute to consistent and optimal integration between SysML and BIM. In this development, the prototype is limited to extract a door-object including two of its key properties. The extension should provide additional functions to the users, whereby they could simply navigate through the model with the possibility to query, extract, adapt and transform data.

The third and last proposition proposes a further research in the application of the integration model in the construction and/or operation phase. The implementation of the model in this research is limited to the briefing and design process. The research approach that will follow from this proposition should focus on how all the design information – which is captured in the system specification model during the design phase – could instrumentally be implemented to build and operate building projects.

Bibliography

Acaroglu, L. (2017, September 7). *Tools Of A System Thinker*. Retrieved from Disrupt Design: https://medium.com/disruptive-design/tools-for-systems-thinkers-the-6-fundamental-concepts-of-systems-thinking-379cdac3dc6a

Akio. (2017, May 2). *Leveraging the Full Value of BIM's Interoperability Potential*. Retrieved from Dassualt Systemes: https://blogs.3ds.com/perspectives/leveraging-full-value-bims-interoperability-potential-2/

Alsharif, M. (2013, December 10). *Semantic Web Core Technlogies.* Retrieved from McKelvey School of Engineering: https://www.cse.wustl.edu/~jain/cse570-13/ftp/semantic/index.html

Barosan, I. (2017, 11 23). *7M900: Fundamentals of Building Information Modeling.* Retrieved from Canvas: https://canvas.tue.nl/courses/4740/files/folder/Week1

Baundains, P., Bischop, S., Duffour, P., Marjanovic-Halburd, L., Psarra, S., & Spararu, C. (2014). A systems paradigm for integrated building design. *Intelligent Building International*(6), 201-214. doi:https://doi.org/10.1080/17508975.2014.935696

Beetz, J. (2009). *Facilitating Distributed collaboration in the AEC/FM sector using Semantic Web Technologies.* Eindhoven: Eindhoven University of Technology. doi:http://dx.doi.org/10.6100/IR652808

Beetz, J., Leeuwen van, J., & Vries de, B. (2009). IfcOWL: A Case of Transforming EXPRESS Schemas into Ontologies. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*(23), 89-101. doi:https://doi.org/10.1017/S0890060409000122

Beetz, J., Leeuwen, J. P., & Vries de, B. (2007). RDF-Based Distributed Part Specifications for the Facilitation of Service-Based Architectures. *Eindhoven University of Technology: Design Systems group*, 183-188. Retrieved from https://www.researchgate.net/publication/250869030_RDF-Based_Distributed_Functional_Part_Specifications_for_the_Facilitation_of_Service-Based_Architectures

Berners-Lee, T., Hendler, J., & Lassila, O. (2001, May). *The Semantic Web: A New form of Web content that is meaningful to computers will unleash a revolution of new possibilities.* Retrieved from https://www-sop.inria.fr/acacia/cours/essi2006/Scientific%20American_%20Feature%20Article_%20The%20Semantic%20Web_%20May%202001.pdf

Borrmann, A., König, M., Koch, C., & Beetz, J. (2018). *Building Information Modeling: Technology Foundations and Industry Practice.* Germany: Springer International Publishing AG. Retrieved from https://link.springer.com/book/10.1007%2F978-3-319-92862-3

BuildingSMART. (1996-2006). *IFC4 Add2 - Addendum 2.* Retrieved from BuildingSMART International: http://www.buildingsmart-tech.org/ifc/IFC4/Add2/html/

Cardoso, J. (2007). *Semantic Web Services: Theory, Tools, and Applications.* Madeira, Portugal: Informantion Science Reference.

Cheng, J., Kumar, B., & Law, H. K. (2002). A Question Answering System for Project Management Applications. *Advanced Engineering Informatics*(16), 227-289. doi:https://doi.org/10.1016/S1474-0346(03)00014-4

Construction User Roundtable. (2004, August). *Collaboration Integrated Information and the Project Lifecycle in Building Design, Construction and Operation.* Retrieved from Kentuckiana Construction Users Council: https://kcuc.org/wp-content/uploads/2013/11/Collaboration-Integrated-Information-and-the-Project-Lifecycle.pdf

Construction User Roundtable. (2004, August). *KCUC.* Retrieved from Kentuckiana Construction Users Council: https://kcuc.org/wp-content/uploads/2013/11/Collaboration-Integrated-Information-and-the-Project-Lifecycle.pdf

Czarnecki, K., & Helsen, S. (2006). Feature-Based Survey of Model Transformation Approaches. *IBM Systems Journal*(45), 621-645. Retrieved from https://doi.org/10.1147/sj.453.0621

Deshpande, A., Azhar, S., & Amireddy, S. (2014). A Framework for a BIM-based Knowledge Management System. *Procedia Engineering*(85), 113-122. doi:https://doi.org/10.1016/j.proeng.2014.10.535

Diederendirrix. (2019). *Sixty 5*. Retrieved from Diederendirrix: https://www.diederendirrix.nl/nl/projecten/sixty5/

Eastman, C., Teicholz, P., Sacks, R., & Liston, K. (2011). *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors* (Second ed.). New Jersey: John Wiley & Sons, Inc. .

Friendenthal, S., Moore, A., & Steiner, R. (2015). *A Practical Guide to SysML* (Vol. Third edition). Waltham , USA: Elsevier Inc. doi:https://doi.org/10.1016/B978-0-12-800202-5.00003-5

Froese, T. M. (2010). The impact of emerging information technology on project management for construction. *Automation in Construction*(19), 531-538. doi:https://doi.org/10.1016/j.autcon.2009.11.004

Gaševic, D., Djuric, D., & Devedžic, V. (2006). *Model Driven Architecture and Ontology Development.* Germany: Springer-Verlag Berlin Heidelberg. doi:https://doi.org/10.1007/3-540-32182-9

Geyer, P. (2012). Systems modelling for sustainable building design. *Advanced Engineering Informatics*(26), 656-668. doi:https://doi.org/10.1016/j.aei.2012.04.005

Hoeve ten, G. (2018). *Improving Collaboration between Client and Contractor in Integrated Contracts in the Dutch Construction Sector.* Delft : Delft University of Technology. Retrieved from http://resolver.tudelft.nl/uuid:7fbe5a20-53b7-4237-be67-67a6ea793502

Ilhan, B. (2014). *An IFC-Based Framework for sustainable construction.* Instabul, Turkey: Instanbul Technical University. Retrieved from https://polen.itu.edu.tr/bitstream/11527/14099/1/10048340.pdf

Jallow, A. K., Demian, P., Baldwin, A. N., & Chimay, A. (2014). An empirical study of the complexity of requirements management in construction projects. *Engineering, Construction and Architectural Management*(21), 505-531. doi:DOI 10.1108/ECAM-09-2013-0084

Jenkins, J. S., & Rouquette, F. N. (2012). Semantically-Rigorous Systems Engineering Modeling Using SysML and OWL. *5th International Workshop on Systems & Concurrent Engineering for Space Applications* (pp. 1-8). Lisbon, Portugal: Jet Propulsion Laboratory, National Aeronautics and Space Administration. Retrieved from http://hdl.handle.net/2014/43338

Johnson, T., Paredis J.J., C., & Burkhart, R. (2008). Integrating Models and Simulations of Continuous Dynamics into SysML. *6th International Modelica Conference* (pp. 135-145). Bielefeld, Germany: The Modelica Association. Retrieved from https://www.modelica.org/events/modelica2008/Proceedings/sessions/session2a2.pdf

Kiviniemi, A., & Laakse, M. (2012). The IFC Standard- A Review of History, Development, and Standardization. *Journal of Information Technology in Construction*(17), 134-161. Retrieved from https://helda.helsinki.fi/dhanken/bitstream/handle/10138/36870/Laakso_Kiviniemi_2012_The_IFC_Standard_A_Review_Of_History_Development_And_Standardization.pdf?sequence=2&isAllowed=y

Kooralla, L. (2018). *Systematic Design of Offshore Pipelay Equipment and Subsea Tools Based on the Application of SysML Profiling.* Eindhoven, Netherlands: Eindhoven University of Technology. Retrieved from https://research.tue.nl/

Locatelli, G., Mancini, M., & Romano, E. (2014). Systems Engineering to improve the governance in complex environments. *International Journal of Project Management*(32), 1395-1410. doi:https://doi.org/10.1016/j.ijproman.2013.10.007

LU, W., Fung, A., Peng, Y., Liang, C., & Rowlinson, S. (2015). Demystifying Construction Project Time-effort Distribution Curves: A BIM and non-BIM Comparison. *Journal of Management in Engineering*(31), 1-23. Retrieved from https://www.researchgate.net/publication/283427563_MacLeamy_Curve_-_With_figures

Maia, L., Mêda, P., & Freitas, G. J. (2015). BIM Methodology, a New Approach - Case Study of Structural Elements Creation. *Procedia Engineering*(114), 816-823. doi:https://doi.org/10.1016/j.proeng.2015.08.032

Moonen, L. T. (2016). *Improving the design process: The implications of automated verification of client specific requirements using semantic web standards and rule checking techniques.* Eindhoven: Eindhoven University of Technology. Retrieved from http://www.ofcoursecme.nl/education/master-thesis-database/

Murguia, D., Brioso, X., Ruiz-Conejo, L., & Fernandez, L. (2017). Process Integration Framework for the Design Phase of a Residential Building. *Procedia Engineering*(196), 462-469. doi:https://doi.org/10.1016/j.proeng.2017.07.225

Nassar, N., & Austin, M. (2013). Model-Based Systems Engineering Design and Trade-Off Analysis with RDF Graphs. *Procedia Computer Science*, 216-225. doi:https://doi.org/10.1016/j.procs.2013.01.023

Pauwels, P., & Terkaj, W. (2016). EXPRESS to OWL for Construction Industry: Towards a Recommendable and Usable ifcOWL Ontology. *Automation in Construction*(63), 100-133. doi:https://doi.org/10.1016/j.autcon.2010.11.017

Pauwels, P., Deursen Van, D., Verstraeten, R., Roo De, J., Meyer De, R., Walle de Van, R., & Campenhout Van, J. (2011). A Semantic Rule Checking Environment for Building Performance Checking. *Automation in Construction*(20), 506-518. doi:https://doi.org/10.1016/j.autcon.2010.11.017

Pauwels, P., Zhang, S., & Lee, Y.-C. (2017). Semantic Web Technologies in AEC Industry: A literature Overview. *Automation in Construction*(73), 145-165. doi:https://doi.org/10.1016/j.autcon.2016.10.003

Pels, H. J., Beek, v. J., & Otter, d. A. (2013). Systems Engineering as first step to effective use of BIM. *Product Lifecycle Management for Society* (pp. 651-662). Nantes, France: IFIP Advances in Information and Communication Technology . Retrieved from https://link.springer.com/content/pdf/10.1007%2F978-3-642-41501-2_64.pdf

Ruijven van, L. (2018). *A Unified Framework improving Interoperability and Symbiosis in the Field of Systems Engineering.* Delft: Delft University of Technology. doi:https://doi.org/10.4233/uuid:e622ee5a-b551-4f8c-a250-9859db791d22

SBR. (2006). *Functional Specificeren in de Bouw: Stimulans voor Innovatie en Optimale Oplossinen.* Delft: Thieme Media Services.

Schenk, D., & Wilson, P. (1994). *Information Modeling: The EXPRESS Way.* New York, Oxford: Oxford University Press.

Shah, A. A., Kerzhner, A. A., Schaefer, D., & Paredis, J. C. (2010). Multi-View Modeling to Support Embedded Systems Engineering in SysML. *Graph Transformations and Model-Driven Engineering*, 580-601. doi:https://doi.org/10.1007/978-3-642-17322-6_25

Shen, L., & K.H. Chua, D. (2011). Application of Building Information Modeling (BIM) and Information Technology (IT) for project collaboration. *EPPM* , 67-46. Retrieved from http://www.ppml.url.tw/EPPM/conferences/2011/download/SESSION3/67_76.pdf

Spek, v. d. (2012). *Het verband tussen invloed en tevredenheid van een Design & Build opdrachtgever.* Delft: Technische Universiteit Delft. Retrieved from https://repository.tudelft.nl/islandora/object/uuid:097c6a84-3b4a-4396-b9fb-90965bd5a83b/datastream/OBJ/download

Stancheva, M. (2017). *Improving the management of structural engineering requirements in the design phase.* Eindhoven: Eindhoven Univerisity of Technology. Retrieved from http://www.ofcoursecme.nl/education/master-thesis-database/

Valdes, F. (2016). *Manufacturing Compliance Analysis For Architectural Design: A Knowledge-Aided Feature-Based Modeling Framework.* Georgia, USA: Georgia Institute of Technology. Retrieved from http://hdl.handle.net/1853/54973

Venugopal, M., Eastman, M. C., & Teizer, J. (2015). An Ontology-based analysis of the Industry Foundation Class Schema for Building Information Model Exchanges. *Advanced Engineering Informatics*(29), 940-957. doi:https://doi.org/10.1016/j.aei.2015.09.006

Weilkiens, T. (2008). *Systems Engineering with SysML/ML - Modeling, Analysis, Design.* Burlington, USA: Morgan Kaufmann Publishers. doi:https://doi.org/10.1016/B978-0-12-374274-2.X0001-6

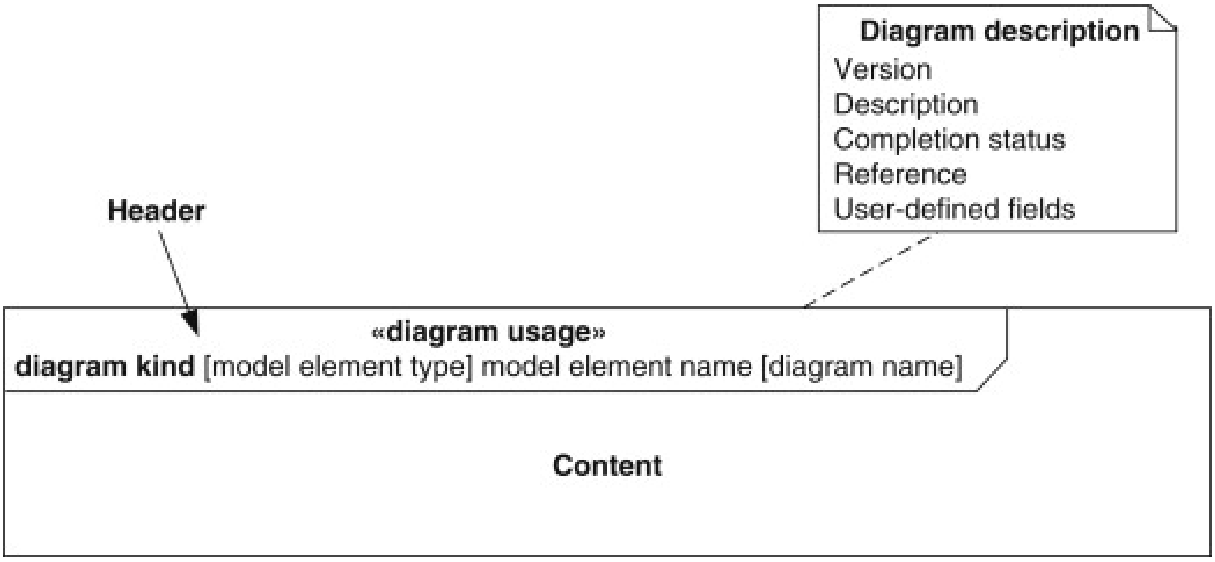
Winch, M. G. (2010). *Managing Construction Projects: An Information Processing Approach* (Second Edition ed.). United Kingdom: Blackwell Publishing Ltd and 2002 Blackwell Sciece Ltd.

Yahiaoui, A., Sahraoui, A.-E.-K., Hensen, J., & Brouwer, P. (2006). A Systems Engineering Environment for Integrated Building Design. *Proceedings of the EuSEC - European Systems Engineering*

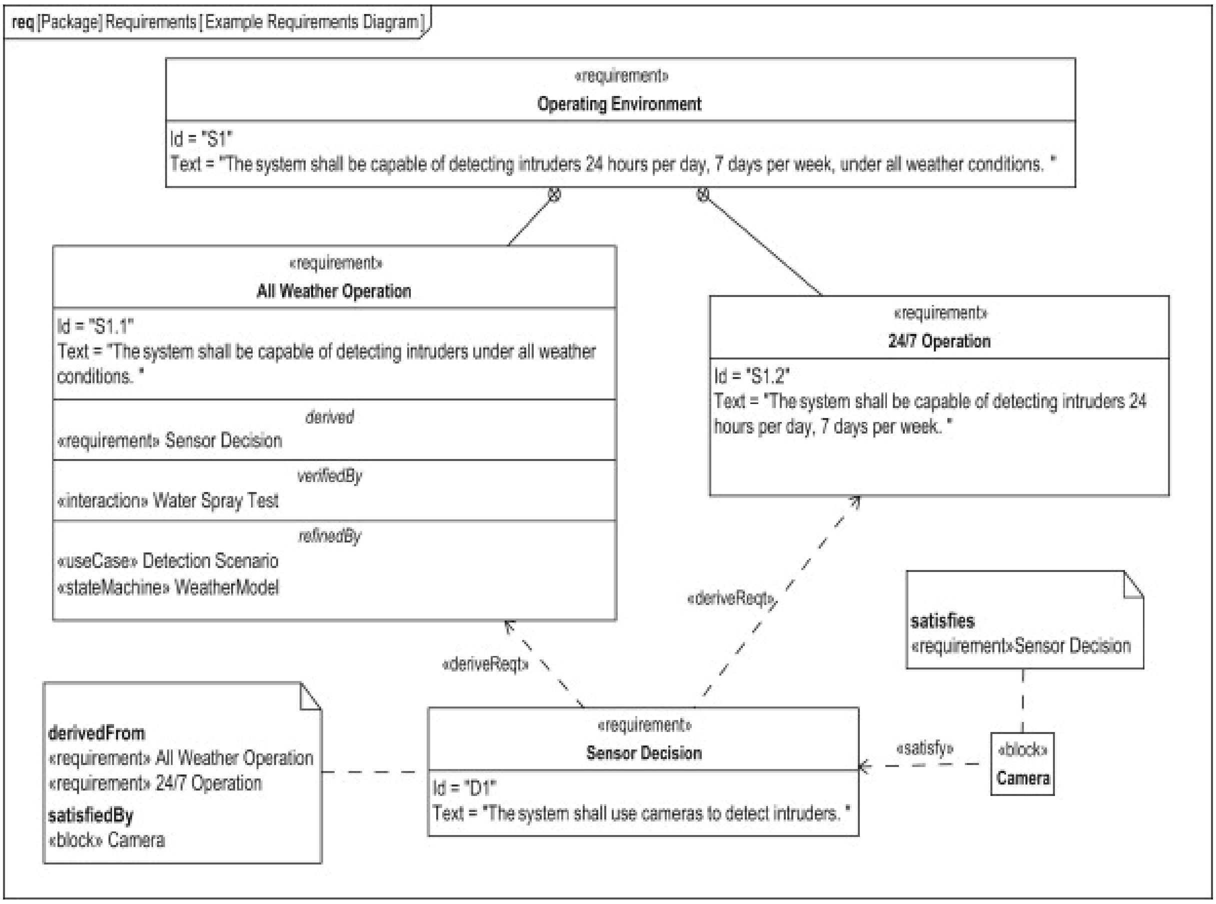
Conference (pp. 1-11). Edinburgh : The International Council on Systems Engineering (INCOSE). Retrieved from purl.tue.nl/619336541214086.pdf

Zhang, J., Webster, A., Lawrence, M., Nepal, M., Pottinger, R., Staub-Frenck, S., & Tory, M. (2011). Improving the Usability of Standard Schemas. *Information Systems*(36), 209-221. doi:https://doi.org/10.1016/j.is.2010.08.005

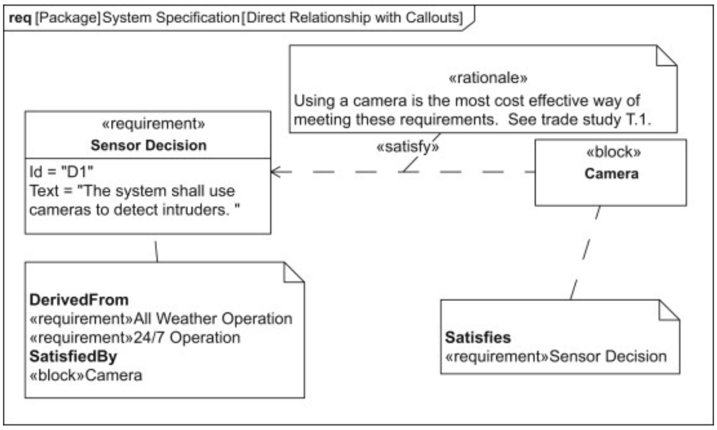
Appendix I Model-Based Systems Engineering (MBSE)



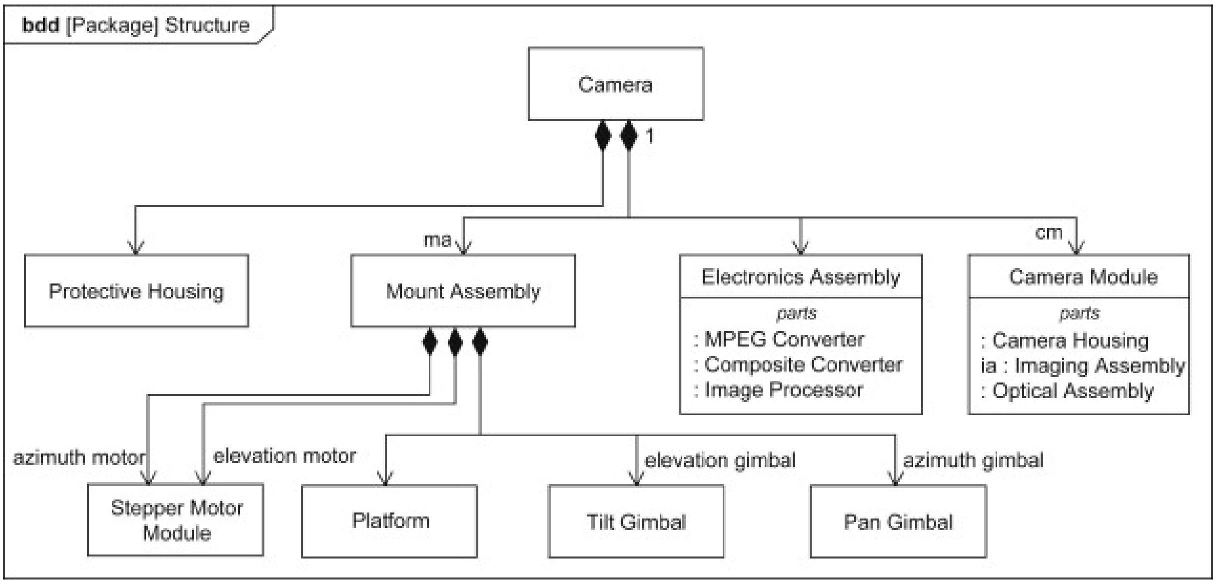
Appendix I-1. The SysML diagram frame including its main components (Friendenthal, Moore, & Steiner, 2015)



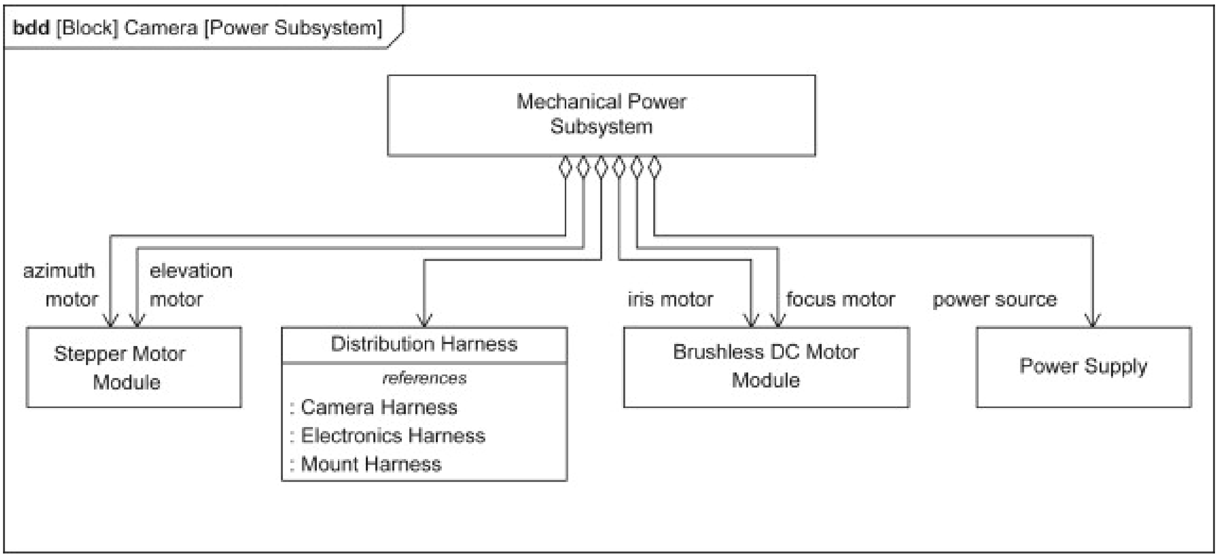
Appendix I-2. An example of a SysML Requirement Diagram (Friendenthal, Moore, & Steiner, 2015)



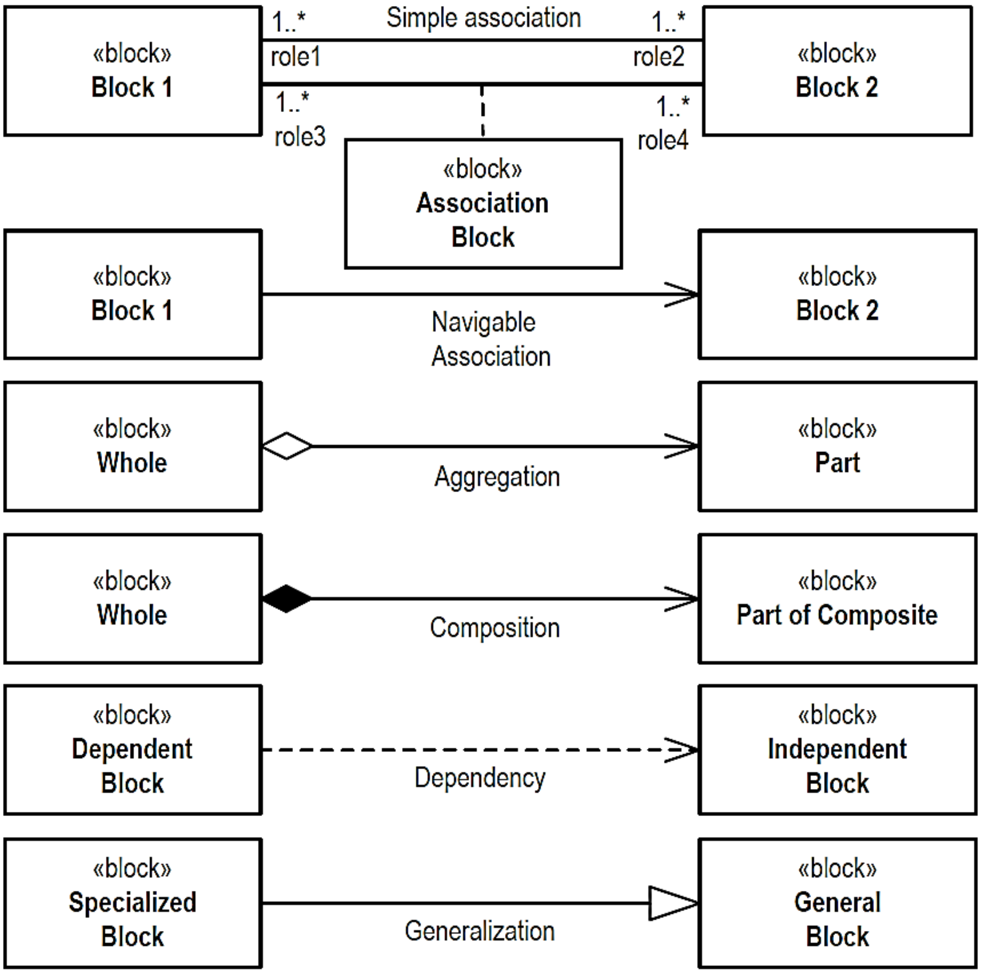
Appendix I-3. An example of a SysML Requirement Diagram illustrating the Callouts and Rationale notation (Friendenthal, Moore, & Steiner, 2015)



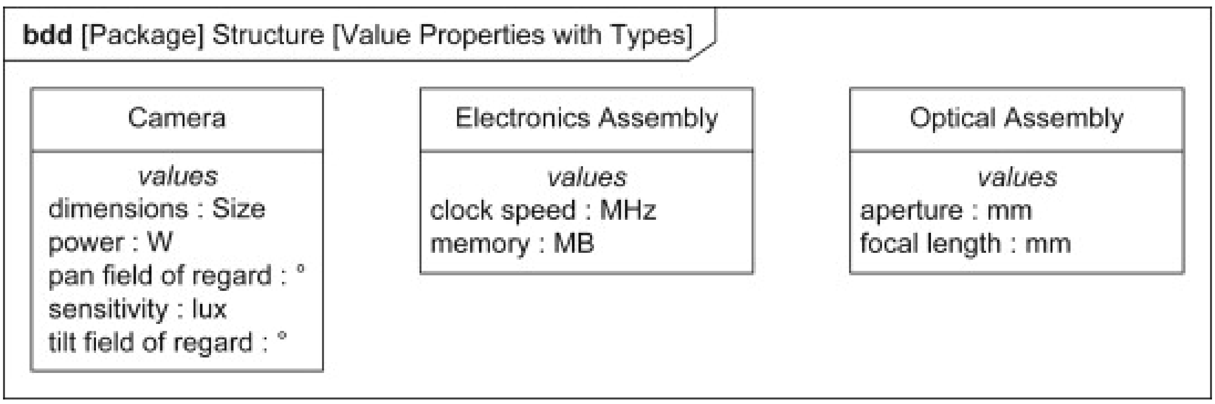
Appendix I-4. An example of a SysML Block Definition Diagram illustrating the composite association of a camera (Friendenthal, Moore, & Steiner, 2015)



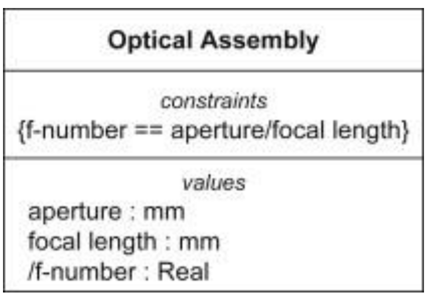
Appendix I-5. An example of a SysML Block Definition Diagram illustrating the reference composition of a mechanical Power subsystem (Friendenthal, Moore, & Steiner, 2015)



Appendix I-6. SysML associations (Barosan, 2017)



Appendix I-7. An example showing Value Properties (Friendenthal, Moore, & Steiner, 2015)



Appendix I-8. Constraints (Friendenthal, Moore, & Steiner, 2015)

Appendix II Building Information Modeling (BIM)

The language elements to build a specific EXPRESS schema according to Schenk & Wilson (1994) are the following ones:

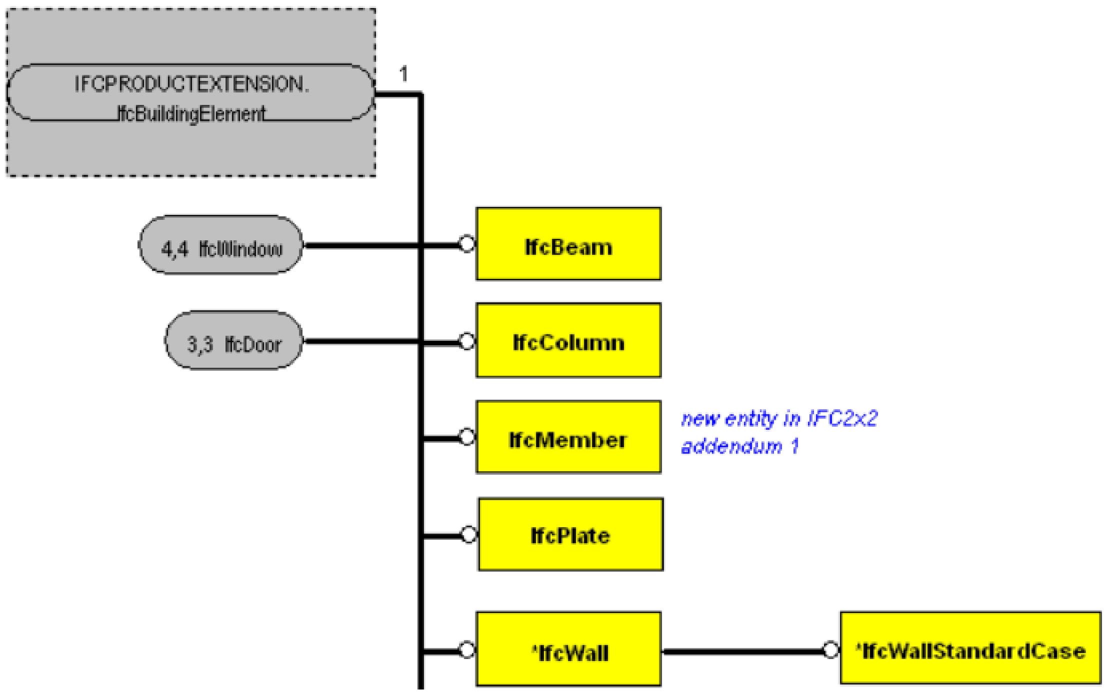
Entities

The entities are representing the respective ''real world'' items according to Schenk & Wilson (1994). In this case, they are classes and objects referencing to the IFC structure and concepts describing the building and construction domain. Their classification to a consistent structure is modelled based on subtypes and supertypes relationships. Which represents a directed graph where the nodes are the entities and the links represent the subtype to supertype relationships. Following the subtype of links leads to a more general types while the supertype of links leads to more specific types (Schenk & Wilson, 1994).

```
ENTITY IfcWall
    SUPERTYPE OF(IfcWallStandardCase)
    SUBTYPE OF (IfcBuildingElement);
    WHERE
        WR1                   : SIZEOF (QUERY(temp <* SELF\IfcObjectDefinition.HasAssociations
END_ENTITY;
```

Appendix II-1. IfcWall written in EXPRESS (BuildingSMART, 1996-2006)

Appendix II-1 shows an example of an entity defined in the IFC structure written in EXPRESS. In this case, the entity *IfcWall* is a supertype of the entity *IfcWallStandardCase* and subtype of the entity *IfcBuildingElement*. The entities are referencing to the entities in the IFC structure that describing the building and constructions concepts. Appendix II-2 shows the same information in a graphical notation instead of a textual one. This notation is denoted by solid rectangular boxes enclosing the name of the entities *ifcBuildingElement*, *IfcWall* and *IfcWallStandardCase*. Supertypes and subtypes are connected by a thick line, with the subtype end of the line denoted by a circle.



Appendix II-2. Graphical notation EXPRESS-G (BuildingSMART, 1996-2006)

Attributes

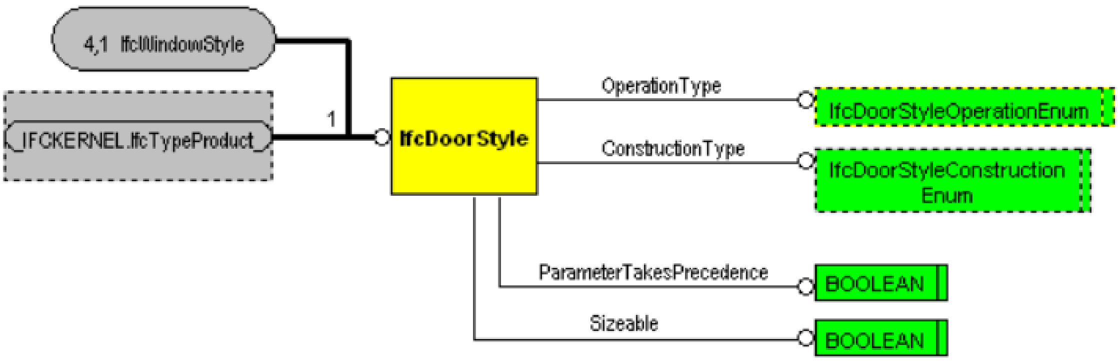
The attributes describe the characteristics of an entity which distinguish one entity from another, and also which values will distinguish one instance in a entity from another instance in the same entity. In EXPRESS, the attribute specification consists of a name for the attribute and the domain of the attribute. A domain can be considered to specify the type of value that the attribute will have in an instance of the entity. The name of the attribute should be indicative of its role within the description of the entity (Schenk & Wilson, 1994). In the EXPRESS-G representation, an entity icon is connected to the icons representing it attributes by thin lines. The circled end of the line indicates the ''attribute end'' of the connection. The name of the attribute is placed adjacent to the connecting line (Schenk & Wilson, 1994)

Appendix II-3 illustrates an example showing the entity *IfcDoorStyle* and its attributes written in EXPRESS*.* The attributes named OperationType, ConstructionType, ParametertakesPrecedence and Sizeable are part of this entity describing its characteristics. The domain of the attribute are respectively represented by the defined types *IfcDoorStyleOperationEnum*, *IfcStyleConstructionEnum*, and a simple type BOOLEAN.

```
ENTITY IfcDoorStyle
    SUBTYPE OF (IfcTypeProduct);
        OperationType                      : IfcDoorStyleOperationEnum;
        ConstructionType                   : IfcDoorStyleConstructionEnum;
        ParameterTakesPrecedence           : BOOLEAN;
        Sizeable                           : BOOLEAN;
END_ENTITY;
```

Appendix II-3. IfcDoorStyle and its attributes in EXPRESS (BuildingSMART, 1996-2006)

In an EXPRESS-G representation, Appendix II-5 shows the interaction between the entity and its attributes graphically. An icon including *IfcDoorStyle* is connecting to other icons representing its attributes by thin lines. The circled end of the lines indicates the end of the attribute. The names of the attributes are placed adjacent tot the connecting line.



Appendix II-4. Graphical notation EXPRESS-G (BuildingSMART, 1996-2006)

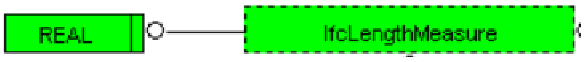
Simple-types

The simple types represent atomic units of data. In both EXPRESS and EXPRESS-G, they are: Binary, Boolean, Integer, Logical, Number, Real and String. However, they effectively carry no semantics. In this case, EXPRESS provides the ''TYPE'' construct that can be used to extend and add semantics to the simple types provided in the language. Appendix II-5 illustrates an example of a used simple type, in this case ''REAL'', to define the atomic unit for the defined TYPE *IfcLengthMeasure*. This TYPE is defined in order to distinguish one from another. In the IFC structure, it refers to ''the value of a distance'' according to BuildingSMART (1996-2006).

```
TYPE IfcLengthMeasure = REAL;
END_TYPE;
```

Appendix II-5. A simple type in EXPRESS (BuildingSMART, 1996-2006)

The simple type in EXPRESS-G is represented as a solid rectangular box, with a vertical line at the right/left-hand end, enclosing the name of the simple type in upper-case characters (see Appendix II-6). In this figure, the icon for the TYPE construct is a dashed box enclosing the type *IfcLengthMeasure.*



Appendix II-6. A simple type in EXPRESS-G (BuildingSMART, 1996-2006)

Select-types

A select type defined a named collection of other types called select list. A value of a select type is a value of one of the types specified in the select lists where each is an entity type or a defined types. This allows an attribute or variable to be one of several possible types. The domain of values for such a type is the union of the domains of the types in its selects (Schenk & Wilson, 1994). Considering an example of a select type from the IFC structure: *IfcMaterialSelect* defined the following types in a select list: *IfcMaterial, IfcMaterialList, IfcMaterialLayerSetUsage, IfcMaterialLayerSet, IfcMaterialLayer* (see Appendix II-7). In this case, these types are entities, which can be assigned to an attribute or variable (BuildingSMART, 1996-2006).

```
TYPE IfcMaterialSelect = SELECT
     ( IfcMaterial,
       IfcMaterialList,
       IfcMaterialLayerSetUsage,
       IfcMaterialLayerSet,
       IfcMaterialLayer);
END_TYPE;
```

Appendix II-7. A select type in EXPRESS (BuildingSMART, 1996-2006)

Enumerations

An enumeration type is an ordered list of values represented by names. The values of the enumeration type are designated by enumeration items. An enumeration item belongs only to the type that defines it and must be unique within that type definition. The order of the values of an enumeration type is determined by their relative position in the enumeration item list: the first occurring item is less that the second, the second is less than the third, etc. Comparison between values in different enumeration types is undefined even if the item names are the same. Two different defined types may have the same enumeration item.

In this case, to ensure that the reference to the enumeration item is unambiguous, the reference must be qualified with its type name on the following way: ''TypeRef.EnumRef'' (Schenk & Wilson, 1994).

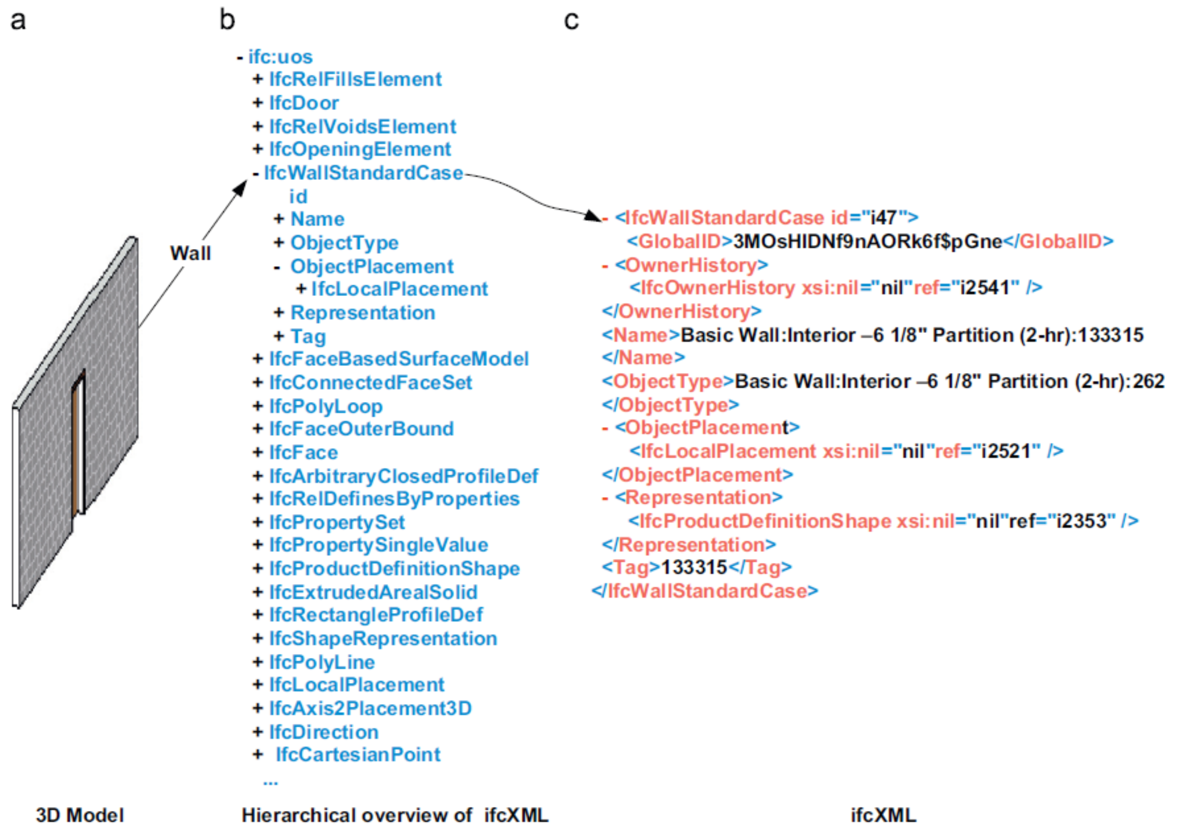
Taking the example from the section attributes, in Appendix II-3 can been seen that one of the attributes of the entity *IfcDoorStyle*, in this case OperationType, is representing by a enumeration type: *IfcDoorStyleOperationEnum.* This type is uniquely defined in order to describe the basic ways how doors operate, whereby includes a ordered list of operations represented by names (see Appendix II-8) (BuildingSMART, 1996-2006).

```
TYPE IfcDoorStyleOperationEnum = ENUMERATION OF
    ( SINGLE_SWING_LEFT,
      SINGLE_SWING_RIGHT,
      DOUBLE_DOOR_SINGLE_SWING,
      DOUBLE_DOOR_SINGLE_SWING_OPPOSITE_LEFT,
      DOUBLE_DOOR_SINGLE_SWING_OPPOSITE_RIGHT,
      DOUBLE_SWING_LEFT,
      DOUBLE_SWING_RIGHT,
      DOUBLE_DOOR_DOUBLE_SWING,
      SLIDING_TO_LEFT,
      SLIDING_TO_RIGHT,
      DOUBLE_DOOR_SLIDING,
      FOLDING_TO_LEFT,
      FOLDING_TO_RIGHT,
      DOUBLE_DOOR_FOLDING,
      REVOLVING,
      ROLLINGUP,
      USERDEFINED,
      NOTDEFINED);
END_TYPE;
```

Appendix II-8. Enumerations in EXPRESS (BuildingSMART, 1996-2006)

Collections-relations

Collection types are used to represent ordered or unordered collections. These collections can have fixed or varying sized depending on which specific collection type being considered. Each collection data type has a different behavior that suits it to different purposes (Schenk & Wilson, 1994). Schenk & Wilson (1994) distinguish the following collection types: List, Array, Set and Bag. In the IFC-structure, they are used to created collection of objects, relations and/or properties.



Appendix II-9. The ifcXML-standard representing information about a wall (Zhang, et al., 2011)

```
ENTITY IfcElement
      ABSTRACT SUPERTYPE OF     (ONEOF(IfcBuildingElement,
                    IfcFurnishingElement,[…]))
END_ENTITY;

ENTITY IfcBuildingElement
      ABSTRACT SUPERTYPE OF     (ONEOF(IfcDoor, IfcWall, IfcSlab, […]))
      SUBTYPE OF (IfcElement) ;
END_ENTITY;

ENTITY IfcDoor
      SUBTYPE OF (IfcBuildingElement) ;
END_ENTITY;
```

```
:IfcElement
      a         owl:Class ;
      rdfs:subClassOf owl:Thing .

:IfcBuildingElement
      a         owl:Class ;
      rdfs:subClassOf :IfcElement ;
      owl:disjointWith :IfcFurnishingElement

:IfcDoor
      a         owl:Class ;
      rdfs:subClassOf :IfcBuildingElement ;
      owl:disjointWith :IfcWall , :IfcWindow
```

Appendix II-10. An example of a transformation of entities (EXPRESS) into classes and subclasses (RDF/OWL) (Beetz, 2009)

Appendix III The practical view

Dit onderzoek draagt bij aan de verbetering van de huidige werkwijze tijdens de vertaalslag van functionele eisen binnen complexe multidisciplinaire bouwprojecten (AEC-projecten). Dit proces kenmerkt zich met het definitiegebied; vastlegging; en verificatie van functionele systeemeisen. Er is geconcludeerd vanuit het literatuuronderzoek dat deze eisen binnen de bouwindustrie onvoldoende (nauwkeurig) vertaald, gespecificeerd en gemonitord worden tot op object niveau binnen de gehele levenscyclus van het bouwproject. Dit resulteert vaak in onvolledig beheersbare projecten die procesmatig moeilijk te controleren zijn. Daarbij wijken de opgeleverde producten op systeemniveau af van de voorgestelde eisen van opdrachtgevers.

Als mogelijke oplossing voor dit fundamentele probleem, stelde ik in mijn onderzoek voor om deze gebouwen in een modelgebaseerde benadering te definiëren; specificeren; en te verifiëren. Als reactie op dit probleem introduceer ik System Modeling Language (SysML) en Building Information Modeling (BIM) als methodiek, om vervolgens op grond van een prototypisch model de benadering te testen op efficiëntie en effectiviteit.

Momenteel zit ik in de laatste fase van de integratie van deze benadering. Hiervoor ontwikkel ik een prototypisch model als proof of concept. Om van gedachtegoed en inzichten te wisselen, wil ik graag de deskundigheid van een aantal experts raadplegen. Hiervoor introduceer ik een interview die betrekking heeft op de volgende hoofdonderwerpen*: **multidisciplinaire bouwprojecten, functionele eisen en de open-standaard IFC.*** Hiervoor wil ik de deskundigheid van een ***Project Manager en BIM Specialist*** raadplegen. Het interview is als volgt gestructureerd:

Introductie

1. Wat is momenteel uw huidige functie en dagelijks werkzaamheden?

Complexe, multidisciplinaire-bouwprojecten (Integrated AEC-Projects)

Gebouwen worden steeds complexer jegens de eisen die opdrachtgevers/gebruikers stellen aan deze gebouwen. De verwachtingen zijn tegenwoordig hoog. Onderzoekers adviseren dit type gebouwen in de vroege ontwerpstadiums integraal te benaderen.

2. Wat verstaat u onder complexe multidisciplinaire bouwprojecten (AEC projects)? Wat zijn volgens u de fundamentele karakteristieke eigenschappen van dit type projecten?
3. Wat zijn volgens u de mogelijke oorzaken dat dit type projecten moeilijk beheersbaar zijn?
4. In welke mate dragen de door de opdrachtgever geformuleerde functionele eisen aan deze complexiteit?
5. Hoe kunnen expliciet vertaalde functionele eisen contribueren aan een beheersbaar proces en/of project?

Functionele-eisen

6. Wat verstaat u onder functionele eisen? Wat zijn volgens u de karakteristieke eigenschappen van deze eisen?
7. Hoe worden deze eisen momenteel gespecificeerd en geverifieerd in multidisciplinaire bouwprojecten? Wat vindt u van deze huidige werkwijze?
8. Waaruit bestaat de complexiteit van deze eisen tijdens het specificeren en verifiëren? Wat zijn volgens u praktische of methodische benaderingen om dit probleem te tackelen?

De open-standaard IFC

9. Wat verstaat u onder Building Information Modeling (BIM) en zijn openstandaard Industry Foundation Classes (IFC)?
10. In welke mate kunnen BIM en zijn openstandaard IFC bijdragen aan een betere consistentie van functionele eisen tijdens de gehele levenscyclus van het bouwproject?

Interview: 1	**Dhr. Steven Knol**	**BIM manager**	**Royal HaskoningDHV**

1. Wat is momenteel uw huidige functie en dagelijks werkzaamheden?

- BIM Coördinator Structural Design;
- BIM Regisseur in diverse projecten;
- Team Coördinator BIM;
- BIM & Revit ontwikkelingen binnen de afdeling Industry & Building.

2. Wat verstaat u onder complexe multidisciplinaire bouwprojecten (AEC projects)? Wat zijn volgens u de fundamentele karakteristieke eigenschappen van dit type projecten?

Hoge eisen omtrent: prijs, kwaliteit, duurzaamheid, toekomst gericht. Daarbij veel stakeholders, veel specialistische kennis vereist. Bij Royal HaskoningDHV wordt in principe bij elk project rekening gehouden met vier hoofdvragen/aspecten die wij moeten beantwoorden, namelijk:

- Requirements of Stakeholders should be met;
- Enough Added Value;
- Result Future Proof;
- Achieve with minimum Resources and Energy: Sustainability.

3. Wat zijn volgens u de mogelijke oorzaken dat dit type projecten moeilijk beheersbaar zijn?

- Traditionele contractvorming (niet integraal);
- Traditioneel ontwerpproces (wachten op elkaar);
- Veel verschillende stakeholders met specialistische kennis en verschillende manieren van werken;
- Niet alle stakeholders zijn gewend de digitale hulpmiddelen te gebruiken die voor handen zijn.

De Nederlandse bouwwereld is nog best traditioneel, en dat daar een slag ingemaakt moet worden. Verder is het ontwerpproces vaak nog heel traditioneel. Het wachten op elkaar. Constant over de schutting gooien en wachten tot je iets terugkrijgt. Terwijl het veel efficiënter en beter ingericht kan worden.

4. In welke mate dragen de door de opdrachtgever geformuleerde functionele eisen aan deze complexiteit?

Vanuit mijn rol, lastig te zeggen. Omdat ik niet aanwezig ben bij het formuleren en specificeren van de eisen. Maar ik denk dat dit zeker een rol speelt als je op een slimme manier doet, alleen is deze niet zo groot als de in vraag 3 genoemde punten.

5. Hoe kunnen expliciet vertaalde functionele eisen contribueren aan een beheersbaar proces en/of-project?

Het duidelijk specificeren, meetbaar maken en verificatie proces kan een positief effect hebben op de beheersbaarheid van een ontwerpproces. Bv. Onderlinge relaties van eisen, samenwerken stakeholders, betrokkenheid opdrachtgever, beslissingen nemen en vastleggen randvoorwaarden (ruimtes, grids, levels), oplossingsgericht werken, geen grote wijzigingen diep in het ontwerp proces.

6. **Wat verstaat u onder functionele eisen? Wat zijn volgens u de karakteristieke eigenschappen van deze eisen?**
   - Een beschrijving van de prestatie die het product of de dienst moet leveren;
   - Kort en helder moeten zijn;
   - Meetbaar en verifieerbaar;
   - Relatie met andere bovenliggende eisen (boomstructuur);
   - Bijstellen eisen indien nodig.

7. **Hoe worden deze eisen momenteel gespecificeerd en geverifieerd in multidisciplinaire bouwprojecten? Wat vindt u van deze huidige werkwijze?**
   - Dit verschilt per project;
   - Afhankelijk van opdrachtgever, grootte en complexiteit van project, ontwerpteam;
   - In Excel PVE en verificatie ontwerp in Excel (informatie uit ontwerp modellen met as designed prestaties);
   - Specificeren en verifiëren in PVE databases zoals Relatics, Briefbuilder en dRofus met informatie uit ontwerp modellen met as designed prestaties;
   - Soms handmatig;
   - Functionele eisen, prestatie van het product geleverd moet worden. De relaties tussen eisen en als het nodig eisen bijstellen;
   - Meeste projecten Excel, of handmatig. Aan het onderzoek om te kijken hoe dit makkelijk gedaan om het proces te verbeteren.

8. **Waaruit bestaat de complexiteit van deze eisen tijdens het specificeren en verifiëren? Wat zijn volgens u praktische of methodische benaderingen om dit probleem te tackelen?**

   Lastig te zeggen vanuit mijn rol: specificeren ben ik nooit bij betrokken. Bij verificatie:

   - Soms zijn eisen niet helder en meetbaar;
   - Soms wil men eisen in modellen zetten: terwijl dit as designed/ as built prestaties moeten zijn;
   - Veel verschillende methodieken, geen standaard manier van werken.

9. **Wat verstaat u onder Building Information Modeling (BIM) en zijn openstandaard Industry Foundation Classes (IFC)?**
   - Managing information and collaboration throughout the life cycle of a building;
   - It’s not just 3D Models, It’s not just about ICT;
   - It’s about the Life Cycle of a building/ structure;
   - It’s about better, faster, smarter;
   - It’s about single source of information, no double work;
   - It’s about Collaboration and clear agreements;

- Openstandaard Industry Foundation Classes (IFC) is een eindproduct, niet heel dynamisch, maar goed te gebruiken voor shared en publish momenten.

10. In welke mate kunnen BIM en zijn openstandaard IFC bijdragen aan een betere consistentie van functionele eisen tijdens de gehele levenscyclus van het bouwproject?

In grote mate, maar alleen indien integrale contracten, samenwerking. En ontwerp proces en BIM proces op elkaar afgestemd zijn (wie doet wat wanneer en waarom moet continu helder zijn). Ondanks het gebruik van Revit (algemeen tekenprogramma), IFC wordt steeds gebruikt steeds meer leven kijkt omdat tegenwoordig meer naar de life-cycle wordt gekeken. Om bijvoorbeeld IFC models te checken met Solibri Model Checker, tot nu toe het meest toegepast.

Interview: 2	**Dhr. Anton Wubben**	**Senior Project Manager**	**Royal HaskoningDHV**

1. **Wat is momenteel uw huidige functie en dagelijks werkzaamheden?**

   Ik ben momenteel senior project manager binnen Royal HaskoningDHV. En dat doe ik op basis van twee vormen. De eerste vorm is dat ik dichtbij bij de opdrachtgevers zit. Ik neem een beetje hun rol over, maar wel in een heel erg samenspraak probeer ik alles te organiseren wat nodig is om uiteindelijk een gerealiseerde project te krijgen dat voldoet aan alles wat ze willen of inzichten krijgen over wat ze willen. Verder is de tweede vorm meer intern. We hebben in huis alle disciplines zoals constructeurs, bouwfysicus, architecten (meestal extern), en technische disciplines. Uiteindelijk deze specialiseten vormen een groep bij elkaar om uiteindelijk een ontwerp te maken dat voldoet aan de programma van eisen, en ik als manager probeer ik deze te structureren. En dit voornamelijk voor complex publieke gebouwen en natuurlijk binnen tijd en geld.

2. **Wat verstaat u onder complexe multidisciplinaire bouwprojecten (AEC projects)? Wat zijn volgens u de fundamentele karakteristieke eigenschappen van dit type projecten?**

   Een van de fundamentele eigenschappen is dat het multidisciplinair is. De definitie zegt het al. Verder zou ik zeggen dat het een of andere manier al de meningen en alle eisen van alle disciplines bij elkaar moet komen. En niet onbelangrijk, dat je daar niet de middenlijn in kiest, dat iedereen een beetje zijn zin krijgt. Maar gewoon dat het beste uithaalt. Bijvoorbeeld de ene zegt dat een donker ruimte moet komen, en de architect zegt ik wil veel glas. Ik ben ervoor om door te vragen en daarna te wegen. Gewoon proberen alle kennis eruit te halen.

3. **Wat zijn volgens u de mogelijke oorzaken dat dit type projecten moeilijk beheersbaar zijn?**

   Wat je krijgt is, je kan een lijst maken van wat je wilt, maar een onverenigbare lijst en de vraag is hoe kom je tot iets dat wat alles in zich heeft en waar de juiste afwegingen inzitten. En dan heb je nog te maken veel mensen zijn met ze allen aan het ontwerpen zijn en hoe structureer je dat. Daar liggen de uitdagingen. Verder in complexe projecten heb je maken met meerdere opdrachtgevers die verschillende dingen willen. Bijvoorbeeld de ene wil een blauwe auto, de tweede een gele auto, en de derde wil een roze auto. Dat leidt tot een onverenigbare lijst. Hoe zorg je ervoor dat überhaupt uiteindelijk een auto komt. En hoe zorg je ervoor dat er keuzes worden gemaakt wat ook heel complex kan zijn.

4. **In welke mate dragen de door de opdrachtgever geformuleerde functionele eisen aan deze complexiteit?**

   Aanvullend vraag 3, veel opdrachtgevers weten vaak niet wat ze willen. Bijvoorbeeld jouw voorbeeld is een mooie. Een opdrachtgever eist een prettige akoestische ruimte, maar wat is het dan? Diegene was een keer in een zaal en vond de ruimte prettig. Hoe vertaal je dat naar meetbare eisen?

5. **Hoe kunnen expliciet vertaalde functionele eisen contribueren aan een beheersbaar proces en/of-project?**
   - Genoeg doorvragen;
   - De eisen meetbaar maken zodat je het kunt controleren;
   - Genoeg aandacht besteden aan dit proces;
   - Initiatieffase uitbreiden.

6. **Wat verstaat u onder functionele eisen? Wat zijn volgens u de karakteristieke eigenschappen van deze eisen?**

   Een beschrijving van een prestatie. In principe zijn ze meetbaar dingen. Soms zijn deze eisen te subjectief. Ze moeten objectief zijn. Bijvoorbeeld als je schrijft mooi zijn, hoe definieer je mooi? Waardoor je kan het testen.

7. **Hoe worden deze eisen momenteel gespecificeerd en geverifieerd in multidisciplinaire bouwprojecten? Wat vindt u van deze huidige werkwijze?**

   In principe schrijf je een programma van eisen en worden in documenten vastgelegd. Die krijg je van de opdrachtgever. Er zijn altijd wat dingen die onduidelijk zijn. Maar die probeer je altijd met de opdrachtgever af te stemmen en probeer je meetbaar te maken. En als je zover ben weet je waar de ruimte op welke verdiepingen komen, maak je een ruimtestaat, en ga je per ruimte omschrijven waaraan hij moet voldoen, en/of de ruimte voldoet aan bepaalde eis.

8. **Waaruit bestaat de complexiteit van deze eisen tijdens het specificeren en verifiëren? Wat zijn volgens u praktische of methodische benaderingen om dit probleem te tackelen?**

   De hoeveelheid eisen en disciplines en de relaties tussen hen. Bijvoorbeeld een ontwerp van een ziekenhuis waar we nu bezig zijn. Er was een ruimte die werd helemaal van glas en ze wilden eigenlijk dat je van binnenkwam en dat er een open balie was. De architect had het bedacht. Maar die mensen die in de balie konden werken moesten heel geconcentreerd kunnen werken. Dan horen bepaalde geluidseisen erbij. Het ging niet. Dan zeggen wij: we kunnen met deze aanpassingen nog wel oplossen, maar als het open blijft, halen we maximaal dat niveau. Dit is een voorbeeldje waar de eisen niet opgeteld zijn, of zijn niet overzichtelijk genoeg op een rij gezet. Oplossing: praktische methodes of programma's die dit soort complexiteit wel op een rijtje kunt zetten.

9. **Wat verstaat u onder Building Information Modeling (BIM) en zijn openstandaard Industry Foundation Classes (IFC)?**

   Voor mij is het BIM en IFC een compleet model/database waarin alles zit wat je nodig hebt, wat je wilt toetsen of wat je wilt hebben.

10. **In welke mate kunnen BIM en zijn openstandaard IFC bijdragen aan een betere consistentie van functionele eisen tijdens de gehele levenscyclus van het bouwproject?**

    Dat zit in de meetbaar maken van de eisen en van te voren vastleggen waarop je wilt toetsen en zorgen dat het zichtbaar is in het model. Waardoor je heel analytisch ermee kan omgaan. Je kan het zo transparant mogelijk krijgen. Daar ligt de kracht van deze methode. Iedereen begrijpt het in een keer.

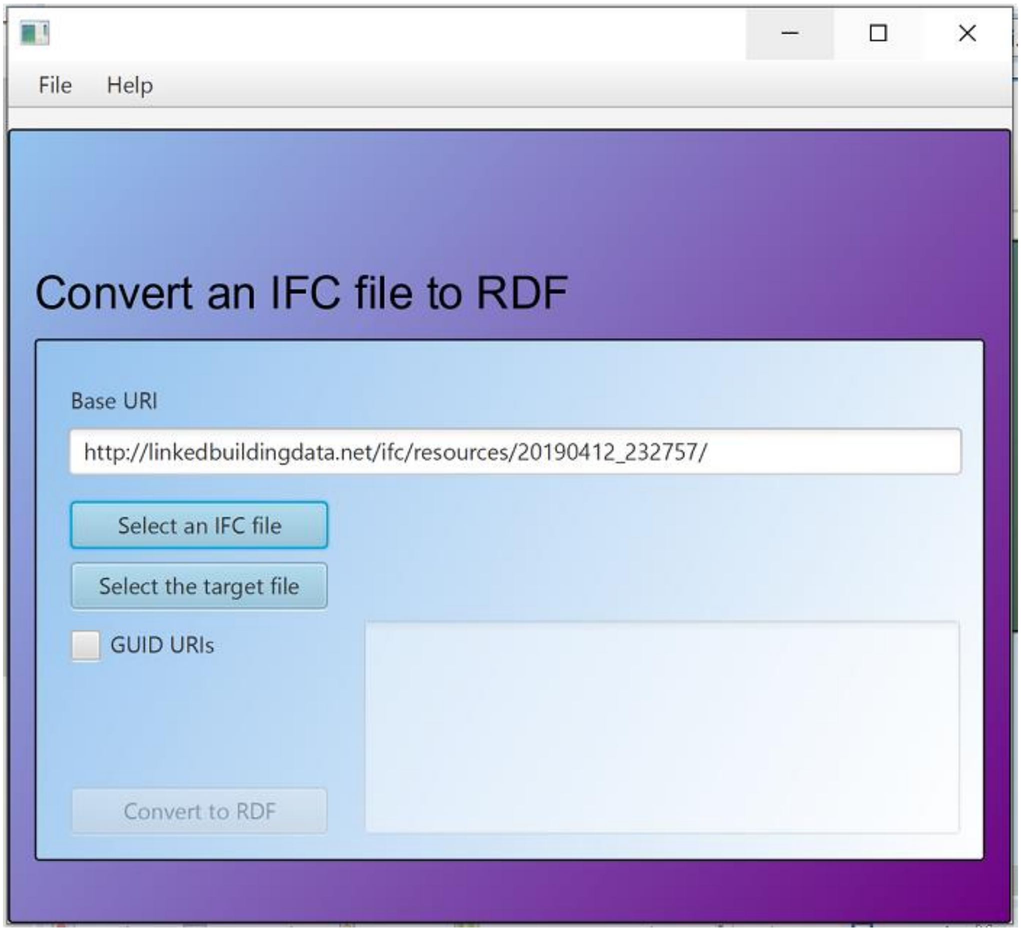
Appendix IV Model Development

PropertySet Name	Pset_DoorCommon
Applicable Entities	IfcDoor
Applicable Type Value	
Definition	Definition from IAI: Properties common to the definition of all occurrences of IfcDoor.

Property Definitions:

Name	Property Type	Data Type	Definition
Reference	IfcPropertySingleValue	IfcIdentifier	Reference ID for this specified type in this project (e.g. type 'A-1')
FireRating	IfcPropertySingleValue	IfcLabel	Fire rating for this object. It is given according to the national fire safety classification.
AcousticRating	IfcPropertySingleValue	IfcLabel	Acoustic rating for this object. It is giving according to the national building code. It indicates the sound transmission resistance of this object by an index ration (instead of providing full sound absorbtion values).
SecurityRating	IfcPropertySingleValue	IfcLabel	Index based rating system indicating security level. It is giving according to the national building code.
IsExternal	IfcPropertySingleValue	IfcBoolean • Default Value: FALSE	Indication whether the element is designed for use in the exterior (TRUE) or not (FALSE). If (TRUE) it is an external element and faces the outside of the building.
Infiltration	IfcPropertySingleValue	IfcVolumetricFlowRateMeasure / VOLUMETRICFLOWRATEUNIT	Infiltration flowrate of outside air for the filler object based on the area of the filler object at a pressure level of 50 Pascals. It shall be used, if the length of all joints is unknown.
ThermalTransmittance	IfcPropertySingleValue	IfcThermalTransmittanceMeasure / THERMALTRANSMITTANCEUNIT	Thermal transmittance coefficient (U-Value) of a material. It applies to the total door construction.
GlazingAreaFraction	IfcPropertySingleValue	IfcPositiveRatioMeasure	Fraction of the glazing area relative to the total area of the filling element. It shall be used, if the glazing area is not given separately for all panels within the filling element.
HandicapAccessible	IfcPropertySingleValue	IfcBoolean	Indication that this object is designed to be accessible by the handicapped. It is giving according to the requirements of the national building code.
FireExit	IfcPropertySingleValue	IfcBoolean	Indication whether this object is designed to serve as an exit in the case of fire (TRUE) or not (FALSE). Here it defines an exit door in accordance to the national building code.
SelfClosing	IfcPropertySingleValue	IfcBoolean	Indication whether this object is designed to close automatically after use (TRUE) or not (FALSE).
SmokeStop	IfcPropertySingleValue	IfcBoolean	Indication whether the object is designed to provide a smoke stop (TRUE) or not (FALSE).

Appendix IV-1. Pset_DoorCommon according to (BuildingSMART, 1996-2006)



Appendix IV-2. IFC to RDF-Desktop Tool (https://github.com/jyrkioraskari/IFCtoRDF-Desktop)

Extracted partial (former) IfcDoor graph – SPARQL Queries

```
PREFIX ifcowl:  <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#>
PREFIX inst:  <http://linkedbuildingdata.net/ifc/resources/20190418_114625/>
PREFIX list:  <https://w3id.org/list#>
PREFIX express:  <https://w3id.org/express#>
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
PREFIX owl:  <http://www.w3.org/2002/07/owl#>

CONSTRUCT {

        ?Door a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcDoor> .
        ?RelDefines <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatedObjects_IfcRelDefines> ?Door .
        ?RelDefines a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcRelDefinesBy Properties> .
        ?RelDefines<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatingPropertyDefinition_IfcRelDefines
        ByProperties> ?PropertySet .
        ?PropertySet a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcPropertySet> .
        ?PropertySet <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#hasProperties_IfcPropertySet>
        ?SingleValue . ?SingleValue a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcProperty
        SingleValue> .
        ?SingleValue <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#name_IfcProperty> ?Name .
        ?Name a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcIdentifier> .
        ?Name <https://w3id.org/express#hasString> ?N .
        ?SingleValue<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#nominalValue_IfcPropertySingleValue>
        ?Value .
        ?Value <https://w3id.org/express#hasString> ?V .

}

WHERE {

        ?Door a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcDoor> .
        ?RelDefines <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatedObjects_IfcRelDefines> ?Door .
        ?RelDefines a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcRelDefinesByProperties> .
        ?RelDefines<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatingPropertyDefinition_IfcRelDefines
        ByProperties> ?PropertySet .
        ?PropertySet a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcPropertySet> .
        ?PropertySet <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#hasProperties_IfcPropertySet>
        ?SingleValue .
        ?SingleValue a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcPropertySingleValue> .
        ?SingleValue <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#name_IfcProperty> ?Name . ?Name a
        <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcIdentifier> .
        ?Name <https://w3id.org/express#hasString> ?N .
        ?SingleValue<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#nominalValue_IfcPropertySingleValue>
        ?Value .
        ?Value <https://w3id.org/express#hasString> ?V .

}

limit 2
```

Extracted partial (new) IfcDoor graph – SPARQL Queries

```
PREFIX ifcowl:  <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#>
PREFIX inst:  <http://linkedbuildingdata.net/ifc/resources/20190418_114625/>
PREFIX list:  <https://w3id.org/list#>
PREFIX express:  <https://w3id.org/express#>
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#>
PREFIX owl:  <http://www.w3.org/2002/07/owl#>

CONSTRUCT {

        ?Door a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcDoor> .
         ?Door <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#name_IfcProperty> ?Name .
        ?Name a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcIdentifier> .
        ?Name <https://w3id.org/express#hasString> ?N .
        ?Door <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#nominalValue_IfcPropertySingleValue>
        ?Value .
        ?Value <https://w3id.org/express#hasString> ?V .

}

WHERE {

        ?Door a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcDoor> .
        ?RelDefines <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatedObjects_IfcRelDefines> ?Door .
        ?RelDefines a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcRelDefinesByProperties> .
        ?RelDefines<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatingPropertyDefinition_IfcRelDefines
        ByProperties> ?PropertySet .
        ?PropertySet a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcPropertySet> .
        ?PropertySet <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#hasProperties_IfcPropertySet>
        ?SingleValue .
        ?SingleValue a <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcPropertySingleValue> .
        ?SingleValue <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#name_IfcProperty> ?Name . ?Name a
        <http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcIdentifier> .
        ?Name <https://w3id.org/express#hasString> ?N .
        ?SingleValue<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#nominalValue_IfcPropertySingleValue>
        ?Value .
        ?Value <https://w3id.org/express#hasString> ?V .

}

limit 2
```

The IfcDoor graph expressed as RDF/XML (input-file)


```xml
<?xml version="1.0"?>

<rdf:RDF
xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
xmlns:express=https://w3id.org/express#
xmlns:owl=http://www.w3.org/2002/07/owl#
xmlns=http://www.w3.org/2005/Atom
xmlns:ifcowl=http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#
xmlns:list=https://w3id.org/list#
xmlns:inst=http://linkedbuildingdata.net/ifc/resources/20190418_114625/
xmlns:xsd="http://www.w3.org/2001/XMLSchema#">

<ifcowl:IfcDoor rdf:about="http://linkedbuildingdata.net/ifc/resources/20190418_114625/IfcDoor_21821">
    <ifcowl:nominalValue_IfcPropertySingleValue>
            <rdf:Description
rdf:about="http://linkedbuildingdata.net/ifc/resources/20190418_114625/IfcLabel_70791">
                        <express:hasString>M_Single-Glass 1:0813 x 2420mm</express:hasString>
            </rdf:Description>
</ifcowl:nominalValue_IfcPropertySingleValue>
<ifcowl:name_IfcProperty>
<ifcowl:IfcIdentifier rdf:about="http://linkedbuildingdata.net/ifc/resources/20190418_114625/IfcIdentifier_39141">
                        <express:hasString>Reference</express:hasString>
</ifcowl:IfcIdentifier>
</ifcowl:name_IfcProperty>
    <ifcowl:nominalValue_IfcPropertySingleValue>
            <rdf:Description
rdf:about="http://linkedbuildingdata.net/ifc/resources/20190418_114625/IfcLabel_47159">
                        <express:hasString>Fire Rating</express:hasString>
            </rdf:Description>
</ifcowl:nominalValue_IfcPropertySingleValue>
<ifcowl:name_IfcProperty>
<ifcowl:IfcIdentifier rdf:about="http://linkedbuildingdata.net/ifc/resources/20190418_114625/IfcIdentifier_46912">
                        <express:hasString>FireRating</express:hasString>
</ifcowl:IfcIdentifier>
</ifcowl:name_IfcProperty>
</ifcowl:IfcDoor>
</rdf:RDF>
```

The XSLT-template based on the XSL-syntax

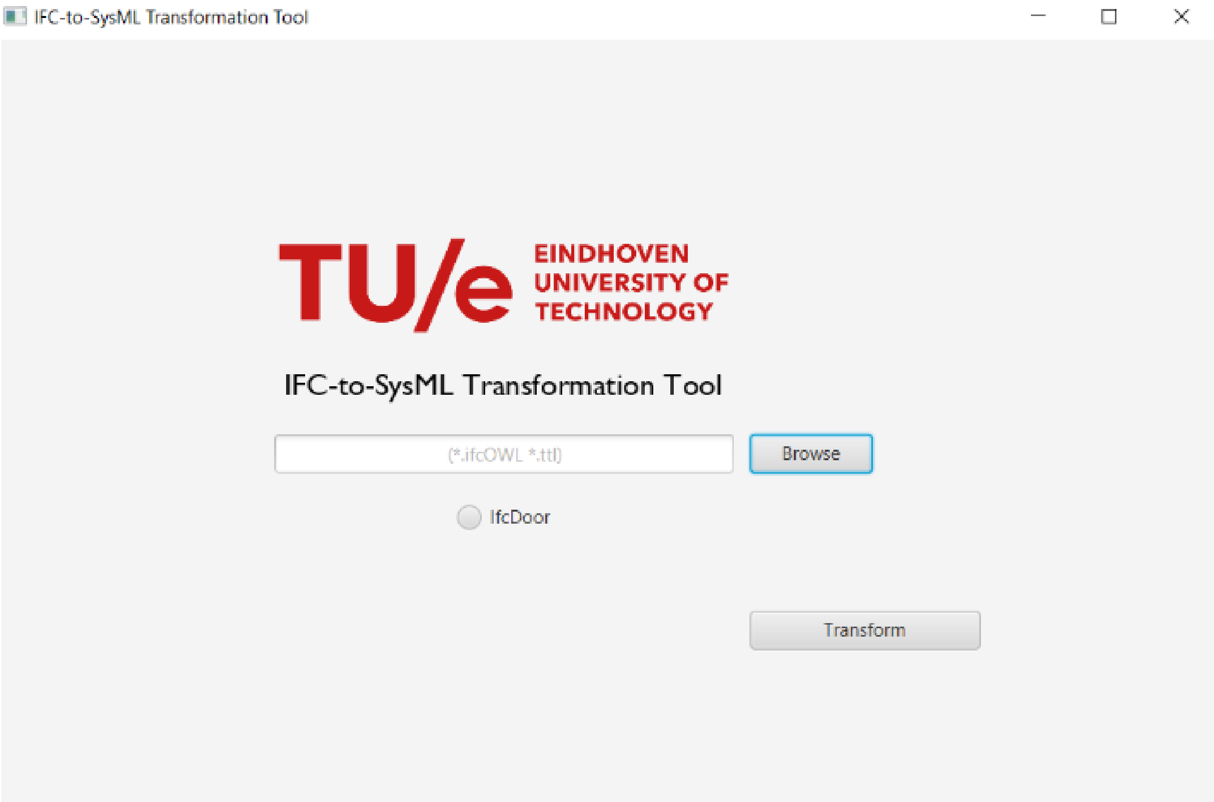
```
<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet version="2.0" xmlns:xsl=http://www.w3.org/1999/XSL/Transform
xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
xmlns:express=https://w3id.org/express#
xmlns:owl=http://www.w3.org/2002/07/owl#
xmlns:ifcowl=http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#
xmlns:list=https://w3id.org/list#
xmlns:inst=http://linkedbuildingdata.net/ifc/resources/20190418_114625/
xmlns:xsd=http://www.w3.org/2001/XMLSchema#
xmlns="http://www.w3.org/2005/Atom">

<xsl:output method= "xml" encoding="UTF-8"/>
<xsl:template match="/">

<xmi:XMI xmlns:xmi="http://www.omg.org/spec/XMI/20131001">
<uml:Package xmlns:uml="http://www.omg.org/spec/UML/20131001" xmi:type="uml:Package"
xmi:id="_18_5_3_429b06e2_1554882936919_674848_14243" name="ifcDoor">

<packagedElement>
<xsl:attribute name="xmi:type">uml:Class</xsl:attribute>
<xsl:attribute name="xmi:id">_18_5_3_429b06e2_1554882987193_743807_14276</xsl:attribute>
<xsl:attribute name="name"><xsl:value-of select="/rdf:RDF/ifcowl:IfcDoor/@rdf:about"/></xsl:attribute>
<ownedAttribute>
<xsl:attribute name="xmi:type">uml:Property</xsl:attribute>
<xsl:attribute name="xmi:id">_18_5_3_429b06e2_1554967218049_369151_14447</xsl:attribute>
<xsl:attribute name="name"><xsl:value-of select=
"/rdf:RDF/ifcowl:IfcDoor/ifcowl:name_IfcProperty[1]/ifcowl:IfcIdentifier/express:hasString/text()"/> </xsl:attribute>
<xsl:attribute name="aggregation">composite</xsl:attribute>
        <defaultValue>
        <xsl:attribute name="xmi:type">uml:LiteralString</xsl:attribute>
        <xsl:attribute name="xmi:id">_18_5_3_429b06e2_1556611363126_30851_14363</xsl:attribute>
        <xsl:attribute name="value"><xsl:value-of select=
"/rdf:RDF/ifcowl:IfcDoor/ifcowl:nominalValue_IfcPropertySingleValue[1]/rdf:Description/express:hasString/text()"/></xsl:attribute>
        </defaultValue>
</ownedAttribute>
<ownedAttribute>
<xsl:attribute name="xmi:type">uml:Property</xsl:attribute>
<xsl:attribute name="xmi:id">_18_5_3_429b06e2_1554967258730_934889_14450</xsl:attribute><xsl:attribute
name="name"><xsl:value-of select=
"/rdf:RDF/ifcowl:IfcDoor/ifcowl:nominalValue_IfcPropertySingleValue[2]/rdf:Description/express:hasString/text()"/> </xsl:attribute>
<xsl:attribute name="aggregation">composite</xsl:attribute>
        <defaultValue>
        <xsl:attribute name="xmi:type">uml:LiteralString</xsl:attribute>
        <xsl:attribute name="xmi:id">_18_5_3_429b06e2_1556611390513_544678_14364</xsl:attribute>
<xsl:attribute name="value"><xsl:value-of select=
"/rdf:RDF/ifcowl:IfcDoor/ifcowl:nominalValue_IfcPropertySingleValue[2]/rdf:Description/express:hasString/text()"/></xsl:attribute>
        </defaultValue>

</ownedAttribute>

</packagedElement>

</uml:Package>

</xmi:XMI>

</xsl:template>

</xsl:stylesheet>
```

The IfcDoor graph expressed as MOF-XMI (output-file)


```
<?xml version="1.0" encoding="UTF-8"?>

<xmi:XMI xmlns:xmi="http://www.omg.org/spec/XMI/20131001"
xmlns="http://www.w3.org/2005/Atom"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:inst="http://linkedbuildingdata.net/ifc/resources/20190418_114625/"
xmlns:list="https://w3id.org/list#"
xmlns:ifcowl="http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:express="https://w3id.org/express#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

<uml:Package xmlns:uml="http://www.omg.org/spec/UML/20131001" name="ifcDoor"
xmi:id="_18_5_3_429b06e2_1554882936919_674848_14243" xmi:type="uml:Package">

<packagedElement xmi:type="uml:Class" xmi:id="_18_5_3_429b06e2_1554882987193_743807_14276"
        name="http://linkedbuildingdata.net/ifc/resources/20190418_114625/IfcDoor_21821">
<ownedAttribute xmi:type="uml:Property" xmi:id="_18_5_3_429b06e2_1554967218049_369151_14447"
        name="Reference" aggregation="composite">
<defaultValue xmi:type="uml:LiteralString" xmi:id="_18_5_3_429b06e2_1556611363126_30851_14363"
        value="M_Single-Glass 1:0813 x 2420mm" />
</ownedAttribute>
<ownedAttribute xmi:type="uml:Property" xmi:id="_18_5_3_429b06e2_1554967258730_934889_14450"
        name="Fire Rating" aggregation="composite"> <defaultValue xmi:type="uml:LiteralString"
        xmi:id="_18_5_3_429b06e2_1556611390513_544678_14364" value="Fire Rating" />
</ownedAttribute>
</packagedElement>
</uml:Package>
</xmi:XMI>
```

Appendix V IFC-to-SysML Transformation Tool implementation

Part 1 Graphical User Interface (GUI) Tool



Appendix V-1. IFC-to-SysML Transformation Tool

package sample;

(Main)

```
package sample;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Main extends Application {

    @Override
    public void start(Stage primaryStage) throws Exception{
        Parent root =
FXMLLoader.load(getClass().getResource("sample.fxml"));
        primaryStage.setTitle("IFC-to-SysML Transformation Tool");
        primaryStage.setScene(new Scene(root, 800, 500));
        primaryStage.show();

    }
```

```
    public static void main(String[] args) {
        launch(args);
    }
}
```

(Module)

```
module Controls {
    requires javafx.fxml;
    requires javafx.controls;
    requires Graphicx;


    opens sample;
}
```

(Control)

```
package sample;

import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.geometry.Pos;
import javafx.scene.control.TextField;
import javafx.scene.layout.GridPane;
import javafx.stage.FileChooser;
import javafx.stage.Stage;

import java.io.File;


public class Controller {



    @FXML
    private GridPane gridPane;

    @FXML
    private TextField textField;


    @FXML
    private void BrowseButton (ActionEvent event) {

        FileChooser fileChooser= new FileChooser();
        fileChooser.getExtensionFilters().addAll(
                new FileChooser.ExtensionFilter("ifcOWL", "*.ttl"));
        fileChooser.setTitle("Open ifcOWL file");

        Stage stage= (Stage) gridPane.getScene().getWindow();

        File file = fileChooser.showOpenDialog(stage);

        textField.setAlignment(Pos.BASELINE_LEFT);
        textField.setText(file.getAbsolutePath());
}

}
```

(Sample)

```
<?import javafx.scene.layout.GridPane?>

<?import javafx.scene.control.Button?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.text.Font?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.text.Text?>


<?import javafx.scene.control.RadioButton?>
<GridPane fx:id="gridPane" fx:controller="sample.Controller"
xmlns:fx="http://javafx.com/fxml" alignment="center" hgap="10" vgap="10"
gridLinesVisible="false" >

        <ImageView fx:id="imageview" fitHeight="300" fitWidth="300"
layoutX="61.0" layoutY="83.0" pickOnBounds="true" preserveRatio="true"
GridPane.rowIndex="1"
            GridPane.columnIndex="2">
            <Image url="https://storage-prtl-
co.imgix.net/endor/organisations/1/logos/1539847178_TUe-logo-descriptor-
line-scarlet-L.png"/>
        </ImageView>

        <Text text="IFC-to-SysML Transformation Tool" GridPane.rowIndex="3"
GridPane.columnIndex="2" GridPane.halignment="CENTER">
            <font>
                <Font name="Gill Sans MT" size="20"/>
            </font>
        </Text>

        <Button text="Browse" GridPane.rowIndex="5"
GridPane.columnIndex="3" prefWidth="80" onAction="#BrowseButton"/>
        <TextField fx:id="textField" promptText="(*.ifcOWL *.ttl)"
GridPane.rowIndex="5" GridPane.columnIndex="2" alignment="CENTER"/>

        <RadioButton text="IfcDoor" GridPane.rowIndex="7"
GridPane.columnIndex="2" GridPane.halignment="CENTER"/>

        <Text text="1" visible="false"/>
        <Text text="2" visible="false"/>
        <Text text="3" visible="false"/>
        <Text text="4" visible="false"/>
        <Text text="5" visible="false"/>
        <Text text="6" visible="false"/>
        <Text text="7" visible="false"/>

        <Button text="Transform" GridPane.rowIndex="12"
GridPane.columnIndex="3" prefWidth="150"/>


</GridPane>
```

Part 2 Extract IfcDoor graph and properties; Create new ifcDoor graph RDF/XML

```
package com.company;


import org.apache.jena.query.*;
import org.apache.jena.rdf.model.Model;
import org.apache.jena.rdf.model.RDFWriter;
import org.apache.jena.util.FileManager;

import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerConfigurationException;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.stream.StreamResult;
import javax.xml.transform.stream.StreamSource;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;


public class Main {

    public static void main(String[] args) {

        //Read ttl file

FileManager.get().addLocatorClassLoader(Main.class.getClassLoader());
        Model model =
FileManager.get().loadModel("c:/users/micha/documents/" +
                "tu eindhoven/duplex woning/O-S1-BWK-BIM bouwkundig.ttl");

        String queryString =
                "PREFIX ifcowl:
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#> \n" +
                "PREFIX inst:
<http://linkedbuildingdata.net/ifc/resources/20190418_114625/> \n" +
                "PREFIX list:  <https://w3id.org/list#> \n" +
                "PREFIX express:  <https://w3id.org/express#> \n" +
                "PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
\n" +
                "PREFIX xsd:  <http://www.w3.org/2001/XMLSchema#> \n" +
                "PREFIX owl:  <http://www.w3.org/2002/07/owl#> \n" +
                "PREFIX : <http://www.w3.org/2005/Atom> \n" +
                "\n" +
                "CONSTRUCT {\n" +
                "   \t?Door a
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcDoor> .\n" +
                "   \t?Door
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#name_IfcProperty
> ?Name .\n" +
                "\t?Name a
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcIdentifier>
.\n" +
                "\t?Name <https://w3id.org/express#hasString> ?N .\n" +
                "\t?Door
```

```
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#nominalValue_Ifc
PropertySingleValue> ?Value .\n" +
                "\t?Value <https://w3id.org/express#hasString> ?V .   \n" +
                "}\n" +
                "\n" +
                "WHERE {\n" +
                " \t\n" +
                "\t?Door a
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcDoor> .\n" +
                "  \t?RelDefines
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatedObjects_I
fcRelDefines> ?Door .\n" +
                "\t?RelDefines a
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcRelDefinesByP
roperties> . \n" +
                "\t?RelDefines
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#relatingProperty
Definition_IfcRelDefinesByProperties> ?PropertySet .\n" +
                "  \t?PropertySet a
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcPropertySet>
.\n" +
                "  \t?PropertySet
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#hasProperties_If
cPropertySet> ?SingleValue . \n" +
                "\t?SingleValue a
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcPropertySingl
eValue> .\n" +
                "  \t?SingleValue
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#name_IfcProperty
> ?Name .\n" +
                "\t?Name a
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#IfcIdentifier>
.\n" +
                "\t?Name <https://w3id.org/express#hasString> ?N .\n" +
                "  \t?SingleValue
<http://standards.buildingsmart.org/IFC/DEV/IFC2x3/TC1/OWL#nominalValue_Ifc
PropertySingleValue> ?Value . \n" +
                "\t?Value <https://w3id.org/express#hasString> ?V . \n" +
                "    \n" +
                " }\n" +
                "\n" +
                "LIMIT 2 OFFSET 5\n" +
                "\n";

        //Query Door and Transform to new Graph
        Query query = QueryFactory.create(queryString);
        QueryExecution qexec = QueryExecutionFactory.create(queryString,
model);
        Model resultModel = qexec.execConstruct();
        qexec.close();



        //Show XML Declaration in output
        RDFWriter writer=resultModel.getWriter("RDF/XML-ABBREV");
        writer.setProperty("showXmlDeclaration", "true");
```

```
        //Print file XML
            FileOutputStream out;
            try {
                out = new FileOutputStream(new File(new
File(System.getProperty("user.home"), "/Desktop/Transformation"),
"Model_Validation.xml"));
          writer.write(resultModel,out,null);
            } catch (FileNotFoundException e) {
                 e.printStackTrace();

                }
```

Part 3 Transform new ifcDoor graph RDF/XML into MOF-XMI

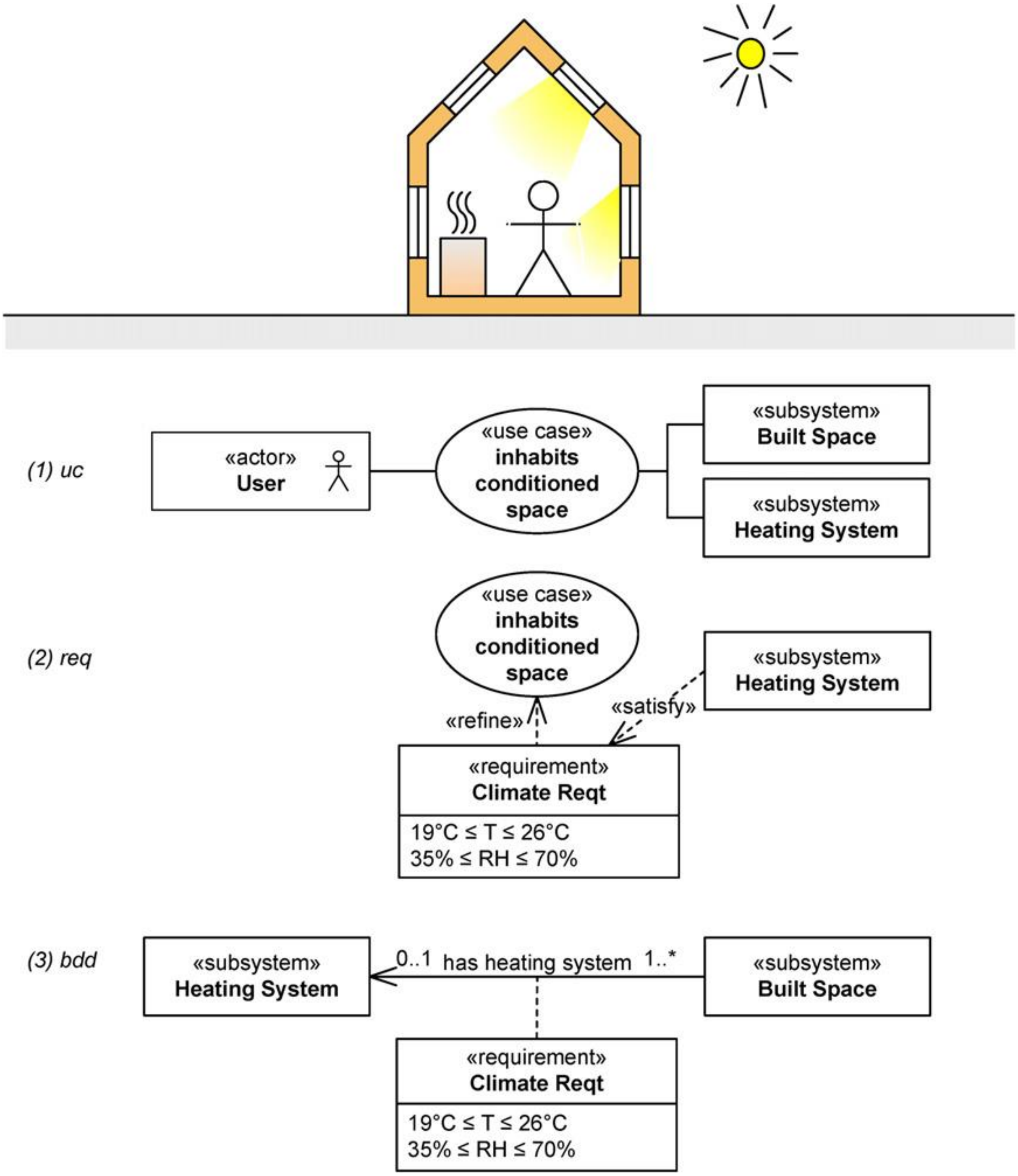
```
package com.company;

import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerConfigurationException;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.stream.StreamResult;
import javax.xml.transform.stream.StreamSource;
import java.io.File;

public class Main {

    public static void main(String[] args) {
       String XSLFile= "c:/users/micha/desktop/transformation/IfcDoor.xsl";
       String INFIle=
"c:/users/micha/desktop/transformation/Model_Validation.xml";
       String
OutFile="c:/users/micha/desktop/transformation/IfcDoor_MOF.cmof";

        StreamSource xslcode= new StreamSource(new File(XSLFile));
        StreamSource input= new StreamSource(new File(INFIle));
        StreamResult output= new StreamResult(new File(OutFile));


        TransformerFactory tf= TransformerFactory.newInstance();

        Transformer trans;
        try{

            trans= tf.newTransformer(xslcode);
            trans.transform(input, output);

        } catch (TransformerConfigurationException e) {
            e.printStackTrace();

        } catch (TransformerException e) {
            e.printStackTrace();
        }
```

Appendix VI Model Validation

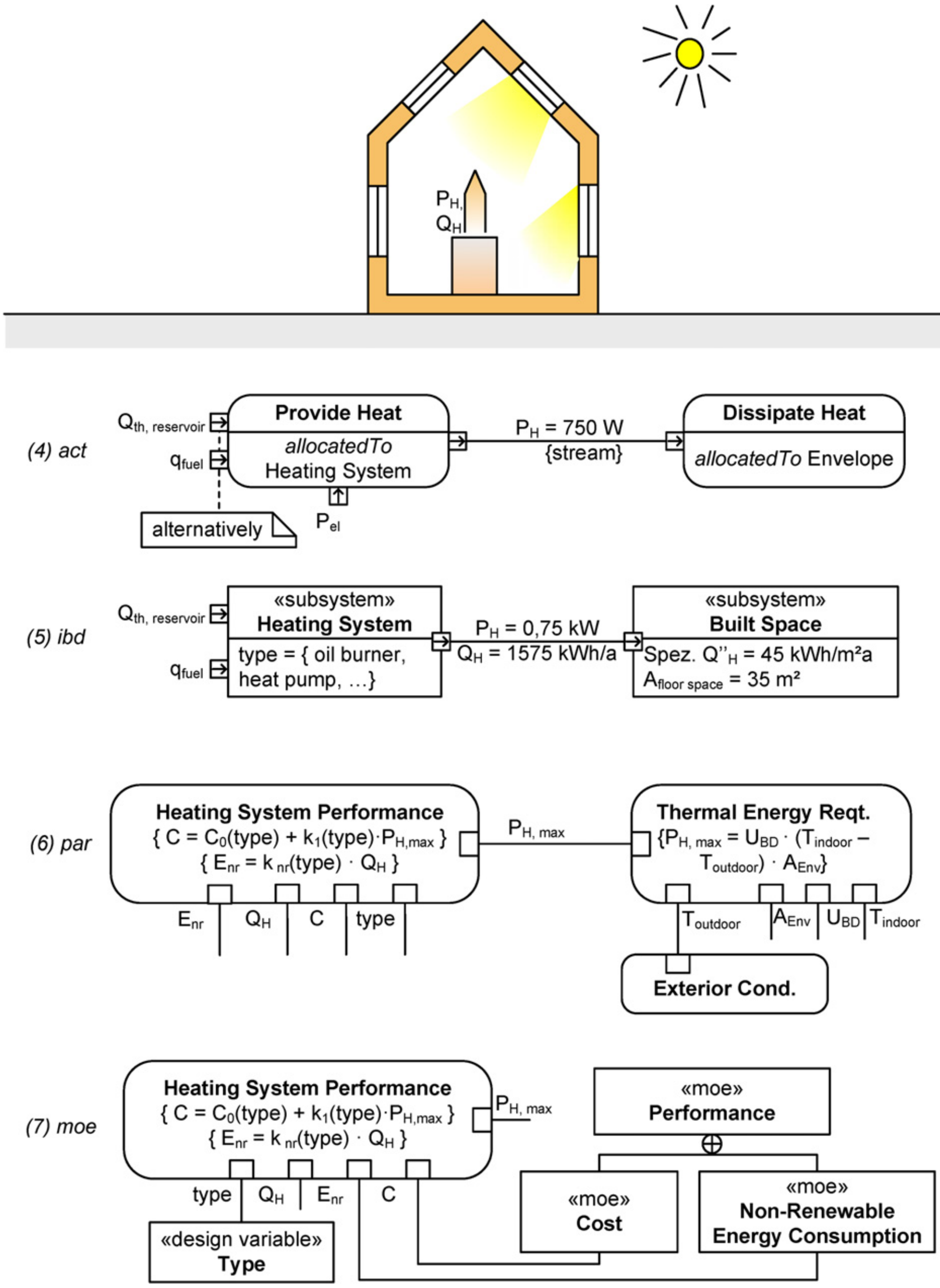
Parametric System Modeling (PSM)

Geyer (2012) developed the PSM method based on SysML in order to model a system for sustainable building design. His approach includes the following steps:

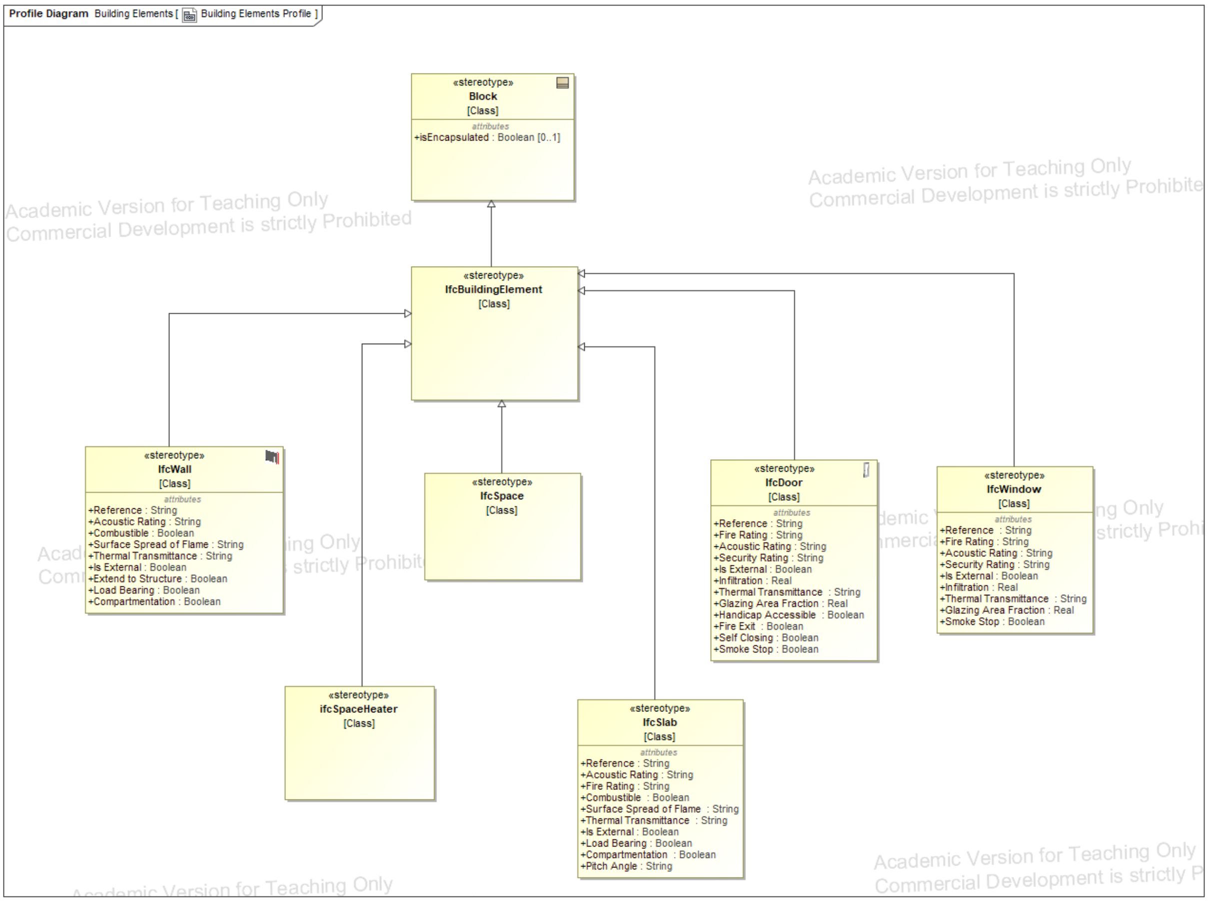
1. Use Case Diagrams (uc) describe the systems's context. They usually handle external prerequisities for a building that are not controllable in the design process.
2. Requirements Diagrams (req) derive requirements fom use cases and refine the design prerequisities. They can assign subsystems or system parts that directly satisfy requirements.



3. Block Definition Diagrams (bdd) define the system structure. They incude requirements as association blocks to enable a requirement-driven design process.
4. Activity Diagrams (act) model the processes occuring in components and the items exchanged between them. Allocating processes to system parts is possible. Activity diagrams can also describe design processes.
5. Internal Block Diagram (ibd) describe the item flows between blocks. They include current flows and the sums of flows. Theese diagrams provide, along with activity diagrams, e.g., energt flows in the building.
6. Parametric Diagrams (par) capture calculations for analysis and represent analytically defined models, e.g., for dimensioning. The contraint blocks (rounded boxes) may also embed suboridinate simulations.
7. Measures of effectives (moe) exten parametric diagrams for evaluation purposes and provde importants tools for performance-oriented design. Design variables mark parameters for design variation.



<u>The profile diagram defining some concepts originated from BIM-IFC standard</u>



(Beetz, Leeuwen, & Vries de, 2007)

(Construction User Roundtable, 2004)

(Nassar & Austin, 2013) (Gaševic, Djuric, & Devedžic, 2006)