Combining Model-Based Systems Engineering and Product Line Engineering to support variability management in the construction industry

A CASE STUDY AT PLEGT-VOS

September 8, 2021

Name	Ir. M.A.J. Velthuizen
Student number	S1437380
Study	Business Administration
Supervisor University of Twente	Dr. Ir. E. Hofman
Supervisor University of Twente	Prof. Dr. M.E. Iacob

Contact information
University of Twente
Faculty Engineering Technology
Postbus 217
7500 AE, Enschede
(+31) 534899111
Info@utwente.nl

Summary

In the next few decades, to meet the Paris agreement the Dutch government aims to reduce the environmental impact of the build environment (United Nations, 2015). To achieve this, many houses and buildings have to be updated and/or renovated with new technologies that reduce the energy usage. The updating and renovating of existing constructions give rise to multiple challenges for the construction industry, since each construction is shape and sized differently, many different technologies exist that can be integrated, and the construction periods have to be short.

Variety management strategies, Model-Based Systems Engineering (MBSE), and Product Line Engineering (PLE) can be implemented to overcome these challenges. However, these methodologies are not yet implemented in the AEC industry. Besides, no approach has been formulated that combines these methodologies for implementation in the system development approach. Therefore, this research answer the following three research questions:

1. *"How to combine Model-Based Systems Engineering (MBSE) with variety management strategies and Product Line Engineering (PLE) to support variety of product lines?"*

2. *"How to integrate Model-Based Systems Engineering (MBSE) to support variety management of the product lines of a company in the AEC industry?"*

3. *"What are the benefits of implementing MBSE compared to current practices in the AEC industry?"*

In this research a framework is developed that combines modularity, product platforms and product families, MBSE and PLE. The framework can be used as guidance to implement these strategies in a practical context of the AEC industry. It includes multiple decision and execution steps, necessary to develop a MBSE and variability model for product lines of an organisation. Additionally, the usage of MBSE is compared to the use of Building Information Modelling (BIM).

Additionally, the framework is applied in a practical context of the AEC industry by means of a case study. The case study organisation is a Dutch construction industry, which already makes use of modularity and product platforms. Therefore, the framework is used only to implement PLE and MBSE and combine it in a MBSE and variability model. Application of the framework at the case study organisation, showed that the steps guide the process for implementation smoothly. However, a few additional decision and execution steps have been included in the case study to develop the MBSE and variability model. Therefore, the framework has been updated to include those steps.

Next, the use of the final variability system model is compared to the current approach at the case study organisation, namely BIM. An experiment is conducted to test for difference in the system traceability, requirement traceability, and information storage. The experimental results show that MBSE is significantly more effective and efficient for requirement traceability. Additionally, the two approaches are equally effective and efficient for system traceability, when the experience level of the user is high. However, a low experience with MBSE has a negative influence on effective and efficient system traceability. Finally, no significant difference were found for the information storage between the two approaches.

To conclude, the developed framework is sufficient to guide the implementation of modularity, product platforms and product families, PLE and MBSE in the AEC industry. Additionally, the use of MBSE positively influences requirement traceability compared to BIM, which means it supports verification and validation of the final system design.

Contents

List of Figures

List of Tables

List of Abbreviations

ADL	Architecture Description Language
AEC	Architecture, Engineering and Construction
AOL	Aspect-Oriented Language
bdd	Block definition diagram
BIM	Building Information Modelling
CSM	Cameo Systems Modeler
DSL	Domain-Specific Language
EFFBD	Enhanced Function Flow Block Diagram
ibd	Internal block diagram
MBSE	Model-Based Systems engineering
OMG	Object Management Group
OOSEM	Object-Oriented Systems Engineering Method
OPD	Objective Process Diagrams
OPL	Objective Process Language
OPM	Object Process Methodology
OVM	Orthogonal Variety Model
PLE	Product Line Engineering
ROI	Return on Investment
RUP SE	Rational Unified Process for Systems Engineering
SDL	System Definition Language
SE	Systems engineering
UML	Unified Modelling Language
2GPLE	Second Generation Product Line Engineering

1 Introduction

In 2015 the Paris Climate Change Conference has been organised. During the conference the Paris agreement has been formulated, which aims at limiting climate change in the next decades. The main objective of the agreement is limiting the global temperature increase to maximum 2 degrees Celsius in the time period from 202 to 2050 (United Nations, 2015). The agreement has been signed by 195 countries, which individually will take actions to limit climate change (Nederlandse Emissieautoriteit, 2015).

To achieve the main objective of the Paris Agreement, the Netherlands divides its focus over twelve areas to meet the objectives set for the country. One focus area is the built environment, which takes up 1/3 of the total energy usage in this country. Therefore, the government aims at achieving an energy neutral built environment by 2050 (Sociaal-Economische Raad, 2020). New housing and construction projects already have to meet multiple requirements to reduce the energy usage of the final constructions. However, construction of new energy neutral housing is not enough. The houses and buildings constructed in the last centuries have to be renovated and updated to meet the energy reduction objectives. The Dutch government aims at renovating and adapting 7 million houses and 1 million constructions to make them energy neutral by 2050 (Klimaatakkoord, 2020). To be able to meet these goals by 2050, each year approximately 365.000 existing buildings have to be turned into energy neutral buildings (TNO, 2016). In the period of 2008 to 2017, 800.000 existing buildings were updated with at least one energy-saving product. Of these, 200.000 existing buildings have been updated with minimally two energy-saving products. However, this is not enough to make the built environment energy neutral by 2050 (Manshanden & Koops, 2019).

Besides, renovating and improving the energy usage of the built environment gives rise to multiple challenges for the construction industry. First, each building is different in size and shape, which means that the techniques and technologies have to be adapted to each individual situation. Second, many technologies and methods exist to make houses and buildings energy neutral, for example solar panels, insulation, or heat pumps. It should be possible to install these technologies in each house or construction building and combine them for the most optimal solution. Finally, to meet the main objective of 2050, approximately 400.000 houses and buildings have to be updated every year. Therefore, it should be possible to install the techniques and technologies for energy saving in a short period of time. Overall, a high variety is required, which has to be developed in a short time period.

Companies in the construction industry have to find solutions to deal with these challenges. Variability management strategies, Model-Based Systems Engineering (MBSE) and Product Line Engineering (PLE) are commonly used in the product development processes in the high-tech industry to shorten development times. These strategies focus on standardisation and reuse of designs and/or parts of system, while offering a variety in design and managing the complexity of the systems. However, this is not common practice within the Architecture, Engineering, and Construction (AEC) industry. Integrating variety management, MBSE and PLE in the AEC industry could shorten development times, while offering a high variety.

Variety management focuses on finding a balance between standardisation/commonality and differentiation of products offered within the product portfolio of a company (ElMaraghy et al., 2013). Additionally, PLE focuses on strategic management of products in a product line of an organisation by taking advantages of the commonalities, while respecting the differences (Lauenroth & Pohl, 2005). A variety management strategy that could be implemented is modularisation. Modularisation of a system splits the system into multiple smaller subsystems. The subsystems can be designed separately, however when combined it results in a fully functioning system (Baldwin & Clark, 2003). With a modular product structure each subsystem can easily be adapted to the changing requirements and the different needs for the final product. Therefore, the system can be adapted to fit to every individual building. Additionally, the system can be developed more rapidly compared to non modular structures (Baldwin & Clark, 2003; Aleksandraviciene & Morkevicius, 2018). Overall, modularisation enables the possibility to increase standardisation within the AEC industry, while still offering a high variety of products in an efficient way.

Even though, modularity decreases the complexity of a system (Baldwin & Clark, 2003; Sanchez & Mahoney, 1996), further reduction and efficient management of the complexity is necessary. A strategy that focuses on these aspects is MBSE. It is used throughout the whole life-cycle of a system and each of the life-cycle phases are taken into account at the start of a development process. MBSE uses one central system model that includes information about the behaviour, structure, requirement, parameters and input from all stakeholders. All information in the central system model is derived from stakeholder input and/or linked to other information in the system, which results in a consistent information model of the system under development. Additionally, the system is flexible to design and/or requirement changes due to the strong linkage and traceability of information in the system model (Friedenthal et al., 2015d), which reduces the risks related to the development of a new system (INCOSE, 2012). Consequently, MBSE effectively supports management of the difference in the requirements and design of each individual house and building and supports the complexity reduction.

Although, long lists of benefits can be found for the implementation of MBSE in academic literature (Friedenthal et al., 2015e; Madni & Sievers, 2018), limited empirical evidence exist that support these claims (Henderson & Salado, 2021). To reassure that implementation of MBSE will improve the development process of systems in the AEC industry, the method is compared to the current development approach. An experiment will be conducted to find empirical evidence to show the difference between the use of MBSE and the current AEC industry approach.

To summarise, the research objective is to formulate, validate and verify an approach to combine variety management strategies, PLE and MBSE to support variability management of product lines in the AEC industry. Therefore, the following research questions are formulated:

1. *"How to combine Model-Based Systems Engineering (MBSE) with variety management strategies and Product Line Engineering (PLE) to support variety of product lines?"*

2. *"How to integrate Model-Based Systems Engineering (MBSE) to support variety management of the product lines of a company in the AEC industry?"*

3. *"What are the benefits of implementing MBSE compared to current practices in the AEC industry?"*

To help answering the three central research questions, multiple sub-research question are formulated that guide the research. The key concepts in the first research question are variability management, product lines, and model-based systems engineering. Therefore, sub-research questions are formulated for each key concept in the first central question. Four sub-questions related to variety management are the following:

1.1 *How is variety management defined?*

1.2 *What variety management strategies exist related to the design and planning of a system?*

1.3 *What are the benefits of the use these variety management strategies?*

1.4 *Which of these variety management strategies are used within the AEC industry?*

Three sub-research questions have been formulated related to product lines and product line engineering. The sub-questions are the following:

2.1 *How is product line engineering defined?*

2.2 *What are the benefits of the implementation of product line engineering?*

2.3 *How is variability of product lines modelled within product line engineering?*

Four sub-research question have been formulated related to systems engineering and model-based systems engineering. The sub-questions are the following:

3.1 *How is systems engineering defined?*

3.2 *How is model-based systems engineering defined?*

3.3 *What are the differences between systems engineering and model-based systems engineering?*

3.4 *What are the benefits of model-based systems engineering?*

3.5 *How does model-based systems engineering compare to commonly used development approaches in the AEC industry?*

A research framework is developed that shows the steps that are taken to answer the central and sub-research question (see figure 1). In part A of the research, relevant literature is studied related to variety management, PLE, SE and MBSE. In step A.1 and A.2, variety management strategies are researched, namely modularisation, modularisation strategies, product platforms and product families. Thereafter, in step A.3, product line engineering literature is explored. Next, in step A.4, literature of systems engineering and model-based systems engineering is analysed (chapter 3).

In part B of the research, a framework is developed to combine MBSE with variety management strategies and implement it in the development of product lines. The developed framework is based on the findings of the literature review of part A (chapter 4). Thereafter, in part C of the research, the framework is applied in a case study at an organisation within the AEC industry, which results in a system model of a product line (chapter 5). The application of the framework is analysed and recommendations for improvements will be done based on the process. Next, in part D, the system model is compared to the current process used at the case study organisation by means of an experiment 6. Thereafter, the results are analysed and interpreted (chapter 7). Finally, the research finishes with a discussion and conclusion about the proposed framework, the application and the experimental outcome (chapter 8).

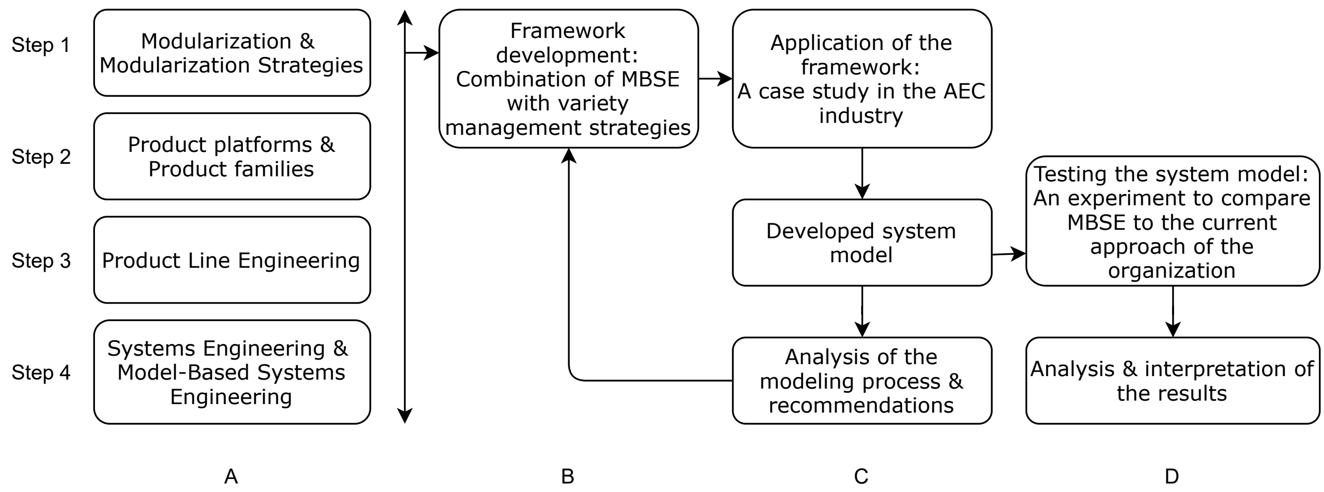


Figure 1 – Research Framework

In this research a framework is developed that shows how to combine and integrate MBSE, PLE and variety management strategies in a development process of a system. The use of the framework should results in one central model, which stores all relevant information regarding the requirements, behaviour, structure, and parametrics of a product line. Additionally, the model should contain the variability of a product line, should be able to configure a single product variant and has to be flexible to requirement and design changes. The case study validates the use the framework and shows how to implement the MBSE, PLE and variety management strategies in the AEC industry. Finally, an experiment is conducted to empirically show the effects of the use of MBSE compared to current practices in the AEC industry. Overall, the research contributes to AEC, MBSE, PLE, and variety management research areas.

2 Research methodology

The chosen methodology for the execution of this research was the Design Science Research (DSR) approach. The DSR approach aims at improving a problematic situation in a practical context by design, development and implementation of a technical artefact, while at the same time gaining and gathering knowledge. Therefore, the two purposes make it possible to directly link theory to practice and study the influence of the theory in a practical context (Hevner et al., 2004; Wieringa, 2014). Furthermore, a technical artefact can be a model, construct, method, or instantiation (Hevner et al., 2004; Wieringa, 2014; Peffers et al., 2007). For this reason, DSR is a suitable research methodology to study how to combine, how to integrate and to verify MBSE with variety management strategies for product lines in the AEC industry.

DSR uses a cyclical approach with 6 phases, namely a problem definition phase, an objective formulation phase, a design and development phase, an demonstration phase, an analysis and evaluation phase, and finally a learning and communication phase. In the problem definition phase the problematic situation is studied and outlined. Furthermore, the importance of a solution is highlighted. In the objective formulation phase, implicit or explicit requirements are formulated for the to-be-designed technical artefact. The objectives are derived from the defined problem to reassure the artefact will solve the problem. Next, in the design and development phase a technical artefact is designed that meets the objectives of the design. Thereafter, in the demonstration phase the artefact is implemented in a practical context. After which, the results are studied in the analysis and evaluation phase. Finally, in the learning and communication phase the problem, objective, results and gained knowledge is shared with the relevant stakeholders. The results and gained knowledge could initiate a new research cycle (Peffers et al., 2007; Wieringa, 2014).

Every research methodology has advantages and disadvantages. One often mentioned problem of DSR, is the knowledge and theory contribution of the research. For this reason, Hevner et al. (2004) defined seven guidelines that have to be taken into account during execution and communication of DSR to reassure quality. Those seven guidelines are the following:

- **Design as an artefact**. The research has to result in a purposeful technical artefact. The artefact has to be described clearly, such that implementation by others is possible.
- **Problem relevance**. The technical artefact has to be designed and implemented to improve an unsolved and important business problem.
- **Design evaluation**. A well-executed evaluation method has to be adopted to show the use and quality of the artefact.
- **Research contribution**. The results of the research have to contribute to the artefact design area, the artefact development process area, and/or the artefact evaluation area.
- **Research rigour**. Research rigour is necessary for the execution and evaluation of the research. For the execution of the research appropriate techniques have to be chosen for construct and theory building. Furthermore, appropriate techniques have to be selected for justification and evaluation of the theories and designed artefact.
- **Design as a search process**. The research has to be conducted as an iterative process to search for the most optimal solution for the problem.
- **Communication of research**. The results and gained knowledge has to be distributed and communicated to direct stakeholders of the research. Additionally, the research contribution has to be communicated to external technology professionals, management, and researchers.

2.1 Design science research specifications

In the previous chapter the research problem and objective has been outlined. The next step is to develop an approach that combines MBSE with variety management strategies that can be applied in a practical context in the AEC industry. The purpose of the approach should be to improve the variety management processes and tasks within the AEC industry. Therefore, the DSR approach is a suitable research methodology to develop a technical artefact that improves variety management practices within the AEC industry (Peffers et al., 2007; Hevner et al., 2004; Wieringa, 2014).

The research has to comply with the seven guidelines of Hevner et al. (2004). First, the technical artefact designed is a central model that combines variety management, MBSE, and PLE by means of a framework for the selection of methodologies, strategies, and tools for combining variety management, MBSE and PLE. Second, the technical artefact was developed to contribute to shorten the development time, while enabling a high variety of systems in a product line within the AEC industry to be able to meet the Dutch 2050 goals for the build environment. Third, the design was evaluated by applying it in a practical context in the AEC industry and by conducting an experiment to show the effects of implementing MBSE, PLE and variety management practices. Fourth, the research contributes to the artefact design area and artefact development process area. The developed framework contributes to the development process area, while the results of applying the framework contributes to the design area of the artefact.

Fifth, a case study was selected for the execution of the research to implement the central model in a practical context. A product line of the case study organisation was modelled in the central model to show the functionality and utility of the methodology. Additionally, an experiment was conducted for the evaluation of application of MBSE compared to the current approach in the AEC industry. Sixth, the research was conducted as a search process. A variation of multiple strategies, methodologies and tools have been studied for each key concept of the research. Next, the framework was developed to show which decisions have to be made regarding selection of the strategies, methodologies and tools. Thereafter, decisions were made based on the framework and the formulated requirements for the artefact to find the optimal solution. Finally, the results have been communicated with the research supervisors of the university of Twente and the innovation manager of the case company.

The research started with the problem definition phase in which the problematic situation is formulated. The problematic situation has been outlined in the introduction (chapter 1). Thereafter, the objective was formulated for a solution of the problem. The objectives were derived from the problem definition and available knowledge (see chapter 1). In the design and development phase a literature review was conducted (chapter 3). Based on the reviewed literature a framework was developed that combines MBSE with variety management strategies (chapter 4). The framework has been used as input for the demonstration phase. In the demonstration phase the framework was applied to a case study within the AEC industry. The case study is documented in chapter 5. Besides, an experiment is conducted to show the effects of implementing MBSE compared to the current development approach. In the analysis and evaluation phase the application of the framework in the case and the experiment results was discussed. Finally, in the learning and communication phase the findings were critically studied and suggestions for improvement are done.

2.2 Case study selection

For the demonstration phase of the design research, a case study was conducted. To effectively show the usage and impact of the developed framework, a suitable case had to be selected. A case can be selected by means of seven different strategies, namely the most similar cases, most different cases, influential, deviant, extreme, diverse or typical (Seawright & Gerring, 2008). The most similar, most different and most divers strategies uses two case subjects to show the similarity, difference or diversity. Furthermore, for the influential case a subject and/or context is chosen based on a perfect fit to the model studied. Additionally, the extreme case strategy makes use of an extreme situation and/or subject compared to other situations and/or subjects. A deviant case study strategy selects cases based on surprising values for the to be studied context and/or subject. Finally, for the typical case situations and/or subjects are selected that present a normal industry or population (Seawright & Gerring, 2008).

The case was selected based on the fit to the research objective to formulate, validate and verify an approach to combine variety management strategies, PLE and MBSE to support variability man-

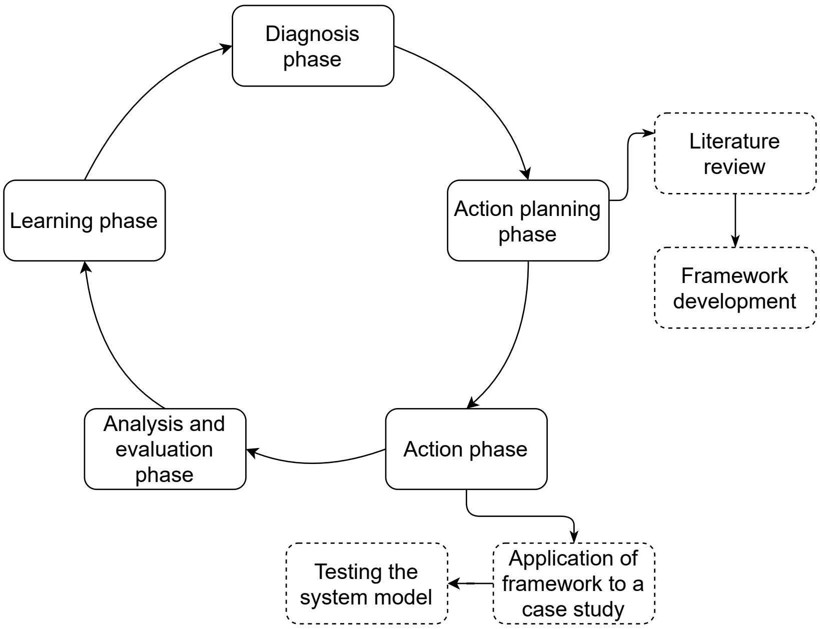


Figure 2 – Overview of the research methodology

agement of product lines in the AEC industry (e.i. influential case). To formulate, validate and verify the implementation of MBSE and PLE it would be helpful if the case organisation would already make use of variety management strategies in their system design, product lines and/or product portfolio. The focus of the case study could than be limited to the implementation of MBSE and PLE and would not be influenced by determination the commonalities and variability in the products of an organisation. For this reason, the selected case organisation is Plegt-Vos, which is a Dutch construction company that uses modularity and platforms in their system design. Additionally, Plegt-Vos has already determined the commonalities and variability in their product lines to meet a wide variety of customer demands.

3 Theoretical framework

The research framework (figure 1) shows that the research starts with a literature review of multiple topics, which is outlined in this chapter. The first topic is variety management. Second, modularity and modularisation strategies are examined. Third, product architecture, product families, and product platforms are studied. Furthermore, product line engineering (PLE) literature is examined. Finally, literature about systems engineering (SE) and model-based systems engineering (MBSE) is reviewed.

3.1 Variety management

In the AEC industry, design and solutions are often formulated for a single project and/or costumer. However, designing individual products can be costly and time consuming. From project to project the designed systems have a high degree of differentiation, while an organisation could benefit from increasing the commonalities. Therefore, the AEC industry can reap the benefits of offering more standardised systems. Standardisation, in the form of modules, could save up to 20% to 50% on the time schedule and 20% on the costs (Bertram et al., 2019). On the other hand, switching to standardised systems in the AEC industry calls for variety of the offered products to meet the individual needs.

Therefore, variety management has to be implemented. Variant management aims at finding a perfect balance between commonalities and differentiation within a product portfolio of an organisation. Variant management uses multiple methods to determine the amount of variants that have to be/can be offered to meet needs of a large share of the market, while the offered systems can be designed, manufactured, used, serviced, and retired efficiently. The methods support variety management of the offered systems and increase productivity of AEC projects (ElMaraghy et al., 2013).

ElMaraghy et al. (2013) have classified the strategies that are part of variety management based on the main activities and on the granularity of the market. The three main activities within projects are the design, the planning and the manufacturing of the system. Besides the granularity of the market is divided into four categories, namely the parts, the products, the enterprise and the market. See figure 3.

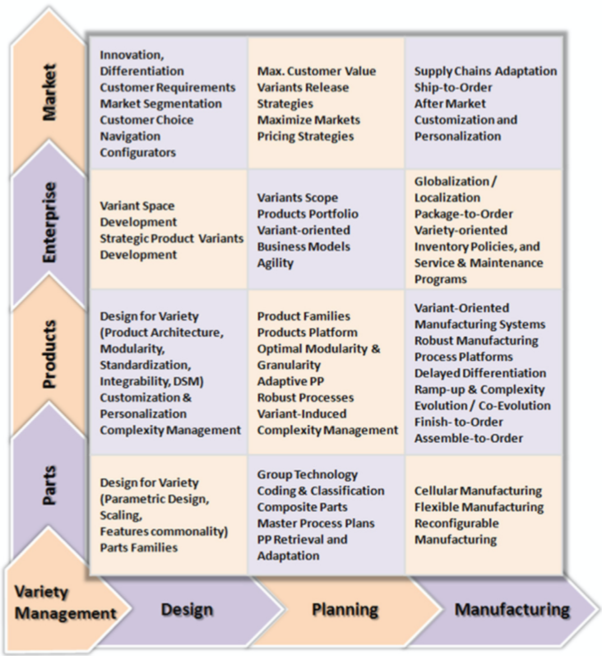


Figure 3 – Overview of variant management strategies from ElMaraghy et al. (2013)

As stated in the introduction, construction companies have to be able to offer a high variety of products due to different shapes and sizes of houses and buildings, and due to the different methodologies that can be installed to achieve energy neutral buildings. These two challenges are directly related to the design and planning of the system. Figure 3, shows multiple variant management strategies related to product design and planning. The first strategy mentioned for product design is Design for Variety (DFV). Enablers for the use of DFV are modularity and integration, product families, product platforms and product portfolios. These enablers are also variant management strategies that can be used for product planning (see figure 3) (ElMaraghy et al., 2013). For this reason, the topics included in the literature review are modularity, modularisation strategies, product platforms, and product families.

Throughout this chapter a single example is used to give a clear picture of the before mentioned topics of the literature review. The research has a focus on the design and planning of a technical system, which is in this case a system within the AEC industry. For the example, a simplified residence is chosen, which has an obvious use and function. The author assumes that most readers are familiar with the global use, functionalities, and aesthetics of the system (see figure 4).

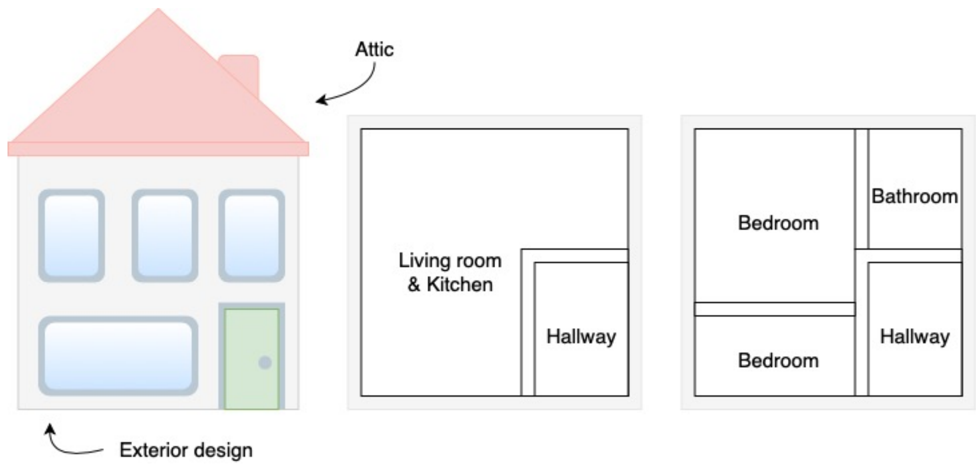


Figure 4 – Residence

The main function of a residence is to provide a safe and comfortable living environment for the users. The residents of the residence wish a place to relax and sit down, a place to prepare and store food, multiple places to sleep, and a place for personal hygiene. Additionally, the residence requires a subsystem that connects the multiple different functional spaces and different floors. Finally, the residence needs to protect against weather and intruders. From this use and functionality it can be concluded that the residence requires the following subsystems:

- Hallway
- Living room
- Kitchen
- Master bedroom
- Second bedroom
- Bathroom
- Attic
- Exterior

3.2 Modularity

In academic research many authors have formulated a definition for modularity. However, there is not a single generally accepted definition. The definitions vary between product specific and domain specific or general and abstract. For this reason, it is important to define what is meant with modularity in this research. Modularity in this research is defined as: a strategy to divide systems into multiple subsystems that have a high independence, however when put together function as a whole system (Ulrich, 1994; Sanchez & Mahoney, 1996; Schilling, 2000; Piran et al., 2016). A subsystem is referred to as a module, which each deliver specified functions in the overall system. Dividing systems and/or products into modules means that each module can be designed and/or developed independently from other modules without influencing their functionality and/or designs. However, clear rules have to be set up to reassure that all subsystems will fit and function together in the final design (Baldwin & Clark, 2003).

The rules are related to the systems architecture and the module interfaces. A systems architecture defines which modules are or can be included in the whole system and it shows their physical arrangements and relationships. Additionally, in the architecture it is specified which functions are or have to be delivered by specific modules (Ulrich, 1994; Baldwin & Clark, 2003; Piran et al., 2016). Furthermore, the module interfaces define how the subsystems fit together physically, how they are connected, and how they communicate. The interface design rules reassure that the subsystems can be combined to form the final system.

Furthermore, choices have to be made regarding the definition of the modules. Modules should contain components, parts and/or subsystems that have a high interaction with one another. The interactions can be information flow, energy transition, material input, physical input or physical connections (Gershenson et al., 2003). Therefore, a systems has to be analysed on the internal interactions and cluster parts, components, and subsystems based on these interactions. The next step is to define the modules accordingly. Ideally a module has a high internal interaction and a low external interaction. Especially the incidental interactions should be minimal (Ulrich, 1994; Browning, 2001).

Modularisation was initially applied to products and software systems (Parnas, 1972). However, over the years the use of modularisation grew throughout multiple areas within organisations, namely production processes, organisational structures, service structures, etc. (Browning, 2001; Li et al., 2019). Furthermore, in each of the different subject areas multiple types of modularity can be defined. It is important to classify the modularity, since it determines how/where modularisation will be implemented, which modularisation strategies could be used, and how to improve communication between stakeholders.

Modularisation could basically be applied to any given complex systems. Browning (2001) stated that any product, process, and organisation is in their own way a complex system and could be divided into modules. Furthermore, Li et al. (2019) classified modularity for technical systems. The technical systems was divided into three parts, namely the product, the service and the supply chain. Additionally, Piran et al. (2016) identified six types of modularity, namely design modularity, modularity in production, modularity in use, organisational modularity, service modularity, and environmental modularity. Based on these classifications the following classes are defined for this research:

- Modularity in product design
- Modularity in (production) process
- Modularity in use
- Modularity in service
- Supply chain modularity
- Organisational modularity
- Environmental modularity

First, modularity in product design refers to dividing the components of a product into modules or clusters with high internal interactions and limited interactions with other modules and external parts. Second, Modularity in process relates to the way the manufacturing and/or assembly process is configured based on the interaction flows between the components and/or modules. The main goal is to improve efficiency of the processes. Third, modularity in use means that during the use phase of the product, flexibility exists to swap modules or change the configuration of the system. Furthermore, modularity in service refers to a flexible modular structure to add and/or delete certain services offered by a service company before and/or after purchase of a service. Additionally, supply chain modularity refers to the degree of proximity of the members within a supply chain (e.i. the company itself and the suppliers). The proximity degree is based on the geographical location, cultural differences, organisational proximity and electronic communication of all suppliers and the organisation (Voordijk et al., 2006). Moreover, organisational modularity means an organisation is divided in groups of people, teams or departments based on the internal interactions and limited external interactions with other modules. Thus, increasing the organisational efficiency (Ulrich, 1994; Browning, 2001; Piran et al., 2016). Finally, Piran et al. (2016) identified a new class, namely environmental modularity. In this case modularity is used to lower the environmental impact throughout the whole life cycle of the system. The architecture enables prolonging the life time by swapping and updating modules, it supports end-of-life processes and takes all environmental impact into account during the design process.

3.2.1 Modularity in the residence design

Let's take a look at what modularity means for the design of the residence example. The system has multiple subsystems which deliver certain functions (e.g. the kitchen or bathroom). The components within these subsystems have a high internal interaction, while the interactions with surrounding parts and/or subsystems is limited. Therefore, the individual subsystems make a good module.

Consequently, the modules are the living room, the kitchen, the hallway, the bathroom, the bedroom, the second bedroom, the exterior design and the attic. The company developing the houses is able to develop a new version of the system by for example changing only the design of the kitchen. The new version might than have improved the user friendliness and/or meet broader market needs (see figure 5).

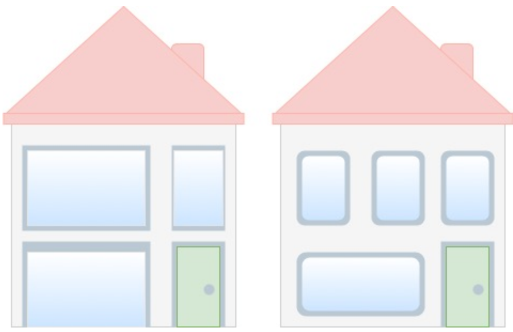


Figure 5 – Residence version one on the left and version two on the right

Additionally, the company can decide to make variants of certain modules to sell a higher variety of products. The exterior design module can be manufactured in multiple colours, materials or shapes, such that each variant product attracts a different part of the market (see figure 6).

3.2.2 Benefits of modularisation

Over the years, many benefits of modularity have been researched and identified. First of all, modularity decreases development time. Adaptations, changes and improvements in the design of modules can be executed in a shorter time period, since they are not influenced by, and do not influence, the designs of other modules in the system. Second, project teams can work on the development of multiple subsystem simultaneously, which shortens the total development time of the whole system. Third, the teams working on the subsystem can become more familiarised with the subsystem, their functions, and their technologies. For this reason, they will specialise their knowledge to further improve the quality, performance and/or functionality of the module, which will increase the performance, the functionality or quality of the whole system. Finally, new and/or improved version of a

Figure 6 – Two variants of the residence design

system can be launched quicker with improved or innovative new modules instead of a whole new design (Ulrich, 1994; Gershenson et al., 2003; Wee et al., 2017).

Second, implementation of modularity increases flexibility in multiple areas. The flexibility in the system can be used to increase the variety of product offerings in a company's product portfolio. The modular architecture reassures that modules within a system could be mixed and matched to the preference and need of specific market segments (Schilling, 2000; Koh et al., 2015; Wee et al., 2017). Additionally, modularity unlocks the potential to implement mass customisation and offer products that meet individual customer needs (Ulrich, 1994; Sanchez & Mahoney, 1996; Gershenson et al., 2003; Piran et al., 2016; Wee et al., 2017). Furthermore, the systems and product become more flexible regarding requirement changes in later stages of the development process. When requirements are updated, only the relevant modules have to be adapted or replaced instead of updating the whole system (Koh et al., 2015), (Wee et al., 2017).

Third, cost reduction can be achieved with the implementation of modularity in the development process and later phases in the life cycle of the system. A modular architecture eases the changes of modules within the system to new or adapted modules. Due to this flexibility certain modules can be reused in new product designs. Thus, a new systems does not have to be developed from scratch, which reduces the costs of the development process. Additionally, the flexibility in the product architecture enables product variety by changing single modules and/or components in a system, which results in economies of scale for the common modules and/or components across the different systems in the product portfolio of an organisation. Furthermore, the separation of modules and functions eases the testing phase. After the development of a module the functionality and performance can be tested independently of the other modules in the system. By the time the whole system is configured and assembled the single modules meet the individual requirements and the testing is executed to reassure that the single modules function as a whole. Therefore, time and costs are saved by prevention of redesigning efforts in later stages of the development process (Ulrich, 1994; Gershenson et al., 2003; Piran et al., 2016).

Additionally, modularity has a positive effect in later phases of the life cycle of the system. During the use phase, the modular architecture eases the maintenance and service of the system. Broken or malfunctioning modules can be replaced or can be taken out of the system to be repaired. Furthermore, it could enables user to swap modules with new and improved modules, which increases the lifetime of the system. Besides, at the end of the lifetime the modular systems can be disassembled, which enables recycle-ability of a system or reuse of modules (Ulrich, 1994; Gershenson et al., 2003;

Piran et al., 2016; Wee et al., 2017).

Finally, implementation of modularity in process has similar advantages. Modularity adds flexibility to the process, such that the process can be more easily changed and adapted when necessary. Besides, it positively influences the learning-curve. Assembly of the products and the interfaces are more common, which saves learning time for each (new) product design. Additionally, costs can be saved due to high commonalities in the production processes of the systems. Furthermore, it is possible to outsource the development and production of specific modules (Gershenson et al., 2003; Piran et al., 2016).

3.2.3 Modules and module interfaces

Each module in a system is included to deliver a certain function. The functionalities that have to be delivered varies between market niches and from customer to customer, however some modules are always included to offer basic functions. Overall, modules can be classified into five different types, namely basic function modules, auxiliary function modules, special function modules, customer specific function modules and adaptive function modules.

Basic function modules are modules that are integrated in all products within a product family. Auxiliary function modules often have multiple variants, however one of the variants has to be integrated in the design of a systems. Adaptive function modules change/adapt inputs between modules to reassure that they will function together as required. Special function modules deliver specific functions that are not integrated in all systems, but only in systems to target specific markets. Finally, customer specific modules can be used to customise a system to the specific needs of a customer (Kusiak & Huang, 1996; Huang & Kusiak, 1998; Chen & Crilly, 2014).

For modularity in product design, modularity in manufacturing and modularity in use, product flexibility and variability of a system will unlock the benefits of modularisation. The interfaces and assembly compatibility determine the degree of variability and flexibility. In academic literature multiple interface types have been defined. The most common mentioned types are component-swapping, component-sharing and bus modularity (Kusiak & Huang, 1996; Huang & Kusiak, 1998; Chen & Crilly, 2014). Furthermore, other types that have been defined are fabricate-to-fit, sectional and slot modularity (Ulrich, 1995; Chen & Crilly, 2014).

- Component-swapping modularity: the ability to change individual components to create variants.
- Component-sharing modularity: components are used in multiple different products from the organisation.
- Bus-modularity: uses a common interface to connect components to a single bus component. Variants can be developed by combining different components to the bus.
- Fabricate-to-fit modularity: combines standard components with components that have certain flexible features (e.g. length or width).
- Sectional modularity: uses common interfaces between components, such that mixing and matching is possible.
- Slot modularity: uses unique interfaces between the different types of components. Components can be swapped, but not configured in a different way.

These interfaces enable and promote changing, swapping and combining a limited set of (standardised) components to form variants. Thus, an organisation can offer a broader variety of products in their product portfolio. Furthermore, the manufacturing process is more efficient, since a lower amount of components is used, produced and/or processed. Additionally, the interfaces can be used for user flexibility. Individuals are able to swap and/or change components or modules after the purchase of the product (Ulrich, 1994, 1995; Chen & Crilly, 2014; Piran et al., 2016). Overall, the interfaces determine the possible configurations, flexibility in use, and number of variants.

3.2.4 modularisation strategies

Over the years, many methodologies and strategies have been developed to determine the grouping and clustering of parts into subsystems or modules. Ideally the components within the modules have a high internal interaction and a limited external interaction. Furthermore, it is recommended to combine parts and/or functions into one module that are highly influenced by changes in customer wishes and needs. Additionally, the modules should have a limited amount of components within one module. Overall, the division of components into modules is an optimisation problem. Jose & Tollenaere (2005) have grouped methodologies for defining modules in five categories, which are the following:

- Clustering methods
- Graph and matrix partitioning methods
- Mathematical programming methods
- Artificial intelligence
- Genetic algorithm and other heuristics

3.3 Platforms and product families

Modularisation of systems can be combined with a platform approach. The platform approach can be implemented by itself, however the approach can be extended by adding modularity to the system. A platform is a common module, subsystem, technology, process and/or component that is used across different products, product families and product lines within the product portfolio of an organisation. The platform requires predefined interfaces to reassure easy integration of other subsystems and/or modules into the design of the system. The common platform in combination with the systems architecture facilitates offering a variety of closely related products (Robertson & Ulrich, 1998; Bowman, 2006; Jiao et al., 2007; Halman et al., 2006; Cameron & Crawley, 2014).

The main reason to implement platforms within a company is to change the focus from single products to the development of product families (Meyer & Lehnerd, 1997). A product family is a set of products with a high similarity in assets and technology, however all differ slightly to target multiple niche markets (Meyer & Lehnerd, 1997; Simpson et al., 2006; Jiao et al., 2007). The platform supports the development and/or production of product family members and derivatives while efficiently managing the use of organisational resources (Halman et al., 2006). The platform is used as the common building block on which the products are built. Each member of a product family is a variant of the set of available modules and the platform. For this reason, it is not only important to manage the commonalities, but also the distinctiveness of the modules and functions to add variety to the product family (Robertson & Ulrich, 1998; Halman et al., 2006).

It is important is to find the right balance between the degree of commonality and the distinctiveness of the products in a product family (Robertson & Ulrich, 1998; Halman et al., 2006; Cameron & Crawley, 2014). Since, a high commonality reduces the costs drastically, while on the other hand a high distinctiveness targets more market niches and offers the ideal product for each individual customer. To determine the 'right' level of distinctiveness the different customer wishes and requirements have to be analysed. Next, it has to be decided how the products have to be differentiated to meet multiple market niches and which markets will be targeted. To decide on which markets to target, the commonalities between the niches are studied. Ideally the markets are selected that have a high commonality and can be entered by easily adding, changing or removing parts, subsystems and/or modules. For this reason, the architecture plays an important role in the ability to differentiate the products, since it determines what and how parts, subsystems and modules can be integrated (Robertson & Ulrich, 1998). Additionally, Cameron & Crawley (2014) highlights the importance of the cost considerations in the trade off. He states that a higher commonality does not always guarantee more economy of scale benefits. Therefore, the cost structure of a company has to be taken into account to make a sound decision.

The use of platforms is not limited to product designs or it's use in product families. A common platform in a company can be found within the design of systems (product platform), within the processes related to manufacturing and/or services (process platform), and in the target markets (customer platform) (Halman et al., 2006). Additionally, platforms can be found in the knowledge base of an organisation (knowledge-platform) or in the organisational structure of teams, groups and departments (people-platform) (ElMaraghy et al., 2013). Furthermore, companies can develop a brand platform, which uses one brand as commonality of which sub-brands are launched. The sub-brands will be perceived as equal quality and equal worth compared to the platform brand. Finally, a global platform can be implemented. The platform is used throughout all systems offered internationally, however the architecture ensures that each system can be differentiated to fit to the rules, regulations, and customer needs of a specific country (Halman et al., 2006).

3.3.1 Platform within the residence design

A platform is a commonality in design, technology and/or process that is used across different products that are part of the product portfolio of a company. In section 3.2.1, the residence has been divided into modules. Each of these modules could be selected as a platform for the residence product family and/or product line. The modules that would be suitable as platform are the subsystems that connect to multiple other subsystems. For the residence this would be the hallway modules. The hallway module has shared walls, doorways and staircases that connect with the other modules, these are the interfaces. The remaining modules are functional spaces for the user. Besides, the exterior design and attic modules are visible to the residents and for the outside environment. Offering a variability in these modules could cover a larger share of the market wants and needs.

The use of a common hallway module will reduce the costs and development time of new version and/or variants. Besides, research and development efforts can be focused on improving the quality of the common module in the residence product family. Furthermore, the platform can be strategically positioned within the market segmentation grid for a specific platform strategy. With a vertical strategy for example, the user interface and exterior design can be upgraded with high end materials to target the best tier within a single segment. On the other hand, more simplistic exterior design can be used in the economy tier of the same segment.

3.3.2 Benefits of product platforms

The implementation of a platform in an organisation has similar advantages as the use of modularisation (see section 3.2.2), since modularity is an extension to platforms. To summarise, it decreases the development time of the system, it increases the flexibility of the design, it supports cost reduction, maintenance, services, and end-of-life of the system (Meyer & Lehnerd, 1997; Simpson et al., 2006; Halman et al., 2006).

Additionally, the use of platforms improves the communication among stakeholders. The platform structure can improve the learning process and training of the users, service personnel, and maintenance personnel. Besides, it supports the communication regarding the product structure and functionality within an organisation or among customers. Finally, it helps the communication about the brand identity and the market positioning of the products (Halman et al., 2006).

Furthermore, product platforms and product family development is perceived to reduce complexity. The complexity is reduced in the context of the product development process, the management of variants and product families, and in the decision-making process about target markets, the development of variants, and the addition of functionalities and modules (Halman et al., 2006; Simpson et al., 2006).

Finally, the use of a platform throughout product families can increase the positive effects of the economies of scale. The common platform is used in all members of the product family, which means a high volume of platform. Besides, certain modules, parts and subsystems could be used frequently in a large share of the product family (Halman et al., 2006; Simpson et al., 2006). For this reason, the economies of scale benefits can be increased further with the use of platforms.

3.3.3 Platform strategies

As stated before, the main reason to implement the platform approach is to direct the focus towards product family development instead of single product development. The product family focus plays an important role in the planning, managing, and the success of platforms. For this reason, it is important to first define the product family members. Two approaches exist for the design of a product family, namely the top-down and bottom-up approach. First of all, the top-down or proactive platform approach focuses on the formulation of a common platform. The product family and the derivatives are strategically managed and developed with the common platform as basis. Second, the bottom-up or reactive platform approach starts with an analysis of the product portfolio to find similarity in product designs. Next, the products are redesigned to increase commonality and design a common platform that will be shared in all redesigned products (Simpson et al., 2006; Marion & Simpson, 2006).

Furthermore, platforms of the product families can be classified into two types. The first type is a module-based platform. These type of platforms make use of a modular architecture with clearly defined interfaces. Members of the product family are developed by excluding, swapping, updating or adding functional modules to the product platform. The module-based platform is the most commonly used type of platform in the manufacturing industry. Furthermore, the second type is the scale-based platform. In this case, certain variables of the platform can be scaled (e.g. stretched or shrunk) to adapt the product to certain market niches. The scaling of platforms is often used to adjust the products according the performance variety between the niches (Simpson et al., 2006; Jiao et al., 2007).

After determining the common platform for a product family, the platform leveraging strategy should be formulated to maximise the benefits of the platform approach. The strategies are formulated to determine which market niche the platform targets and which niches will be targeted with the introduction of product family members. For this reason, the market has to be analysed and segmented (Meyer & Lehnerd, 1997; Marion & Simpson, 2006). With the use of a matrix the market can be divided into niches based on the price-performance and the customer segments (see table 7) (Meyer & Lehnerd, 1997; Marion & Simpson, 2006). The horizontal axis is for the customer segments and the vertical axis is for the price-performance segments. The price-performance segments can be adjust to more specific value if necessary.

	Segment 1	Segment 2	Segment 3	Segment 4	Segment 5
Best					
Better					
Good					
Economy					

Figure 7 – Market segmentation grid based on Meyer and Lehnerd (1997)

The market segmentation grid can initially be used to position the platform in a single niche-market tier. Thereafter, it can be used to formulate the platform leveraging strategy by analysing which niche markets can/should be targeted. Meyer & Lehnerd (1997) formulated four platform leveraging strategies: the niche-specific platform strategy, the horizontal platform strategy, the vertical platform strategy, and the beachhead platform strategy (see figure 8). The niche-specific platform strategy uses a single platform to develop products that closely meet all customer wishes in a single, specific niche-market tier. The platform is used to offer a variety of products within a niche or for the development of new versions of the products. The sharing of subsystems, manufacturing processes or parts/components between different niche-market platform products is limited. The use of this strategy can result in high costs, a long development time, and limited knowledge sharing (Meyer &

Lehnerd, 1997).

Second, is the horizontal platform strategy. The key platform modules, manufacturing processes or parts/components are shared across multiple customer segments. A company can decide in which price-performance segment it would want to implement this strategies (Meyer & Lehnerd, 1997; Marion & Simpson, 2006). Within practical context it is found that companies most often target the highest price-performance segment, however successes can also be found in the lowest segment. The main benefits of this strategy is that the development efforts can be shared across a wide range of products in multiple customer segments, which prevents reinventing the wheel for each single tier. Besides, the products can be developed more rapidly and benefits of economies of scale can be achieved. Furthermore, all members of the product family will benefit if the quality of the platform increases, a new functionality is introduced or a new technology is implemented, (Meyer & Lehnerd, 1997; Marion & Simpson, 2006).

Third, a vertical platform strategy can be implemented. In this strategy the platform is used to target multiple price-performance segments in a single customer segment. Organisation can do this in two ways. The first method is to start with targeting the high-end segments of the niche market. The platform and product can be changed by removing certain functionalities or by certain key modules, subsystems or technologies in lower segment of the niche market. The second method is to start from a low-end market segment and work up to higher segments. The higher segments can be target by adding modules, functionalities, and technologies or swapping modules and technologies for better quality ones. The main benefit is the leveraging of the knowledge about the customer segment. Furthermore, the costs are reduced for the members of the product family, compared to developing products for each price-performance segment in a specific niche market (Meyer & Lehnerd, 1997; Marion & Simpson, 2006).

Finally, a beachhead strategy can be used, which combines the horizontal and vertical platform strategy. The initial platform is most commonly positioned in a low price-performance segment within one customer segment. The platform is designed to be able to add new and/or different modules, technologies, or functionalities. New modules, technologies or functionalities are used to target higher segments in the market. While different modules, technologies, or functionalities are used to target another customer segment in the same price-performance segment. With the use of the beachhead strategy the commonality between the members of a product family is maximised. For this reason, economies of scale can be achieved for the common platform and certain key subsystems (Meyer & Lehnerd, 1997; Marion & Simpson, 2006).

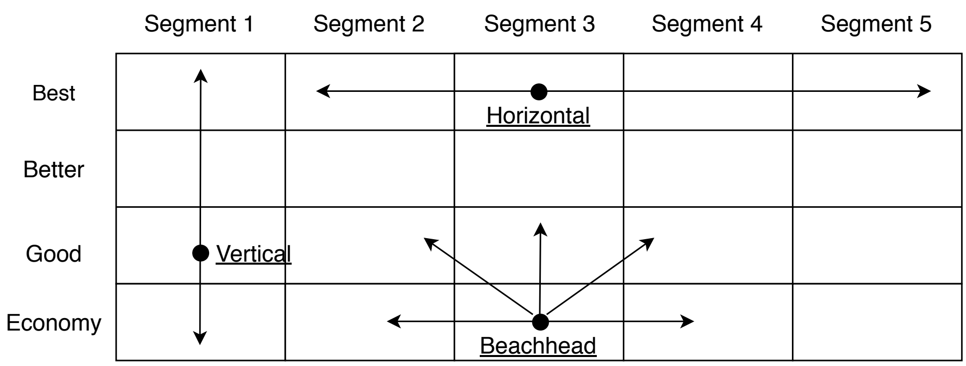


Figure 8 – Market segmentation grid with platform strategies based on Meyer and Lehnerd (1997)

3.4 Modularisation and platform within the Architecture, Engineering and Construction Industry

Many companies within the Architecture, Engineering and Construction (AEC) industry try to reduce the costs and lead time for development projects to stay ahead of competition. Often companies approach the construction projects as one-offs and made-to-order, however some companies have implemented modularity and platform strategies to achieve cost and time benefits (Gibb & Isack, 2003; Voordijk et al., 2006). For this reason, the modularity and platform strategies already used within the construction industry are studied and compared to the theory from the previous sections.

Within the AEC industry a distinction is made between product modularity and process modularity. The product modularity refers to the modularity and more specifically the interfaces used within the design of a construction. The process modularity refers to the manufacturing process of the building, which influences the degree of on-site production or off-site production (Gibb & Isack, 2003; Voordijk et al., 2006; Jonsson & Rudberg, 2014; Peltokorpi et al., 2018).

In the context of product modularity the interfaces play an important role, as mentioned in section 3.2.3. The modular interfaces have been classified in six types. Peltokorpi et al. (2018) states that bus-, slot, sectional, and fabricate-to-fit modularity are used within the AEC industry. Furthermore, it has to be noted that within one building architecture a combination of the four interfaces could be found. The buildings are constructed and configured based on the interfaces and the architecture for a building (Peltokorpi et al., 2018). Additionally, Voordijk et al. (2006) found that apart from the mentioned modularity types, component-sharing modularity is used within product families in one organisation. Finally, according to the authors knowledge, component-swapping modularity has not been mentioned in context of the AEC industry within academia.

Traditionally, buildings are constructed at location, however with the use of modularity it is possible to (partly) manufacture the building at the supplier's factory and assemble at the construction site. This is referred to as process modularity, which can be further classified into four categories: component manufacture and sub-assembly, non-volumetric pre-assembly, volumetric pre-assembly, and modular building (Gibb & Isack, 2003; Jonsson & Rudberg, 2014; Peltokorpi et al., 2018). Component manufacture and sub-assembly is the traditional way of construction, which uses on-site construction of the building with the use of components and small sub-assemblies. Non-volumetric pre-assembly are two dimensional systems that can be used to be assembled to make up parts of the three dimensional building (e.g. walls or floors). The systems are manufactured off-site by the supplier and will be assembled on site. The volumetric pre-assembly is a three dimensional system (e.g. toilet units, shower units etc.), which is delivered as a closed to finished product. The whole pre-assembly is constructed off-site and on-site is connected and attached to other subsystems of the building. Finally, a modular building is constructed with the use of fully defined three dimensional pre-assembled (sub)systems. The pre-assemblies are fully produced off-site and on-site only connection and fixation is necessary (Gibb & Isack, 2003; Jonsson & Rudberg, 2014; Peltokorpi et al., 2018). Moving from component manufacture and sub-assembly to modular building influences the ratio of on-site and off-site production, however on-site construction will always be necessary to a certain degree (Jonsson & Rudberg, 2014).

Finally, the top-down and bottom-up family design approaches (see section 3.3.3) have been implemented within the AEC industry (Shafiee et al., 2020). According to Shafiee et al. (2020) both methodologies have significant advantages and disadvantages. The bottom-up approach is perceived to be faster to implement, less complex and requires lower initial investments compared to the top-down approach. However, the top-down approach would result in a higher the Return On Investment (ROI) over the long run compared to the top-down appraoch (Shafiee et al., 2020).

3.5 Product line engineering

Closely related to product platforms and product platform strategies is product line engineering (PLE). A product line is a group of similar products of a certain brand with slightly different features and different prices, which together target a larger share of the market (Product Line, n.d.). PLE is the strategic management of these product lines and the corresponding product portfolio's. It focuses on the advantages of the commonalities between the products in a product portfolio by adoption of platforms and modularity within a product line, while taking the difference into account (Böckle, 2005). Commonalities in this context refer to similarities in design, components, manufacturing processes, assembly processes, delivery, use, sustaining of a product, and end of life of a product. The management of product portfolio, and more specifically the commonalities, increases efficiency and performance within an organisation (Birk et al., 2003; C. Krueger & Clements, 2013; Clements, 2014).

Product line engineering finds its beginnings in the software engineering domain, however it has found its way to into a broader area in the last two decades (C. W. Krueger, 2015; Wozniak

& Clements, 2015). The PLE approach has been used to improve the reuse of software in the development process of new systems, instead of starting from scratch assets of older systems and projects are reused in the new project to reduce development time. Nowadays, PLE is also implemented in the development of products and systems to support efficient management and reuse of all assets of an organisation.

Van der Linden et al. (2007) formulated four principles that played a central role in the first generation PLE. The first is variability management, which focuses on managing the variation within all assets of an organisation. Variability is communicated by means of features. Features are distinctive and/or prominent characteristics, quality, or aspects perceived by the user of the system (Kang et al., 1990). The features determine what functionalities a systems has to deliver, how it can be used and what it looks like. Features can be divided into functional features, operational features, and presentation features (Kang et al., 1990). Second, is the business-centric principle. The principle is used to put emphasis on the importance to align the product line with the long-term strategy of an organisation. Third, the architecture-centric principle refers to the importance of the use of an architecture within the product line to take full advantage of commonalities between individual systems. Finally, the two-life cycle principle, which include the domain engineering and application engineering life cycle. The domain engineering life-cycle focuses on the development of the architecture, platforms, and variation points in the assets portfolio. The application engineering focuses on the detailed development of a single variant (Van der Linden et al., 2007).

More recently, a second generation PLE (2GPLE) theory has evolved. C. Krueger & Clements (2013) identified 5 characteristic differences of 2GPLE compared to the first generation. First of all, the use of features increased drastically. Features are used to communicate the functionality and behaviours of a system with the end-users. Each variant system is defined by the feature that is offered in comparison to the other variants of a product line. Furthermore, features have become the common language to determine the final configuration of a system from the shared assets within an organisation.

Second, the focus shifted towards the domain engineering life cycle and the application engineering life-cycle almost disappears due to highly automated configuration processes. In other words, the development efforts within an organisation are mainly focused on developing common and variant assets for the whole product line. Furthermore, the assets are designed to be able compatible with relevant other assets. On the other hand, the actual development and configuration of a single variant system or product is automated, which in the first generation PLE was done within the application engineering life-cycle. Simultaneously, this automation is the third characteristic of 2GPLE. The automation is of high importance for configurations of systems, but not limited to configuration purposes only. Automation is used for the storage, management, and construction of assets, variant points, and product lines. Furthermore, it supports mapping from the variability model (e.g. feature model) to the required assets (C. Krueger & Clements, 2013; Clements, 2014).

Fourth, configuration management reduces complexity, since it is limited to configuration of assets in the PLE library compared to configuration of products, assets, and product lines in the first generation PLE. Furthermore, all assets and artefacts are treated as equals. The first generation PLE was introduced for the development of software systems, however in the 2GPLE hardware, requirements, tests plans, budgets, etc. becomes just as important as software assets (C. Krueger & Clements, 2013; Clements, 2014).

Finally, PLE stimulates cross boundary models and constructs to stimulate cooperative working on shared assets. The organisations structure is not divided into departments that work on a single subsystem, product, or product line. Instead the employees all work together on the whole asset portfolio of the company (C. Krueger & Clements, 2013; Clements, 2019).

On the other hand, variation management is still of high importance in the 2GPLE as it was in the first generation. The assets are managed to achieve variety for the customer, while stimulating the commonality among the components and subsystems within the asset library. For this reason, the commonalities and variability between the systems and within the assets have to be identified. Modelling of the identified commonalities and variability can support variability management, which is referred to as variability modelling.

3.5.1 Variability modelling

Variability modelling uses an unit of variety and a specific notation. The unit of variety support communication about the variety within the assets and support configuration of a systems variant from the asset library. As mentioned before, features are often used as unit of variety. However, multiple other units exists, which can be used. Berger et al. (2013) surveyed which units are most often used within industry. They found that five units are regularly used, which are features, variation points, configuration options, decisions, and calibration parameters. The remaining units were often company specific. The most commonly used units (more then 70%) are features, variation points, and configuration options.

Additionally, a modelling method has to be chosen. Some modelling methods are directly related to specific units of variability, however this is not always the case. The modelling methods used within industry are feature models, spreadsheets, key pairs, Domain-Specific Language (DSL), UML-based representation, decision models, free-text description, product matrix, Aspect-Oriented Language (AOL), and Architecture Description language (ADL). Feature models are used most often, compared to the other modelling methods (70% versus 20-30%). However, most practitioners use multiple methods simultaneously, since one method does not fully meet all the needs of the users. The tools used for the modelling method vary from organisation to organisation, and between projects (Berger et al., 2013).

3.5.2 Product Line Engineering of the residence example

The company producing the residences, has extended the product line of the residence by offering a variety of products. The company focused on determining the commonalities and differences between the products to reap the advantages of PLE. Each of the individual products contain slightly different features to target a larger share of the market. Additionally, each of the systems in the product line contain similar modules and subsystems to reduce the cost and development efforts. The commonality and variability is modelled with the use of a feature model, which can be seen in figure 9.

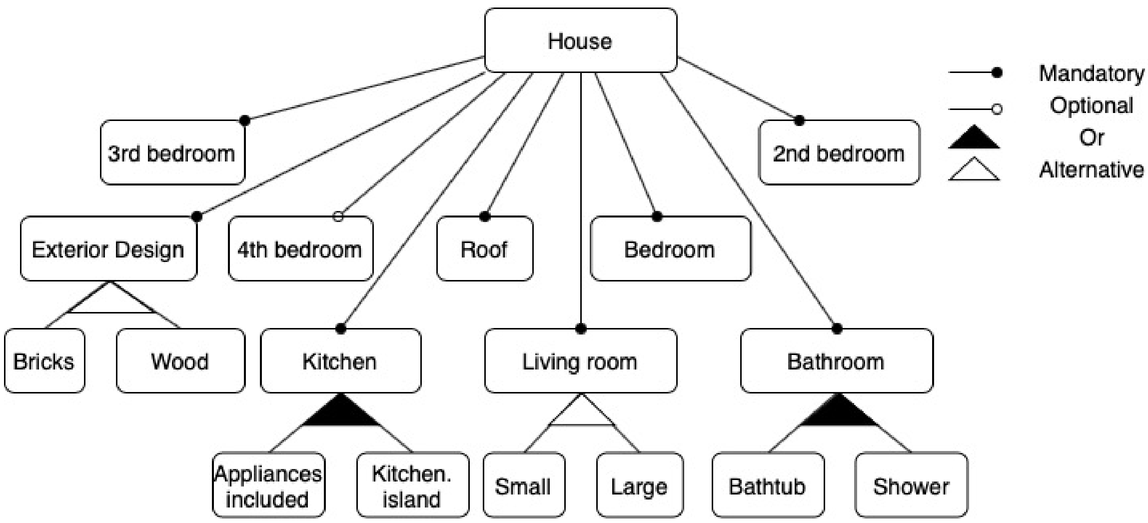


Figure 9 – Feature model of an espresso machine

A customer can make multiple choices regarding the residence design. The features that can be adapted are the exterior design, kitchen, living room, bathroom and 4th bedroom. For the kitchen and bathroom a single option can be chosen or both can be combined. This means that a residence can have a bathroom with both a bathtub and a shower. On the other hand, for the living room only one size can be chosen, either the small or large. Finally, the customer can choose to include or exclude the fourth bedroom, depending on their wishes. The common modules for the residence are the bedroom, second bedroom, third bedroom, and roof. The combination of the variable modules an common modules gives a large variety of products that meet more customer wishes.

3.5.3 Benefits of product line engineering

The main goal of PLE is increasing the reuse of assets within an company. The reuse of parts, subsystems, and/or whole systems results in multiple benefits for a company. The most often mentioned reasons for implementation of the PLE methodology are cost reduction and shorter time to market (Böckle, 2005; Van der Linden et al., 2007; C. Krueger & Clements, 2013; Clements, 2014). Due to reuse of subsystems and/or parts economies of scale can be achieved, which reduces manufacturing costs. Additionally, the subsystems and parts do not have to be designed for each individual system, which reduces development costs. For the same reason, the development time is also reduced drastically (Böckle, 2005; Van der Linden et al., 2007).

Other benefits that have been found are quality increase, increased customer satisfaction, and reduced maintenance efforts. Parts and subsystems are developed to be integrated in multiple or even all products in the product portfolio. Since these will be used often, testing and quality checks are done more extensively. Additionally, when flaws are found within the assets after introduction of a system, correction and improvement of the asset positively influences the quality of multiple systems offered by the company. Furthermore, the defect rate can be reduced by improving the robustness of parts and/or subsystems of multiple systems. Therefore, the quality increases for all systems developed by a company (Böckle, 2005; Van der Linden et al., 2007).

Furthermore, customer satisfaction in positively influenced with the implementation of the PLE approach. The systems can be developed at a lower cost, a faster time to market, and a higher quality as stated above. Additionally, PLE enables mass customisation or the ability to offer a large variety of products to meet individual customer needs and wishes. Furthermore, the look-and-feel of the systems produced are similar to one another, since many assets are reused in multiple systems. Overall, all these aspects increase the customer satisfaction (Böckle, 2005; Van der Linden et al., 2007).

Finally, maintenance efforts are reduced. Maintenance is lower since the quality of the offered systems are increased. Besides, the systems developed make use of a modular architecture. As mentioned in section 3.2.2 a modular architecture positively affects maintenance tasks, since modules can be exchanged or taken out to be repaired. Additionally, test procedures and protocols are part of the assets of a company and thus can be reused for multiple systems within PLE. On the whole, maintenance efforts decrease (Böckle, 2005; Van der Linden et al., 2007).

3.6 Systems engineering and model-based systems engineering

In the previous sections enablers and methodologies for variant management have been presented. The research aims to integrate MBSE with the above mentioned methodologies and therefore relevant MBSE literature will be highlighted here. First, a definition for systems engineering is formulated. After that, model-based systems engineering is studied in more detail.

3.6.1 Systems Engineering

Over the last decades, the complexity of systems and products have increased drastically. One driver of the increased complexity is the technological advancements, which gives the systems more and innovative functionalities. In addition, the complexity is influenced by the increase in stakeholders that play a part in the development and/or use of the system. To deal with the increased complexity of systems, the systems engineering approach can be implemented in the development process.

System engineering does not have one general accepted definition in the literature. The International Counsel On Systems Engineering (Sillitto et al., 2019) defines systems engineering as follows:

> "Systems Engineering is a trans-disciplinary and integrative approach to enable the successful realisation, use, and retirement of engineered systems, using systems principles and concepts, and scientific, technological, and management methods." (p. 2)

Furthermore, Friedenthal, Moore and Steiner 2014 define systems engineering as the following:

> "Systems engineering is a multi-disciplinary approach to develop balanced system solutions in response to diverse stakeholder needs. Systems engineering includes both management and technical processes to achieve this balance and mitigate risks that can affect the success of the project" (p. 4)

These definitions are similar to other definitions found in literature, which all have overlapping subjects. The definitions all put emphasis on the interdisciplinary approach of systems engineering. In addition, the approach focuses on adequately translating and defining requirements of all stakeholders involved in the development and use of the system. Furthermore, systems engineering uses a life-cycle view perspective for the development of the system, which means that each life-cycle phase of the system is taken into account directly from the start of the project. Finally, systems engineering takes an overview perspective and focuses on the whole system instead of looking at single components or subsystems (Blanchard et al., 1990; Friedenthal et al., 2014; Buede & Miller, 2016; Sillitto et al., 2019).

Many companies have implemented systems engineering in practice. The main reasons for using systems engineering within an organisation are the management of the complexity related to the system and handling changes in market needs and technological advancements. Accordingly, it reduces the risks related to the development process of a system. Furthermore, systems engineering focuses on tracking of the requirements, changes in the design and/or requirements, and interfaces of subsystems throughout the whole process, which limits cost overruns or lowers the costs compared to the initial budget (INCOSE, 2012).

3.6.2 Model Based Systems Engineering

Model Based Systems Engineering (MBSE) is an SE methodology. Equal to SE, MBSE is used for managing complexity of a system and deal with changing requirements and customer needs. However, MBSE uses a central system model to capture all the information of the SE process, instead of documents which are used in the original SE approach. The system model is a graphical representation of the system to be developed, which contains information related to the structure (e.g. architecture), behaviour, requirements, and parametrics of the system (see figure 10). Additionally, the model contains information related to testing, analysis procedures and verification of the designed system (Friedenthal et al., 2015e). The system model is further divided into model elements, which contain specific parts of the information about either the requirements, structure, behaviour or parametrics. The central model stores all information and documents the interrelations between the model elements (Friedenthal et al., 2015f).

The main advantage of the use of MBSE is that all information is collected and documented in one model repository instead of the multiple documents and models used in the SE approach. Additionally, the interrelations between all elements are modelled, which makes it easy to track the effect of changes in one element on other elements. The model information is automatically updated when possible and otherwise it shows where inconsistencies exist. Therefore, the model is always up to date and improves consistency of the information (Friedenthal et al., 2015e).

The model can be used by all stakeholders involved in the development process, since the view of the model can be adapted to show only the relevant information about the system for an individual stakeholder. Therefore, the model eases the collaboration and communication between stakeholders (Friedenthal et al., 2015e).

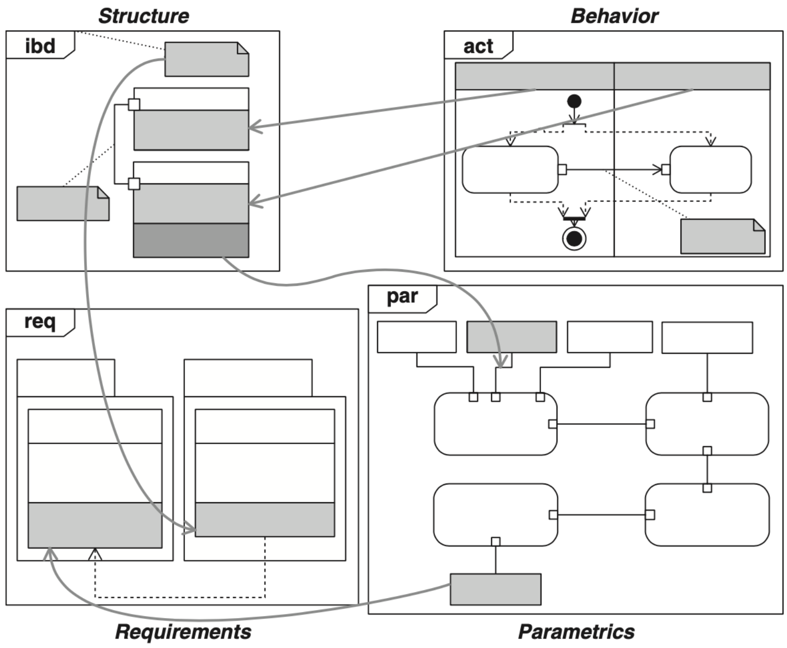


Figure 10 – Representation of a system model used within MBSE from Friendenthal et al. (2015e)

3.6.3 MBSE and the residence

As seen in figure 10, MBSE connects the structure, behaviour, parameters, and requirements of a system in one central information model. The residence is simplified and modelled according to the MBSE rules and these four pillars. A short list of requirements is formulated for the system, which are the following:

1. The maximum size of the system is 10000x7000mm (length x width)
2. The kitchen area should be minimally 8m2
3. The living room area should be minimally 12m2
4. The residence has minimally 3 bedrooms
5. The bathroom is located on the first floor

As can be seen, the requirements refer to specific parameters of subsystems or multiple subsystems, which are mapped to those subsystems within the central information model of MBSE. Additionally, the structure of the residence, the parameters and the relations with the requirements are shown in figure 11. The structure shows the residence and the it's subsystems. Furthermore, the behaviour of the system is modelled, which can be seen in figure 12. The behaviour shows the signals to initiate multiple processes, namely the provision of water, electricity and fresh air throughout the residence. In short, MBSE includes all the separate requirement, structure, behaviour and parameter in formation in one central model and links them together.

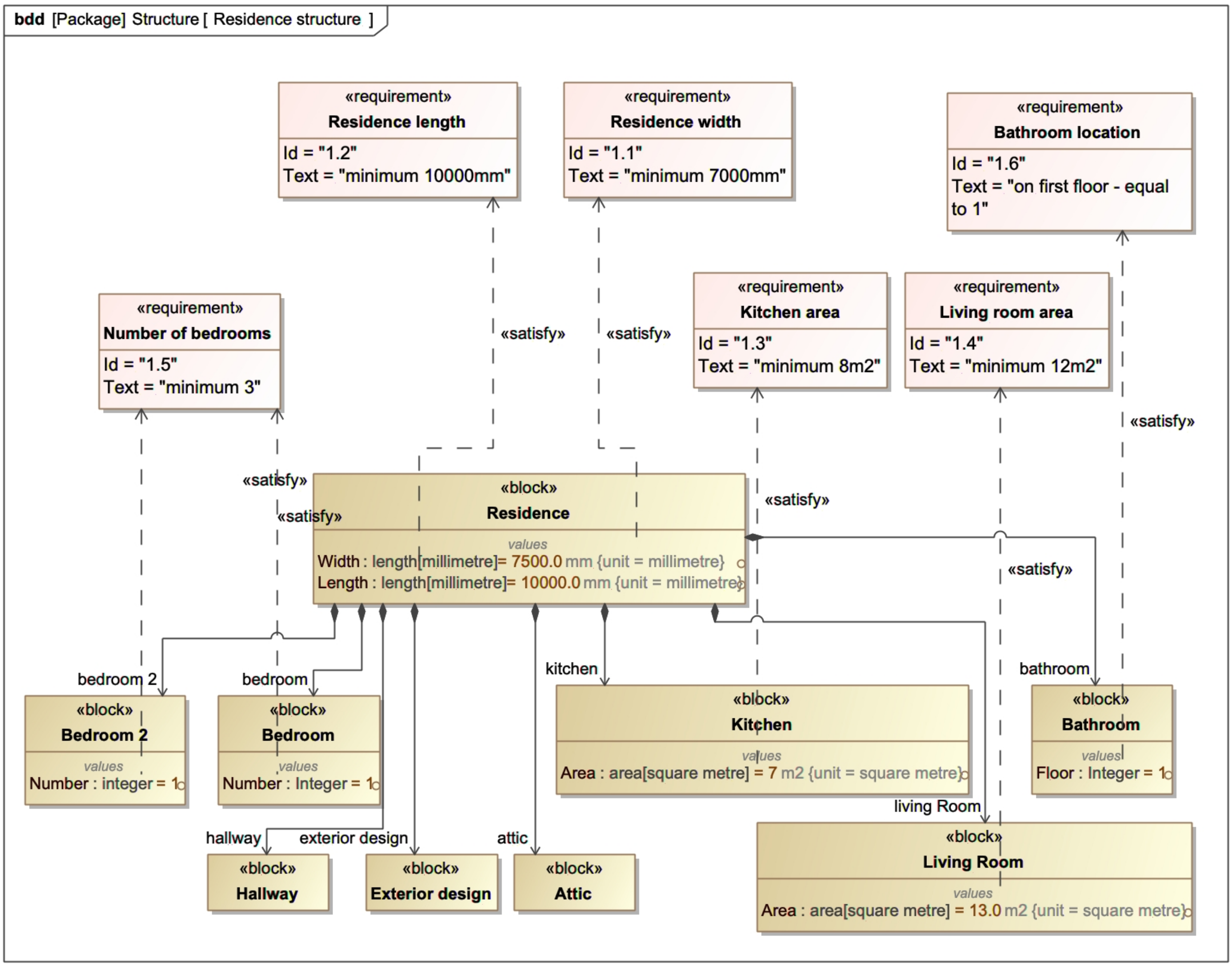


Figure 11 – System structure with parameters connection to the requirements

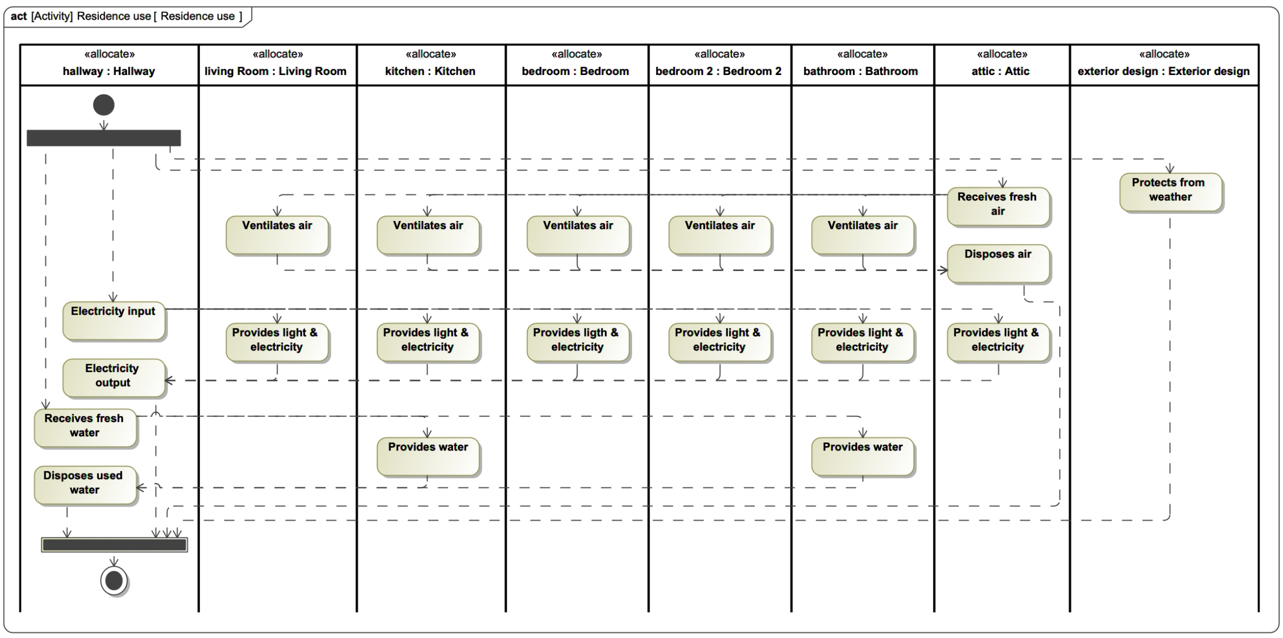


Figure 12 – Behaviour of the system connected to the system structure

3.6.4 Benefits of MBSE

The use of MBSE has multiple benefits over the document based SE approach. As stated above, the main advantage is the collection of information into one central system model. Due to this centrality, many other benefits come forth. First, it enhances communication during the whole life-cycle of the system. The model enables common understanding by all the stakeholders. Besides, the stakeholders can adjust the model to only view the relevant information for them individually. Second, the quality of the final system can be improved. All requirements are stored in one model, which reassures completeness of requirements and prevents conflicts in information and versions. Besides, the central model supports the trace-ability of information between the requirements, the design, the analysis and the testing results (Friedenthal et al., 2015e; Madni & Sievers, 2018). Third, the risks associated with the development and launching of the product are reduced as result of the improved requirement management and extensive testing (Friedenthal et al., 2015e).

Additionally, the use of MBSE supports knowledge transfer and capturing. The model contains and captures all the information about the system, which can be accessed any given time. Therefore, the knowledge is easy accessible, can be analysed, can be added on where possible or necessary, and it stimulates reuse of the information and/or knowledge (Friedenthal et al., 2015e).

Furthermore, the model can be employed in later stages of the life-cycle of the system. In the use phase of the system, the model supports training and learning. The model functions as example how the system functions and how the system can and should be used by the operator. Besides, it eases the maintenance and service processes required during the life-time of the system. The model can be used to diagnose what problems occur and where they occur within the system. Furthermore, it shows how the problems can be solved (Friedenthal et al., 2015e).

Finally, it increases productivity throughout the development process of the system. The MBSE methodology supports tracking and studying the effects of changes in the requirements and/or the design. The central model directly shows how changes in a certain part, subsystem, or requirement influence specific other subsystem or parts. For this reason, the design efforts can be limited to the influenced subsystems instead of changing the whole system. Second, reuse is stimulated, which reduces design effort, and thus increase productivity. Third, the trade-space can be analysed faster and more effective. Furthermore, the failures in the integration and testing phase of the development process are reduced, due to the tracking of the requirements and the in between testing of the subsystems. Finally, MBSE enables automatic document generation when the model systems has adequate information (Friedenthal et al., 2015e).

3.7 Building information modelling (BIM)

Building information models are regularly used within the AEC industry. BIM is defined by the National Institute of Building Sciences (2015) as "a digital representation of physical and functional characteristics of a facility" (p.1). In other words, the model shows a visual representation of what the construction will look like and how it will function (see figure 13). The visual model contains 3D, geometrical and functional information, which can be used for multiple purposes. The main functions of BIM are 3D visualiation, clash detection, code validation, constructability analysis, structural analysis, energy analysis, material tracking, material delivery, material management, cost estimation, construction scheduling, stakeholder engagement, project turnover, project closeout, and facility management (Gholizadeh et al., 2018). From this summation it can be seen that BIM is multidisciplinary and can be used by all stakeholders involved in the development process. For this reason BIM stimulates communication and collaboration between the disciplines. Furthermore, due to the improved communication between stakeholders, the model supports decision making throughout the whole life-cycle of the project (Polit Casillas & Howe, 2013; Ghaffarianhoseini et al., 2017; Gholizadeh et al., 2018).

Additionally, the model can be used as input for forensic analysis. Simulations and analysis can be run to check the design for failures, leaks, evacuations plans, etc. If errors are detected the design can be adapted before the actual construction of the building is started. Additionally, checks for conflict, interference and collision can be conducted. These checks function as verification that all separate subsystems fit together in the physical facility. The use of these analysis prevent rework or redesign, which results in a reduced or no schedule/budget overruns (Azhar, 2011; Ghaffarianhoseini et al., 2017).

More recently, BIM is extended to 4D (planning and scheduling information) and 5D (cost estimation information) (Ghaffarianhoseini et al., 2017). With 4D BIM, the model can be used as input for the sequencing of construction of the building. The model contains information from each discipline, which can be used to analyse and optimise the steps of constructing the facility. Besides, the model contains all information about the materials and fabrication, which supports the development of the delivery schedule for the project (Azhar, 2011; Ghaffarianhoseini et al., 2017; Gholizadeh et al., 2018).

With 5D BIM, the model can be employed in cost calculations and cost estimations processes, since the model contains all relevant information about sizes, geometries, and materials. The extensive information within the model will result in a reliable cost indication. Therefore, the probability increases that the project will stay within budget of the project (Azhar, 2011; Ghaffarianhoseini et al., 2017; Gholizadeh et al., 2018).

The many functions and capabilities of BIM vary from organisation to organisation. The extent, purpose and capabilities of BIM within an organisation can be assessed by it's maturity. Siebelink et al. (2018) proposed five maturity levels. In the first level, BIM is only used once in a while or when it is perceived to be beneficial to the project outcomes. Level 2 BIM maturity is achieved, when an organisation has project specific processes in place for the execution of a project that makes use of BIM. An organisation reached level 3 maturity, when management includes strategic BIM goals, when the performance and processes can be track and measured, and when cooperation within the supply chain is initiated. Level 4 maturity, includes quality targets and risk management in BIM projects. Besides, supply chain partnerships and cooperation has become part of the business strategy. Finally, level 5 maturity focuses on evaluation, learning and improving of the BIM processes. Additionally, effort is put into the alignment of the internal BIM processes with the supply chain. The supply chain collaboration is based on mutual trust, openness and transparency, which makes the supply chain function as a multifirm (Siebelink et al., 2018).

Overall, implementation of BIM in projects within the AEC industry results in multiple advantages as mentioned above. The most common mentioned advantages are improved communication between stakeholders, decision making support, cost savings due to increased productivity and efficiency, more reliable cost estimations, shorter development times, reduced overruns of budget and schedule, and life-cycle focus of the building (Azhar, 2011; Polit Casillas & Howe, 2013; Ghaffarianhoseini et al., 2017; Gholizadeh et al., 2018).

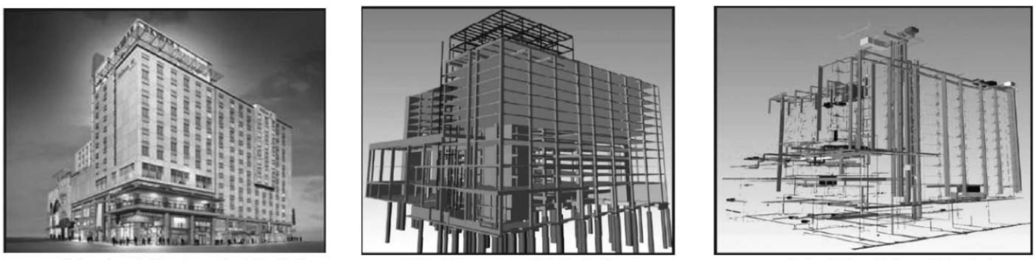

(a) Architectural Model (b) Structural Model (c) Plumbing Model

Figure 13 – Different view from a Building Information Model from Azhar (2011)

3.8 BIM versus MBSE

BIM and MBSE seem similar in purpose, functionality and benefits for use. However, multiple differences exist. First of all, BIM has a large focus area with many functions, which varies from 3D visualisation, clash detection, code validation, structural analysis, energy analysis, constructability analysis, construction sequencing (4D), material tracking, material delivery, material management, cost estimations and facility management (Ghaffarianhoseini et al., 2017; Gholizadeh et al., 2018).

On the other hand, MBSE has a smaller scope, which is specifically systems engineering. Even though, information from other disciplines can and are stored in the central model, it is mainly used for systems engineering purposes, such as system architecture, requirement traceability, and systems behaviour. The MBSE tool gives a clear overview of the relation between the requirements, structure, behaviours, and parameters of the system (Friedenthal et al., 2014).

Secondly, BIM uses the visual input of the development tools used by each discipline. The BIM show cases all the information in a central 3D visual model of a specific building project (Azhar, 2011). While MBSE uses an abstract information model to store all information regarding the to-be-designed system and later on the actual information of the designed system in an abstract modelling language (Friedenthal et al., 2014). Both BIM and MBSE central information models can be used for analysis, communication and decision making. Furthermore, they can be accessed at any given moment to check and update information (Azhar, 2011; Friedenthal et al., 2014).

Thirdly, MBSE software tools are interoperabile with other software tools (CAD programs or requirement management programs) from multiple disciplines. Changes within the MBSE model are related those other models and vice versa. In some cases the information is automatically updated both in the MBSE tool and the other software tools (Friedenthal et al., 2015e; Valdes et al., 2016). On the other hand, BIM has difficulties with interoperability of information and data contained in other software programs. The interoperability capabilities is dependent on the maturity of BIM within the organisation and the use of available software. Therefore, the building information model has to be updated, check, and verified when design changes are made in other documentation and/or software (Cerovsek, 2011; Azhar, 2011; Miettinen & Paavola, 2014; Siebelink et al., 2018).

Furthermore, the use of MBSE stimulates reuse of information, architecture, and modules. Besides, the central model is able to store multiple variants of components and/or subsystems. Therefore, it is possible to use MBSE to model a product line and/or product family and use the central model to configure single products (Friedenthal et al., 2014; Bilic et al., 2018; Colletti et al., 2019). In spite of BIM being a central information model, the information and/or knowledge is not often reused in other projects. Furthermore, BIM is not used in the context of variant design, product lines and/or product families. In organisation with a high maturity BIM can be implemented for these purposes, however this is not seen much in the industry (Siebelink et al., 2018).

An overview of the benefits of MSBE and BIM are shown in table 1. As can be seen, MBSE is more suitable for information storage, information reuse and variant design modelling. Besides, it focuses is directed to systems engineering areas such as traceability of requirement, parameters, systems behaviour, and systems architecture. Furthermore, MBSE software is able to be used interoperable, while BIM could face difficulties in this area. For this reason, MBSE would be more appropriate to use for support of variety management.

Benefits	**MBSE**	**BIM**
One central information model	x	x (excluding requirements)
Enhances communication	x	x
Quality improvement	x	x
Traceability of information	x	x (excluding requirements)
Reduced risk for development	x	
Knowledge capture & transfer	x	x
Training & learning support	x	x
Maintenance & service support	x	x
Productive increase	x	x
Stimulate reuse	x	x
Variant modelling	x	
Faster trade-space analysis	x	x
Better system integration	x	
Automatic document generation	x	x

Table 1 – Benefits of MBSE and BIM

3.9 MBSE methodologies, language and tools

In the previous sections MBSE has been defined and the benefits have been highlighted. Furthermore, MBSE seems to be a good option to use to support variety management within the AEC industry. Now it is important to highlight how MBSE can be integrated within an organisation.

Implementation of MBSE within an organisation requires a MBSE methodology, a modelling language and a modelling tool. Before the start of the process to implement MBSE a selection have to be made regarding these requirements. The methodology, modelling tool and the modelling language will support the transition towards the MBSE approach within an organisation (Friedenthal et al., 2015c). For this reason, it is important to review the methodologies, languages and tools commonly used within the industry. Furthermore, the selection criteria for methodologies, languages and tools are studied.

3.9.1 MBSE methodologies

Over the years, many methodologies have been developed in the industry and have been suggested by academics. However, a limited amount have been regularly implemented. One methodology is Harmony SE, which is developed by IBM Telelogic. The methodology is consistent with a part of the Vee-model often used within SE practices. The Harmony methodology splits the Vee-model into a Harmony SE part and a Harmony Software Engineering part (Harmony SWE). Harmony SE includes the requirements analysis phase, goes on to the system functional analysis phase, the architecture development phase, and finishes with the System acceptance phase. The actual design, development, and testing of the subsystems and the parts are included in Harmony SWE (see figure 14). Furthermore, the Harmony SE methodology makes use of the SysML modelling language (Estefan, 2007; Ramos et al., 2011).

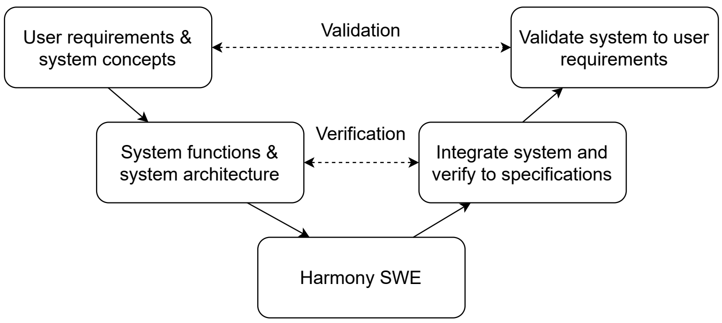


Figure 14 – Harmony SE methodology based on Estefan (2007)

Second, is the Object-Oriented Systems Engineering Method (OOSEM) proposed by INCOSE. The main objectives of the methodology are to formulate and document requirements and design information for complex systems, integration of object-oriented software, hardware and other engineering methods, and supporting the reuse of design, knowledge and information at a system level and support of design evolution (Estefan, 2007). The methodology makes use of the Vee-model approach. However, has a more elaborated workflow that includes the management processes at the left side of the Vee (see figure 15). The methodology starts with analysing the stakeholders needs, goes on with the definitions of the systems requirements, the definition of the architecture of the system, synthesises of the allocated architectures, optimisation and evaluation of the proposed alternatives, and finalises with the validation and verification of the designed system. OOSEM uses SysML as the modelling language (Ramos et al., 2011; Friedenthal et al., 2015a).

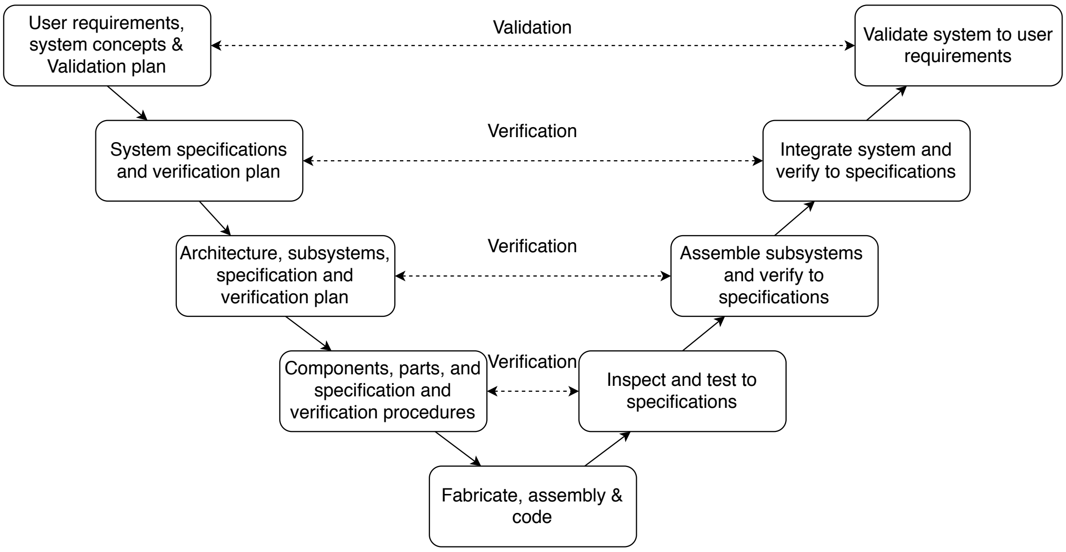


Figure 15 – The Vee-model based on Clark (2009)

Third, is the Rational Unified Process (RUP) for SE methodology from IBM rational. RUP SE is consistent with the spiral model and object-oriented methods (see figure 16). The methodology is developed to support best practices in software development. The best practices formulated are iterative development, requirements management, use of a component-based architecture, visually modelling of the software, quality verification, and tracking and controlling of changes in requirements and design (Kruchten, 2004). The phases of the methodology starts with inception of the project. Next, it has an elaboration phase, construction phase, transition phase. Finally, it finishes with a use case flow down activity. In these phases multiple workflows are executed, namely business modelling, requirements, analysis and design, implementation, testing, and deployment flows. Additionally, three supporting workflows are executed, which are configuration and change management, project management, and environment analysis workflows (Kruchten, 2004). the The RUP SE methodology uses a combination of SysML and Unified Modelling Language (UML) (Kruchten, 2004; Ramos et al., 2011).

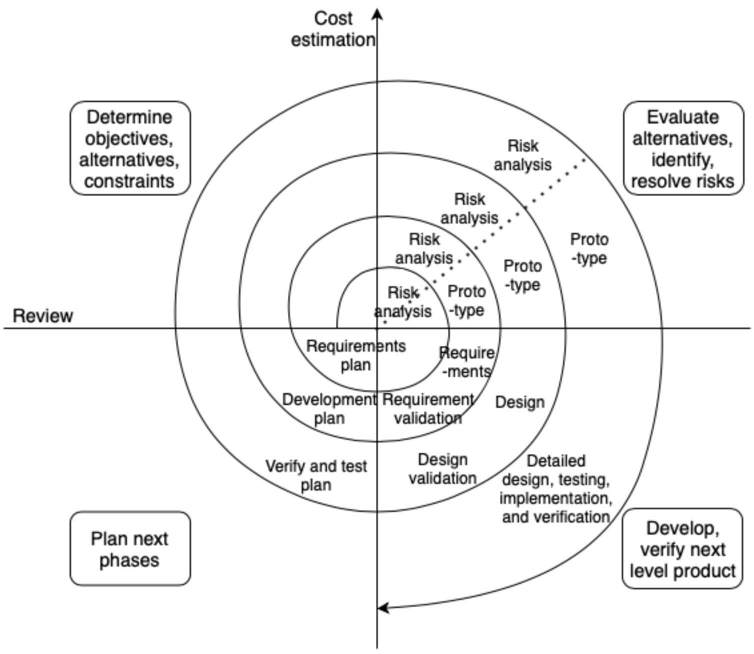


Figure 16 – Spiral model based on Boehm (1988)

Furthermore, Vitech Corporation developed the Vitech MBSE methodology. It is consistent with the concurrent and the incremental approach, which aims at simulations execution of multiple development phases (see figure 17). The work flow of the Vitech MBSE methodology starts with the requirements analysis, goes on to the behaviour analysis, architecture and synthesis of the system. Finally, it finalises with the validation and verification of the design. The languages used within these are System Definition Lanuage (SDL) or Enhanced Function Flow Block Diagram (EFFBDs) (Estefan, 2007; Ramos et al., 2011).

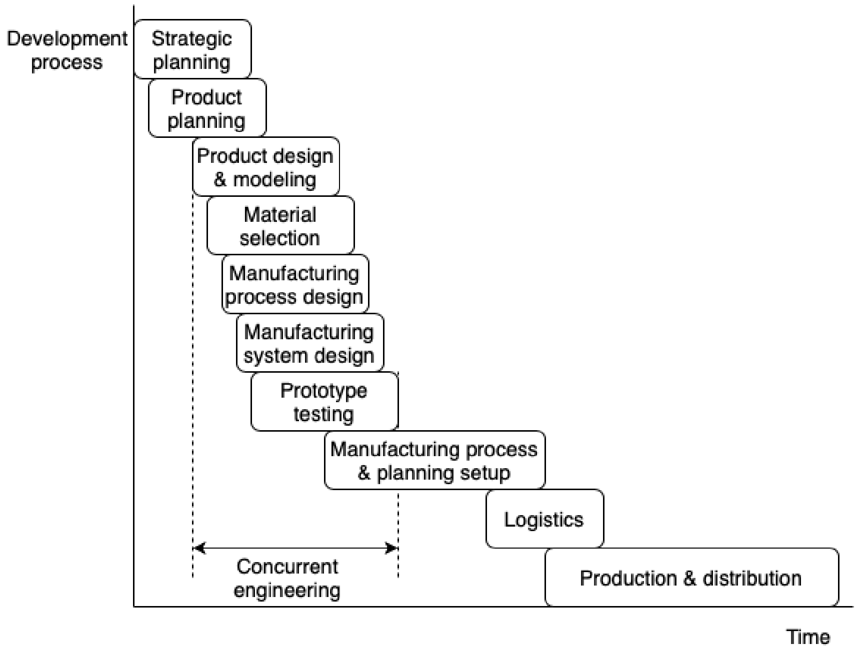


Figure 17 – Concurrent engineering model based on Sohlenius (1992)

Finally, Professor Dori developed the Object Process Methodology (OPM), which uses objects, processes and states to model a system. The methodology has been developed to support a full life-cycle approach. Additionally, it focuses on shortening the development processes, while at the same time delivering quality products at a low cost (Dori, 2002). The OPM system life-cycle contains one important process, namely the system evolving process. The system evolving can be further divided into three separate processes, namely the initiating, developing, and deploying. The first phase is the initiating process in which three sub-processes are executed. The identifying, conceiving and initialising process. In the developing process contains three sub-processes, namely the analysing, designing, and implementing processes of the system. Finally, the deploying process contains the assimilating, using and maintaining, and terminating sub-processes (Dori, 2002). The language is consistent with the main development approach, namely the Objective Process Diagrams (OPDs) and Objective Process Language (OPL) (Dori, 2002; Ramos et al., 2011).

The selection of the methodology depends on the context of the project and/or system. Selection criteria have been developed for selection the right methodology for the a project and/or system.

3.9.2 Modelling tools for MBSE

Several software tools exist that support systems engineers with modelling of the central system model. Some tools are open source and free to use, while others are commercially available. In the previous section five methodologies have been highlighted. Multiple tools are used to model the central model for MBSE, namely IBM Rational Rhapsody, CORE from vitech, and OPCAT used within OPM (Ramos et al., 2011). These tools do all comply with the OMG SysML standards. Additional common used commercially available tools are Eclipse Graphical Editing Framework (Eclipse), Magic Draw (No Magic), Cameo Systems Modeler (No Magic), Enterprise Architecture (Sparx Systems), and Visual Paradigm (Visual Paradigm) (SysML.org, 2020). Free and open source software tools are Modelio, Papyrus, and Capella (Eclipse) (SysML.org, 2020; Eclipse, n.d.). The use of the tools is often based on preference of the systems engineer. However the preference can differ from project to project, since the each tool has different advantages and disadvantages (Rashid et al., 2015).

3.9.3 Modelling language

Each methodology and tool requires the use of a modelling language. It can be seen from section 3.9.1, that multiple modelling languages exist and are used. The modelling languages previously mentioned are SysML, SDL, EFFBDs, OPDs, and OPL (Ramos et al., 2011). The SysML language evolved from the Unified Modelling Language to fit to the purpose, goals and use of MBSE (Reichwein & Paredis, 2011). Overall, the most commonly used language is SysML, however the selection of the language is fully connected to the use of a certain MBSE tool (which are mentioned in the previous section).

3.9.4 Criteria for selecting methodologies, tools and language

Ramos et al. (2011) gave an overview of a selection of methodologies, the tools and language used within MBSE context. From the previous sections it can be seen that those three are closely connected. The selection of a certain methodology mostly determines the use of a certain tool, however a little flexibility can be found within this relation. On the other hand, the tool and the language are intertwined. Selection of a specific tool determines which language has to be used.

Right at the start of a project it has to be decided which methodology, tools and language will be implemented. Within one organisation this decision can change between projects. Overall, in the selection process many aspects have to be taken into account.

Weilkiens et al. (2016) developed a framework for the evaluation and comparison of MBSE methodologies. They defined 27 evaluation criteria, which are divided over 5 groups. The five groups are the following; essential criteria (table 2), efficiency criteria (table 3), usability and experience criteria (table 4), support criteria (table 5), and practical criteria (table 6). The criteria are related to either the methodology, process, language or tool. The evaluation metric can be a list, yes/no, or a scale. The list contains all the optional aspects. The scale is a qualitative scale, with A - fully complaint, B - Acceptable performance, C - limited Applicability, G - generalisation, and X - Not addressed. For full elaboration of the criteria see the article of Weilkiens et al. (2016).

Regarding	Criteria	Evaluation Metric
Process	ISO Standard	List
Process	Framework	List
Language	Philosophy	Yes/No
Tool	Precision	Scale

Table 2 – Essential criteria

Regarding	Criteria	Evaluation Metric
Tool	Perspectives	Scale
Tool	Checking	Yes/No
Tool	Reporting	Scale
Tool	Admin	Scale

Table 3 – Efficiency criteria

Regarding	Criteria	Evaluation Metric
Tool	Navigation	Scale
Tool	Intuition	Scale
Tool	View	Scale
Tool	UI	Scale

Table 4 – Useability and experience criteria

Regarding	Criteria	Evaluation Metric
Methodology	Documentation	Scale
Methodology	Training	Scale
Tool	Support	Scale

Table 5 – Support Criteria

Regarding	Criteria	Evaluation Metric
Language	Language	List
Methodology	Scalability	Scale
Methodology	Scope	List
Methodology	Tailoring	Scale
Process	Consistency	Yes/No
Methodology	Variants	Scale
Methodology	Complexity	Scale
Tool	Connectivity	Scale
Language	Integration	Scale
Methodology	Simulation	Scale
Methodology	Redundancy	Scale
Tool	Reusability	Yes/No

Table 6 – Practical Criteria

Additionally, Friedenthal et al. (2015b) formulated a long list of criteria that can be used for the tool selection, which are the following:

- Metrics support
- Analysis execution capabilities
- Access to information repository of the model
- The performance
- Availability of model libraries
- Life-cycle considerations, such as costs, configuration, installation, etc.
- Ability to customise the tool

4 Framework to integrate Model Based Systems Engineering for variant management

In the previous chapter modularity, platforms, PLE, and MBSE have been highlighted. The next step is to integrate the tools into one process to support variety management practices within AEC industry projects. Before integration can be achieved, multiple decisions have to be made regarding the selection of strategies, methods and tools for the modularity, platforms, PLE and MBSE. Therefore, the chapter starts with a framework to guide the selection process. Thereafter, it shows how the selected methodologies are integrated into one process to support variety management.

The use of the framework should results in one central model, which stores all relevant information regarding requirements, behaviour, structure, and parametrics of a product line. Additionally, the model should be able to contain the variability of a product line. Furthermore, the model should support configuration of single variant of the product line. Finally, the model should be flexible to design and requirement changes.

4.1 The framework

The developed decision framework can be seen in figure 18. The framework includes all variety management strategies and the MBSE methodology highlighted in the previous chapter. In the figure a distinction is made between blocks with dotted lines and solid lines. The boxes with dotted lines means that a decision has to be made relating the topic mentioned in the box. The boxes with solid lines are outputs after the strategies have been implemented in the development process and have to be executed.

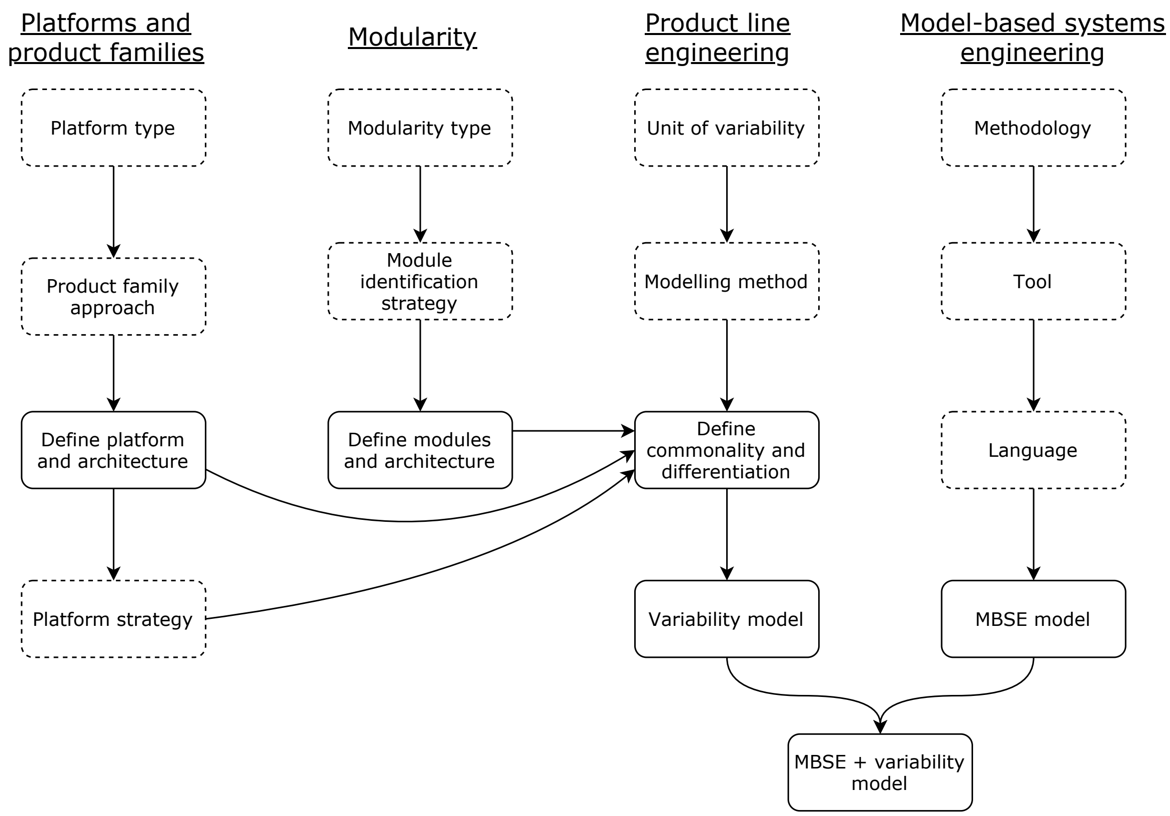


Figure 18 – Implementation framework

Decisions have to be made regarding platforms and product families, modularity, PLE and MBSE. The decision branches can be executed concurrently or one by one. However, the PLE process requires information about the platform, modules and architecture before it is possible to model the variability.

The 'platform and product family' branch start with the definition of the platform type. In the previous chapter six types have been determined, namely product platform, process platform, customer platform, knowledge platform, people platform, and brand platform. It is necessary to determine the type of platform that will be used within a organisation before implementation of the platform strategy. The focus of this research is limited to product platforms only. For this reason, the next step in the decision process is the product family approach. Two types exist, namely top-down or bottom-up. An approach is chosen to decide on the platform design that fits the product portfolio of an organisation. Finally, a platform strategy has to be selected (e.g. niche-specific platform strategy, horizontal platform strategy, vertical platform strategy, or beachhead platform strategy) to determine how and what other market tiers will be targeted with the use of the platform.

Similarly, the 'modularity' branch starts with the selection of the modularity type. Six types have been identified in the previous chapter (section 3.2.3), namely design, process, use, service, supply chain, organisational, and environmental modularity. The research is limited to design modularity, since the focus is on flexibility in design and to shorten system development stages. Next, a strategy has to be selected for the identification of the modules. The strategies that can be selected are clustering methods, graph or matrix partitioning methods, mathematical programming methods, artificial intelligence, or genetic algorithms and other heuristics. Selecting and execution of these processes will result in defined modules that can be used for configuration of system.

The defined platform, modules, architecture and the platform strategy are used as input in the PLE process. The 'PLE' branch starts with determining the unit of variability that will be used within the organisation. A selection has to be made from the following: features, variation points, configuration options, decisions, and calibration parameters. Next, the modelling method has to be selected. The options of modelling methods are feature models, spreadsheets, key pairs, DSL, UML-based representation, decision models, free-text descriptions, product matrix, AOL and ADL. The selected unit of variability could influence the decision and exclude certain methods, which has to be checked before selecting a method. Subsequently, the commonality and differentiation can be determined with the use of the unit of variability and the defined platform, modules and platform strategy. Finally, the determined variability can be made explicit with the use of the modelling method, which results in the variability model.

In the last 'MBSE' branch a selection has to be made regarding the methodology, tool, and language. It is suggested to first select the methodology, since they all differ slightly in their objectives. The methodology influences the choice of the tools. The tools that can be used are Rational Rhapsody (IBM), CORE from Vitech, OPCAT used within OPM, Eclipse Graphical Editing framework (IBM), Magic Draw and Cameo System Modeller (No Magic), Enterprise Architecture (Sparx Systems), and Visual Paradigm. The selection of the tool will determine the language that have to be used. In the next phase, the system can be modelled with the use of the selected methodology, tool, and language.

Finally, the variability model and MBSE model have to be combined to explicitly integrate the variability in the systems engineering model. The final model supports communication about the variants, supports reuse of the assets of the organisation, supports configuration of products, and shows the effects of changes in the requirements.

4.2 Model-Based systems engineering

The previous section shows the decisions that have to be made to be able to integrate MBSE with variety management strategies. In this section the decision process and the final decisions are outlined for the 'MBSE' branch of the framework. Since, the 'MBSE' branch does not require input of the other branches it can be executed simultaneously or separately from the other branches. It is decided to start with this branch, because it is of high importance within the research and might influence decisions in the other branches.

The first decision that has to be made is the methodology that will be implemented. Each of the methodologies mentioned in the previous chapter differ slightly in their main objectives, tool use, support, etc. Weilkiens et al. (2016) developed 27 criteria to compare MBSE methodologies and select tools that suit best to the purpose of the project (see section 3.9.4). The efficiency, usability and experience criteria are mainly focused on the tool that is used. Since, the tool has not been selected yet, those criteria will be left out in the selection process. Exclusion of tool criteria leave 15 evaluation criteria. One of those criteria is variant support and reusability. Since, the research focuses on variant management, the variant support/reusability criteria is of highest importance. For the OOSEM it is stated that it supports reusability of subsystems and components. For the other methodologies variant support/reusability is not mentioned (Estefan, 2007). For this reason, OOSEM is selected as methodology.

4.2.1 Object-oriented systems engineering method

Object-oriented systems engineering method (OOSEM) integrates object-oriented concepts with the general systems engineering process. The main objective is the support of the analysis, specification, design and verification of the system to be designed. Additionally, OOSEM supports flexibility of the system and accommodates evolution with the use of object-oriented concepts. For this reason, the OOSEM is very useful to integrate with variability management strategies, since those strategies require flexibility and evolution possibilities.

The OOSEM makes use of the full systems engineering lifecycle process. This means that all stages of the lifecycle of the system are taken into account during the development of the process. The lifecycle includes the development phase, production phase, deploying phase, operating phase, support phase, and disposing phase. At the end of the systems engineering process a system is delivered, which is verified, validated, and meets the requirements of all the phases.

To achieve the delivery of a verified and validated system, OOSEM uses multiple processes on each level of the design. The processes included are the management process, the system specification and design, the development processes, and the systems integration and verification process. The method is similar to the Vee model (see figure 15). However, within OOSEM the left side of the Vee-model consists out of the management process and system specification and design process for each level. The right side of the Vee-model consists out of integration and verification process for each level of the design. The bottom of the Vee contains the detail design of the components and parts necessary for the higher levels designs.

The management process starts with the planning of the project for each hierarchy level. Thereafter, the plan is used to track and control the execution of the development of the system. The execution is analysed on the basis of the costs, time plan, and performance metrics. Furthermore, the management concentrate on tracking and managing risks related to the development. Additionally, it deals with changing technical designs. Finally, the management adjusts the development processes and designed artefacts to fit to the context of the project (Friedenthal et al., 2015a).

In addition to the management process, the specification and design process is executed as part of the left side of the Vee-model. The specification and design process has as main objective to formulate the requirements of the system. After defining the system level requirements, the process continues to formulate requirements for or relate requirements to the lower hierarchical levels of the system (e.g. subsystems, components). Furthermore, the process defines requirements and test cases that are implemented in the verification and validation process (Friedenthal et al., 2015a).

Finally, the verification and validation process is executed as the right side of the Vee-model. The process first formulates test cases and protocols to show that the designed components, subsystems and system meet the requirements defined in the specification and design process. Thereafter, the tests are executed at the lowest hierarchical level of the system. If the test results comply with the requirements, the process moves to the next hierarchical level. All the separate parts and subsystems will be integrated and are subjected to new tests. This process is repeated till the highest hierarchical level is reached. Finally, the whole integral system is tested to check the validity and verification of the system.

4.2.2 Cameo Systems Modeller from NoMagic

The next step is to select a software tool in which to model the systems engineering practices. The tool has to be be compatible with the OOSEM. The OOSEM methodology uses SysML or UML as the common language. Therefore, the software tool has to be able to work with these languages. Furthermore, it is necessary to include the variability of the product line in the central model within the tool.

In section 3.9.2, the most common used tools have been highlighted. These are IBM Rational Raphsody, Eclipse Graphical Editing Framework, Magic Draw and Cameo Systems Modeler (No Magic), Enterprise Architecture (Sparx Systems), Visual Paradigm (Visual Paradigm), Modelio, and Papyrus. Of these tools, Rational Rhapsody, Eclipse Graphical Editing Framework, Magic Draw, Cameo Systems Modeler and Enterprise Architecture support product line variability modelling. Furthermore, all of these methodologies can be combined with OOSEM.

In this research, Cameo systems modeller from NoMagic is selected to be implemented. The tool is chosen due to availability of the software at the university. Furthermore, NoMagic offers tutorials, courses and self-study materials. Additionally, it is compatible with OOSEM and with product line variability modelling. The interface of Cameo Systems Modeler can be seen in figure 19. The right hand side of the tool is used to model the content of the diagram. Finally, the lower left corner contains a zoom tab, documentation tab, and properties tab for the selected diagram.

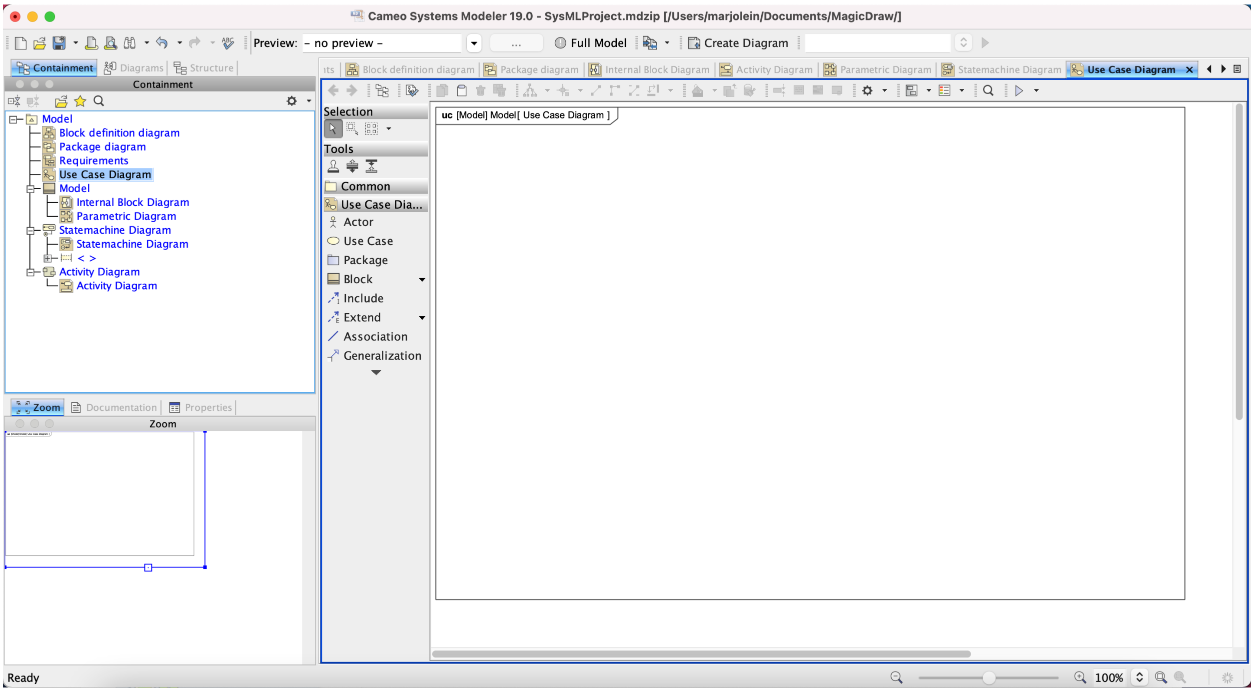


Figure 19 – Interface of Cameo Systems Modeler

4.2.3 Modelling with SysML

Within the OOSEM methodology and in Cameo Systems Modeller SysML and UML are the modelling languages. Overall, SysML is the standard modelling language used within MBSE. SysML is developed to have a general-purpose and graphical language, which is an adaptation of the Unified Modelling Language (UML) used in software engineering (Friedenthal et al., 2015d). As stated before, the system model is divided into system elements. In SysML these elements are a multitude of different diagrams. The diagrams used are package diagrams, requirement diagrams, activity diagrams, sequence diagrams, state machine diagrams, use case diagrams, parametric diagrams, block definition diagrams, and internal block diagrams. The SysML diagrams can be seen in figure 20 (Ramos et al., 2011; Friedenthal et al., 2015f).

As mentioned in section 3.6.2, the central model contains information related to the behaviour, structure, parametric, and requirements of the system. The diagrams are each related to either one of those four pillars. In the requirements diagram the requirements of the system are documented. In the parametric diagram the parameters of the system design are document. However, the behaviour and structure are less straight forward and use multiple diagrams to store the information.

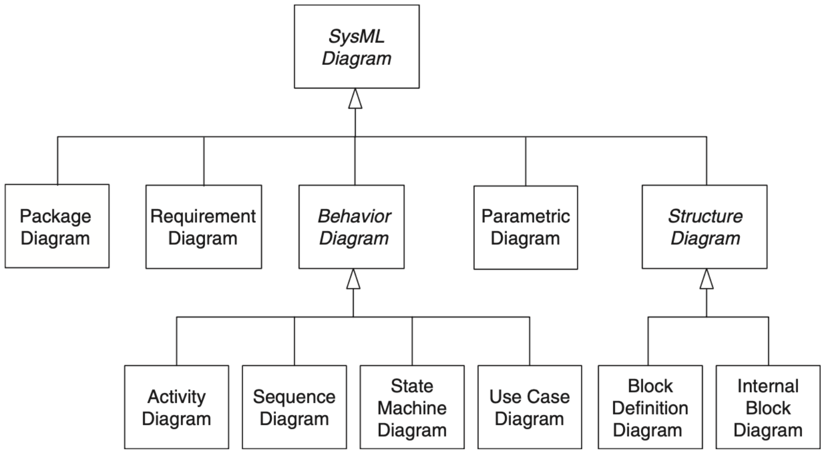


Figure 20 – SysML diagrams from Friendenthal et al (2015f)

For the behaviour of the system, activity, sequence, state machine and use case diagrams are used. Furthermore, the structure of the system is captured in block definition diagrams, and internal block diagrams. Finally, in figure 20, it can be seen that one diagram is not mentioned, which is the package diagram. In the package diagram the organisation of the model is documented (Friedenthal et al., 2015f). A simplified example of a system is documented in SysML and can be seen in figure 21.

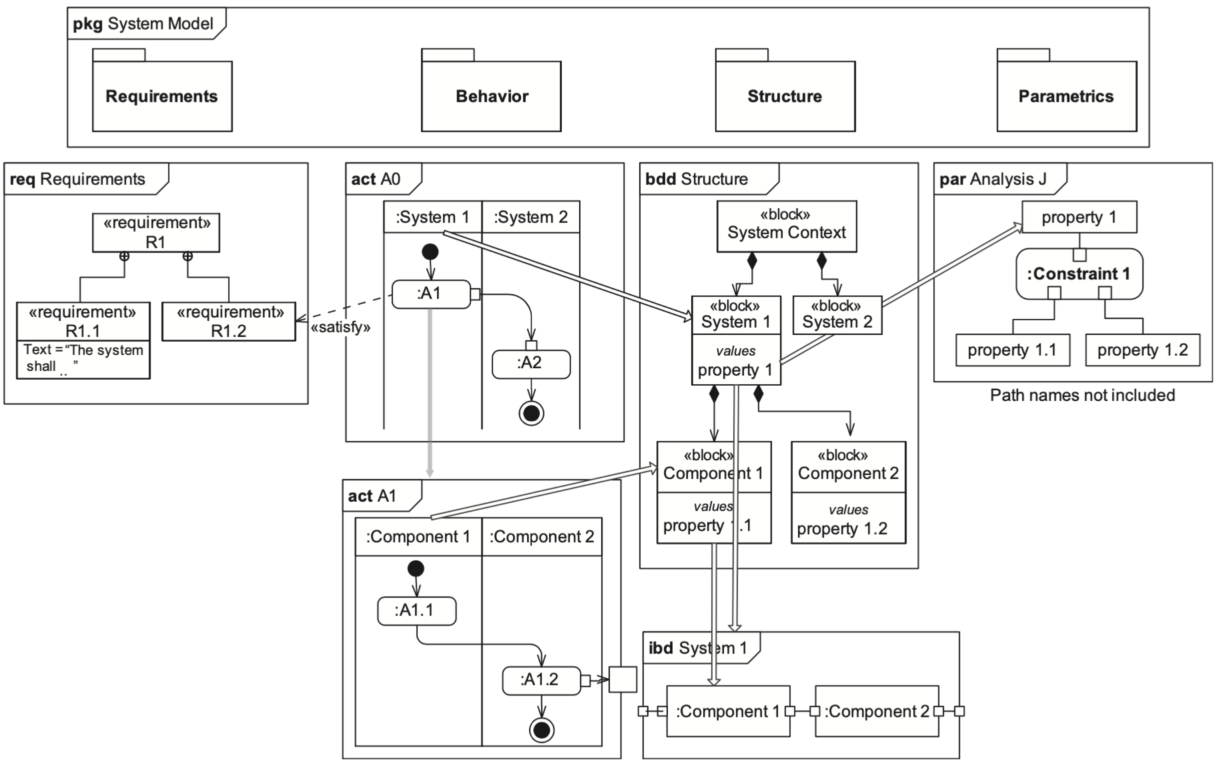


Figure 21 – Central model documented in SysML diagrams from Friendenthal et al. (2015f)

4.3 Product line engineering

In the previous section OOSEM, Magic Draw, and SysML are chosen as methodology, tools, and modelling language for MBSE. The decisions for the 'PLE' branch are outlined in this section. First, a modelling method is chosen. The method is selected before the unit of variability to match it with the chosen MSBE tool and methodology, such that it is possible to integrate PLE and MBSE in one single model. The language used within MBSE process is SysML, which evolved from UML. For this reason, it is decided to choose a variety modelling method that is UML-based to match the language of the two methods. The UML-based method selected is the orthogonal variety model (OVM).

The orthogonal Variety Model (OVM) makes use of a separate variability model in which the the variation of the product lines are defined and documented. The method uses variation points as the unit of variability. The OVM is able to connect the variability model with other models used within the development process, such as feature models, use cases, design models, component models, and test models (Lauenroth & Pohl, 2005). Since, multiple of these models are also included in the SysML language, a connection can be made between MBSE and PLE. For this reason, the OVM is a suitable variety modelling method.

4.3.1 Orthogonal Variety Model (OVM)

The OVM uses variant points (VP) and variants (V) to denote the variability of a product line (see figure 22). The variation point denotes where (e.g. which subsystems, features, or components) could vary between products. The variants show how the products vary.

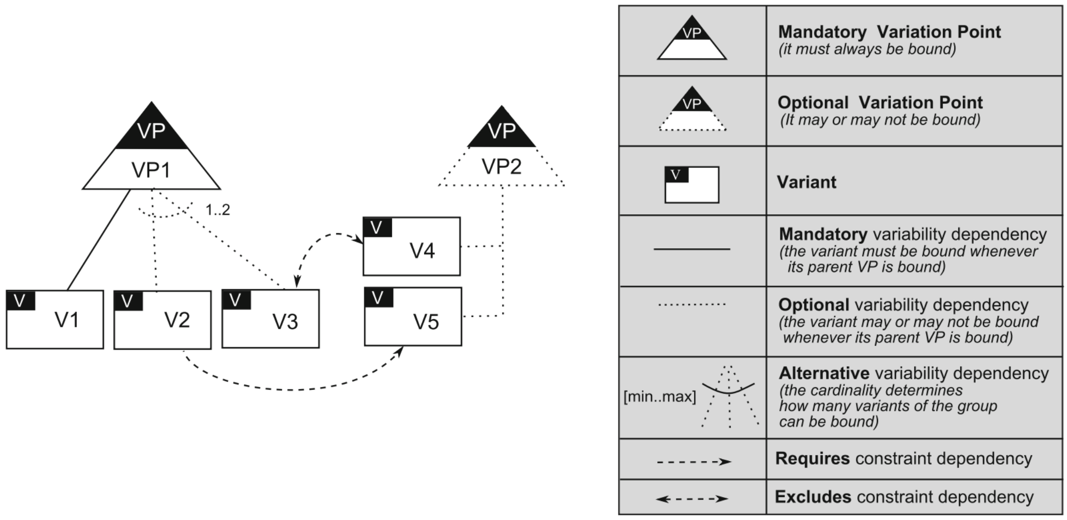


Figure 22 – The orthogonal variability model annotation from Lauenroth and Pohl (2005)

Furthermore, dependencies play an important role in OVM to determine the relations between the variant points and the variants. Three types of dependencies exist, namely variability dependencies, artefact dependencies, and constraint dependencies. The variability dependencies determine how variants are related to a variant point. Three types of variability dependencies exist, namely with a mandatory dependency, an optional dependency, and alternative choice. The mandatory dependency means that a variant has to be integrated in the system. The optional dependency means that a variant can be integrated in the system if the organisation or customer wishes. The alternative choice dependency groups multiple optional variants and gives a minimum and maximum amount of optional variants that can be integrated in the system (Lauenroth & Pohl, 2005).

An example is show in figure 23. In the example the variability of a residence is modelled. When a customer wishes to design and buy a residence, multiple decisions have to made before the final system can be build. In figure 23, it can be seen that the variantion points kitchen, number of bedrooms, exterior, bathroom and living room have a mandatory dependency with the residence, which means that the these have to be included in the final product. For each of these variation points a variant has to be chosen. For the exterior design and living room the variant point contains two variants. The variants are connected by an alternative choice dependency with a minimum and maximum equal to 1, which means that the one variant has to be selected to be included in the residence. The kitchen and bathroom variant point similarly are connected to two variants by a alternative choice dependency. However, for these variant point the minimum is 1 and no maximum is given. For this reason, both variants could be included according to the customer wishes. Finally, the variant point 'number of bedrooms' contain three mandatory relationships and one optional. Each residence has to include at least three bedrooms, however the fourth bedroom is optional.

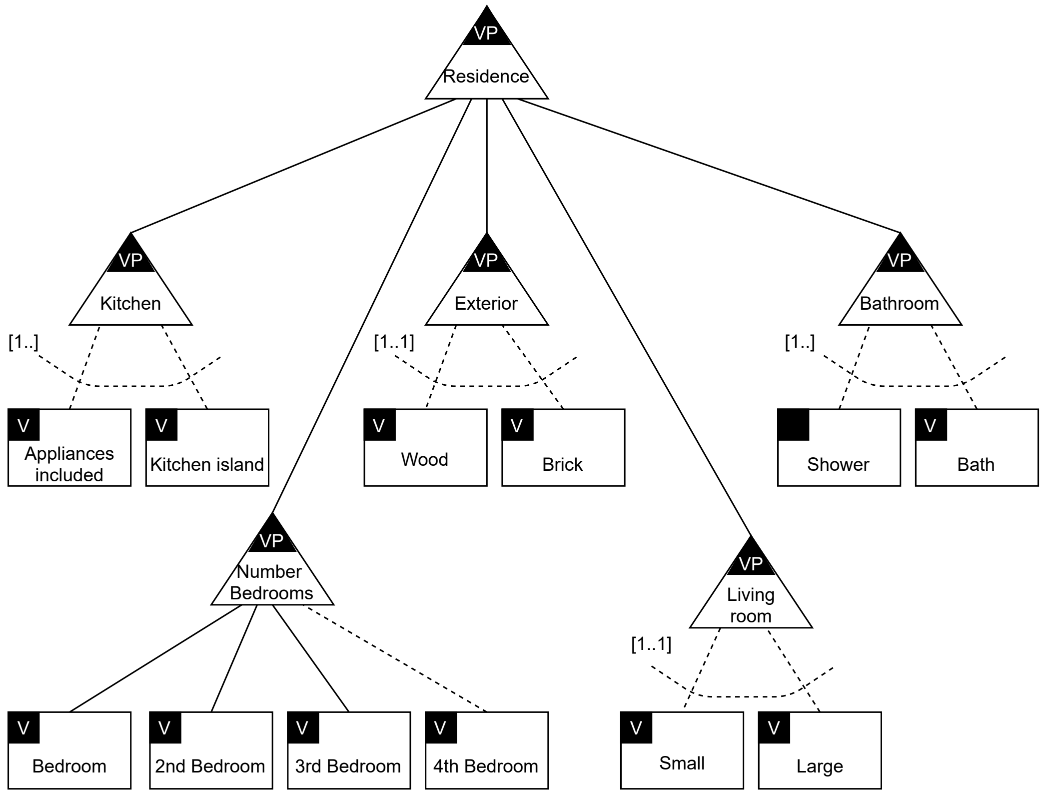


Figure 23 – OVM for a residence

Additionally, the constraint dependencies define the constraint between variants and variants, between variants and variant points, and between variant points and variant points. Two type of constraint dependencies exists, namely require constraints and exclude constraint. A requirement constraint means when a certain variant or variation point is included in the system design it requires the integration of a specific variant or variation point. In figure 22, the relation is denoted with an one way arrow from V2 to V5. When V2 is selected, V5 has to be integrated in the system. The excluding constraint means that specific variants or variation points can not be included in the design when certain variants or variation points are selected for the design of a system. In figure 22, this relationship is denoted with a two way arrow between V3 and V4. When V3 is selected, V4 can not be integrated in the design, this relationship goes the other way around as well (Lauenroth & Pohl, 2005).

Finally, the artefact dependencies are used to connect the variety model with the design of the system. The developed artefacts range from requirements, hardware, software, etc. Dependencies have to be formulated to link the variants and variation points to the artefacts they influence in the actual design of the system. Each variant has to be linked to minimally one artefact, however an artefact does not have to be related to any variant. The variation points can, but do not have to be, related to an artefact, and vice versa. The artefact dependencies support the traceability between the variety model formulated for the whole product line and a specific developed system as a member of the product line (Lauenroth & Pohl, 2005).

To model the variability, a tool has to be selected. Two type of tools exist that can be combined with Cameo Systems Modeller, which are Gears from BigLever and Pure::Variants from Pure Systems. Either one of the tools can be used to model the variability and integrate them into Cameo Systems Modeller. Within Cameo Systems Modeller the variability can be denoted with the use of variation points and variants related to the requirements, hardware, software, etc. Furthermore, specific system variants can be configured within Cameo Systems Modeller based on the defined system variants within the BigLever or Pure::Variants. In this research it is decided to use Pure::Variants due to availability of the software tool.

4.4 Platforms and modularity

Multiple decisions have to be made regarding platforms and modularity. First of all, a type of platform and the type of modularity that will be implemented, has to be selected. However, in section 4.1, it is stated that the research limits its scope to one type of platform (i.e. product platform) and one type of modularity (i.e. design modularity). Therefore, the selected platform type that will be implemented is a product platform and the selected modularity type that will be implemented is design modularity.

Secondly, a product family approach and platform strategy have to be selected. The product family approach is selected based on the right fit to the product portfolio of an organisation. Besides, the platform strategy has to match the business strategy of an organisation. Therefore, the selection is context dependent and will be influenced by the organisation. Therefore, the decision has to be made right at the start of the implementation of the framework within a specific organisation or project.

Finally, a module identification method has to be selected. All methods have their advantages and disadvantages. Therefore, the method should be selected based on the project, preferences and previous experiences within an organisation. Thus, a method is selected when the framework is implemented within a specific organisation or project.

5 Case study

In the previous chapters a framework has been developed that supports the selection of tools and methodologies to model MBSE, PLE, and variability. In this chapter the framework and the central variability and MBSE model will be integrated in a practical context to demonstrate it's use (demonstration phase of DSR, see chapter 2). A case study is executed to show the use and effect of the framework. This chapter starts with an overview of the case specifics. Thereafter, the use of the framework and central model are explained. Finally, the product line of Plegt-Vos is modelled with the use of the central model.

5.1 Case study details

As stated in chapter 2, the case study is executed at Plegt-Vos bouwgroep B.V.. Plegt-Vos bouwgroup is a 115 year old family-owned company. The organisation operates in the north, east, and central parts of the Netherlands. With it's 450 employees and approximately €250 revenues, it is a medium-sized organisation. Since 2013, Plegt-Vos focuses on construction of buildings with the use of modular units and prefab subsystems. Consequently, Plegt-Vos is able to shorten their development and construction times. Additionally, they are still able to offer a unique solution and design for their costumers[1].

In chapter 2, the case study organisation has been selected. At the organisation a suitable system has to be chosen to be modelled with MBSE and PLE. An existing system has been selected due to the time constraint of the research, while it will still show the use, benefits and drawbacks of the framework. The selected system is a the hallway core of a terrace house. The hallway core is chosen, because Plegt-Vos uses it as a platform to build up houses around it (see appendix G.1). Thus, the hallway module plays a central role in the development of systems in the organisation. Furthermore, the hallway core contains many technical installations required for a residence. Consequently, it provides a certain level of complexity to model, although not too complex for the limited time period of the research. Therefore, modelling of the hallway core will give a good illustration of implementing MBSE and PLE in the organisation.

The case study starts with the analysis of the modularity, platform, and architecture of the system. Thereafter, the technical information from the system provided by Plegt-Vos is used as input for MBSE and PLE. The developed framework (chapter 4) will be used to determine the process.

5.2 Use of the framework

The main objective of the research is to implement MBSE within the AEC industry to support variety management practices within the organisation. A framework has been developed in the chapter 4, which guides the selection process of methodologies and tools that result in a model that combines product line variability with MBSE. For the final model, multiple requirements have been formulated, which can be found in 3.9.4. With the requirements in mind, a methodology, tool, and language have been selected for Model-based systems engineering. Additionally, the unit of variability and modelling method have been selected based on the requirements.

The selected MBSE methodology is OOSEM, the selected tool is Cameo Systems Modeler from NoMagic, and the modelling language is SysML. Furthermore, the chosen units of variability are variation points. Accordingly, the variability modelling method is OVM, which will be modelled in Pure::Variants from Pure Systems. The orthogonal variability model can be directly integrated into Cameo Systems Modeler to connect the variation points to the physical subsystems and parts.

An existing modular system has been selected for the case study. For this reason, it is not necessary to determine which platform and modularity type, product family approach and/or module identification strategy has to be implemented. However, the system will be analysed to identify the architecture, modules and (possibly) the platform. Additionally, the variability of the product family has

[1] Kimberly Camu (2019). Flexwonen is een antwoord op het groeiende woningtekort. Retrieved from https://vastgoedjournaal.nl/news/42810/-lsquo-flexwonen-is-een-antwoord-op-het-groeiende-woningtekort-rsquo-

been defined by Plegt-Vos, therefore it is not necessary to define the commonality and differentiation between multiple systems.

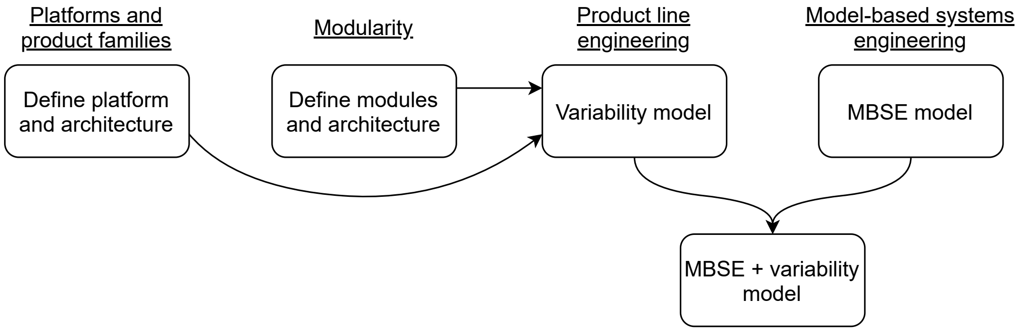


Figure 24 – Steps of framework executed in the case study

Overall, many steps of the framework have been executed. The remaining steps of the process can be seen in figure 24. The case starts with the identification of the architecture, modules and (possibly) platform of the residence system. Next, the variability in the residence system will be identified and modelled within Pure::variants. Thereafter, the system will be modelled within Cameo Systems Modeler. The defined architecture, modules, platform, and variability will be used as input for the MBSE model. Additionally, more detailed information from Plegt-Vos (e.g. dimensions, requirements, etc.) will be used as input to the model.

5.3 Identification of modules, platforms and related strategies

As mentioned in the previous section, an existing terrace house of Plegt-Vos is selected for the case study module. Furthermore, Plegt-Vos implements modularity strategies in their organisation and development process. For this reason, it is not necessary to define modules and/or a platform for the reference residence, since Plegt-Vos has already defined it for the system.

The defined modules for the terrace house are the construction structure, the living room, the kitchen, the hallway on the ground floor, the hallway on the first floor, the master bedroom, two additional bedrooms, and the attic. Multiple of these subsystems consist of sub-subsystems and/or components. However, the main focus of the case study is the detailed analysis of the hallway core, which is used as the platform for the systems developed at Plegt-Vos.

To summarise, the platform of the system is the hallway core. Furthermore, the system contains six other modules, namely the living room, kitchen, bathroom, master bedroom, bedroom two and bedroom three. The architecture of the system contains three floors (including the ground floor). The living room, kitchen, and hallway ground floor are located on the ground floor. The bathroom, master bedroom, bedroom 2, bedroom 3, and the hallway first floor are located at the first floor. Finally, the attic is located on the second floor.

Plegt-Vos modules/platform

Platform	Hallway core ground floor Hallway core first floor Attic
Modules	Living room Kitchen Bathroom Master bedroom Bedroom 2 Bedroom 3 Structure

Table 7 – Overview of the modules and platform of Plegt-Vos

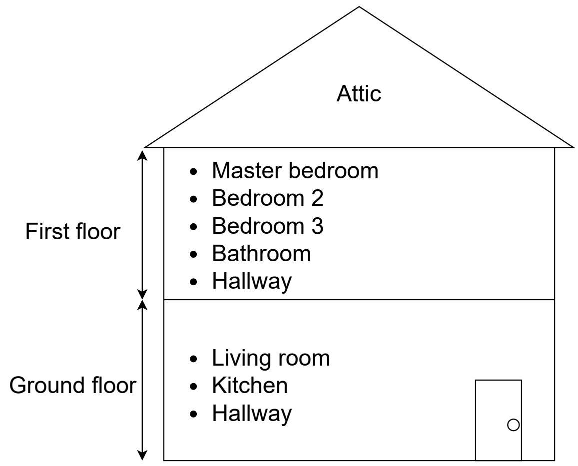


Figure 25 – Architecture of the PV system

5.4 Variant modelling with OVM

To market the terrace house to a wide range of different customers, Plegt-Vos offers variety in the configuration of the system. The PMC department of the company formulated a so called "shopping list" with customer wishes for the system. The list includes the optional features that have to be offered and several requirements for the system. In this research the "shopping list" is referred to as the customer wishes document and it can be found in appendix G.2. Based on the customer wishes the variation points and the variants are determined. The overview of variations points and variants can be seen in appendix G.3.

The results of the analysis of the customer wishes is used as input for modelling the variety of the system in Pure::variants. Furthermore, all relationships between the variant modules and the variant points are included. The variety model for the terrace house can be seen in appendix G.4.

Additionally, a variant can be defined in Pure::Variants by selection of mandatory, optional and alternative variant modules for the system. The modules can be selected by checking the boxes of the variation points and variant modules that has to be included. Pure::Variants shows if all necessary decisions are made, otherwise it will give a warning that it has incomplete information. For this reason, variants of a system from the product line will always be fully defined.

5.5 Model Based System Engineering in Cameo Systems modeler

In the previous chapter, OOSEM has been chosen for the MBSE methodology. OOSEM has specific phases, however it has no specific steps to model a system in a modelling tool. The user is flexible in organising the model and the use of different diagrams to their own needs and wants. However, the high flexibility and the complexity of SysML could cause trouble in the later stages of the project or the system life-cycle. Furthermore, it could negatively influence the reuseability of the central model. Therefore, it is necessary to define how a system model should be built up and which SysML diagrams have to be used for different purposes.

Since, OOSEM does not have a step-by-step model approach, multiple approaches have been developed to guide the systems engineer through the modelling process (Morkevicius et al., 2017; Mhenni et al., 2014; Casse, 2017). Mhenni et al. (2014) developed a methodology with two phases, namely a black-box analysis phase and a white-box analysis phase. In the black-box analysis phase the System of Interest (SoI) is seen as a black-box, which is analysed in the context of it's application environment. The black-box analysis is executed in 9 steps, which are each modelled with specific SysML diagrams. In the white-box analysis the internal requirements, structure, behaviour, and parameters are studied. The white-box analysis is executed in 7 steps with corresponding SysML diagrams. An overview of the steps can be seen in table 13 in appendix A.

Casse (2017) proposed an approach with four phases, namely the operational analysis, system requirements analysis, logical architecture design, and physical architecture design. Each of these phases makes use of multiple steps and diagrams to model the system. Within this approach the system is seen as a black-box in the first two phases and as a white-box in the last two phases. The operational analysis is executed in 7 steps and with 7 diagrams, the system requirement analysis phase is completed in 4 steps and modelled in 4 diagrams. The logical architecture design requires 7 steps and 7 corresponding diagrams. Finally, the physical architecture design phase contains 4 steps and uses 4 different diagrams (see table 13, appendix A).

Morkevicius et al. (2017) developed the MagicGrid approach. MagicGrid is a step-by-step approach to model a system in Cameo Systems modeler. The framework uses on the x-axis the four pillars of MBSE, namely requirements, structure, behaviour and parameters. The y-axis contains the different phases of modelling process, which are the problem domain analysis phase, solution domain analysis phase, and implementation domain phase. Furthermore, the problem domain is split up in a black-box and white-box analysis. The framework can be seen in figure 26. The steps are also documented in table 13 in appendix A.

The three different approaches mentioned above have overlapping phases and steps. First of all, all three methodologies make use of a black-box and white-box approach. Secondly, the steps included within the black-box phase are similar. Each approach studies the context in which the system

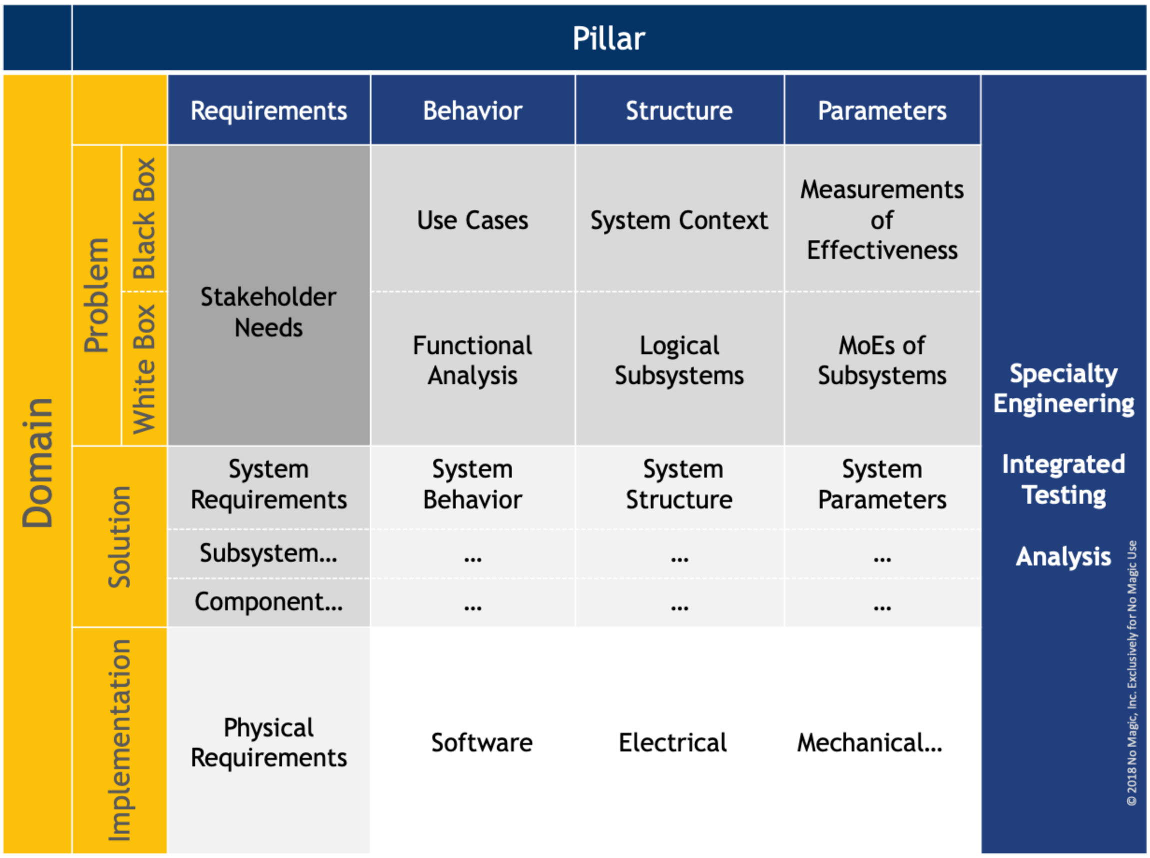


Figure 26 – MagicGrid framework from Aleksandraviciene and Morkevicius (2018)

is used, including the internal stakeholders and interfaces. Furthermore, they all study the usage of the system in the form of use cases. Third, the white-box analysis phase has similarities in modelling steps. Each approach includes the development of a logical architecture (e.i. abstract architecture showing the functions of the system) and a physical architecture (e.i. concrete architecture showing the technologies delivering the functions of the system). Furthermore, they include the allocation of requirements to specific subsystems and/or components. Furthermore, the functionality of the system plays an important role. Throughout the black- and white-box phase the functionality is studied and linked to the system and subsystems. Additionally, each approach contains steps to define the logical, physical architecture, and link the two. Finally, each approach includes steps to include the parameters to the system, subsystem and components.

On the other hand, multiple differences exist. The most distinguishing difference in the approaches is the extra phases within MagicGrid compared to the other two approaches (see figure 26). MagicGrid includes an implementation phase which includes the physical designing of the system, subsystems, and components. Furthermore, it contains an integration and verification step of the final design of the system. Each of the phases of the MagicGrid correspond to the phases of OOSEM. Furthermore, the MagicGrid solution phase includes three depth levels (system, subsystem and components) compared to the two depth levels (system and subsystems) of the other two approaches.

The case study focuses on the modelling of an existing design of the terrace house of Plegt-Vos. The analysis level of the design is mainly focused on the subsystem level, however the sub-subsystem level and/or component level are taken into account for the hallway core modules. The system model will be used to analyse check and verify the requirements and make changes in the design on subsystem and/or component level. For these reasons, the MagicGrid approach seems a better fit for the modelling of the system and is thus selected as modelling approach.

5.6 The modelling process

As mentioned in the previous section, the MagicGrid approach is selected for modelling within Cameo Systems Modeler. The suggested approach is to start at top level, problem domain - black box analysis, and work down to the implementation domain. Within each domain of the approach the requirements, behaviour, structure, and parameters have to be formulated and/or designed. The approach is formulated as a step-by-step process, however should be seen as iterative. Information within the top level domains can always be updated, changed or added on to. When changes are made within the top-level the lower levels will be impacted, thus have to be revised and updated.

MagicGrid is usually used in the development process of a new system, however for the case study an existing system has been selected. Therefore, the problem domain section does not play an important role. In the problem domain analysis phase, the requirement pillar has been included, but the other three pillars are excluded from the case study.

Furthermore, in the solution domain the requirements, structure, and parameter pillar are included. However, the behaviour pillar has been excluded. The functions offered by a residence are often not a dynamic mechanisms. To illustrate, the main function of a house is; offering a safe living environment, which protects from the weather and intruders. To offer this function, the residence is providing a static behaviour (or function) provided by static components. On the other hand, the main function of a car is to transport their user from point A to B. Driving of the car is a dynamic behaviour (or function), during which many components dynamically interact. For this reason, the behaviour pillar for the residence does not play a crucial rule in the design and is excluded from the modelling of the system.

Finally, the implementation phase already has been executed, since the residence has already been designed and developed. The mechanical, electrical and software aspects have been designed for the components and subsystems. Additionally, the integration of the whole system has been verified and validated. Therefore, the implementation phase does not need to be executed and is excluded from the case study.

5.6.1 Problem domain - black & white box analysis

The case study started with the black box analysis of the problem domain. In this phase the stakeholder requirements have been formulated. Initially the customer wishes formulated by the PMC department have been used as input (see appendix G.2). The customer wishes contain information about the variability and general requirements that the residence has to comply with.

Next, the system context of the residence has been defined, which shows what for (e.i. purpose) and by whom/what (e.i. stakeholders) the system is used. The main system purpose is the usage of the residence by the owners. The stakeholders that are involved during the useage of the system are the owners of the house, the weather, water, electricity, air and the control. Additionally, the interfaces between the stakeholders and the residence are defined. The system context is modelled in an internal block diagram (ibd) to show the internal interactions of the context (see in figure 27). The interfaces are modelled as proxy ports, which are non-physical points between the stakeholders and the residence to visualise the relationships.

5.6.2 Solution domain

In the MagicGrid approach, the solution domain starts with the formulation and analysis of the requirements, structure, and behaviour on the systems level. Thereafter, it does the same for the subsystem level, sub-subsystem level and component level. The MagicGrid approach is used as guidance for the modelling in the solution domain. First, the requirements and structure are formulated and modelled on a system level. Thereafter, the requirements and structure are modelled for the subsystem, sub-subsystem and component level. Afterwards, the parameters are added to each depth level. Finally, the parameters are connected to the requirements. The process in executed iteratively, which means that changes, additions and updates on the (sub)system level would initiate another modelling cycle.

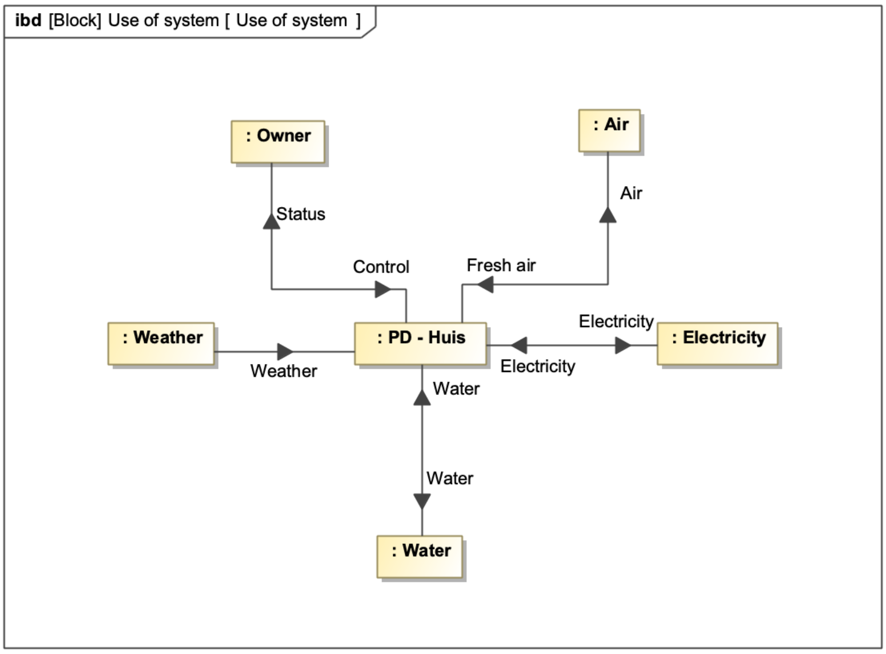


Figure 27 – System context of the usage of the Plegt-Vos residence

System requirements and structure

The first step in the solution domain is the translation of the stakeholder needs into quantified system requirements. The quantification of the system requirements makes it possible to verify and validate the design parameters of the system and/or components and subsystems. The system requirements are derived from the stakeholder needs (see Appendix G.5). Each of the stakeholder needs has to result in minimum one system requirement, but could also be translated into more then one requirement.

After formulating the system requirements, the system structure has to be modelled. The system architecture and the modules had already been defined by Plegt-Vos (see appendix G.1). The modules included in the terrace house are a building structure, living room, kitchen, bathroom, master bedroom, bedroom 2, bedroom 3, attic, hallway on the ground floor, and hallway on the first floor. The living room, kitchen and hallway are located on the ground floor. The bathroom, master bedroom, bedroom 2, bedroom 3 and hallway are located at the first floor. The attic is located on the second floor directly underneath the roof. This information is used as input for the model in CSM. The system structure is modelled in a block definition diagram (bdd) (see figure 28).

System interfaces

In the previous section the subsystems and system structure have been defined. Next, the interfaces of the system have to be defined and modelled. The system has internal and external interfaces. The external interfaces are the material and information flows from the external environment to the system. The external interfaces are the control of the residence by the user (iControl), the weather (iWeer), the water (iWater), the electricity (iElectriciteit) and fresh air (iLucht) (see figure 27). The water, electricity and control enter the hallway core on the ground floor and from there are connected to the other subsystems that make use of it. The air enters and exists the residence in the attic subsystems and is distributed throughout the house. Finally, the weather has influence on the external structure and walls of the residence.

The internal interfaces are the physical connections and/or material and information flows between the subsystems of the system. To define the interfaces between the subsystems, the connections between the subsystems are analysed in the design of the residence (see appendix G.7). The subsystems can be connected by walls, water pipes, air pipes, sewage systems and/or electricity cables. An interfaces between subsystems is defined when the subsystems have minimum one connection. The interfaces between the subsystems are named according the following rule: iSubsystem1-Subsystem2. An overview of the interfaces is shown in table 8.

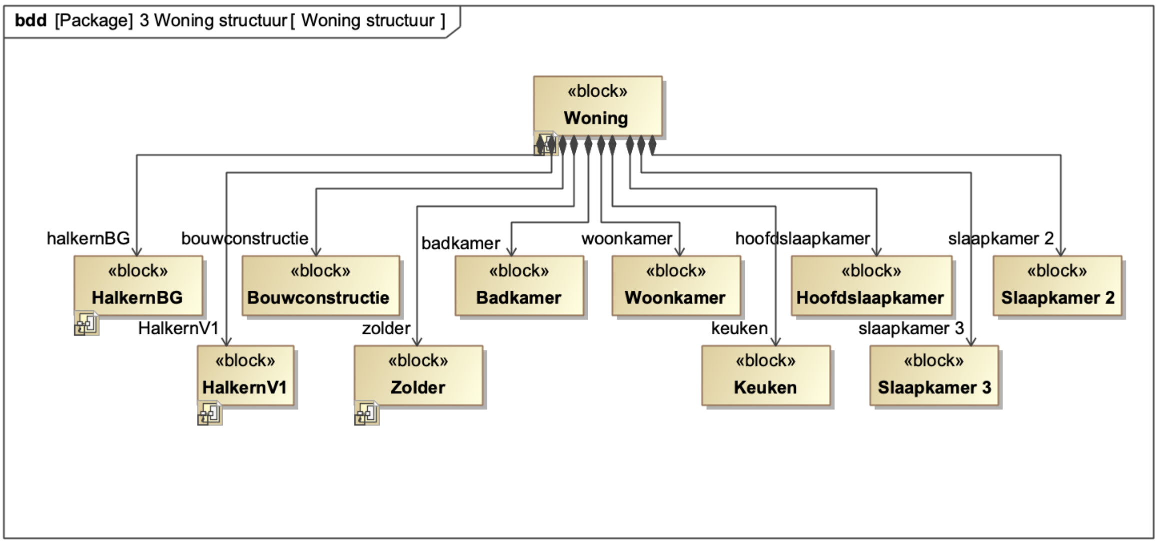


Figure 28 – System structure of the terrace house of Plegt-Vos

Subsystem	**Connects to**	**Interface**
Construction structure	Hallway GF	iBinnen-Buiten
	Hallway F1	iBinnen-Buiten
	Kitchen	iBinnen-Buiten
	Living room	iBinnen-Buiten
	Master bedroom	iBinnen-Buiten
	Bedroom 2	iBinnen-Buiten
	Bedroom 3	iBinnen-Buiten
	Bathroom	iBinnen-Buiten
	Attic	iBinnen-Buiten
Hallway GF	Kitchen	iHalkernBG-Keuken
	Livingroom	iHalkernBG-Woonkamer
	Hallway F1	iHalkernBG-HalkernV1
Hallway F1	Bathroom	iHalkernV1-Badkamer
	Master Bedroom	iHalkernV1-Hoofdslaapkamer
	Bedroom 2	iHalkernV1-Slaapkamer2
	Bedroom 3	iHalkernV1-Slaapkamer3
	Attic	iHalkernV1-Zolder
Kitchen	Hallway GF	iHalkernBG-Keuken
Livingroom	Hallway GF	iHalkernBG-Woonkamer
Master Bedroom	Hallway F1	iHalkernV1-Hoofdslaapkamer
Bedroom 2	Hallway F1	iHalkernV1-Slaapkamer2
Bedroom 3	Hallway F1	iHalkernV1-Slaapkamer3
Attic	Hallway F1	iHalkernV1-Zolder

Table 8 – Interfaces between subsystems

The internal and external interfaces of the system are modelled in CSM. A internal block diagram (ibd) is used for the modelling, which can be seen in figure 50 (see appendix G.6). The water, air, electricity, control, and weather are modelled as proxy ports, since these are material and/or informational flows between the subsystems and not physical parts. The remaining interfaces are modelled as full ports, since those are physical components between the subsystems. From figure 50 (see appendix G.6), it can be seen clearly that the hallway modules are the core modules, which distribute the air, water, and electricity throughout the residence by means of air pipes, water pipes, sewage system and/or electricity cables to all remaining subsystems.

Hence, the two hallway modules (hallway module ground floor and hallway module first floor) provide all the other subsystems with air, water and electricity, where necessary. These modules also contain the many components and/or sub-subsystem that are necessary to transport the water, air and electricity. As mentioned in section 5.3, the hallway modules can be seen as the platform of the system design. Therefore, the company aims to keep the design of these modules rigid. Design changes within these modules should be prevented or minimised.

Subsystem requirements, structure and interfaces

Subsequently, the requirements of the subsystem have to be formulated and the structure has to be modelled. The subsystem requirements are derived from the system requirements. Studying the system requirements, showed that the system requirements contained quantified requirements for the subsystems. Therefore, it is decided to not make separate requirement tables for the subsystems.

Additionally, to model the structure of the subsystems the components have to be identified. The sub-subsystems and components are identified on the basis of the drawing in appendix G.1 and in consultation with the innovation manager of Plegt-Vos, Berri de Jonge. As mentioned before, the hallway modules are seen as the platform of the system. For this reason, the hallway will be modelled in detail with all sub-subsystems and components included. On the contrary, only the variant components and/or sub-subsystems defined in section 5.4 are modelled for the remaining subsystems. The subsystem requirements and the subsystem structures can be seen in Appendix G.8.

The analysis of the drawing of the system showed that the sub-subsystems and components of the hallway modules ground floor are the toilet space, the fuse box space, the staircase, the staircase closet, the traffic space, the technical installation space and the pipe space. The sub-subsystems and components of the hallway first floor are the staircase to the attic, the traffic space, the technical installation space, and the pipe space. The sub-subsystems and components of the SKID are the heat pump, the PV inverter, the hot water boiler, and the heat recovery system.

Up to this point, general interfaces are defined between subsystems if a connection exist. For each of these interfaces the components and/or sub-subsystems have to be identified that make the physical connection between the subsystems. To do so, each general interface is studied and the parts that make up the connection are modelled as components of the interfaces. The parts that make up the interfaces can be related to the water pipes, air pipes, sewage system, or structural (walls, doors, etc.) The interfaces and their components are documented in appendix G.10.

The defined interfaces are the external interfaces of the subsystems. For some subsystems the internal structure with their sub-subsystems and components are included in CSM. The subsystems with an internal structure also have internal interfaces between their sub-subsystems and/or components. However, the case study focuses on the system and subsystem level. For this reason, the internal interfaces of the subsystems are not included in the model.

Variety modelling of the system in CSM

In section 5.4, the variant points and variability has been determined for the terrace house. The variability is modelled in Pure::Variants, which can be linked to the model in CSM. Currently, the system modelled in CSM is a single system. A single system does not include multiple variant modules, but is a single configuration of a system included in the product line. Therefore, the CSM model has to be extended from a single system model to a 150% model. A 150% model includes all variant points and variant modules of the whole product line.

Consequently, the variability has to be modelled in CSM. The variant points and the variant modules are added to the system and/or subsystem structure in the block definition diagrams. The variant points are denoted to existing blocks or added as blocks to the relevant subsystems in the model. The variant modules are included as components of the variant point, which are modelled as blocks.

Addition of the variant points and variant modules to CSM enables the connection between the Pure::Variants model and CSM. Each of the variant points from Pure::Variants are connected to the according blocks in CSM. The same goes for the variant modules. The variant point and variant blocks in the CSM model are indicated with a 'V' (see figure **??** in appendix G.11 for an example).

Additionally, a variant formulated in Pure::Variants can be selected in CSM. The system model shows which variant modules are included and excluded from the design of the selected variant of the system (see figure 83 in appendix G.11).

System, subsystem and component parameters
After the requirement formulation and structure modelling of the system, subsystem and components, the parameters of the system have to be added. The parameters are first added to the system level, then to the subsystems level, and finally to the sub-subsystems or components level. The parameters are defined based on the information from the technical drawings of the terrace house (see appendix G.1). Each dimension and location of the subsystem is added to the model in CSM.

Additionally, constraints are added to the model. By means of constraints equations, parameters can be calculated based on other parameters of the system. For example, if the width of the house and the width of the hallway are available, the width of the kitchen can be calculated (see figure 29). In the modelling process of the parameters it is aimed to maximise the number of dependent parameters. By doing so, a change in one of these values, instantly updates all other parameters. The dependency increases the consistency of information within the model. The fixed values for the terrace house model can be seen in appendix G.12.

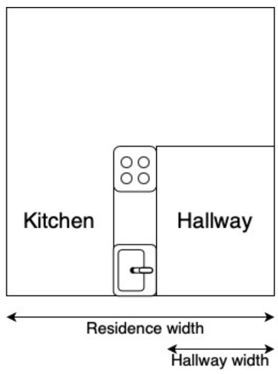


Figure 29 – Constraint calculation example

After the modelling of the structures of the system and formulating measurable requirements, a link has to be made between the two. Therefore, a satisfaction matrix is used. The satisfaction matrix connects all the requirements to the parameters of the system, subsystems, sub-subsystems, and components. In the matrix the parameters are connected to the relevant requirements. Each requirement has to be linked to minimal one parameter. The connection makes it possible to quickly check if the system requirements are met by the proposed design and parameters of the system. In appendix G.9, the satisfaction matrix can be seen and the links between the requirements and subsystems are shown in a relation map diagrams.

5.6.3 Requirement gaps

Filling in the satisfaction matrix showed that not each subsystem and/or component is connected to a requirement. It has to be noted that it is not necessary to connect each subsystem and/or parameter to a requirement. However, after discussing the results with Berri de Jonge, it was concluded that multiple subsystems have to meet requirements that were not part of the customer wishes formulated by the PMC departement.

The customer wishes contains 68 requirements that are included in the model. However, the list does not specifically state which subsystems, sub-subsystems, rooms and/or functions have to be included in the residence. Each of the individual subsystems are implicitly mentioned, however the sub-subsystems are not. As a consequence it is not stated in the customer wishes that the hallway has to contain a toilet space and a fuse box. Furthermore, it is not expressed that the bathroom has to contain a toilet, shower and sink. On the other hand, it is stated that the bathroom should optionally include a bath and a second sink, suggesting that the shower and sink are part of the design. However, not explicitly stating what has to be included in a design, could cause problems in the designing process. Berri de Jonge mentioned that often the fuse box still has to be added and fitted in, when the design is already 'finalised'.

Consequently, the system requirement list in CSM is expanded. The 'Bouwbesluit', 'woonkeur' and 'woonstandaard' documents are used as input for the formulation of additional requirements. The documents contain extensive information about the requirements a residence and/or building has to meet according to governmental rules and/or regulations. For this reason, they are used as reference books for requirements related to this specific residence of Plegt-Vos. First, the reference books are used to find requirements for the subsystems and parameters that have no connection to the requirements from the customer wishes. Thereafter, the reference books are used to add on, update, and check the existing requirements for the remaining subsystems and parameters. The system requirements list for the system model in CSM is expanded with 23 additional requirements, which makes a total of 91 requirements.

Three gaps have been identified in the customer wishes. The first gap is the lack of identifying the required sub-subsystems and/or components for the subsystems. As mentioned before, the hallway on the ground floor has to contain a toilet space, fuse box and stair case to the first floor. It was not explicitly stated in the stakeholder needs that these have to be included. Secondly, the requirements regarding the dimensions are not complete. The left out sub-subsystems and components did not have requirements for the dimensions. Analysing the reference books showed that these sub-subsystems and components have to meet certain regulations and standards regarding the minimal dimensions. Finally, the 'WoonStandaard' formulated dimensional mats that have to fit in the subsystems for specific living room spaces. For example, the living room should be able to contain a table setting mat of 2500x2500mm. The table setting mat and the entrance mat are included in the customer wishes. However, the 'WoonStandaard' contains six more dimensional mats that need to fit in the residence, namely the master bedroom, bedroom, seating area (living room), kitchen, bathroom, toilet and washing machine mat have not been included.

Updating the model in Cameo Systems Modeler

The formulation of new stakeholder needs, means that the model in CSM has to be updated. First, the new formulated requirements are added to the problem domain - black box in the stakeholder needs requirement table. Secondly, new system requirements (in the solution domain) are then derived from the additional stakeholder needs. Accordingly, subsystem, sub-subsystems and component requirements are formulated. Finally, the satisfaction matrix is updated. The new requirements are connected to the corresponding parameters and it is checked that all requirements are connected to minimum one parameter.

Checking the model

The system should be fully modelled in CSM. The full model is checked to reassure complete and correct information of the system. The checking process of the model starts with analysis of the derivation and satisfaction matrix. In the derivation matrix each stakeholder need have to be connected to minimal one system requirement. In the satisfaction matrix each system requirements has to be connected with minimal one (sub)system parameter. If this was not the case a connection is added.

Additionally, the system structure and interfaces in CSM are compared to the drawing of the terrace house of Plegt-Vos. The extra check is done to identify if subsystems, sub-subsystems, components and/or interfaces are excluded which should be included. The analysis showed that multiple interfaces are not included in the model, which would result in inconsistencies. The wall interfaces and/or ceiling interfaces between multiple subsystems are not included in the model, while they are included for other interfaces between subsystems. For consistency, the missing interfaces are added to the system model in CSM. The additional interfaces are documented in table 9, can be seen in appendix G.10 and an updated internal block diagram can be seen in figure 51 (Appendix G.6).

Next, a simulation is run, which checks the requirement values against the parameter values. The simulation showed that not all requirements are connected to the right parameter. Additionally,

Subsystems	Connects to	Interface
Kitchen	Bedroom 2	iKeuken-Slaapkamer2
	Bathroom	iKeuken-Bathroom
Living room	Master bedroom	iWoonkamer-Hoofdslaapkamer
	Bedroom 2	iWoonkamer-Slaapkamer2
	Bedroom 3	iWoonkamer-Slaapkamer3
Master bedroom	Bedroom 2	iHoofdslaapkamer-Slaapkamer2
	Bedroom 3	iHoofdslaapkamer-Slaapkamer3
	Attic	iZolder-Hoofdslaapkamer
Bedroom 2	Attic	iZolder-Slaapkamer2
	Bathroom	iSlaapkamer2-Badkamer
Bedroom 3	Attic	iZolder-Slaapkamer3
Bathroom	Attic	iZolder-Badkamer

Table 9 – Additionally identified interfaces

it showed that a few constraint equations contained errors, which resulted in wrong or no parameter values. Furthermore, it showed that some parameter values are independent, while they should be dependent on other values. All the identified errors are solved and an additional check is executed to reassure the errors are changed correctly.

5.7 Analysis of applying the framework in the AEC industry

The framework has been applied at Plegt-Vos to model the hallway core module and the system modules in a MBSE and variability model. The modelling process showed multiple points for improvement. Furthermore, some suggestions are done regarding the selection and decisions that have to be made.

In this research, the decisions for the PLE and MBSE branch have been made independently from the case context. The decisions are largely influenced by the reusability and variability of the (modelling) methods and the tools, since the main objective of the research is to support variability management of product lines with the support of MBSE. However, different (modelling) methods and tools for PLE and MBSE could be selected that would result in a MSBE and variability model. It is recommended to make the decisions not only based on the reusability and variability of the (modelling) methods and the tools, but also include the experience of the employees and the availability of the software within an organisation.

The platforms, modules, system architecture and variability were already defined by Plegt-Vos and that information and structure is used as input for the modelling process of MBSE and the variability model. In the PLE branch, variation points and OVM have been selected, which showed to be a suitable choice to model variability. Due to the variation points, the variability model focuses on specific points in the system model where variance exist in the product line. For this reason, the amount of information is limited to the differences in the product line and excludes information about the commonalities (which is included in for example feature models). The limitation of information supports reduction of the complexity of the variability model. Especially, when the size of the 150% model would increase drastically. Overall, variation points and OVM seem to be a suitable choice to model variability.

However, a software tool was needed to model the variability in a digital environment, such that it could be linked with the MBSE model. The selection of the software tool is not included in the implementation framework, but plays a crucial role in the final MBSE and variability model. For this reason, the tool selection step is added to the implementation framework. Additionally, the software tool for PLE should be compatible with the software tool of MBSE to model and select the different variants of the product line within the system model. Not every PLE software tool is compatible with every MBSE software tool and thus the MBSE software tool influences the decision for the MBSE software tool. Therefore, the tool selection in the MBSE branch is linked to the tool selection in the PLE branch. The updated framework can be seen in figure 30.

For MBSE, CSM and SysML have been used to model the 150% system model of an existing product line of Plegt-Vos. Both CSM and SysML provided the necessary functions, diagrams, analyses and notations to model the product line. On the other hand, OOSEM does not include a step-by-step process to model a system or product line. Therefore, the MagicGrid approach has been implemented in the case study to guide the modelling process step-by-step. The phases in the MagicGrid overlap with the steps of OOSEM. The execution of the case study showed that the MagicGrid approach smoothly guided the modelling process of the existing system within the MBSE tool (Cameo Systems Modeler). It was a step-by-step approach with clear traceability and relationships between the different abstraction levels of the model (e.g. problem domain black box, problem domain white box, solution domain system level, solution domain subsystem level, etc.). For this reason, iterations over the multiple levels of abstractions was uncomplicated.

On the other hand, understanding of the modelling language (SysML) and the modelling tool (CSM) progresses slowly. Multiple tutorials and literature needs to be studied to understand the language and tools. Besides, hands on experience is required to better understand and learn the language and tool. Furthermore, the usage of the MBSE and variability model showed the need of a clear language and notation document. During the modelling process specific name notations have been used to specify the difference between (sub)systems, parameter values and interfaces of the system. However, this naming process has not been documented explicitly. The document would have supported communication about the system with other stakeholders. For this reason, a training step and language documentation step are added to the implementation framework. The updated framework can be seen in figure 30.

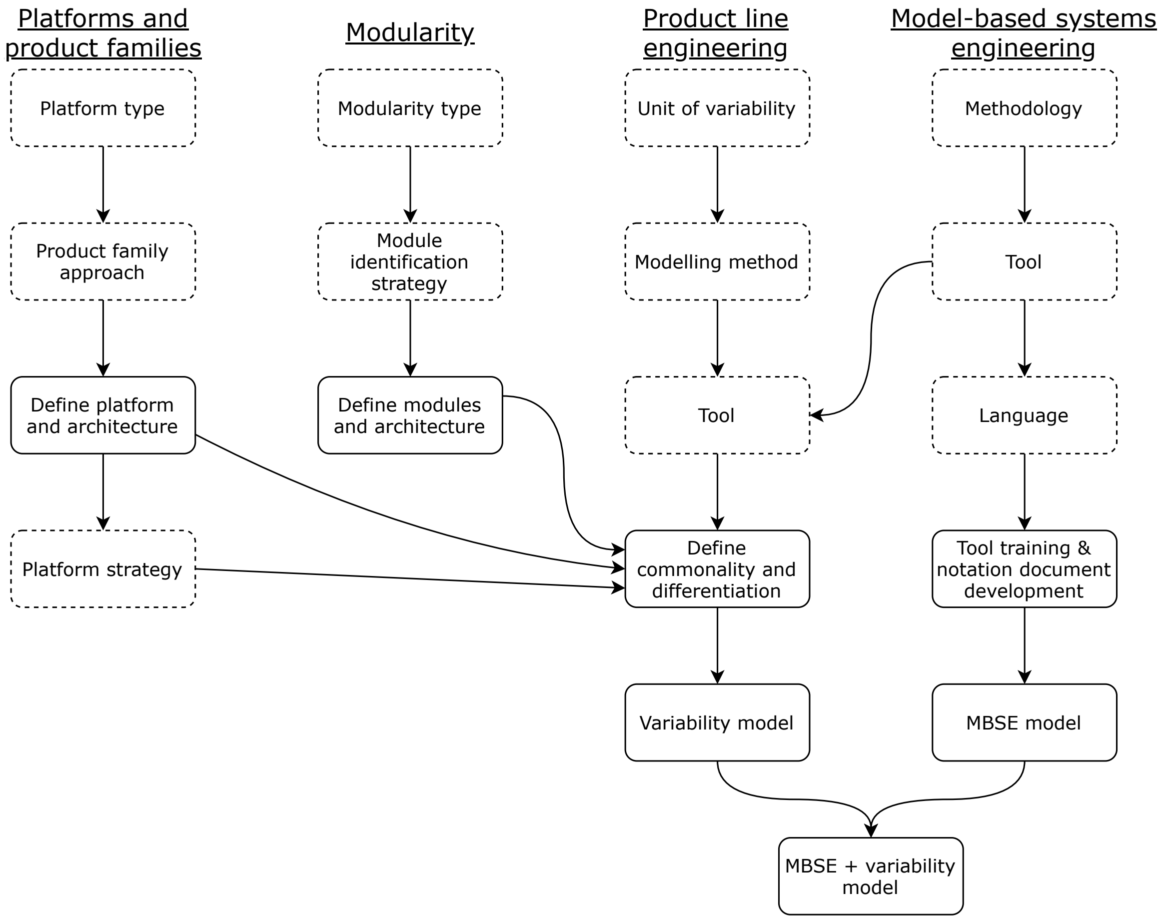


Figure 30 – Updated implementation framework

6 Testing the use of the system engineering model

In the first part of the case study the system has been modelled within Cameo Systems Modeller and Pure::Variants. Next, the effects of implementing MBSE will be tested and quantified by means of an experiment. The effects will be studied by comparing the usage of MBSE to the current approach at Plegt-Vos. In this chapter, the development approach of Plegt-Vos is analysed and compared to MBSE to determine the dependent variables for the experiment. Additionally, the experimental design is documented.

6.1 Plegt-Vos development approach versus MBSE

Plegt-Vos makes use of Building Information Modelling (BIM) for the designing and developing of systems and system variants. The BIM is used to store information of multiple different disciplines, for example piping layout and structural layout. The final model includes a 3D imagine of the designed and configured system design for a single system. BIM is used for the design, development and configuration of a single variant of the system (see figure 13).

As mentioned in section 3.8, BIM and MBSE are relatively similar. However, a few differences have been identified. First, MBSE stores information related to the requirements, behaviour, structure and parameters of the system, while BIM stores information related to behaviour, structure and parameters. Since, the BIM approach does not include requirement documentation, separate requirement documents have to be kept, tracked and updated. Furthermore, the separate documents have to be used to manually check the design parameters against the requirements.

Furthermore, MBSE can be used to model a whole product line including all the variant models, while BIM is limited to store information about one variant of the product line. Information can be reused and adapted based on other variant models to independently model a new variant. For this reason, BIM is more prone to inconsistencies in information between variant models.

Additionally, the information stored within MBSE is all linked to one another, which enables traceability between related information sources (e.g. requirements and parameters, parameters from different subsystems, etc.). On the other hand, BIM does not include information about the requirements, thus the parameter values can not be linked and automatically checked in the model. Requirement checking and verification has to be done manually. The information about the structure, behaviour and parameters are imported and integrated in BIM from different software tools. This information is linked and shows where collisions and/or errors occur in the configuration of the whole system design.

Consequently, the experiment will focus on showing the impact of implementation of MBSE compared to BIM on the basis of these difference in the approach. The experiment is designed to measure the difference in requirement traceability and information storage. Additionally, the system traceability is included to reassure the system traceability is equal for the two different approaches.

6.2 Experimental design

To compare the system traceability, requirement traceability and information storage of MBSE and BIM, an experiment is designed that includes 11 scenarios that have to be executed by the participants. The execution of the scenarios allowed for measurement of the traceability and information storage. A between-subject design has been applied in the experiment, which means that each participant is assigned to a single experimental treatment group (e.i. the MBSE beginner group, the MBSE expert group, and the BIM expert group). The two expert groups are used to compare the results between MBSE and BIM. The two different MBSE experience levels are used to control for experience, since the effectively use of MSBE has a steep learning curve. Thus, to see if the same effects can be found for inexperienced users. An overview of the experimental procedure over time is shown in figure 31.

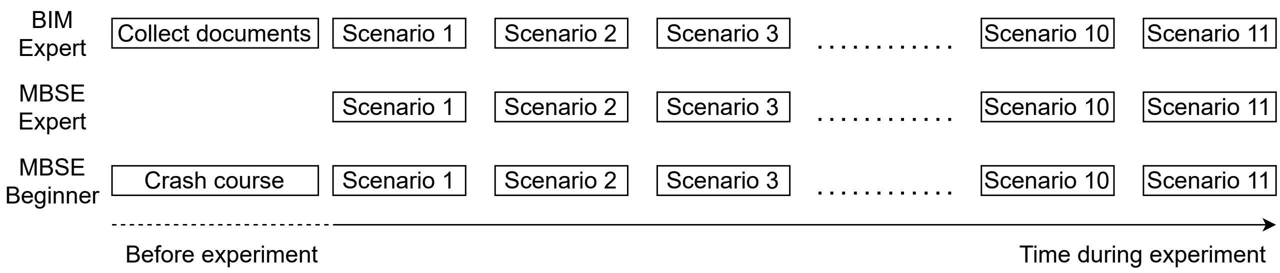


Figure 31 – Timeline of the experiment

6.2.1 (In)dependent variables and measurements

The effects of implementing MBSE instead of BIM on the system traceability, requirement traceability and the information storage will be studied. Consequently, system traceability, requirement traceability and information storage have to be operationalised such that the effects can be measured.

Traceability is a popular topic within software and systems engineering. However, within these research areas traceability does not have a general accepted definition. In this research, the definition of Gotel et al. (2012) is used, who defined traceability as below. In this definition 'traces' is a noun, which refers to original artefact, the target artefact and the link itself. The concept of traceability is visualised in figure 32

> *"Traceability – The potential for traces to be established and used. Traceability (i.e., trace "ability") is thereby an attribute of an artifact or of a collection of artifacts. Where there is traceability, tracing can be undertaken and the specified artifacts should be traceable." (p.9.)*

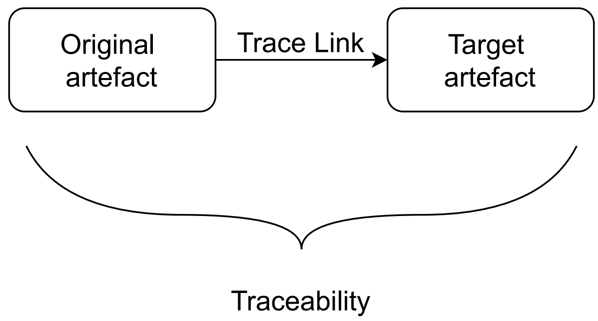


Figure 32 – Traceability definition visualised

The information used in the design and development of a new system (either within BIM or MBSE) should have links (e.i. traces) between original artefacts and target artefacts. For example, component requirements have traces to the stakeholder, requirements have traces to (sub)systems and/or components, parameters have traces to other parameters, etc. Traceability of information within a system will be measured based on the completeness, correctness and time. Firstly, completeness is measured as the number of identified linked artefacts as percentage of the total linked artefacts. Secondly, the correctness is measured as the number of correctly identified linked artefacts as percentage of the total linked artefacts. Finally, the time is measured in minutes and seconds. An overview can be seen in figure 33

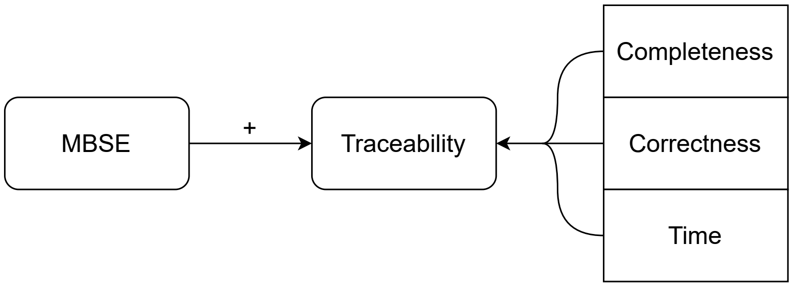


Figure 33 – Overview of the effect of implementation of MBSE on traceability

Secondly, the information storage of MBSE and BIM will be analysed. It is assumed that the information about the structure and the parameters of the system are equal between the BIM approach and the MBSE approach, since the information of the BIM approach is used as input for the modelling in CSM. However, Plegt-Vos did not offer a single requirement document as input, but referred to multiple different sources (e.g. WoonKeur, woonstandaard, etc.). Therefore, the information storage analysis will be based on the requirement documents of both approaches. Information storage is measured on the basis of completeness, consistency and extendability, similar to Maurandy et al. (2012). The completeness will be measured on the basis of a ratio between the number of relevant requirements included. The consistencies will be measured as the amount of inconsistent information (e.g. parameter values, requirement values, etc.) within the requirement documents. Finally, extendability is measured on a scale from 1 to 7 (1 = extremely difficult - 7 = extremely easy). An overview can be seen in 34.

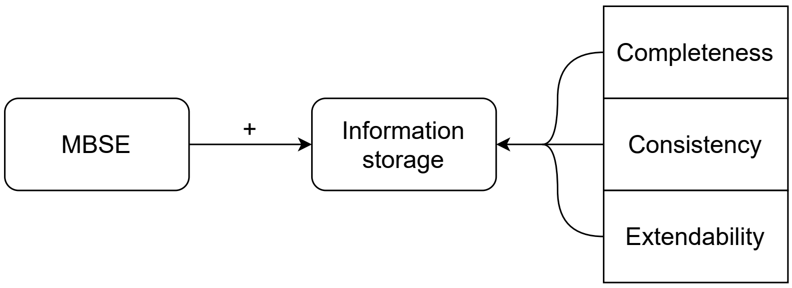


Figure 34 – Overview of the effect of implementation of MBSE on information storage

6.2.2 Experimental subjects

To execute the experiment, subjects have to be selected. Subjects are needed for both the Plegt-Vos approach (BIM) and the MBSE approach. To participate in the experiment the subjects either has to have knowledge about the current development process of Plegt-Vos and/or knowledge of usage of Came Systems Modeler. The participants minimally should be able to identify traces between subsystems, components, parameters, interfaces and requirements.

The subjects have been selected from within the organisation. The first subject is an employee of Plegt-Vos, which is the BIM expert in the organisation. He works with BIM on a day to day basis and is an expert in designing, developing, modelling and configuring new systems for the company. He has 10 years of experience working with BIM. He has been asked to collect all requirement, design and modelling documents and files that he has used for the development of the reference terrace house to bring to the experiment. Since, the treatment for the experiment is the implementation of MBSE within the construction industry, he will be in the control group of the experiment.

On the other hand, there are no employees who have experience with MBSE or CSM within Plegt-Vos. The process manager innovation, volunteered to be part of the experiment as the MBSE participant subject. He will be using the CSM model of the Plegt-Vos residence during the experiment. Since, he has no experience with MBSE and/or CSM he has gotten a crash course in how to use the model to change requirements, check requirements, find relations (traces) between requirements and (sub)systems, components and/or interfaces, and were to find all the relevant information in the model. With the knowledge he should be able to participate in the experiment without facing challenges or having disadvantages. The system model in CSM is provided to him at the beginning of the experiment, so he will not be able to study the system model beforehand. Since, implementation of MBSE will be the treatment of the experiment, he will be in the treatment group (beginner level).

Finally, the researcher will participate in the experiment. Since, the BIM expert and the MBSE beginner have different expertise and knowledge levels regarding BIM/MBSE and the terrace house residence, it will be difficult to link the difference in results to the effect of the usage of BIM versus MBSE or to the difference in experience and knowledge. Therefore, the researcher will participate as the MBSE expert. Furthermore, the researcher also has sufficient knowledge about the terrace house model and requirements. The researcher will be seen as the a second treatment group (expert level).

6.2.3 Experimental procedure

To measure the effects of implementing MBSE compared to BIM, multiple scenarios have been formulated that could occur during the development process of a system at Plegt-Vos. The eleven scenarios are related to requirement changes, design changes or configuration of a new system with available modules. The experiment participants have to identify the trace links between the changed artefacts and the impacted artefacts. The changed artefact can either be the original or the target artefact.

The experiment starts with a small introduction of what is expected, how to document the their findings and a description of scenario 1. The scenarios are related to changes in a certain artefact, which has trace links to other artefacts. The participants are expected to identify either the original artefacts or the target artefact that the changed artefact has a trace link with. For each scenario it is specified what type of artefacts have to be included in the analysis (e.g. subsystems, requirements, parmaters, etc.) The identified artefacts have to be documented on a paper sheet that has been provided by the researchers (see appendix A for an example). The experiment focuses on the identification of the artefacts and it is not necessary to elaborate how the artefact is impacted. The subjects have a maximum amount of 30 minutes for each scenario.

Scenario 1: Design change.
At the moment, the hallway ground floor subsystem within the terrace house does not meet the 'WoonKeur'[2] requiremements. The 'WoonKeur' requires a free space of minimal 1500x1500mm or 1350x1850mm behind the entrance door. Therefore, the hallway width will have to be adapted, such that the final system will meet the 'WoonKeur' requirement. The width of the hallway is changed from 2445mm to 2545mm (see figure 35).

Figure 35 – Visualisation scenario 1

Scenario 2: Design change.
At the moment, the hallway first floor subsystem within the terrace house does not meet the minimum traffic space requirements of 1100mm. To meet the requirement, the width of the hallway first floor subsystem is changed. The new width is 2440mm instead of 2410mm. Changing the width of the subsystem influences other surrounding subsystems, components, and/or interfaces (see figure 36).

[2]WoonKeur (2015). *Woonkeur Bestaande woningen. Certificaat D. 'rolstoelgeschikt'.* Retrieved from https://repository.officiele-overheidspublicaties.nl/externebijlagen/exb-2016-37683/1/bijlage/exb-2016-37683.pdf

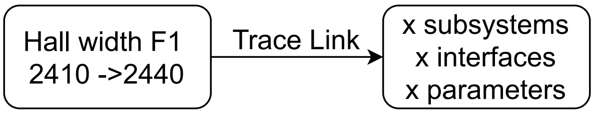


Figure 36 – Visualisation scenario 2

Scenario 3: Design change.
Regularly, it is requested to integrate the hallway core in a residence with adapted depth and width compared to the terrace house. Changing those parameters will have an impact on the floor plan and the dimensions of the remaining modules (e.i. living room, kitchen, bedrooms and bathroom). One of the first thing that need to be reassured, is that the modules will meet the requirements set for the modules dimensions (in the form of mats from WoonStandaard)[3]. For this scenario the dimensions for the residence have to be updated, and each of the module dimensions have to be checked against the living standard requirements (see figure 37).

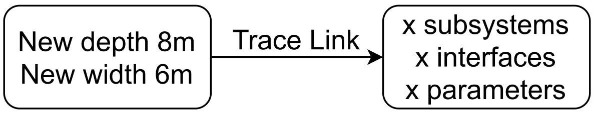


Figure 37 – Visualisation scenario 3

Scenario 4: Requirement change.
A customer wishes to have a larger kitchen then the current requirement of 10 m2 in the design of the system. The customer asks for a 12m2 kitchen. Changing the customer wish has an influence on the requirements of the system. Therefore, it is necessary to adapt the system requirements such that it corresponds to the customer wishes/stakeholder needs. Additionally, it is needed to check if the system meet the requirement or if redesign is required. If the system has to be redesigned, it is needed to identify which subsystems, components and/or interfaces are affected and have to be adapted (see figure 38).

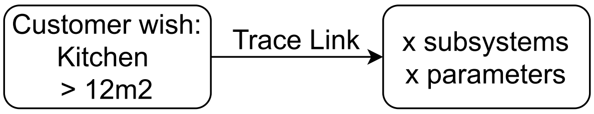


Figure 38 – Visualisation scenario 4

Scenario 5: Requirement change.
A customer wishes to have more space in the three bedroom on the first floor. Therefore, the customer wishes to have three bedrooms with a minimum area of 8m2 instead of the current stakeholder need of 7m2. For this scenario the stakeholder needs and/or system requirements have to be updated. Furthermore, it is needed to check if the system meets the changed requirements. If not, the subsystems, components and/or interfaces have to be identified that need to be adapted and/or redesigned (see figure 39).

[3]Netwerk conceptueel bouwen (2020). De Woonstandaard. basiseisen per product markt combinatie. Retrieved from https://www.conceptueelbouwen.nl/dewoonstandaard

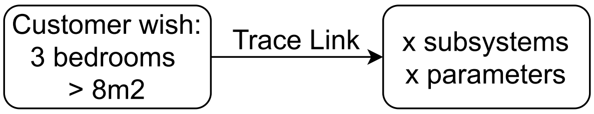


Figure 39 – Visualisation scenario 5

Scenario 6: Configuration of a system.
Plegt-Vos enabled customisation of the house by adding on modules or changing the appearance of certain modules. In this scenario, a customer has configured a system based on the optional choices offered by Plegt-Vos. The customer decided on including solar panels in the design of their residence. Next, it is up to the engineers to check the compatibility, configure the system, and verify the requirements. (see figure 40).

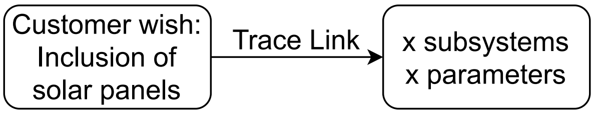


Figure 40 – Visualisation scenario 6

Scenario 7: Checking requirements.
The traffic space on the ground floor has been changed in the design from 4144mm to 4300mm. By changing the parameter, the requirements of the system have to be check to reassure that the system is still meeting all of them. Therefore, the engineers have to check which requirements are not met by changing this parameter (see figure 41).

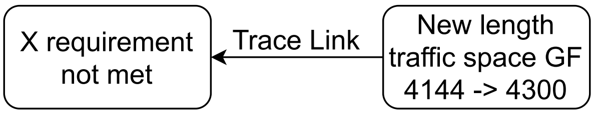


Figure 41 – Visualisation scenario 7

Scenario 8: Checking requirements.
The hall width on the first floor is adapted from 2410mm to 2750, to reassure that the traffic space is wide enough to meet the WoonStandaard requirement. By changing the parameter, the remaining requirements of the system have to be check to reassure that the system still meets all of them. Therefore, the engineers have to check which requirements are not met by changing this parameter (see figure 42).

Figure 42 – Visualisation scenario 8

Scenario 9: Checking requirements.
The hall width on the ground floor is adapted from 2445mm to 2800, such that the entrance mat from the WoonStandaard fits into the traffic space. By changing the parameter, the remaining requirements of the system have to be checked to reassure that the system still meets all requirements. Therefore, the engineers have to identify if and which requirements are not met by changing this requirement (see figure 43).

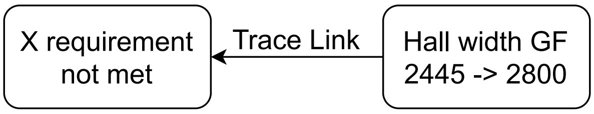


Figure 43 – Visualisation scenario 9

Scenario 10: Checking requirements.
The depth of the residence is adapted from 8750mm to 8000mm, because the customer wishes to have a larger garden. By changing the parameter, the remaining requirements of the system have to be checked to reassure that the system still meets all requirements. Therefore, the engineers have to identify if and which requirements are not met by changing the depth of the residence (see figure 44).

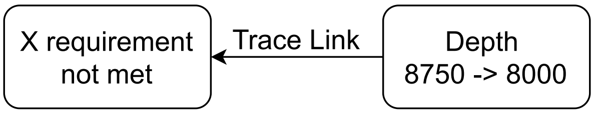


Figure 44 – Visualisation scenario 10

Scenario 11: Checking requirements.
The length of the toilet space is shortened from 1200mm to 1000mm to save some space within the hall module on the ground floor. This results in a larger traffic space and thus more room for the wardrobe. By changing the parameter, the remaining requirements of the system have to be checked to reassure that the system still meets all requirements. Therefore, the engineers have to identify if and which requirements are not met by changing this parameter (see figure 45).

Figure 45 – Visualisation scenario 11

7 Results of the experiment

In the previous chapter the process of modelling the terrace house of Plegt-Vos has been documented. Additionally, the experiment has been outlined, which studies the effect of implementation MBSE compared with BIM on the basis of information storage, system traceability and requirement traceability. In this chapter the results of the experiment are documented.

7.1 Data measurements of the experiment

Three different groups participated in the experiment, namely the MBSE beginner group, the MBSE expert group and the BIM expert group. The results of the eleven scenarios of the three different groups are analysed and compared on the basis of the completeness of the answers (as percentage of total), the correctness of the answers (as percentage of the total) and the time. The first six scenarios are related to system traceability, which focuses on the traceability of (sub)systems information to related (sub)systems information. Scenarios seven to eleven are related to requirement traceability, which focuses on the traceability from (sub)systems information to the related requirements. For all scenarios the time needed to finish the scenario is measured. The measured results can be found in appendix C.

Additionally, the information storage has been analysed during the experiment. The information storage is operationalised by measurement of the completeness of the requirements, consistency of the requirements and the extendability of the requirements. However, during the experiment it became clear that the documents used by the BIM expert are the 'Bouwbesluit', 'WoonStandaard', 'Woningborg', and the 'PMC customer wishes' documents, which also have been used as input document for the MBSE system model. Consequently, the completeness and the consistency is equal for both the BIM requirement documents and the MBSE model. Furthermore, the extendability for both BIM and MBSE requirement documents are perceived as extremely easy. Therefore, no difference exist in information storage between MBSE and BIM.

7.2 Analysis of system and requirement traceability

Three different groups did participate in the experiment, MBSE beginner group, MBSE expert group, and BIM expert group, each executing the eleven scenarios. This resulted in data about the completeness of identified artefacts, correctness of identified artefacts and time required for each scenario. From the data of the three different experiment groups it is difficult to draw conclusions about the effects of the usage of MBSE versus BIM, therefore an analysis of variance (ANOVA) has been executed to analyse the effects. ANOVA is used to analyse difference in means between the three groups, and thus to study if the usage of BIM versus MBSE has a significant effect on system and requirement traceability (Tomarken & Serlin, 1986).

A difference in the means for the completeness and correctness between the experiment groups implies that either the approach, MBSE vs BIM, or the expertise level, beginner vs expert, has an effect on the effectiveness of system and/or requirement traceability. Besides, a difference in means between the experiment groups for the time implies that either the approach or the expertise level has an effect on the efficiency of system and/or requirement traceability. Therefore, the results of the experiment show the effects of implementation of BIM vs MBSE on the system and requirement traceability.

The experimental data is split up into two separate groups, namely the traceability between subsystem information (scenario 1 to 6) and the traceability between subsystem information and requirements (scenario 7 to 11). Scenario 1 to 6 are related to system traceability, while scenario 7 to 11 are linked to requirement traceability. Futhermore, scenario is 5 is left out of the data analysis. Scenario 5 is formulated to examine the difference in time that is needed to realise that the customer wishes already meet the existing requirements. The 100% complete and correct system traceability influences the data significantly.

7.2.1 Testing assumptions

The ANOVA test is based on 3 assumptions. First, the dependent variables are continuous, while the independent variables are categorical. Second, the data is normally distributed. Finally, the variance of the data sets are equal. The first assumption is met with the variables used in the experiment. The other two assumptions have to be checked by testing the data in SPSS. To test for normal distribution a Shapiro-Wilk test has been used. The results of the Shaprio-Wilk test can be seen in Appendix E. The results of the test showed that the data in each data set is normally distributed.

Next, the equal variance has to be tested. A Levene's test has been used to check equal variance. The results of the Levene's test are shown in table 10. The results show that p>0,05 for the correctness of the system traceability, time of the system traceability, completeness of the requirements traceability, correctness of the requirements traceability and the time of requirements traceability, which means that the null hypothesis can not be rejected. Therefore, the variance of the three groups (e.i. BIM expert, MBSE expert, MBSE beginner) are equal for these dependent variables. On the other hand, for the system traceability completeness p<0,05 and therefore the null hypothesis can not be rejected, which means that no statistical evidence exist for equal variance between the three groups.

Levene's Test - Based on Mean	Levene Statistic	df1	df2	Sig.
Completeness System Traceability	3.993	2	12	0.047
Correctness System Traceability	1.984	2	12	0.180
Time System Traceability	2.030	2	12	0.174
Completeness Requirements Traceability	0.077	2	12	0.926
Correctness Requirements Traceability	0.015	2	12	0.986
Time Requirements Traceability	0.843	2	12	0.454

Table 10 – Results of the Levene's test for the different data sets

7.2.2 Analysis of Variance (ANOVA)

For the dependent variables that have a normal distribution and homogeneity in variance, an ANOVA analysis is conducted. The results of the ANOVA test are shown in table 11. The ANOVA results show that the means are significantly different for the system traceability correctness (F(2,12) = 8,209, p = 0,006), requirement traceability completeness (F(2,12) = 47,166, p = 0,000), requirement traceability correctness (F(2,12) = 42,407, p = 0,000), and requirement traceability time (F(2,12) = 4,928, p = 0,027). For the system traceability time (F(2,12) = 0,5178, p = 0,609) the null hypothesis can not be rejected, which means that it does not show a statistical difference between the means of the three groups.

		Sum of Squares	df	Mean Square	F	Sig.
Correctness System Traceability	Between Groups	6235200	2	3117600	8,209	,006
	Within Groups	4557200	12	379767		
	Total	10792400	14			
Time System Traceabiltiy	Between Groups	101237733	2	50618867	,517	,609
	Within Groups	1175770000	12	97980833		
	Total	1277007733	14			
Completeness Requirements Traceabiltiy	Between Groups	14923,333	2	7461,667	47,166	0,000
	Within Groups	1898,400	12	158,200		
	Total	16821,733	14			
Correctness Requirements Traceability	Between Groups	14132,800	2	7066,400	42,407	0,000
	Within Groups	1999,600	12	166,633		
	Total	16132,400	14			
Time Requirements Traceability	Between Groups	136088,133	2	68044,067	4,928	0,027
	Within Groups	165685,600	12	13807,133		
	Total	301773,733	14			

Table 11 – Results ANOVA

7.2.3 Welch test

Since the equal variance was violated for the system traceability completeness, an ANOVA test could not be conducted. Instead a Welch test has been selected and used to test for difference in the means of the three groups for the system traceability completeness (Tomarken & Serlin, 1986). The results of the Welch test are documented in table 12. The significance level of the Welch test (F(2, 6,565) = 8,296, p = 0,016) is below p = 0,05, thus the null hypothesis can be rejected. As a result, the Welch analysis shows a statistical difference between the means of the three groups for the completeness of the system traceability.

	Statistica	df1	df2	Sig.
Welch	8,296	2	6,565	0,016

Table 12 – Results of the Welch test

7.2.4 Post hoc testing

For the dependent variables that have a significant difference in the means or variance between the three groups (e.i. BIM expert, MBSE expert and MBSE beginner), it is interesting to study which group means or group variance differ. A post hoc ANOVA test is used to show the difference in means between the groups, namely the Tukey HSD. Furthermore, a pairwise levene's test is conducted to show the differrence in variance between the groups. The results of the post hoc analysis is documented in appendix F. As stated above a difference in mean exists for completeness and correctness for the system traceability and the completeness, correctness and time for the requirement tracaeability. For each of these dependent variables the a Tukey HSD is conducted. Additionally, a difference in variance exist for the completeness of the system traceability.

First, the Tukey HSD test shows significant difference in means between the MBSE beginner level and the two remaining categories for the completeness of the system traceability. The completeness is significantly lower for the MBSE beginner group (33,00±26,163) compared to the MBSE expert group (83,60±7,829, p = 0,003) and the BIM expert group (70,60±17,184, p = 0,020). On the other hand, no significant difference exist between the means of the MBSE expert group and the BIM expert group.

Additionally, the pairwise Levene's test shows significant difference in variance between the MBSE expert group and the other two groups. The variance of the system completeness is significantly lower (F(1,10) = 6,096, p = 0,033) for the MBSE expert group (±7,829) compared to the BIM expert group

(±17,184). Furthermore, the variance of the system completeness is significantly different (F(1,10) = 7,526, p = 0,021) for the MBSE expert group (±7,829) compared to the MBSE beginner group (±26,163). No significant difference in mean exists for the BIM expert group and the MBSE beginner group.

Second, the Tukey HSD test shows similar results for the difference in means between the three groups for the dependent variable correctness of the system traceability. The correctness is significantly lower for the MBSE beginner group (33,00±26,163, p = 0,013) compared to the MBSE expert group (77,40±18,474, p = 0,009) and the BIM expert group (75,00±10,654). However, no significant difference in mean exist between the MBSE expert and BIM expert group.

Furthermore, the results show a significant lower completeness of requirement traceability for the BIM expert (12,20±13,572) compared to the MBSE beginner (56,20±12,112, p = 0,000) and MBSE expert (89,20±11,987, p = 0,000). Additionally, the results show a significant lower correctness for the MBSE beginner (56,20±12,112) compared to the MBSE expert (87,00±11,987, p=0,004).

Additionally, the Tukey HSD results show a significant lower correctness of requirement traceability for the BIM expert (12,20±13,572) compared to the MBSE beginner (56,20±12,112, p = 0,000) and MBSE expert (8,00±13,00, p = 0,000). Additionally, the results show a significant lower correctness for the MBSE beginner (56,20±12,112) compared to the MBSE expert (87,00±13,00, p = 0,007).

Finally, the Tukey HSD results show a significant lower time of requirement traceability for the MBSE expert group (3:05±0:54 minutes) compared to BIM expert group (6:55±2:38 minutes, p = 0,023). There is no significant between the MBSE expert group (3:05±0:54 minutes) and MBSE beginner group (4:27±1:54 minutes, p = 0,529) or the MBSE beginner group (4:27±1:54 minutes) and the BIM expert group (6:55±2:38 minutes, p = 0,157).

8 Discussion and conclusion

In the next three decades, the Dutch government aims at achieving an energy neutral build environment. To achieve their goals existing buildings have to be updated and renovated to reduce energy usage and new construction projects have to be energy neutral. Hence, the construction industry is facing many challenges, since each construction is different, different technologies exist for energy reduction, and the projects have to be finished in a short time period. To overcome these challenges, construction companies can focus on implementation of modular building and combine this with model-based systems engineering (MBSE) and product line engineering (PLE). Therefore, this research focus on the implementation of MBSE to support variety management in the product lines of the Architecture, Engineering and Construction (AEC) industry.

In chapter 4, a framework has been developed to implement MBSE and PLE within an AEC company. The framework has been applied to model an existing system of a construction company, Plegt-Vos, with MBSE and PLE. The implementation of the framework in a practical context shows the effectiveness and validates the use. Additionally, the final MBSE and variability model is tested by means of conducting an experiment to compare the adoption of MBSE to BIM. In this chapter the theoretical implications, the findings of the experiment, and the managerial implications are discussed.

8.1 Theoretical implications

In this research, the main objective was to formulate, validate and verify an approach to combine variety management strategies, PLE and MBSE to support variability management of product lines in the AEC industry. To my knowledge, no approach is formulated that combines variability management, MBSE and PLE. Furthermore, PLE and/or MBSE is not yet applied in the practical context of the AEC industry. Previous research has studied the use of MBSE and PLE in other industries (Bilic et al., 2018; Polit Casillas & Howe, 2013), however the research focuses on a context specific implementation of the methodologies. Other research has developed a framework for the implementation of MBSE in the construction industry, however it has not been applied in a practical context (Valdes et al., 2016). Consequently, this research fills the literature gap by formulating a methodology for the implementation of modularity, product platforms and product families, PLE and MBSE in a practical context in the AEC industry.

Additionally, many benefits have been formulated for the implementation of MBSE in academic literature (Friedenthal et al., 2015e; Madni & Sievers, 2018; Henderson & Salado, 2021). However, empirical evidence supporting the beneficial claims is lacking (Henderson & Salado, 2021). This research provides empirical evidence for the benefits of information storage, system traceability and requirement traceability of MBSE compared to BIM. The results shows that MBSE is significantly more effective and efficient for requirement tracing in comparison to BIM, which is commonly used within the AEC industry. Therefore, the empirical evidence in this research support the claims that the implementation MBSE has a positive influence on requirement traceability in the system development process.

8.2 Discussion of findings

The main contribution of this research is that it provides a framework that combines and can be used for implementing MBSE and PLE to support variety management strategies of product lines within the AEC industry. Additionally, it provides empirical evidence for the difference of MBSE and BIM regarding the system and requirement traceability.

Application of and reflection on the implementation of MBSE and PLE in the case study organisation led to updating the framework by including multiple steps. The added steps have been executed during the modelling of the system of the case study organisation, but have been made explicit in the updated framework. The updated framework can be used as guidance for implementing MBSE and PLE in an organisation and/or specific project. The developed framework enriches the literature about combining MBSE and PLE (Hummell & Hause, 2015; Bilic et al., 2018) and applying it in the AEC industry (Valdes et al., 2016).

The findings of the experiment show that MBSE is significantly more effective and efficient for requirement traceability, which is unaffected by the expertise level. Additionally, BIM and MBSE shows no difference in the completeness and correctness of the system traceability when the expertise level is equal. However, when the experience and knowledge is limited the effective use of the system model is low.

To be more precise, the system traceability shows significant difference in means for the completeness and the correctness between the three experiment groups, however not for the time. For the correctness and the completeness, the MBSE beginner group has a significant lower mean compared to the BIM expert group and the MBSE expert group. No significant difference in means exist between the BIM expert and MBSE expert group. The usage of BIM or MBSE are equally effective and efficient for the system traceability, when used by experts. However, it is less effective, when an employee has a limited experience and knowledge about the MBSE/PLE software and build up of the system. These results show that the learning curve of the software and modelling approach is quite steep. The short crash course provided to the MBSE beginner group is not sufficient for tracing the links between the (sub)subsystems and other (sub)systems), the (sub)systems) and parameters, and parameters and parameters. Furthermore, the build up of the system in the software and corresponding notations in the system model had not been document and thus the MBSE beginner had no reference book during the experiment. Consequently, it makes the understanding and usage of the system model more difficult, since the MBSE beginner group had to learn, understand and remember everything taught during the crash course.

Additionally, the MBSE expert group has a significantly higher completeness and correctness of the traced requirements compared to the MBSE beginner group and BIM expert group. Furthermore, the MBSE beginner group has a significantly higher correctness and completeness of the traced requirements compared to the BIM expert group. Finally, the time to trace the requirements is significantly shorter for the MBSE expert group compared to the BIM expert group. The results show better requirement traceability for both MBSE groups compared to the BIM expert group. The difference in means can be explained by the difference in documentation and storage of the requirements. MBSE incorporates the requirement documents and the links them to the corresponding (sub)systems and parameters. Furthermore, within CSM the system requirements can be checked against the parameter values with a simple analysis. On the other hand, BIM stores the requirements in separate documents and tracing, tracking and checking has to be done manually. Manual requirement traceability give a lot of opportunity to errors, especially when the complexity of the system and amount of requirement increase, while the simplicity of the analysis in CSM makes it possible for MBSE beginners to do a better job regarding requirement traceability then the BIM expert.

These findings complement and enrich the benefits of MBSE listed by Friedenthal et al. (2015e) with empirical evidence to strengthen the claims, as suggested by Henderson & Salado (2021). Furthermore, the empirical evidence complement and enrich the comparison of MSBE and BIM of (Valdes et al., 2016).

8.3 Managerial implications

MBSE and PLE have been combined and applied in a practical context (Hummell & Hause, 2015; Bilic et al., 2018). The combination of the two approaches enables modelling of variability of product lines in a single model. The product lines models can be used for configuration of system variants by selection of appropriate modules in the variability model. Furthermore, the model supports reuse of modules and information in the development of new system variants. New variants can be developed by adding new features and/or new modules and combining them with existing modules in the product line (Hummell & Hause, 2015; Bilic et al., 2018).

The developed framework and the findings of this experiment, have valuable meaning for managers in the AEC industry. The framework can be used for guidance in the implementation of variety management strategies, PLE and/or MBSE in an organisation or project in the AEC industry. Furthermore, the framework does not dictate what approaches and tools have to be used, but rather shows where to start, what to start with and the decisions that have to be made regarding the approaches. Consequently, it can be used by a variety of organisations or projects and in different contexts.

Switching from MBSE to BIM is definitely a big decision and a time consuming transition. However, Valdes et al. (2016) suggests to implement MBSE in the AEC industry. MBSE can be used to overcome multiple disadvantage of BIM, such as lack of verification methods of BIM models. Additionally, the experimental results of this research show that implementation of MBSE is indeed more suitable for requirement traceability, which means it is better for requirement verification of the final system. However, there should be caution in switching from BIM to MBSE. Even though, MBSE might have requirement traceability benefits, the initial investment costs are high and learning curve is steep, while BIM offers a large range of benefits (see table 1). A big difference between the two approaches is that MBSE is an abstract method, while BIM uses concrete 3D models of the system. For this reason, some research suggests to combine BIM with MBSE to improve requirement traceability, while also having the benefits of designing a 3D model (Polit Casillas & Howe, 2013; Keskin & Salman, 2020).

8.4 Limitations and directions for future research.

Each research has their limitations and this research is not an exception. However, limitations give rise to new research opportunities and directions. Firstly, the developed framework has been applied in a practical context. The PLE and MBSE branch have been applied and the methodologies have been implement in a practical context. However, Plegt-Vos already implements product platforms, modules and architectures for their products and product lines. Consequently, the "platforms and product families" and the "modularity" branch of the framework have not been applied in the practical context. Thus, it is unknown if the steps and decisions are sufficient to define product platforms, modules and the architecture. For this reason, it is recommended for future research to apply the full framework in a practical context in the AEC industry.

Secondly, at Plegt-Vos an existing system has been used to be modelled with MBSE and PLE. An existing system is selected due to time constraints of the research project and to initially show the use, effects and the benefits of MBSE in the case study organisation. The framework could be used to model an existing product line with an MBSE variability model, however it is suggested to implement the four methodologies at the start of the development of a new product lines and/or product. Therefore, it is recommended to research the use of the framework in the development of a new product (line).

Third, three people participated in the experiment. Each participant is part of a different experiment group, namely the MBSE beginner group, MBSE expert group and BIM expert group. The participant have been selected to test for difference in the information storage, system traceability and requirement traceability. The experiment results show that a significant differences. However, due to the limited amount of participant it is unsure if the results would be the same with a larger amount of participants. Therefore, it is recommended to redo the experiment with a larger number of participant.

Finally, in the experiment 11 scenarios have been executed and measured on the bases of time, correctness and completeness. The first six scenarios are part of the system traceability and the last five scenarios are used to test the requirement traceability. The results have been used as input for an ANOVA. The N for both the system traceability and the requirement traceability is 5. ANOVA can be used with a minimum of N=3, however the strength of the results goes up with a higher N. Consequently, the strength of the results for this experiment is limited. For this reason, it is recommended for future research to do an experiment with a higher number of scenarios for both the system traceability and the requirement traceabilty.

8.5 Conclusion

In this research, a literature gap has been addressed on the implementation of PLE and MBSE to support variability management of product lines in the AEC industry. A framework has been developed to apply the strategies in the construction industry. The framework has been used to model an existing design of a system from the case study organisation. The case study showed that the decision and execution steps of the framework are effective for the implementation of PLE and MBSE in the construction industry. Finally, the system model is used in an experiment comparing the system traceability, requirement traceability and information storage of MBSE and BIM. The results show that MBSE is more effective for requirement traceability, thus MBSE shows promising results to support changing requirements and difference in customer wishes during the system development process.

References

Aleksandraviciene, A., & Morkevicius, A. (2018). Magicgrid (r) book of knowledge. *Kaunas: Vitae Litera*.

Azhar, S. (2011). Building information modeling (bim): Trends, benefits, risks, and challenges for the aec industry. *Leadership and management in engineering*, *11*(3), 241–252.

Baldwin, C. Y., & Clark, K. B. (2003). Managing in an age of modularity. *Managing in the modular age: Architectures, networks, and organizations*, *149*, 84–93.

Berger, T., Rublack, R., Nair, D., Atlee, J. M., Becker, M., Czarnecki, K., & Wasowski, A. (2013). A survey of variability modeling in industrial practice. In *Proceedings of the seventh international workshop on variability modelling of software-intensive systems* (pp. 1–8).

Bertram, N., Fuchs, S., Mischke, J., Palter, R., Strube, G., & Woetzel, J. (2019). Modular construction: From projects to products. *McKinsey & Company: Capital Projects & Infrastructure*, 1–34.

Bilic, D., Sundmark, D., Afzal, W., Wallin, P., Causevic, A., & Amlinger, C. (2018). Model-based product line engineering in an industrial automotive context: An exploratory case study. In *Proceedings of the 22nd international systems and software product line conference-volume 2* (pp. 56–63).

Birk, A., Heller, G., John, I., Schmid, K., von der Maßen, T., & Muller, K. (2003). Product line engineering, the state of the practice. *IEEE software*, *20*(6), 52-60.

Blanchard, B. S., Fabrycky, W. J., & Fabrycky, W. J. (1990). *Systems engineering and analysis* (Vol. 4). Prentice Hall Englewood Cliffs, NJ.

Böckle, G. (2005). Chapter 1 - introduction to software product line engineering. In *Software product line engineering.* Springer. doi: https://doi.org/10.1007/3-540-28901-1_1

Boehm, B. W. (1988). A spiral model of software development and enhancement. *Computer*, *21*(5), 61–72.

Bowman, D. (2006). Effective product platform planning in the front end. In *Product platform and product family design* (pp. 19–26). Springer.

Browning, T. R. (2001). Applying the design structure matrix to system decomposition and integration problems: a review and new directions. *IEEE Transactions on Engineering management*, *48*(3), 292–306.

Buede, D. M., & Miller, W. D. (2016). *The engineering design of systems: models and methods*. John Wiley & Sons.

Cameron, B. G., & Crawley, E. F. (2014). Crafting platform strategy based on anticipated benefits and costs. In *Advances in product family and product platform design* (pp. 49–70). Springer.

Casse, O. (2017). *Sysml in action with cameo systems modeler*. Elsevier.

Cerovsek, T. (2011). A review and outlook for a 'building information model'(bim): A multi-standpoint framework for technological development. *Advanced engineering informatics*, *25*(2), 224–244.

Chen, C.-C., & Crilly, N. (2014). Modularity, redundancy and degeneracy: cross-domain perspectives on key design principles. In *2014 ieee international systems conference proceedings* (pp. 546–553).

Clark, J. O. (2009). System of systems engineering and family of systems engineering from a standards, v-model, and dual-v model perspective. In *2009 3rd annual ieee systems conference* (pp. 381–387).

Clements, P. C. (2014). Product line engineering comes to the industrial mainstream. *INSIGHT*, *22*(2), 7-14.

Clements, P. C. (2019). Product line engineering comes to the industrial mainstream. *INSIGHT*, *22*(2), 7–14.

Colletti, R. A., Qamar, A., Nuesch, S. P., & Paredis, C. J. (2019). Best practice patterns for variant modeling of activities in model-based systems engineering. *IEEE Systems Journal*, *14*(3), 4165–4175.

Dori, D. (2002). *Object-process methodology: a holistic systems paradigm*. Springer Heidelberg.

Eclipse. (n.d.). *Open source solution for model-based systems engineering.* https://www.eclipse.org/capella/. (Accessed: '04-01-2021')

ElMaraghy, H., Schuh, G., ElMaraghy, W., Piller, F., Schönsleben, P., Tseng, M., & Bernard, A. (2013). Product variety management. *Cirp Annals*, *62*(2), 629–652.

Estefan, J. A. (2007). Survey of model-based systems engineering (mbse) methodologies. *Incose MBSE Focus Group*, *25*(8), 1–12.

Friedenthal, S., Moore, A., & Steiner, R. (2014). *A practical guide to sysml: the systems modeling language*. Morgan Kaufmann.

Friedenthal, S., Moore, A., & Steiner, R. (2015a). Chapter 17 - residential security system example using the object-oriented systems engineering method. In S. Friedenthal, A. Moore, & R. Steiner (Eds.), *A practical guide to sysml (third edition)* (Third Edition ed., p. 417 - 504). Boston: Morgan Kaufmann.

Friedenthal, S., Moore, A., & Steiner, R. (2015b). Chapter 18 - integrating sysml into a systems development environment. In S. Friedenthal, A. Moore, & R. Steiner (Eds.), *A practical guide to sysml (third edition)* (Third Edition ed., p. 507 - 541). Boston: Morgan Kaufmann. doi: https://doi.org/10.1016/B978-0-12-800202-5.00018-7

Friedenthal, S., Moore, A., & Steiner, R. (2015c). Chapter 19 - deploying sysml in an organization. In S. Friedenthal, A. Moore, & R. Steiner (Eds.), *A practical guide to sysml (third edition)* (Third Edition ed., p. 543 - 554). Boston: Morgan Kaufmann. doi: https://doi.org/10.1016/B978-0-12-800202-5.00019-9

Friedenthal, S., Moore, A., & Steiner, R. (2015d). Chapter 1 - systems engineering overview. In S. Friedenthal, A. Moore, & R. Steiner (Eds.), *A practical guide to sysml (third edition)* (Third Edition ed., p. 3 - 14). Boston: Morgan Kaufmann. doi: https://doi.org/10.1016/B978-0-12-800202-5.00001-1

Friedenthal, S., Moore, A., & Steiner, R. (2015e). Chapter 2 - model-based systems engineering. In S. Friedenthal, A. Moore, & R. Steiner (Eds.), *A practical guide to sysml (third edition)* (Third Edition ed., p. 15 - 29). Boston: Morgan Kaufmann. doi: https://doi.org/10.1016/B978-0-12-800202-5.00002-3

Friedenthal, S., Moore, A., & Steiner, R. (2015f). Chapter 3 - getting started with sysml. In S. Friedenthal, A. Moore, & R. Steiner (Eds.), *A practical guide to sysml (third edition)* (Third Edition ed., p. 31 - 51). Boston: Morgan Kaufmann. doi: https://doi.org/10.1016/B978-0-12-800202-5.00003-5

Gershenson, J., Prasad, G., & Zhang, Y. (2003). Product modularity: definitions and benefits. *Journal of Engineering design*, *14*(3), 295–313.

Ghaffarianhoseini, A., Tookey, J., Ghaffarianhoseini, A., Naismith, N., Azhar, S., Efimova, O., & Raahemifar, K. (2017). Building information modelling (bim) uptake: Clear benefits, understanding its implementation, risks and challenges. *Renewable and Sustainable Energy Reviews*, *75*, 1046–1053.

Gholizadeh, P., Esmaeili, B., & Goodrum, P. (2018). Diffusion of building information modeling functions in the construction industry. *Journal of Management in Engineering*, *34*(2), 04017060.

Gibb, A., & Isack, F. (2003). Re-engineering through pre-assembly: client expectations and drivers. *Building research & information*, *31*(2), 146–160.

Gotel, O., Cleland-Huang, J., Hayes, J. H., Zisman, A., Egyed, A., Grünbacher, P., ... Mäder, P. (2012). Traceability fundamentals. In *Software and systems traceability* (pp. 3–22). Springer.

Halman, J. I., Hofer, A. P., & van Vuuren, W. (2006). Platform-driven development of product families. In *Product platform and product family design* (pp. 27–47). Springer.

Henderson, K., & Salado, A. (2021). Value and benefits of model-based systems engineering (mbse): Evidence from the literature. *Systems Engineering*, *24*(1), 51–66.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS quarterly*, 75–105.

Huang, C.-C., & Kusiak, A. (1998). Modularity in design of products and systems. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, *28*(1), 66–77.

Hummell, J., & Hause, M. (2015). Model-based product line engineering-enabling product families with variants. In *2015 ieee aerospace conference* (pp. 1–8).

INCOSE. (2012). *Systems engineering handbook: A guide for system life cycle processes and activities*. International Council on Systems Engineering.

Jiao, J. R., Simpson, T. W., & Siddique, Z. (2007). Product family design and platform-based product development: a state-of-the-art review. *Journal of intelligent Manufacturing*, *18*(1), 5–29.

Jonsson, H., & Rudberg, M. (2014). Classification of production systems for industrialized building: a production strategy perspective. *Construction Management and Economics*, *32*(1-2), 53–69.

Jose, A., & Tollenaere, M. (2005). Modular and platform methods for product family design: literature analysis. *Journal of Intelligent manufacturing*, *16*(3), 371–390.

Kang, K. C., Cohen, S. G., Hess, J. A., Novak, W. E., & Peterson, A. S. (1990). *Feature-oriented domain analysis (foda) feasibility study* (Tech. Rep.). Pittsburgh, PA: Carnegie-Mellon University, Software Engineering Institute.

Keskin, B., & Salman, B. (2020). Building information modeling implementation framework for smart airport life cycle management. *Transportation Research Record*, *2674*(6), 98–112.

Klimaatakkoord. (2020). *Afspraken in sectoren - gebouwde omgeving.* Retrieved from https://www.klimaatakkoord.nl/gebouwde-omgeving/documenten/publicaties/2019/06/28/klimaatakkoord-hoofdstuk-gebouwde-omgeving.

Koh, E. C., Förg, A., Kreimeyer, M., & Lienkamp, M. (2015). Using engineering change forecast to prioritise component modularisation. *Research in Engineering Design*, *26*(4), 337–353.

Kruchten, P. (2004). *The rational unified process: an introduction*. Addison-Wesley Professional.

Krueger, C., & Clements, P. (2013). Systems and software product line engineering with biglever software gears. In *Proceedings of the 17th international software product line conference co-located workshops* (pp. 136–140).

Krueger, C. W. (2015). Mechanical product lifecycle management meets product line engineering. In *Proceedings of the 19th international conference on software product line* (pp. 316–320).

Kusiak, A., & Huang, C.-C. (1996). Development of modular products. *IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part A*, *19*(4), 523–538.

Lauenroth, K., & Pohl, K. (2005). Principles of variability. In *Software product line engineering* (pp. 57–88). Springer.

Li, H., Yang, M., & Evans, S. (2019). Classifying different types of modularity for technical system. *International Journal of Technology Management*, *81*(1-2), 1–23.

Madni, A. M., & Sievers, M. (2018). Model-based systems engineering: Motivation, current status, and research opportunities. *Systems Engineering*, *21*(3), 172–190.

Manshanden, W., & Koops, O. (2019). *Prognose woningbouw 2019-2024* (Tech. Rep.). Rotterdam: NEO observatory.

Marion, T. J., & Simpson, T. W. (2006). Platform leveraging strategies and market segmentation. In *Product platform and product family design* (pp. 73–90). Springer.

Maurandy, J., Helm, A., Gill, E., & Stalford, R. (2012). 11.5. 3 cost-benefit analysis of sysml modelling for the atomic clock ensemble in space (aces) simulator. In *Incose international symposium* (Vol. 22, pp. 1726–1745).

Meyer, M. H., & Lehnerd, A. P. (1997). *The power of product platforms*. Simon and Schuster.

Mhenni, F., Choley, J.-Y., Penas, O., Plateaux, R., & Hammadi, M. (2014). A sysml-based methodology for mechatronic systems architectural design. *Advanced Engineering Informatics*, *28*(3), 218–231.

Miettinen, R., & Paavola, S. (2014). Beyond the bim utopia: Approaches to the development and implementation of building information modeling. *Automation in construction*, *43*, 84–91.

Morkevicius, A., Aleksandraviciene, A., Mazeika, D., Bisikirskiene, L., & Strolia, Z. (2017). Mbse grid: A simplified sysml-based approach for modeling complex systems. In *Incose international symposium* (Vol. 27, pp. 136–150).

Nederlandse Emissieautoriteit. (2015). *Klimaatakkoord van parijs.* Retrieved from hhttps://www.emissieautoriteit.nl/onderwerpen/klimaatakkoord-van-parijs.

of Building Sciences, N. I. (2015). *National bim standard-united states® version 3.*

Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. In *Pioneers and their contributions to software engineering* (pp. 479–498). Springer.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of management information systems*, *24*(3), 45–77.

Peltokorpi, A., Olivieri, H., Granja, A. D., & Seppänen, O. (2018). Categorizing modularization strategies to achieve various objectives of building investments. *Construction Management and Economics*, *36*(1), 32–48.

Piran, F. A. S., Lacerda, D. P., Antunes, J. A. V., Viero, C. F., & Dresch, A. (2016). Modularization strategy: analysis of published articles on production and operations management (1999 to 2013). *The International Journal of Advanced Manufacturing Technology*, *86*(1-4), 507–519.

Polit Casillas, R., & Howe, S. A. (2013). Virtual construction of space habitats: Connecting building information models (bim) and sysml. In *Aiaa space 2013 conference and exposition* (p. 5508).

Product Line. (n.d.). *In cambridge dictionary.* Retrieved from https://dictionary.cambridge.org/dictionary/english/product-line.

Ramos, A. L., Ferreira, J. V., & Barceló, J. (2011). Model-based systems engineering: An emerging approach for modern systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *42*(1), 101–111.

Rashid, M., Anwar, M. W., & Khan, A. M. (2015). Toward the tools selection in model based system engineering for embedded systems—a systematic literature review. *Journal of Systems and Software*, *106*, 150–163.

Reichwein, A., & Paredis, C. (2011). Overview of architecture frameworks and modeling languages for model-based systems engineering. In *Proc. asme* (Vol. 1275).

Robertson, D., & Ulrich, K. (1998). Planning for product platforms. *Sloan management review*, *39*(4), 19.

Sanchez, R., & Mahoney, J. T. (1996). Modularity, flexibility, and knowledge management in product and organization design. *Strategic management journal*, *17*(S2), 63–76.

Schilling, M. A. (2000). Toward a general modular systems theory and its application to interfirm product modularity. *Academy of management review*, *25*(2), 312–334.

Seawright, J., & Gerring, J. (2008). Case selection techniques in case study research: A menu of qualitative and quantitative options. *Political research quarterly*, *61*(2), 294–308.

Shafiee, S., Piroozfar, P., Hvam, L., Farr, E. R., Huang, G. Q., Pan, W., ... Korell, M. (2020). Modularisation strategies in the aec industry: a comparative analysis. *Architectural Engineering and Design Management*, 1–23.

Siebelink, S., Voordijk, J. T., & Adriaanse, A. (2018). Developing and testing a tool to evaluate bim maturity: Sectoral analysis in the dutch construction industry. *Journal of construction engineering and management*, *144*(8), 05018007.

Sillitto, H., Martin, J., McKinney, D., Griego, R., Dori, D., Krob, D., ... Jackson, S. (2019). *Systems engineering and system definitions.* Retrieved from https://www.incose.org/docs/default-source/default-document-library/incose-se-definitions-tp-2020-002-06.pdf?sfvrsn=b1049bc6_0.

Simpson, T. W., Siddique, Z., & Jiao, J. R. (2006). Platform-based product family development. In *Product platform and product family design* (pp. 1–15). Springer.

Sociaal-Economische Raad. (2020). *Gebouwde omgeving.* Retrieved from https://www.ser.nl/nl/thema/energie-en-duurzaamheid/energieakkoord/ /link.aspx?_id=D88E14B4D54B43808C

Sohlenius, G. (1992). Concurrent engineering. *CIRP annals*, *41*(2), 645–655.

SysML.org. (2020, May). *Free & commercial sysml tools for mbse.* https://sysmltools.com. (Accessed: '19-11-2020')

TNO. (2016). *Versnelde renovatie van woningen en gebouwen.* `https://www.tno.nl/nl/aandachtsgebieden/energietransitie/roadmaps/naar-een-energie-producerende-gebouwde-omgeving/versnelde-renovatie-van-woningen-en-gebouwen/`. (Accessed: 15-12-2020)

Tomarken, A. J., & Serlin, R. C. (1986). Comparison of anova alternatives under variance heterogeneity and specific noncentrality structures. *Psychological Bulletin*, *99*(1), 90.

Ulrich, K. (1994). Fundamentals of product modularity. In *Management of design* (pp. 219–231). Springer.

Ulrich, K. (1995). The role of product architecture in the manufacturing firm. *Research policy*, *24*(3), 419–440.

United Nations. (2015). *Paris agreement.* Retrieved from https://unfccc.int/process-and-meetings/the-paris-agreement/the-paris-agreement.

Valdes, F., Gentry, R., Eastman, C., & Forrest, S. (2016). Applying systems modeling approaches to building construction. In *Isarc. proceedings of the international symposium on automation and robotics in construction* (Vol. 33, p. 1).

Van der Linden, F. J., Schmid, K., & Rommes, E. (2007). Chapter 1 - the product line engineering approach. In F. J. Van der Linden, K. Schmid, & E. Rommes (Eds.), *Software product lines in action: the best industrial practice in product line engineering* (First edition ed., p. 3-20). Berlin: Springer Science & Business Media.

Voordijk, H., Meijboom, B., & de Haan, J. (2006). Modularity in supply chains: a multiple case study in the construction industry. *International journal of operations & production management*.

Wee, T., Aurisicchio, M., & Starzyk, I. (2017). Evaluating modularisation tools in construction. In *Isarc. proceedings of the international symposium on automation and robotics in construction* (Vol. 34).

Weilkiens, T., Scheithauer, A., Di Maio, M., & Klusmann, N. (2016). Evaluating and comparing mbse methodologies for practitioners. In *2016 ieee international symposium on systems engineering (isse)* (pp. 1–8).

Wieringa, R. J. (2014). *Design science methodology for information systems and software engineering*. Springer.

Wozniak, L., & Clements, P. (2015). How automotive engineering is taking product line engineering to the extreme. In *Proceedings of the 19th international conference on software product line* (pp. 327–336).

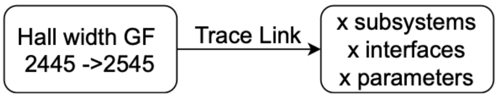
A Modelling approaches of MBSE with SysML

Mhenni et al. (2014)	Casse (2017)	Morkevicius (2017)
Black-Box	**Operational analysis**	**Problem domain - Black-Box**
1. Global mission	1. User requirements	1. Stakeholder needs
2. Life cycle phases	2. Visualisation of requirements	2. Use cases
3. System context	3. Mission, vision, goals	3. System context
4. External interfaces	4. System context	4. Measurement of effectiveness
5. User operating modes	5. Expected functionalities	**Problem domain - White-Box**
6. Services provided by the system(functions)	6. Use scenarios	1. Stakeholder needs
7. Functional scenario's	7. System modes	2. Functional analysis
8. Requirements specification	**System requirement analysis**	3. Logical subsystem communication
9. Requirements traceability	1. Mapping of user requirements to system requirements	4. Measurement of effectiveness of subsystems
White-Box	2. External interfaces	**Solution domain**
1. Functional architecture	3. System scenario's	1. System requirements, behaviour, structure and parameters
2. Logical breakdown of system and allocation functions to subsystems	4. Function traceability	2. Subsystem requirements, behaviour, structure and parameters
3. Requirements to logical subsystem traceability	**Logical architecture design**	3. Component requirements, behaviour, structure and parameters
4. Logical architecture	1. Main functions	**Implementation domain**
5. Parametric diagrams	2. Internal interfaces	1. Physical requirements
6. Allocation of logical to physical structure	3. Block behaviour	2. System integration and verification
7. Physical architecture	4. Allocation of requirements to subsystems	
	Physical architecture design	
	1. Candidate solutions	
	2. Physical interfaces	
	3. Constraints	
	4. Allocation of functions to physical subsystems	

Table 13 – Overview of the steps of three modelling approaches of MBSE with SysML

B Answer sheet for the experiment

Scenario 1: Aanpassing ontwerp. Breedte van de verkeersruimte op de begane grond schuift van 2445 naar 2545mm.



Subsystemen:

Interfaces

Parameters

Figure 46 – Answer sheet used during the experiment

C Results of the experiment

Scenario	MBSE Beginner	MBSE Expert	BIM Expert
Scenario 1	0	93	80
Scenario 2	17	96	83
Scenario 3	30	85	50
Scenario 4	55	82	64
Scenario 5	100	100	100
Scenario 6	63	75	63
Scenario 7	50	100	0
Scenario 8	50	100	0
Scenario 9	71	71	14
Scenario 10	67	89	33
Scenario 11	43	86	14

Table 14 – Ratio of completeness of each scenario for the different groups

Scenario	MBSE Beginner	MBSE Expert	BIM Expert
Scenario 1	0	93	80
Scenario 2	17	96	83
Scenario 3	30	85	50
Scenario 4	55	73	64
Scenario 5	100	100	100
Scenario 6	63	75	63
Scenario 7	50	100	0
Scenario 8	50	100	0
Scenario 9	71	71	14
Scenario 10	67	78	33
Scenario 11	43	86	14

Table 15 – Ratio of correctness of each scenario for the three different groups

Scenario	MBSE Beginner	MBSE Expert	BIM Expert
Scenario 1	13:00	08:50	09:15
Scenario 2	10:22	06:54	09:07
Scenario 3	20:01	08:50	19:49
Scenario 4	06:41	10:41	14:35
Scenario 5	02:30	01:00	01:00
Scenario 6	03:36	05:26	03:36
Scenario 7	01:20	02:01	08:11
Scenario 8	04:00	04:15	06:49
Scenario 9	05:40	02:50	07:02
Scenario 10	05:15	03:48	09:53
Scenario 11	06:03	02:33	02:43

Table 16 – Time needed to finish each scenario for the different groups

D Descriptive statistics experiment data

		N	Mean	Std. Deviation	Std. Error	95% Confidence Interval for Mean		Min	Max
						Lower Bound	Upper Bound		
Compl. Total	Control Group	5	70,60	17,184	7,685	49,26	91,94	50	93
	Treatment Group Beginner Level	5	33,00	26,163	11,700	0,51	65,49	0	63
	Treatment Group Expert Level	5	83,60	7,829	3,501	73,88	93,32	75	96
	Total	15	62,40	28,119	7,260	46,83	77,97	0	96
Correct. Total	Control Group	5	75,00	10,654	4,764	61,77	88,23	63	85
	Treatment Group Beginner Level	5	33,00	26,163	11,700	0,51	65,49	0	63
	Treatment Group Expert Level	5	77,40	18,474	8,262	54,46	100,34	50	96
	Total	15	61,80	27,765	7,169	46,42	77,18	0	96
Time	Control Group	5	11:16	06:09	02:45	03:37	18:54	03:36	19:49
	Treatment Group Beginner Level	5	10:44	06:17	02:49	02:54	18:33	03:36	20:01
	Treatment Group Expert Level	5	08:08	02:01	00:54	05:37	10:38	05:26	10:41
	Total	15	10:02	05:02	01:17	07:15	12:50	03:36	20:01
Compl. Req.	BIM Expert	5	12,20	13,572	6,070	-4,65	29,05	0	33
	MBSE Beginner	5	56,20	12,112	5,417	41,16	71,24	43	71
	MBSE Expert	5	89,20	11,987	5,361	74,32	104,08	71	100
	Total	15	52,53	34,663	8,950	33,34	71,73	0	100
Correct. Req.	BIM Expert	5	12,20	13,572	6,070	-4,65	29,05	0	33
	MBSE Beginner	5	56,20	12,112	5,417	41,16	71,24	43	71
	MBSE Expert	5	87,00	13,000	5,814	70,86	103,14	71	100
	Total	15	51,80	33,946	8,765	33,00	70,60	0	100
Time Req.	BIM Expert	5	06:55	02:38	01:11	03:38	10:12	02:43	09:53
	MBSE beginner	5	04:27	01:54	00:51	02:05	06:49	01:20	06:03
	MBSE Expert	5	03:05	00:54	00:24	01:57	04:13	02:01	04:15
	Total	15	04:49	02:26	00:37	03:28	06:10	01:20	09:53

E Results Shapiro-Wilk test

Time System Traceability	Statistic	df	Sig.
BIM Expert	0.966	5	0.846
MBSE Beginner	0.974	5	0.902
MBSE Expert	0.960	5	0.811
Completeness System Traceability			
BIM Expert	0.945	5	0.700
MBSE Beginner	0.952	5	0.749
MBSE Expert	0.938	5	0.651
Correctness System Traceability			
BIM Expert	0.813	6	0.102
MBSE Beginner	0.952	6	0.749
MBSE Expert	0.918	6	0.517
Time Requirements			
BIM Expert	0.924	5	0.557
MBSE Beginner	0.857	5	0.218
MBSE Expert	0.950	5	0.735
Completeness Requirement Traceability			
BIM Expert	0.870	5	0.265
MBSE Beginner	0.879	5	0.305
MBSE Expert	0.893	5	0.371
Correctness Requirement Traceability			
BIM Expert	0.870	5	0.265
MBSE Beginner	0.879	5	0.305
MBSE Expert	0.894	5	0.378

F Results Tukey HSD test

Dependent Variable	(I) Experiment Group	(J) Experiment Group		Std. Error	Sig.	95% Confidence Interval	
						Lower Bound	Upper Bound
Compl. System Trace.	BIM Expert	MBSE Beginner	37.600*	11.782	.020	6.17	69.03
		MBSE Expert	-13.000	11.782	.530	-44.43	18.43
	MBSE Beginner	BIM Expert	-37.600*	11.782	.020	-69.03	-6.17
		MBSE Expert	-50.600*	11.782	.003	-82.03	-19.17
	MBSE Expert	BIM Expert	13.000	11.782	.530	-18.43	44.43
		MBSE Beginner	50.600*	11.782	.003	19.17	82.03

Table 17 – Results of the Tukey HSD test for the dependent variable completeness system traceability

Dependent Variable	(I) Experiment Group	(J) Experiment Group		Std. Error	Sig.	95% Confidence Interval	
						Lower Bound	Upper Bound
Correct. System Trace.	BIM Expert	MBSE Beginner	42.000*	12.325	.013	9.12	74.88
		MBSE Expert	-2400	12.325	.979	-35.28	30.48
	MBSE Beginner	BIM Expert	-42.000*	12.325	.013	-74.88	-9.12
		MBSE Expert	-44.400*	12.325	.009	-77.28	-11.52
	MBSE Expert	BIM Expert	2400	12.325	.979	-30.48	35.28
		MBSE Beginner	44.400*	12.325	.009	11.52	77.28

Table 18 – Results of the Tukey HSD test for the dependent variable correctness system traceability

Dependent Variable	(I) Experiment Group	(J) Experiment Group		Std. Error	Sig.	95% Confidence Interval	
						Lower Bound	Upper Bound
Compl. System Trace.	BIM Expert	MBSE Beginner	-44.000*	7.955	.000	-65.22	-22.78
		MBSE Expert	-77.000*	7.955	.000	-98.22	-55.78
	MBSE Beginner	BIM Expert	44.000*	7.955	.000	22.78	65.22
		MBSE Expert	-33.000*	7.955	.004	-54.22	-11.78
	MBSE Expert	BIM Expert	77.000*	7.955	.000	55.78	98.22
		MBSE Beginner	33.000*	7.955	.004	11.78	54.22

Table 19 – Results of the Tukey HSD test for the dependent variable completeness requirement traceability

Dependent Variable	(I) Experiment Group	(J) Experiment Group		Std. Error	Sig.	95% Confidence Interval	
						Lower Bound	Upper Bound
Compl. System Trace.	BIM Expert	MBSE Beginner	-44.000*	8.164	.000	-65.78	-22.22
		MBSE Expert	-74.800*	8.164	.000	-96.58	-53.02
	MBSE Beginner	BIM Expert	44.000*	8.164	.000	22.22	65.78
		MBSE Expert	-30.800*	8.164	.007	-52.58	-9.02
	MBSE Expert	BIM Expert	74.800*	8.164	.000	53.02	96.58
		MBSE Beginner	30.800*	8.164	.007	9.02	52.58

Table 20 – Results of the Tukey HSD test for the dependent variable correctness requirement traceability

Dependent Variable	(I) Experiment Group	(J) Experiment Group		Std. Error	Sig.	95% Confidence Interval	
						Lower Bound	Upper Bound
Compl. System Trace.	BIM Expert	MBSE Beginner	02:28	01:14	.157	*****	05:46
		MBSE Expert	03:50*	01:14	.023	00:31	07:08
	MBSE Beginner	BIM Expert	*****	01:14	.157	*****	00:50
		MBSE Expert	01:22	01:14	.529	*****	04:40
	MBSE Expert	BIM Expert	******	01:14	.023	*****	*****
		MBSE Beginner	*****	01:14	.529	*****	01:56

Table 21 – Results of the Tukey HSD test for the dependent variable time requirement traceability

Experiment Group	N	Subset for alpha = 0.05	
		1	2
Treatment Group Beginner Level	5	33.00	
Control Group	5		70.60
Treatment Group Expert Level	5		83.60
Sig.		1.000	.530

Table 22 – Results Tukey HSD for the dependent variable completeness system traceability

Experiment Group	N	Subset for alpha = 0.05	
		1	2
Treatment Group Beginner Level	5	33.00	
Control Group	5		75.00
Treatment Group Expert Level	5		77.40
Sig.		1.000	.979

Table 23 – Results Tukey HSD for the dependent variable correctness system traceability

Experiment Group	N	Subset for alpha = 0.05		
		1	2	3
Control Group	5	12.20		
Treatment Group Beginner Level	5		56.20	
Treatment Group Expert Level	5			89.20
Sig.		1.000	1.000	1.000

Table 24 – Results Tukey HSD for the dependent variable completeness requirement traceability

Experiment Group	N	Subset for alpha = 0.05		
		1	2	3
Control Group	5	12.20		
Treatment Group Beginner Level	5		56.20	
Treatment Group Expert Level	5			87.00
Sig.		1.000	1.000	1.000

Table 25 – Results Tukey HSD for the dependent variable correctness requirement traceability

Experiment Group	N	Subset for alpha = 0.05	
		1	2
Treatment Group Expert Level	5	03:05	
Treatment Group Beginner Level	5	04:27	04:27
Control Group	5		06:55
Sig.		0.529	0.157

Table 26 – Results Tukey HSD for the dependent variable time requirement traceability

Based on mean	**Levene Statistic**	**df1**	**df2**	**Sig.**
Completeness system traceability	2,306	1	10	,160

Table 27 – Results Levene's test comparing BIM expert group with MBSE beginner group

Based on mean	**Levene Statistic**	**df1**	**df2**	**Sig.**
Completeness system traceability	6,096	1	10	,033

Table 28 – Results Levene's test comparing BIM expert group with MBSE expert group

Based on mean	**Levene Statistic**	**df1**	**df2**	**Sig.**
Completeness system traceability	7,526	1	10	,021

Table 29 – Results Levene's test comparing MBSE expert group with MBSE beginner group

G Annex - Confidential

G.1 Terrace house of Plegt-Vos

The hallway core is part of a terrace house, with two floors and an attic. The ground floor contains a hallway module and a living room with open kitchen. The first floor has a separate hallway module, three bedrooms and a bathroom. Finally, the attic has no living space purpose, however contains all the technical installations (e.g. water boiler and ventilation) and a connection for a washing machine. The hallway plays an important role in the design of the final residence. The hallway provides all the subsystems with fresh air, water and electricity by means of water pipes, sewage, ventilation shafts and electricity cables. Defining, designing and optimising the layout of all components and connections in the hallway core is challenging and time consuming. Additionally, freezing the design of the makes it possible to produce the module at a manufacturing facility for multiple buildings instead of producing it at the construction site for a single building. Therefore, Plegt-Vos attempts to include the hallway core in multiple different systems without changing in the design.

The hallway core consist out of three modules, namely the hallway module on the ground floor, the hallway module on the first floor, and the SKID module (which is part of the attic). It plays an important role in the development of new systems at Plegt-Vos. The company attempts to include the hallway core in a variety of systems without changing the design. Since, it is not possible to include one type of hallway core design in each residence or building, the company developed a few variants of hallway cores. The residence design is adapted and changed around a selected hallway core in the product portfolio of Plegt-Vos. Therefore, a hallway core can be seen as the platform of the system.

G.2 Customer wishes formulated by the PMC department of Plegt-Vos

PMC boodschappenlijst				
Doelgroep: 2 tot 5 persoonshuishouden, samenwonend, samenwonend met gezinsambitie, compacte tot middelgrote gezinnen				
Ruimtelijk & Functioneel	**Duurzaamheid/Energie**	**Afwerking**	**Verschijning**	**Financieel**
Beukmaat 5,4, binnenmaats 5,1m	EPC 0,4 / BENG norm CGB 352	Keuken met apparatuur	Langskap	Scherpe kostprijs ivm gezonde ontwikkelwinst (eigen ontwikkeling)
Diepte 8,5m tot 10m	Stadsverwarming/ WKO 'ready' CGB 380	Badkamer incl. basis tegelwerk en sanitair	Dwarskap CGB-344	Scherpe kostprijs ivm concurrentiepositie ism externe opdrachtgever
Straat- en tuingericht wonen mogelijk	Aandacht voor positie ventilatie (ivm geluid en beeld niet tegen achtergevel)	Flexibiliteit in keukenopstelling (voor/ midden/ achter)	Combinatielangs- en dwarskap	Kostprijs tussenwoning 2-lagen met kap vanaf: € 110.000
Woonkamer >22 m2			Assymetrische nok CGB-404	Kostprijs hoekwoning 2-woonlagen met kap vanaf: \eur{124.500}
Keuken >10 m2, min. 4000mm diepte			Voldoende ontwerpvrijheid voor gevelopeningen tbv reageren op beeldkwaliteitsplannen (BKP's)	
Totale leefruimte >32m2			Aandacht voor lichtinval (liever iets te veel, wordt men blij van)	
Ruimte voor tafelopstelling 2500x2500 als onderdeel leefruimte (matjes morgen overlappen)			Materialisatie (look & feel): baksteen, hout, beplating, natuursteen, etc.	
Entreematje 1500x1500mm / 1350x1850mm			Verschillende architectuurstijlen:	
vaste trap in hal/entree			Jaren '30 (overstek en betonbanden)	
Garderobe ruimte in entreehal			Jaren '60 (grote kozijnen en ingetogen)	
Verkeersruimte hal minimaal 1100mm breed			Stedelijk (dwars/plat)	
Bergruimte binnen 2-3 m2 (kinderwagen, stofzuiger, proviand, etc.)			Landschappelijk wonen / schuurwoning	
3 slaapkamers mogelijk op verdieping			Tuindorpstijl	
Hoofdslaapkamer >13 m2				
Hoofdslaapkamer min. 3000mm breed				
Hoofdslaapkamer kastruimte voor 650mm diep				
Slaapkamer 2 en 3 >7m2				
Slaapkamer 2 en 3 min. 2100 breed				
Slaapkamer 2 en 3 bureau- en kastopstelling				
Badkamer >5 m2				
2e toilet op badkamer				
Vaste trap naar zolderverdieping				
Techniek geclusterd tbv effectief ruimtegebruik				

Table 30 – Requirements for the basis of the reference residence of Plegt-Vos

PMC boodschappenlijst				
Ruimtelijk & Functioneel	**Duurzaamheid/Energie**	**Afwerking**	**Verschijning**	**Financieel**
Ligbad in badkamer	Energiepakket EPC 0,0	Keuze in tegelwerk en sanitair (wellicht pakketten in verschillende prijsklassen aanbieden);		
Extra dakraam	Energiepakket NOM - CGB-355	2e wastafel		
Dakkapel	Zonnepanelen (voorkeur indaks)	Douchewand		
Uitbouw 1,2 m	Koeling	Binnendeuren- en kozijnen (staal, stomp, etc.)		
Uitbouw 2,4 m		Wandafwerking (stuc, glasvlies, gesausd, etc.)		
middenbadkamer slaapkamer voorzijde over breedte van de woning CGB-357		Vloerafwerking (gietvloer, hout, tegel, PVC, laminaat, etc.)		
		Zonwering CGB-263		
		Inbouwkasten		
		Buitenkraan		
		Ruime keuze in elektra (wcd's)		

Table 31 – Options for the reference residence of Plegt-Vos

G.3 Variant and variant points in the terrace house of Plegt-Vos

Variant point	Variant
Shower wall	Stone Glass Other
Bathroom position	Front side Back side
Bathroom finish	Package 1 Package 2 Package 3
Bathroom	Bathtub 2e sink
Kitchen position	Backside Front side Middle
Living room position	Garden facing Street facing
Master bedroom	Built-in wardrobe
Attic	Knee bulkheads Dormer Extra skylight Solar panels
Roof	Asymmetrical Long hood Cross cap Combo hood
Heat pump	cooling

Variant point	Variant
Building structure	Blinds Outside tap
Architecture styles	The 30's The 60's Urban Scenic Garden village
Extension	Big extension Small extension
Inside door & window frames	Steel Stub Other
Floor finish	cast floor Laminate Wood Tile Other
Wall finish	Doused Plastered Glass fibre Other

Table 32 – Variant points and variants for the terrace house of Plegt-Vos

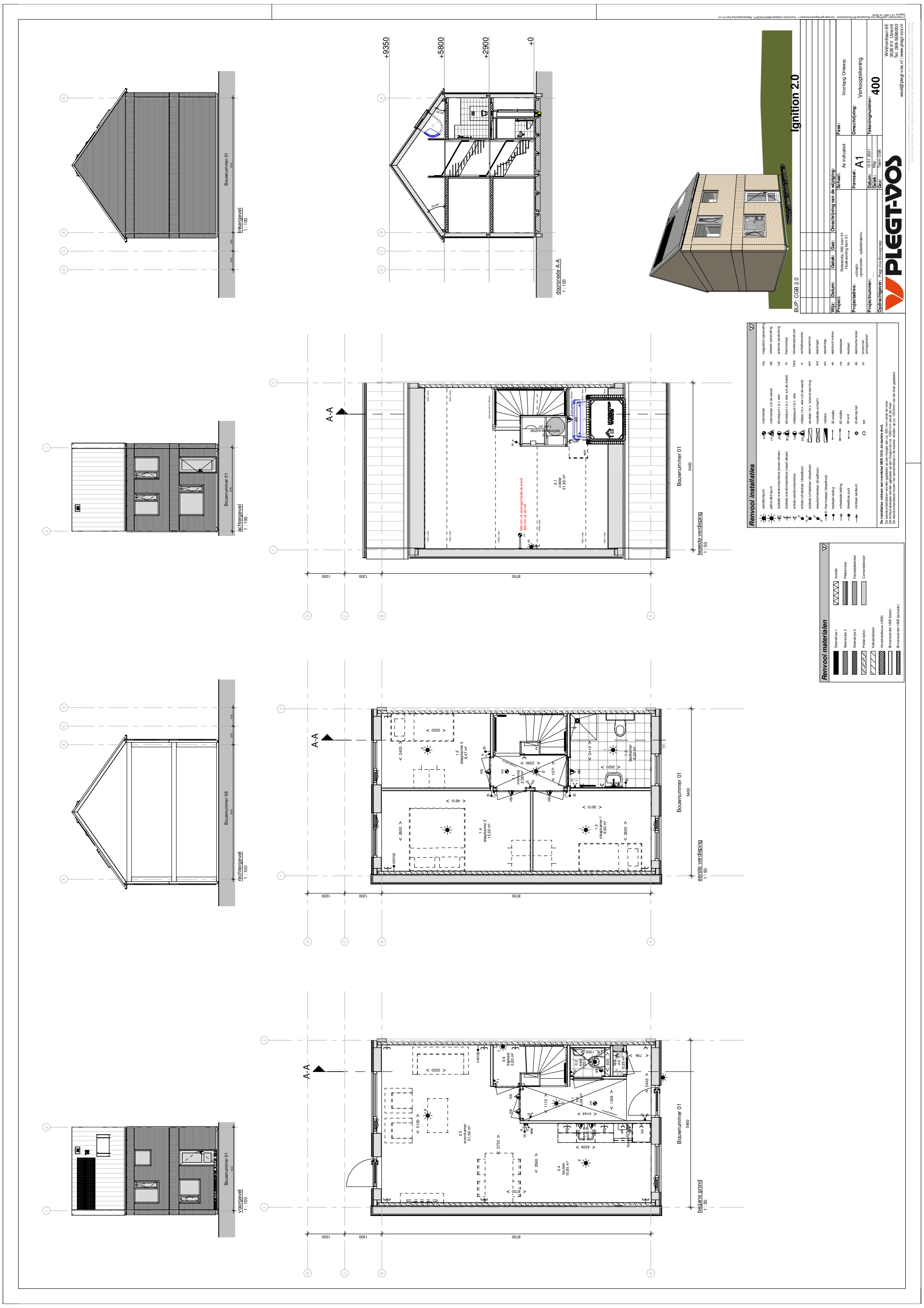


Figure 47 – Plegt-Vos terrace house

G.4 Variability model of the terrace house product line of Plegt-Vos

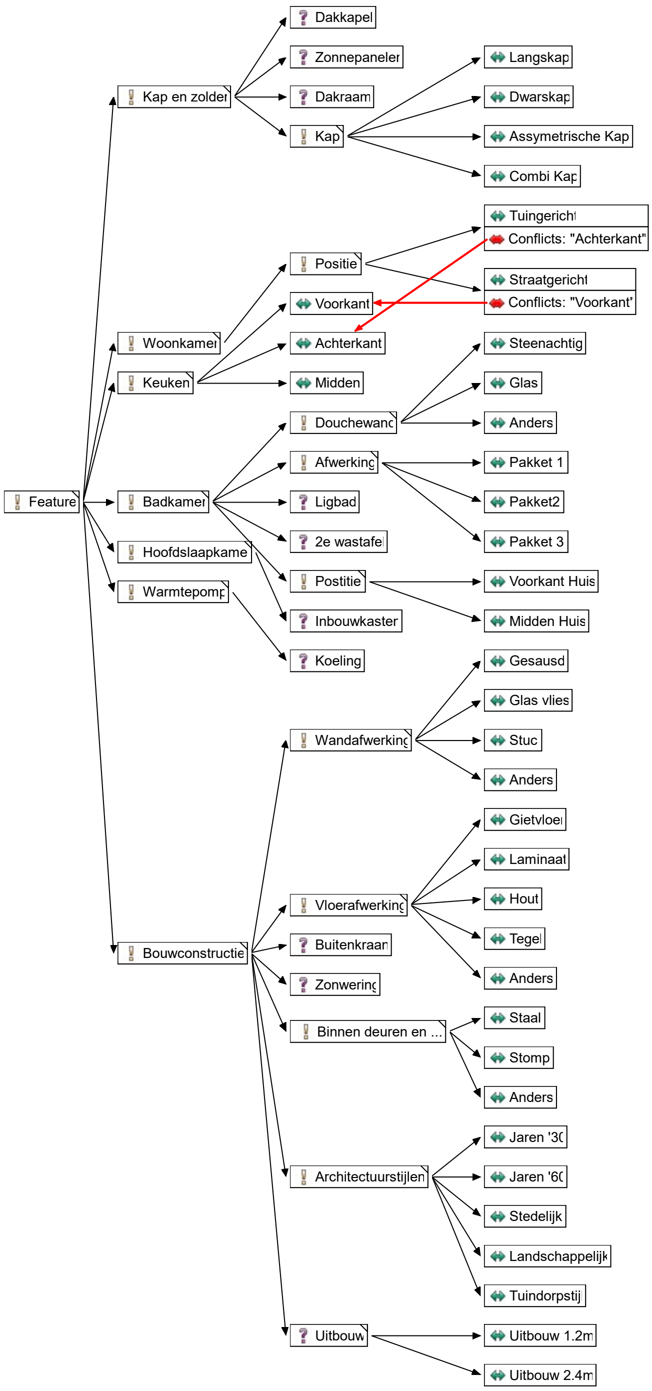


Figure 48 – Variability model of the product line of terrace houses of Plegt-Vos

G.5 Derived system requirements from stakeholder needs

Figure 49 – Overview of derived system requirements from the stakeholder needs

G.6 Internal block diagram of the Plegt-Vos residence

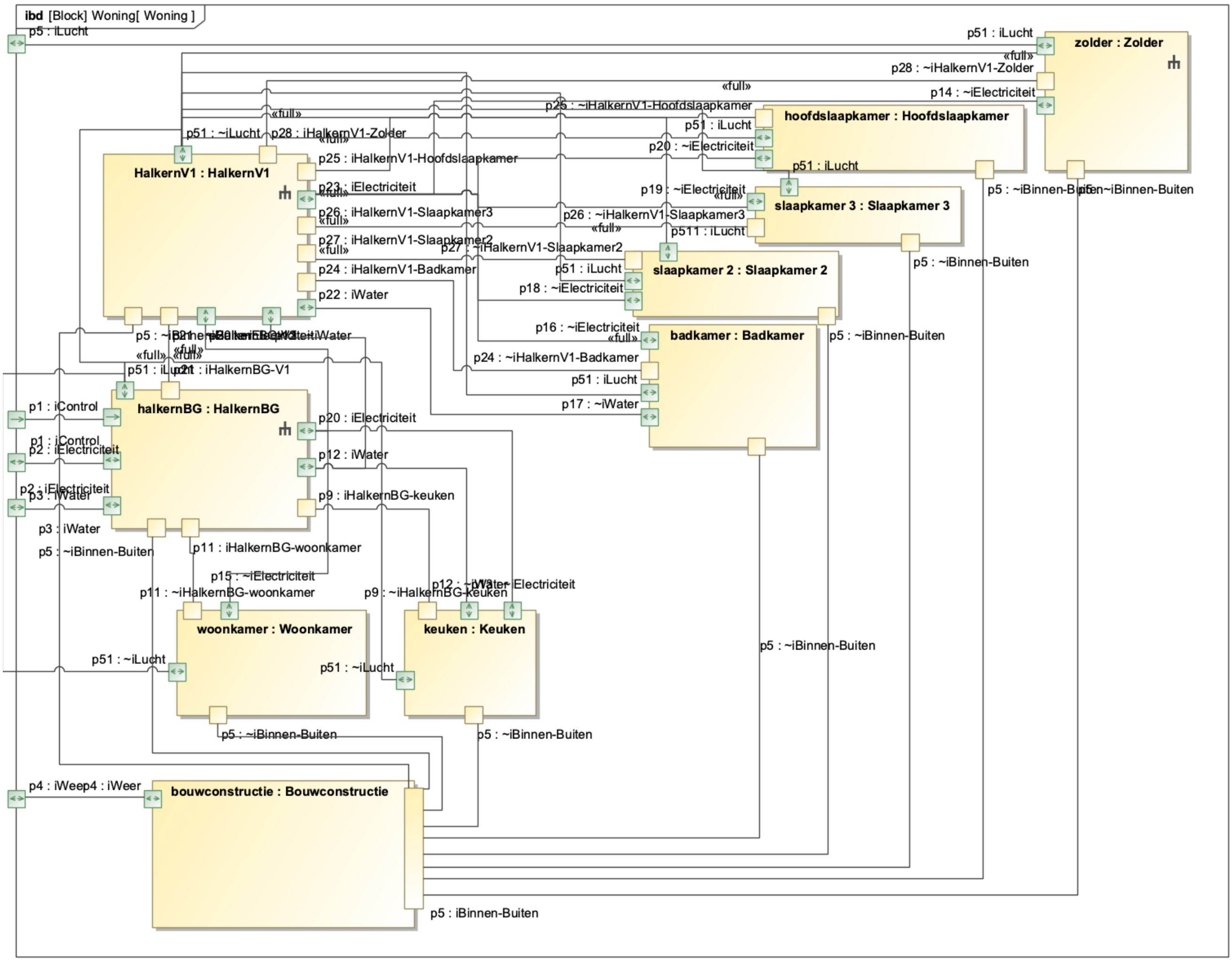


Figure 50 – Internal system structure with interfaces of the terrace house of Plegt-Vos

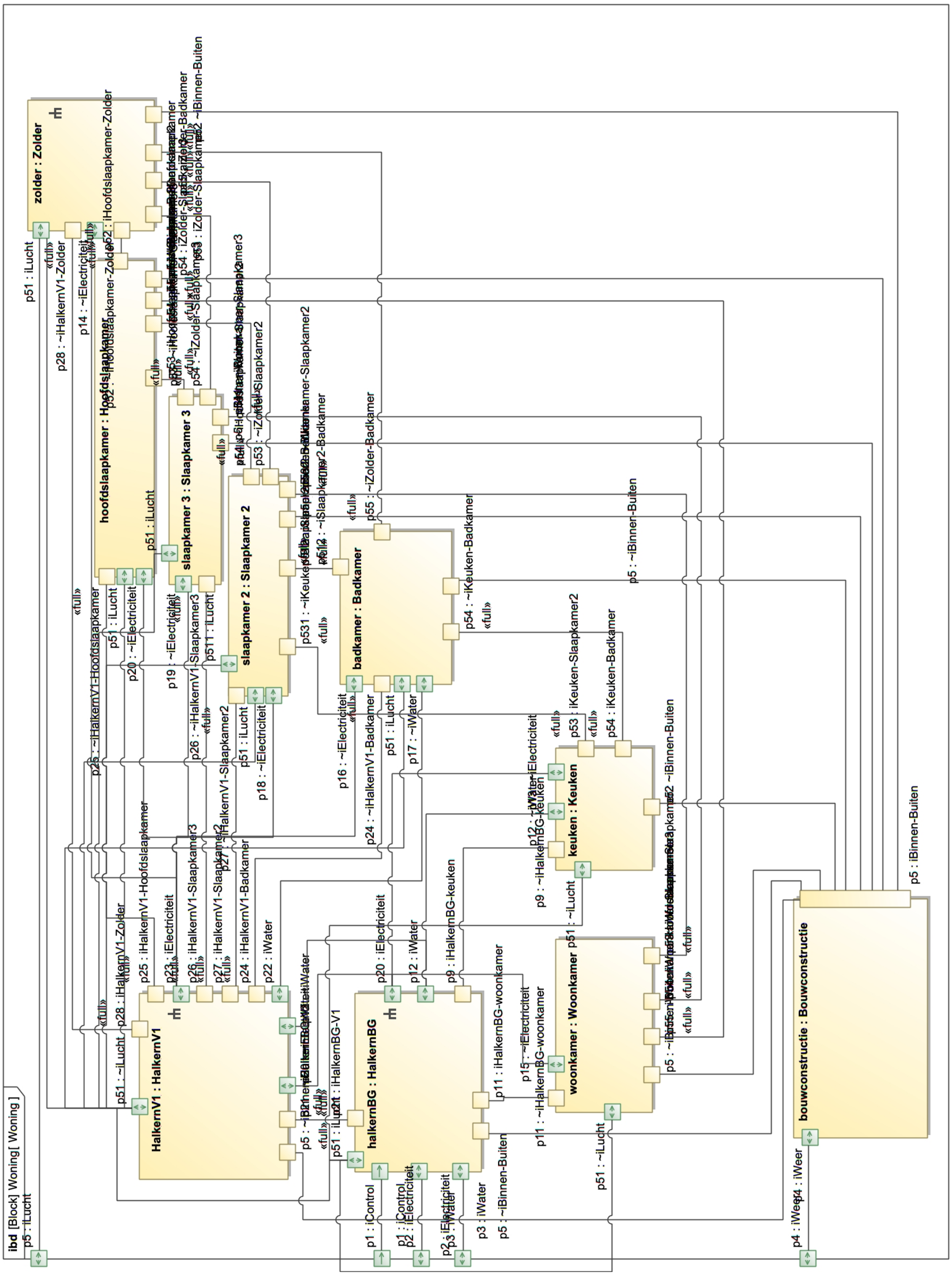


Figure 51 – Updated internal system structure with interfaces of the terrace house of Plegt-Vos

G.7 Technical installation and pipes within the design of the terrace house of Plegt-Vos

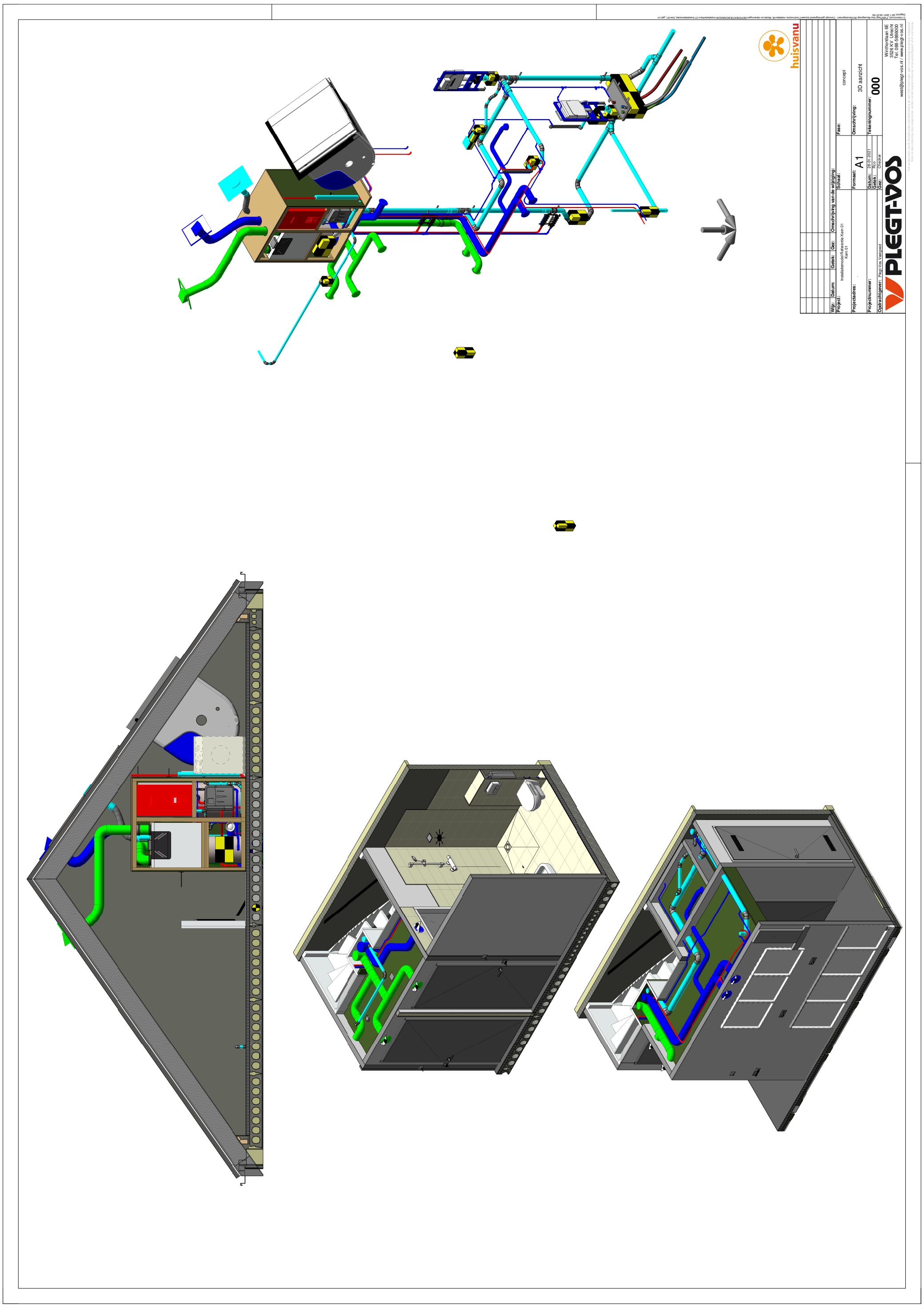


Figure 52 – Technical installations in the hallway core

G.8 Subsystem structures

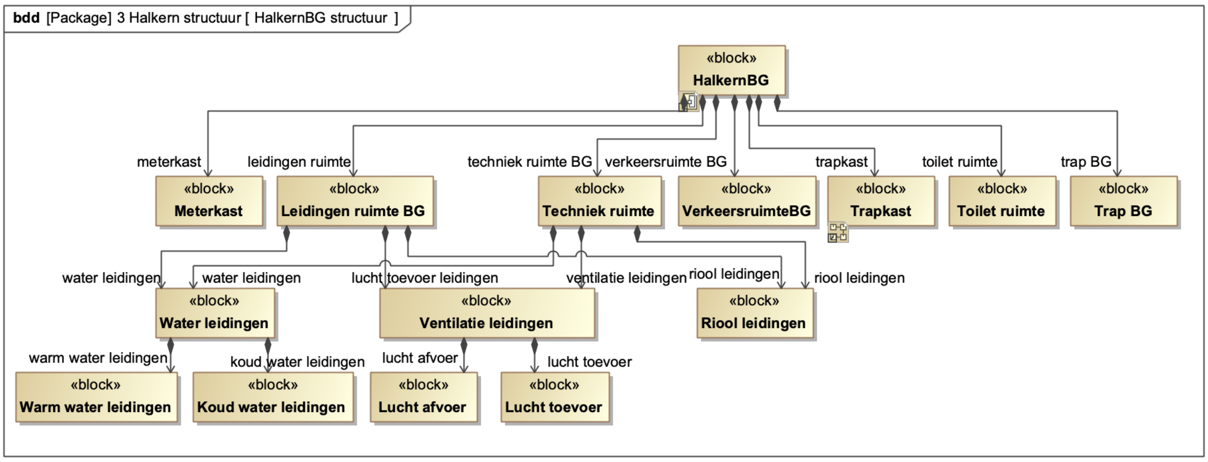


Figure 53 – Structure of the hallway module ground floor

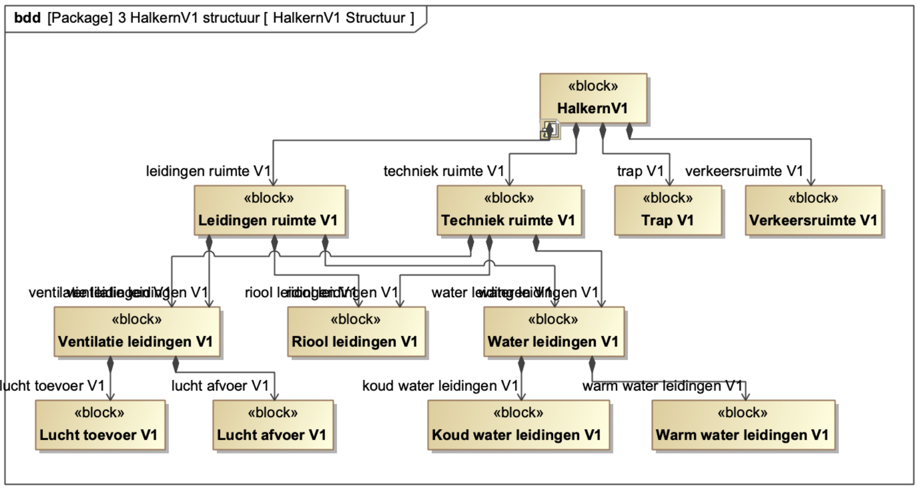


Figure 54 – Structure of the hallway module first floor

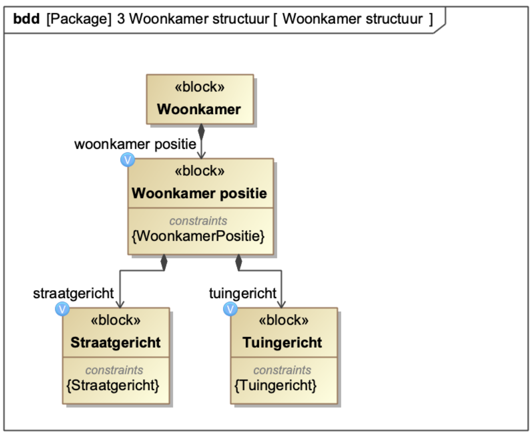


Figure 55 – Structure of the living room module

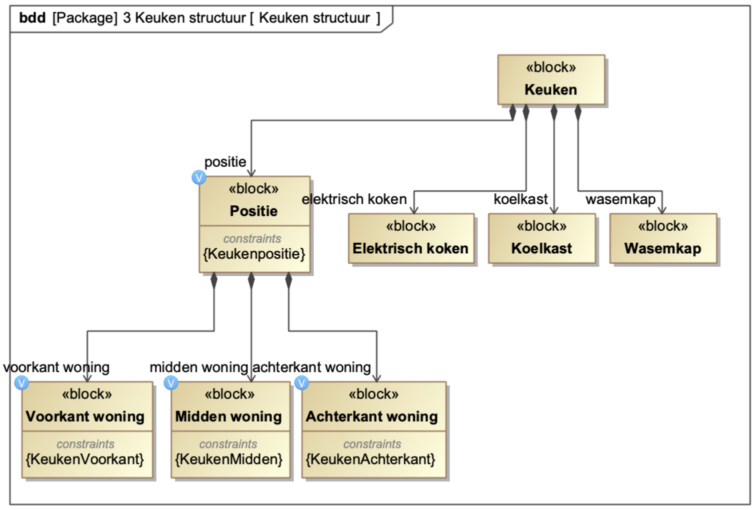


Figure 56 – Structure of the kitchen module

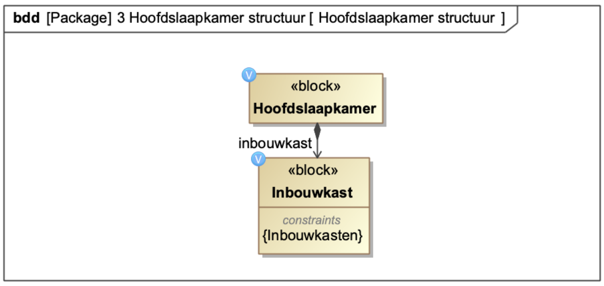


Figure 57 – Structure of the master bedroom module

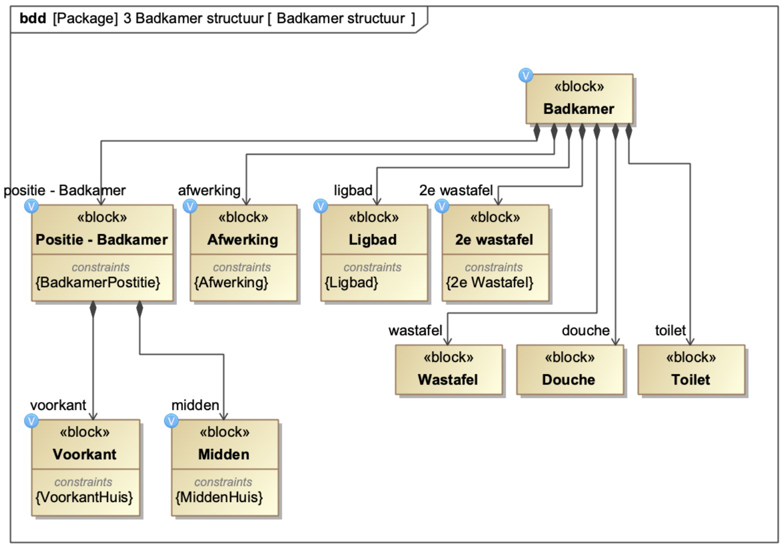


Figure 58 – Structure of the bathroom module

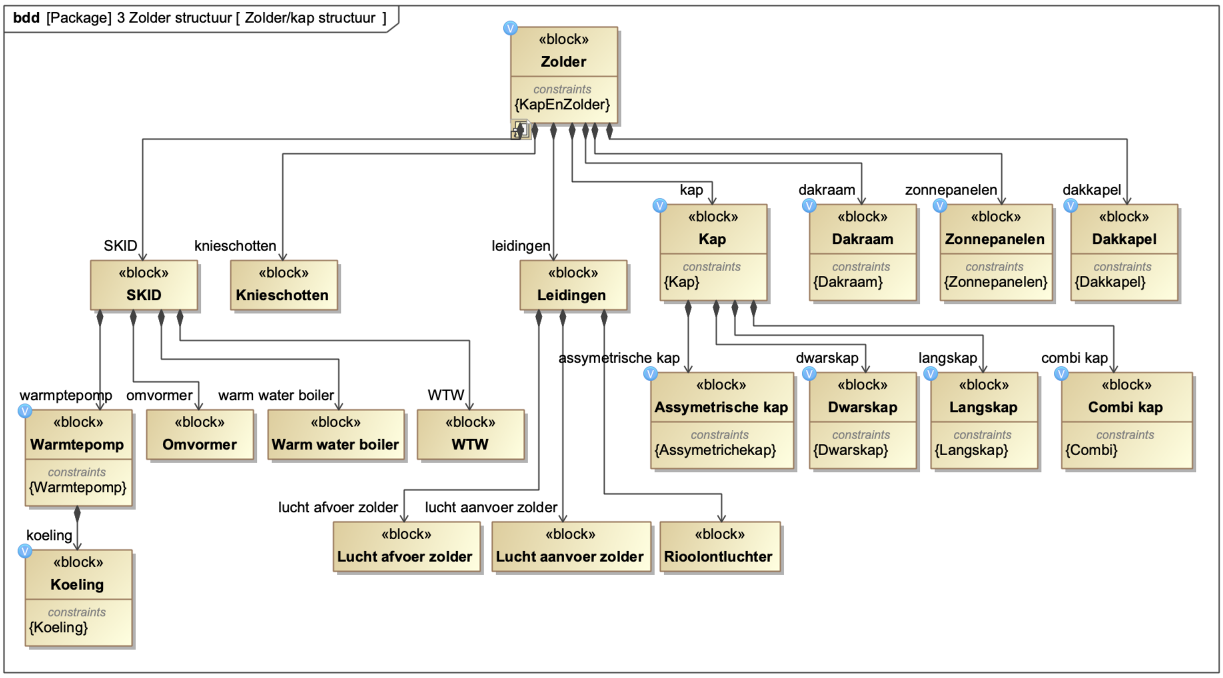


Figure 59 – Structure of the attic module

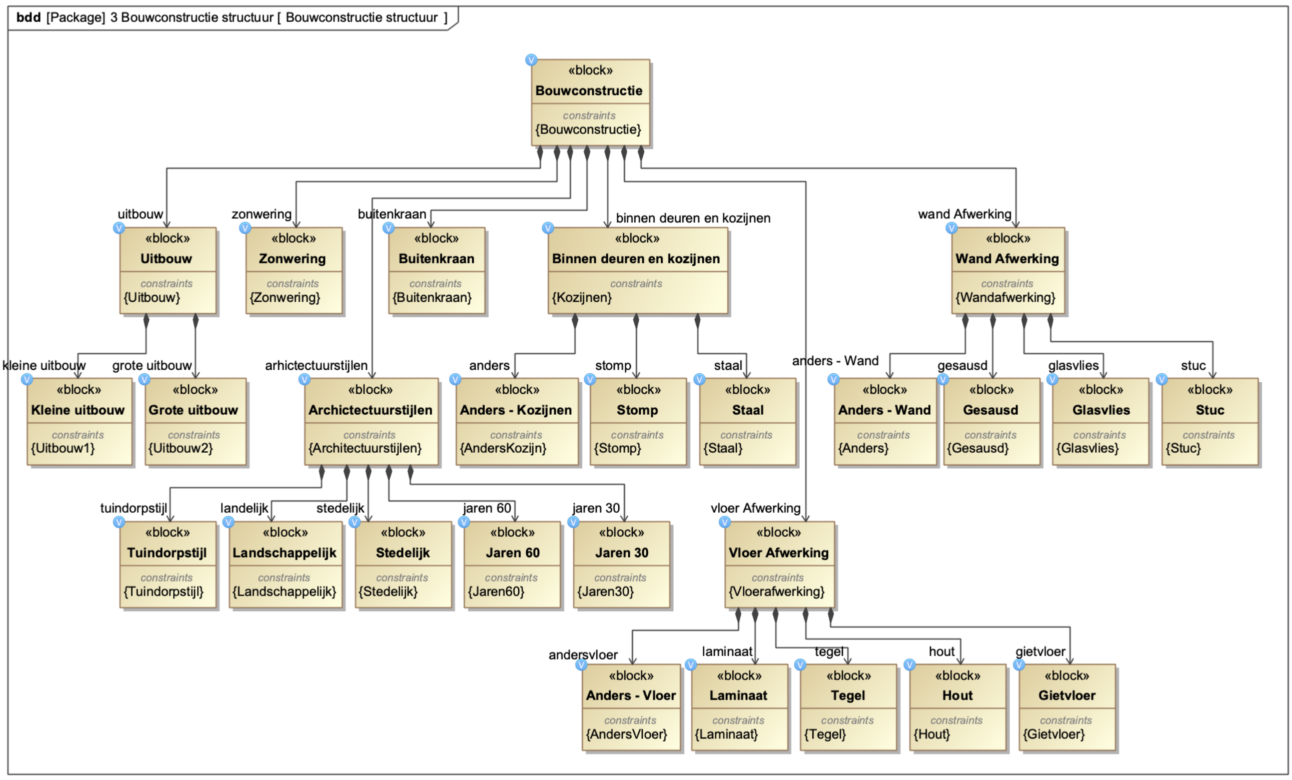


Figure 60 – Structure of the construction module

G.9 Satisfaction matrix and relation map diagram

Legend
↗ Satisfy
⇢ Satisfy (Implied)

Element	SR-1 System Requirements	SR-1.1 Ruimtelijk & Functioneel	SR-1.1.1 Halkern	SR-1.1.1.1 Locatie BG	SR-1.1.1.2 Toilet	SR-1.1.1.3 Locatie V1	SR-1.1.1.4 Toilet breedte	SR-1.1.1.5 Toilet lengte	SR-1.1.1.6 Meterkast	SR-1.1.1.7 Meterkast breedte	SR-1.1.1.8 Meterkast lengte	SR-1.1.1.9 Meterkast hoogte	SR-1.1.1.10 Trap breedte	SR-1.1.1.11 Entreematje	SR-1.1.1.12 Begane grond	SR-1.1.1.13 Bergruimte binnen	SR-1.1.1.14 Eerste verdieping	SR-1.1.1.15 Geclusterde techr	SR-1.1.1.16 Verkeersruimte bree	SR-1.1.1.17 Garderobe ruimte in	SR-1.1.1.18 Garderoberuimte	SR-1.1.1.19 Vrije ruimte gardero	SR-1.1.2 Keuken	SR-1.1.3 Badkamer	SR-1.1.4 Woonkamer	SR-1.1.5 Slaapkamer	SR-1.1.6 Constructie	SR-1.2 Warmtedrager	SR-1.3 Verschijning	SR-1.5 Financieel	SR-1.6 Keus & optioneel	SR-1.7 Interfaces	SR-1.8 Energie eis
3 Badkamer structuur																								24						5	57		
3 Bouwconstructie structuur																											6		3	105	101		
3 Halkern structuur			2	2	3		3	3	3	3	3	3	3	2	3	3			5	2													
1 Interfaces																																	
3 Components					2		2	2	2	2	2	2	2		2	2			4	1													
Leiding ruimte BG																														40			
Meterkast									2	2	2	2								1										5			
Techniek ruimte																		6												10			
Toilet ruimte					2		2	2																						5			
Trap													2		2																		
Interfaces																																	
Trap BG													⇢		⇢															5			
Aantal : Real = 1.0															↗																		
BTrapBG : length[millimetre] = 1335.0													↗																				
HTrapBG : length[millimetre]																																	
LTrapBG : length[millimetre] = 2390.0																																	
Trapkast																2																	
Trapkast																⇢														5			
Aantal : Real = 1.0																																	
Breedte : length[millimetre]																																	
Lengte : length[millimetre] = 2120.0																																	
OppTrapkast : area[square metre]																↗																	
VerkeersruimteBG																			4														
Interfaces																																	
VerkeersruimteBG																			⇢											5			
Aantal : Integer = 1																																	
BVerkeersBG1 : length[millimetre]																			↗														
BVerkeersBG2 : length[millimetre]																			↗														
BVerkeersBG3 : length[millimetre]																			↗														
HVerkeersBG : length[millimetre]																																	
LVerkeersBG : length[millimetre]																																	
LVerkeersBG2 : length[millimetre]																																	
LVerkeersBG3 : length[millimetre]																																	
HalkernBG			⇢	⇢	⇢		⇢	⇢	⇢	⇢	⇢	⇢	⇢	⇢	⇢	⇢		5	⇢	⇢										10	3		
Aantal : Real = 1.0			↗																														
B1Hal : length[millimetre] = 2645.0																																	
B2Hal : length[millimetre] = 1347.0																																	
Entreematje : Boolean														↗																			
HHal : length[millimetre]																																	
L1Hal : length[millimetre] = 4144.0																																	
L2Hal : length[millimetre] = 5200.0																																	
Verdieping = 0.0				↗																													
3 HalkernV1 structuur			2			2							3				2	11	2											65			
3 Hoofdslaapkamer structuur																										13				15	7		
3 Keuken structuur																							10				1			30	28		
3 Slaapkamer 2 structuur																										11				5	3		
3 Slaapkamer 3 structuur																										10				5	3		
3 Woning structuur	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	1	1			5	10	4	14	16	4	3	5	62		
3 Woonkamer structuur																									8					5	14		
3 Zolder structuur																											5	12		10(	31		

Figure 61 – Satisfaction matrix linking system structure and parameters with requirements

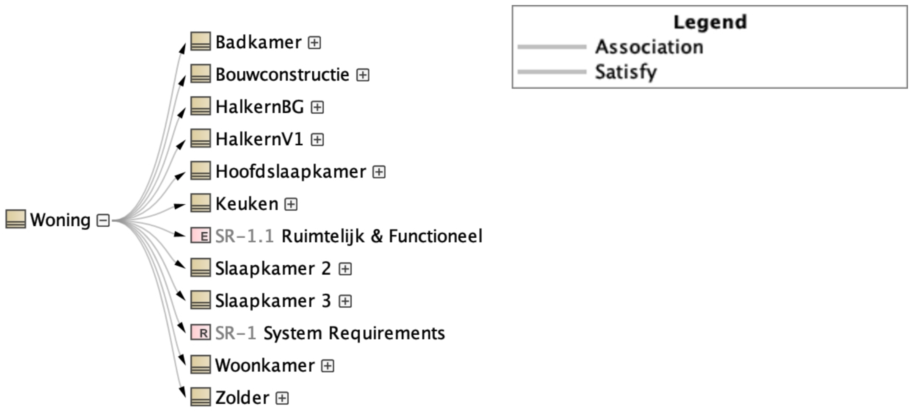


Figure 62 – Connections between structure and requirements

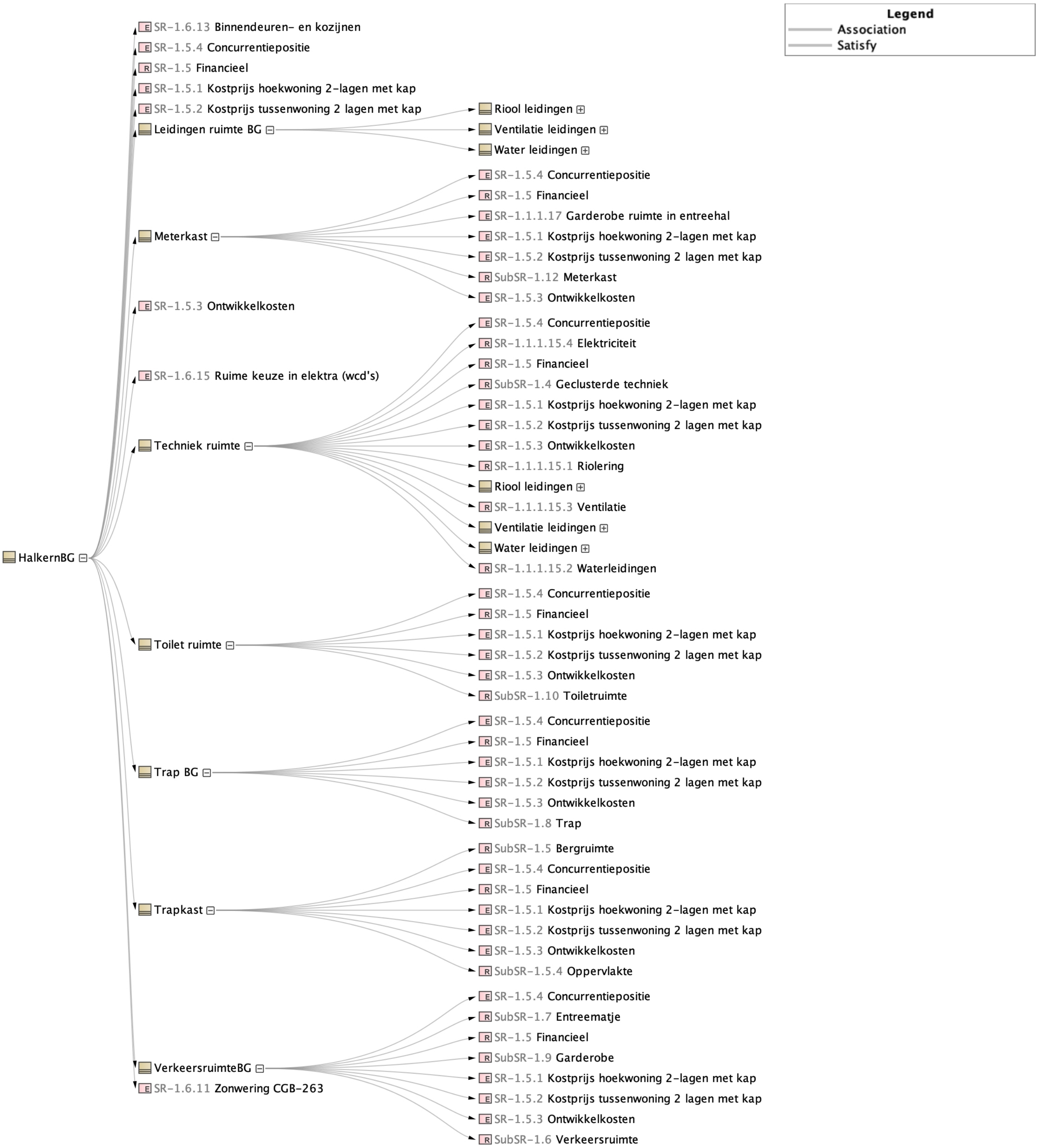


Figure 63 – Connections between hallway ground floor to requirements

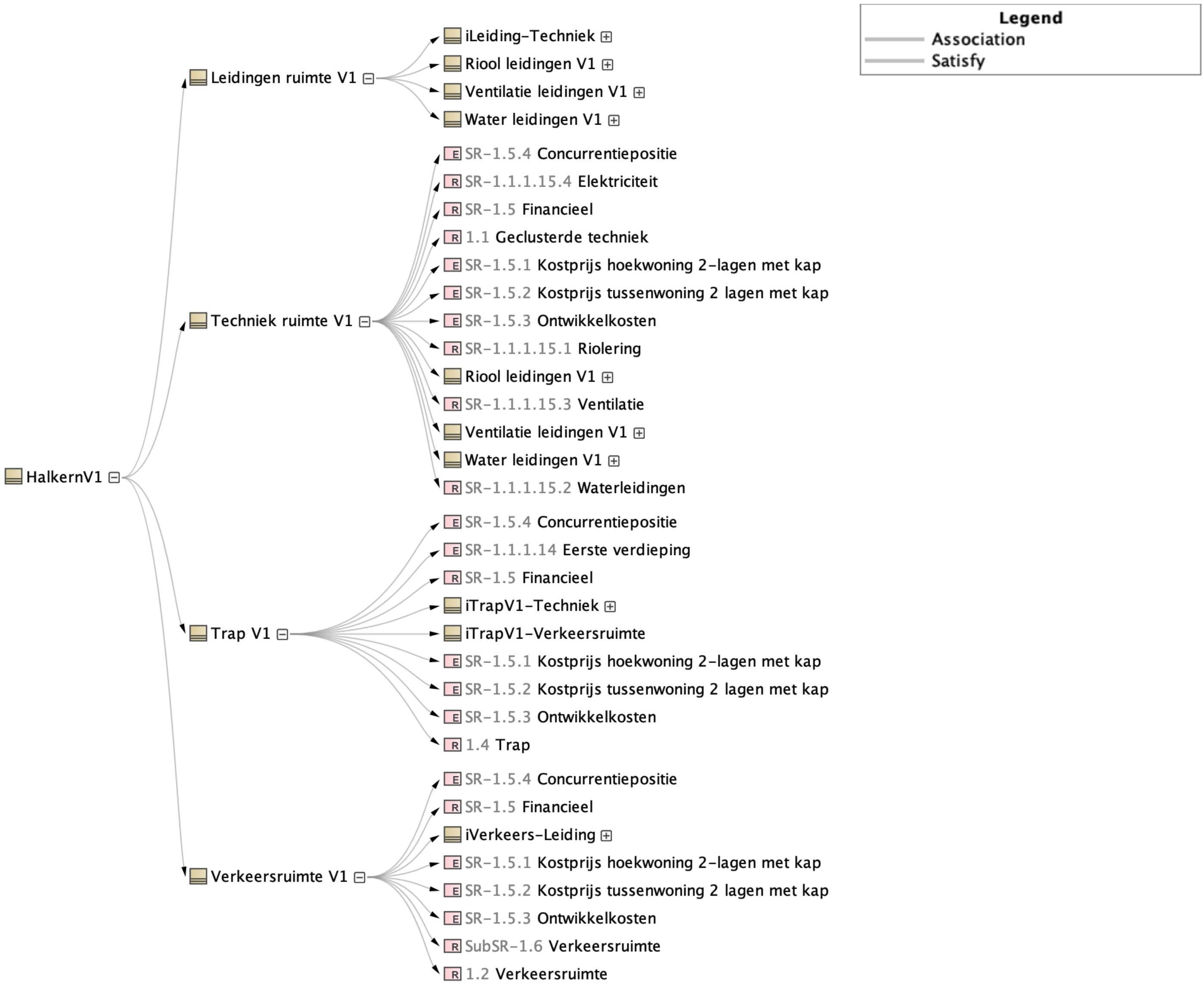


Figure 64 – Connections between hallway first floor to requirements

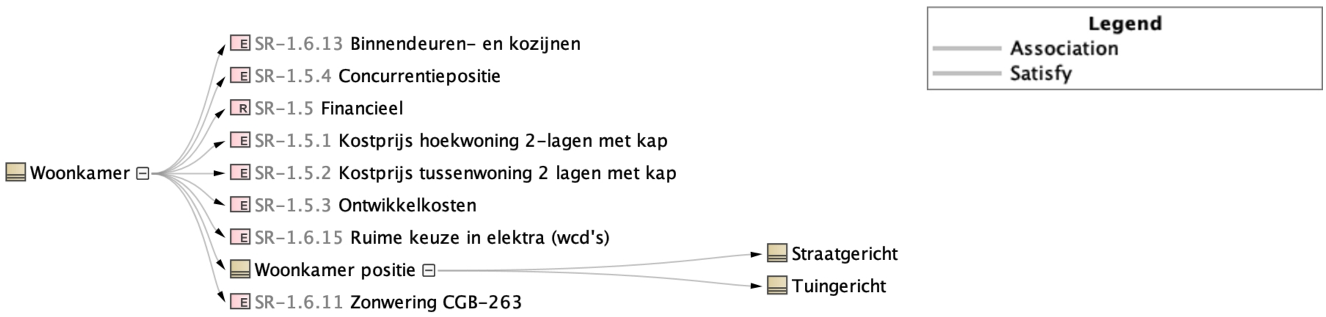


Figure 65 – Connection between living room and requirements

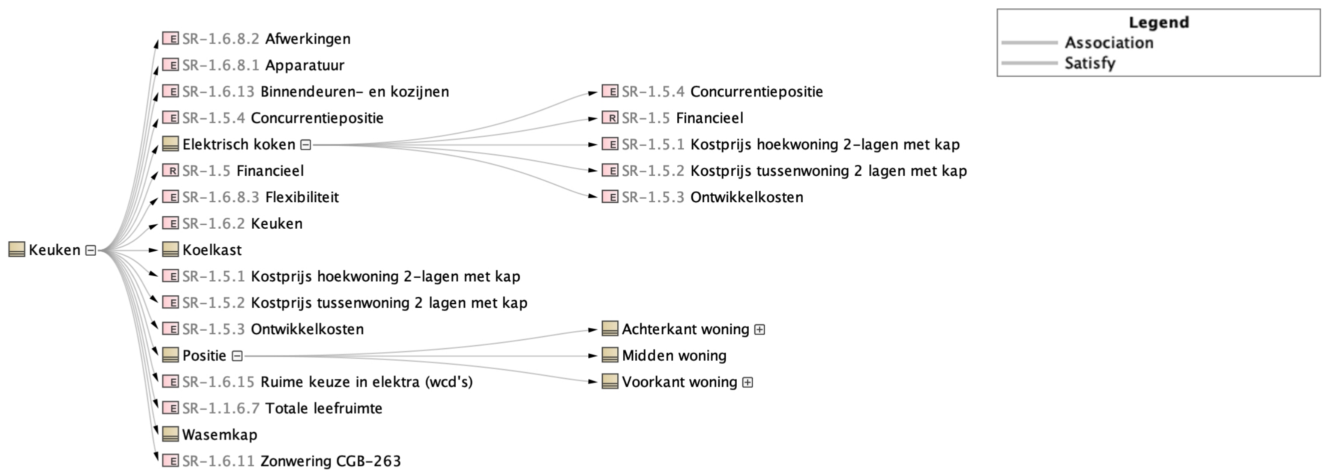


Figure 66 – Connection between kitchen and requirements

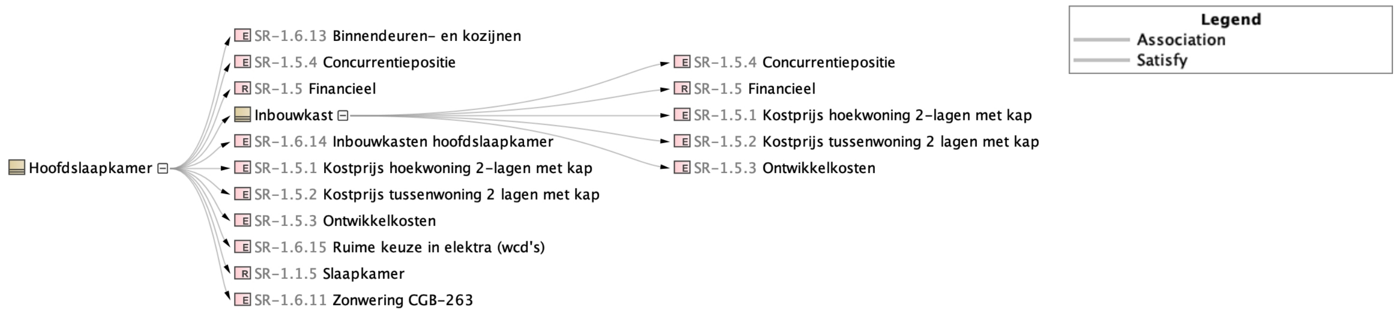


Figure 67 – Connection between the master bedroom and requirements

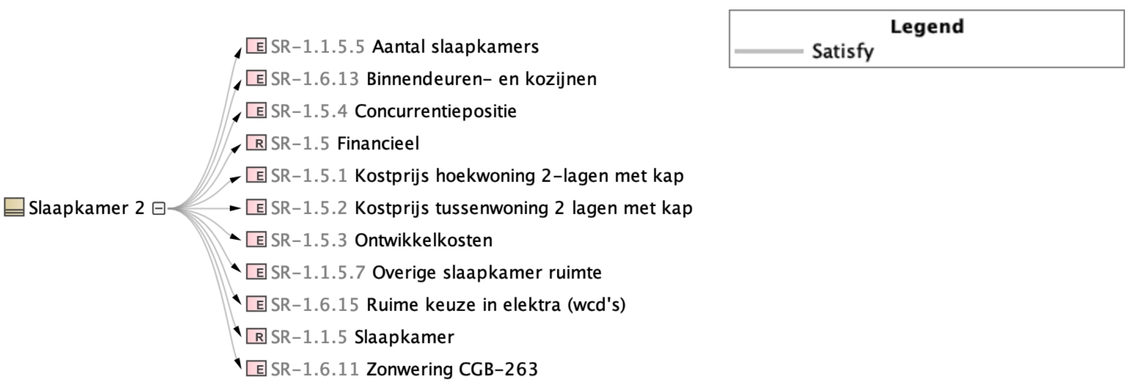


Figure 68 – Connection between bedroom 2 and requirements

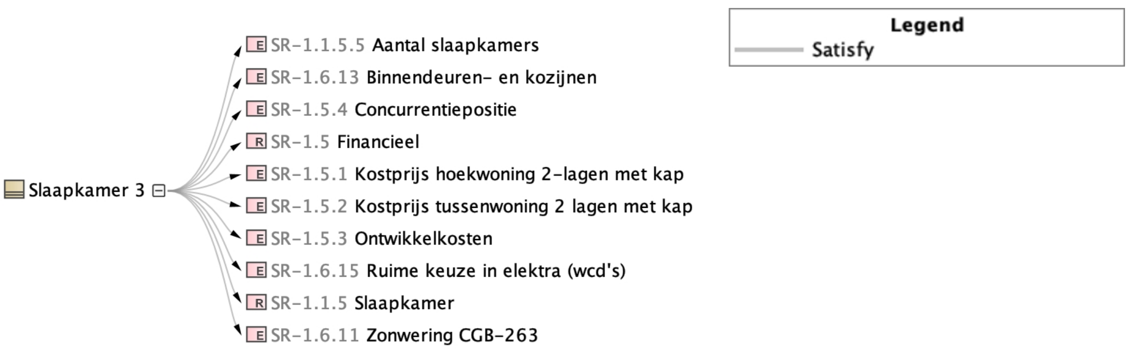


Figure 69 – Connection between bedroom 3 and requirements

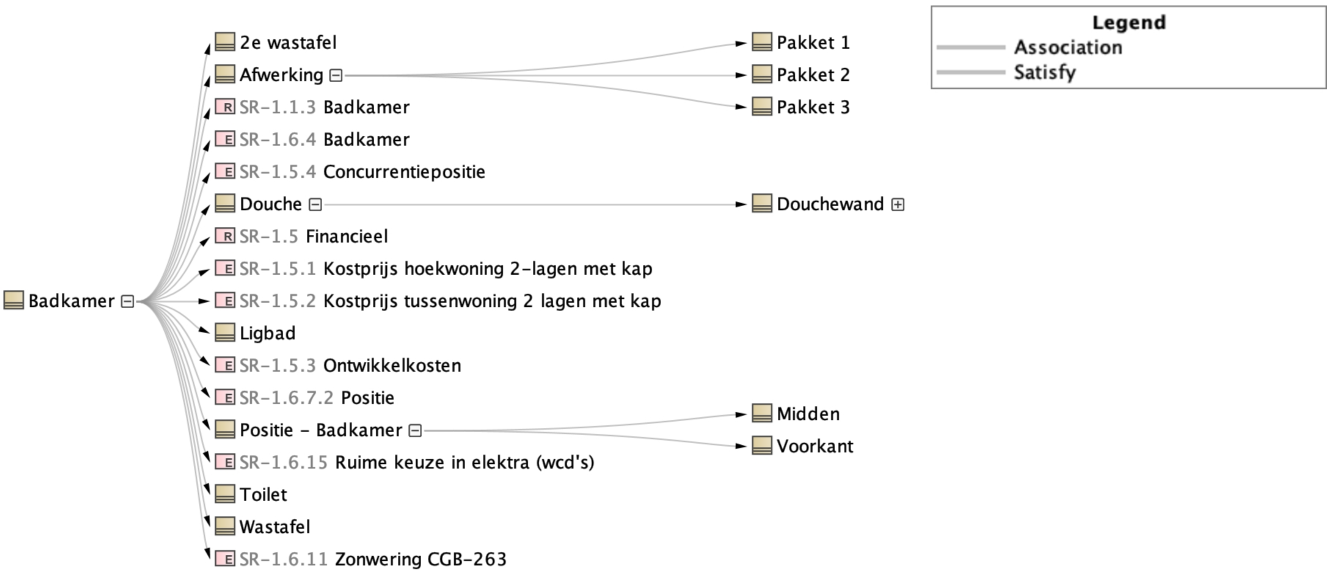


Figure 70 – Connection between bathroom and requirements

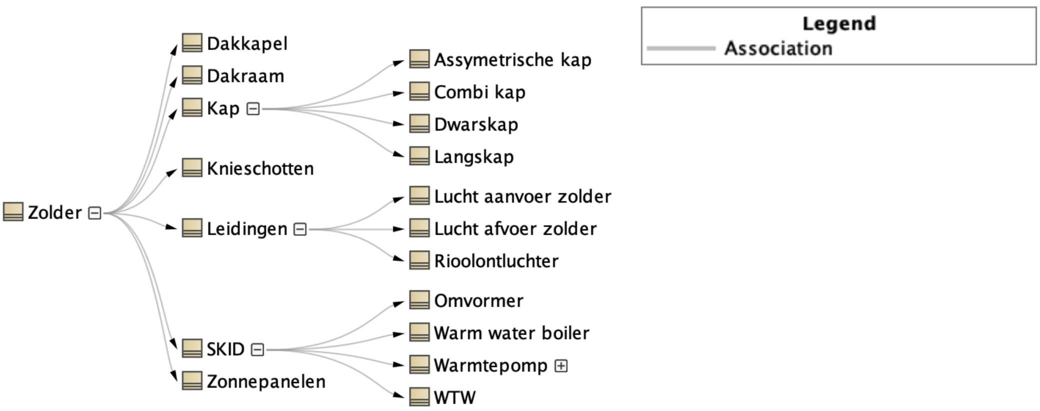


Figure 71 – Connection between attic to requirements

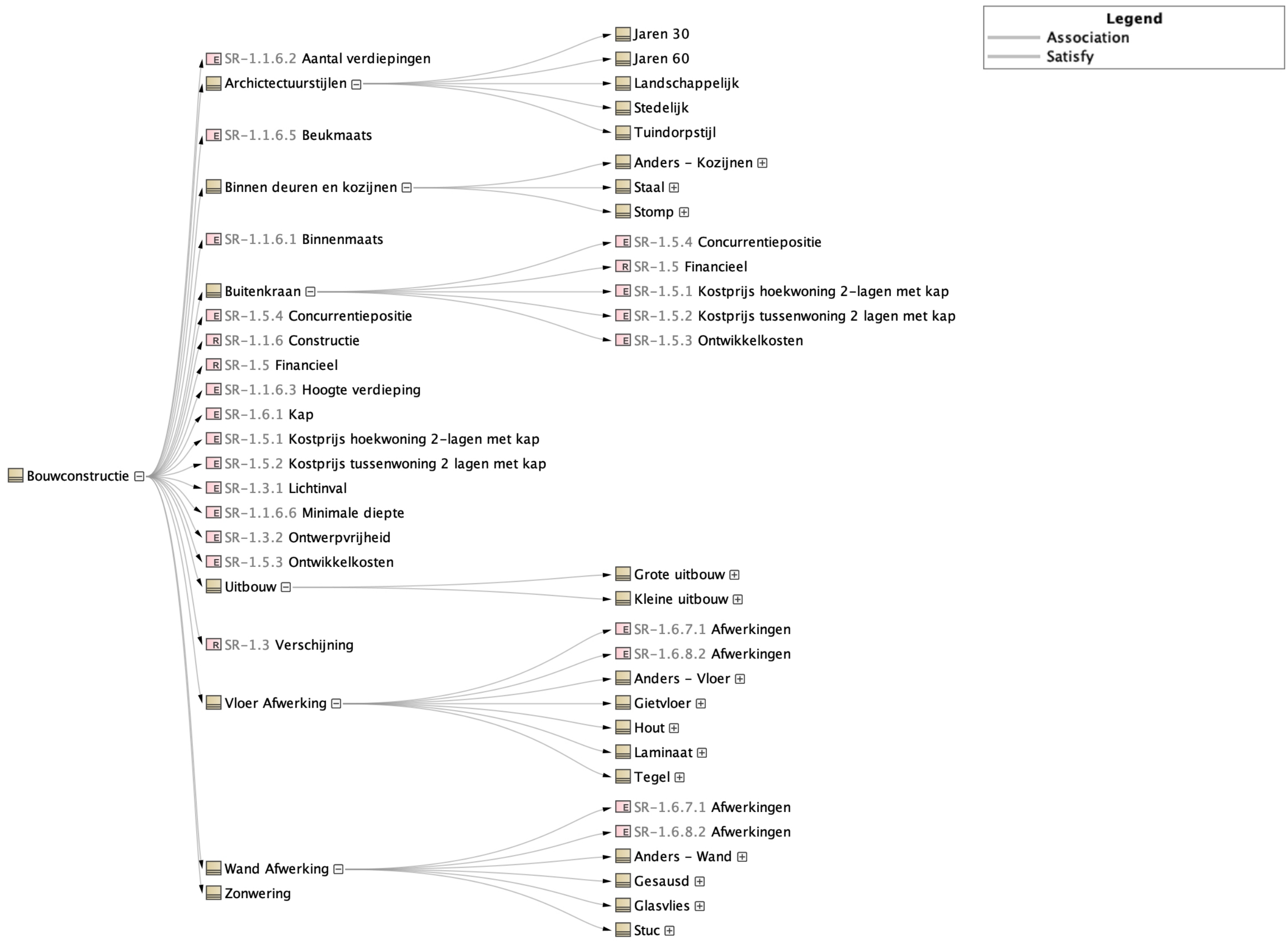


Figure 72 – Connection between the construction structure and the requirements

G.10 External interfaces between subsystems

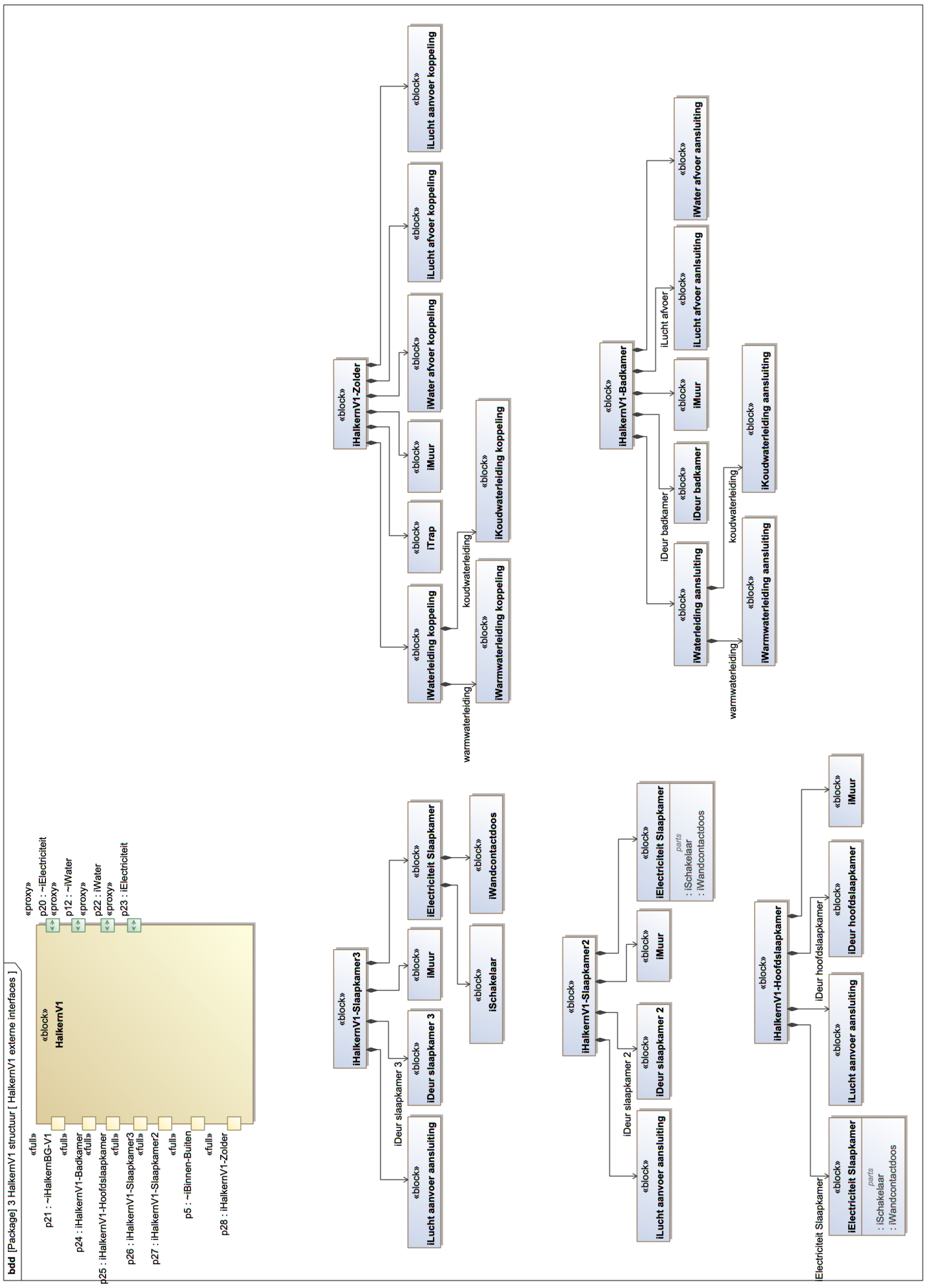


Figure 73 – Interfaces of the hallway module first floor with other subsystems

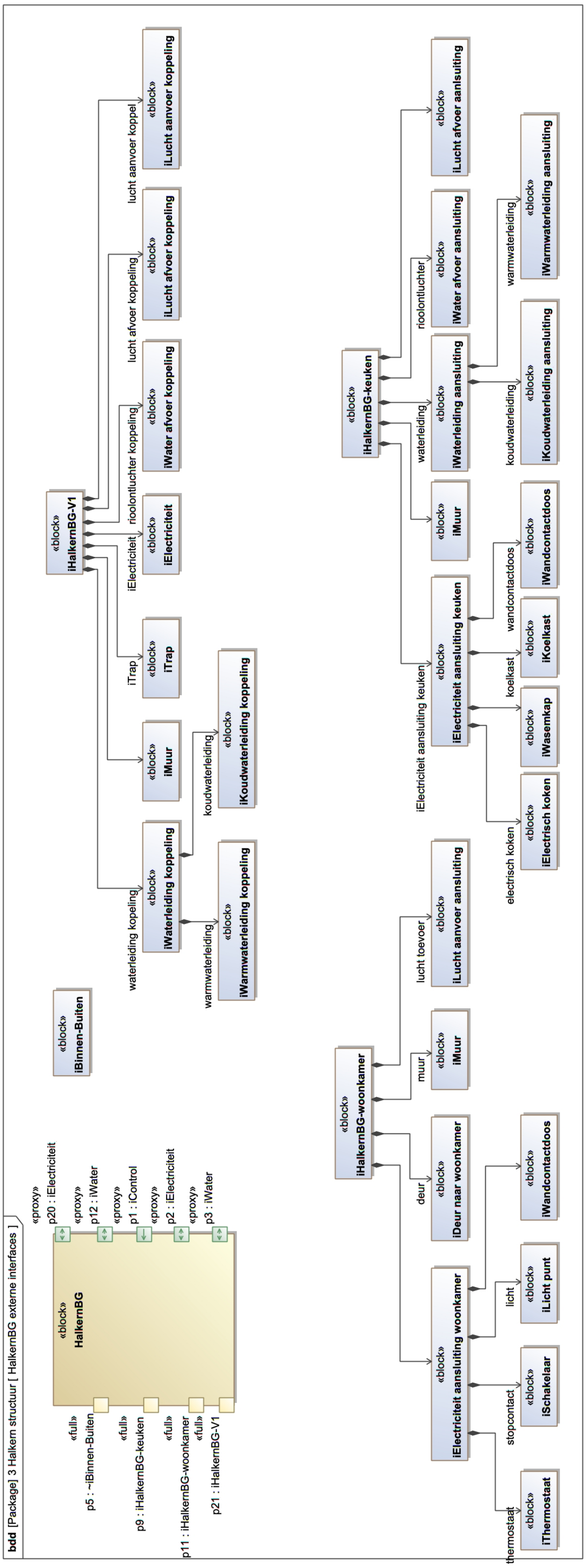


Figure 74 – Interfaces of the hallway module ground floor with other subsystems

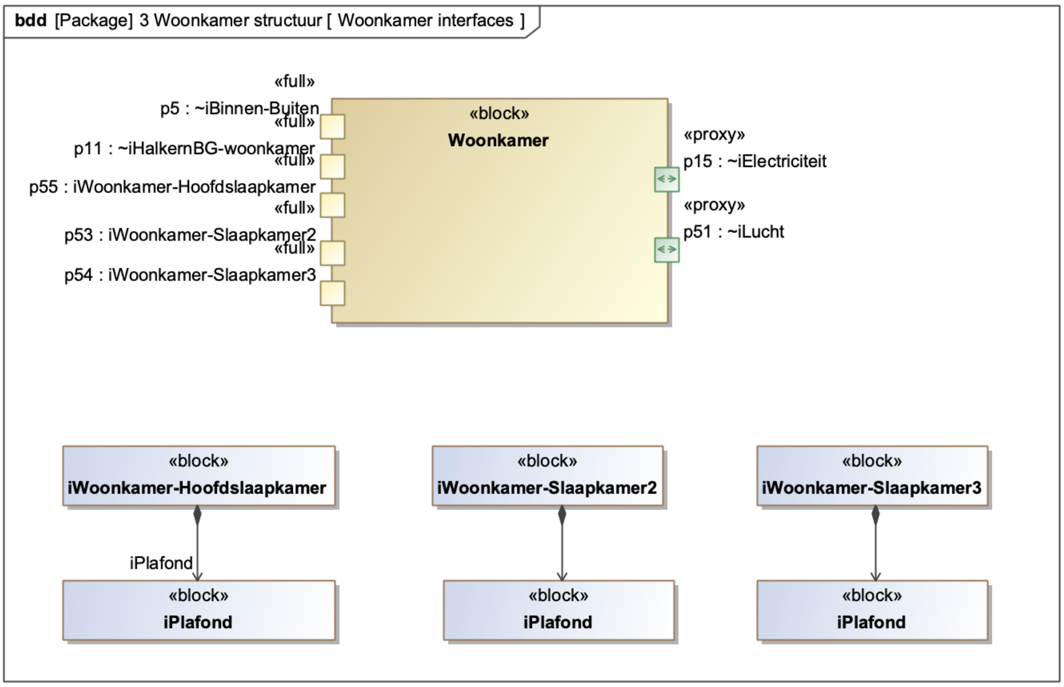


Figure 75 – Interfaces of the hallway module first floor with other subsystems

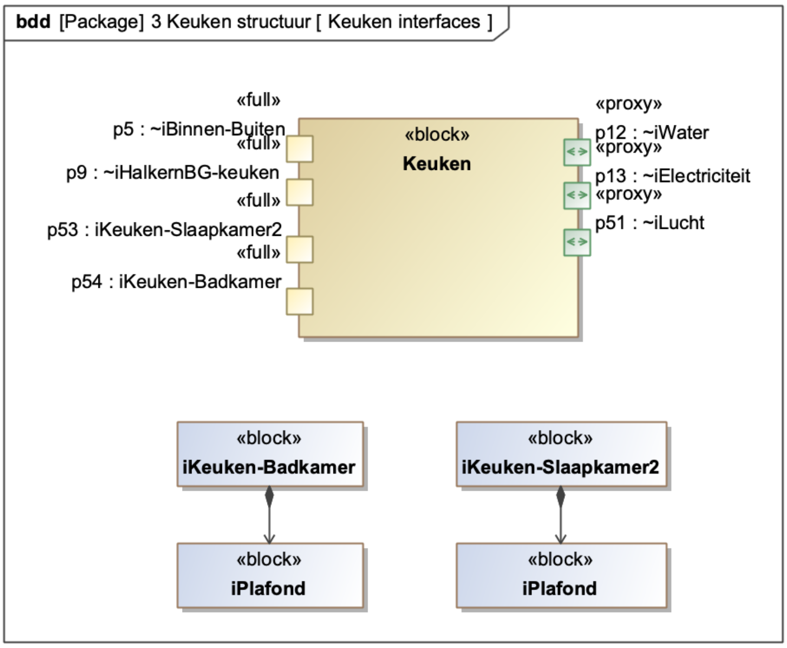


Figure 76 – Interfaces of the hallway module first floor with other subsystems

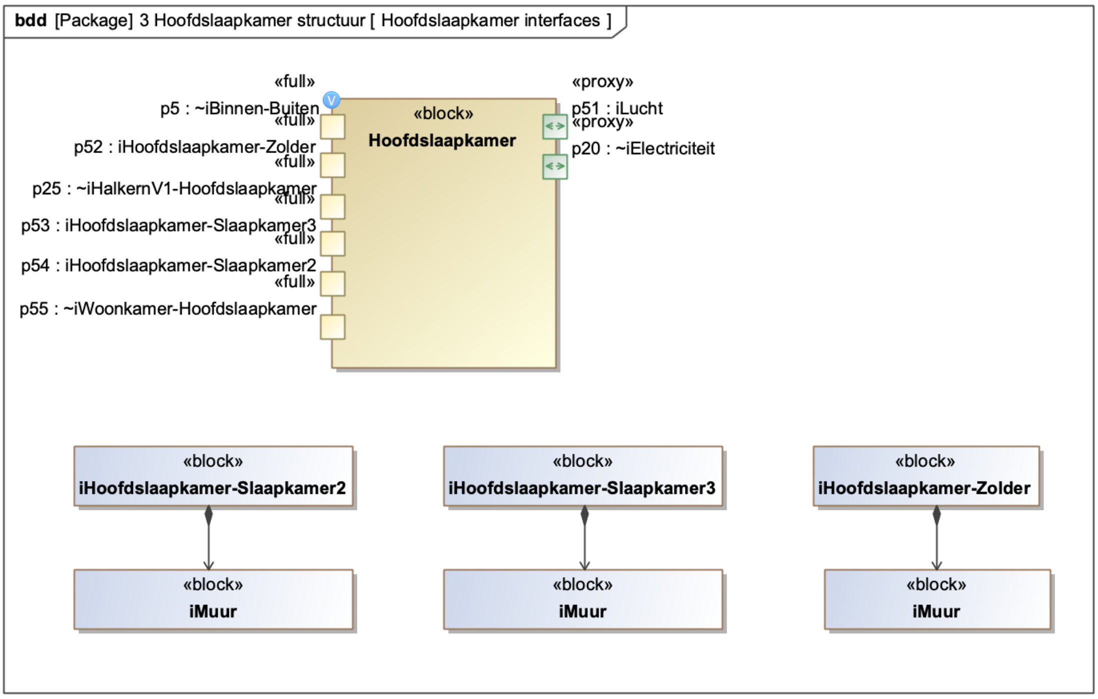


Figure 77 – Interfaces of the master bedroom with other subsystems

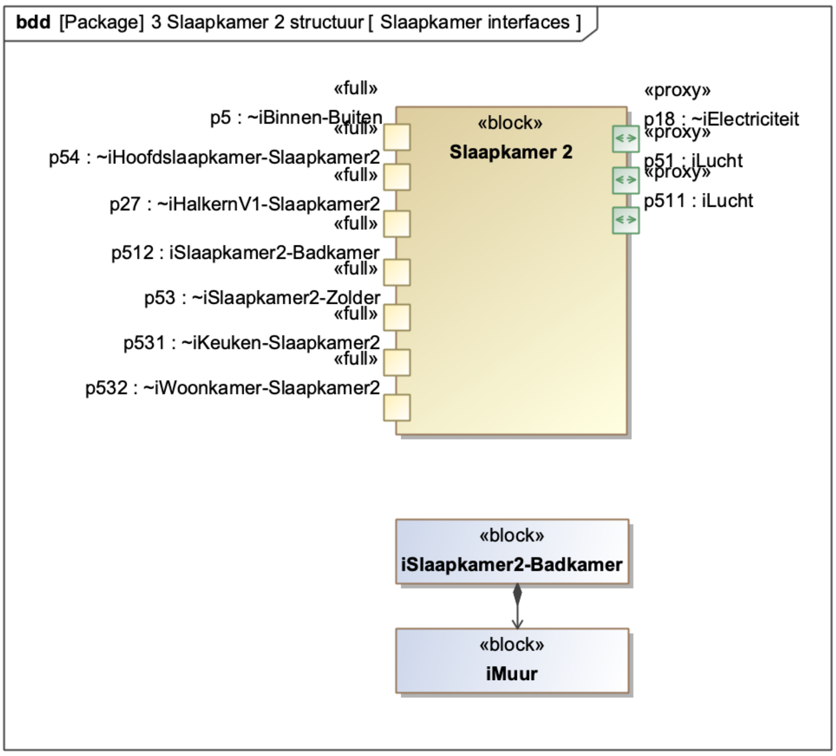


Figure 78 – Interfaces of the bedroom 2 with other subsystems

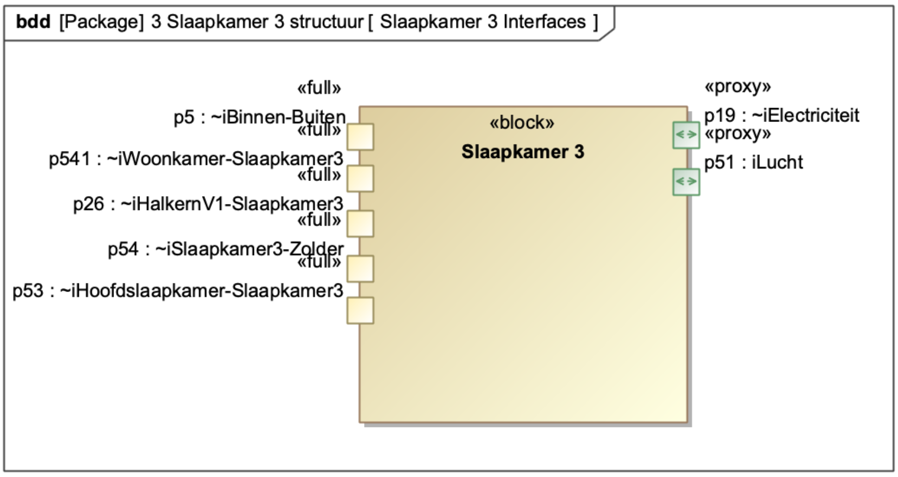


Figure 79 – Interfaces of the bedroom 3 with other subsystems

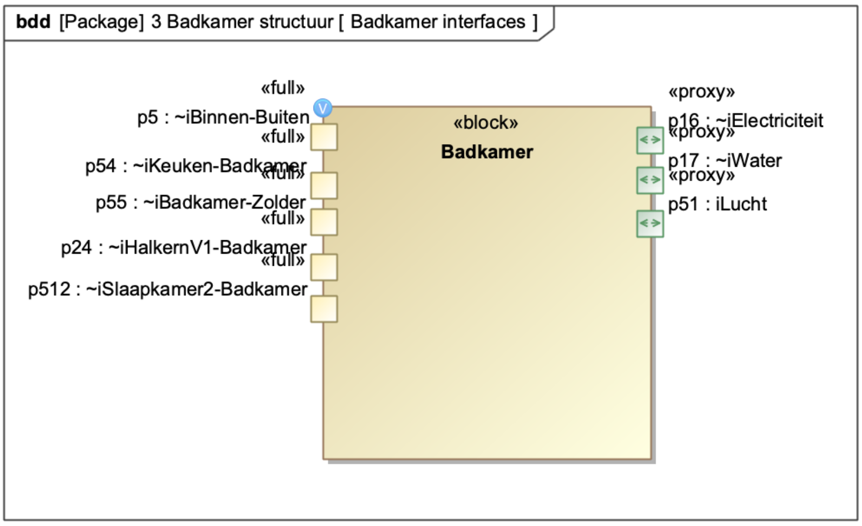


Figure 80 – Interfaces of the hallway bathroom with other subsystems

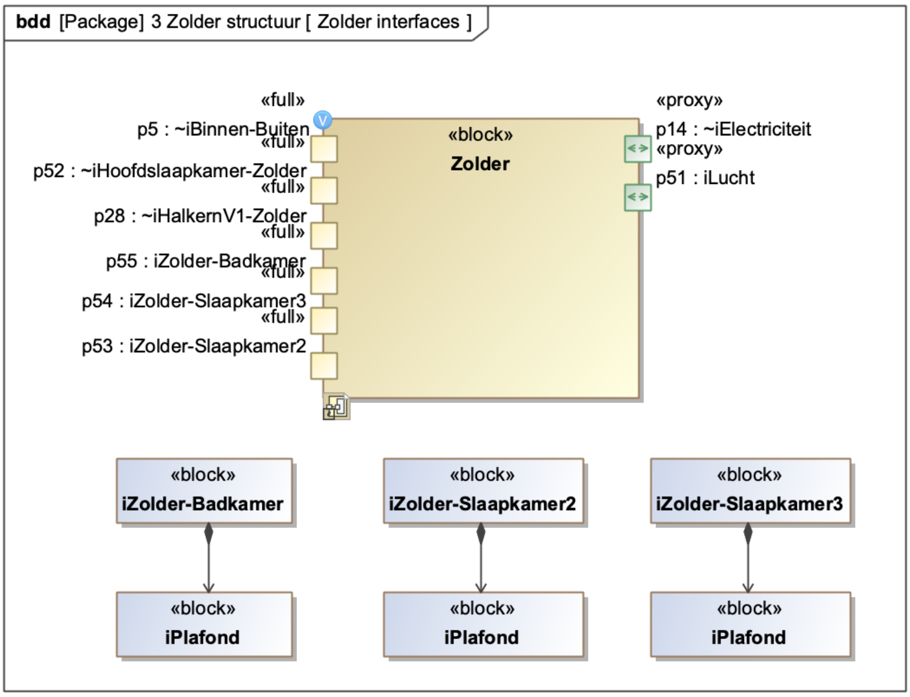


Figure 81 – Interfaces of the attic with other subsystems

G.11 Variant points, variant modules and variant configuration of the bathroom module

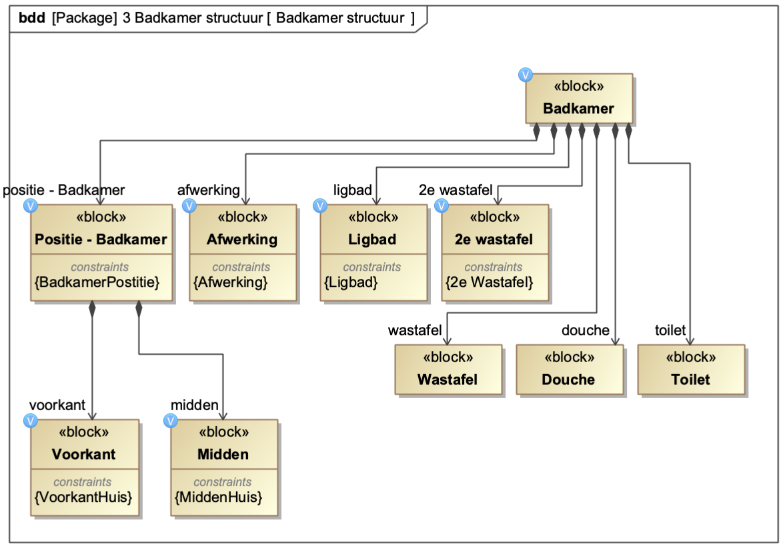


Figure 82 – Bathroom structure with variant points and variant modules

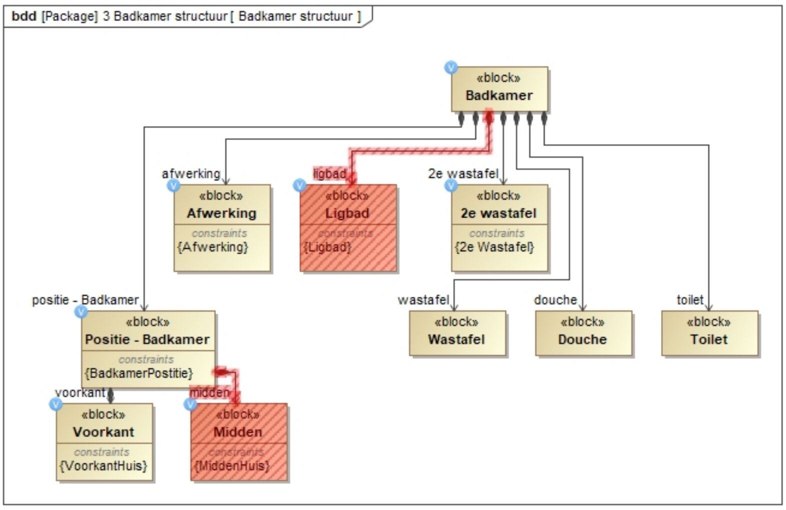


Figure 83 – Variant selection of the bathroom

G.12 Independent parameters in the system design

Parameter	Value	Parameter	Value
Inner width of the house	5100 mm	Floor height	2900 mm
Inner depth of the house	8720 mm	Number of floors	3
Wall thickness	95 mm	Subsystem floor number	0 or 1
Hallway GF length(s)	5200 mm 4144 mm	Hallway F1 length	2390 mm
Hallway GF width(s)	2645 mm 1347 mm	Hallway F1 width	2405 mm
Toilet length	1300 mm	Master bedroom length	4810 mm
Toilet width	900 mm	SKID height	
Stairways length	2390 mm	SKID Length	1500 mm
Stairways width	1335 mm	SKID width	1000 mm
Stair cupboard length	2120 mm	Pipes space height	246 mm
Technical installation space length GF	650 mm	Technical installation space length F1	650 mm
Technical installation space width GF	450 mm	Technical installation space width F1	450 mm
Fuse box length	429 mm		

Table 33 – Independent parameters in the construction design