PLM support to architecture based development

Contribution to computer-supported architecture modelling

Bruun, Hans Peter Lomholt; Mortensen, Niels Henrik; Hvam, Lars

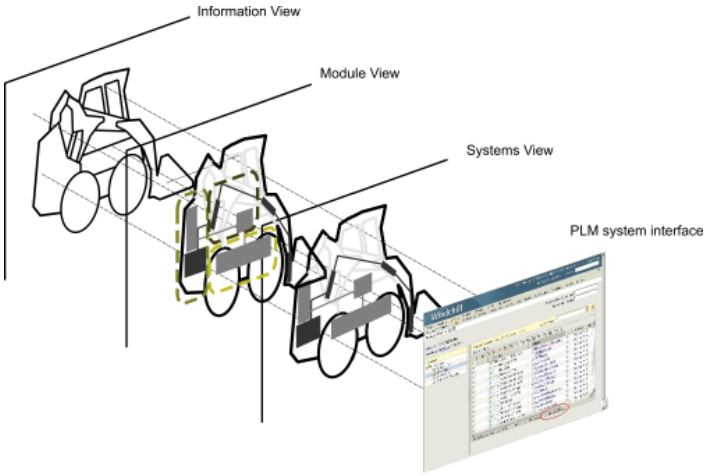
Publication date:
2015

Document Version
Publisher's PDF, also known as Version of record

PLM support to architecture based development

Contribution to computer-supported architecture modelling

PhD Thesis



Hans Peter Lomholt Bruun
DCAMM Special Report No. S174
January 2015

DTU Mechanical Engineering
Department of Mechanical Engineering

Thesis for the degree of Doctor of Philosophy

PLM support to architecture based development

Contribution to computer-supported architecture modelling

Hans Peter Lomholt Bruun

January 2015

ISBN nr.: 978-87-7475-397-1
DCAMM Special Report: S174

DTU Mechanical Engineering
Technical University of Denmark
Section of Engineering Design and Product Development
Produktionstorvet
Building 426
DK-2800 Kgs. Lyngby
Denmark
info@mek.dtu.dk
phd@mek.dtu.dk
Tel. (+45) 4525 1960
VAT 30060946

Supervisors

Main supervisor

Niels Henrik Mortensen, Professor, PhD
Section of Engineering Design and Product Development
Department of Mechanical Engineering
Technical University of Denmark

Co-supervisor

Lars Hvam, Professor, PhD
Section of Operations Management
Department of Management Engineering
Technical University of Denmark

Opponents

Torgeir Welo, Professor, PhD
Department of Engineering Design and Materials
Norwegian University of Science and Technology
Norway

Johan Malmquist, Professor, PhD
Dean of education MATS
Division of Product and Production Development
Chalmers University of Technology
Sweden

Tommy Bysted, PhD
R&D Director
Radiometer Medical
Copenhagen, Denmark

Abstract

Designers doing product architecture based development look to convert desired behaviour to solutions for a portfolio of products, and through modularisation pursue commonality among different variants without increasing the internal task proportional to handling variety. To develop product architectures for a portfolio of products that support the right balance between commonality and variety is today a foremost part of most large companies' development operations. A challenge is that product architectures are influencing external and internal performance of markets, production, technology, organisation, processes, etc. To identify, evaluate, and align aspects of these domains are necessary for developing the optimal layout of product architectures.

It is stated in this thesis that architectures describe building principles for products, product families, and product programs, where this project focuses on architecture's ability to describe product families. Architectures are developed with different objectives in mind, i.e. to obtain a certain effect for a company, such as reducing time-to-market, reducing product cost, increasing R&D efficiency, etc. Visual models with cross functional languages are, in architecture design, seen as key means for supporting designers from different domains and with different backgrounds, in accessing the structures of architectures and their behavioural effects.

This PhD project focuses on prescribing how to model structural elements and address behavioural effects in graphical modelling formalisms of architectures. The objective of using the product architecture formalisms is to support designers in identifying, evaluating, and optimising the architecture satisfying the goals of the company in the best way in the view of the resource constraints. This thesis is particularly focusing on one product architecture modelling formalism - *The Interface diagram*. The formalism has an objective of supporting interdisciplinary designers in developing a product architecture for a product family. However, the large amount of information generated when identifying and developing architectures can be difficult to manage, update, and maintain during development. The concept of representing product architectures in computer-based product information tools has though been central in this research, and in the creation of results. A standard PLM tool (*Windchill PDMLink©*) is applied for representing a model of a product architecture, and for enabling fast, precise, and safe data transfer, as well as reducing the effort to replicate and modify information.

This PhD thesis describes research into the phenomena of developing products based on architectures and how to represent architectures in computer systems. Presented results build on research literature and experiences from industrial partners. Verification of the theory contributions, approaches, models, and tools, have been carried out in industrial projects, with promising results. This thesis describes the means for: (1) *Identifying and modelling architectures*, (2) *Multi-viewpoint modelling* for supporting reasoning in converting desired product behaviour (given by requirements and/or functions) into a solution (given by components), (3) *Assessing product cost and cost deviations during design,* and (4) *Assessing completeness of designs during development*.

Keywords: Product lifecycle management, product architecture, product modelling, computer-based support.

Resumé

Produktudviklere, som udvikler produkter baseret på arkitekturer, forsøger til stadighed at konvertere den ønskede funktionelle performance til løsninger for en portefølje af produkter. For ikke at øge den interne håndteringsopgave proportionalt med antallet af varianter, forsøger udviklere at opnå ensartethed mellem produktvarianter ved hjælp af modularisering. At udvikle produktarkitekturer for en produktportefølje, som understøtter den rette balance mellem ensartethed og variation, er således i dag en væsentlig del af store virksomheders udviklingsopgave. Kompleksiteten består i at produktarkitekturer influeres af og påvirker, multiple aspekter af marked, produktion, teknologi, organisation, processer m.fl. Disse sammenhænge må kunne identificeres og belyses for at kunne afstemme produktarkitekturers optimale udlægning.

I denne afhandling fastslås det, at arkitekturer beskriver byggeprincipper for produkter, produktfamilier og produktprogrammer. Dette Ph.d.-projekt fokuserer imidlertid på arkitekturer som beskriver produktfamilier. Arkitekturer udvikles med forskelige formål, dvs. for at opnå effekter for en virksomhed som f.eks. at reducere udviklingstid, reducere produktomkostninger, optimere brugen af udviklingsressourcer m.fl. Visuelle produktmodeller med tværfunktionelle modelleringssprog ses i arkitekturdesign som vigtige værktøjer til at støtte udviklere fra forskellige ingeniørdiscipliner - og med forskellig baggrund- i at få både adgang til og fælles forståelse for arkitekturers opbygningen og de effekter som de disponerer.

Ved hjælp af nye modelleringsformalismer og forskellige produktperspektiver som basis for modellering, fokuserer dette projekt på både funktionelle og strukturelle aspekter af arkitekturer. Formålet med at benytte arkitekturformalismerne i udviklingsprocessen er at yde støtte til udviklere, som forsøger at identificere, evaluere og optimere arkitekturer, som opfylder målsætninger for virksomheder på den bedst mulige måde med de for den enkelte virksomheds tilgængelige ressourcer. Denne afhandling fokuserer på præsentationen af specielt én modelleringsformalisme - *The Interface diagram*. Denne modelleringsformalisme har som formål at støtte tværfunktionelle udviklere i at designe en arkitektur for en produktfamilie, således at funktionelle krav kan adresseres i en optimeret modulær opbygning. Den store mængde information, som genereres når arkitekturer identificeres og udvikles, kan dog være besværlig at kontrollere, opdatere og vedligeholde. At kunne repræsentere og håndtere beskrivelser af arkitekturer i computerbaserede systemer, har derfor været central i denne forskning. Et standard PLM-system (*Windchill PDMLink©*) benyttes i dette projekt til at repræsentere arkitekturer og dermed muliggøre en hurtig, præcis og sikker dataoverførsel, samt til at reducere opgaven med at skabe og ændre information.

Resultaterne i denne afhandling bygger både på forskningslitteratur og samarbejde med industrielle partnere. Verificering af teoribidrag, metoder, modeller og værktøjer er udført i industriprojekter med lovende resultater. Denne afhandling beskriver midler til at (1) identificere og modellere arkitekturer, (2) modellere multiple produktperspektiver for at understøtte designere i at konvertere ønsket produktadfærd (fastslået i krav og/eller funktioner) til løsninger (bestående af komponenter), (3) vurdere direkte produktomkostninger samt afvigelser mellem det faktiske og planlagte, samt (4) til at vurdere et designs færdiggørelse under udvikling.

Nøgleord: Product lifecycle management, produktarkitektur, produktmodellering, computerbaseret support.

Preface

This PhD thesis documents a research project carried out at the Technical University of Denmark (DTU). The project has been carried out in the Department of Mechanical Engineering at DTU and co-financed by The Real Dania Foundation.

The project was initiated in January 2011 and ended June 2014. The project had a total duration of 36 months. The project had been interrupted for 4 months, while respectively, consultancy was carried out and paternity leave was granted.

During the project and thesis, I have had contact with many persons that, in different ways, have contributed to my work. First of all I would like to express my gratitude to my main supervisor, Professor Niels Henrik Mortensen, to whom I am grateful for the guidance and enthusiastic support. I also owe gratitude to my co-supervisor, Professor Lars Hvam, for helping me focus my work. Thank you also to Professor Emeritus Mogens Myrup Andreasen, for the stimulating discussions concerning my thesis development and for the interesting discussions about engineering design and product development in general.

Thanks also to my colleagues at The Section of Engineering Design and Product Development at the Technical University of Denmark, for their interest and support. Especially, I remember the work within the Architecture Group and my international network of researchers, who made a difference. I wish to thank each and every one of you for the many constructive discussions on my research topic and for making it fun and joyful to show up for another day of research. I am also grateful for the practical insight PhD and CEO Ulf Harlou, has provided me with and for the valuable discussions on application of developed methods and tools. To those who in any way have contributed to the PhD, but have not been mentioned above, consider this as my sincere thanks for your contribution, help, and support.

The financing from Realdania and the Technical University of Denmark was a precondition for the project, and it is thankfully acknowledged, as well as the fruitful collaboration with the personnel of the industrial partners.

Last but not least, I send my thankful thoughts to my family, and unconditionally my girlfriend Louise, who has never complained when I have been away, physically as well as mentally. And to you Edith and Martha, our children, for the joy and daily smiles that put energy into my mind when work was increasing.

Hans Peter Lomholt Bruun

Copenhagen, January 2015

List of appended publications

- **Paper A**

  Bruun, H.P.L., Mortensen, N.H., Harlou, U., ***Interface diagram: Design tool for supporting the development of modularity in complex product systems,*** Journal of Concurrent Engineering, Research and Applications, published online 29 December 2013.

- **Paper B**

  Hansen, C.L., Bruun, H.P.L., Mortensen, N.H., Hvam, L., ***Identification of a scalable architecture for customization of complex parts***, Journal of Concurrent Engineering, Research and Applications, In 1st review 2014.

- **Paper C**

  Bruun, H.P.L., Mortensen, N.H., Harlou, U., ***PLM support for development of modular product families,*** Conference proceedings of ICED 2013, Seoul, South Korea.

- **Paper D**

  Bruun, H.P.L., Mortensen, N.H., Harlou, U., Wörösch, M., Proschowsky, M., ***PLM system support for modular product development,*** Journal of Computers in Industry, In 2nd review 2014.

- **Paper E**

  Bruun, H.P.L., Hauksdóttir, D., Harlou, U., Mortensen, N.H., ***Mapping requirements to a product architecture supported by a PLM system***, Conference proceedings of Design Conference 2014, Dubrovnik, Croatia.

Additional publications

The following publications cover the initial results of this research. They are not appended as their results are covered in the appended Papers A, C, and D, which include a more thorough description of their contributions.

- Bruun, H.P.L., Mortensen, N.H.*,* ***Modelling and using product architectures in mechatronic product development***, Conference proceedings of NordDesign 2012, Aalborg, Denmark.
- Bruun, H.P.L., Mortensen, N.H., ***Visual product architecture modelling for structuring data in a PLM system,*** Conference proceedings of PLM 2012, Montreal, Canada.

The following publications are related to the research presented in this thesis, although not making a central contribution to the results.

- Andreasen, M.M., Howard, T., Bruun, H.P.L., ***Domain Theory, its models and concepts,*** in book: *Anthology of Theories and Models of Design*, Chakrabarti, A., and Blessing, L. (editors), Springer Verlag 2014.
- Mortensen, N.H., Gamillscheg, B., Bruun, H.P.L., Hansen, C.L., Cleemann, K.K., Junkov, K.H., ***Radikal Forenkling via Design, in Danish,*** DTU Mechanical, 2012.
- Wörösch, M., Bruun, H.P.L., Howard, T.J., Mortensen, N.H., ***A proposed framework model for managing requirements in the construction industry,*** Journal of Construction Management and Economics, in 1st review 2014.

Table of Contents

1 Introduction to the research area

This PhD thesis describes a research project within the topic of product architecture based development and Product Lifecycle Management (PLM) system support for representing architectures. The first part of this thesis introduces: The problem area for the research, the encountered industrial challenges, the research objectives, and the research questions. The scope and boundaries of the content presented in this thesis are treated, and finally the structure of the thesis is illustrated and explained.

1.1 Background / Problem area

Manufacturing companies developing products are finding themselves in a persistent need of designing and manufacturing products that are successful in an always changing marketplace. The situation puts pressure on companies, requiring them to increase their R&D efficiency to improve time-to-market by providing new competitive solutions faster and at the same time cut costs. Most large companies' markets have become global which makes the need for customised local variants increase. In return, fluctuation of demands between the different variants makes production planning more difficult, as the production volume of each product variant decreases. Globalisation also means companies are challenged by stronger global competition, stressing the importance of getting the right balance between product performance and the cost of creating it. Consequently, companies face a dilemma between the need for a variety of products offered to the marketplace and a desire to reduce complexity of handling that variety within the company. Moreover, as products today often contain solutions based on multiple technologies, designing is an interdisciplinary activity between teams of specialised designers. As a consequence of globalisation issues and the diversity of involved engineering disciplines, the complexity of the development process has increased in the last few decades.

In order to stay competitive, companies strive to optimise their organisations, processes, products, production, services, and markets, by different means. The focal area for companies' efforts to stay competitive is product development, as improvements in the product development area have dispositional effects on the performance of the market area, the manufacturing area and many other functional areas of a business, Olesen *(1992)*, Harlou *(2006)*. There seems to have been a general transformation in the industry in the last two decades from developing products in independent projects to the development of product families. *Architecture based product development* is one initiative in the product development area to address this changed way of developing products, which focuses on the design of product ranges instead of individual products, and by reuse of knowledge, components, processes, and utilisation of economies of scale in many of the activities that are necessary to provide products for customers. Architecture based product development can be considered as a means to solving the conflicting task of providing variety to the marketplace while seeking to reduce complexity among internal company operations, in order to achieve an attractive cost level of products, Hansen, Mortensen & Hvam *(2012)*.

It is stated in this thesis that an architecture describes a structure of a product, product family, or product program, and describes how they are partitioned into subsystems and/or modules, and components, and

how functionality is allocated to them. In other words, a representation of an architecture should describe the building principle of a product system by its structural characteristics and behavioural effects. During development when design characteristics are determined, structure can be derived from the elements and relations, i.e. what the system is. The problem is that behaviour, i.e. what a system does, is realised by means of the designed structure (function is stated to be the purposeful part of behaviour). The challenge is to take behaviour into account when synthesising structures in an architecture model. Evaluating the performance of an architecture is considered a challenging task of looking both at relations between structure and behaviour and the dispositions the architecture creates, i.e. the effects on the surrounding systems of production, distribution, use, etc. Part of this challenge is how to establish key functional performance in different product variants, subsystems and/or modules, at the right cost level.

The application of modelling techniques of architectures is in academia described as a means for designers from different disciplines to intervene in the process of identifying and developing architectures that satisfy goals of the company. Common for models of architectures today, is that they are made up of large numbers of information elements, making their documentation a complex information management task. Consequently, inclusion of computer-support is considered a prerequisite for effective and efficient identification, development, and maintenance of architectures, Bergsjö, Malmqvist & Ström *(2006)*. During the last decades, the tools and methods applied in product development have undertaken a revolutionary change, due to computer-supported procedures in design and development activities. Computer-support can in a general be characterised as *Information technology (IT)*, which is used for storing, retrieving, and sending information in the context of an enterprise.

The increasing amount of data from multiple IT systems can however be hard to combine, as the large datasets generated by different systems tend to remain trapped in their respective environment because of their different format and information structure, Stark *(2007)*. The information is usually buried within various models and/or documents, and tends to be processed by different software tools that have no common 'language', Weber, Werner & Deubel *(2003)*. This makes communication between domains more difficult and increases the risk of making errors when transferring information among IT systems and people. Consequently, there seems to be a need in the industry for IT systems that support architecture based development.

Companies are challenged by managing the growing amount of information belonging to a product family or product program; finding the correct information at the correct time; and avoiding spending time on non-value adding activities of searching for information or reproducing information. Engineering data management (EDM), such as Product lifecycle Management (PLM), is today perceived as a systematic way to design, manage, direct, and control all the information needed to document the product through its entire lifespan: development, planning, design, production, use and disposal. The PLM concept is today supported by information management computer systems – *PLM systems*. PLM system support, however, faces several challenges in handling architecture based development. Among the most important ones are:

- Representation of an architecture is complex as it involves information generated in multiple domains from different disciplines of stakeholders.
- The use of commercial IT systems is often at a concrete and detailed level, making it somewhat difficult to work on a conceptual level.
- Several systems are often used to serve the same purpose, e.g. two different CAD system brands, with a parallel set of models.

- Similar, yet different, product models are sometimes found in different IT systems, e.g. different bills of materials in PDM (*Product Data Management*) and ERP systems (*Enterprise Resource Planning*).
- The various systems are rarely well integrated and a lot of manual information exchange between systems often takes place.
- As the IT system portfolio often evolves in steps and without a predetermined plan, the result is little or no integration between the different systems.

The transformation in the industry from developing individual products to development of product families implies changes in the way development is carried out and how products are designed. The development process has become more complex as more products are developed simultaneously along with the diversity of technologies implemented in products. This implies that designers need support, e.g. methods, models, and tools, in handling the increased complexity. Finally, there seems to be a need for IT systems that support the way of working with architecture representations.

1.2 Objectives of the research

This PhD project has been carried out with collaboration between industrial companies and The Technical University of Denmark (DTU), at The Department of Mechanical Engineering (MEK), in The Section of Engineering Design and Product Development (K&P). The research presented in this thesis aims to contribute towards architecture based development supported by visual models in interaction with computer-based information management systems (PLM). The aim is focused on understanding the challenges designers face while identifying and developing architectures, and providing model and tool support for addressing some of the key challenges.

This research belongs to the field of applied research, i.e. the research is focused on practical applicability of the research results. According to Jörgensen *(1992)*, such research has two objectives: (a) a theoretical objective and (b) a practical objective, of which the theoretical objective serves primarily as a means to reaching the practical objective. The theoretical objective of the study is to contribute to and enhance the knowledge and understanding of theories within engineering design science. The theoretical research objective looks to:

- Contribute to the research areas of systems theories and model based design theory, with knowledge and understanding of how to model and address desired behavioural effects and structural composition of architectures.
- Contribute to information theory with knowledge and understanding of architecture modelling in interplay with a computer-based information management system.

The theoretical contributions are formulated in relation to the models of the investigated phenomena.

The primary practical objective of the study is to enhance the use of PLM systems for supporting the designer in identifying, developing, and representing architectures. The practical research objective looks to:

- Enhance the knowledge and understanding of architecture models represented in PLM systems for supporting design synthesis and design analysis activities.
- Enhance the knowledge and understanding of architecture models represented in PLM systems for handling multiple views on products.

- Enhance the knowledge and usability of PLM systems for visualising and assessing the completeness of a design, i.e. the status of a design during development.
- Enhance the knowledge and usability of PLM systems for assessing product cost during development.

Summarising the above objectives into more intangible goals: the theoretical goals aim at contributing to theory by means of insight, descriptions and axioms. The practical goals are prescriptively to develop models and methods that can be applied within companies.

1.3 Research Questions

Throughout this research project several different research questions have guided the work. The project has been conducted in iterations and the questions have been refined as the insight into the research area increased. The refinements have converged into five research questions described in the following sections. They are divided into two areas, namely *theoretical research questions* and *practical research questions*. The first two research questions address architecture modelling aspects that are of a theoretical nature, while the last three address practical aspects related to PLM support for architecture representation.

1.3.1 Theoretical research questions

In the *Theory of Technical systems*, Hubka and Eder describe structure as elements and relations between elements, Hubka & Eder *(1988)*. There exist attributes which define a technical system and thereby the elements and relations. Andreasen *(2011)* designates these attributes as design characteristics. The design characteristics define a technical system's elements and their relations i.e. structure of the system

The first research question is formulated with the incentive to understand the phenomena related to architecture models' ability to address structural composition and behavioural effects. Harlou describes the structural elements of architectures as organs and/or parts, Harlou *(2006)*, where the definition of organs and parts belongs to *The Theory of Domains,* Andreasen *(1980)*. The organ should be understood as an entity of a product that solves different tasks and interacts with other entities. An organ can be defined as a functional element (function carrier) of a product, based upon a working principle, which brings together a physical effect and form and material characteristics, able to create a desired effect (function), Andreasen, Howard & Bruun *(2014)*. The behavioural effects arise in the functionality of organs and/or parts, and in the meetings in the life phase systems, e.g. manufacturing, assembly, disposal, etc. RQ1 aims to investigate and understand the challenges for designers in modelling architectures' structural elements (organs and/or parts) and at the same time address the behavioural effects of the architecture.

> *RQ1: What phenomena related to modelling of product architectures, from a structural and behavioural point of view, pose a challenge to designers when developing product systems?*

To complement research question R1, which is centred on understanding the phenomena related to modelling structural elements and addressing behavioural effects, a second research question is stated for prescriptive research in creating modelling support.

> *RQ2: How can both structural and behavioural aspects, be addressed in integrated architecture models?*

The answers to RQ1 and RQ2 are treated in the suggested models and methods of Papers A and B, presented in Part 4.

1.3.2 Practical research questions

The practical research questions address PLM support for modelling and representing architectures. The third research question, R3, is centred on the practical implementation of architecture models in PLM systems. The intention here is to prescribe methods for using PLM systems in a systematic way to design, manage, direct, and control information needed to document an architecture, see Papers C, D and E.

> *RQ3: How can PLM systems support architecture models that emerge in steps of the design process?*

The fourth research question, R4, points at a specific task in the design process, namely the task of assessing product cost and thereby supporting control of whether a design differs from a set cost target. Supporting metrics for assessing the product cost are a part of the answer to this question, see Part 4 and Papers C and D.

> *RQ4: How can PLM systems be used for assessing product cost during the design process?*

The fifth research question, R5, is formulated with the incentive of creating support for assessing the status of designs during the development process.

> *RQ5: How can PLM systems be used for assessing the completeness of a design during the design process?*

Supporting metrics for assessing the progress of designs are a part of the answer to this question, see Part 4 and Papers C and D.

1.4 Scope and boundaries of this thesis

The scope and anticipated boundaries of this research are described in this chapter in the following paragraphs.

Theoretical research scope

The theoretical basis from which this research aims to give answers to the proposed research questions, see Chapter 1.3, has its cornerstone within engineering design science and it has a systems and design perspective on the topic of architecture based development. Many of the interesting phenomena related to product architectures have however to do with design activities, and a design process theory perspective is therefore also relevant. Moreover, architectures are information intensive which underline the need for support in information management. This gives three fundamental theoretical viewpoints, from which the research can be studied: (1) theories of systems, (2) theories of design processes, and (3) information management theories.

Assumptions for guiding the research

The scope of this research is based on five basic assumptions or working hypothesis. The believe in this research is that further progress can be made in direction of the assumptions. The assumptions are provisionally accepted as a basis for the research in order to guide it into tenable results.

1. *A product architecture has to be aligned with a market and a production architecture in order to achieve an optimal balance between external and internal performance demands.*
2. *Modularisation in architecture design is a means for accomplishing different objectives of a business.*
3. *Introduction of functional systems into computer-based systems supports the designers in synthesis activities when composing and characterising architectures.*
4. *Visual models of architectures create an overview; they work as boundary objects at meetings, and thereby create an extended basis for design decisions.*
5. *Management of architecture related information is a critical aspect of today's interdisciplinary design process, and has to include aspects of information, processes, organisation, and IT systems.*

The assumptions are initial statements and their argumentation will be treated in Part 4 and 5. The five assumptions together with the needs in industry and the interest from academia lead to the focus areas for this research. These are modelling of product architectures and computer-support to architecture based development. The aim is to contribute to:

- Enhancing the knowledge and understanding of modelling architectures that assist designers in converting desired behaviour to solutions for a range of products, by explicitly addressing structural and behavioural aspects of architectures.
- Enhancing the knowledge and understanding of how PLM systems can support architecture based development by representing architectures and related information.

Product and business scope

The product scope of this thesis is mainly on mechanical designs, but the proposed approaches can be used in other domains as well, of which mechatronics is the most apparent.

The empirical part of this research is carried out only in Denmark. However, several common denominators exist among the case companies: Selling products globally, having development activities globally, and having global production facilities.

Modelling scope

The combination of subsystems, components, interactions, and interfaces, defines the architecture of any single product, meaning that every product has an architecture. However, the scope of this research is to create support for making an architecture common across several products, i.e. the architecture modelling formalisms must be able to address product ranges and not only individual products. If nothing else is stated in the text, the word *architecture* is in this thesis referring to a *product architecture* for a product family.

The focus of modelling support is on both early and later development phases. The early development phases refer to the time period of a development project, where it is still unclear which requirements to fulfil, what elements to develop and how to produce them. Later development phases refer to the time period where most components are developed and their realisation determined, but there are still ongoing optimisation issues.

Additional boundaries

Although organisational aspects and motivation of designers play an important role in improving effectiveness and efficiency of the design process and quality of products, this research does not explicitly address those aspects. This is mainly due to the skills and background of the author, and the time constraints of the project.

1.5 Structure of the thesis

This thesis has been structured as illustrated in Figure 1. Part 1 gives an introduction to the research area and its objectives and research questions. It is further explained what the scope is for this thesis.

Part 2 is a description of the research approach followed and explains the focus with the underlying assumptions. Part 3 accounts for the basic theories relevant for this thesis.

Part 4 presents the research and its results, structured by the appended papers (in Part 7). Part 5 outlines the contributions by reflecting on results in relation to the stated research questions, validity and verifiability of research and results, the impact, and the encountered limitations of this research. Finally, an outlook on future research is described in the end of Part 5. Part 6 holds the reference list for this thesis.

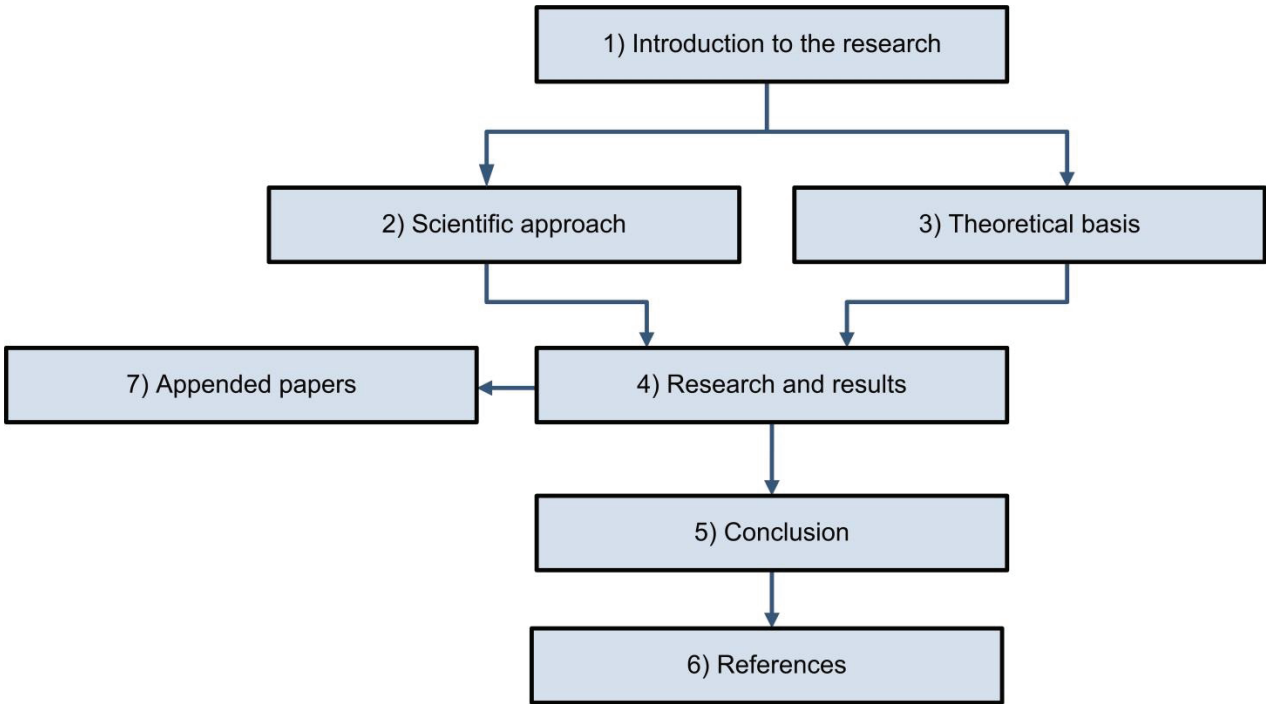


Figure 1 Graphical overview of thesis structure

2 Scientific approach

Part 2 describes: The research area, the research methods applied, the structure of the research design, and outlines the criteria for evaluating the research.

2.1 Research area

A model for representing the focus areas of a research project is the ARC diagram (*Areas of relevance and contribution*), Blessing & Chakrabarti *(2009)*. The diagram is useful in terms of visually representing, discussing, and reflecting on areas found relevant for a research project. It is furthermore a helpful tool for structuring the literature search and for focusing a research topic in terms of aims and research boundaries. An ARC diagram for the project, shown in Figure 2, illustrates the areas where contributions have been claimed, e.g. development of complex product systems, product architectures, graphical product modelling, computer-support, engineering data management, etc.

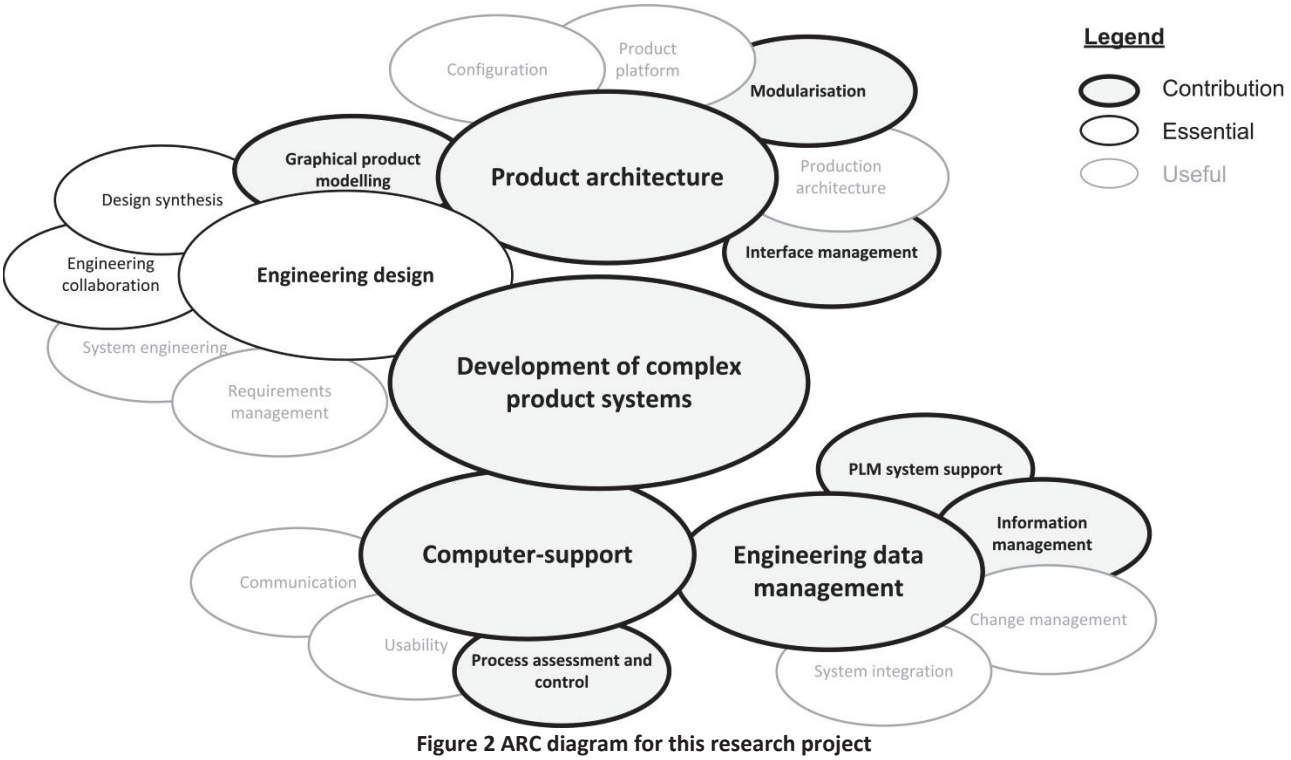


Figure 2 ARC diagram for this research project

The main research area of this project is stated to be *Development of complex product systems*. 'Complex product systems' is a term used in this thesis for describing product systems made up of a large number of parts or products (a product family or program) in a non-simple way. Simon *(1981)*, p. 195, describes complex systems in the following way: "*In such systems, the whole is more than the sum of the parts, not in an ultimate, metaphysical sense, but in the important pragmatic sense that, given the properties of the parts and the laws of their interaction, it is not a trivial matter to infer the properties of the whole*". The description can be used for understanding the product scope of the suggested models that are not aiming at supporting simple products with few parts.

Contributions in the research project also address the following areas:

- Computer-based support (Computer-support for representing architectures by structural models and combining them with behavioural and property models for deriving behavioural effects. Moreover, to support process assessment and control).
- Graphical product modelling (Creating a common language for assisting designers from different disciplines and with different backgrounds).
- Engineering data management (Combining information from multiple systems to represent architectures).
- PLM support (Using a standard IT system for representing architectures).
- Information management (Providing information structures for managing information).
- Modularisation (Supporting the creation of encapsulated structural modules).
- Architectures (Models of product architectures for decomposing a product system into variants, subsystems, modules, components, and interfaces among these and to the surroundings).
- Interface management (Enabling interface design, interface control and allocation of interface responsibility).

The contributions will be presented in Part 4 and they will be further elaborated on in Part 5.

2.2 Research methods

This research belongs to the class of applied research in the field of engineering design. The sort of research performed in this PhD project can be categorised as design science. The purpose of design science is to raise the quality of designing and designs. Design science differs from natural science on the aim of improving design practice, and thus does not have a purpose without its applicability in industry, Andreasen *(2009)*. By a similar definition, Simon *(1981)*, p. 129, describes that the task of design science is to create the artificial: how to make artefacts and how to design. Design Methodology may be seen as the label for the part of design science that critically studies the working procedures that designers follow, Roozenburg & Eekels *(1995)*.

Several authors have described approaches for design research methodology, e.g. Jörgensen *(1992)*, Blessing & Chakrabarti *(2009)*, etc. The approach used in this research cannot be described as one single design research methodology, but rather as sub-sets of several available approaches, as it has been beneficial to adapt the most relevant parts of different methodologies for this research. The methodologies described in the following sections have formed the basis of this project's research methodology, where the most significant influence comes from the *Design Research Methodology,* Blessing & Chakrabarti *(2009)*.

2.2.1 Action research

Action research is when the researcher enters a real-world situation and aims to improve it and acquire knowledge, Checkland & Holwell *(1998)*, see Figure 3. Action research is concerned with the study of phenomena that would not have occurred without the researcher's influence, Coughlan & Coghlan *(2002)*. It is a process where the experience and knowledge gained are the research results. This can affect the situation in action-based research, where it can be difficult to distinguish clearly between descriptive and prescriptive phases.

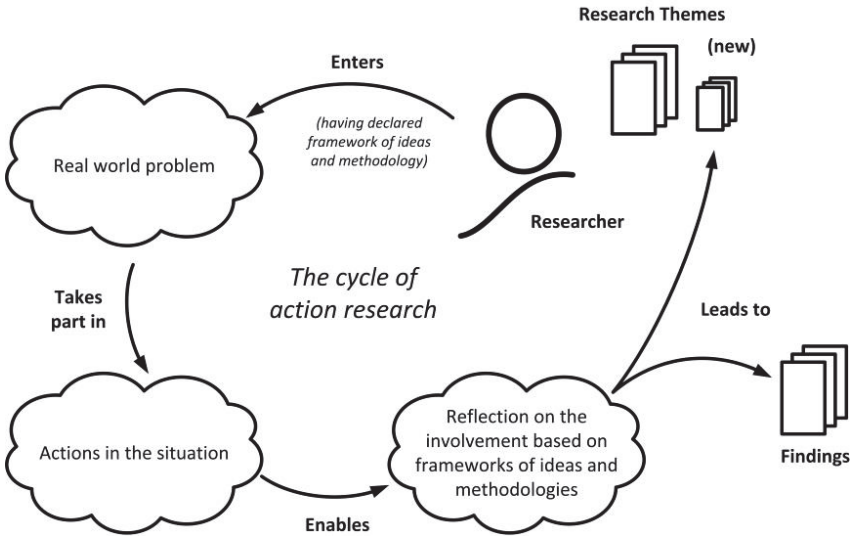


Figure 3 The action research cycle, redrawn from Checkland & Holwell *(1998)*

This project has tried to benefit from industrial collaboration, and has been undertaken by using real-world situations and their sometimes unpredictable nature. The research as a result switches between being a descriptive and prescriptive study, e.g. when implementing and refining methods and tools.

2.2.2 Applied research

Applied research is research which has been designed to apply findings to solve a specific, existing problem, Collis, Hussey & Hussey *(2003)*, p. 223. According to Jörgensen *(1992)*, research is both problem based and theory based, as was the case with this research project, see Figure 4.

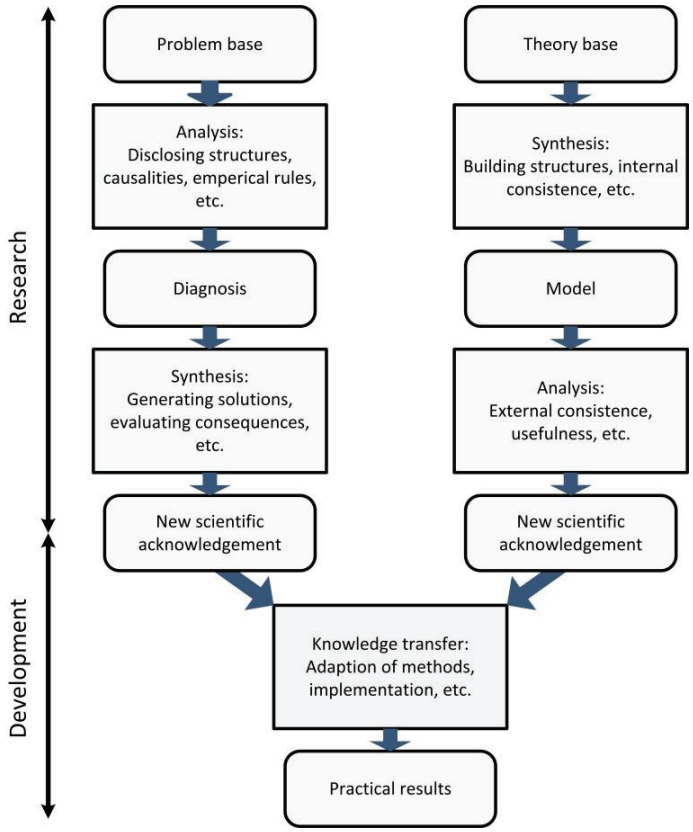


Figure 4 Problem-based and theory-based engineering design research methods, redrawn from Jörgensen *(1992)*

Problem based research is a consequence of the participation in projects within companies and being in continuous dialogue. The theory based approach is the involvement in research communities at universities. Therefore, research may be approached with two different methods, i.e. starting by either analysing or synthesising.

However, in practice, most research projects will involve both paradigms to various degrees, Jensen *(1999)*. This has also been the case during this project's execution, where the research progress has switched between analysing and synthesising activities.

2.2.3 Case study

Yin *(2009)*, describes case studies as a research methodology that allows the researcher to study a phenomenon in real circumstances. The method is useful where the number of variables far outstrip the number of data points, e.g. for understanding new phenomena, Eisenhardt *(1989a)*. As the methodology of this research, the case study looks like action research, with the exception that the case study does not imply interference by the researcher, though it can be difficult to avoid since the mere presence of the researcher can lead to changed behaviour of the involved people. The case study approach has been applied in the descriptive phases of the project to improve the understanding of practical problems.

2.2.4 Engineering design research framework

The framework of the Design Research Methodology (DRM) proposed by Blessing & Chakrabarti *(2009)*, is shown in Figure 5. The main concept of the DRM framework is to view the research as progressing through distinct research stages each with consigned objectives. *The Research Clarification phase* is used for finding evidence for the initial assumption. The main outcomes of the stage are to define the goals for the research and to determine the criteria against which the research can be evaluated. In the following phase, *Descriptive study I*, the researcher elaborates on the understanding of the situation and the influencing factors which can positively affect the problem area of interest. Having decided what influencing factors to target, the *Prescriptive Study* is used for synthesising support which will improve the situation. In *Descriptive Study II* the support is tested to validate to what extent the support has had the intended effect.

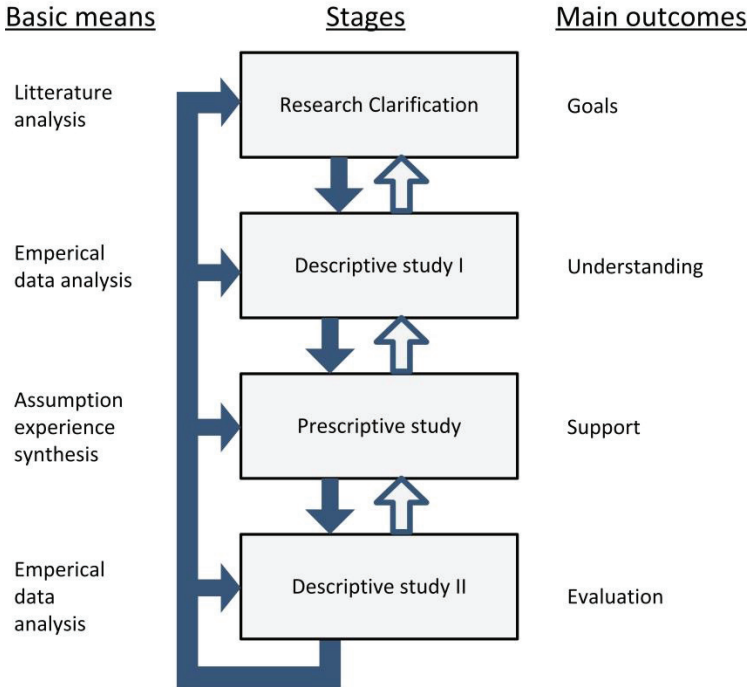


Figure 5 DRM framework, redrawn from Blessing & Chakrabarti *(2009)*

This research project has used the DRM as guidance for progressing and for delivering outcomes, i.e. goal setting, understanding the phenomenon, developing support, and evaluating the support. The model shows that the feedback between stages is of great importance for refining and clarifying the direction of the research.

2.3 Research design

2.3.1 Research plan

The clarification of the research in this thesis is best described as a continuous process. The synthesis of theory, methods, and models drove the clarification and research activities in general, rather than being encapsulated into a single phase. The DRM by Blessing & Chakrabarti *(2009)* is a clear methodology which facilitates the research project, and can be used for devising the overall research design for addressing the stated research questions. Blessing and Chakrabarti classify 7 types of design research projects. The project types differ in terms of research questions and hypotheses, and are also influenced by available time and resources. The project presented in this thesis best fits with the combination of stages represented in the 3rd type, emphasised by the dotted blue box, see Figure 6.

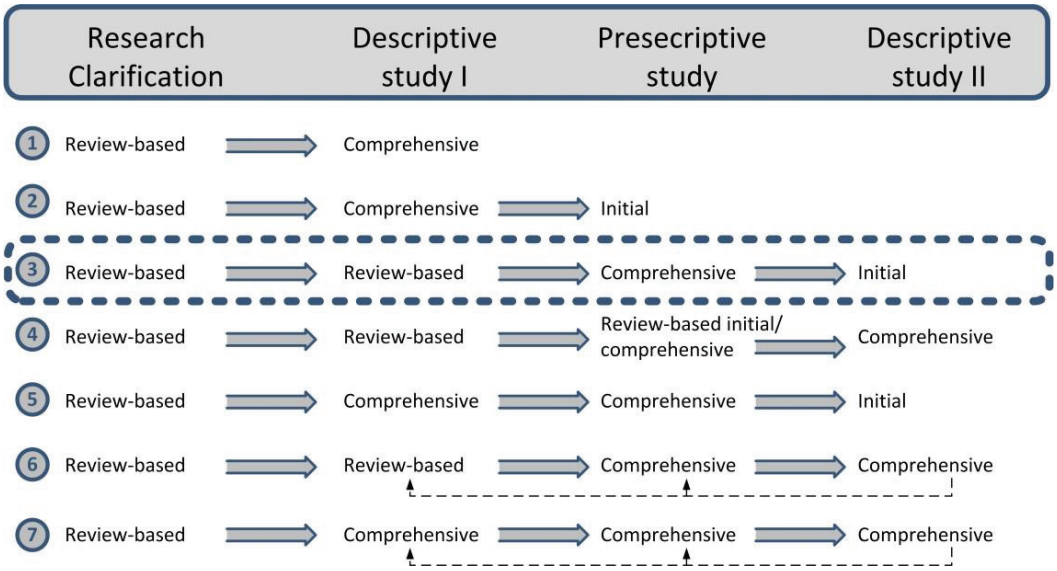


Figure 6 Types of design research projects and their main focus, redrawn from Blessing & Chakrabarti *(2009)*. **This project fits with type number three.**

The stages are not undertaken in the same depth as the research questions of this project are focused around practical support. The prescriptive activities have in that way been more comprehensive than the descriptive activities. Evaluating the suggested support in Descriptive Study II has, considering the time limitations in this project, been of an initial nature. Even though this research project has been shifting between clarification, descriptive research and prescriptive research, the DRM has been used as the overall applied methodology for managing the progress of the project. Figure 7 illustrates which papers have contributed to which research questions and activities. The mapping does not provide a detailed picture of the process followed in this project, but it is sufficient to describe the relations between questions, papers and research stages. Each of the different stages in this research is described in the section below.

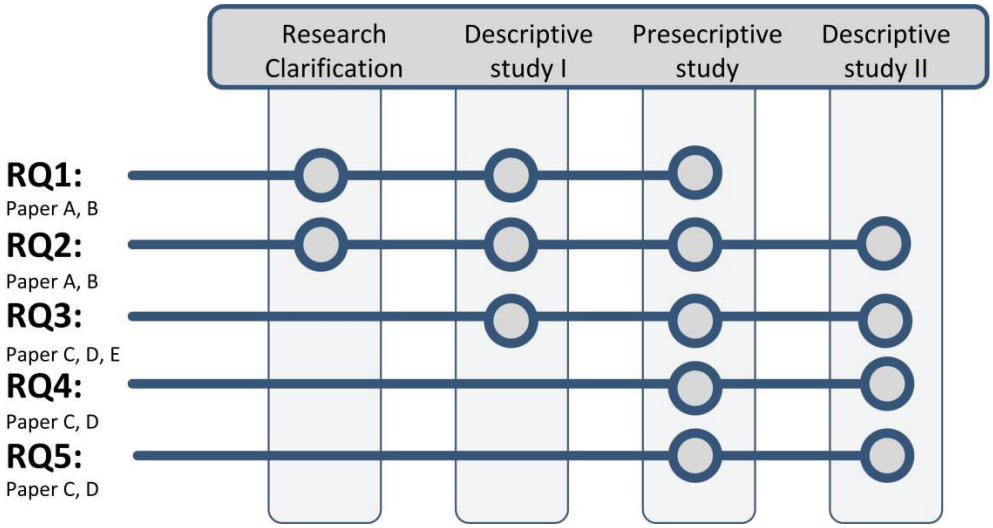


Figure 7 Relation between research questions, research stages and produced papers.

Research clarification

The role of the Research clarification stages is to identify the goals the research is expected to realise, the focus, the main research problems, questions, the relevant disciplines and areas to be reviewed and those to contribute. Furthermore, the role of this stage is to develop an initial representation of the existing situation, the *Reference Model* (not included in this thesis), and the desired situation, the *Impact Model*. This is achieved by a literature review in the area of the stated research problems and questions.

Descriptive study I

The role of the Descriptive study I stage is to obtain a better understanding of the existing situation by identifying and clarifying in more detail the factors that influence the preliminary criteria (from the research clarification stage). The outcome is to complete the reference model and to suggest key factors suitable for addressing the prescriptive stage, as these are likely to lead to an improvement of the situation. This is achieved by a literature review of empirical studies and own empirical studies.

Prescriptive study

The role of the Prescriptive study is to use the understanding from the previous stages to develop the intended support and to realise this in the actual support to such a level of detail that its effects compared to the stated criteria can be evaluated. The intended support being implemented in an industrial setting can then be initially evaluated in Descriptive study II within the timeframe of this project.

Descriptive study II

The role of Descriptive study II is to identify whether the support can be used in the situation for which it is intended and whether it addresses the factors it is supposed to i.e. evaluating the usability. Furthermore, the role of the stage is to identify whether the support contributes to success, addressing usefulness, its implication and side effects. This has been undertaken and described in Part 5 by evaluating the research and its results.

2.3.2 Case studies

Experience from working in the industry as a product development specialist and architecture consultant, has been a valuable basis for initiating the case studies. During the last 7 years, I have participated in a number of projects and gained insights into the challenges related to engineering design and product development encountered by development teams and decision makers. I have worked as a project manager, a domain specialist, and a team member in these projects and thus have encountered problems from different angles, providing me with a broad understanding of practical challenges in designs and in the design process.

The research project behind this thesis includes 5 case studies conducted in Scandinavian based companies from 2011 to 2014, see Table 1.

Several common denominators exist among the case companies:

- Selling products globally.
- Having development activities globally.
- Having global production facilities.
- All companies develop products of a mechanical nature with a varying degree of electronic and software content.

The 5 case studies are described in Table 1, illustrating the case characteristics and relations to research questions. The characteristics are: *Product type*, *Business type*, *Industry type*, whether *project based or product based*, *Research focus*, relation to *Research questions*, and the *Duration of the study*.

Table 1 Anonymised case studies conducted during the research project

Case study #	Buisness type	Industry type	Project/ product based	Description	Research focus	Research questions	Duration
1	Professional	Energy	OEM/ Project	Support architecture identification	Support for architecture identification - architecture for OEM product family	RQ1, RQ2	12 months
2	Consumer	Medical	Product	Support proactive architecture development (I)	Proactive modelliing of architectures: - Product - Production	RQ1, RQ2	4 months
3	Consumer	Medical	Product	Support proactive architecture development (II)	Proactive modelliing of architectures: - Market - Product - Production	RQ1, RQ2	4 months
4	Professional	Energy	Product	Identify modular architecture	Support identification of modular architecture - Assesment on architecture impact in lifephase meetings - Quantification of architecture	RQ1, RQ2	4 months
5	Professional	Energy	Product	Support PLM implementation	Suport implementation of architectures in PLM system - Development of new PLM centered processes and tools	RQ1, RQ2, RQ3, RQ4, RQ5	4 months

Anonymisation of cases

There is a trade-off between a researcher's access to data and the company's willingness to make results public. Having access to confidential information inside development activities and access to the most experienced resources have been weighed heavier than making all case details public afterwards.

The disadvantage of this approach is that it has not been possible to present the cases without having to anonymise the companies and their products. Knowing the companies and their products, as they all are

major players in their respective industry segments, could have made the presentation of results more powerful to readers.

However, the benefit of this approach is that the author has been able to work with interesting cases of central importance to case companies, and to work with the best talent in the companies, as they are often selected for the most challenging and critical projects.

2.4 Criteria for evaluating the research

For a research area aiming to improve a situation, formulating criteria for success is essential to be able to determine whether the results help achieve the aim, Blessing & Chakrabarti *(2009)*. The criteria for evaluating the research are part of the research clarification stages, but have been refined during the project into the ones represented in Figure 8. Blessing and Chakrabarti propose models of the research to describe networks of influencing factors. The models represent two situations of research, namely the existing situation (The Reference Model) and the desired situation (The Impact Model). The model for the desired situation should represent the criteria for evaluating the research. Blessing & Chakrabarti *(2009)* distinguish between success criteria, which refer to the overall aim of the research, and measurable criteria, which translate the overall aim into a criterion that can be used to determine whether the aim is achieved or not. The success criteria as well as the measurable criteria should be in concordance with the research objectives formulated in chapter 1.2. The network of influencing factors is established based on evidence in literature and partly on assumptions. The (+)/(-) attributes in the model are indications of values, i.e. the combination of values on an arrow describe how the value of the attribute of the factor at one end relates to the value of the attribute of the factor at the other end.

Figure 8 represents a model of a line of argumentation. The model is a very simplified representation of the desired situation since many other factors have a potential influence on the measurable factors. Ultimately, the aim of this research is to improve the ability of companies to gain profit. It should be taken into account that it is often quite difficult to measure the effects of a single research contribution on broad success criteria like the overall profit. Since engineering design research is often based on case studies, the studies tend to be blurred by a number of different uncontrollable factors in the case companies. Therefore, it can be difficult to separate the effects of the research contribution on the performance of the company, Pedersen *(2009)*.

In Figure 8 the success criterion of the research is shown as the desire to effectively and efficiently develop products. Although there are ways to measure effectiveness and efficiency, it is not feasible to expect a measurable impact within the time frame of this research work, though it is necessary to interpret this criterion into something more measurable. The measurable criteria are shown as the factors of lead time in product development, product cost, and product quality. Due to the time limitations of this PhD project, it is not possible to measure the research's isolated impact on product quality and product cost. However, it has been possible to gain input on the reduction of lead time in development, resulting from the application of suggested models and tools in the industrial cases, based on reactions from stakeholders involved in the case projects and their perception of the results, see Papers A, B, C and D.

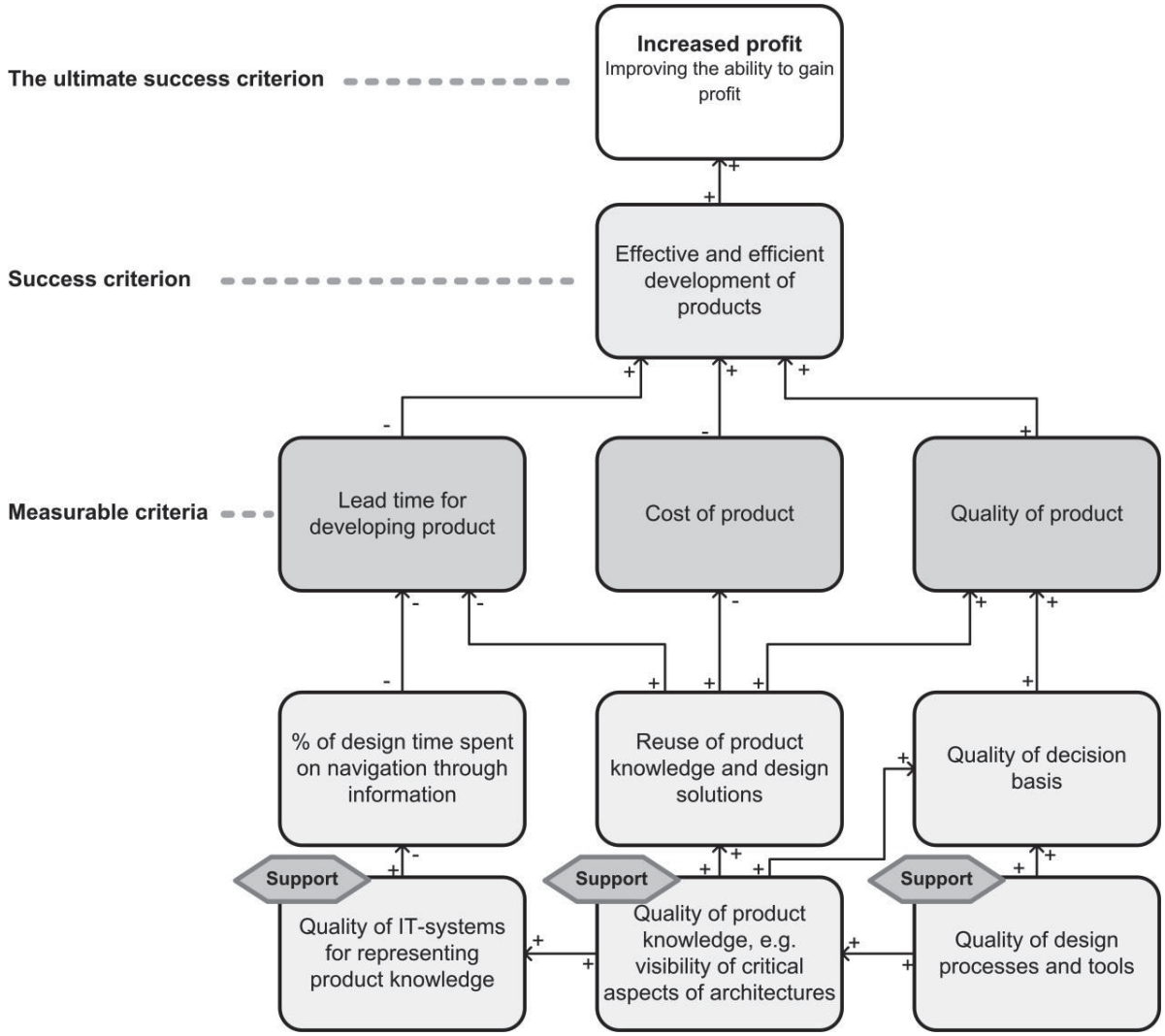


Figure 8 Impact Model for the research indicating the focus of evaluation of the support

The evidence in literature and assumptions behind the network of factors leading to the measurable criterion '*Lead time for developing products*' are as follows:

- The quality of design processes and supporting tools is essential for the quality of a design and evaluation of its life phase meetings, Olesen *(1992)*. A meeting is whenever a product takes part in an action where the product, an operator, and a life phase system are interacting. By increasing the quality of synthesis activities, the quality of the decision basis can be increased leading to the probability of increasing the quality of the actual design (product), Mørup *(1993)*. This research seeks to contribute to the theory of design processes with knowledge and understanding of architecture modelling in interaction with a computer-based information management system. The support of this research is proposed to increase the quality of design processes and design tools.
- The quality of product knowledge, e.g. architecture descriptions of product families, is influenced by the quality of the design process and supporting tools, Harlou *(2006)*, Ulrich *(1995)*. By increasing quality, the ability for reusing product knowledge and design solutions is increased, Gershenson, Prasad & Zhang *(2004)*. This again influences the lead time for developing products, the cost and the quality of products, Andreasen, Mortensen & Harlou *(2004)*. This research seeks

to contribute with knowledge and understanding on how to model structural and behavioural aspects of architectures in order to, among other objectives, enable reuse of product knowledge and design solutions. The support of this research is proposed to increase the quality of product knowledge representations by means of architecture models in interaction with computer-based design support systems.

- The various IT systems in a company are rarely well integrated and a lot of manual information is exchanged between systems, Weber, Werner & Deubel *(2003)*, Sääksvuori & Immonen *(2008)*. As the IT system portfolio in manufacturing companies often evolves in many pulses and without a predetermined plan, the result is little or no integration between the different systems. This again leads to design time being used for navigation through different systems to find, extract, correct, align, and create product information used for developing products, Sudarsan et al. *(2005)*. Yet again, this has a negative impact on the total lead time, i.e. the time spent to solve a given design task. This research seeks to enhance the knowledge and understanding of using PLM systems for not only integrating IT systems, but also for supporting synthesis and analysis activities in architecture based development.

The measurable criterion is used when evaluating the contributions of this research, see Chapter 2.4.

2.5 Research activities

This research is conducted through a diversity of activities: Literature studies, collaboration with research groups, participation in conferences, research workshops, teaching, experiments, and interviews within companies.

2.5.1 Literature

Literature studies have been conducted on systems theories, design process theories, model based design theory, product architecture theories, information theory, and engineering data management research, platform development and modularisation, and PLM system support approaches. The theoretical basis is presented in Part 3.

2.5.2 National and international research collaboration

The Product Architecture Group at DTU has contributed to the research focus for this project. Throughout the project, informal research discussions and presentations have been carried out with colleagues from the group. The research has furthermore benefitted from being presented at international research group seminars at *Produktentwicklung und Konstruktionstechnik* (PKT) at *Technische Universität Hamburg* (TUHH), and *Department of Product and Production Development*, at *Chalmers University of Technology.*

2.5.3 Conferences and workshops

Participation at conferences and workshops has played an important role in forming, focusing, and refining the contributions of this research. The conferences and workshops attended during this PhD project are:

- *ICED 2011*, Denmark, Copenhagen, 2011
- *Produktudviklingsdagen 2011*, Denmark, Copenhagen, 2011
- *PLM conference*, Denmark, Odense, 2011
- *PLM12*, Canada, Montreal, 2012
- *NordDesign 2012*, Denmark, Aalborg, 2012
- *Produktudviklingsdagen 2012*, Denmark, Copenhagen, 2012
- *Radikal Forenkling*, Denmark, Copenhagen, 2012.

- *Produktudviklingsdagen 2013*, Denmark, Copenhagen, 2013
- *Design 14*, Croatia, Dubrovnik, 2014

2.5.4 Courses

Courses are an important sub-activity in the Danish set-up of a PhD project. The courses followed in this project include engineering and general research courses, a system engineering course, and a management technology course:

- The Technical University of Denmark (DTU*): Research and PhD studies* (2011), 2,5 ETCS points
- Copenhagen Business School (CBS): *Management Technology, inter-organisational relations and performance management* (2011), 7 ETCS points
- Design Society: *Summer school in Engineering Design Research* (2012), 5 ETCS points
- The Technical University of Denmark (DTU): *Systems Engineering* (2012), 10 ETCS points
- Copenhagen Business School (CBS): *Getting my research into journals* (2013), 1,5 ECTS points
- EDEN doctoral seminar: *Doctoral dissertation writing and publishing* (2013), 4 ETCS points

2.5.5 Teaching at DTU

Results from this research have been taught in single lectures to Master students at The Technical University of Denmark (*Design and Documentation* and *Technical platforms and architectures*). A part of the PhD study was also to supervise Master thesis students. The supervision has been carried out together with Professor Niels Henrik Mortensen and colleagues from the Product Architecture Group.

2.5.6 Experiments in industry

The empirical data in the case studies, see Section 2.3.2, gathered in this research, have been gained by participation in different development projects in the industry. The involvement in different projects has been over timespans of 4-12 months. The participation in projects has contributed to insight within the phenomena of developing products based on architectures, and for supporting modelling and representation of architectures in PLM systems.

3 Theoretical basis

The aim of Part 3 is to introduce the theoretical basis. This research's need for theoretical support and its relations to identified relevant theory are explained. The objective of this part is not to question the theoretical basis, but rather to outline how the basis contributes to and positions this research.

3.1 Introduction of theoretical base

This chapter positions the theoretical basis for this research, and its objective is to investigate theories that form the theoretical basis. The investigation has to clarify the limitations of the present research knowledge and working practice and later help justify the research contribution of this thesis. Moreover, the theoretical basis serves as the basis for the prescriptive work in this thesis. The central questions in relation to establishing a theoretical basis in this part are:

1. *What is the theoretical basis of this research?*
2. *How can the theoretical basis contribute to this research?*

The first question will be answered in the introduction, while the second will be answered for each of the identified theories when they are introduced. For identifying the theoretical basis, one has to look at the research questions, see Chapter 1.3. The first two research questions investigate the phenomena of how to take behaviour into account when synthesising structure in a product architecture model. These questions call for support in theories and models within engineering design and product development, e.g. systems theory, design process theory, design modelling theory, and architecture theory including theories of platforms and modularisation. The last three research questions, see Chapter 1.3, are positioned for understanding the phenomena of how computer-systems can be used for representing product architectures and information related to them. These questions call for support in theories, models, and tools within information theory and the means for computer-supported engineering design. Consequently, the theoretical basis, or the frame of reference, for this research work is divided into three areas:

- ***Engineering design.*** This chapter presents theories and models within: engineering design, product development, and the design science community.
- ***Multi-product development.*** This chapter presents theories and models in relation to multi-product development: architecture theory, platform-based product development, and modularisation.
- ***Computer-support for information management and engineering design.*** This chapter presents theories, models, and tools/IT systems in relation to: information management and computer-support to engineering design.

The theoretical basis is influenced by the work of researchers from the Section of Engineering Design and Product development at the Department of Mechanical Engineering (DTU). This is due to a strong tradition of building on past contributions in the research group. Some could argue that the theories are local and unknown to the broad community of international design science researchers. Though, in order to get a more rigorous frame of reference, theories from international research groups have been consulted. The purpose of this part is not to provide a detailed review of literature, as this is reserved for the appended

papers. Instead, the purpose is to explain in the context of this PhD project how the presented contributions form the basis for this research's results.

3.2 Engineering design

3.2.1 Systems theories in general

Systems theory is a meta-theory about laws, methods, and modelling for systems, i.e. objects which have a system nature, Mortensen *(1999)*. Hubka & Eder *(1988)* describe a hierarchical sub-division of systems to see the boundaries of the individual system terms, see Figure 9. The systems of focus in this thesis are technical systems while machine systems of a mechanical nature are the main research objects.

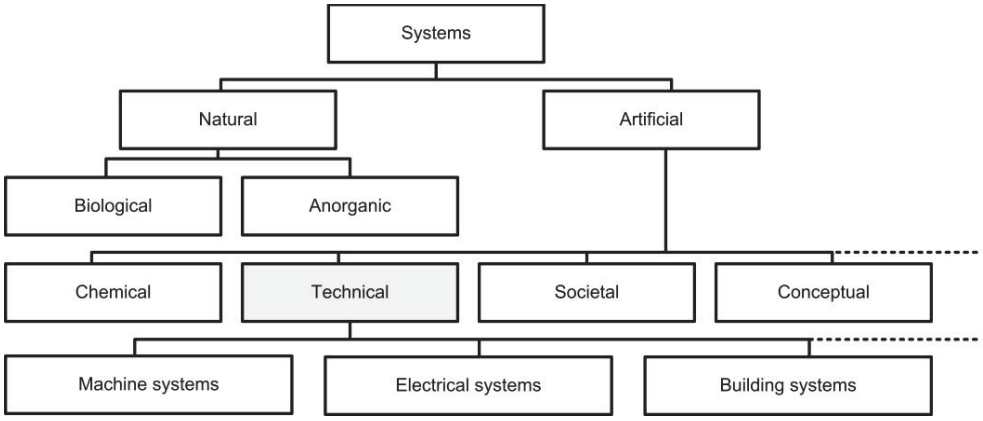


Figure 9 Hierarchy of systems, redrawn from Hubka & Eder *(1988)*

The word *system* is tainted by different meanings, and is used in many sciences and by many authors. A common perception is that a system is a concept and thus an abstraction used to describe and model a specific area of interest. The problem with these many theories is that their basic concepts cannot be proven. Their validity in design science should however be found in their productivity to assist design tools of some general application value. Systems theory provides a concept and a set of rules for the abstract modelling of technical artefacts and for the decomposition of such artefacts into subsystems on hierarchical levels, Buur *(1990)*. Any object which is composed of elements and their relations, and can be regarded separately from its surroundings may be considered a system, e.g. an architecture. A system can be represented in a model based upon a certain viewpoint which defines the elements of the system and their relations. A system carries structure, i.e. the elements and their relations (arrangement, architecture), and behaviour, i.e. the system's response, dependent on stimuli, structure and states, see Figure 10. A system is considered recursive, thus enabling it to be an element of a larger system.

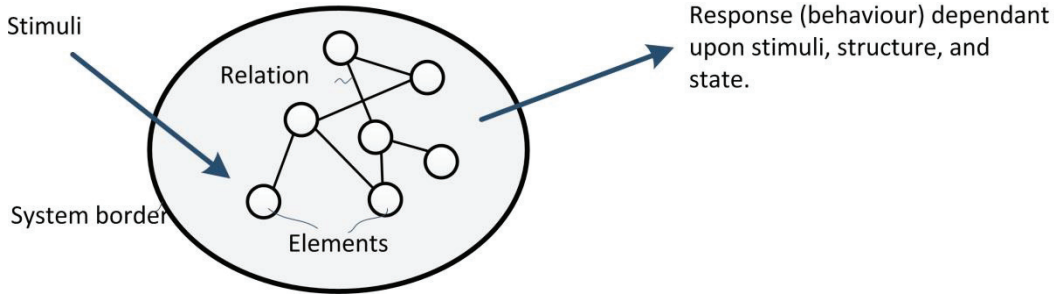


Figure 10 A general model of a system of an artefact, redrawn from Hubka & Eder *(1988)*.

Structural and behavioural systems phenomena

The concepts of a system's structure and behaviour are central to this thesis and will be treated in this chapter. Systems theory and its models articulate systems' structural composition and their derived

behavioural attributes like function and properties. Based on the work of Hubka & Eder *(1988)*, Andreasen *(1980)* and Mortensen *(1999)*, Andreasen *(2011)* proposed a classification of a system's attributes. Andreasen et al. propose the following definitions of the two classes, namely characteristics and properties, Andreasen, Howard & Bruun *(2014)*, p. 173:

- *Characteristics, which are a class of attributes of an object that define the means by which the object's properties are realised.*
- *Properties, which are a class of attributes of an object by which it shows its appearance in the widest sense and by which it creates its relation to the surroundings.*

From a systems point of view a system's structure is to be seen as the composition of characteristics. It means that some characteristics define the elements; others define the structure, which presents how these elements are related. Properties are relational, i.e. they are observed or present in situations where somebody is in relation to the device (looking, buying, deploying etc.), or they are brought in relation to certain surroundings and operate in synergy with these: systems for production, assembly, sales, distribution, installing, use, maintenance etc., Andreasen *(2011)*. Figure 11 shows how a system's attributes can either be structural characteristics describing what the system *is*, or behavioural properties describing what the system *does*. The later described *Theory of Domains* proposes a set of concepts, which articulate causal relations between structural characteristics and behavioural properties.

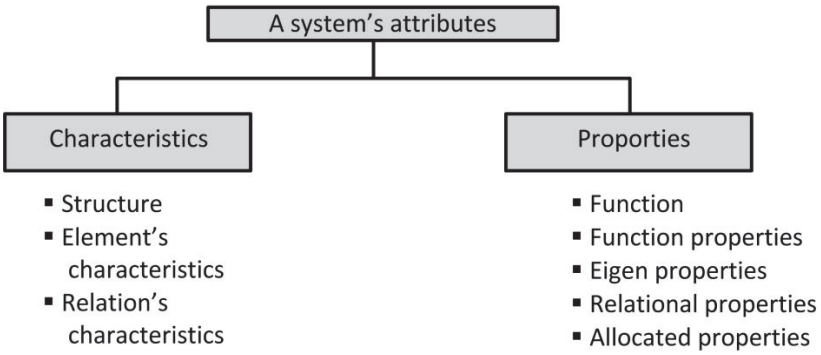


Figure 11 Classes of structural and behavioural attributes, based on Andreasen *(2011)*

This understanding of characteristics and properties is very much in accordance with the work of Weber *(2014)*, p. 329, who also employs a distinction between the characteristics and properties of a product:

- *Characteristics are made up of the parts structure, shape, dimensions, materials and surfaces of a product. They can be directly influenced or determined by the development engineer/designer.*
- *Properties describe the product's behaviour, e.g. function, weight, safety and reliability, aesthetic properties, but also things like manufacturability, assemblability, testability, environmental friendliness and cost. They cannot be directly influenced by the developer/designer.*

Weber's work is articulated in two theories, one called *Characteristics - Properties Modelling* (CPM) and one called *Property - Driven Development* (PDD), it means respectively product and process modelling.

Gero, Tham & Lee *(1992)* suggest a Function-Behaviour-Structure (FBS) framework to represent the process of designing as a set of transformations between function, behaviour and structure. Gero et al. state that the most basic view of designing consists of transformations from function to behaviour, and from behaviour to structure. In this view, behaviour is understood as the performance expected to achieve the desired function. Once a structure is produced, it must be checked whether the artefact's actual

performance, based on the structure produced and the operating environment, matches the expected behaviour.

Yet again, the function-based design methods (Function structures) are characterised by establishing either a function model, Pahl, Beitz & Wallace *(1996)*, Otto & Wood *(2001)*, or the schematics of the product, Ulrich & Eppinger *(2011)*. Both approaches have a visual representation as an outcome. The function structure describes the flow of material, data, and energy through sub-functions of the product using a set of rules (e.g. the rules that are referred to as the functional basis which basically is a common language to describe functional elements). The schematic of the product is somewhat similar to the function model. But where the function model describes the product using functional elements, the schematics can describe both functional and physical elements - whichever is the most meaningful for the purpose of the representation. The functional structure forms the basis for several different approaches to design or re-design the products.

How systems theory contributes to this research

In general, systems theory provides a reference modelling framework for all products and interacting systems, which are all central when working with architectures. Structural and behavioral systems phenomena, provide explanatory power to this research's investigation of structural and behavioral aspects of architectures. Function reasoning is an activity in the design process that relates to the reasoning of structure-behaviour relations which is one of the support areas of the suggested approaches of this thesis. The Interface diagram presented in Paper A has similarities to a schematic of a function structure, as the model can describe both functional and physical elements.

3.2.2 Theory of Technical Systems

The Theory of Technical Systems, Hubka *(1985)*, is a systems theory that explicitly articulates the behaviour and structure of technical systems. The theory has its origin in the concepts of cybernetic modelling, Ashby *(1957)*, Klir, Valach & Klir *(1967)*. A technical system refers to all types of human made artefacts. In the Theory of Technical Systems, Hubka defined a model of a technical process system, see Figure 12. The technical process is a subset of a transformation process including the technical system. The technical process system is defined as consisting of four subsystems, a technical system (the artefact), a human system (the human operator), the environmental system (influences from the environment) and, finally, a technical process system (the meeting).

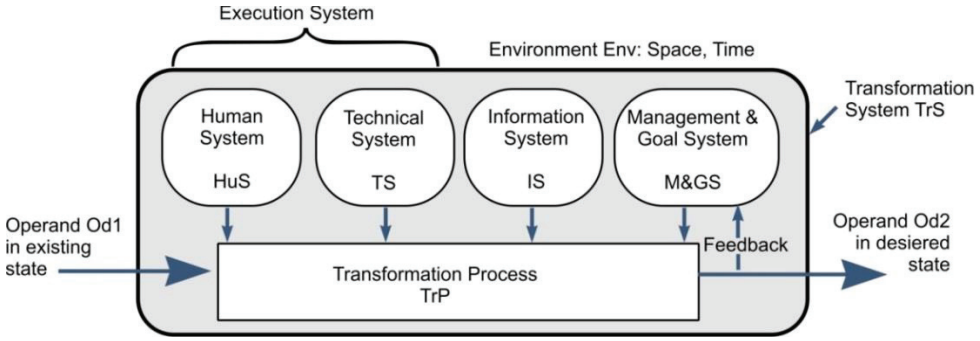


Figure 12 A technical process system consists of a technical process, an operand(-s), a technical system, an environment system and a human system, redrawn from Hubka & Eder *(1988)*

In the technical process, one or more operands (Op) are transformed from an input state to an output state. An operand can be material, data or energy. This process is enabled by one or more operators (TS, En, Hu) working together to deliver the effects that are necessary to execute the process.

How the Theory of Technical Systems contributes to this research

The Theory of Technical Systems contributes to this research because it provides an explanation to understanding and describing technical systems. The theory provides a reference for all descriptions of products and the interacting systems, which are central when working with the interaction of structural and behavioural aspects of technical systems.

3.2.3 Theory of domains

The Domain Theory, Andreasen *(1980)*, aims to understand artefacts in an analytical and synthesising way. The basic idea is illustrated in Figure 13, showing the three domains, where the activity, organ and part views lead to three system models. The three views are the answer to the question: 'How do you spell a product?', namely to spell how the product is used, how it functions and how it is built up. The result of the spelling is seen as the synthesis result: a full definition. The quality of the spelling should be precise semantics and syntax. By 'domain' the Theory of Domain refers to a taxonomic subdivision of the theory for the purpose of understanding the artefact from perspectives of design; it is reflected in the differences in the explanations of the concepts and phenomena of the domains, Andreasen, Howard & Bruun *(2014)*.

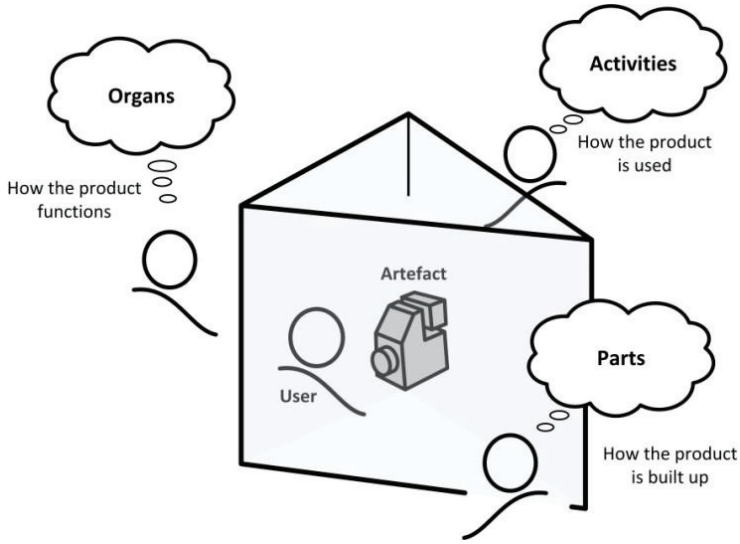


Figure 13 A system model of an artefact: The Domain Theory, Andreasen, Howard & Bruun *(2014)*

The three viewpoints are:

- ***The activity domain.*** The activity domain represents a single or sequence of transformations in which the product is utilised or transformed. The technical activity is determined by the user's application of the product. When the product is used it contributes to the transformation of operands of the classes: material, energy, information and/or biological objects. The operands are changed in terms of: material, energy, information or biological objects, characterised by their input and output state.

- ***The organ domain.*** The organ domain represents the function elements (or *means*) of a product, displaying a mode of action and a behaviour, which realise its function and carry its properties. An organ is based upon physical, chemical or biological objects. When stimuli (external effects) act on the organ, the organ delivers an effect which interacts with the surroundings. Stimuli and effects may be material, energy, information or biological objects.

- ***The part domain.*** The part domains represent the part system. A part is the elementary material element of a part system. Parts are the building elements of an organ, realising the organ's mode of action by the parts' physical states and interactions. A part's function (the part's work function) is

based upon physical, chemical and biological objects. The parts interface with other parts and the surroundings create the effects of the part.

The three domains are considered as isolated and interacting viewpoints for allowing structural and behavioural reasoning. The three viewpoints are used to divide the product system and depend on the intended purpose. The approach of modelling in multiple viewpoints is acknowledged as beneficial in design, e.g. by Pahl, Beitz & Wallace *(1996)*, Simon *(1981)*, Lindemann, Maurer & Braun *(2009)*.

How the Theory of Domains contributes to this research

The Theory of Domains is a powerful foundation for multi-view modelling from which reasoning about products can be carried out. The Theory of Domains contributes to this research by forming the basic idea of creating multiple views of architectures for analysing and synthesising product systems. Other viewpoints can be derived as combinations of the basic ones from The Theory of Domains, which again create a strong basis for analysing and evaluating the objects modelled. The causal linking between organ view and part views is in this thesis used as a concept for converting desired behaviour (functions) to solutions (parts) in a product system.

3.2.4 Theory of dispositions

According to Olesen *(1992)*, a product will during its lifetime go through a sequence of *meetings*. A meeting is in this context whenever a product takes part in an action where the product, an operator, and a life phase system are interacting, see Figure 14. The meetings are interesting because it is in the meetings that the performance of a product can be evaluated. Olesen proposes that performance can be evaluated along seven dimensions, the so-called *seven universal virtues*: cost, time, quality, flexibility, efficiency, risk and environmental effects. A consequence is that in order to reach the optimal performance, the product must be fitted to its life phase systems and/or the life phase systems must be fitted to the product.

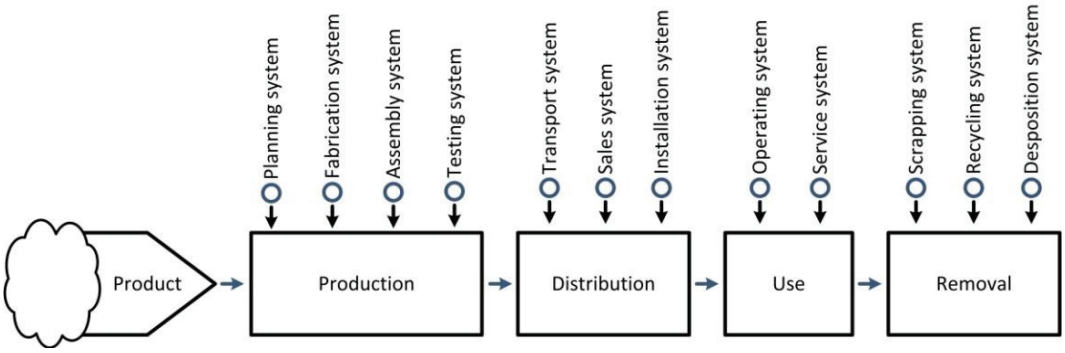


Figure 14 A product in its life phase system meetings, redrawn from Olesen *(1992)*

According to Olesen (*1992*), p.53, a disposition is understood as that part of a decision taken within one functional area which affects the type, content, efficiency or progress of activities within other functional areas.

Decisions about the product design facilitate or impede what solutions can be chosen in the design of the life cycle systems, see Figure 15. These decisions are made before the actual production, transport, use or disposal of the product, hence the term disposition.

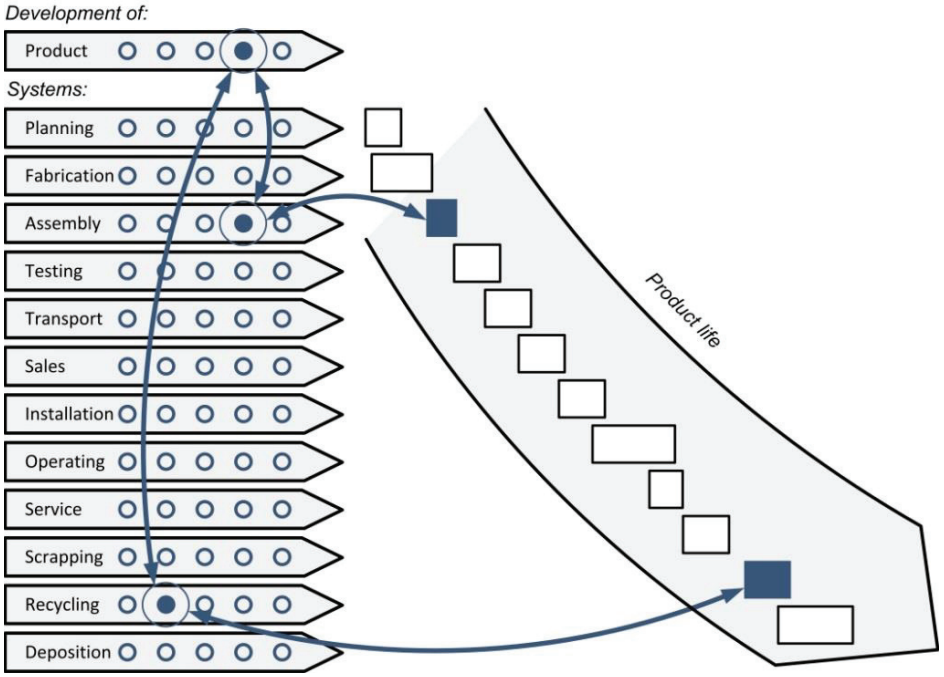


Figure 15 Theory of Dispositions – a score model, redrawn from Olesen *(1992)*

How the Theory of Dispositions contributes to this research

The Theory of Dispositions contributes to this research in a very fundamental way in the sense that a successful architecture of a product system needs to take many dispositions into account during the design phase to ensure an optimal fit of its life phase systems. The basic task of identifying a successful architecture is very much a task of balancing and prioritising dispositions for a product, which in this research is addressed by visual architecture models interacting with PLM systems.

3.2.5 The Genetic Design Model System

Inspired by the Theory of Technical Systems, Ferreirinha et al. *(1990)*, launched the word chromosome for a composed model consisting of four views or domains. Andreasen formulated the product modelling concept *The Chromosome Model* that includes causal links to connect elements from those different domains, Andreasen *(1992)*. An elaboration and refinement of the theory is found in the Genetic Design Model System (GDMS), proposed by Mortensen *(1999)*. The GDMS contains a structural/physical and a functional/behavioural part; while at the same time mapping the activity, organ and part domains from the Theory of Domains, see Figure 16. The top refers to the technologies in the product that are seen as the carriers of effects in the activity domain of The Domain Theory. The introduction of the term task adds to the functional/physical split in the domain theory. A task is the purposeful job done by a part in the part domain. The parts contribute to organs that in return carry functions. The functions belong to the behaviour class, where behaviour is split into an intended (Soll) and realised (Ist) situation. An important split is the clear separation of modelling units, i.e. the meeting and the design. In the case of the transformation, i.e. the top level of activities, the modelling unit is the meeting, as described in Olesen's Theory of Dispositions, Olesen *(1992)*.

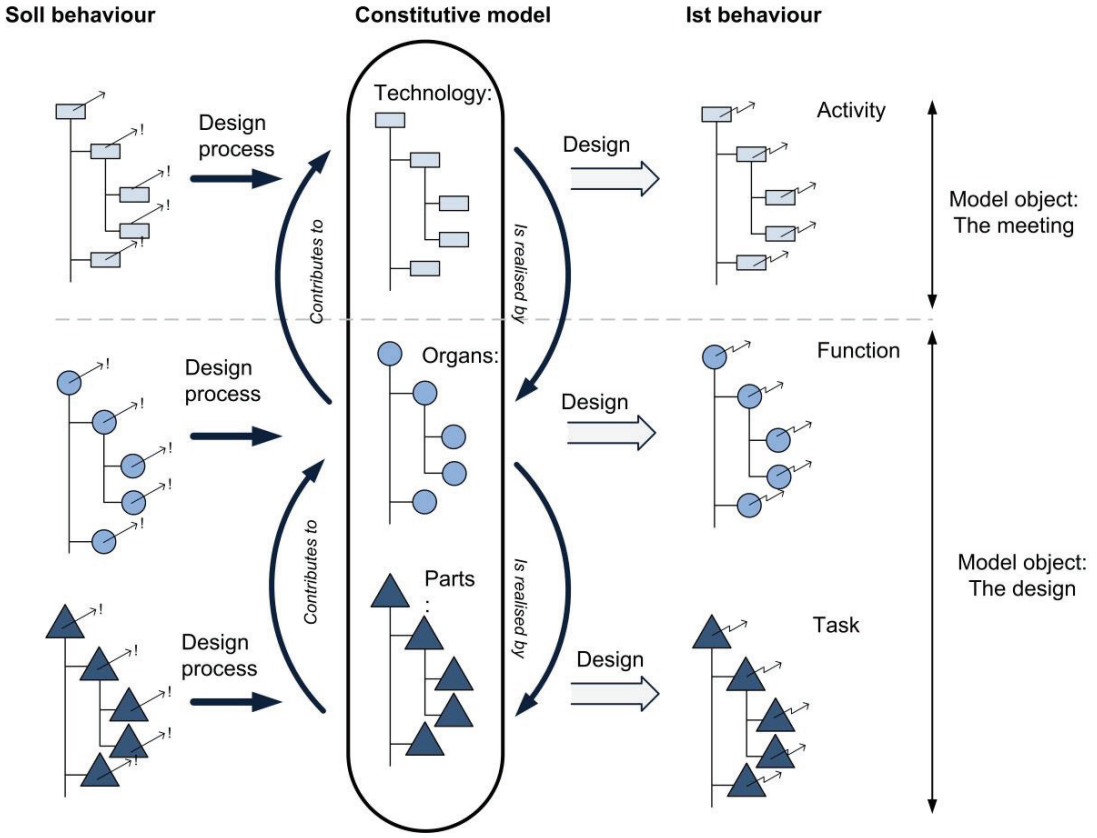


Figure 16 The GMDS provides a constitutive and behavioural description of a product model, redrawn from Mortensen *(1999)*

Handling multiple views in a Genetic Design Model System is done by view models (life phase and property models) and models which can describe the design in the different life phases and the related structure changes. Property views are for example strength, stiffness and cost. Life phase views are the design in different life phases, e.g. production, transport, use and service. When the design is transferred from one life phase to another, the structure of the design will normally change, e.g. the transitions from production to the transport phase, packing, handling equipment and labelling might be added to the design.

How the GDMS contributes to this research

The GDMS is especially relevant when models are intended as support for the activity of defining the structural elements and their desired behaviour. The concept of adding multiple property views, i.e. cost, weight, strength, etc., in the description of an architecture is applied in this research by means of the utilised PLM system. The GDMS points out the causal nature of relations between design characteristics and properties. Thus the behaviour can be derived when the design characteristics have been determined, which is an important aspect of using modelling techniques for supporting the development of architectures.

3.2.6 Design process theories

According to Andreasen & Hein *(1987)*, the development process can be described in terms of single models on four levels: Product planning, product development, product synthesis, and problem solving, see Figure 17. The design process theory is based on descriptive and prescriptive models from Pahl, Beitz & Wallace *(1996)*, Hubka & Eder *(1988)*. Product planning is related to activities where decisions regarding the introduction of new products and phasing out of existing products are made. Product development holds the activities, which together create the business. *The Integrated Product Development* model, Andreasen & Hein *(1987)*, proposes interplay among three classes of activities: Activities related to market, product and production. Tjalve *(1976)* suggests a model for synthesis of products. The model has its starting

point in the decomposition of the functions of the products. Hereafter, solutions are found by means of principle and quantified structures. Finally, general problem solving can be described as the steps which lead to the solution of a problem.

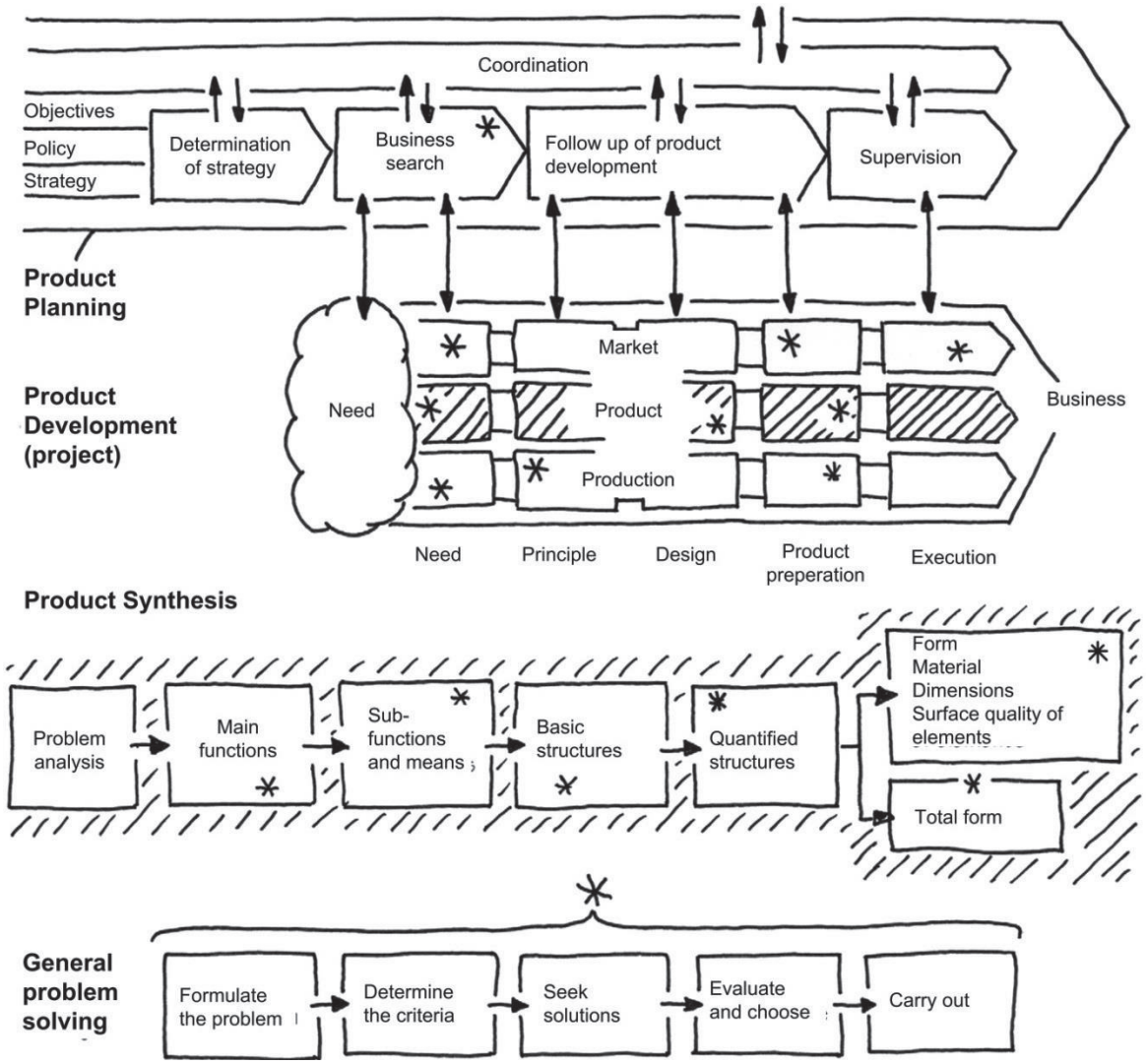


Figure 17 The development process can be described in terms of single models on four levels: Product planning, product development, product synthesis, and problem solving, Andreasen & Hein *(1987)*, **redrawn from** Harlou *(2006)*.

In relation to design process theory and systems theory, the approach of *Systems engineering* is defined as "*an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, and then proceeding with design synthesis and system validation while considering the complete problem: Operations, cost and schedule, performance, training and support, test, manufacturing, and disposal. Systems engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs*", International Council on Systems Engineering & Haskins *(2011)*, p. 6. The definition has a focus on the system design. The domain of systems engineering emerged as an effective way to manage complexity and change and it is primarily applied to large and complex projects (systems). The discipline of Systems Engineering can be characterised as technology independent procedures for development of systems.

How design process theory contributes to this research

The theories of design processes form the basis for understanding the developing processes of this research. Concerning application of architectures, it will have impact on all levels of the design process, i.e. product planning, product development, product synthesis and problem solving. In order to succeed with an architecture based approach it has to be fitted to the nature of product development, meaning that solutions are determined gradually, and architectures have to include certain flexibility in order to match different design projects and product variety. Furthermore, the spilt between market, product, and production activities, and their relations, are seen as an important theoretical basis for aligning architectures of market, product, and production. Alignment is about optimisation towards the architectures that satisfies the goals of the company in the best possible way with the given resource limitations. The concept of Systems engineering contributes to this research with descriptions of technical independent procedures for development, especially with focus on requirements and specification engineering, addressed in Paper E.

3.2.7 Model based design theory

Model based theories are theories about designing and designs based upon a model of an artefact. Tomiyama et al. *(1989)* define a model as a theory-based description of an object, see Figure 18. The theory is the filter through which the object is looked upon. Modelling can therefore be defined as a process in which observed reality is filtered through a theory.

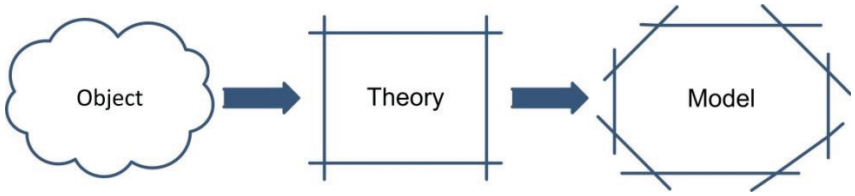


Figure 18 Models defined as theory-based descriptions of an object, redrawn from Mortensen *(1999)*

The relation between model and theory has at least two implications, Mortensen *(1999)*: (1) Modelling has choice of a theory as a prerequisite. (2) Designs have multiple models corresponding to different artefact theories.

Duffy & Andreasen *(1995)* launched the idea of modelling classes, see Figure 19. Design research is about studying a 'reality' or practice of design and creating a model of certain phenomena belonging to that reality. From a phenomenon model it is possible to formulate an information model and implement this in a computer system. The different types of models can all be fed back into practice and influence that practice. Design theory supports the transformations between the modelling classes. Andreasen *(2009)* argues that the most central behavioural characteristic of a design theory is that the theory leads to productive design through the created mind-set of the designer and models, methods and tools; i.e. that theory raises the probability of results and creates a space for solutions.

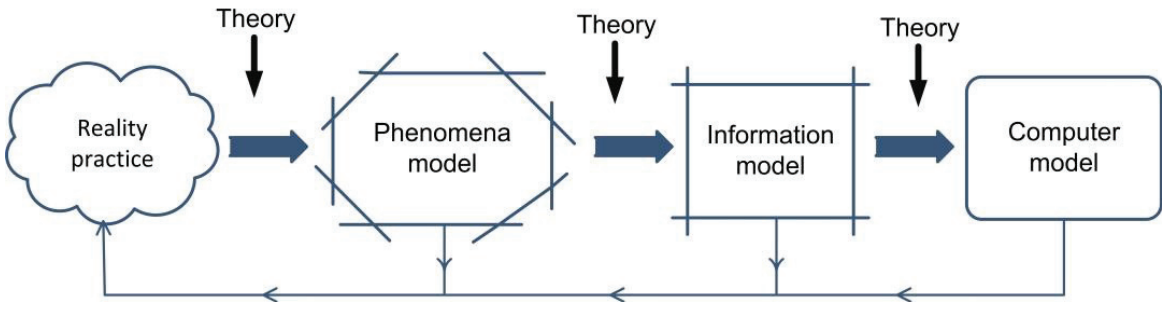


Figure 19 Deriving models from practice for use in practice, redrawn from Duffy & Andreasen *(1995)*

How the concept of modelling classes contributes to this research

The phenomenological understanding of a product architecture can be seen as a prerequisite for modelling it. An model of a product architecture has been suggested in *The Interface diagram.* The model is a part of the phenomena, and the model builds on theory, modelling principles, and understanding of the object of study, the product architecture. The model has been transformed as an information model into a computer system, hence the transformation builds on theory and understanding. The basis for modelling in this research work is the TTS, Hubka & Eder *(1988)*, and Domain theory Andreasen *(1980)*. In addition, the GDMS, Mortensen *(1999)*, and the PFMP and Generic Organ diagram, Harlou *(2006)*, have had influence on the suggested architecture modelling formalisms and framework.

3.2.8 The role of visualisation in models

Several authors report success with the use of some sort of visual modelling as a means to supporting designers and decision makers, e.g. Pedersen *(2009)*, Tjalve *(1976)*, Kvist *(2009)*, Hansen, Hvam & Mortensen *(2011)*, Mortensen et al. *(2012)*, Larkin & Simon *(1987)* . Visual is meant in the way that an overview is easily obtained from the model. They acknowledge the usefulness of visual modelling and graphical representations of products with sketches, drawings, diagrams etc., as a powerful way to hold information and to share knowledge between designers and decision makers. Latour *(1986)*, emphasises and argues for the power of using visual two-dimensional inscriptions of the world because they are mobile, scalable, can be reproduced, recombined, and be superimposed. A common characteristic for the above references is their acknowledgement of the problem of presenting data and information in a visually presentable way without losing the depth of the information. Czarniawska & Mouritsen *(2009)* state that in order to make hard things simple and soft, management technologies are needed, which are defined as mediators allowing users to operate on the material world from a distance. Visual models can be characterised as management technologies, as they are mediators for their users enabling them to work on the objects modelled from a distance and from a certain perspective.

How the visualisation in models contributes to this research

The visual aspect of the proposed architecture models in this research is seen as most important in the process of creating them, modifying them, and using them as a boundary object in meetings. By means of visual models of architectures, professionals from different technical domains and with different backgrounds are able to get the same ‘picture’ of the modelled reality.

3.3 Multi-product development

The following sections will introduce this research's theoretical basis belonging to Multi-product development, i.e. theory of architectures, product platforms, and methods and tools for developing modularity in product systems.

Multi-product development is in itself a big step in the direction of high complexity, but at the same time the concept is today supported by methodologies for reducing mental complexity, e.g. architecture initiatives, product platforms, and modularisation. Modular architectures are in this research perceived to be the basis for identifying platforms, i.e. reusable subsystems and/or modules and interfaces that can be common for a range of products. The commercial exploitation of a product platform is perhaps better known as mass customisation. Mass customisation is originally a manufacturing strategy aiming at satisfying individual customer requirements while keeping manufacturing cost and delivery times close to those of mass-produced products, Pine *(1999)*, Pine, Joseph & Victor *(1993)*. When modular architectures and modules are combined with rationalisation, for instance fitting them to suppliers, manufacturing processes, repair, upgrading, it is possible to see the large effects on costs, time and quality in the company. Several methodologies and models are developed to support the activities of multi-product development based on modular architectures and platforms, where the most relevant for this thesis are: *The Product Family Master Plan,* Harlou *(2006)*, *Design structure matrix,* Hölttä-Otto & de Weck *(2007)*, Pimmler & Eppinger *(1994a)*, and *The Modular Function Deployment* (MFD), Erixon *(1998)*, Ericsson & Erixon *(1999)*. The concepts of modular architectures, platforms, and the methodologies and models for supporting them, are explained in the following by outlining how they contribute to this thesis's theoretical basis.

3.3.1 Theory of architectures

Many definitions of architectures exist in the literature. Some propose that architectures describe a mapping between domains, mainly of a functional and physical kind respectively, Ulrich *(1995)*, Ulrich & Eppinger *(2004)*. Some broaden the perception of mapping to include functional elements, Andreasen, Mortensen & Harlou *(2004)*, Miller *(2001)*, Erens & Verhulst *(1997)*. Others include the manufacturing processes, Hansen, Hvam & Mortensen *(2011)*, Jiao & Tseng *(2000)*, Du, Jiao & Tseng *(2001)*. Finally, some others put emphasis on the decoupling or interdependencies between the various elements in a system rather than mapping between domains, Gershenson, Prasad & Zhang *(2004)*, Baldwin & Clark *(2000)*, Hölttä-Otto et al. *(2014)*, Rahmani & Thomson *(2011)*, Sanchez & Mahoney *(1997)*.

Andreasen, Mortensen & Harlou *(2004)* describe an architecture as a purposefully aligned structure of systems, emphasising that designing an architecture is to align systems with different purposes in mind, i.e. ensuring that systems interact as intended. A classic example is the alignment between the product system and the manufacturing system. The importance of aligning the market aspect and the production setup when developing a new product architecture is also suggested by Hansen et al. *(2012)*, where the alignment is illustrated by the DTU framework for architecture initiatives, see Figure 20. The framework builds upon the classical partitioning of the market, product and production/supply domains, Andreasen & Hein *(1987)*. The elements described in each pyramid can be seen as the behavioural and constitutional elements of an architecture that can be affected by an architecture initiative.

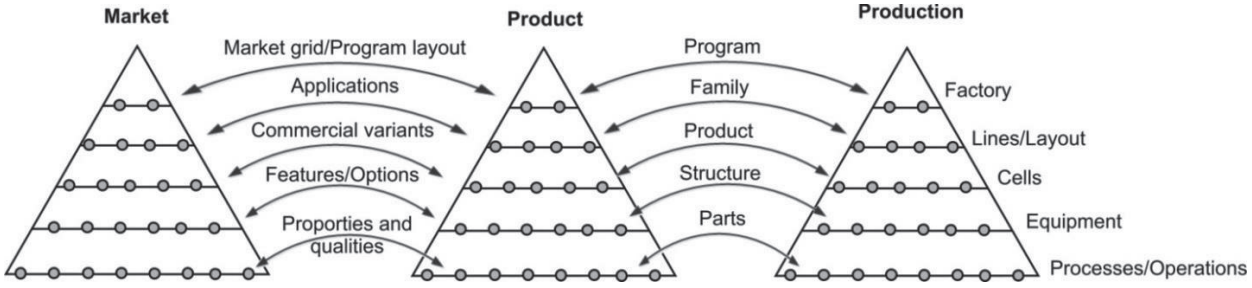


Figure 20 DTU framework for architecture initiatives

An architecture is characterised by being structured after such principles that it becomes for instance modular, integral, part modular, or product family oriented. To explain modularity, one must distinguish between modular and integral product structures where the distinction is based on the correlation of elements in the product, Ulrich & Tung *(1991)*. A modular structure is a structure consisting of self-containing, structural elements with standardised interfaces in accordance with a system definition. A product structure will rarely be classified as either completely modular or completely integral and the structure of a product should thus be classified as being more or less modular or integral.

A module is a composed element of a part structure characterised by realising a single function or a predominant function among a limited set of functions, and characterised by having such specific (standardised) interfaces that the module becomes interchangeable with other modules, Baldwin & Clark *(2000)*. Modular architectures can be perceived as a way of reducing structural complexity, Miller *(2001)*. Structural complexity can be defined as the number of interactions between components in a system, and a system can be perceived as complex when it consists of many closely connected parts with interdependencies. This again makes it difficult to decompose and decouple the systems into structural modules. Modularisation is in this context perceived as arranging components in such a way that relations are within modules rather than among modules.

How theory of architectures contributes to this research

An architecture is in this research treated as a description of how a product, product family or product program is partitioned into subsystems and/or modules and components, and how functionality is allocated to them respectively. Representations of architectures are fundamental to this research, and the theory of architectures and systems theory are extensively used when formulating the modelling formalisms presented. The important aspect of addressing both structural elements and behavioural effects, i.e. functional performance and dispositions in life phase meetings, in architectures are central when using architecture models for supporting the synthesis of a product system. Finally, the phenomenon of aligning different architectures of market, product, and production, contributes to the understanding and fulfilling of business goals, e.g. by balancing internal and external demands.

3.3.2 Product platform

The concept of a product platform has been defined by many researchers, especially during the last two decades. Two of the most cited are Meyer & Lehnerd *(1997)* and their definition, p.4: "*A product platform is a set of subsystems and interfaces that form a common structure from which a stream of derivate products can be efficiently developed and produced".* A product platform is yet again perceived as the basis on which a product family is developed. A product family from an engineering point of view refers to a group of related or similar products derived from a product platform to satisfy a variety of market niches, Simpson, Maier & Mistree *(2001)*. Each individual product within a product family, i.e., a family member, is called a product variant or instance. While a product family targets a certain market segment, each product variant is developed to address a specific subset of customer needs of the market segment, Meyer & Lehnerd *(1997)*. Robertson & Ulrich *(1998)* expand the general concept of platforms to the collection of assets that

are shared by a set of products. These assets can be divided into four categories: Components, processes, knowledge, and people and relationships.

Commonality and variety are important aspects when describing the complexity of a product family and to understand the entitlement of product platforms, see Figure 21. Variety is the diversity of the product assortment seen from a customer's point of view, whereas commonality is the assets that are carried over between variants on the assortment.

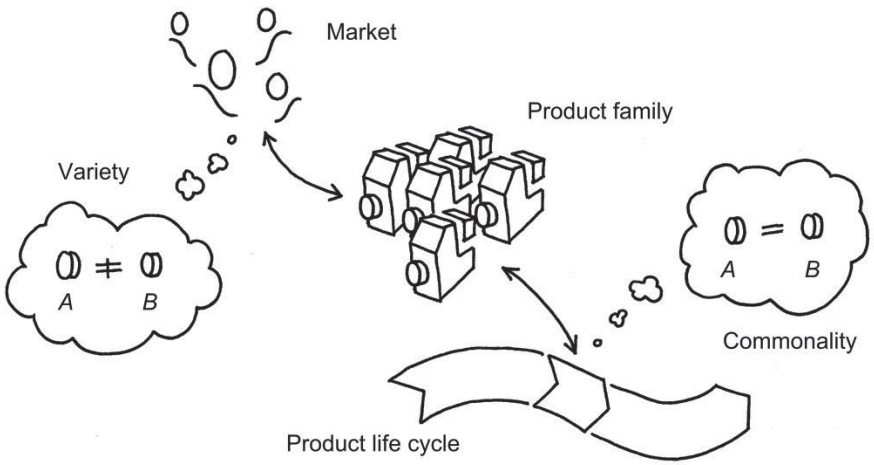


Figure 21 Internal commonality and external variety in a product family, adapted from Andreasen, Mortensen & Harlou *(2004)*.

A product family or assortment should provide variety towards the market to ensure that the customer gets the right product. Platform thinking is the process of identifying and exploiting common features/similarities among a company's products, target markets, and the processes for creating and delivering products. In other words it can generally be stated that companies should aim to have commonality in the company and bring variety to the market, Miller *(2001)*.

How the concepts of product platforms contribute to this research

The scope of the architecture approaches in this thesis is complex product systems, with a focus on architectures that are common for a range of products, e.g. product families. A product platform is in this thesis perceived as the subsystems and/or components, and their interfaces that are common for a range of products. The phenomenon of a product platform thus contributes to the prescriptive part of this research by providing methodology support for identifying subsystems and/or components, and interfaces that can be common for a range of products.

3.3.3 The Product Family Master Plan

The product Family Master Plan (PFMP), Harlou *(2006)*, is a visual object oriented modelling approach. The basic idea behind the PFMP is to gather large quantities of information in a visual way on a poster in order to present an overview. It makes it possible to gather information of several product variants in one model without requiring large sets of redundant data. A PFMP introduces three views upon a product family: *Customer view,* describing a product family as seen from the customer's point of view; *Engineering view,* describing the product family from an engineering point of view (similar to an organ view); and *Part view,* describing the product family from a physical point of view, i.e. explaining how the physical entities (parts and assemblies) are structured and how they vary. Causal relations between the views can be modelled to identify how a feature is realised (links from customer view to engineering view and to part view) and how a part creates functionality and adds value to a customer (links from part view to engineering view and customer view).

How the PFMP contributes to this research

The Product Family Master Plan has its basis in The Theory of Domains, and holds elements of the framework for Integrated Product Development. The framework has been tested in a range of industrial projects for modelling product families. Application of the tool demonstrates modelling capabilities of product families similar to the framework model described in Paper B. Moreover, the understanding of causal relations between the engineering/organ view and part view in the PFMP, contributes to the idea of relations between the systems and module views of the Interface Diagram, see Papers A, C, D, and E.

3.3.4 Design Structure Matrix

The Design Structure Matrix methodology takes a starting point in the decomposition of a product into components/systems and an identification of interfaces/relations among these, Pimmler & Eppinger *(1994b)*, Hölttä-Otto & de Weck *(2007)*, Lindemann, Maurer & Braun *(2009)*. Subsystems are mapped against each other for correlation purposes, in order to cluster subsystems that are closely interrelated and separate those that are not. By the use of algorithms it is possible to encapsulate components into modules or chunks that are closely related to each other from an interaction point of view, Steward *(1981)*. This process is referred to as clustering. The Design Structure Matrix is not a particularly visual model, but it has the ability to hold a large amount of information for complex systems and make it available for analytical purposes. The outcome of a DSM can be a proposal for a future modular product architecture by pointing at module candidates.

How DSM contributes to this research

The DSM methodology contributes to this research with a mind-set related to understanding the importance of interfaces and interactions among subsystems, modules, and components, which is fundamental in this research. Furthermore, the DSM's support in decomposition and composition has similarities with the suggested approach of The Interface Diagram, as the creation of modular simple structures in a DSM is comparable with the creation of modularity in the diagram.

3.3.5 Modular Function Deployment

The Modular Function Deployment (MFD) methodology builds on the Quality Function Deployment (QFD) method and on the formulation of eight so-called module drivers, p. 72, Erixon *(1998)*. The purpose of MFD is to enable cross functional teams to create a mapping from the physical structure of the products within a family to the functional structure of those products and to ensure that the functional structure corresponds to the demands of the customers. MFD consists of five consecutive steps. Customer requirements are mapped to functional criteria and sub-system design characteristics, and subsequently form a physical design in which a modular architecture supports a selected set of modularisation incentives called module drivers. Module drivers are formulated as modularisation incentives, i.e. different reasons to modularise a product family. Some drivers are more related than other drivers and each will have different implications on the product design. The fourth and fifth steps are rather traditional product development steps in which concepts are evaluated and modules refined.

How MFD contributes to this research

MFD and its concept of module drivers is an approach for identifying modules that has its starting point in customer requirements. The MFD approach touches some of the central parts of the identification of modules, and the formulation of module drivers is used in this thesis to point out incentives for creating modules.

3.4 Information management and computer-support

3.4.1 Information management theory

Even simple product designs are defined by an extensive set of information. The process of creating, refining, and releasing this information involves many different employees, roles, and organisational levels in the company. The complexity of handling information increases in an environment of concurrent engineering of multiple products in many variants. Different revisions and versions of information have to be managed effectively. Information management is a means for dealing with these challenges of information complexity. Svensson *(2003)* splits the process of taking care of information throughout the product development process into four different views of *Engineering Information Management* (EIM), see Figure 22.

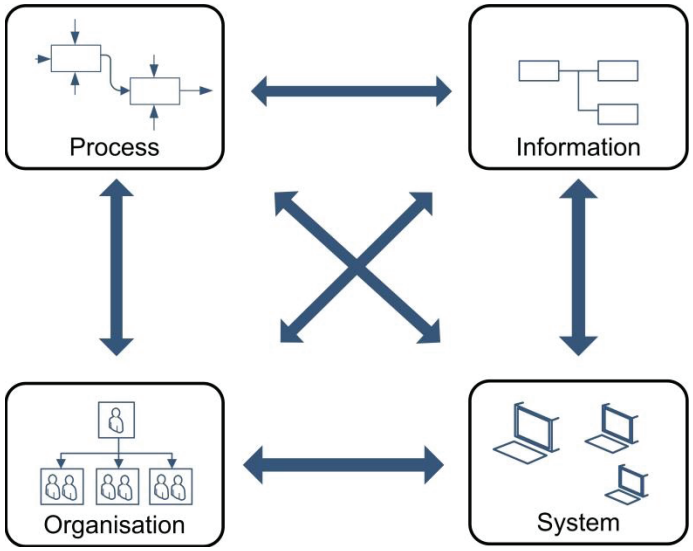


Figure 22 The four views of Engineering Information Management, redrawn from Svensson *(2003)*

The views of EIM contribute to its purpose, and none of them should be approached as a discrete phenomenon, illustrated by the arrows in the figure, when managing information. The process view refers to how work-tasks are structured in the development process. The information view refers to information representations of the product design and their internal relations. The organisation view addresses human resources and assignment of responsibility in the development process. Finally, the system view refers to computer tools intended for information management in the development process.

How information management theory contributes to this research

The research aims to create support for architecture information management in the development process, and information management theory can be used for understanding the broad activity dimensions of processes, information, organisations, and system tools, that have to be considered when developing a company's information management system.

3.4.2 Information modelling

The information needed to capture the work during product development includes various kinds of structures and design documents. A prerequisite for models of information is their ability to specify data semantics for a chosen domain of interest, Mortensen *(1999)*. In general, an information model specifies relations between kinds of information, but could also include relations between individual parts of information. An information model is based on information theory, e.g. object oriented modelling or entity relation modelling formalisms.

One formalism for information modelling that has become widely applied is *Unified Modelling Language* (UML). UML illustrates items, attributes, and relations, Bergsjö, Catic & Malmqvist *(2008)*. UML is an object oriented modelling language developed and standardised by the Object Management Group (OMG) in 1997. UML has its origin in software intensive projects, but today it addresses many types of modelling purposes. In UML 2.0 there exist a large number of diagrams for different types of information modelling tasks, OMG *(2014)*. The most common one for information modelling is called the class diagram, a formalism also applied in this research. SysML is a further development of the UML standard, being less software-focused and better managing systems engineering information. Despite the limitations of UML, it is still possible to model complex relations involving both software and hardware in UML diagrams, OMG *(2014)*.

How information modelling contributes to this research

Many specific modelling languages exist like; mathematics, logic, Simulink, Systems Theory, IDEF0, Entity Relation modelling, etc., Mortensen *(1999)*. Information modelling languages do not explain what to model in designs and products. Consequently, information models have to be based on theories explaining the nature of what is being modelled, i.e. phenomena theories, see Figure 19. Information modelling languages like UML and Entity relation modelling contribute to this research as a means for establishing and communicating a data model in a PLM system.

3.4.3 Information management in product development

Product Data Management (PDM) systems are computer systems which are used to handle product data. The functionalities of enterprise PDM tools have evolved in the last decades and today PDM is in many cases seen as a subset of Product Lifecycle Management (PLM) system tools, Sääksvuori & Immonen *(2008)*. Product Data Management (PDM) technology is intensively used in industry and today its application is mainly focused on particular product lifecycle phases, e.g. development, prototyping or production, Abramovici *(2007)*, Stark *(2011)*, Sääksvuori & Immonen *(2008)*. The user functions of a PDM system are:

- **Data/information vault:** Documents are stored in an organised way. Information about the documents (meta-data) is stored in a central database.
- **Document management**: Procedures for managing releases and changes of documents.
- **Workflow and process management:** Rules and procedures for processes can be monitored and controlled by a PDM system.
- **Product Structure Management:** Product structures, e.g. BOM, are defined and change controlled during their lifecycle. Some PDM systems can also facilitate the management of product configurations.
- **Classification management:** Standard parts and/or modules can be classified to support re-use.
- **Project Management:** Tracking the progress of a project.

Standard Configuration Management (CM) is a part of some PDM/PLM systems today, which means that versions and revisions can be base-lined in order to draw information from consistent data in the system, International Council on Systems Engineering & Haskins *(2011)*.

In recent years, PDM vendors and integrators have had a multitude of acronyms; in reality, acronyms and descriptions congregate in Product Lifecycle Management (PLM), Abramovici *(2007)*. PLM is a comprehensive approach for product-related information and its management within an enterprise. PLM refers to the management of the product definitions, e.g. product data within product development, but in essence product data of the whole lifecycle of the product. This includes planning and controlling of

processes that are required for managing data, documents and enterprise resources throughout the entire product lifecycle, Abramovici *(2007)*.

Computer Aided Design (CAD) systems are today integrated into most PDM/PLM applications, where CAD models and drawings are managed in terms of user rights, version control, and change management. CAD is the generic name for all computer-based tools using interactive graphic techniques for representing mechanical and electrical designs, Stark *(2007)*. Representations can be either 2D or 3D and many CAD systems today have additional features to aid in engineering analysis tasks. They include Finite Element Analysis (FEA), Computational Fluid Dynamics (CFD), Multibody dynamics (MBD), and optimisation. All information produced in the CAD feature systems can normally be controlled in a PDM/PLM system in terms of variants, versions, revisions, and user restriction management, Sääksvuori & Immonen *(2008)*.

Some PDM/PLM systems hold functionalities for product configuration. Product configurators are IT based expert systems that support the users in the specification of customised products by providing design choices for the user, while restricting how different elements and their properties may be combined, Haug, Hvam & Mortensen *(2011)*. The result is that product specification tasks, which normally require human experts, can be automated. Examples of applications of configurators are: automating the creation of quotation prices, sales prices, bill of materials, and other product specifications. Configuration of products, which are traditionally mass-produced, implies very little complexity of the knowledge base of the configurator compared to configurators aimed at engineered products, which can include thousands of rules for how elements and properties may be combined, Hvam, Mortensen & Riis *(2007)*.

How information management in product development contributes to this research

This research seeks to take advantage of features in standard PDM/PLM tools for representing architectures and thereby enabling related information to be directly linked to a product architecture model. The extensive amount of information necessary for defining and representing a product architecture is preferably handled in an IT system, as manual management is an overwhelming task. This research provides evidence of the usability of utilising a standard PLM system for the task of managing information related to architectures. By linking CAD information (models) to a product architecture, different views can be imposed and visually represented for assisting the designers in the synthesis activities. Product configurability logic in the utilised PLM system enables configuration of product variants (modelled in the product architecture).

3.4.4 Information management in manufacturing

Competing systems to PDM/PLM for information management, but with a focus on the manufacturing side of the development process, represent Enterprise Resource Planning (ERP) systems. ERP systems are designed to manage production facilities orders, production plans, and inventories, Shaul & Tauber *(2013)*. ERP systems integrate inventory data with financial, sales, and human resources data, allowing organisations to price their products, produce financial statements, and manage resources of people, materials, and money.

How information management in manufacturing contributes to this research

ERP systems have not been the focus for representing architectures in this research. However, in the identification of a product architecture it is necessary to use insight of the related manufacturing system. Information from ERP systems has been used when insight of the supply chain was needed in the product architecture identification process.

3.4.5 Information management in sales

Customer Relationship Management (CRM) tools are accepted as an effective approach for collecting, analysing, and translating valuable customer information into managerial action, Kumar *(2010)*. CRM tools hold models for managing a company's interactions with primarily current, but also future customers. It involves using CRM tools to organise, automate, and synchronise sales, marketing, customer service, and

technical support, Kratochvil & Carson *(2005)*. Most of the existing work on CRM focuses only on relationship management for existing products and examines strategies for improving their commercial performance, but misses the possibility of feeding the voice of customers to form future performance requirements, Ernst et al. *(2011)*. Part of an effective CRM process is to manage requirements coming from the market side. In combination with other internal and external requirements, customer requirements are today supported by Requirements management (RM) tools. RM tools handle qualitative and quantitative users-, business-, technical-, functional-, and process requirements for a product, Stark *(2007)*. RM tools are either specifically intended for managing requirements only, or to support the entire systems engineering process, International Council on Systems Engineering & Haskins *(2011)*. A key characteristic of RM tools is the granularity of the information that it manages. This granularity determines the level of traceability that can be supported. The granularity can for a given information type be either *object granularity* or *document granularity*, Malmqvist *(2001)*, i.e. object based meaning requirement defined by objects in an RM structure, or document based meaning text documents in a document structure.

How information management in sales contributes to this research

CRM and RM systems have been consulted when identifying and establishing requirements of a product architecture. The two systems and their information management options are however not central to the described approaches in this thesis.

3.5 Concluding the theoretical basis

The theoretical basis of systems theory in engineering design contributes to this research in an important way, by representing the fundamental guidance regarding the research aim and objectives, as well as the research approach. The theories described in this part of the thesis are all considered as a necessary foundation for research in the area of architecture representation in computer systems. The Theory of Technical Systems, The Theory of Domains and the derived contributions of the Chromosome model and GDMS, form an understanding of what a product is perceived to be in this research. The Theory of Dispositions' influence is widespread in this research, in the sense that architecture representations are seen as a means for balancing and optimising structures to meet requirements from its life phases. Certain aspects of computer-based architecture representations already exist in companies, but are often prevalent in different IT systems. In order to determine the 'goodness' of a product , a product family or a product program, the links between the systems have to be described. The theories belonging to information management and computer support in engineering design are all regarded as important when prescribing support for representing architectures in a PLM system.

4 Research and results

The aim of part 4 is to present the research and its results. Corresponding to the research questions, the results of the research work are divided into two main groups: (I) identify and model structural and behavioural aspects of product architectures, and (II) architecture representations handled in a computer-based information management system (PLM).

4.1 Introduction

This research and its achieved results are presented as contributions captured in five scientific papers. The papers are labelled A, B, C, D, and E. The papers are presented in a uniform way by outlining: their title, their media of publication, their case study number, which research questions are answered in them, their contribution, the applied research method, and additional reflections on their contribution.

The papers are divided into two main areas:

- *Identify and model structural and behavioural aspects of product architectures (Papers A, B)*
- *Architecture representations handled in a PLM system (Papers C, D, E)*

Papers A and B belong to the part of the research focusing on identifying and modelling structural and behavioural aspects of product architectures. Papers A and B handle different case studies. Paper A introduces a modelling formalism, *The Interface diagram*, which again is the basis for handling architectures in a PLM system, see Papers C, D and E. Paper B includes the presentation of a step-wise approach for identifying an architecture for a product and production setup of highly integrated product structures of complex parts. Papers C and D belong to the part of the research focusing on architecture representations in a PLM system. They make the same contribution, and show progress in the area of using PLM systems for handling architecture models for product information management and for offering support to access the design in the development process. Paper E handles an individual contribution of utilising a PLM system for mapping requirements to a product architecture.

A simplified graphical overview of the papers and how they are approached is provided in Figure 23. The two main research areas are investigated and the described theoretical basis forms the foundation of the papers. Not all illustrated theoretical areas are applied in all papers, e.g. Paper B does not address the PLM aspect of this research.

The central papers of this thesis are the three journal papers, Papers A, B and D.

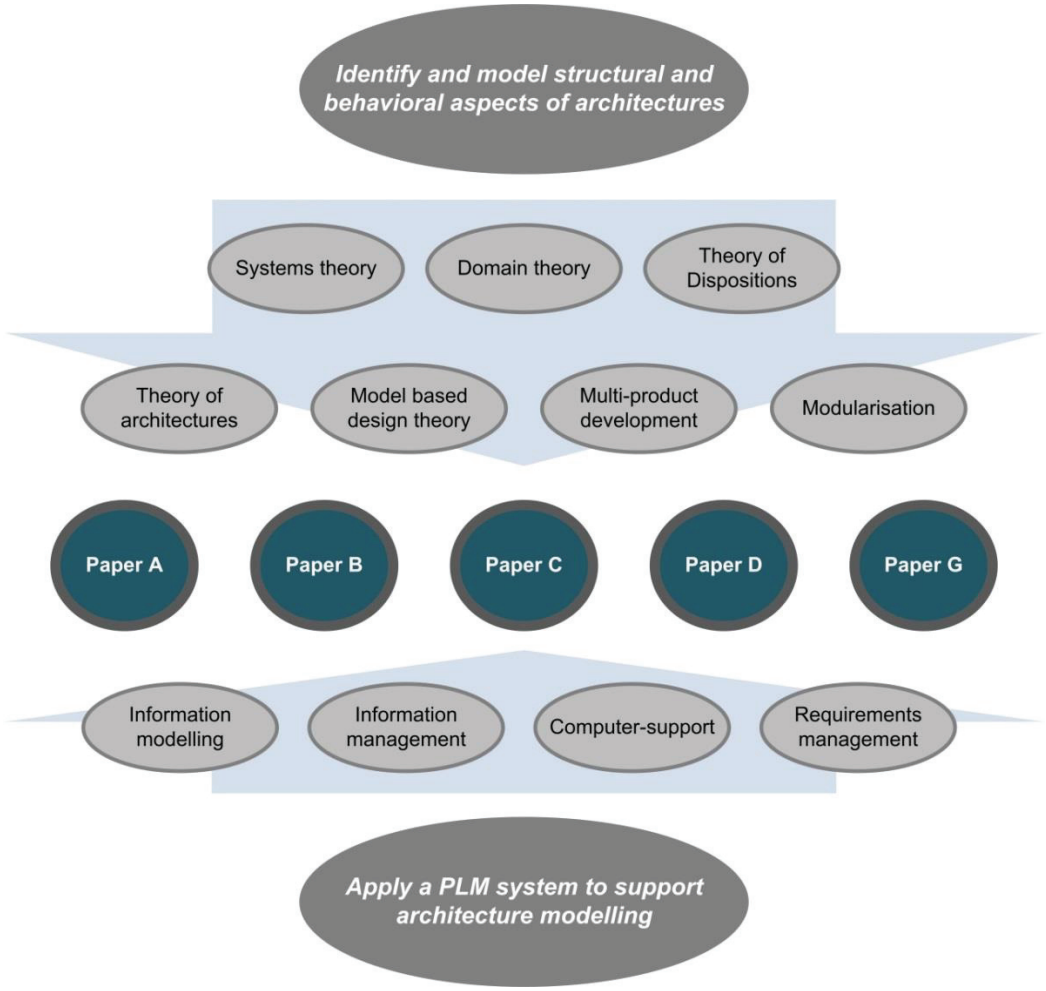


Figure 23 The papers are approached from two areas and supported by the theoretical basis

4.2 Identify and model structural and behavioural aspects of product architectures

4.2.1 Paper A

Title: "*Interface diagram: Design tool for supporting the development of modularity in complex product systems*"

Journal: Concurrent Engineering - Research and Application (*Published*)

Contributor: First author

Case study: for # 1,2,3,4,**5** (Case study 5 is the primary study. The other studies have for this paper served building knowledge in the area of architecture identification and modelling aspects), see Section 2.3.2.

Research question

The research questions for Paper A are RQ1 and RQ2: *What phenomena related to modelling of product architectures, from a structural and behavioural point of view, pose a challenge to designers when developing product systems?* And: *How can both structural and behavioural aspects be addressed in integrated architecture models?*

Research contribution

The contribution presented in Paper A covers a product architecture tool for modelling structural elements (organs and/or parts) of architectures. The diagram follows the approach of modelling architectures by use of block diagrams, see Figure 24.

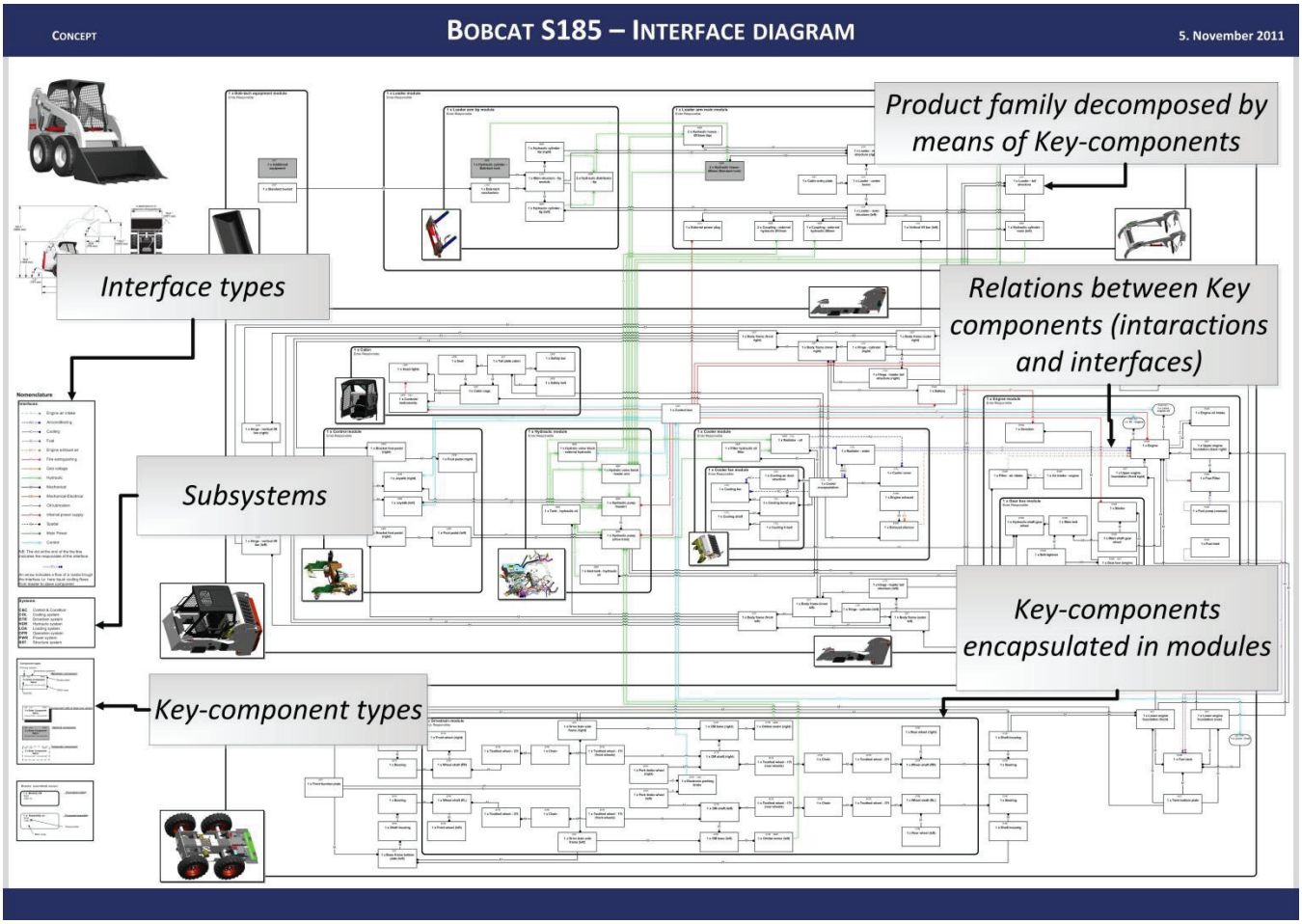


Figure 24 Interface Diagram of a family of power loaders (Bobcats)

The main elements of the diagram formalism are structural elements (organs and/or parts) denoted *Key components*. The structural elements can be arranged in layers of the model according to a *Systems view* or a *Module view*, see Figure 25. Key components and/or modules can be modelled as variants, and by using the layer functionality of the diagram created in MS Visio; different product variants can be modelled in layers. The model provides different professionals with a common modelling language to describe a high level architecture for a product, product family, or product program— enabling designers from different domains to get the same picture of the architecture.

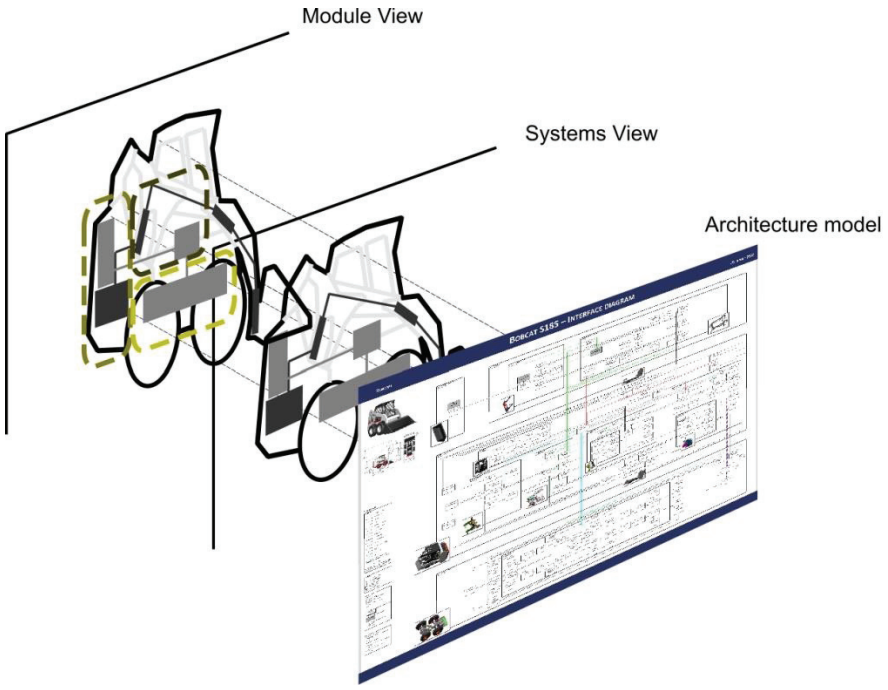


Figure 25 The Interface diagram handles different views on the product

The Systems view shows functional systems, or as stated in the paper, subsystems of the overall product system. Subsystems can be modelled and thereby control important properties of the final product. The Key components are in this view illustrating organs and/or parts, and the relations between them are interactions and/or interfaces. The reason for Key components both representing organs and parts is due to the time aspect in the development process. In the conceptual phase where solutions (given by parts) are still to be found, the Key components address desired functions (the purposeful part of behaviour). In this phase, Key components represent functional elements. When solutions have been determined, Key components represents parts. The differences between organs/parts and interactions/interfaces are:

- An organ is a functional element (function carrier) of a product's organ structure based upon a working principle, which brings together a physical effect and form and material characteristics, able to create the effect (function).
- A part is a materialised entity of a product. (Mechanical) parts are characterised by form, material, dimensions surface quality and state. A part may contribute to more organs or be a contribution to a single organ.
- Interactions are of a functional kind, i.e. being the transfer of material, energy (forces, movement), and information. The interaction may be transferred via an interface, but can also be 'touch-less' or indirect.
- Interface is a characteristic of relations between parts and/or modules which comprises the part structure. The interface between two elements of the structure may be a connection, support or a moveable relation. The interface's characteristics are form, material, dimension, surface quality and state.

The Module view aims to support the engineering process of developing modularity in an architecture. Modules are modelled by arranging Key components and representing structural elements inside boxes with a thick black boundary and rounded corners. The interactions and interfaces between Key components are drawn with lines, their nature is stated in a nomenclature in the model, e.g. information, flow, mechanical, etc. The purpose of working with interactions and interfaces is to ensure responsibility for the

organs and/or parts interaction and to ensure that modules and/or parts are interchangeable, when relevant for creating product variants. The structure of the Interface diagram appears as the relations (interactions and interfaces) are added to the diagram.

A formalism template of the architecture model is shown in Figure 26 and Paper A. Names can be replaced with meaningful ones, e.g. a module could be *Drivetrain,* a Key component could be *Engine,* and a system could be *Hydraulic system*. The formalism is further explained in the appended paper.

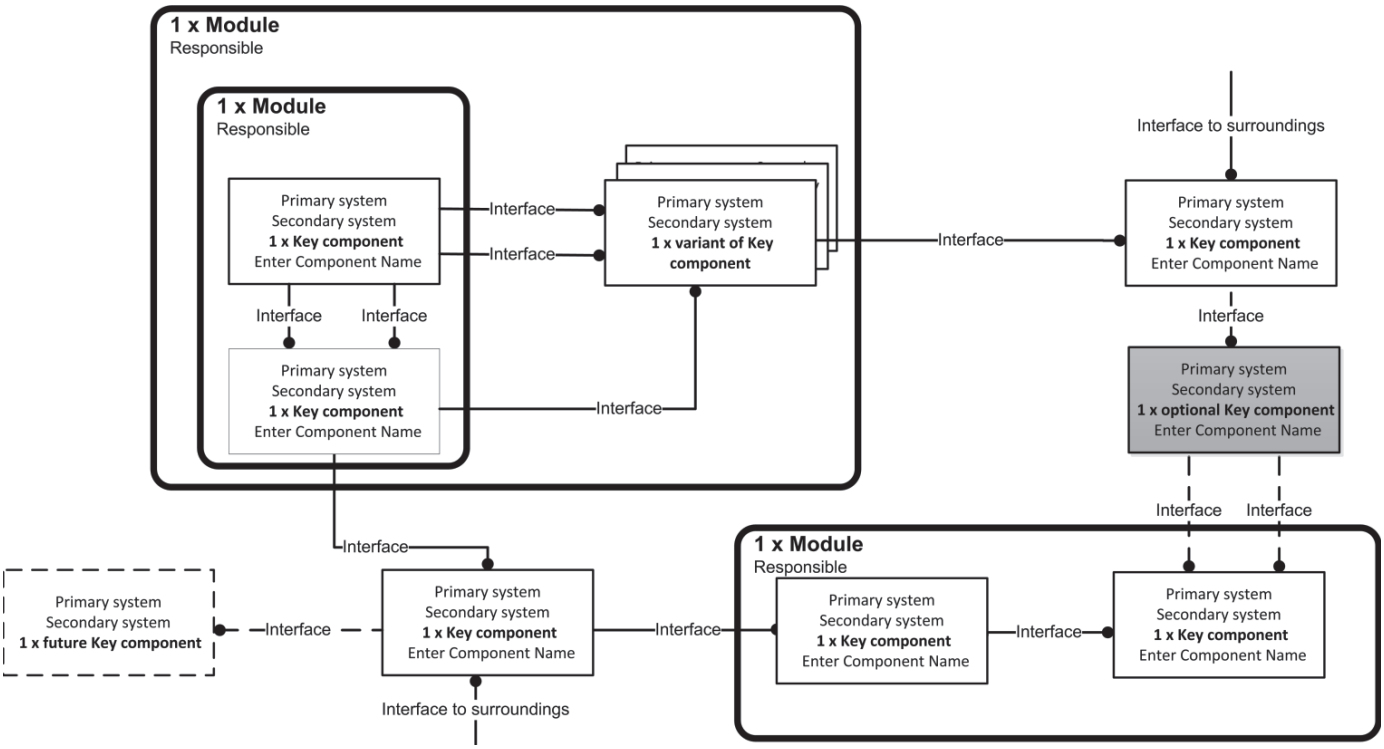


Figure 26 Symbolic representation of an Interface diagram and the types of Key components.

The model deals with transformation of the product over time (from desired functions to parts) and is maintained and updated during the development process, and it offers a fundamental approach for supporting proactive modular architecture development by conceptual modelling the relational aspects of functional subsystems and physical modules. Key components seem to be a pragmatic way of modelling both organs and parts, and by shifting between Systems and Module view; it is supporting the designer in the conceptual activity of converting desired functions to solutions given by parts (components). When working with the diagram, the different views of systems and modules can be monitored in MS Visio and printed on large posters, in order to support the focus on specific design elements of the architecture for designers. Moreover, explicit modelling of subsystems, modules, and interfaces is suitable for splitting responsibility in the product development process, which is also defined in the model.

Paper A includes the initial concept for using the architecture model in interaction with a computer-support system (PLM). This approach is elaborated, detailed, and tested, and then described in Papers C and D.

Research method

Paper A has been developed based on both literature and experience from all case studies, but with a focus on Case study #5. The paper contains descriptions, observations, and lessons learned from case study #5 in which the authors tested the modelling tool to represent a product architecture in a larger Danish company. The paper supports the fundamental viewpoint of Theories of Technical Systems. In this framing, the paper addresses the engineering activity of developing products supported by product architecture

descriptions. Paper A considers the concrete challenges of modelling and capturing essential structure-behaviour information and thereby supporting designers when developing products based on architectures.

Reflection on contribution

- The contribution in this thesis lies in architecture modelling. The modelling discipline is seen as a part of the development process of an architecture. However, it is not the intention of this paper to prescribe a certain stepwise methodology for product architecture development.
- The fundamental idea for identifying and modelling architectures supported by the tool is that an improved understanding of the whole product system and the relation between function and structure, will lead to a better synthesis of results and reduce the risk of missing out on important functionality when developing architectures.
- The Interface diagram is a visual representation of a product's architecture, and thereby is tangibly instantiated, in order for designers to successfully share it and use it.
- The motivation for using the Interface diagram is elaborated in relation to the support of synthesis activities when developing complex product systems by handling decomposition, characterisation and composition.
- A proposal for rules that a module should comply with is included in the paper.
- The paper opens up the concept of transferring the architecture model into a computer-based information management system (PLM system). At the point in time when the paper was created, this functionality was not fully implemented.

4.2.2 Paper B

Title: "*Identification of a Scalable Architecture for Customization of Complex Parts*"

Journal: Concurrent Engineering - Research and Application (In review)

Contributor: Second author

Case study: #1

Research question

The research questions for Paper B are RQ1 and RQ2: *What phenomena related to modelling of product architectures, from a structural and behavioural point of view, pose a challenge to designers when developing product systems?* And: *How can both structural and behavioural aspects be addressed in integrated architecture models?*

Research contribution

The contribution presented in paper B is a framework including a step-wise approach for the identification of an architecture for the production setup. A framework is tailored to support project-based companies doing customisation by engineering of products with highly integrated product structures of complex parts. Those companies are often serving their customers in an OEM setup.

The contribution differentiates itself in its ability to identify and define an architecture for the product and production setup for project-based companies that serve their customers by developing and producing products on a project-basis. The framework focuses on those companies that cannot apply traditional modularisation, as the functionality provided through their highly integrated product design cannot become encapsulated in traditional physical modules separated by interfaces. The paper presents a framework and a step-wise approach for identifying a scalable architecture for customisation of complex parts in order to earn the benefits of modular architectures, without the aid of traditional modularisation. Scalable refers to when even if no physical parts exist, the design is fully scalable within defined ranges.

The paper answers research question RQ2 in the sense that it enables the identification of critical program decisions of an existing product program during its step-wise approach in nine steps, using lead variant designs for focusing on the architecture definition and using the architecture for deriving behavioural effects in the market and production domains. The framework can be used on existing product programs in the sense that it does not require new development of sub-solutions, but a new scaling principle used for the production equipment.

Research method

The framework has been developed based on both literature and experience. The authors' knowledge in doing case studies in project based companies has ensured that the framework has been developed on a thorough basis of experience. Within the Product Architecture Group a number of earlier case studies have been carried out, providing a solid foundation for proposing the framework presented in Figure 27.

Following the classic partitioning from Integrated Product Development, Andreasen & Hein *(1987)*, the framework model follows the split between market, product, and production aspects. The steps are treated from 1 to 9, with iterations in the steps if necessary. Each step contains a modelling formalism as well as a link to other steps - indicated by the red lines. The linking is an important feature in the framework, as it is impossible to achieve attractive cost- and price points without the coordinated development of product and production scaling principles in alignment with the market envelope and requirements from key-customers. See the appended Paper B in Part 7 for a full-scale version of Figure 27.

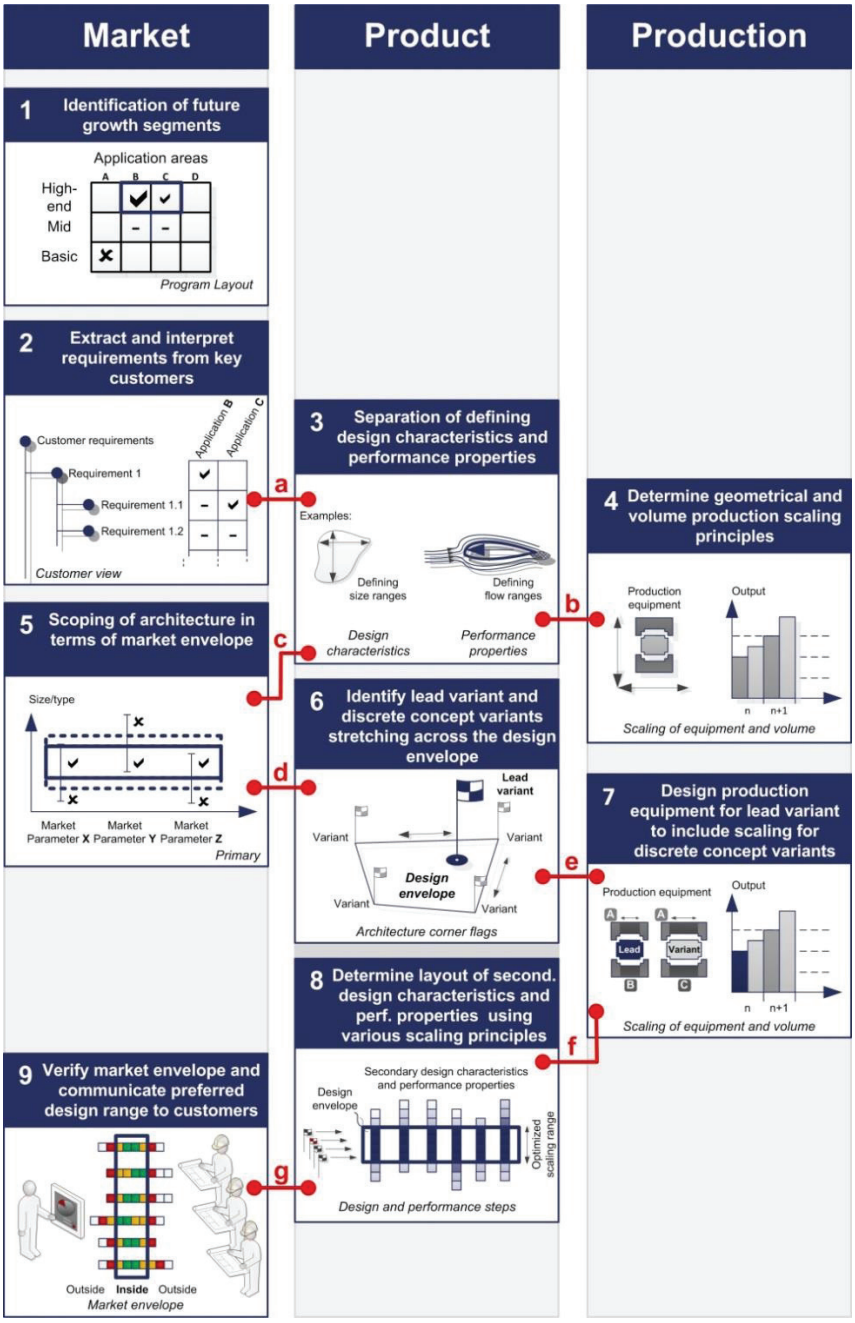


Figure 27 Framework including step-wise approach.

Reflection on contribution

- The framework is especially developed for high performance products where no +3-5% over-engineering can be tolerated.
- The approach can be characterised as a bottom-up approach in relation to the domain of product design, as no compromises on performance are allowed for the targeted high performance products.
- The identification of an architecture for highly integrated complex parts, shows that some basic virtues of traditional modularisation apply also for this type of product. The virtues are that it is possible, even for highly integrated parts, to decide on commonality and variety within a range of products. This does not compromise on the performance of the individual variants.

4.3 Architecture representations handled in a PLM system

4.3.1 Paper C

Title: *"PLM support for development of modular product families"*

Conference: International Conference on Engineering Design, ICED13, Seoul, Korea

Contributor: First author

Case study: #5

Research question

The research questions for Paper C are RQ3, RQ4 and RQ5: *How can PLM systems support architecture models that emerge in steps of the design process?* And: *How can PLM systems be used for assessing product cost during the design process?* And: *How can PLM systems be used for assessing the completeness of a design during the design process?*

Research contribution

The contribution presented in Paper C is about applying a standard PLM system for representing a product architecture model (*The Interface diagram*). Where Paper A focuses on the formalism of the visual architecture model, Paper C focuses on the capability of a PLM system to support identification, development, and representation of an architecture.

The concept of modelling classes, Duffy & Andreasen *(1995)*, is applied for describing the relation between The Interface Diagram and the PLM system, see Figure 28. The figure illustrates the idea that design research is about studying a 'reality' or practice of design and creating a model of certain phenomena belonging to that reality. The Interface diagram is a phenomenon model of an architecture, and it forms the basis for formulating an information model and then implementing it in a computer environment (PLM).

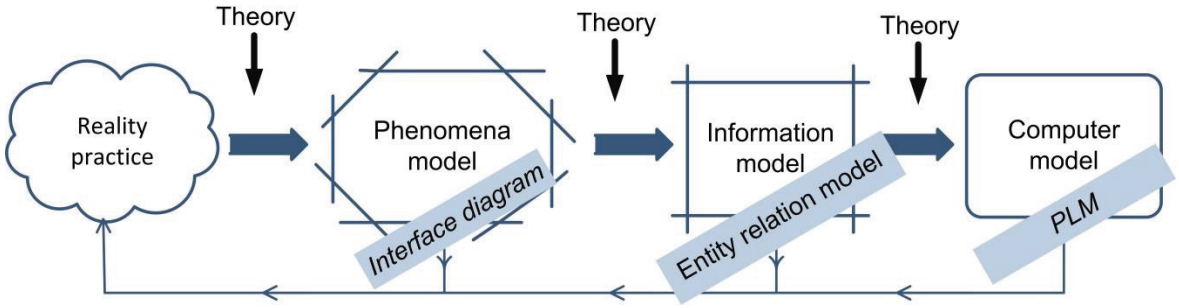


Figure 28 Modelling classes in relation to this research

The architecture model is by a transfer file protocol loaded into a standard PLM system (*PTC Windchill PDMLink 9.1©)*, which enables the assignment of information created in other systems like CAD, ERP, DSM, etc. Information as text documents, sketches and drawings, test protocols, etc. can be assigned to structures defined in the architecture model, i.e. product variants, functional systems (in the Systems View), physical modules (in the Module view), structural elements (Key components in both views), and to interfaces/interactions (in the Interface view). The most important reason for handling the architecture model in a PLM system is that it supports the management of a large amount of information belonging to an architecture, so relevant information for designers can be traced directly from the part of the architecture they are working on.

Paper C presents a description of a step-wise approach of first defining architectural views in a visual model, and then loading the model into a PLM system. Information in the PLM system can be applied for purposes of analysis, specification, test, control, reasoning, etc. The PLM system provides designers with the means for monitoring the different views of the architecture by simply choosing the view of interest in the user interface. Information such as CAD models, property models (strength, thermodynamics, fluid dynamics, etc.), interface control documents, and purchase specifications, can be linked directly to elements (products, systems, modules, Key components, interfaces/interactions) of the architecture model. This in return enables designers to efficiently navigate through the large amount of information belonging to the architecture, see Figure 29, without having redundant information when changing between views of the architecture. The PLM system creates traceability of the information by enabling users to find information belonging to their specific design element, and by using a '*where used*' feature in the system it is possible to trace relations to other parts of the architecture and/or information.

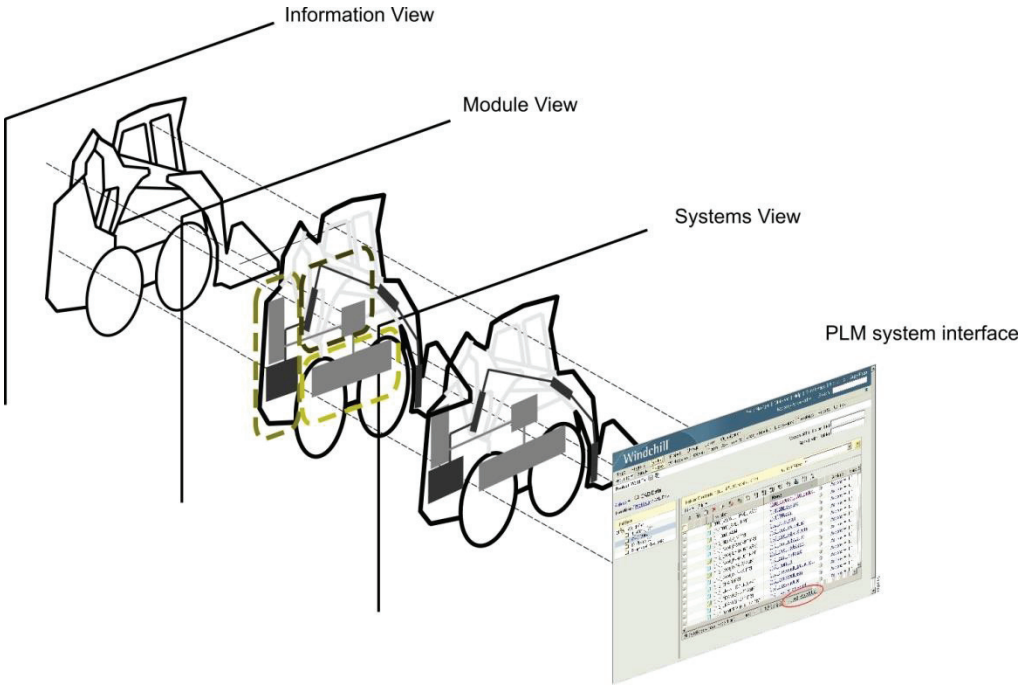


Figure 29 PLM system interface to focus on the architecture's Systems view and Module view and the related information views like requirements, specifications, sketches, CAD, etc.

Every Key component in the system can be allocated an initial target cost attribute. Gradually direct cost (direct material cost or purchase cost) can be added to Key components as their underlying solution, given by parts, is designed or purchased. By means of simple roll up mechanisms costs are added up from the top product level and down to the smallest components. If a cost figure is missing on a component, a target cost is used instead. Target cost numbers are provided by financial controllers in collaboration with the relevant designers and purchasers. Cost reporting is in that way a strong tool for supporting a design according to budget, because the difference between budget and actual cost is discovered quickly. The valuation of actual cost vs. budgeted cost is often referred to as *Earned Value Analysis,* Anbari *(2003).*

Reporting functionality for assessing design status is also established in the PLM system and can be applied for monitoring the status of designs on a weekly basis. The numbers of new, modified, and approved parts are counted, and the numbers can be used as indicators of the progress of the design. This resolution level enables early detection of design conflicts e.g. the risk of not being on time with designs.

The approach of using a PLM system for supporting architecture identification, development, and representation has been tested in Case study #5. The approach, in terms of procedures and technical issues concerning transformation of the visual model to the PLM system, is detailed in Paper D and will not be described again here. The most important results are listed in terms of the experienced effects on the modular design process in the company:

- Having clearly defined systems; an enabler for focusing on system design and converting desired behaviour to solutions given by parts.
- Having clearly defined modules; an enabler for reuse of modules.
- Having clearly defined interfaces; an enabler for reuse of modules.
- More accurate cost reporting due to automated reporting.
- More accurate design progress reporting due to automated reporting.
- Multiple CAD structures: Visualisation of products from different perspectives is also seen as a strength in the conceptual phase in order to see implications of structure concepts, and when focusing on integration of different systems.

Research method

The paper addresses the engineering challenges of developing products supported by architecture representations in interaction with PLM systems. The paper builds on literature studies and a participatory case study, Case study #5, which is the central empirical study in this research.

Reflection on contribution

- The approach is seen as valid for the development of complex product systems, e.g. a product family with a large number of parts, and to a lesser degree as an approach to choose when developing simple products with few components and simple technology.
- Information is not assigned to the architecture automatically. Consequently, every piece of information has to be manually assigned to the relevant elements of the architecture model. In the case company an Architecture/PLM team was established for supporting and training designers in assigning and maintaining information related to the architecture.
- The paper describes the approach being implemented in an industrial setting and the first results by doing this have been described. More details of the approach and elaborations on the results are described in Paper D.

4.3.2 Paper D

Title: "*PLM system for development of modular product development*"

Journal: Computers in Industry (in second review)

Contributor: First author

Case study: #5

Research question

The research questions for Paper D are RQ3, RQ4 and RQ 5: *How can PLM systems support architecture models that emerge in steps of the design process? How can PLM systems be used for assessing product cost during the design process?* And: *How can PLM systems be used for assessing the completeness of a design during the design process?*

Research contribution

The contribution presented in Paper D is an extension of the descriptions presented in Paper A and C, and adds new elements to their contributions. The paper unites the former papers by describing in detail the applications of a visual architecture model to create overview, improve communication and collaboration, support the conversion from desired behaviour (functions) to solutions given by components, and to support the creation of modular product structures. The model interchanges with a PLM system to manage and integrate product information belonging to the architecture effectively. The results from the study encompass PLM capabilities for handling multiple architectural views, controlling interfaces, and quantifying and communicating the status and progress of product cost and design completeness.

The paper holds a refined Entity relation model of the information structure. Information defined inside the PLM system, or in associated computer systems such as CAD or text documents, are denoted *Lower Structures* and structures of the architecture are denoted *Upper structures,* see Figure 30. The objects are defined and explained in detail inside the paper.

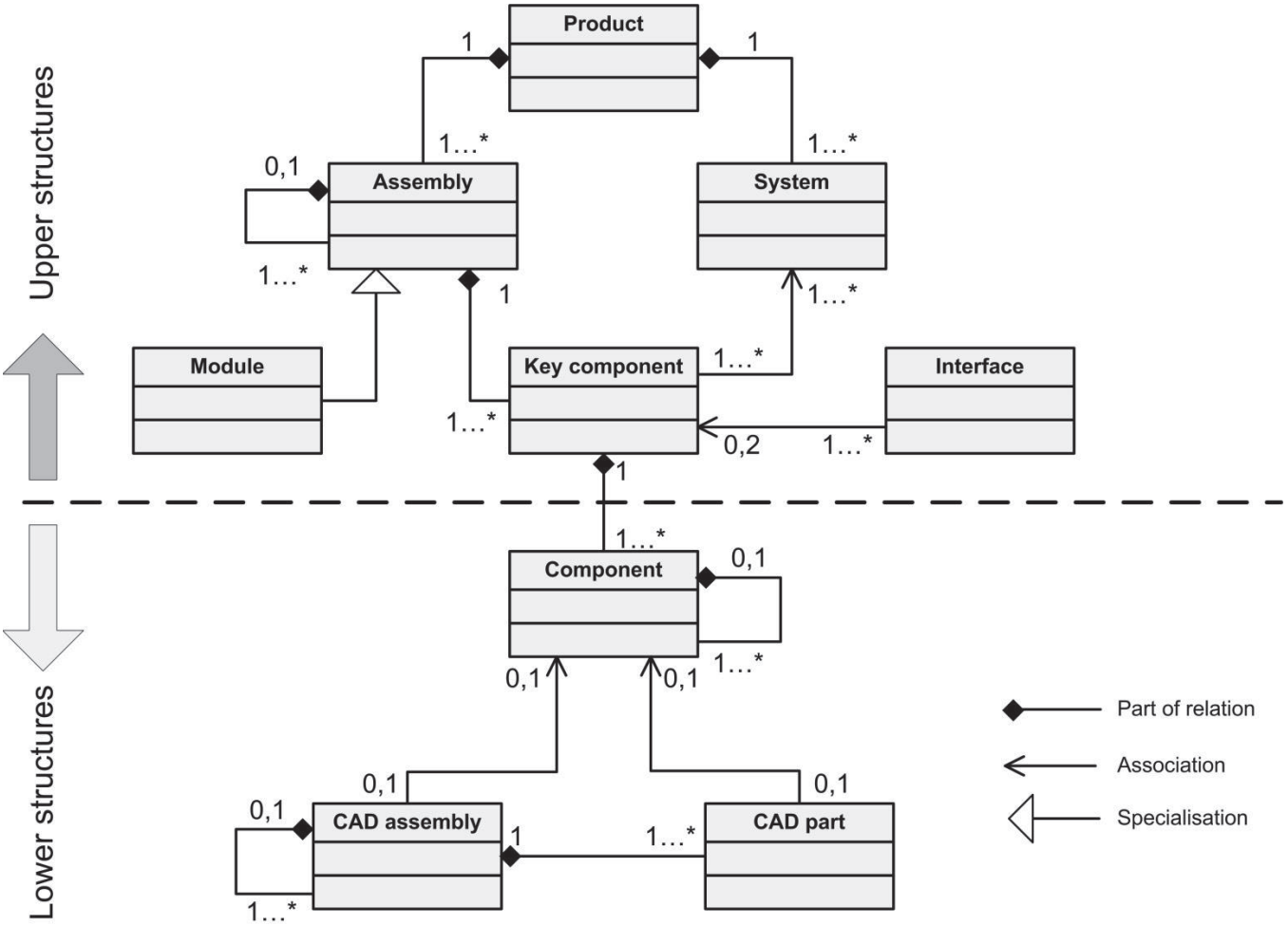


Figure 30 Entity relation model of the information structure in the PLM system. Only CAD information is illustrated in the information model, yet, other information can be linked to the elements in the Upper structures.

The implementation of the approach of using a visual architecture description in combination with a PLM system showed some promising qualitative results, as reported in the paper. The PLM system provides continuous support in the development process, as there is no fundamental difference between 'early' and 'late' phases, because of the pragmatic definition of Key components being organs and/or parts.

The Systems view handles the architecture from a functional perspective and the Module view from a physical perspective, see Figure 31. Based on behavioural requirements and identified functions, the Systems view is modelled by Key components (organs and/or parts) and their relations. Solutions that realise the intended behaviour are determined by product synthesis, and modularity of the architecture in relation to different module incentives is gradually determined and modelled in the Module view. Key components are in this view representing parts, and their relations are of an interface nature. The modelled structure in the Module view can then be applied for predicting whether the intended behaviour is fulfilled by focusing on the functional Systems view. Synthesis, analysis, and evaluation are supported by additional models, tests, and simulations.

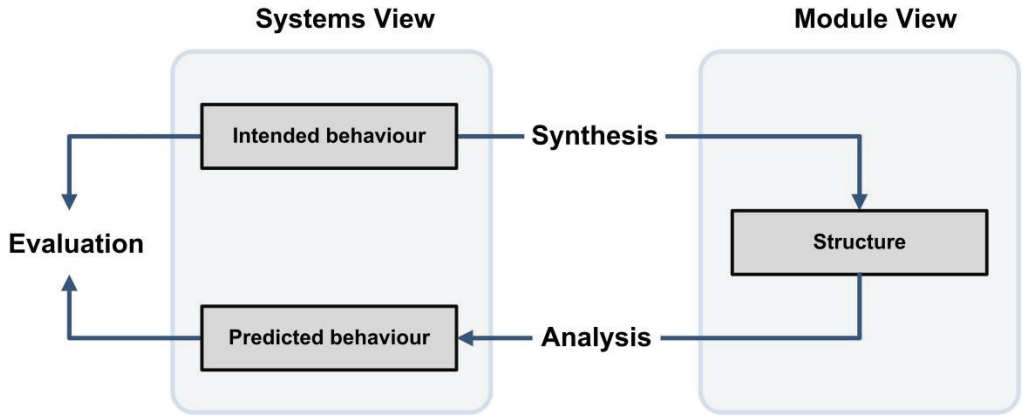


Figure 31 Applying the architecture model for synthesis, analysis, evaluation of the relation between predicted and intended behaviour, based on Jensen *(1999)*, **p.22, *General tasks of the design process*.**

By interviewing the members of the management team and the 14 different design teams of the case company, the following statements could be reported as results of the implemented approach, model, and tool:

- Management
    - Design readiness much clearer earlier in the projects.
    - More transparent cost deviations.
    - Design progress more accurate to measure on a weekly basis.
- Designers
    - Very little extra work when thinking in architectures for a family compared to single product development.
    - All engineering teams have access to the same single source of information.
    - Easier to develop and evaluate module concepts in the early design phases.
    - More effective reviews.

Research method

Paper D is the last paper describing the PLM system capabilities for representing product architectures developed in this research project. Thus, it has had access to the latest findings by implementing the approach in the case company, case study #5.

Reflection on contribution

- The main contribution of this research is the elaboration of the distinction between structural and functional contents of architectures, and the ability to handle this distinction when using them as a basis for development. By actively using this distinction it is possible to improve modular design, as modularity is based on both characteristics and properties of products.
- The implemented approach has shown that benefits are especially dominant in the related CAD system, to represent CAD models in both a functional and a physical modular grouping.
- The approach needs to include systematic procedures on how to link and maintain information in the PLM system. During development Key components are created in the conceptual phase and perhaps later deleted if a structure concept is abandoned. Information linked to a Key component that is going to be deleted has to be evaluated: link to other elements of the architecture or seen as obsolete and deleted? The approach of linking and re-linking is today mainly based on manual procedures but research in doing them more automatically is ongoing.

4.3.3 Paper E

Title: "*Mapping requirements to a product architecture supported by a PLM system"*

Conference: Design Conference 2014, DESIGN14, Dubrovnik, Croatia.

Contributor: First author

Case study: Conceptual research study

Research question

The research question for Paper E is RQ3: *How can PLM systems support architecture models that emerge in steps of the design process?*

Research contribution

The contribution presented in Paper E is a conceptual approach for supporting the interaction between the problem (requirement) domain and the solution (architecture) domain of a product family. The paper furthermore describes how the suggested approach can be accomplished by using a Product Lifecycle Management (PLM) tool. The approach encompasses the activity of mapping requirements to elements of an architecture.

In the approach 5 view models are suggested: 2 requirement model views, a customer view and a functional view; and 3 views describing a product architecture and its functional systems, physical modules and interface view, see Figure 32.

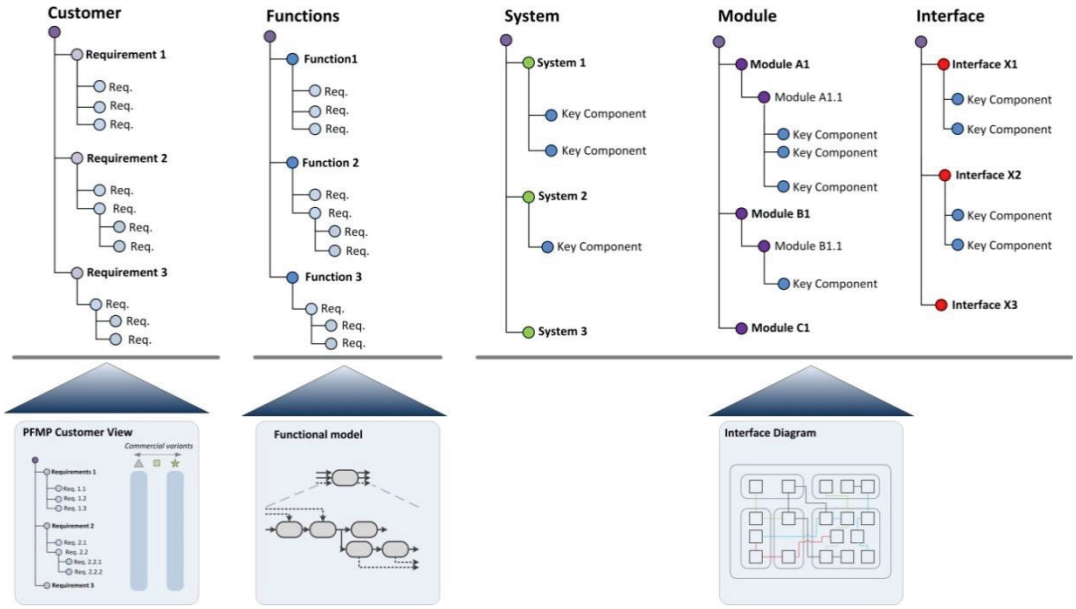


Figure 32 Mapping requirements to architecture views

The function model is used for identifying behavioural requirements, while the Customer view includes both behavioural requirements and design constraints for ranges of products, e.g. a product family. The requirements and design constraints are mapped to the elements (Products, Systems, Modules, Key components, Interfaces/Interactions) of the product architecture by using the PLM system's standard features for linking objects. Based on the architecture design, further requirements are identified, see Figure 33.

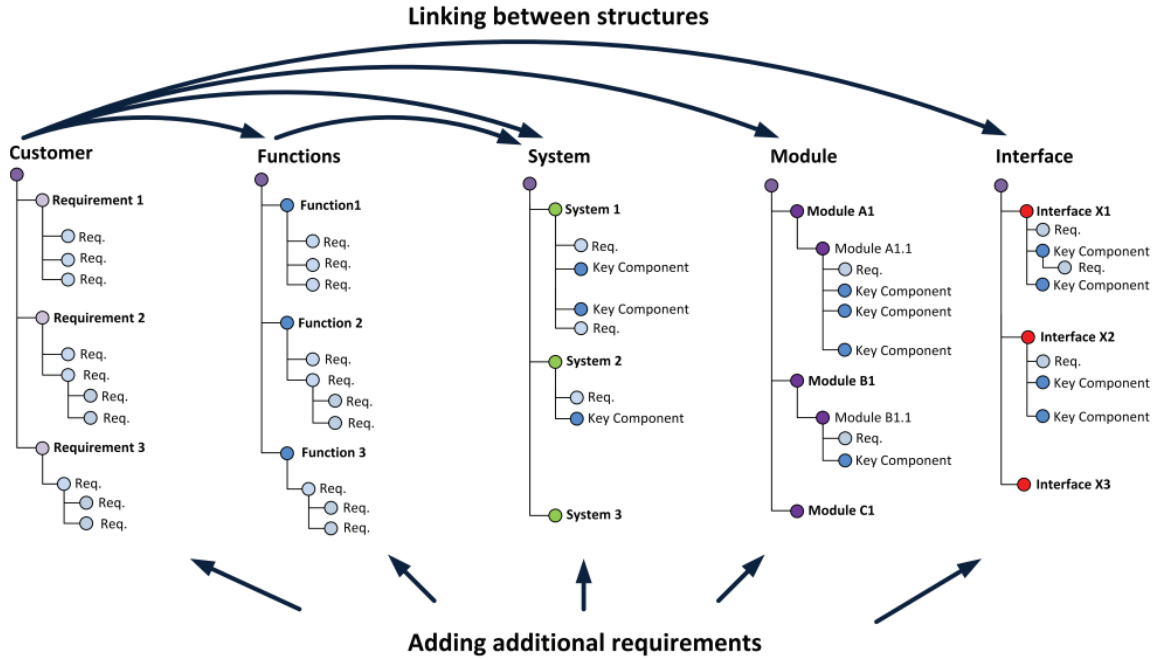


Figure 33 The process of linking, adding and refining requirements in the data model.

The architecture model therefore contains information both regarding design as well as requirements, enabling the designers to specify requirements originating from architectural decisions as those decisions are taken and to continue to work with and elaborate on requirements through the design process. The result is an interactive approach to work on requirements and architectures. The traceability and continuous evolvement of requirements is a crucial aspect of all product development projects. By using a

PLM system for representing the product family defining architecture and for mapping requirements to Systems, Modules and Interfaces, it is possible to trace requirements in a straightforward way. Using a PLM system in the approach provides a design team with full traceability from requirements to architecture elements and from architecture elements to requirements.

Research method

The research presented in this paper is part of a conceptual research study in which the authors were working closely together with PLM specialists from a vendor company (*PTC©*).

Reflection on contribution

The approach has been validated for functionality, but still remains to be tested in an industrial setting. This work is in progress in collaboration with the software vendor and a large manufacturing company developing products to the construction industry. Reflections on the contribution include:

- The approach is especially developed for complex products where many designers from different technical domains are involved in the development process.
- The approach can be characterised as a top-down approach in relation to the domain of product design, as requirements can be added and refined when decomposition and characterisation of the design is progressing.
- Some central questions arise when reflecting on the approach concerning the characteristics of requirement models that are intended (soll) and property models that are about the realisation (ist). The nature of their relations has to be investigated in future work.
- It is stated in the paper that functions have input and output; this is a fault. The right word is that functional elements (organs) have input and output.
- The approach has to be refined and detailed in order to prescribe procedures for how linking is performed in detail. The pattern of relations between requirements and elements of the architecture are highly complex for large product systems having thousands of requirements. Furthermore, requirements have multiple relations in cross functional designs, and the endeavour to establish these links manually can be overwhelming.
- The approach has, however, some advantages compared to having requirements in large specification documents, where it can be difficult to relate specific requirements to specific design elements. Moreover, standard RM systems miss the ability to have all the additional product information from a PLM system mapped to a model of an architecture.

4.4 Bridging to the conclusion

This part of the thesis describes the research and its results presented in the appended papers. Moreover, it has been chosen to enhance detailed descriptions of the results presented in the papers, for assisting the reader to better understand the presented contributions and their relations. Part 5 will elaborate on the results and state the argumentation for answering the research questions, and finally the results will be critically evaluated.

5 Conclusion

The aim of Part 5 is to conclude the research and its contributions. Part 5 presents the argumentation of the extent to which the research questions could be answered and outlines the considered main contributions. Additionally, the validity and verification of the research contributions are evaluated, and the results critically evaluated. Finally, suggestions on future work identified in this research are explained.

5.1 This research's argumentation

As described in Chapter 1.3, five research questions are formulated for this research. The following sections will elaborate on the argumentation when answering these. The contributions are split in the two areas introduced in Part 4, namely: (1) *Identification and modelling of structural and behavioural aspects of product architectures*, and (2) *Architecture representations in PLM systems.*

5.1.1 Identification and modelling of structural and behavioural aspects of product architectures

The contributions in Papers A and B are both compliant with research question RQ1 in the sense that they elaborate on what challenges designers in manufacturing companies face when trying to identify, develop, and model architectures. The formulation of the first research question is:

> *RQ1: What phenomena related to modelling of product architectures, from a structural and behavioural point of view, pose a challenge to designers when developing product systems?*

By understanding the phenomena related to modelling of product architectures from a structural and behavioural point of view, it is essential once more to outline what structure and behaviour are in this research. Building on constructs of The Theory of Technical Systems and The Domain Theory, a product's nature can be articulated as follows, Andreasen, Howard & Bruun *(2014)*:

- A product is defined by its *structure* which is a 'static' description of its anatomy.
- When the product is deployed by the user, it means brought into a context and utilised in a use process, then certain stimuli will be present and the product will show its *behaviour*.

The application of modelling techniques of architectures is a means for designers from different disciplines to intervene in the process of identifying architectures that satisfy the goals of the company. The dominant challenge in this process is to convert desired behaviour to solutions under the constraints given, i.e. resources in the company. The challenge of modelling an architecture from both structural and behavioural points of view, relates to the synthesis process of finding the structural characteristics of a design, which is believed to possess the required properties. If the design is based upon past designs, the engineering designer has a more or less complete insight into the structure-behaviour relation, Hansen & Andreasen *(2002)*. In new product development or if the required functionality is new, it is much more complex and difficult to establish the structure-behaviour relation. Then it is necessary to create structural proposals for being able to predict and evaluate the relations, e.g. to allow prediction of a solution's properties: size, weight, style, functional performance, price, etc.

In engineering design normally several models of the design are created. These models can be considered as different views, e.g. characteristics (dimension, form, material, and surface quality), properties (strength, control, costs, etc.) or models of product life (production, sales, use etc.). It can be postulated that in engineering design there is only one design to be designed and only one product to be produced. A part can for example only have one length independent of any viewpoint, Mortensen *(1999)*. The different viewpoints on products are created to support designers to evaluate designs, e.g. for articulating properties, for capturing use, form, functionality, for clarifying layout and interfaces, for creating commonality between product variants of a product family, for getting insight into the product's and the component's properties, etc. The challenge to designers is to orchestrate all these different models to support both conceptualisation and alignment between interdisciplinary domains. Computer systems are often used to develop and define different models, but the challenge is to see models and their relations in the context of the goals of the company. One of this research's objectives is to use a PLM system to represent different models and thereby ease the task of navigating through information when developing architectures.

The contributions in Papers A and B are both compliant with research question RQ2, as they propose architecture models that address structural and behavioural aspects. The formulation of the second research question is:

> *RQ2: How can both structural and behavioural aspects be addressed in integrated architecture models?*

Paper A is the first paper of this thesis that describes *The Interface diagram*. The Interface diagram is a proposal for a model used for supporting the process of architecture development by conceptual modelling of decomposition, characterisation, and composition. Decomposition means to divide the architecture into subsystems, Svendsen *(1994)*. Characterisation is a process in which characteristics and values of characteristics are determined. Composition consists of integrating the subsystems into an entirety. During composition interfaces between subsystems are determined, thus there does not exist a sharp distinction between characterisation and composition, Mortensen *(1999)*. The format of the proposed architecture model is a printed poster made in MS Visio providing a visual overview of an architecture in block diagram form. The reason for printing the model on a poster is that a poster has the advantage of providing a better overview, and a more interactive modelling media than most computer screen interfaces can provide.

The main elements of the diagram formalism are structural elements (organs and/or parts) denoting Key components. The structural elements can be arranged in layers of the model according to a Systems view or a Module view. Thereby the model can represent subsystems (System view) holding structural elements that can be spread throughout the design, as illustrated in the first part of Figure 34. The purpose of modelling subsystems is to support the evaluation of functionality and properties in solutions that are spread across multiple modules. The same structural element can, from a behavioural perspective, be a part of multiple systems, where the coloured boxes represent structural elements that belong to different subsystems. To identify subsystems, a functional model can be created, providing a graphical representation of the transformation of energy, material or information flow as they pass through the system. The main reason for modelling architectures focusing on functions is to be able to describe what the derived designs shall deliver, and after designing, control that the designs are actually able to deliver the desired functions.

The second viewpoint in the model, see the middle part of Figure 34, is a modular viewpoint in which structural elements are encapsulated into modules. Modules can consist of structural elements belonging to different subsystems, i.e. developed by different engineering teams. The main reason for having

modules is to find the best arrangement of structural elements that will optimally support the product lifecycle; some examples can be manufacturability, encapsulation of complexity, upgradability either for a current program or aftermarket needs and serviceability. Additional reasons for creating modules are to support business strategies as mass customisation and/or standardisation, i.e. designing modules that can constitute product platforms and be carried over in several products belonging to a family or program.

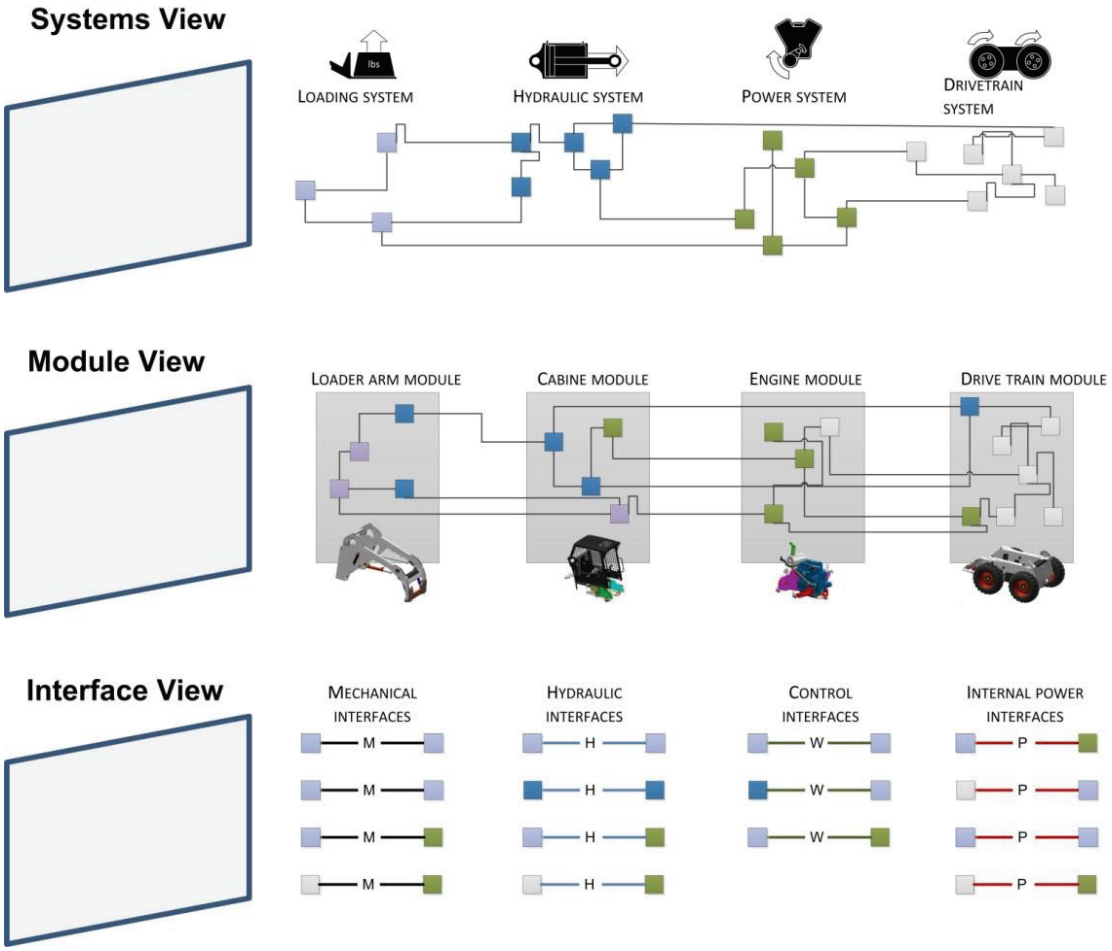


Figure 34 A simplified illustration of the views handled in The Interface diagram; Systems View, Module View and Interface View.

The last view handled in The Interface diagram is the Interface View, see Figure 34, which is not an unconnected view, but based on the relations modelled in the Systems and Module views. The name is slightly misleading because the view also represents interactions, although for reasons of simplicity named *Interface view* and not *Interaction and Interface view*. Interactions and interfaces are created by composing and characterising structural elements. The overview of interactions and interfaces in the architecture is an important part of the support for predicting the behaviour of the structure and for allocating responsibility of critical interactions and interfaces in an architecture.

The objective of working with subsystems is to address functional decomposition of the product system and to ensure realisation of functionality in subsystems and in the product system as a whole. The objective of working with modularity in The Interface diagram is both to explore the behavioural effects of a module candidate and to support the creation of commonality in a product architecture, e.g. reuse between different product variants. A subsystem is a product of an active decision process, but should not be

understood in realisation of best physical arrangement of components, but in realisation of best functional performance of the product system as a whole. To outline the characteristics of the two views:

- Systems view – is a function/organ synthesis view defining the organ and property aspects and the gradual functional composition of the organ structure.
- Module view – is a structure synthesis view defining the gradual part structure synthesis.

Additional effort for answering RQ1 and RQ2 is described in Paper B. The framework presented in the paper is tailored to support project-based companies doing customisation by engineering of products with highly integrated product structures of complex parts. It is supposed to support the identification of integrated architectures of the product and production set-up and solve the problems of modelling structure-behaviour relations in architecture representations. The framework approach takes its starting point in bridging the defining design characteristics and behavioural performance parameters of the products with the market needs that are critical to achieving competitiveness within a target segment. By systematically identifying and implementing scalable solutions, architectures for the product and production setup are developed. The paper concludes that products with highly integral product structures which cannot be modularised in a traditional way still can benefit from identifying areas of commonality and variety. The architecture models included in the framework are used for describing both structural elements and behavioural effects.

The contributions in Papers A and B outline some main challenges of modelling architectures to describe the structural building principle for product systems and evaluate their behavioural effects. However, predictions of solution properties are strongly subjective imaginations from experience, and should be based upon adding details to the solution. These details can be added to an architecture model by means of a PLM system holding multiple behaviour models and property models. The practical research questions RQ3, RQ4 and RQ5 all involve adding details to an architecture representation.

5.1.2 Architecture representations in a PLM system

RQ3, RQ4 and RQ5 are all addressing the implementation of architecture models in a PLM system, and the ability to manage and process information in PLM systems. The third research question, RQ3, is centred on the practical implementation of architectures in PLM systems. The objective is to use PLM systems in a systematic way to: design, manage, direct, and control information needed to document an architecture. The formulation of the third research question is:

RQ3: How can PLM systems support architecture models that emerge in steps of the design process?

The contributions in Papers A, C, D, and E are all compliant with research question RQ3, because they give answers to the utilisation of PLM systems for supporting architecture models that emerge during the design process.

Pedersen *(2009)* argues that an information model outside the computer modelling domain has the potential to be independent of the different IT systems and may help to orchestrate the information in the different systems, see Figure 35. This idea is concordant with the use of the suggested architecture model, The Interface diagram, in this research. The architecture model becomes an information model in the PLM system and in that way helps to orchestrate information created in multiple computer systems. Structures of the information model (Products, Systems, Modules, Interfaces, and Key components) are transferred to the PLM system and information from models created in other computer systems such as commercial CAD and ERP systems, can be linked to structures of the architecture, see Figure 36.

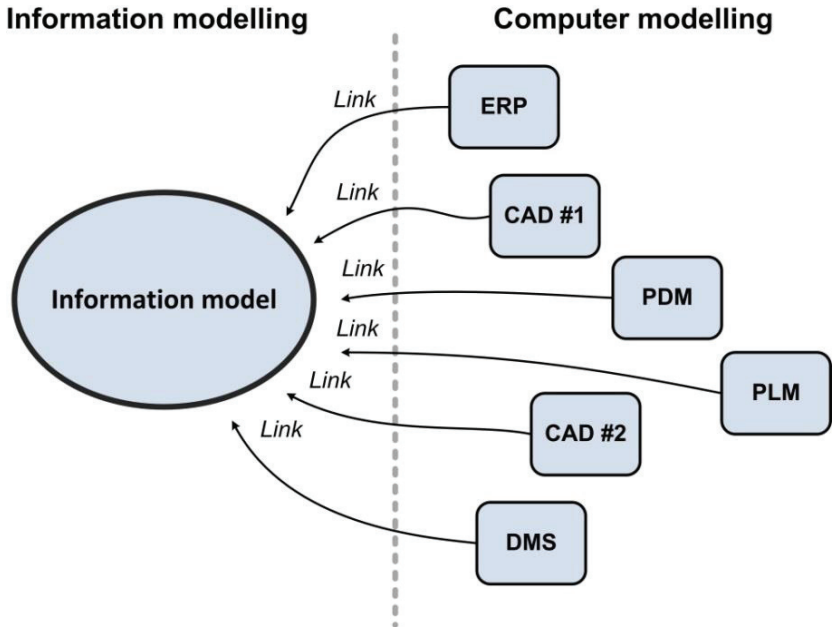


Figure 35 Bridging between Information and computer models, redrawn after Pedersen *(2009)*

The process of getting the architecture information from The Interface diagram and into the PLM system is basically divided into two steps: Exporting from Interface diagram and importing into PLM. An interchange format based on XML is defined to contain all information from The Interface diagram between export and import, see Figure 36. For more details see Paper D.

Instead of being a PLM environment with focus mainly on the administration of files generated from specific tools like CAD, the PLM system now contains a master product model. The system contains the model of the architecture by means of the modelled structures from The Interface diagram. The structures in the PLM system are named *Upper structures*, to underline that they constitute the master product model. Information from multiple IT systems can be associated with elements in the Upper structures.

Information defined inside the PLM system or in associated computer systems, such as CAD or text documents, are denoted *Lower Structures,* see Figure 30. Information that can be linked to the Upper structures is, e.g.:

- CAD models, e.g. 3D parametric models and 2D drawings.
- ERP system information, e.g. purchase specifications and cost targets.
- RM system information, e.g. internal and external requirements. External requirements from CRM can be handled in a RM system, but are outside the scope of this thesis. A proposal for associating requirements to a product architecture model can be seen in Paper E.
- DMS (*Document Management System*) system information, e.g. Interface Control Documents.

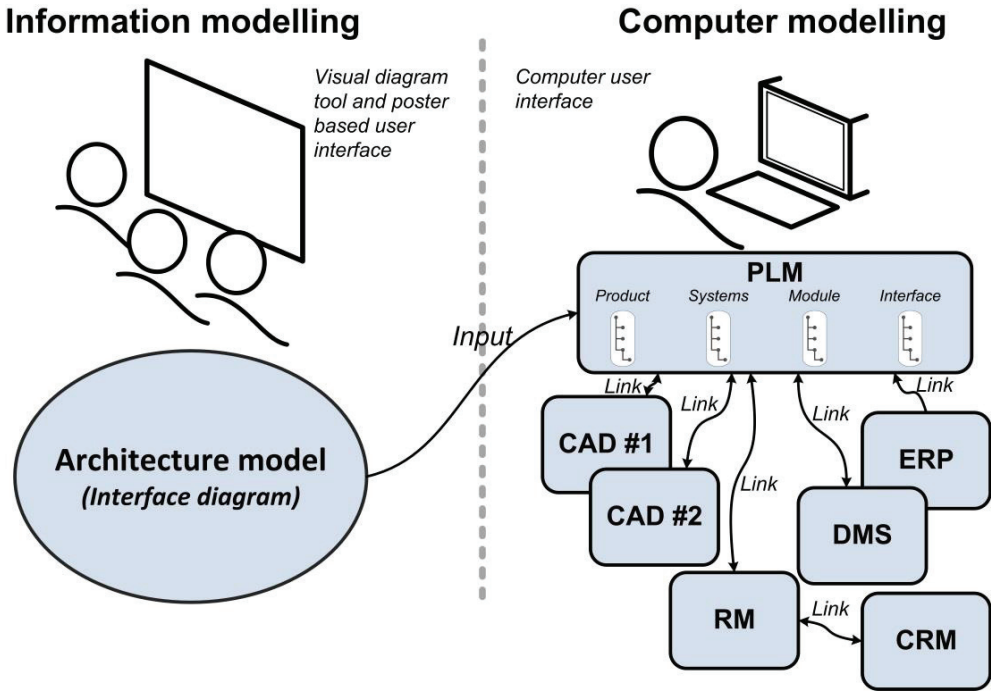


Figure 36 Importing the architecture model to the PLM system to orchestrate the information to and from multiple IT systems.

People responsible for designing subsystems, modules, and interfaces are certified to enrich, create, manipulate, associate and/or delete information belonging to their specific designs, and every certified project member is able to view information from all elements of the architecture, see Figure 37 .

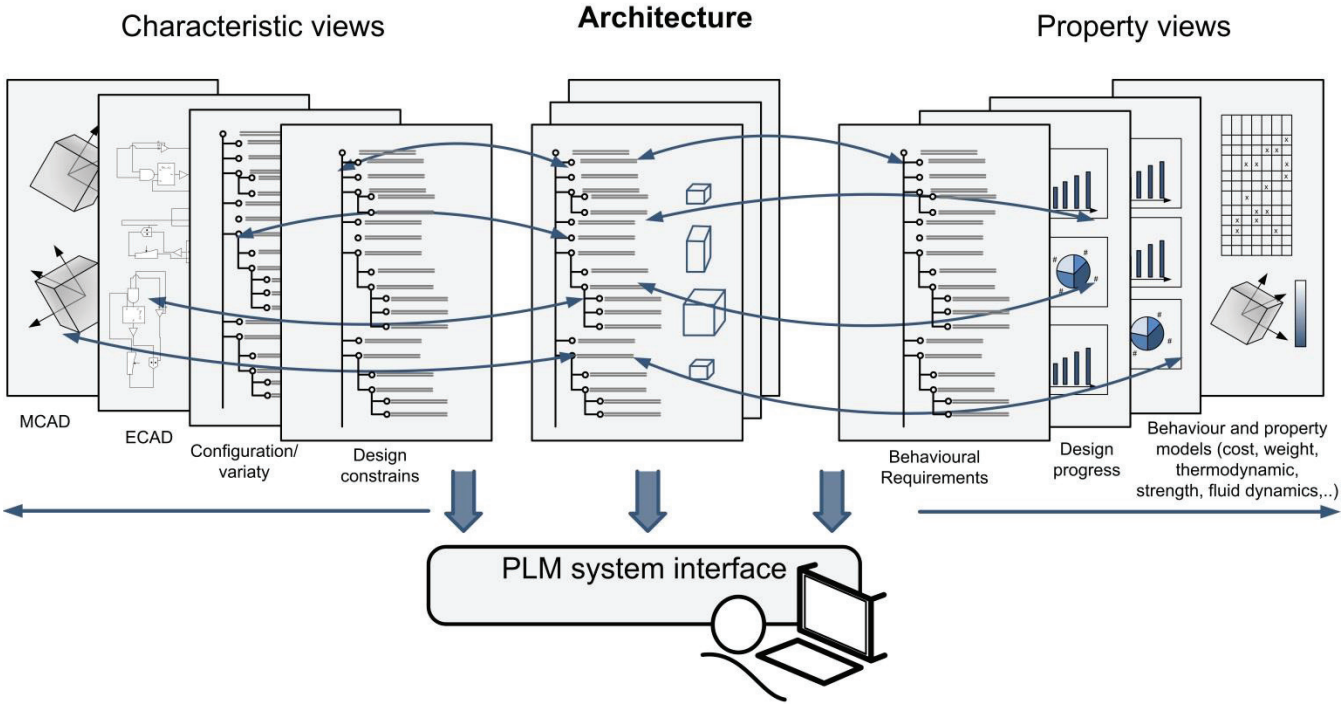


Figure 37 PLM system support for detailing solutions by associated information in the design process.

By superimposing the different views of the architecture, designers are supported in reasoning, evaluating, and optimising towards the architecture that satisfies the goals of the company. Properties of the design are related to goals of the company, as they describe the behavioural effects of the design, i.e. what is the

design able to do for the company. Different models with different purposes can be used for supporting the designers towards the structural characteristics of a design, which is believed to possess the required properties.

The contributions in Papers C and D are both compliant with the research questions RQ4 and RQ5 in the sense that they give answers in the form of two specific developed applications supported by a PLM system. The fourth research question, R4, points at a specific task in the design process, namely the task of assessing product cost in terms of direct materials, i.e. the cost of materials used to manufacture or purchase a product. The formulation of the fourth research question is:

> *RQ4: How can PLM systems be used for assessing product cost during the design process?*

The fifth research question, R5, is formulated with the incentive of creating support for assessing the completeness of a design during the design process, by counting the number of new, modified and approved structural elements of an architecture. The formulation of the fifth research question is:

> *RQ5: How can PLM systems be used for assessing the completeness of a design during the design process?*

Both applications can be characterised as reporting capabilities, see Figure 38. The first application refers to research question RQ4 on assessing product cost during the design process. Every element defined and modelled in the architecture can be allocated an initial target cost attribute, and gradually direct material cost can be added to all elements as they are developed or purchased. By means of roll-up mechanisms, cost is added from the top product level and down to the smallest components. If a direct cost attribute is still missing for a component, a target cost should be used instead. Sometimes product cost is estimated for modules or components where no solution exists, which makes the estimate difficult and sometimes speculative. However, the application of cost reporting in a PLM system is seen as a hands-on tool for supporting development of a design according to budget because derivations between budget and actual cost can be discovered quickly, thereby raising the issue of choosing the best solution within the resource constraints.

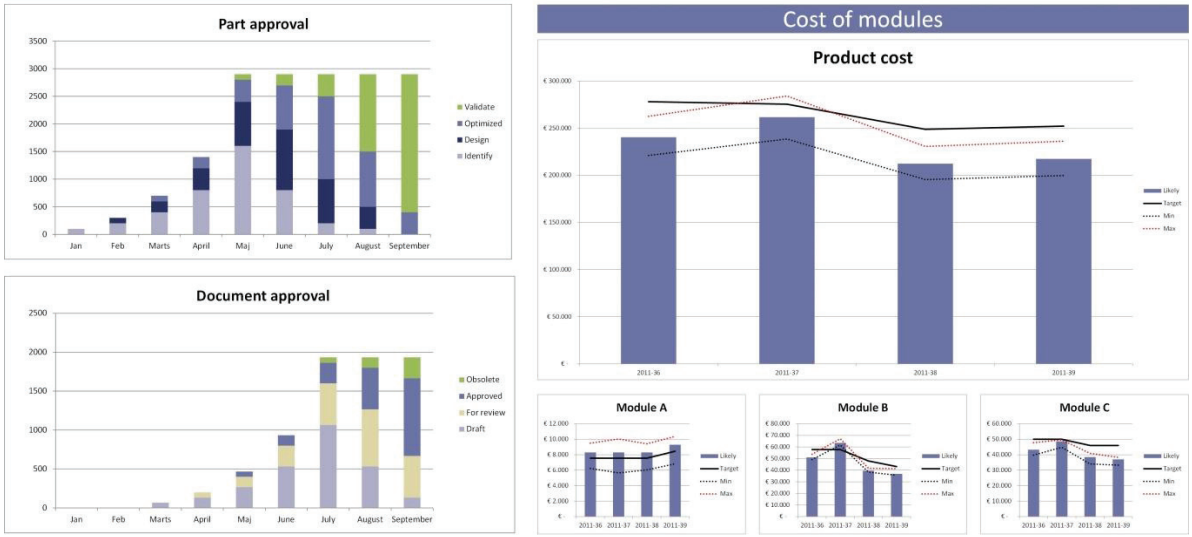


Figure 38 Reporting capabilities of the PLM system. The export format is shown in MS Excel.

The second application refers to RQ5 about assessing the completeness of designs during development. Reporting functionality is established in the PLM system and is used to monitor design progress on a weekly basis. The numbers of new, modified, and approved components are used as indicators of the progress of the design. By seeing the numbers of new and modified components drop, an indication of the completeness of the entire design can be seen and evaluated. This is not an unambiguous measure, but is seen as complementary to status reports from the design team responsible and project managers. A more complete measure of design completeness should include clarified requirements and finished production documentation. The functionality of assessing the status of documentation in the PLM system can be assessed by the same principle as status of designs. The resolution level of components' status enables early detection and thereby assessing the progress of design and more importantly, if a planned schedule for a design is kept.

The contributions in Papers C and D are implemented in the case company, and the experience of applying them leads to the claim that developing products based on architectures can be done more effectively and efficiently when designers are supported by a PLM system. The effectiveness relates to the improved handling of information by having one unifying information model for structuring, associating, evaluating, and monitoring product architecture information. Efficiency describes the extent to which time and effort in implementing and following the approach is well used for the intended task. It was not possible to measure the quantification of the efficiency of the approach at the point in time this thesis was written. However, the qualitative statements from the users reported in Paper D, points to a more efficient developing process, as time spent on navigating through information and the risk of creating redundant information has gone down.

5.2 Core contributions

The core contributions of this research are the proposals for modelling structural elements and addressing behavioural effects of architectures. The structural elements of architectures are organs and/or parts, and they are modelled in an integrated model showing the causal relations between them by using two views, namely a Systems view and a Module view, see Papers A, C, D, and E. When developing architectures and identifying their structural elements, this research postulates that there is a need for taking multiple behaviour models (describing the ability of the design to create effects) and multiple property models (describing relational properties as a consequence of meetings between the design and the life phase systems) into account. Handling multiple view models of the design is in this research supported by a PLM system, and the core contributions encompass descriptions of approaches and tools for doing that.

This elaboration leads to three statements of the core contributions of this research:

(1) A contribution to systems theory and to model based design theory with knowledge and understanding of how to model structural elements and address behavioural effects in graphical architecture representations.

(2) A contribution to design process theory with knowledge and understanding of an approach for architecture representation in a computer-support system (PLM) to support inclusion of structural and behavioural models.

(3) A contribution to knowledge and understanding of representing architectures in computer systems (PLM) for supporting synthesis and analysis activities, i.e. supporting the designer in converting desired behaviour to solutions given by components and to analyse properties and effects of architectures.

The following chapter will evaluate the validity and verifiability of the contributions.

5.3 Method for research validation and verification

There exist a vast number of definitions for *verification* and *validation*, Haskins *(2006)*. As the two words *verification* and *validation* often are used interchangeably, the understanding of their meaning in this thesis is stated, referring to Pedersen et al. *(2000)*, as: verification of results refers to the acceptance of the process of achieving them whereas validation refers to the acceptance of the usefulness of the research and its results.

An important part of this research's results are models of product architectures. Models can be used for three purposes: (a) structural exploration, (b] prediction, and/or (c) prescription, McCarl *(1984)*. The purpose of using models in this research is covering all three purposes in different stages of the design process. The ultimate validation of a model involves observing whether an available model is used for its intended purpose, McCarl *(1984)*. Evidence of model validity is often required for a model to be considered for use. However, this is often time consuming and therefore to give models such 'trial by fire' in a limited PhD project is impossible. However, it is possible in a PhD project to evaluate the experience of implementing a model and get the initial indication of its usefulness. Andreasen, Howard & Bruun *(2014)*, describe validity of models and design theories in that they should be meaningful to practitioners, they should be easy to integrate into their practice and support their reasoning; this is where the validity should be found. The experience of implementing and applying the models in the case studies indicates the validity of them.

This chapter seeks to describe how this research and its results are verified and validated. Several relevant validation frameworks are considered and discussed, where one framework is chosen for evaluating the research contributions.

5.3.1 Case study rigor

Very few case-study authors explicitly label the rigor criteria in terms of the concepts commonly used in the positivist tradition; *construct*, *internal*, and *external validity*, as well as *reliability,* Gibbert & Ruigrok *(2010)*. The concepts are however found purposeful for qualitative researchers to ensure research rigour when doing case studies, Yin *(2009)*. The four concepts of validity address the methodological issues described below:

Internal validity:

- To what extent does the researcher provide a plausible causal argument with logical reasoning that is powerful and compelling enough to defend the research conclusions? Cook, Campbell & Day *(1979)*.

Construct validity:

- To what extent does a research procedure lead to an accurate observation of reality? Denzin & Lincoln *(1994)*.

External validity:

- To what extent may the findings obtained be generalised to other settings? Eisenhardt *(1989b)*.

Reliability:

- To what extent may the same findings be obtained if another researcher conducted the study following the same research procedure? Denzin & Lincoln *(1994)*.

The four concepts for ensuring case study rigor are relevant for this research, as its results are based on case studies. The concepts for ensuring validity are important and support the later chosen framework in evaluating the research and its results.

5.3.2 Design research methodology – Research evaluation

The DRM methodology Blessing & Chakrabarti *(2009)* recommend three concepts for evaluating research contributions: Usability, applicability and usefulness:

- Usability: The ease with which the method can be used for the intended task.
- Applicability: Whether the method has the direct intended effect on the design process.
- Usefulness: Whether the introduction of the method leads to the overall success of the project measured by a number of parameters taking into account possible uncontrollable influencing factors.

The three evaluation parameters are seen as beneficial in this research context, as the contributions are centred on approaches for applying models in a company setting. The evaluation parameters support and compliment the later chosen framework for evaluating the research and its results.

5.3.3 Two-approach framework – derived from mechatronic design research

Buur suggests two approaches for how to verify a design theory, Buur *(1990)*.

Logical verification

- A theory must be consistent: Internal conflicts between the theory constituents are not accepted.
- A theory must be complete: The theory must explain or reject observed phenomena of relevance.
- A theory has to support established and widely accepted methods as well as specific design problems.

Verification by acceptance

- A theory must be accepted by a relevant scientific community.
- A theory must be accepted by industrial practitioners.

The framework is pragmatic in its way of addressing the verification of a design theory. The two approaches are considered relevant for the process of validating and verifying the results of this research. The framework of Buur is complementary to the later chosen framework for evaluating the research and its results.

5.3.4 Validation Square framework

The question of how one validates design research in general, and design methods in particular, has been addressed by Pedersen et al. *(2000)*. They assert that research validation is a process of building confidence in its usefulness with respect to a purpose. Moreover, they associate usefulness of a design approach with

whether the approach provides design solutions 'correctly' (effectiveness), and whether it provides 'correct' design solutions (efficiency). The two measures for usefulness are embodied in six aspects:

Effectiveness:

(1) Accepting the individual constructs of the method/theory.

(2) Accepting the internal consistency of the way the constructs are put together in the method/theory.

(3) Accepting the appropriateness of the example problems that will be used to verify the performance of the method/theory.

Efficiency

(4) Accepting that the outcome of the method/theory is useful with respect to the initial purpose for some chosen example problem(s)

(5) Accepting that the achieved usefulness is linked to applying the method/theory.

(6) Accepting that the usefulness of the method/theory goes beyond the case studies.

Pedersen uses The Validation Square framework; see Figure 39, for validating design research.

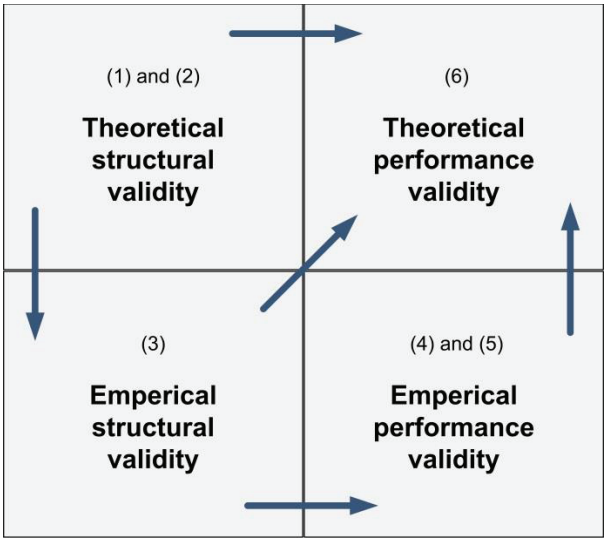


Figure 39 Validation Square framework, redrawn from Pedersen et al. *(2000)*

The framework focuses on both theoretical and empirical validity, which are the two cornerstones of this project. The Validation Square framework accounts for the qualitative measurable criteria of engineering design research. Often, engineering design research is characterised by few quantitative indicators, making it hard to ensure a strict validation based on observations, Pedersen *(2009)*. This is also the case in this research, where the measurable criteria are of a very qualitative nature, and the case projects are characterised by many factors involving the overall performance of the projects. The research results are in the following chapter evaluated using the six aspects of validating the results of this research.

5.4 Evaluation of the research and its results

The results from the case studies are verified and validated using The Validation Square framework, Pedersen et al. *(2000)*. The papers will be evaluated individually or in groups if they address the same case study and/or result.

5.4.1 Evaluation of the research and its results as a whole

First, an evaluation, based on the Validation Square framework, of the research and its results as a whole, see Part 4, is executed. There is trickiness to assessing the research and its results as a whole, as the stated contributions are of a different nature. This will be commented on in the following evaluation of the contributions in the individual papers.

(1) *Individual constructs:* The research builds on a widely accepted theoretical basis of system theory, theory of architectures, model based design theory, and information theory. This indicates, but does not ensure, that the new constructs of the suggested approaches, models, and tools are widely accepted in the research community and the empirical field. However, the acceptance of the proposed constructs is indicated in peer-reviews and presentations of papers in journals and at conferences, and by the feedback from practitioners in the case studies.

(2) *Internal consistency:* The constructs of modelling formalisms and approaches are based on theory presented in Part 3. Even though new terminology is introduced for constructs, the internal consistency follows the internal consistency of a widely accepted theoretical basis, e.g. Theory of Technical Systems and Theory of Domains. When new terminology is introduced for constructs, emphasis is put on defining them in relation to the used theoretical basis.

(3) *Appropriateness of the example problems:* The five case studies represent a small diversity of industries in Denmark. However, they all share the common challenge of developing complex product systems and managing the product related information while doing it. The focus of the research has been the modelling of architectures of mainly mechanical product systems, which is a task conducted in all case companies, and it is assessed that they serve as an appropriate basis for the empirical validation. Having said that, the fact that all case studies are anonymous makes it difficult to report the example problems in a detailed way so the appropriateness of them can be externally evaluated. Issues such as product complexity, engineering competences, and development setup, are all important when evaluating the example problems. A skid-steer loader (BobCat) has been used as an example product when illustrating the proposed modelling formalisms and approaches. This product is in many ways similar to the actual products treated in the central case study #5, see Section 2.3.2.

(4) *Useful outcome:* The contributions of this research are, as explained earlier, graphical architecture models aimed at supporting designers when identifying and developing architectures. Part of the contribution is the method of applying a computer system (PLM) to model architectures. The reported effects of the contributions are of a qualitative nature, and have been assessed by interviewing key stakeholders, and in some cases estimating the quantitative impact. The expected outcome of supporting the designers when developing architectures has been assessed in terms of: support of converting desired behaviour to solutions for architectures, range design, modularisation, evaluating behavioural effects of architectures, interdisciplinary collaboration, and information management. Moreover, the measurable criteria of the impact model (reducing development time) have been assessed as being met, see Figure 8 and Papers A and B.

(5) *Link the achieved usefulness to the applied approaches and tools:* The case studies have been conducted partly following an action research-based method, which again implies that the research has been concerned with the study of phenomena that would not have occurred without

the researcher's influence. Consequently, the case studies cannot be evaluated objectively. Moreover, the practical setting of the case studies is influenced by other impacting factors, making it difficult to isolate a direct link between the applied approach and tools and the experienced usefulness. Graphical architecture, the method of applying a computer system (PLM) to represent them, and the framework presented in Paper B, were all created with the incentive to support designers when identifying and developing architectures. Common for the case studies, even though many factors are clearly influential, is that no other competing agendas for modelling support and information management were introduced during the studies. This supports the acceptance of a link between the applied approaches and tools and the achieved usefulness.

(6) *Usefulness beyond the example problems:* The case studies represent a setting which many companies are finding themselves in. The challenges of developing architectures for ranges of multi-domain products and to actively use necessary product information from multiple IT systems in the process, have been identified by other research communities described in the theoretical basis, see Part 3. Literature about architecture based development and information management addresses a wider range of example problems than in this thesis, e.g. different product types, business types, industry types, and project types. However, there is no one-fits-all when it comes to the tailoring of architecture initiatives to a specific situation of a company, Hansen et al. (2012). By including contextual characteristics the probability for successful scoping and goal setting of architecture initiatives eventually leads to good performing product programs. This again supports the acceptance of, or at least indicates, the usefulness beyond the case studies of this research.

5.4.2 Evaluation of contributions in Paper A

The contributions presented in Paper A cover a product architecture formalism for modelling structural elements (organs and/or parts) of architectures in order to support designers in converting desired behaviour to solutions given by components in architectures. Evaluation of this contribution is performed in the following list:

(1) *Individual constructs:* The contribution builds on an architecture modelling tool, *The Generic Organ Diagram* and the *PFMP* framework, Harlou *(2006)*, both accepted approaches for modelling architectures in a function-oriented manner. Moreover, the acceptance of the proposed constructs of the architecture tool is indicated in peer-reviews and presentations of the paper and by the feedback from practitioners in the case studies.

(2) *Internal consistency:* Even though new terminology is introduced for constructs, the internal consistency follows the accepted theoretical basis, e.g. Theory of Technical Systems and Theory of Domains. When new terminology is introduced for constructs, emphasis is put on defining them in relation to the used theoretical basis. The architecture model has been developed, tested and refined in an industrial setting, with the constructs tested for internal consistency.

(3) *Appropriateness of the example problems:* The product architecture model has been deployed in case study #5, which represents a company developing complex product systems following a mass customisation strategy. The company is challenged by the task of integrating different engineering disciplines and by the task of managing their diverse product information. Case studies #1, #2, #3, and #4 have formed the creation of the described architecture formalism, as identification and modelling of architectures has been the subject of these studies. The example problems are assessed to serve as an appropriate basis for the empirical validation because of their diversity of product, business and project types.

(4) *Useful outcome:* The results of this research presented in Paper A, are of a qualitative nature, and have been assessed by interviewing key stakeholders, and by getting them to evaluate the realised support and reduction of development time. It is reported in the paper that the

application of the architecture model has contributed to a reduction of development time of -25%. The usefulness of the application is centred on the model's ability to serve as a boundary object in interdisciplinary product development. The expected outcome of supporting the designers when developing architectures has been assessed as being realised in terms of support with: converting desired behaviour to solutions for architectures, range design, modularisation, and interdisciplinary collaboration.

(5) *Link the achieved usefulness to the applied approaches and tools:* As described in Chapter 2.4, the practical setting of the case studies is influenced by other impacting factors, making it difficult to isolate a direct link between the applied approach and tools and the experienced usefulness. As for the other case studies, case study #5 was not influenced by other competing agendas of modelling support during the study. This supports the acceptance of a link between the applied tool and the achieved usefulness.

(6) *Usefulness beyond the example problems:* The relatively broad experience base of the case studies represents a setting that many companies experience. The base of experience is considered as an indication of usefulness beyond the example problems themselves.

5.4.3 Evaluation of contributions in Paper B

The contribution presented in Paper B is a framework including a step-wise approach for the identification of an architecture for the product and the production setup. A framework is tailored to support project-based companies doing customisation by engineering of products with highly integrated product structures of complex parts.

(1) *Individual constructs:* The framework presented is made on the basis of the accepted PFMP methodology, Harlou *(2006)*, merged with the function-oriented focus on performance to identify consistent features, performance and production scalability. The individual constructs making up the framework are accepted both in the academic world and in practice. Moreover, the acceptance of the proposed constructs of the framework is indicated in peer-reviews and presentations of the paper and by the feedback from practitioners in the case studies.

(2) *Internal consistency:* As the framework represents a combination of the PFMP-based approach with a function oriented focus on performance properties in line with the classification of attributes, Andreasen, Howard & Bruun *(2014)*, the internal consistency is regarded as accepted.

(3) *Appropriateness of the example problems:* The framework has been tested in case study #1 and conducted in a global company experiencing major challenges in identifying and modelling architectures for a family of complex integral products. An important part of identifying architectures in this setting has been to support reasoning of structure-behaviour relations in order to have the best layout of the family concerning commonality and variety. The measurable criterion of improving time-to-market has in this study been possible to quantify. The example problem is considered relevant for field testing of the framework.

(4) *Useful outcome:* It is reported in the paper that the application of the framework has contributed to a significant time-to-market reduction. The usefulness of the method outcome is considered substantial.

(5) *Link the achieved usefulness to the applied approaches and tools:* The practical setting of the case studies is influenced by other impacting factors, making it difficult to isolate a direct link between the applied framework and the experienced usefulness. However, the authors can report that no other competing agendas were introduced during the study. This supports the acceptance of a link between the applied framework and the achieved usefulness.

(6) *Usefulness beyond the example problems:* As for many frameworks, there is a need to adapt it to the specific purpose, and the framework was tailored particularly for the company in Paper B. However, the company was in a situation many OEM companies are finding themselves in, which supports the acceptance of the usefulness beyond the case study.

5.4.4 Evaluation of contributions in Papers C and D

As the research results of Paper C and D build on the same case studies and cover the same contribution, the evaluation of these is made jointly. The papers describe the combination of a visual architecture model to create an overview, improve communication and collaboration, support the conversion from desired behaviour (functions) to solutions given by components, and to support the creation of a modular architecture. The model is used in interaction with a PLM system to manage and integrate product information belonging to the architecture effectively. The results from the study encompass PLM capabilities for handling multiple product structures, visualising multiple architectural views, controlling interfaces, and quantifying and communicating the status of product cost and progress of design completeness.

(1) *Individual constructs:* The research builds on a widely accepted theoretical basis of system theory, theory of architectures and information theory, which indicates that the constructs of the suggested approach and tools are accepted in the research community and the empirical field. Moreover, the acceptance of the proposed constructs of the architecture tool and of the information model is indicated in peer-reviews and presentations of the two papers, and by the feedback from practitioners in the case studies.

(2) *Internal consistency:* The research builds on an accepted theoretical basis of systems theory, architecture theory, model based design theory, and information theory; moreover, its results have been published and peer-reviewed. This indicates acceptance of the constructs' internal consistency. The research results have been deployed in case study #5, where they have been developed, tested and refined, and therefore tried out for consistency.

(3) *Appropriateness of the example problems:* The case study represents only one company, due to the effort and time demand of implementing computer-supported information systems and the limited time of a PhD project. However, the case study has similarities with related research about application of information systems, e.g. Bergsjö, Malmqvist & Ström *(2006)*, Bergsjö, Catic & Malmqvist *(2008)*, Malmqvist *(2001)*, Li et al. *(2013)*, Prasad, Wang & Deng *(1997)*; this indicates that the example problem is appropriate for the suggested purpose.

(4) *Useful outcome:* The results of this research presented in the appended papers C and D, are qualitative in nature, and have been assessed by interviewing key stakeholders. The qualitative statements presented in Paper D serve as an acceptance of the usefulness of the suggested approach. The statements cover: Design readiness much clearer earlier in the projects, more transparent cost deviations, design progress more accurate to measure on a weekly basis, all engineering teams have access to the same single source of information, easier to develop and evaluate module concepts in the early design phases, and more effective reviews.

(5) *Link the achieved usefulness to the applied approaches and tools:* The results presented in the appended papers D and C are both addressing the application of the architecture modelling formalism in interplay with a PLM system. The authors can report that no other competing initiatives of modelling support and/or computer-supported information management were introduced during the study. This supports the acceptance of a link between the applied approach and tools as well as the achieved usefulness.

(6) *Usefulness beyond the example problems:* The challenges of developing architectures for ranges of multi-domain products and to actively use necessary product information from IT systems in the process, have been identified by other research communities described in the theoretical basis, see Part 3. Literature about architecture based development and information management addresses a wider range of example problems than in this thesis, e.g. different product types, business types, industry types, and project types. This again supports the acceptance of, or at least indicates, the usefulness beyond the case studies of this research.

5.4.5 Evaluation of Paper E

The contribution presented in Paper E is a conceptual approach for supporting the interaction between the problem (requirement) domain and the solution (architecture) domain of a product family. The paper furthermore describes how the suggested approach can be accomplished by using a PLM system. The approach encompasses the activity of mapping requirements to architectural views of a product family. Paper E represents a conceptual research study and is not based on a case study. However, the PLM system applied in case study #5 has been used as a test environment for assessing the functionality and consistency of the suggested approach.

(1) *Individual constructs:* The research builds on a widely accepted theoretical basis of system theory, theory of architectures, model based design theory, information theory, and requirements/specification theory, which indicates that the constructs of the suggested approach and tools are accepted in the research community and the empirical field. Moreover, the acceptance of the proposed constructs is indicated in peer-reviews and by presenting the paper at the *Design Conference 14*.

(2) *Internal consistency:* The constructs of modelling formalisms and approaches are based on theory presented in Part 3. Even though new terminology is introduced for constructs, the internal consistency follows a widely accepted theoretical basis, e.g. Theory of Technical Systems and Theory of Domains. When new terminology is introduced for constructs, emphasis is put on defining them in relation to the used theoretical basis.

(3) *Appropriateness of the example problems:* The paper does not report from any case study, but addresses identified problems in the industry reported in literature, see Paper E. The described problems in industry are evaluated to serve as an appropriate basis for accepting the purpose of making a contribution.

(4) *Useful outcome:* The results of this research have not been exposed to a specific field test, and their usefulness has to be proven in future research.

(5) *Link the achieved usefulness to the applied approaches and tools:* See (4).

(6) *Usefulness beyond the example problems:* Literature has identified a gap in prescriptive research on how to operationally handle requirements when developing architectures. The paper does not cover a specific field test, but the problems identified in academia and in practice support the probability of a useful outcome.

5.5 Evaluation of the research impact

The evaluation of the research results were treated in Chapter 5.4. This chapter describes and concludes the evaluation of the possible impact of this research. The evaluation of the impact is divided into *academic impact* and *industrial impact*.

5.5.1 Academic impact

The academic impact is seen as the research contributions' relation and effect on the body-of-knowledge in the research area of architecture based development and computer-supported engineering design/information management. The areas of relevance and contribution, see Figure 2 (*The Areas of Relevance and Contributions*), show the considered relations between the contributions of this research and distinct research areas of engineering design research. The research results are considered as contributing to the areas of: *Development of complex product systems, Product architectures, Engineering data management, Computer-support*, and *Engineering design*.

The considered academic impact on the outlined areas will be elaborated on next.

Development of complex product systems

This research focuses on the improvement of design through the development of support to the designer in the design process. The means described in this thesis are intended to support the development of complex product systems, e.g. a family of products or products with a large number of subsystems, components, and interfaces. The computer-based approach to product architecture representation includes a number of means for supporting synthesis activities (decomposition, characterisation, and composition of architectures), identifying and modelling structure-behaviour relations (by means of an integrated view model of organs and parts), and managing design information. The author considers these as important academic contributions, as they are integrated and implemented in a standard computer-system, ready to use and tested with promising results.

Product architecture (architecture based development)

Architecture representations are in this thesis perceived as a means for synthesis and analysis (deriving relations between structure and behaviour, assessing product cost, assessing the progress of designs, and evaluating effects in life phase meetings). The author considers the described architecture models and framework as important academic contributions, as they include new formalisms for modelling structural elements and addressing the behavioural effects of architectures.

The Interface diagram addresses modularity in architectures. Modularity is perceived as a characteristic of an architecture and modularisation for achieving desired effects, e.g. reducing structural complexity, enabling sourcing, enabling concurrent engineering, carry over between products in a family, etc. The aspect of addressing physical modularity and functional subsystems in an integrated architecture model is considered to be of significant academic interest, as the combined representation of a product system enables strong reasoning between structure and behaviour.

Important aspects of modular architectures are interactions and interfaces between modules. This research adds to the legacy of previous contributions in the area of Interface management, in suggesting an operational tool (the Interface diagram in combination with the applied PLM system) for defining, controlling, and allocating responsibility of interfaces and interactions.

Engineering data management

This research suggests an approach for representing architectures in a PLM system and thereby orchestrates engineering data related to the architecture. From a theoretical point of view, the approach is considered as an important contribution, as it strives to make a common product definition for data created in dissimilar technology domains. The functional abstraction when decomposing an architecture into structural elements (organs and/or parts) is seen as an enabler for connecting different types of information elements, i.e. elements defined in mechanics, electronics and software. This increases the likelihood of more effective technology domain integration as data is linked to the same product defining information model.

Computer-support

Besides handling multi-viewpoint models of architectures, managing- interfaces, requirements, and product information in general, the results of this research encompass means for computer-support in process assessment and process control. The aspect of assessing design completeness is considered as a contribution to academic and practical undertakings in assessing and controlling design processes. Moreover, the suggested approach for assessing product cost is considered as a contribution for impacting design decisions based on a relational cost property.

Engineering design

In most engineering design research studies there is an initial assumption that support, i.e. a tool, a method, an approach, etc., employed by designers, will lead to some sort of improvement of the design or the design process. The architecture formalisms from this research are considered as support for designers doing architecture product development. The proposed architecture modelling formalism in combination with a computer-support system are considered as important contributions to engineering design research, as they support designers in evaluating architectures from both structural and behavioural points of view.

5.5.2 Industrial impact

The main industrial impact is evaluated based on the experiences gained through the empirical base of the case-studies. The empirical focus of this research has been companies operating in and from Denmark, and the evaluation of the industrial impact is based on the experience gained during the described case studies. The conclusion of the thesis implies that the proposed PLM support to architecture based development enables the designer to identify and develop architectures' structural elements and at the same time get support in evaluating the architectures' behavioural effects, so that the goals of the company can be met in the best possible way given the resource constraints. The conclusion is based on testing and evaluating the results in case studies, which showed promising results. The industrial setting along with the promising results indicates that the contributions presented in this thesis have industrial impact.

The case studies showed that designers in the manufacturing industry today rely profoundly upon IT and computer-support during the product development process. The evolution of 3D models in CAD systems and analysis performed in CAE systems (*Computer Aided Engineering*) enable complex models of the structure and behaviour of a product to be created. Product data models in PDM/PLM systems provide important core information, which is accessible to all members of a product development team. Hence, product models created in computer systems are the dominant support for providing core information about the form, geometry, dimensions and structure of individual parts, sub-assemblies or complete products. In this context, the proposed results from this research have been welcomed by practitioners familiar with PDM/PLM, CAD, RM, and other computer-support systems. The results of this research are considered as an expansion of the possibilities for using computer systems for synthesis, analysis, optimisation, simulation, visualisation, and manufacturing purposes.

The industrial impact of this research can be outlined as reported and considered impact for the case companies. For each theme I will present my considerations about why and how to deploy the contributions from this PhD project in industrial projects.

- *Visual models of architectures in interaction with PLM systems as means for interdisciplinary communication in development teams and for stimulating product synthesis.* Architectures represent abstract phenomena and need support from appropriate models in order to enable working on them. The visual aspect of models is considered important for gathering professionals with different backgrounds, different positions and different knowledge level, and breaking down barriers of understanding. The models created in this research make use of visualisation on different levels for enabling collaboration between designers from different domains. By utilising a PLM system for representing architectures, additional models can be applied to support the

synthesis of structural elements and to support analysis and evaluation of behavioral effects and relational properties. However, collaboration and common understanding between designers does not arise merely from looking at the visualisations. It requires training of users, explanation of formalisms and terminology, and exemplification of alternatives.

- *Visual models of architectures as means for evaluating module concepts and alternative product structuring principles.* The architecture formalisms of this research are primarily intended to be used for developing product families based on common encapsulated modules. However, the framework presented in Paper B underlines the role of modularisation as being means to achieve effects, and not a goal in itself. It is relevant when scoping a product family to analyse what is or could be common or what has to be differentiated in that product family, because this split disposes of many other systems in the product family life cycle; this in return is the focal point for evaluating whether an architecture support the goals of a company. By using visual models as mediators in meetings for interdisciplinary collaboration between design teams, module concepts can be evaluated in a high-level and pragmatic way by analysing the effects of alternative structuring principles. Supporting designers in evaluating structural concepts and their behavioral effects, i.e. functionality and dispositions, are seen as central to where this research has industrial impact.

- *Visual models of architectures for identifying the most critical aspect of product program decisions and making these explicit.* The critical aspects of program architectures relate to alignment of market, product and production architectures. Alignment is about optimisation towards the program architecture that satisfies the goals of the company in the best possible way with the given resource limitations. Visual models of architectures are able to frame alignment of cross domain issues in a way that creates a common understanding between designers and other professionals. This is considered as an impact for the industry, as alignment of cross domain issues today is not extensively supported by tools.

- *PLM systems as means to assess and evaluate architectures during development.* The application of cost reporting in a PLM system is a tool for supporting the development of a design according to budget because derivations between budget and actual cost can be discovered quickly, thereby raising issues of choosing the best solution for a budgeted cost constraint. Additional assessment and control features in the utilised PLM system is the reporting on design status. The numbers of new, modified, and approved components are used as indicators of the progress of the design. The resolution level enables early detection and thereby of assessing the progress of design and more importantly, if not on a planned schedule with designs. Both applications have been tested and reported as having a positive impact in the case companies for assessment and evaluation support.

- *PLM systems as means for interaction/interface control and as means for allocating responsibility in product architecture design.* Interfaces are a part of many product architecture definitions and in the classic perception of modularisation it is the interface that ensures decoupling and thereby enables reuse, sharing, substitution, and serviceability. The purpose of controlling interfaces is to communicate all interfaces and interactions in a product system and allocate responsibility to interactions/interfaces. Interface control documents for specifying relation characteristics and properties can be linked to the Interface view of the architecture. Interface control documents are especially helpful where subsystems and/or modules are developed asynchronously in time, since they provide a structured way to communicate information about interfaces between different design teams. The approach has been tested in Case company #5 and the feedback from design managers has supported the contribution as having a positive impact on controlling the parallel work streams of a modular architecture development process.

5.6 Limitations of this research and its results

The anticipated boundaries and scope of this research were addressed in Chapter 1.4. This chapter describes the limitations that became apparent during and after the research process. The chapter should make it clear to the reader that I recognise the limitations of my own research, that I understand why such factors are limitations, and can point to ways of opposing these limitations if future research was carried out.

Theory based and problem based

The contributions of this research are the results of a research method conducted from a theoretical basis and in a problem-based industrial setting. The presented results are the outcome of research in action and are thus limited by not taking the researcher out of the equation, see Section 2.2.1. The method is evaluated as beneficial for reaching results that have practical applicability, but could on the other hand not have been achieved without the theoretical basis, presented in Part 3.

Experience and practice in architecture modelling

Modelling of both 'The Interface diagram' and the architectures based on the framework in Paper B, requires experience and practice. The ability to reason on and predict relations between structure and behaviour is a task that requires knowledge in engineering design and product development; training in architecture modelling is seen as a prerequisite for valuable support for the suggested approaches. The application of the proposed modelling formalisms would not have been successfully implemented without experience from the author and by providing training to designers.

Maintenance of architecture representations

In order for architecture models to have relevance, it is necessary that they are updated during the design process in order to let them reflect the most present status of the design. As the models should support evaluation and decisions of decomposition, characterisation and composition, it is critical that they also address the status of the design in progress. Reviews of architecture models are thus seen as important for capturing design choices and supporting the activities making them. Though important, the aspect of facilitating intra- and interdisciplinary engineering domain reviews has not been the focus of this research.

Keeping models of architectures up to date in computer systems can be an overwhelming manual task. This research has created diagnostic tools for maintaining architectures in a PLM system, which enables users to control alignment between architecture models inside and outside the computer-system. A limitation to this research is that it has not been possible to analyse the full resource demand for maintaining information belonging to an architecture in the later phases of development. This is mainly due to the point in time when the case study was conducted, where conceptualisation was still carried out by the design teams.

Modularisation

Modularisation is in this research seen as a means for achieving desired effects, and thus not a goal in itself. The Interface diagram interacts with the computer-support system, support modelling and evaluation of modules, but is not elaborating in detail on the relations between modularisation and desired effects. Consequently, the suggested support of interface control is more prevalent than the support in interface creation. Furthermore, PLM configuration management of product variants based on architectures has not been a major part of this research. The approach of modelling architectures in interaction with a PLM system, however, points to further progress can be made in this direction.

Implementation of approaches

Comprehensive approaches, like the ones described in this thesis, will not be adopted by a company without considerable efforts. To implement such an approach in combination with a computer-support system will typically require dedication from top-management, experienced product architects, and computer-system specialists (PLM) for training purposes. This effort requires investments in terms of

employees' time, and in the procurement and implementation of a PLM system. The approach of using architecture models in interaction with a PLM system has generated positive empirical effects in the initial testing and further potential has been argued for; nevertheless it requires an undisputable effort to implement it in companies. Hence, the effort of implementing it has to be carefully evaluated and compared to which desired effects are being pursued.

Paper based thesis and growing insight

The challenge of a paper based thesis is to establish a coherent presentation of research and results. One can argue that a paper based dissertation as a whole is less coherent than a monograph, partially because journal editors and reviewers may direct the articles separately based on the interests of the respective journals, and partially because you cannot go back and edit an already published or presented paper after you have reached some new insight with regard to your constructs. To counter the issues of incoherency, I have tried to detail explanations of contributions in this thesis, so the reader is assisted in understanding the coherence of the research. Papers that are submitted cannot be edited but their presented results and constructs can be carefully explained, which has been the motivation in this thesis.

5.7 Suggestions on future work

Reflecting on the results and conclusions, several gaps remain that provide opportunities for future research. These research efforts should be directed at the following areas:

- Study a larger amount of example problems and in particular address architectures for larger ranges of products, e.g. expanding with a proposal of handling product variety and configuration in a computer-support system.
- Interface creation and identification of critical interactions and interfaces supported by modelling formalisms and tools need to be taken to a higher level of understanding.
- Handling market and production architectures and aligning them to product architectures, supported by computer systems, need further investigation. New insights and modelling formalisms are desired.
- In order to support the evaluation of product architectures, alignment between product architectures and multiple life phases, is seen as a prerequisite for elaborating on their performance. New modelling formalisms and tools are needed.
- This research shows modelling formalisms constituted by graphical block diagrams, suited to high-level conversion of desired behaviour to solutions, decomposition, and information modelling. More visual architecture modelling techniques, suitable for supporting detailed design decisions are needed for addressing the multiple aspects of architectures in other contexts, e.g. configurability, saleability, technology preparedness, etc.
- Additional tools, metrics, and methods are needed that support designers in identifying and evaluating the relations between structural modularisation and the desired effects of a business.

5.8 Concluding remarks

The primary outcome of research is often building knowledge. Also for me, the outcome has been to build knowledge and gain a deep understanding of phenomena related to engineering design and product development.

The research work presented in this thesis represents the outcome of an exciting, but at times also very perplexing process. Having worked for several years in the industry as a designer and project manager before initiating the PhD project, I entered the project with a strong ambition to create operational support to designers.

The PhD study process has been fruitful and inspiring when meeting researchers and practitioners that both share, but also differ from, your understanding of phenomena and problems. After three years of research, I can look back on having learned methods for finding useful solutions that are based on theoretical insight and are applicable in a practical setting. Industry is the laboratory of research in engineering design research, and it has been a great pleasure to interact with practitioners in industry as well as researchers in academia, experiencing the best of both worlds.

The time span from initiation to completion has led to a giant jump in insight and knowledge of the field of engineering design and product development research. Today, I can look back and see that the process has provided me with insights and tools, which probably will influence the rest of my working life.

I hope that my enjoyment of performing the research and reporting the findings has been possible to sense when reading this thesis.

6 References

Abramovici, M. 2007, "*Future trends in product lifecycle management (PLM)*", The future of product development, pp. 665-674.

Anbari, F.T. 2003, "*Earned value project management method and extensions*", Project Management Journal, vol. 34, no. 4, pp. 12-23.

Andreasen, M.M. 1992, "*Designing on a Designer's Workbench (DWB)*", Proceedings of the 9th WDK WorkshopIrwin Professional Pub.

Andreasen, M.M. 1980, "*Syntesemetoder på systemgrundlag – Bidrag til en konstruktionsteori*", (in Danish), Department of Machine Design, Lund Institute of Technology.

Andreasen, M. 2009, "*Complexity of industrial practice and design research contributions – we need consolidation*", 20. Symposium Design for X, Neukirchen, 24.09. 2009, pp. 1.

Andreasen, M., Mortensen, N. & Harlou, U. 2004, "*Multi Product Development–New Models and Concepts*", 15. Symposium" Design for X, Neukirchen, 2004, pp. 75.

Andreasen, M.M. 2011, "*45 Years with design methodology*", Journal of Engineering Design, vol. 22, no. 5, pp. 293-332.

Andreasen, M.M., Howard, T.J. & Bruun, H.P.L. 2014, "*Domain Theory, its models and concepts*", An Anthology of Theories and Models of Design, Springer, pp. 173-195.

Andreasen, M.M. & Hein, L. 1987, "*Integrated product development*", Institute for Product Development, Technical University of Denmark, Lyngby, Denmark, 1987.

Ashby, W.R. 1957, "*An introduction to cybernetics*", Chapman & Hall, London, 1957.

Baldwin, C.Y. & Clark, K.B. 2000, "*Design rules, Volume 1: The power of modularity*", MIT Press.

Bergsjö, D., Catic, A. & Malmqvist, J. 2008, "*Towards Integrated Modelling of Product Lifecycle Management Information and Processes*", Proceedings of NordDesign 2008, pp. 253-264.

Bergsjö, D., Malmqvist, J. & Ström, M. 2006, "*Architectures for mechatronic product data integration in PLM systems*", Proceedings of the 9th International Design Conference DESIGN 2006, pp. 1065.

Blessing, L.T.M. & Chakrabarti, A. 2009, "*DRM, a design research methodology*", Springer, 2009.

Buur, J. & Myrup Andreasen, M. 1989, "*Design models in mechatronic product development*", Design Studies, vol. 10, no. 3, pp. 155-162.

Buur, J. 1990, "*A theoretical approach to mechatronics design*", Institute for Engineering Design, Lyngby, 1990.

Checkland, P. & Holwell, S. 1998, "*Action research: its nature and validity*", Systemic Practice and Action Research, vol. 11, no. 1, pp. 9-21.

Collis, J., Hussey, R. & Hussey, J. 2003, "*Business research: a practical guide for undergraduate and postgraduate students*", 2nd edn, Palgrave Macmillan, Basingstoke.

Cook, T.D., Campbell, D.T. & Day, A. 1979, "*Quasi-experimentation: Design & analysis issues for field settings*", Houghton Mifflin, Boston.

Coughlan, P. & Coghlan, D. 2002, "*Action research for operations management*", International journal of operations & production management, vol. 22, no. 2, pp. 220-240.

Czarniawska, B. & Mouritsen, J. 2009, "*What is the object of management? How management technologies help to create manageable objects*", Accounting, organizations, and institutions: essays in honour of Anthony Hopwood, pp. 157.

Denzin, N.L. & Lincoln, Y. 1994, "*Handbook of Qualitative Research*", SAGE Publications.

Du, X., Jiao, J. & Tseng, M.M. 2001, "*Architecture of product family: fundamentals and methodology*", Concurrent Engineering, vol. 9, no. 4, pp. 309-325.

Duffy, A. & Andreasen, M. 1995, "*Enhancing the evolution of design science*", Proceedings of ICED, pp. 29.

Eisenhardt, K.M. 1989a, "*Building theories from case study research*", Academy of management review, vol. 14, no. 4, pp. 532-550.

Eisenhardt, K.M. 1989b, "*Building theories from case study research*", Academy of management review, vol. 14, no. 4, pp. 532-550.

Erens, F. & Verhulst, K. 1997, "*Architectures for product families*", Computers in Industry, vol. 33, no. 2, pp. 165-178.

Ericsson, A. & Erixon, G. 1999, "*Controlling design variants modular product platforms*", Society of Manufacturing Engineers, Dearborn, Michigan.

Erixon, G. 1998, "*MFD–Modular Function Deployment, A systematic method and procedure for company supportive product modularisation*", Dissertation, KTH, Royal Institute of Technology, Sweden, 1998 .

Ernst, H., Hoyer, W.D., Krafft, M. & Krieger, K. 2011, "*Customer relationship management and company performance—the mediating role of new product performance*", Journal of the Academy of Marketing Science, vol. 39, no. 2, pp. 290-306.

Ferreirinha, P., Grothe-Møller, T., Hansen, C. & Hubka, V. 1990, "*TEKLA, a language for developing knowledge based design systems*", Proceedings of ICED, pp. 1058.

Gero, J.S., Tham, K. & Lee, H. 1992, "*Behavior - a link between function and structure i design* ", IFIP Transactions B-applications in Technology, vol. 4, pp. 193-220.

Gershenson, J.K., Prasad, G.J. & Zhang, Y. 2004, "*Product modularity: measures and design methods*", Journal of Engineering Design, vol. 15, no. 1, pp. 33-51.

Gibbert, M. & Ruigrok, W. 2010, "*The ''What''and ''How''of case study rigor: three strategies based on published work*", Organizational Research Methods, vol. 13, no. 4, pp. 710-737.

Hansen, C.L., Hvam, L. & Mortensen, N.H. 2011, "*Proactive modeling of product and production architectures*", Proceedings of ICED 11, Copenhagen, Denmark, August 2011.

Hansen, C.L., Mortensen, N.H. & Hvam, L. 2012, "*On the Market Aspect of Product Program Design: Towards a Definition of an Architecture of the Market.*", 12th International Design Conference, Dubrovnik, Croatia.

Hansen, C.L., Mortensen, N.H., Hvam, L. & Harlou, U. 2012, "*Towards a Classification of Architecture Initiatives: Outlining the External Factors*", initiatives (see Figure 1), vol. 1, pp. 2.

Hansen, C.T. & Andreasen, M.M. 2002, "*Two approaches to synthesis based on the domain theory*", Engineering design synthesis, Springer, pp. 93-108.

Harlou, U. 2006, "*Developing product families based on architectures contribution to a theory of product families*", Department of Mechanical Engineering, Technical University of Denmark, Lyngby.

Haskins, C. 2006, "*Systems engineering handbook*", INCOSE.Version, vol. 3.

Haug, A., Hvam, L. & Mortensen, N.H. 2011, "*The impact of product configurators on lead times in engineering-oriented companies*", AI EDAM-Artificial Intelligence Engineering Design Analysis and Manufacturing, vol. 25, no. 2, pp. 197.

Hölttä-Otto, K., Chiriac, N., Lysy, D. & Suh, E.S. 2014, "*Architectural Decomposition: The Role of Granularity and Decomposition Viewpoint*" in Advances in Product Family and Product Platform Design Springer, pp. 221-243.

Hölttä-Otto, K. & de Weck, O. 2007, "*Degree of Modularity in Engineering Systems and Products with Technical and Business Constraints*", Concurrent Engineering, vol. 15, no. 2, pp. 113-125.

Hubka, V. & Eder, W.E. 1988, "*Theory of technical systems: a total concept theory for engineering design*", Berlin and New York, Springer-Verlag, 1988, 291 p., vol. 1.

Hubka, V. 1985, "*Theory of Technical Systems: A new Concept in the Engineering Curriculum*", International Journal of Applied Engineering Education, vol. 1, no. 2, pp. 153-157.

Hvam, L., Mortensen, N.H. & Riis, J. 2007, "*Produktkonfigurering kundetilpasning af produkter*", 1. edition, Nyt Teknisk Forlag, Copenhagen, Denmark.

International Council on Systems Engineering & Haskins, C. 2011, "*Systems engineering handbook: A guide for system life cycle processes and activities*", International Council of Systems Engineering.

Jensen, T.A. 1999, "*Functional Modelling in a Design Support System: Contribution to a Designer's Workbench*", Department of Control and Engineering Design.

Jiao, J. & Tseng, M.M. 2000, "*Fundamentals of product family architecture*", Integrated Manufacturing Systems, vol. 11, no. 7, pp. 469-483.

Jörgensen, K. 1992, "*Videnskabelige arbejdsparadigmer*", (In Danish), Institute for Production, Aalborg Univesity, Denmark.

Klir, J., Valach, M. & Klir, J. 1967, "*Cybernetic modelling*", Liffe Books, London, 1967.

Kratochvil, M. & Carson, C. 2005, "*Growing modular: mass customization of complex products, services and software",* Springer, 2005.

Kumar, V. 2010, "*Customer relationship management*", Wiley Online Library.

Kvist, M. 2009, "*Product Family Assessment",* PhD dissertation, Department of Management Engineering, DTU, 2009.

Larkin, J.H. & Simon, H.A. 1987, "*Why a diagram is (sometimes) worth ten thousand words*", Cognitive science, vol. 11, no. 1, pp. 65-100.

Latour, B. 1986, "*Visualization and cognition*", Knowledge and society, vol. 6, pp. 1-40.

Li, S., Chen, J., Yen, D.C. & Lin, Y. 2013, "*Investigation on auditing principles and rules for PDM/PLM system implementation*", Computers in Industry, vol. 64, no. 6, pp. 741-753.

Lindemann, U., Maurer, M. & Braun, T. 2009, "*Structural complexity management*", Springer Verlag.

Malmqvist, J. 2001, "*Implementing requirements management: A task for specialized software tools or PDM systems?*", Systems Engineering, vol. 4, no. 1, pp. 49-57.

McCarl, B.A. 1984, "*Model validation: an overview with some emphasis on risk models*", Review of Marketing and Agricultural Economics, vol. 52, no. 3, pp. 153-173.

Meyer, M.H. & Lehnerd, A.P. 1997, "*The power of product platforms building value and cost leadership"*, The Free Press, New York.

Miller, T.D. 2001, "*Modular engineering ; An approach to structuring business with coherent, modular architectures of artifacts, activities, and knowledge"*, Technical University of Denmark, Department of,Mechanical Engineering; DTU, Lyngby.

Mortensen, N.H., Gamillscheg, B., Bruun, H.P.L., Hansen, C.L., Hansen, K.K. & Junkov, K.H. 2012, "*Radikal Forenkling via Design*", 1st edn, DTU Mekanik, Kgs. Lyngby.

Mortensen, N.H. 1999, "*Design modelling in a designer's workbench contribution to a design language"*, Department of Control and Engineering Design, Technical University of Denmark, Kgs. Lyngby.

Mørup, M. 1993, "*Design for quality*", PhD dissertation, Institute for Engineering Design, Technical University of Denmark.

Olesen, J. 1992, "*Concurrent development in manufacturing - based on dispositional mechanisms*", Dissertation from the research program Integrated Production Systems, Lyngby.

OMG 2014, Object Management Group. Available: http://www.omg.org/spec/.

Otto, K. & Wood, K. 2001, "*Product Design - Techniques in reverse engineering, systematic design, and new product development*", Prentice Hall, Upper Saddle River, NJ, USA, 2001

Pahl, G., Beitz, W. & Wallace, K. 1996, "*Engineering design A systematic approach"*, 2.rev edn, Springer, London.

Pedersen, K., Emblemsvag, J., Bailey, R., Allen, J.K. & Mistree, F. 2000, "*Validating design methods and research: the validation square*", ASME Design Theory and Methodology Conference, 2000.

Pedersen, R. 2009, "*Product Platform Modeling*", PhD dissertation, Department of Management Engineering, DTU, 2009.

Pimmler, T.U. & Eppinger, S.D. 1994a, "*Integration analysis of product decompositions"*, Alfred P. Sloan School of Management, Massachusetts Institute of Technology.

Pimmler, T.U. & Eppinger, S.D. 1994b, "*Integration analysis of product decompositions*", American Society of Mechanical Engineers, Design Engineering Division (Publication) DE, vol. 68, pp. 343-351.

Pine, I.,B.J. 1999, "*Mass Customization: The New Frontier in Business Competition (Paperback)*", Harvard Business School Press Books, .

Pine, I., Joseph, B. & Victor, B. 1993, "*Making mass customization work*", Harvard business review, vol. 71, no. 5, pp. 108-117.

Prasad, B., Wang, F. & Deng, J. 1997, "*Towards a computer-supported cooperative environment for concurrent engineering*", Concurrent Engineering, vol. 5, no. 3, pp. 233-252.

Rahmani, K. & Thomson, V. 2011, "*Managing subsystem interfaces of complex products*", International Journal of Product Lifecycle Management, vol. 5, no. 1, pp. 73-83.

Robertson, D. & Ulrich, K. 1998, "*Planning for product platforms*", Sloan management review, vol. 39, no. 4.

Roozenburg, N.F. & Eekels, J. 1995, "*Product design: fundamentals and methods*", Wiley Chichester.

Sääksvuori, A. & Immonen, A. 2008, "*Product lifecycle management"*, Springer Verlag, 2008.

Sanchez, R. & Mahoney, J.T. 1997, "*Modularity, flexibility, and knowledge management in product and organization design*", IEEE Eng Manage Rev, vol. 25, no. 4, pp. 50-61.

Shaul, L. & Tauber, D. 2013, "*Critical success factors in enterprise resource planning systems: Review of the last decade*", ACM Computing Surveys (CSUR), vol. 45, no. 4, pp. 55.

Simon, H.A. 1981, "*The sciences of the Artificial*", Massachusetts Institute of Technology, 1. edn. 1969.

Simpson, T.W., Maier, J.R. & Mistree, F. 2001, "*Product platform design: Method and application*", Research in Engineering Design - Theory, Applications, and Concurrent Engineering, vol. 13, no. 1, pp. 2-22.

Stark, J. 2011, "*Product lifecycle management*", Product Lifecycle Management, pp. 1-16, Springer.

Stark, J. 2007, "*Global product: Strategy, product lifecycle management and the billion customer question"*, Springer Verlag.

Steward, D.V. 1981, "*The design structure system: A method for managing the design of complex systems*", IEEE Transactions on Engineering Management, no. 3, pp. 71-74.

Sudarsan, R., Fenves, S.J., Sriram, R.D. & Wang, F. 2005, "*A product information modeling framework for product lifecycle management*", Computer-Aided Design, vol. 37, no. 13, pp. 1399-1411.

Svendsen, K. 1994, "*Diskretoptimering af sammensatte maskinsystemer*", (In Danish), Dissertation, Institute of Engineering Design, DTU, Denmark.

Svensson, D. 2003, "*Towards product structure management in heterogeneous environments*", Chalmers University of Technology.

Tjalve, E. 1976, "*Systematisk udformning af industriprodukter. Værktøjer for konstruktøren*", (In Danish), Akademisk Forlag, Copenhagen.

Tomiyama, T., Kiriyama, T., Takeda, H., Xue, D. & Yoshikawa, H. 1989, "*Metamodel: a key to intelligent CAD systems*", Research in engineering design, vol. 1, no. 1, pp. 19-34.

Ulrich, K.T. & Eppinger, S.D. 2011, "*Product design and development"*, McGraw-Hill.

Ulrich, K. 1995, "*The role of product architecture in the manufacturing firm*", Research Policy, vol. 24, no. 3, pp. 419-440.

Ulrich, K.T. & Eppinger, S.D. 2004, "*Product design and development"*, 3. edn., international edn, Irwin McGraw-Hill, Boston, Mass.

Ulrich, K. & Tung, K. 1991, "*Fundamentals of product modularity*", American Society of Mechanical Engineers, Design Engineering Division (Publication) DE, vol. 39, pp. 73-79.

Weber, C. 2014, "*Modelling products and product development based on characteristics and properties*", An Anthology of Theories and Models of Design, Springer, pp. 327-352.

Weber, C., Werner, H. & Deubel, T. 2003, "*A different view on Product Data Management/Product Life-Cycle Management and its future potentials*", Journal of Engineering Design, vol. 14, no. 4, pp. 447-464.

Yin, R.K. 2009, "*Case study research: Design and methods"*, Sage, 2009.

7 Appended papers

7.1 Paper A: Interface diagram: Design tool for supporting the development of modularity in complex product systems

Starting at page A1.

7.2 Paper B: *Identification of a scalable architecture for customization of complex parts*

Starting at page B1.

7.3 Paper C: *PLM support for development of modular product families*

Starting at page C1.

7.4 Paper D: *PLM system support for modular product development*

Starting at page D1.

7.5 Paper E: *Mapping requirements to a product architecture supported by a PLM system*

Starting at page E1.

Paper A

Bruun, H.P.L., Mortensen, N.H., Harlou, U.,
Interface diagram: Design tool for supporting the development of modularity in complex product systems,
Journal of Concurrent Engineering, Research and Applications, published online 29 December 2013.

Original Article

Concurrent Engineering: Research and Applications
2014, Vol. 22(1) 62–76

DOI: 10.1177/1063293X13516329
cer.sagepub.com
SAGE

Interface diagram: Design tool for supporting the development of modularity in complex product systems

Hans Peter Lomholt Bruun[1], Niels Henrik Mortensen[1] and Ulf Harlou[2]

Abstract

For products with a myriad of systems, groups of specialised engineers develop entire technical sub-systems, and great effort is needed to integrate these systems for fulfilling the product's intended properties describing its purposeful behaviour. This way of developing products gets even more complex when using a mass customisation strategy because standard designs (reusable modules) have to be designed to fit a range of products. This product development set-up requires that engineers working in different technical domains collaborate and are able to share information in a unified way. This article presents a visual design tool –*the Interface diagram*– which aims to support the engineering process of developing modularity in complex product systems. The tool is a model of a product system representing the arrangement of its elements and their interfaces. The tool has similar characteristics to a high-level product architecture model, aiming at supporting integration of technical sub-systems by documenting interfaces and interactions among components from different functional sub-systems and among different physical modules. One of the objectives for using the design tool is to support the activity of decomposing a product system into modules consisting of components developed by different engineering teams. The usefulness of the Interface diagram has been tested in an industrial development project showing positive results of shortening the lead time and minimising rework. Moreover, the Interface diagram has been used in interplay with a broader Product Lifecycle Management system. This allows the product structures from the Interface diagram to be enriched with detailed product documentation like computer-aided design, requirements, view models, design specifications and interface descriptions.

Keywords

Architecture-based design, system design, modularisation, concurrent engineering, Product Lifecycle Management

Introduction

For products with a vast amount of sub-systems, each related to one or more of a product's main functions, groups of specialised engineers develop discrete technical sub-systems, and great effort is needed to integrate these systems for fulfilling the product's intended properties. This way of developing products gets even more complex when using a mass customisation strategy, as standard designs have to be designed to fit a range of products in a product program. Development of product families is a multidisciplinary activity involving inputs from a diversity of technical domain experts, production personnel, sales personnel, service personnel and so on. Different departments in a company have different understandings of product families from their individual perspectives. Such incoherence in semantics and subsequent deployment of information is a barrier for concurrent engineering, and the activity of developing reusable solutions crosses technical

[1]DTU Mechanical Engineering, Technical University of Denmark, Lyngby, Denmark
[2]Center for Product Customization (CPC), Technical University of Denmark, Lyngby, Denmark

Corresponding author:
Hans Peter Lomholt Bruun, DTU Mechanical Engineering, Technical University of Denmark, 2800 Lyngby, Denmark.
Email: hplb@mek.dtu.dk

domains. Managing complexity in this development setting is given significant attention in all kind of industries. The concept for coping with complexity in product development is often to break down systems or problems into manageable parts, referred to as modularisation. However, despite many reported success stories, there still seems to be confusion in industry about managing and supporting modularisation activities (Jiao and Tseng, 2000). This is to a certain extent attributable to different understandings and definitions of modularisation, and partially due to there being few existing formalised methods and tools representing different perspectives of products and family of products in a single context (Bruun and Mortensen, 2012a). Another core problem from a practical point of view is the lack of ability to visualise complex product systems, for example, product families, and to use them actively for design and decision-making purposes (Pedersen, 2009). The basic assumption, in respect to this research, is that decisions concerning product modularity must be enabled by means of explicit and visual models. Hereby, the 'complexity' becomes manageable, and a project team acquires a shared product system definition. The implication of this assumption is that models, which visualise several aspects of a product system, enable better decision-making concerning the design of modules and interfaces between them. The aim of this research is to provide companies with an approach and a design tool to support engineering teams to lay out the architecture of a product system's sub-systems. The aim is moreover to support modularisation, that is, encapsulating different system components into optimal modules. This article introduces a visual product model named Interface diagram. The Interface diagram represents in a graphical way different aspects of a product system and aims to support high-level decision-making related to system integration and modularisation during the design process. The concept of the Interface diagram has been presented in previous work of the authors (Bruun and Mortensen, 2012b), but this article describes in detail the experiences gained by application in a real industrial development project. The Interface diagram can represent product variants, different technical sub-systems, standard designs (modules) and their relations. The industrial setting for this study has been a large Danish manufacturer of complex products for the renewable energy sector. The company has had a traditional focus on mechanical engineering, but evolution in their product's technologies has now brought multiple technologies into play when developing products. The research is based upon action research carried out for 5 months in a project with more than 150 engineers working in five global research and development (R&D) locations.

This article is structured as follows: first, an introduction to significant contributions to the research area of product models for representing complex product systems, for example, product families. Then, the formalism of the proposed design tool is described. The following sections elaborate on the results by applying the Interface diagram in combination with a Product Lifecycle Management (PLM) system. A discussion extracts principals, relationships and generalisations of the results. Finally, a conclusion sums up the results of implementing the model in a company setting and outlines perspectives for future work.

Adjacent research

The design approach described in this article builds upon the creation of a visual model handling structural and functional aspects of product systems. This approach will primarily reserve the word 'structure' to how individual products are built up, and 'architecture' will be reserved for describing how a product family is built up. Contributions in the research area of visual product models and architectures handling product variety are included in this section.

The visual aspect of product models

Several authors report success with the use of some sort of visual modelling as a means to support designers and decision-makers (Hansen et al., 2011; Kvist, 2009; Mortensen et al., 2012; Pedersen, 2009; Tjalve, [1976] 1989). They acknowledge the usefulness of visual modelling and graphical representations of products with sketches, drawings, diagrams and so on as a powerful means to hold information and to share knowledge between designers and decision-makers. Visual is meant in the way that an overview is easily obtained from the model. A common characteristic for the above references is their acknowledgement of the problem of presenting data and information in a visually presentable way without losing the depth of the information. Product models are often related to a development methodology or a certain approach, and there is a strong correlation between the models and the nature of the development methodology (Pedersen, 2009).

Product models

Product models have either a structural or a functional starting point. Modelling functions is a way of representing products on a relatively abstract level. Formulating functions is a starting point for establishing design solutions. A product structure is viewpoint dependent; still some sort of assembly hierarchy often determines the decomposition and sequence of parts.

Product structures focus more on relations (mainly geometrical/attachment) between elements rather than the flow of information, material or energy, which is often a key aspect of function models. The function-based design methods (*Function structures*) are characterised by establishing either a function model (Otto and Wood, 2001; Pahl et al., 1996) or the schematics of the product (Ulrich and Eppinger, 2004). Both approaches have a visual representation as an outcome. The function structure describes the flow of material, data and energy through sub-functions of the product using a set of rules (e.g. the rules that are referred to as the functional basis which basically is a common language to describe functional elements). The schematic of the product is somewhat similar to the function model. But where the function model describes the product using functional elements, the schematics can describe both functional and physical elements – whichever is the most meaningful for the purpose of the representation. The functional structure forms the basis for several different approaches to design or redesign the products. In general, the strength in these methods lies in the use of the functional viewpoint on the product structure, that is, functional structure. A function flow diagram can be effective in revealing functional modularity, but has less strength in the encapsulation of physical parts which have behaviour due to relations in the part domain (interactions in and between parts) that are not clearly modelled. The Interface diagram has its origins in the Generic Organ Diagram (Harlou, 2006). The Generic Organ Diagram uses the viewpoint of organs from the Theory of Domains (Andreasen, 1980) instead of functions. Organs are mapped, and not only are input, output and flow emphasised but also the interactions between the organs. The Interface diagram also builds upon the modelling framework Product Family Master Plan (PFMP) (Harlou, 2006). The PFMP is a visual object-oriented modelling approach for representing commonality and variety in product families. The basic idea behind the PFMP is to gather large quantities of information in a visual way on a poster in order to present an overview. The object-oriented approach makes it possible to gather the information of several product variants in one model without requiring redundant data. The Interface diagram aims to gather information from several product variants. The Design Structure Matrix (DSM) method is an approach for modelling complex relations between entities in a product or product family. The DSM is perhaps not a decidedly visual model, but it does have the ability to hold a large amount of information for complex systems and make it available for retrieval. The approach takes a starting point in the decomposition of a product into components/systems and an identification of interfaces/relations among these (Hölttä-Otto and de Weck, 2007; Pimmler and Eppinger, 1994). Sub-systems are mapped against each other for correlation purposes, in order to cluster sub-systems that are closely interrelated and to separate those that are not. By the use of algorithms, it is possible to encapsulate components into modules or chunks that are closely related to each other from an interaction point of view (Steward, 1981). This process is referred to as clustering. The outcome of a DSM can be a proposal of a future modular product architecture. The Interface diagram does not have the ability to cluster components by the use of algorithms. Instead, it uses visual capabilities to show the complex relations in a product system and thereby support decisions on encapsulation of components in modules. In this way, the Interface diagram is seen as a more easy to use tool than DSM tools. In the line of models that handle product variety is the *Generic Bill-of-Material* (BOM). The Generic BOM originates from the assemble-to-order environment (Van Veen and Wortmann, 1987). The end products typically have a number of features for which a number of options are available to choose from. A specific BOM is generated from the generic BOM by replacing the generic items with specific items. When all generic items are replaced by specific items, the product is unambiguously defined. The generic BOM is used to describe related products in one all-embracing model by using generic and specific items. The idea of modelling a generic product structure for a system of products is embraced by the Interface diagram. The alignment of the Interface diagram with a broader PLM system creates functionalities for configuring specific variants of a family. A product variant can be configured in the PLM system because of generic interfaces, where components can be interchanged to form a specific variant.

Conclusion on product modelling contributions

The review of existing methods and models reveals a multitude of different visualisation approaches and models for assessment and development support of product designs and flow in operations in relation to single products and product families. All the methods can play a role in identifying structural aspects of architectures and some of them also take functional aspects into consideration. Common to all the methods is that they do not address the use of product models to support interdisciplinary, modular development in a design process from concept to detailed design. The models can generally be divided into two groups: (a) methods that are relatively visual but only focus on aspects related to a single or few requirements (Functional structures, PFMP, Generic Organ Diagram) and (b)

methods that have a broad focus and touch many aspects but lack visual qualities (DSM, Generic BOM).

The Interface diagram

This section introduces the framing of the proposed approach for modelling complex product systems. The framing describes the aim and the dilemma the researchers are trying to resolve and shares the assumptions about the situation of the researched case. After introducing the framing, the Interface diagram modelling formalism is presented. A mobile power loader (Bobcat) is used whenever possible to illustrate and to exemplify. The case company is anonymised for competitive reasons and in order to be able to report more interesting details than a public case allows for.

Framing the approach

The Interface diagram is a visual design tool for decomposing a product system into sub-systems, modules, components and interfaces. Application of the tool for analysis and synthesis is a proposal for supporting the engineering process when developing complex product systems. One of the objectives for deploying the Interface diagram is that it should provide the designer with an aid to decompose, characterise and compose product systems. Another equally important objective is to enable a computer-based tool to support this in interplay with the Interface diagram. The Interface diagram is one leg of the suggested support, and the computer system (PLM system) is the other. This article focuses on describing the Interface diagram, while an additional article will focus on the interplay with a PLM system. The aim is to speed up the design process and enhance design quality by using a visual overview. The Interface diagram is treated as a shared information model of the product system. Having a shared information model in a product development team has been recognised as a prerequisite for project members to work at a high degree of concurrency in a distributed environment (Prasad et al., 1997). The Interface diagram has similar characteristics with the architecture definition made by Ulrich (1995). Ulrich defines product architecture as the arrangement of functional elements, the mapping from functional to physical elements and the specification of interfaces between interacting physical components. The Interface diagram is representing functional properties and structural characteristics of a product system, mostly combining the aspect of mapping between technical domains, and mainly a mapping between functional and physical elements. The Interface diagram puts emphasis on managing technical interfaces between the modelled entities, hence the chosen name of the modelling tool. The tool puts emphasis on handling a product system seen from different viewpoints. The main viewpoint is a systems perspective, that is, the perspective that deals with the product's main functions or one of its related life cycles. A simplified example of a Bobcat is used to illustrate this. Figure 1 shows two systems in the Bobcat: the hydraulic system and the drive train system. The two systems consist of different components connected by interfaces (here structural components) such as hydraulic hoses, electrical wires and transmissions belts.

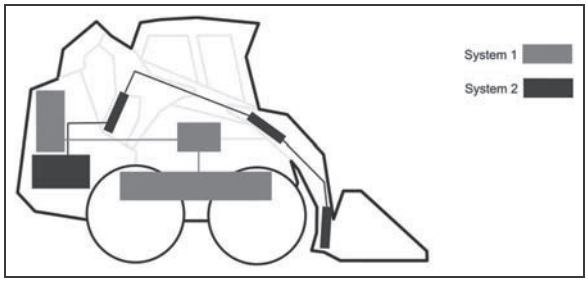


Figure 1. System structure of a Bobcat showing entities belonging to systems and relations between them.

As seen above, the components included in the system can be spread throughout the complete vehicle. The purpose of systems is to support in developing functionality in components that are spread across multiple modules. Systems are often characterised by one or more of the following:

- Deliver important functionality, for example, steering, braking and loading.
- Is a complex or new technology, for example, hydraulics and cooling.
- Organisation – alignment with organisational structure and/or affecting organisational structure.

The objective of the system design is to focus the design on functionality and performance. The same component can, from a functional perspective, be a part of multiple systems as illustrated in Figure 2, where the coloured boxes represent components and the lines between them are interfaces.

The second viewpoint is a modular viewpoint in which physically joined components are encapsulated into modules. Modules can consist of components belonging to different systems, that is, developed by different engineering teams. It is therefore necessary to integrate system components into modules. Figure 3 shows the Bobcat where the two systems are divided into three modules: base frame module (C), engine module (B) and hydraulic module (A).

A given module is composed of components and/or other modules. Unlike sub-systems, a component can only be part of one module. The main reason for

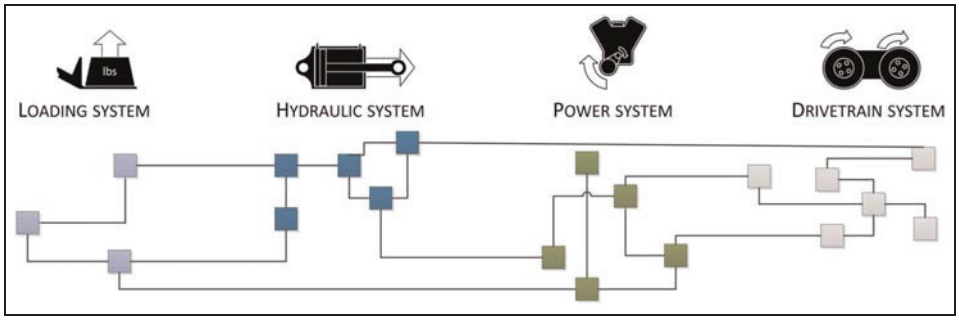


Figure 2. Components related in a functional system context.

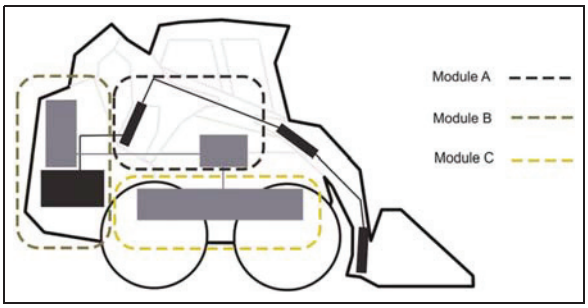


Figure 3. Module structure in a Bobcat showing entities belonging to modules and relations between systems and modules.

having modules is to find the best physical arrangement of a component that will optimally support the product life cycle; some examples can be manufacturability, encapsulation of complexity, upgradability either for current program or aftermarket needs and serviceability. A given module can therefore have components that are part of multiple sub-systems as illustrated in Figure 4.

A module definition could be that it should comply with the following rules:

- *Active choice*. A module is not a module before it has been actively decided upon.
- *Documentation*. A module has to be documented in such a way that it is possible to implement the module for novices.
- *Responsibility*. The module organisation has the continuous ownership of the module.
- *Continuous development*. A module is continuously developed and frequently updated.

Both perspectives on the product system are considered necessary. The objective of working with sub-systems is to address functional decomposition of the product system and to ensure realisation of functionality in sub-systems and in the product system as a whole. It makes sense to use this sub-system decomposition because functionality often is realised in a combination of different means allocated in different physical modules. The objective of working with a module view is to conceptualise the physical encapsulation of components. The Interface diagram provides freedom to evaluate different module concepts in relation to module drivers and functionality and to explore the effect of a module candidate. A module creation is decided on according to one or more of the module drivers. A sub-system is equally a product of an active decision

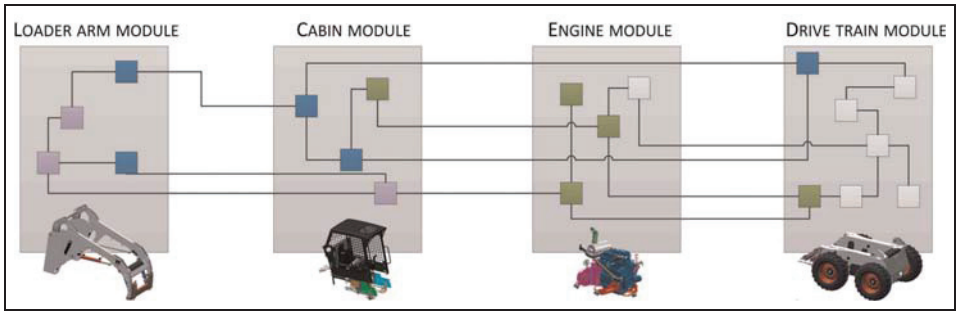


Figure 4. Components in a modular context.

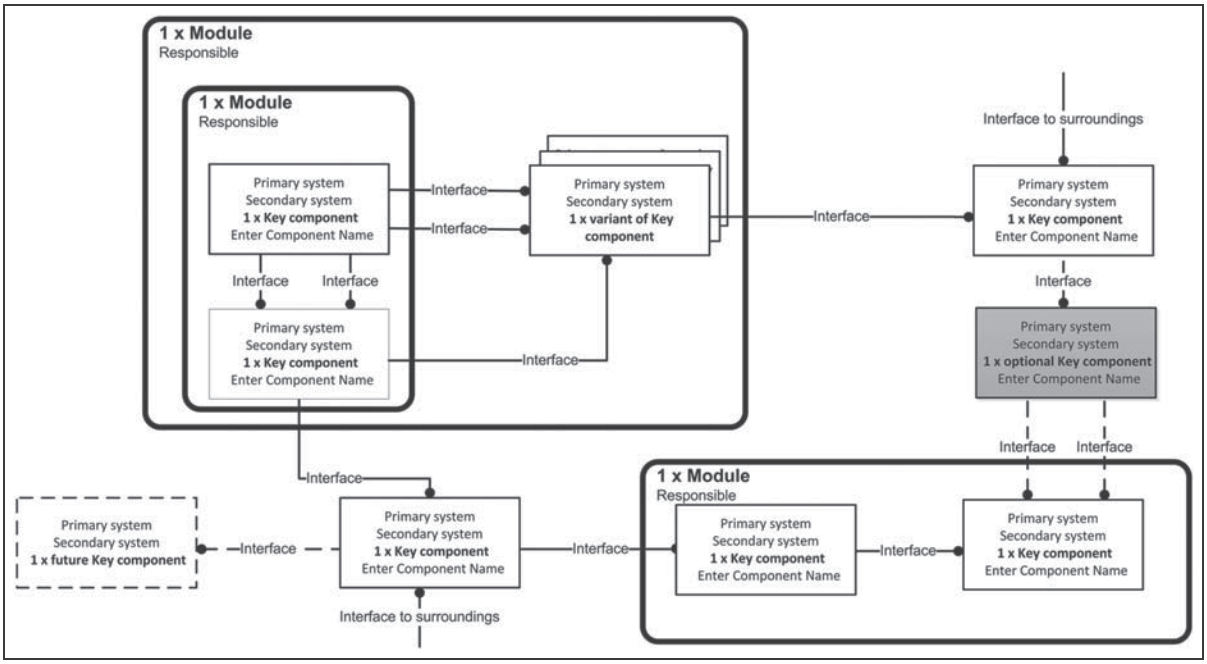


Figure 5. Symbolic representations of an Interface diagram and the types of Key components.

process, but should not be understood in realisation of best physical arrangement of components, but in realisation of best functionality of the product system as a whole.

Modelling formalism. The following sections describe the Interface diagram modelling formalism and the process for creating and maintaining the product system representation. The Interface diagram is modelled by means of blocks and lines in the software program Microsoft Visio. The Interface diagram is printed on large blueprints in order to get the overview of the product system and to allow for stakeholders to write and draw directly on the modelled structures during meetings. The concept of modelling a product system by means of an Interface diagram is a technique that does not assume that the structure of the model is fixed from the start. Typically, the model grows in size to accommodate decomposition and to represent the results of synthesis. Figure 5 is a symbolic representation of an Interface diagram. The main elements of the diagram formalism are objects denoted as *Key components*. The purpose of the Key components is to decompose the product system into smaller building blocks. Key components can have different characteristics and are thus modelled in different ways (see Figure 5). The Interface diagram is a supportive tool in the entire design process because it models the product system from an early phase – even before actual components have been developed or decided on. In the requirement specification phase and the conceptual design phase when few or no components have been developed or chosen, the Interface diagram has only a few elements. The nature of the modelled elements in the early phases is functional elements or organs, that is, elements that hold and create functionality described by requirements to properties and characteristics. In this phase, functional modularity and interactions between organs are used to seek solutions and to evaluate module candidates. Later in embodiment and detailed design phases, where functions have been realised by means (components), these are the elements that are modelled in the Interface diagram. When physical modularity is the objective of the design, interface creation and module composition are the focus of the support from the Interface diagram. Thus, a Key component represents in the early phases a functional element and later in the design process a generic physical assembly or part of the product system and is symbolised by a white block. An optional assembly or part is symbolised by a grey block. A future assembly or part which has to be taken into consideration in the product system, but has not been developed yet, has a dotted line around a white block. Finally, a variety of assemblies or parts are modelled by placing blocks on top of each other with a little offset. This representation shows that the specific Key component has at least two variants. Each Key component belongs to one or more functional sub-systems. To avoid confusion, sub-systems and their interaction must be clearly

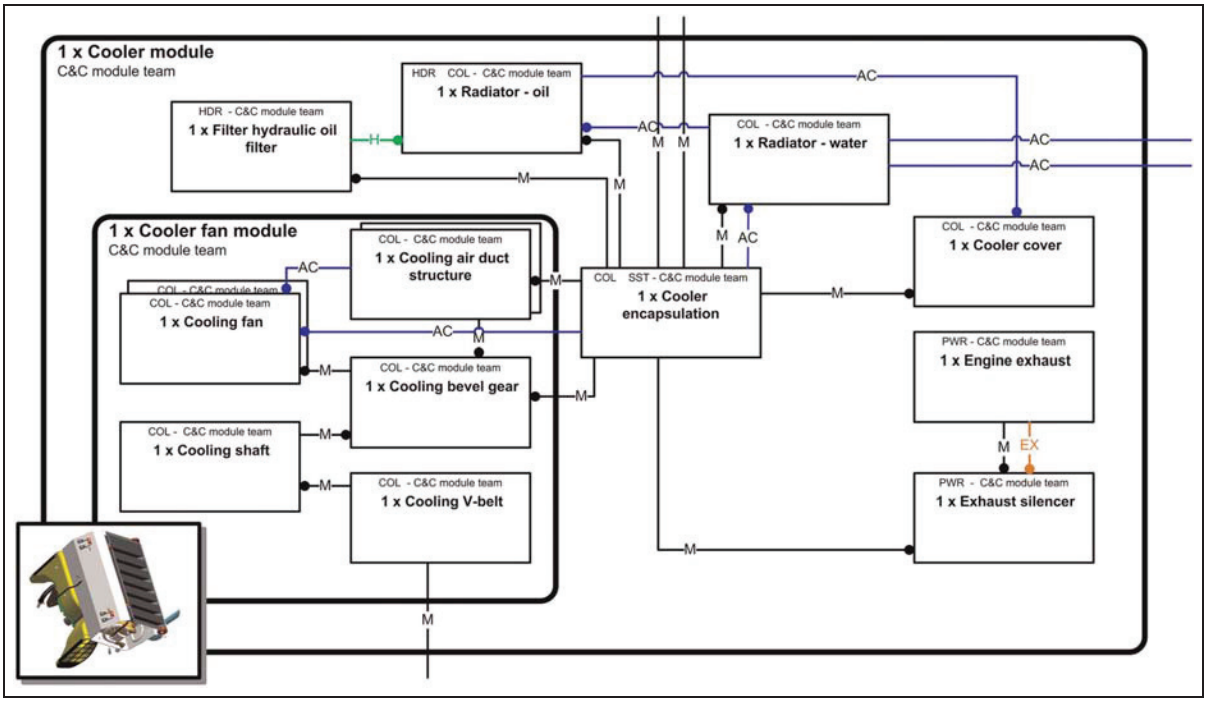


Figure 6. Excerpt from Bobcat Interface diagram.

defined. This is done by choosing the relevant interaction as a basis for determining the system boundary. There is no fixed list of sub-systems to be included in the development. Sub-systems can be modelled and thereby control important properties of the final product. Each block has a designation relating to the sub-system relationship. A primary sub-system means that the responsible system team designs the underlying parts and holds the responsibility for the functionality, while a team with secondary sub-system responsibility has requirements to the design. An example could be that a cooling unit is designed by the team responsible for cooling and conditioning, while the sub-system responsible for the engine has requirements to the cooling performance they need for their engine design.

Modules are modelled by arranging Key components inside boxes with a thick black boundary and rounded corners (see Figure 5). Modules can contain smaller modules, but they do not overlap as it is clearly defined to which modules any component in a product belongs. Figure 6 shows an excerpt from the Interface diagram of the Bobcat. Modules consist of Key components belonging to different systems. The interface classes illustrated are mechanical, hydraulic, air cooling and exhaust gas.

The structure of the interface diagram appears as the relations are added to the diagram. The relations between Key components are drawn with lines which represent an interface or an interaction. Interfaces and interactions are linkages shared among components, modules and sub-systems of the product system. Interfaces between two Key components represent physical relations, for example, constituting physical connections. Interactions between two Key components represent the transfer of material, energy (forces, movement) and information. The interaction may be transferred via an interface, but can also be 'touch less' or indirect. The purpose of working with relations is to ensure responsibility for the component interaction and to ensure that components are interchangeable, when relevant. In a product modelling context, the relation has to belong to a common structure, or some sort of generic placeholder, in order for the relation to be inherited to the involved elements. In every set of Key components, it is defined which is the master and which is the slave. This enables whoever is responsible for a Key component to monitor whether he owns the right to change or modify the relation. In a modularisation context, all interfaces and interactions between different modules and the outer environment should be carefully specified in order to complete the module. In addition, there exist a number of interface classes, and the list can be extended in order to support the context in which a product belongs. Examples of interface and interaction classes are as follows: mechanical, spatial, cooling air, cooling liquid, electrical measurement, electrical signal,

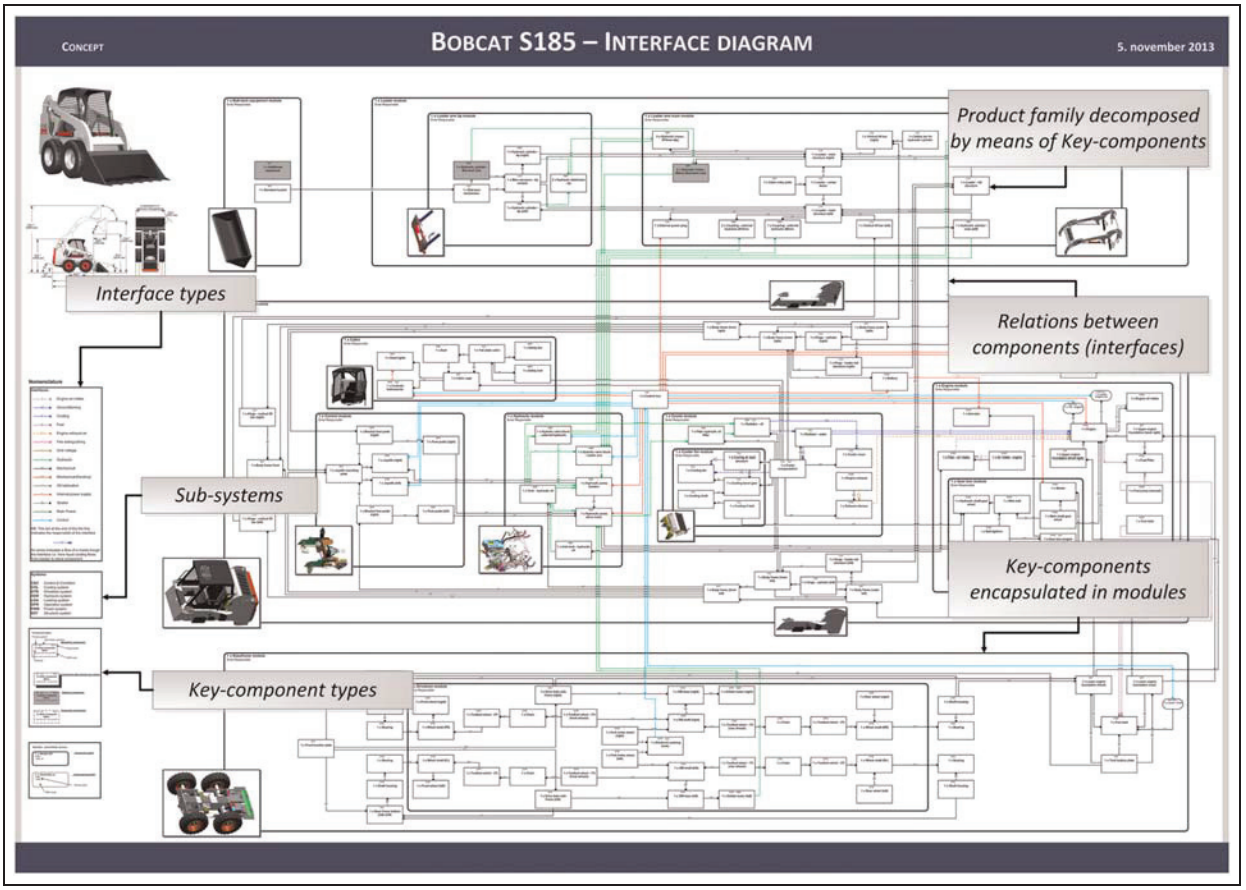


Figure 7. Overview of elements in an Interface diagram for a Bobcat.

electric grid and energy. A dot at the end of the line indicates who is responsible for the relation, that is, the master and slave relationship. Optional interfaces and interactions can be modelled on the diagram to show affected entities if the relation is established. Interfaces are a part of many product architecture definitions, and in the classic perception of modularisation, it is the interface that ensures decoupling and thereby enables reuse, sharing, substitution and serviceability (Sanchez and Mahoney, 1997). In principle, a line is drawn for each interface in the model, but for relations like electronic products that would mean a myriad of lines. In such cases, the interface diagram is simplified by only drawing one line for each type of interface between the related Key components. The interfaces modelled in the Interface diagram are documented by functional and physical requirements that the interacting elements have to accommodate. Some interfaces have more characteristics of interactions. An interface description can be a description of how the interfaces are to interact. This differentiation between the word interface and interaction is also acknowledged in other product models (Gedell et al., 2011). The design specifications for interfaces are named *IF descriptions*. The purpose of the IF descriptions is comparable to *Interface control documents* (ICD) which are the key element in product architecture design in system engineering (Haskins, 2006). The purpose of an IF description is to communicate all possible inputs and outputs from a sub-system. All IF descriptions are linked up to the interface structure when the Interface diagram is uploaded to the PLM system. Figure 7 presents an Interface diagram for a Bobcat product system with explanations of the overall layout and content. In order for the reader to quickly understand the structure of the product, it is suitable to model the diagram so the layout is established as a cross section of the actual product. In that way, the physical layout of the product can be recognised in the diagram. The diagram can be read following the interfaces. For products that process or transform objects, this gives a logical reading direction; for other products, it is up to the reader to find a suitable flow in the model.

When designing a complex product system, multiple life-cycle Design for X (DFX) issues have to be considered simultaneously. Conflicting DFX issues have to be balanced through trade-offs, which requires metrics and measures for dealing with these conflicts. Metrics are used extensively in the product development process by product development teams. Metrics and measures for assessing conflicting life-cycle DFX issues can be arranged in four groups (Prasad, 1997): simulation and analysis, product feasibility and quality assessment, DFX ability assessment and process quality assessment. The Interface diagram, in combination with other metrics and measures, is used for assessing a product system in terms of modular design, design for interchangeability, design for compatibility, design for variety, design for flexibility and so on.

Experience from implementation and application

This section describes the studied company's situation and the reasons for implementing a modular development approach that is supported by the Interface diagram. In addition, the implementation of the Interface diagram and outcome of the application are described.

Company situation

As described in the 'Introduction' section of this article, the researched company is a Danish manufacturer of complex products to the renewable energy sector. The company is currently developing the next generation of a family of products, which would be larger and more complex than before. The company has in recent years been under pressure from the market to deliver a broader and more customised product assortment. In addition, the customers have an increasing focus on low cost and fast delivery time. By using a modular product strategy, the company has strived to rationalise their development process and to produce a larger variety of products at lower cost. The modularisation initiative of developing reusable modules demanded a great deal of coordination between different developer teams: Product Management, Production, Purchase & Supply, Transport, Service, Installation, Operation and Systems Integration. The company had chosen what they thought was a pragmatic and agile solution for coordinating development activities and documentation of products in an Excel file. There were good intentions behind the strategy of getting control of product data and supporting coordination. The actual outcome of the initiative was, however, frightening for the company, whose goal was to break down complexity and to lower lead time in development. Quickly, the Excel repository exploded in size. The Excel file had approximately 7.500 $\times$ 500 cells. The monthly manual update time was approximately 100 h (input was given by approximately 100 engineers), and still, product definitions in separate sub-systems had to be maintained. Working with this platform for supporting a modular product strategy for some years, the company encountered operational challenges, as the data transparency was more or less lost. The reasons for implementing a new approach were, among others, as follows:

- The diversity of methods for documenting products by different technical domain experts made it hard to see dependencies between different technical sub-systems. Development activities as such were difficult to decouple, and it became challenging to perform concurrent development.
- Nobody had the grant overview, and as such, it was challenging to balance designs and integrate them in a whole.
- The different way of documenting products made it difficult to recognise and manage modules and interfaces.
- The dispersed product definitions made it difficult for designers to find and reuse existing modules. It was unclear what was standard and what was customised for a specific project or product, which again could compromise the integrity of standard designs and their interfaces.
- Because development was done in many platforms, it was difficult to carry out proper monitoring of relational properties and design progress during project execution.

Implementation and application of the Interface diagram

With these reasons in mind, the Interface diagram in combination with a PLM system was implemented to create a common platform for representing the company's products and the complex relations between variants, technical sub-systems, physical modules and interfaces. One of the research hypotheses is that companies following a multi-product development strategy can use PLM systems to handle the underlying logic of modular architectures, that is, modules, interfaces and design rules. Our approach to use the Interface diagram in interplay with a PLM system is outlined in Figure 8. This article focuses on the creation and application of the Interface diagram, whereas the steps of handling product structures and product data in the PLM system will be covered here, but described in detail in a future planned article.

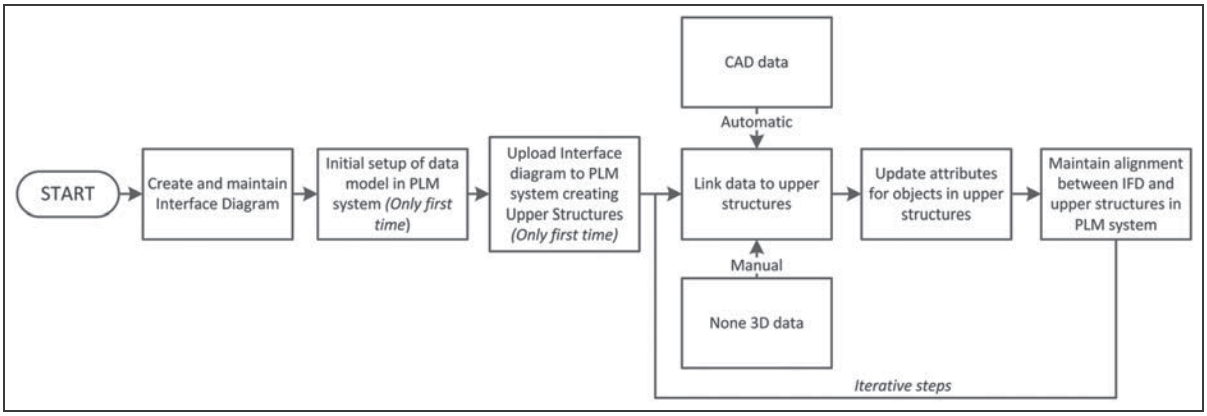


Figure 8. Steps in the approach of supporting modularisation in product system development.
CAD: computer-aided design; PLM: Product Lifecycle Management; 3D: three-dimensional.

The Interface diagram is a dynamic tool which is updated and refined during the project's life cycle. The Interface diagram describes the design during the design process, that is, it describes the in-progress design. The tool is both descriptive (documenting the design at a point in time) and prescriptive (supporting the designer in coming from the imagination of a product's functions, utilisation and properties to the design of its way of working). The technique for modelling the Interface diagram is based on interviewing the persons with the insight to model sub-systems in their totality. No single person in the company had the required insight to draw the diagram. Therefore, several domain experts had to be involved in giving input to the diagram. It took several months to create a diagram that represented all important Key components and their interfaces across sub-systems and modules. The task of maintaining the Interface diagram was a parallel activity conducted in the design process. The task has taken up resources, but because the tool was used as a boundary object in meetings and representation of different design teams' work, the interviews with team members revealed that they did not see the tool as taking up more of their time. The process of establishing and maintaining an Interface diagram for a product system can be described as an iterative process consisting of the following activities:

- Interviews with sub-system responsible domain experts (the technical stakeholders) in a product.
- Drawing the design in a block diagram (MS Visio).
- Reviews with domain experts in order to approve the present status.

The purpose of using the Interface diagram in the company was to lay out the architecture of the product's sub-systems in an optimum set of modules. The tool enabled designers, module owners and other stakeholders to evaluate consequences of redesign and upgrade activities, for example, evaluation of impact when implementing a new design of a module or a new sub-system. Furthermore, the Interface diagram was utilised for decomposing the product system into modules and enabling development of the modules in parallel. Compared to the previous project in the company, the project that used the Interface diagram had shortened the lead time through the early phase of the development process. An estimation made by the manager of systems integration was a reduction of 25% of the time from G1 (Business Plan) to G3 (Product Design). The reduction of lead time in development was mainly achieved because of an improvement in concurrency of development activities. The R&D resources saved have not been quantified in a structured way due to two factors. First, the project was not completed yet. Second, it was difficult to compare this project with others because the project also included development of radical new technologies. Instead, the outcome of applying the tool has been qualified by asking participant in the project. More than 40 project members have given their input in unstructured interviews. The original objectives for using the tool and the related outcome of application (based on the project member's feedback) are described below:

- *Acts as a vehicle for communication between stakeholders*. The Interface diagram presents for the first time the architecture of the product system to all stakeholders in a unified way.
- *Provides guidance for the decisions of detailed design*. The Interface diagram is used in systems integration meetings to form a common understanding of dependencies between components from different sub-systems.

- *Supports a modularisation process*. The Interface diagram visualises 'isolated' module candidates or if they are part of a more complex and interrelated set of module candidates.
- *Keeps track of latest design changes*. A printed version of the latest Interface diagram is on the wall in all the development departments. This makes the present status of the project visible for everybody.
- *Ensures a basis for developing simple interfaces*. Several times solutions for interfaces are changed because the Interface diagram highlights many-to-many relations between modules.
- *Deals with transformation of the product over time*. The Interface diagram is under revision control, and it is thus easy to get an overview of prior concepts of modules and interfaces.
- *Point out responsibility of interfaces and designs*. The clear distributed responsibility enables the project managers to run development activities more in parallel.

Interface diagram as a basis for PLM system support

Describing the detailed functionality of the used PLM system is outside the scope of this article. This section gives a short introduction to the approach of using the Interface diagram as a basis for structuring product data in a commercial PLM system and using the PLM system actively in the design process. One of the assumptions behind this research is that companies can have beneficial use of PLM systems for handling the underlying logic of modular architectures, interfaces and design rules. The strength of the Interface diagram is its visual capabilities for representing complex relations in a product system in a straightforward way. There are, however, shortcomings by using the Interface diagram related to representing and managing underlying, detailed information as interface descriptions, three-dimensional (3D) models of components and assemblies, design specifications, configuration of products and so on. It has therefore been an objective to align the modelled structures from the Interface diagram, with product definitions in the company's PLM system. PLM can be seen as an integrated, information-driven strategy of managing the whole life cycle of a product starting from generating an idea, concept description, business analyses, product design and solution architecture, technical implementation and product testing, to the entrance to the market, service, maintenance and product improvement (Sääksvuori and Immonen, 2008). PLM systems are computational tools for enabling the management of product information. Instead of only being an environment with a focus mainly on the administration of files generated from specific tools like computer-aided design (CAD), the PLM system applied in this research contains a master model of the product system represented by the structures from the Interface diagram (sub-systems, modules and interfaces). The structures in the PLM system are named *Upper structures*, to underline that they constitute the master product model. The object class Key component is present in all three structures, and the object can be associated with both geometric data (CAD models) and non-geometric product characteristics (described in documents or by attributes). In the PLM system, associated information is named *Lower structures*. The reason for working with Upper and Lower structures is to use the Interface diagram in Interplay with the PLM system. Upper structures are defined in the Interface diagram (systems, module and interface), and Lower structures consist of all other data handled in the PLM system linked to objects in the Upper structures. The object class Key component is present in all three structures (systems, module and interface), and the objects can be associated with both geometric data (CAD models) and non-geometric product data characteristics and properties (requirements, specifications, calculations and other view models) (Figure 9).

A platform was set up in the company's PLM system in order to prepare the alignment with the objects defined in the Interface diagram. In this research, PTC's Windchill 9.1 was used for the approach. In collaboration with the software vendor, functionalities for loading product structures from the Interface diagram were included in the commercial version. By means of a small written software program, the modelled structures from the Visio diagram could be retrieved and an output was generated in XML format that could be imported to the PLM system. By accessing the PLM system via a web browser, it was possible for project members to retrieve information in multiple contexts (sub-system, module and interface). An example from the Bobcat is used to illustrate this in Figure 10. CAD files (Lower structure) are linked to Key components (Upper structure). The Key components are present in a sub-system, a module and an interface structure. By selecting a structure in the PLM system, the data linked from lower structures are represented according to the chosen viewpoint. By selecting to monitor the product system in a sub-system context, the design team responsible for the hydraulic system, for example, can monitor the sub-system in its totality. Instead, if selecting the module structure, engineers are able to see documentation of how the product variants will physically be assembled. A sub-system solves a functional task, while modules are created because of one or more drivers of modularity (carry over, variety, testing, transport, etc.). The drivers of modularity are adopted from the work of Erixon (1998).

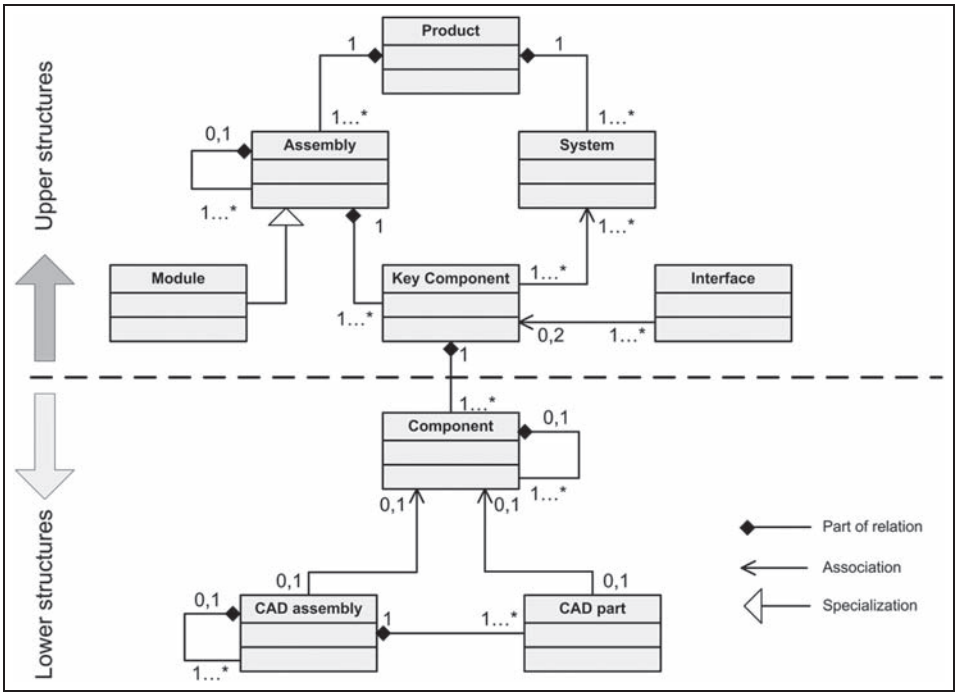


Figure 9. Entity relation model of the structures loaded into the PLM system.
PLM: Product Lifecycle Management; CAD: computer-aided design.

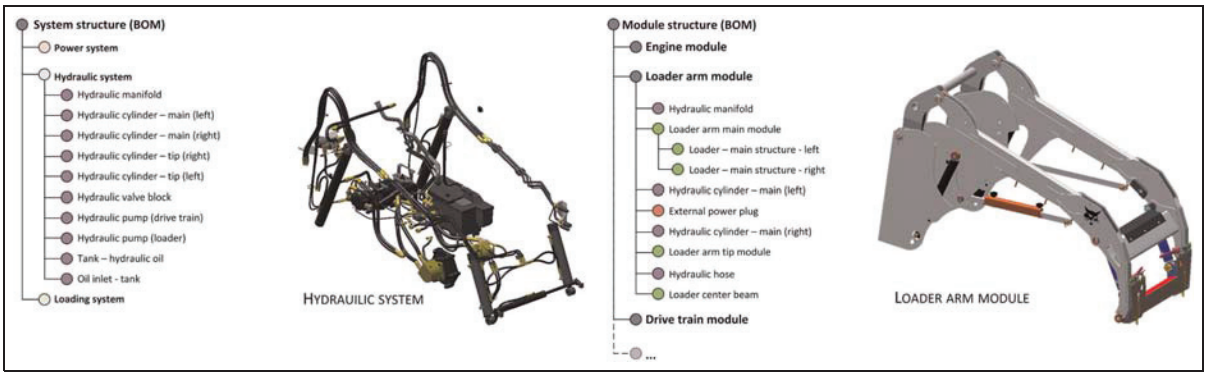


Figure 10. Bobcat examples of system and module structures.
BOM: *Bill-of-Material*.

It is in the PLM system where additional criteria as requirements to properties and characteristics can be linked to entities as sub-systems, modules, components, interfaces and interactions. When monitoring a component in the PLM system, it is possible to see where the component is used (sub-systems, modules and interfaces/interactions) and to see attributes (properties and characteristics) for it and its parents. Furthermore, additional view models for sub-systems, modules and components can be linked. Additional view models can be thermodynamic representations, strength models, fluid dynamic models, electrical diagrams and so on. Relational performance properties such as weight and cost can in the PLM system be calculated in terms of adding up assigned properties of elements. By choosing module or sub-system views in the PLM system, the designer has the possibility of justifying in-progress design properties, for example, weight and cost. Different variants of a product family can be configured in the PLM system. Similar to the approach supported by the Generic BOM, the end products defined in the PLM system have a number of features from

which a number of options are available to be chosen. A specific module BOM can be generated for each product variant.

Discussion

This article focuses on modelling a product system, and with that supporting the engineering process of handling structural complexity by developing modularity in product systems. It proposes an approach that prescribes or aims to decompose a product system into modules and to develop modules in parallel in order to achieve 'concurrency and simultaneity' (Prasad, 1999). The architecture for a product system can be a complex concept to develop. All products, sub-assemblies and parts in a product system must be orchestrated to perform together harmoniously and to fulfil one or more purposes in an application context. The very scale of a development project can make coordination challenging. One team's solution may create obstacles for another team, and coordinating and balancing solutions may require a great deal of iterative discussion. The engineering process must necessarily be supported by a shared product definition, for example, an architectural model like the Interface diagram. The optimum of a visual model like the Interface diagram is that it provides the necessary details and, yet, still maintains the overview in order to support and improve decision-making. Choosing the right level of decomposing sub-systems and interfaces in the Interface diagram is consequently of great importance to the productivity of the tool. There is no clear answer to what the appropriate decomposition level in the Interface diagram is, given that it is contextual to the product system it represents, depending on the number of components and technical complexity. A rule of thumb is, however, that the number of levels to which the decomposition is performed will depend on the size of the initial functional building block, the level of definition required and the lowest level of decomposition that is meaningful from an integration perspective. Meaningful means that decomposition should support integration between different sub-systems and modules, and to a lesser degree, in the design of internal interfaces in specific sub-systems and modules. Products with a large number of interfaces cause many difficulties for engineers if they are managed manually. A visual product model can therefore not stand alone to support interface management. The interfaces of a complex product can be better managed through the use of a computer-integrated environment like a Product Data Management/Product Lifecycle Management (PDM/PLM) system (Rahmani and Thomson, 2011).

The object for the Interface diagram model is the design and thereby the object of designing how the product functions and how it is built. In this process, there has to be multiple trade-offs, not only between performance and commonality. The Interface diagram shows its usefulness in combination with a PLM system. The PLM system enriches the modelled structures of the product with properties, additional characteristics and additional view models. This enables the designer and decision-makers to perform a balancing of multiple objectives such as cost, time and performance during the design process. Even aesthetic considerations are possible to perform when geometry models are linked in the PLM system to the elements defined in the Interface diagram (products, modules and components). The usefulness of using the Interface diagram in combination with a PLM system shall be seen in the light of handling the complexity of aligning cross technical and global development. The Interface diagram and its way of representing the product system give the company a common product definition, without reducing the support from inter-domain tools in the development process. The drawback of the approach is mainly linked to the complexity of the approach itself. Procedures for creating and maintaining the Interface diagram during the design process have to be in place. Transferring the modelled structures to the PLM system and keeping structures aligned in both the Interface diagram and PLM system are not without effort. Managing user privileges for defining and linking attributes, CAD, electronic computer-aided design (ECAD) and other view models to elements in the PLM system can also be complex for large organisations. The usefulness has, however, been reported in the case project because all this information often is scattered around a lot of system tools, view models and documents, making it very difficult for designers to get access and to see relations and the arguments for why a component has been designed like it has.

Conclusion and future work

This article presents a means of support for the development of modular product systems. It does so by introducing a product model that can be used as a design tool for decomposing a product system into manageable parts, which again can support the creation of modules. The motivation of using the Interface diagram is to support the synthesis activities when developing complex product systems by handling decomposition, characterisation and composition. The support constitutes a visual design model in interplay with a PLM system. This article is based on a real-life industrial implementation and is reported by means of action research. As the name suggests, action research is an approach to research that aims both at taking

action and creating knowledge or theory about that action. Our proposals shall be meaningful to practitioners, they shall be easy to integrate into their practice and support their reasoning; this is where the validity shall be found. The contribution, seen from an academic perspective, is the further development of modelling methods of the Domain Theory (Andreasen, 1980). One part of the contribution is the formalism, used for representing a modular product system. An additional part of the contribution is the approach by which the visual design tool supports the creation of modules. The model presented here does not represent a complete framework to describe the development of modular product systems, but serves as a contribution to the framework of authors in the area of practical application of supporting tools. The strength of this approach is that it can be used in companies developing products belonging to a diversity of types and industry. Even if strict definitions differ, the fundamental principles of modular design are common: break product systems into modules, ensure modules can interchange with each other, provide well-defined interfaces and resolve the inherent trade-off between commonality and distinctiveness that exists within a product family. The reported outcome of applying the tool in the project is based on questioning the participants. The use of the tool supported an improved understanding of the whole product system. No claim can, however, be made that the approach is an 'one-size-fits-all' solution, as it still needs to be tested on more projects and other segments of the manufacturing industry. The team responsible for maintaining the Interface diagram in the company made a quantitative estimation of the reduction of lead time in early development of 25%. The reduction of lead time in development was mainly achieved because of an improvement in concurrency of development activities. The R&D resources saved have not been quantified in a structured way due to two factors. First, the project was not completed yet. Second, it was difficult to compare this project with others because the project also included development of radical new technologies. The product structures established in the product model can be loaded into a PLM system. This again facilitates the management of product information at sub-system, module and interface levels. The product definitions in the PLM system in combination with the visual overview in the Interface diagram were considered as a great improvement compared to the prior Excel-based approach. The Interface diagram offers strong support when assessing issues on structural modularity, allocation of functions and interface creation. It provides less useful support when evaluating failure modes, assembly variations, material substitution, feature assessment and so on. Other metrics and measures in combination with simulations and analysis are consequently needed in order to assess, verify, balance and improve the realisation of a complex product system. One of the weaknesses of our approach is that it is difficult to handle product variety by the Interface diagram formalism. The support for handling product variety is based on the PLM system's ability to support configuration. The objective of using the Interface diagram is that it should provide an overview for all designers of a product system. However, the complexity of handling variants of a product system is not easily modelled by the formalism without compromising its visual usefulness. The Interface diagram, seen alone, is mainly supporting the mapping of different conceptual proposals for realising means in a product system consisting of one lead design – both for realising functional performance in the systems view and for supporting conceptual encapsulation of components in modules (standard designs). Part of the future work would therefore be to better address the visual handling of product variety in the Interface diagram. Detailing and elaboration on the ability to support the creation and evaluation of modules should also have a strong focus.

Declaration of conflicting interest

The authors declare that there is no conflict of interest.

Funding

This research received no specific grant from any funding agency in the public, commercial or not-for-profit sectors.

References

Andreasen MM (1980) *Syntesemetoder på Systemgrundlag – Bidrag Til En Konstruktionsteori*. PhD Thesis, Department of Machine Design, Lund Institute of Technology, Lund (in Danish).

Bruun HPL and Mortensen NH (2012a) Modelling and using product architectures in mechatronic product development. In: *Norddesign 2012*, Aalborg, 22–24 August.

Bruun HPL and Mortensen NH (2012b) Visual product architecture modelling for structuring data in a PLM system. In: *PLM12*, Montreal, QC, Canada, 9–11 July.

Erixon G (1998) *MFD – modular function deployment: a systematic method and procedure for company supportive product modularisation*. PhD Thesis, The Royal Institute of Technology, Stockholm.

Gedell S, Michaelis MT and Johannesson H (2011) Integrated model for co-development of products and production systems-a systems theory approach. *Concurrent Engineering: Research Applications* 19(2): 139–156.

Hansen CL, Hvam L and Mortensen NH (2011) Proactive modeling of product and production architectures. In: *Proceedings of 18th ICED 11*, Copenhagen, 15–18 August.

Harlou U (2006) *Developing product families based on architectures - contribution to a theory of product families*. PhD

Thesis, DTU Mechanical Engineering, Technical University of Denmark, Lyngby, 173 pp.
Haskins C (2006) *Systems engineering handbook, version 3*. INCOSE-TP-2003-002-03, June. San Diego, CA: INCOSE.
Hölttä-Otto K and de Weck O (2007) Degree of modularity in engineering systems and products with technical and business constraints. *Concurrent Engineering: Research and Applications* 15(2): 113–125.
Jiao J and Tseng MM (2000) Fundamentals of product family architecture. *Integrated Manufacturing Systems* 11(7): 469–483.
Kvist M (2009) *Product family assessment*. PhD Thesis, DTU Mechanical Engineering, Technical University of Denmark, Lyngby.
Mortensen NH, Gamillscheg B and Bruun HPL (2012) *Radikal Forenkling via Design*. Lyngby: DTU Mechanical Engineering, Technical University of Denmark.
Otto K and Wood K (2001) *Techniques in Reverse Engineering, Systematic Design and New Product Development*. New York: Prentice Hall.
Pahl G, Beitz W and Wallace K (1996) *Engineering Design: A Systematic Approach*. London: Springer, 544 pp.
Pedersen R (2009) *Product platform modeling*. PhD Thesis, DTU Mechanical Engineering, Technical University of Denmark, Lyngby.
Pimmler TU and Eppinger SD (1994) *Integration analysis of product decompositions*. USA: Alfred P. Sloan School of Management, Massachusetts Institute of Technology, pp.343–351.
Prasad B (1997) *Concurrent Engineering Fundamentals: Integrated Product Development*. Upper Saddle River, NJ: Prentice Hall PTR.
Prasad B (1999) Enabling principles of concurrency and simultaneity in concurrent engineering. *AI EDAM: Artificial Intelligence for Engineering Design Analysis and Manufacturing* 13(3): 185–204.
Prasad B, Wang F and Deng J (1997) Towards a computer-supported cooperative environment for concurrent engineering. *Concurrent Engineering: Research Applications* 5(3): 233–252.
Rahmani K and Thomson V (2011) Managing subsystem interfaces of complex products. *International Journal of Product Lifecycle Management* 5(1): 73–83.
Sääksvuori A and Immonen A (2008) *Product Lifecycle Management*. Berlin, Heidelberg, Germany: Springer Verlag.
Sanchez R and Mahoney JT (1997) Modularity, flexibility, and knowledge management in product and organization design. *IEEE Engineering Management Review* 25(4): 50–61.
Steward DV (1981) The design structure system: a method for managing the design of complex systems. *IEEE Transactions on Engineering Management* 28(3): 71–74.
Tjalve E ([1976] 1989) *Systematisk udformning af industriprodukter. Værktøjer for konstruktøren*. København: Akademisk Forlag.
Ulrich K (1995) The role of product architecture in the manufacturing firm. *Research Policy* 24(3): 419–440.
Ulrich KT and Eppinger SD (2004) *Product Design and Development*. Boston, MA: Irwin/McGraw-Hill, 366 pp.
Van Veen E and Wortmann J (1987) Generic bills of material in assemble-to-order manufacturing. *International Journal of Production Research* 25(11): 1645–1658.

Author biographies

Hans Peter Lomholt Bruun holds an MSc in Mechanical Engineering and is a PhD student at the section of Engineering Design and Product Development at the Technical University of Denmark. The project involves a close industrial cooperation with Danish companies. His main research focus is related to procedures and methods supporting development of Product Families based on Architectures and Platforms. Furthermore, his focus is on using PLM systems to support the development of product systems as, for example, product families.

Niels Henrik Mortensen holds a PhD and an MSc in Mechanical Engineering and is employed as a Professor at the Technical University of Denmark. He is the head of section at DTU Mechanical Engineering at the Technical University of Denmark. His main research focus is procedures and methods supporting development of Product Families based on Architectures and Platforms. Currently, there are nine researchers within the field of architecture-based product development.

Ulf Harlou holds a PhD and an MSc in Mechanical Engineering from the Technical University of Denmark and is the CEO of Center for Product Customization (CPC). His main work and research focus is procedures and methods supporting development of Product Families based on Architectures and Platforms. Currently, he has been involved in more than 100 industrial projects in global companies.

Paper B

Hansen, C.L., Bruun, H.P.L., Mortensen, N.H., Hvam, L.,
Identification of a scalable architecture for customization of complex parts, Journal of Concurrent Engineering, Research and Applications, In 1st review 2014.

IDENTIFICATION OF A SCALABLE ARCHITECTURE FOR CUSTOMIZATION OF COMPLEX PARTS

Christian Lindschou Hansen[1*], **Hans Peter Lomholt Bruun**[1], **Niels Henrik Mortensen**[1] **& Lars Hvam**[2]

Department of Mechanical Engineering[1]
Product Architecture Group
The Section of Engineering Design & Product Development
Technical University of Denmark
Building 426
DK-2800 Kgs. Lyngby, Denmark
Email: chrlh@mek.dtu.dk, hplb@mek.dtu.dk, nhmo@mek.dtu.dk
Telephone to corresponding author: +45 4046 4233

Department of Management Engineering[2]
Center for Product Modeling
Technical University of Denmark
Building 424
DK-2800 Kgs. Lyngby, Denmark
Email: lhv@man.dtu.dk

** corresponding author*

Abstract

Many Original Equipment Manufacturers (OEMs) are experiencing an ever increasing pressure on their ability to develop solutions at a faster pace and at competitive prices. In particular, OEMs developing and manufacturing high-performance complex parts with highly integrated product structures are struggling with time-to-market and unpredictable quality levels. Orders are often fulfilled in an engineer-to-order workflow and very little commonality exists between solutions across the product program. These OEMs also experience significant challenges in applying modularization to their product programs as a means to overcome the challenge, due to the reason that highly integrated product structures of complex parts are not easily modularized using traditional methods for modularization. In such cases resulting compromises on performance and cost are most often difficult to unite in a competitive product. This paper presents a framework that enables such companies to overcome these challenges by identifying an architecture of the product and production setup. The architectures enable the companies to scale their solutions and production setup in a profitable way, and at the same time maintain a sufficient degree of commonality to significantly improve time-to-market, R&D resource utilization and the level of quality.

Significance: Many companies experience problems with applying modularization to highly integrated product structures of complex parts. This paper suggests a framework for identifying modularity even beyond the traditional physical product interfaces by an architecture that allow these companies to harvest the benefits of modularization without compromising functionality, performance, or cost.

Keywords: Product architecture, product platform, product customization, scalable architecture, product complexity

1 Introduction

Many Original Equipment Manufacturers (OEMs) experience an increasing pressure on their ability to develop solutions faster and at lower costs. It is not unusual that large companies expose their OEMs to cost cutting strategies forcing them to cut 5-10% of cost every year (Zoia, 2013; Bickerstaffe, 2012). This leaves OEMs with no choice but to innovate. Only through developing new solutions with improved performance can they postpone the pending price drops and remain profitable. As many businesses also become more and more project-oriented, large companies are minimizing their risks and become reluctant to co-finance R&D activities at their OEMs, while at the same time expecting their OEMs to supply new solutions at a faster pace. This is a tendency which has increased in the wake of the global financial crisis.

These changed circumstances put the OEMs under pressure, requiring them to increase their R&D efficiency to improve time-to-market by providing new competitive solutions faster; and at the same time cut costs by improving product quality and productivity in production. The globalization of most large companies has also forced OEMs to act in the global stage, making the need for local variants increase, while the fluctuation of demand between the different variants makes production planning more difficult as the production volume of each product variant decreases.

Requirements on OEMs	
Decrease	*Increase*
Time-to-market	R&D Efficiency
CAPEX (Capital Expenditure)	Production efficiency
Volume per product variant	Quality level
	No. of product variants

Table 1 – Conflicting requirements on OEMs

The requirements listed in Table 1 are conflicting. And they apply challenge an OEMs agility of supplying new solutions, whilst at the same time making it very costly to expand the current portfolio using traditional approaches. Developing new solutions in a traditional engineer-to-order workflow would severely compromise R&D efficiency while imposing negative consequences on production ramp-up times and product quality, as every solution is often new from an R&D and production point of view. At

the same time the OEM customers require that OEMs push the envelope of the products' performance to enable them to achieve higher efficiency and improve their own offerings.

Traditionally the notion has been that applying modularization can make it possible to develop a modular architecture for the product program (Meyer and Lehnerd, 1997; Ulrich, 1995). Modularization when appropriately applied could serve as a means to provide the variety needed from a customer point of view and at the same time reuse sub-solutions across different products to improve time-to-market, and maintain predictable product quality. However, many companies do not succeed in this, as modules are not easy to identify in products where key functionality is highly integrated and distributed across the product structure (Hölttä-Otto and de Weck, 2007). Examples of these types of complex parts include e.g. rotor blades, complex manifolds and engine parts, hulls etc. that in addition contain a high degree of engineering from several disciplines as fluid mechanical engineering, solid mechanical engineering, process engineering and chemical engineering. In these cases, the compromises on performance often become too significant following traditional modularization approaches (Guo and Gershenson, 2007; Gershenson et al., 2003). In the development of highly performance oriented critical components – no extra 3-5% of material/weight can be added to the products. Such over-engineering is most often not an option.

This paper is based on the assumption that the definition of an architecture can enable the OEMs to overcome the challenging situation described above in Table 1. So far, traditional modularization has played an important role in architecture-based product development (Ericsson and Erixon, 1999). But as traditional modularization takes its starting point in functional decoupling in order to isolate functionality in modules (Jiao and Tseng, 1999), there is a need for a new approach applicable to products where functionality is highly integrated and distributed across the product structure.

This paper presents a framework seeking to expand the recent body of knowledge within modularization and architecture-based development of product programs by including the situation of OEMs supplying engineer-to-order solutions with integrated functionality distributed across the product structure of

complex parts. Thus, the framework disclaims the current notion of modularization and architectures being a compromise to achieving competitive performance in highly integrated designs. The framework contains a set of coherent models and a stepwise approach.

This paper will continue by elaborating the motivation and requirements for the framework. This will be followed by a review of the state-of-the-art methodology and a presentation of the proposed framework. The framework has recently been applied in an industrial case study with promising results, which are also presented here. The paper is concluded by a discussion, reflection and a conclusion.

2 Challenges and barriers

In the experience of the authors no OEM management disagrees with facing challenges in meeting the requirements mentioned in Table 1. Many of these challenges are closely related to the current ways OEMs are developing unique solutions to every customer. The observations below are based on a collection of interviews with management of OEM companies:

Customers are dictating solutions: As many OEMs do not proactively show their customers which solutions they would prefer, the natural result is that OEMs end up providing solutions that are unique to every individual customer. In the pursuit for customer satisfaction many OEMs enter a sales dialogue with a mindset of accepting a level of customization close to 100% – even regarding sub-solutions that are not critical to the overall performance of the product.

No proactive go-to-market approach: Many OEMs satisfy themselves with their upstream position in the value chain as being a rather passive supplier in the market place, not recognizing the need for an active marketing effort to analyze market developments and predict which particular segments are growing. Therefore they tend to fall into a reactive role resulting in lack of responsiveness towards technological trends and changing requirements from customers.

Very limited development outside customer projects: Many OEMs do not start development before the customer has accepted to cooperate. This leaves customer projects being notoriously behind schedule and leaves little-to-none opportunity for including forward-looking development work in customer projects.

Customer projects are fulfilled with zero outlook: The natural consequences of the reactive market approach and customer dictated solutions are that customer projects often fail to look beyond their first delivery. The focus on the first delivery results in solutions that are not prepared for upgrades or predisposed for future variants.

Solutions are developed in individual work streams: In the attempt of becoming customer focused, OEMs organize development teams in individual work streams, with only little or no coordination between them. The result is often lack of solution overview and sub-solutions that vary from customer project to customer project, making it difficult to harvest benefits of common solutions across customer projects while prolonging time-to-market.

Complex interplay between specialized engineering disciplines: The high performance products in scope here require very specialized engineering design work from different disciplines. Even though every engineering discipline has a separate optimization task closely related to the unique customer requirements many dependencies exist between them, making it difficult for individual disciplines to predict and see through their influence to other disciplines.

R&D resources are tied up to individual customer projects: OEMs tend to tie a relatively large share of their R&D resources into specific customer projects, when customers are willing to co-finance development costs. However, “renting out” R&D resources for e.g. EUR70-80 per hour makes a very low return on investment compared to investing in R&D resources developing solutions that could be common for a range of customers. The issue becomes particularly crucial when customers exit the projects before time and the OEM is left behind with a customer specific solution and zero orders.

Lack of task definition: It is the experience of the authors that the challenges mentioned above is a natural consequence of the way OEMs orchestrate their engineering resources and define their tasks and priorities. There are often very few or no cross-functional initiatives that aim to overcome these challenges as they span all the way from initial customer dialogue to the engineering execution and project delivery. Moreover customer projects are often kept isolated and 100% focused on customer dictated requirements making it difficult to prioritize efforts in order to change the condition mentioned above.

These challenges end up becoming barriers for OEMs to fulfill the requirements listed in Table 1.

3 Requirements for the framework

To overcome the challenges and barriers presented in the previous section, there is a need for OEMs to address their development of customer specific solutions in a different manner. As the problem reaches all the way from early sales dialogue to the delivery of engineering solutions, it is not sufficient to make small and local changes in order to turn the situation around – there is a need for a more profound change. A change that needs the support of a framework, in order to

- Identify an architecture for the product and production setup that allows for profitable customization of complex customer specific solutions with highly integrated product structures by improving the utilization of R&D resources and improving time-to-market.

In order to do so, the framework should enable OEMs to

- Scope an architecture from a market point of view. This includes focusing upon which segments and applications with which performance steps to cover and which not to cover. This again determines the market envelope i.e. the collective of the preferred offered product variants and their desired performance steps.
- Identify the defining design characteristics and properties based on primary market parameters. This includes the scaling principles for design characteristics and performance properties.

- Identify an architecture that utilizes R&D resources to provide value instead of solving similar requirements for every customer project as repetitive work. This includes decoupling of work tasks to enable different disciplines to work in parallel.
- Identify an architecture that allows for scaling of sub-solutions to allow for reuse between customer projects.
- Identify an architecture with product scaling principles that are coordinated with the production scaling principles with respect to geometry and volume.
- Identify an architecture that can serve as foundation for deriving several future product variants beyond the first delivery

With these requirements literature has been reviewed for contributions that address the challenges described and fulfill the requirements listed here.

4 State-of-the-art

This section covers significant contributions in literature to: The modeling of architectures, the support for developing modular product families, and methods for improving the development process in terms of reducing lead time.

Architectures based on design process theory: According to (Andreasen and Hein, 1987) the development process can be described in terms of single models on four levels: product planning, product development, product synthesis and problem solving. The design process theory is based on descriptive and prescriptive models from (Pahl et al., 1996; Hubka and Eder, 1988). Product planning is related to activities where decisions regarding introduction of new products and phasing out existing products are made. Concerning application of architectures and platforms, it will have impact on all levels above, i.e. product planning, product development, product synthesis and problem solving. In order to succeed with a platform it has to be fitted to the nature of product development, meaning that solutions are determined

gradually, and platforms have to include certain flexibility in order to match different design projects and product variety.

The concept of Concurrent Engineering (CE) is a parallel approach, replacing the linear process of serial engineering. CE is intended to encourage the product developers, from the start, to consider the total job (Prasad, 1999; Prasad, 1996). The concept of CE is very much linked to the concepts of architecture based development. In a modular architecture, there is a division of labor between architects who first split a product into modules, and those who work within the parameters of a specific module. The latter group needs to know only about the specific module and the design rules which ensure that the module can be integrated into the larger system. Modules can then be developed in parallel, which again lowers lead time in development. A prerequisite is that architects possess the requisite knowledge of parameter and task interdependencies of the whole product.

The concept of Architecture for Product Family (APF) is introduced as a conceptual structure, proposing logics for the synthesis of product families (Du et al., 2001; Jiao and Tseng, 1999) . The Generic Product Structure (GPS) is then proposed as the platform for tailoring products to individual customer needs. (Ko and Kuo, 2010) presents another systematic method for concurrent development of product families, by combining QFD-based methods with quantified DSM-techniques and morphology analysis to visualize concepts.

Product Family Master Plan (PFMP): The Product Family Master Plan (Mortensen et al., 2010; Harlou, 2006), describes a product assortment from three points of view: Customer, engineering and part view, equivalent to the partitioning in market, product and production domain of the Integrated Product Development framework (Andreasen and Hein, 1987). Each of the views is causally linked meaning that certain types of traceability can be described. The relation between customer view and engineering view describes how certain customer features are realized by means of certain functional units. The relation between engineering view and part view explains how functionality is realized by means of physical parts and sub-assemblies. Reading the PFMP from the part view to the engineering view explains how a certain

part contributes to delivering functionality to the products. From the engineering view to customer view the relation describes how functional units deliver customer features and, value to the customer. Another important aspect of the PFMP is that it enables a professional dialogue between three very important stakeholders, namely sales, engineering and production. There must also exist a professional media for communication and decisions concerning the exact scope and content of possible platforms.

Modular Function Deployment (MFD): The Modular Function Deployment (Ericsson and Erixon, 1999) builds on the methodology of the Quality Function Deployment (QFD) and on the formulation of eight so-called module drivers. The purpose of MFD is to enable cross functional teams (including mainly marketing, development and production) to create a mapping from the physical structure of the products within a family to the functional structure of those products and to ensure that the functional structure corresponds to the demands of the customers. Modular Function Deployment method consists of five consecutive steps. Customer requirements are mapped to functional criteria and subsystem design characteristics and subsequently form a physical design in which a modular architecture supports a carefully selected set of modularization incentives called module drivers.

Design Structure Matrix (DSM): This approach takes a starting point in the decomposition of a product into components/systems and an identification of interfaces/relations among these (Hölttä-Otto and de Weck, 2007; Pimmler and Eppinger, 1994). By the use of algorithms, it is possible to encapsulate components into modules or chunks that are closely related to each other from an interaction point of view (Steward, 1981). This process is referred to as clustering. The outcome of a DSM is a proposal for a future modular product architecture.

The function-based design methods (Function structures) are characterized by establishing either a function model (Otto and Wood, 2001; Pahl et al., 1996) or the schematics of the product (Ulrich and Eppinger, 2000). Both approaches have a visual representation as an outcome. The function structure describes the flow of material, data, and energy through sub-functions of the product using a set of rules (e.g. the rules that are referred to as the functional basis which basically is a common language to describe

functional elements). The schematic of the product is somewhat similar to the function model. But where the function model describes the product using functional elements, the schematics can describe both functional and physical elements — whichever is the most meaningful for the purpose of the representation. The functional structure forms the basis for several different approaches to design or re-design the products.

The German school of Variant Management provides a number of methods and techniques to optimize the design of variance in product families (Krause et al., 2013). The methods and techniques form an integrated approach, which aims to reduce internal variety of product programs. The overall idea of the approach is that non-value adding variety of the product program is both constituted by elements belonging to product variety and process variety. In order to reduce internal non-value adding variety of the product program, one has to address technical-functional and product-strategic module drivers along the product life cycle phases, and to redesign components to enable a modular (variety optimized) product structure.

Configurable Component framework: The configurable component concept (CC) is a means of representing systems and their subsystems using a generic building block, the configurable component (Claesson, 2006). The original purpose of the concept is to handle data, information and knowledge sharing, as well as managing the conflict between commonality and reuse, while having the ability to represent variant-rich and complex products and, more generally, entire product platforms. The CC concept declares a bandwidth within which platform elements, including interfaces may vary. Thus, the interfaces are co-configured to fit each other, which allows for keeping design flexibility intact throughout the development process. The CC framework has been implemented and tested in the automotive and aerospace industry. The framework has been enlarged to cover an approach for integrating modeling of products and production systems (Gedell and Johannesson, 2013; Levandowski et al., 2013).

Impact of product configuration in engineering oriented companies: Studies have shown that engineering oriented companies (companies in which each customer order requires some engineering work) can gain significant reduction of lead time in quotation and production by implementing product configurators (Haug et al., 2011). Product modularization and configurations are used to structure and model the product assortment in order to configure a customer tailored product unambiguously. However, the creation and use of configurators is often a risky and highly time-consuming project. Thus, although for example for a 90 percent reduction of lead time and man-hours achieved, this may still be an unprofitable project if the costs of achieving this are too high.

4.1 Gap

The contributions definitely all play an important role in identifying architectures. However, very few contributions have dealt with the definition of architecture initiatives for highly integrated products or complex parts. For these product types it is often not possible to consider integration or balancing of different modules, because modules (in a classical understanding) cannot be decoupled from the rest of the integrated product structure. It is simply not possible to identify such. Integrated products or complex parts can be characterized as functional feature-based products in which the customer's perceived value in the products is based on properties such as peak performance and efficiency in operation. To optimize such functionality of a product family, it is necessary to consider compromises between the product variants (total systems) instead of compromises between modules (sub-systems), because the performance of the technical system is dependent on the balancing of design characteristics between product variants. The current frameworks, methods and models proposing to support architecture initiatives, do not enable companies to overcome these challenges. The gap in prior contributions is centered on the task of identifying flexible and scalable architectures, for highly integrated products, for the product and production setup. Yet, the theoretical basis mentioned in section 4 provides a thorough basis for deriving a framework integrating the PFMP-based methodology of concurrent design of market, product and

production aspects merged with a function-oriented modeling of performance properties to identify coherent feature, performance and production scalability.

5 Research methodology

The framework presented in Figure 1 is the result of experience derived from several previous case studies conducted with OEMs producing high performance mechanical products for a relatively small amount of large customers. These studies have matured the framework till its current state and subsequently it has been tested in yet another case study with very promising results.

According to (Joergensen, 1992), research is both problem and theory based. The problems in the industrial practice is described in section *Challenges and barriers* and which is based on several interviews with managers and decision makers in European engineering companies experiencing these problems. Therefore, many descriptive research activities lie ahead of this framework presentation, which can be characterized as being prescriptive.

5.1 Type of inquiry

Different types of inquiries were used while engaging in the practical setting. During the analysis phase of the study, the inquiries were rather exploratory and diagnostically based, helping the researchers to understand the situation and assess the applicability of the framework. Moving on to the synthesis phase, the inquiries changed to being more of confronting in character and directly prescriptive. The last type challenges the company to see their products from a new perspective, and was absolutely necessary in order to make them adopt the framework and ensure a successful intervention.

5.2 Visualization as working method

As the research aims to bridge information from sales and marketing with engineering and production development, there is a need to create a *boundary object* enabling the different competences to interact, exchange ideas, and understand each other's work challenges (Latour, 1986). Thus visualization has been used to create such a boundary object to facilitate collective alignment among sales/marketing,

engineering and production professionals. From the early stages of the project a concept architecture has been illustrated on A0 sized posters, allowing professionals with different backgrounds to gather around a large poster and make review meetings efficient and by taking advantage of the optical consistency such a visualization represents. This approach enables participants to lay aside their daily working habits and see the challenges in the project as being of the 'same type'.

6 Framework

The illustration presented in Figure 1 represents the framework for developing a scalable architecture including a step-wise approach. It is not the intention to present a complete framework, but instead emphasis has been put on the elements of crucial importance when an OEM wishes to:

- Move away from a dedicated engineer-to-order workflow where unique solutions feature in every customer project
- Explore their design envelope to investigate the potential for generating solutions with lower lead time
- Develop an architecture to enable faster development of new and competitive solutions
- Improve the efficiency and effectiveness of R&D resources
- Enhance the preparation level towards generating future derivative variants

The framework follows the classic partitioning in a market, product and production domain known from (Andreasen and Hein, 1987). The 9 steps represent sequenced excerpts from the framework that was discussed, reviewed and documented on large A0 format posters to keep a coherent overview of the architecture's status.

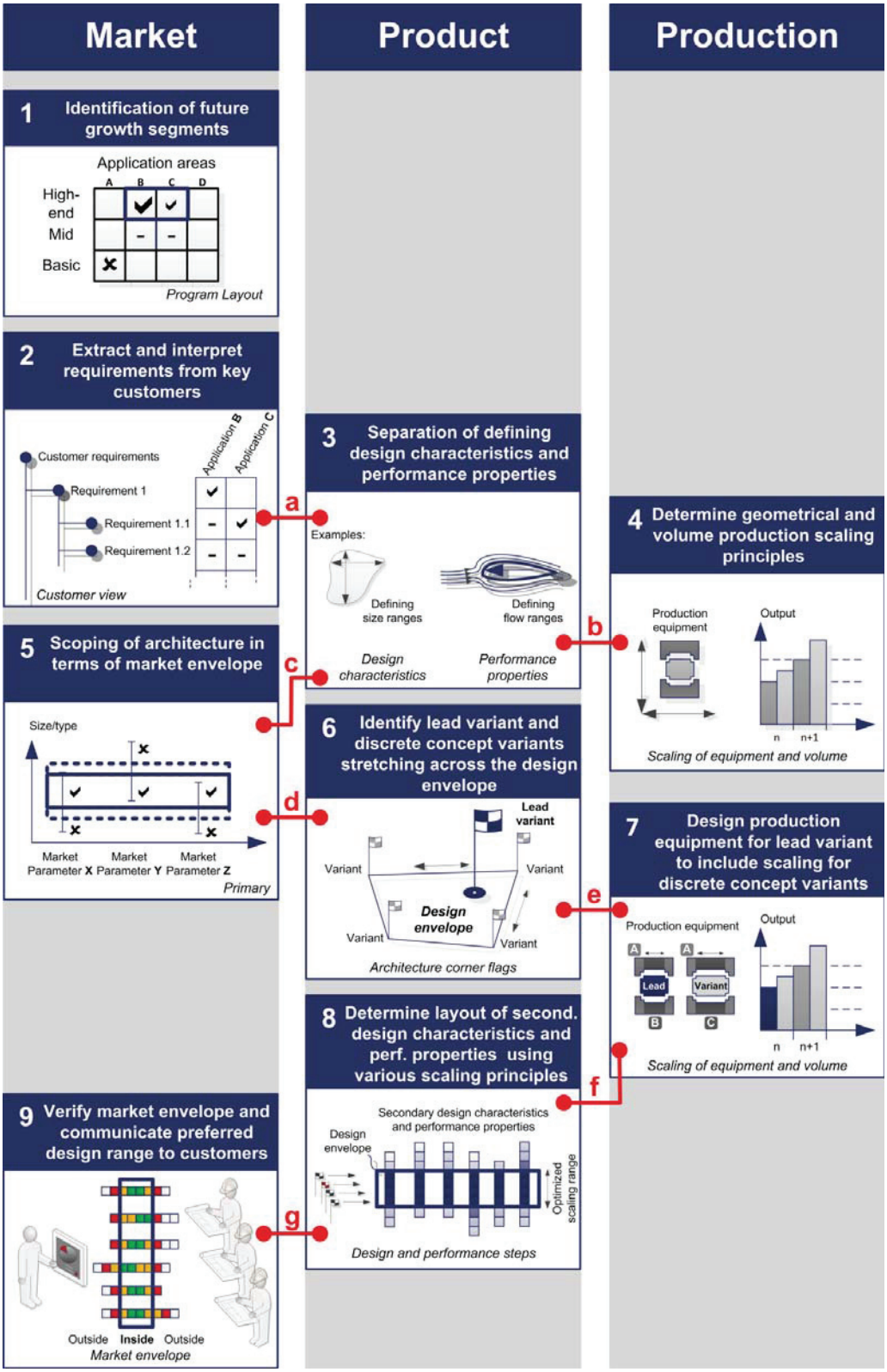


Figure 1 – Framework including approach

1. **Identification of future growth segments**

Developing an architecture which is prepared for the future, requires a much more systematic approach to analyzing and interpreting the growth of the most favorable market segments than traditional OEMs are used to. It is not necessarily good to be 100% customer driven in the sense that customers might not include the OEM in their 3-5 year product roadmaps thus leaving the OEM behind with shorter time to react. Therefore it is necessary to use for example the Program Layout to outline the growth in all relevant business areas across different tiers from high-end to basic (Hansen et al., 2012). This can form a basis for identifying which market segments to focus on and isolate the application areas of relevance.

2. **Extract and interpret requirements from key customers**

From the analysis of the future growth segments, the next step is to select a few key customers within these segments and extract and interpret their requirements. The Customer View can serve to support the systematic mapping of these requirements based on for example interviews with senior representatives within technical sales (Harlou, 2006), and categorize these requirements dependent on which application they are applicable for.

3. **Separation of defining design characteristics and performance properties**

The next step is to separate those design characteristics and performance properties that answer to the primary market parameters. These are derived as parameters fulfilling the most important customer requirements – e.g. efficiency, load limits or other characteristics and properties. This is a very critical step, as it is necessary to initiate the development of the architecture by solely focusing on those fulfilling the few requirements that are capable of positioning the products in the market place.

Link a: The interpretation of the most important application specific requirements into the few design characteristics and performance properties that can fulfill these.

4. **Determine geometrical and volume production scaling principles**

Using the direct input of the defining design characteristics (e.g. primary geometries) the scaling principles for production can now be determined. As it is critical to achieve scaling towards future variants and upgrades, this must be determined in coordination with the geometrical possibilities and limitations outlined by the defining design characteristics.

Link b: The physical scaling of production equipment is coordinated directly with the defining design characteristics and scaling of volume is taken into account.

5. **Scoping of architecture in terms of market envelope**

As it is impossible to satisfy all primary market parameters in their full range, it is necessary to scope the architecture in terms of which will be boundaries of the market parameters seen from the customer's point of view. The fulfillment of these parameters will often be subject to trade-offs, as the fulfillment of one parameter can be increased by compromising the fulfillment of another.

Link c: The market parameters might be direct market oriented translations of certain design characteristics (e.g. size) and performance properties (e.g. thrust, power, efficiency).

6. **Identify lead variant and discrete concept variants stretching across the design envelope**

Identifying the lead product variant, meaning the first product variant of the new product family, is the next step. The identification of this would normally be the result of close dialogue with a lead customer. While specifying the lead variant it is of crucial importance to specify a number of additional concept variants. These concept variants are not going to be completely designed in detail just yet. They serve as important instantiations of the architecture in order to investigate scaling principles of design characteristics between the different variants, including the lead variant. This is in order to achieve appropriately balanced performance steps of the most critical properties.

The sum of these architecture variants spans the total design envelope of the architecture.

Link d: The architecture variants are closely linked to the market parameters in the way that the design envelope matches a certain market envelope, where individual market parameters are covered within a certain range.

7. **Design production equipment for lead variant to include scaling for discrete concept variants**

In terms of production the basic scaling principles are already determined (step 4), but taking the starting point in the lead variant the actual equipment can now be designed and specified. The discrete concept variants are taken directly into account deciding the geometrical scaling principles as it is now possible to prepare production equipment (moulds, fixtures etc.) for the exact design characteristics of these. However, no capital is invested covering the discrete concept variants – only preparation activities to maximize future reuse of production equipment for the lead variant

Link e: Production equipment for the lead variant is prepared for scalability towards the design envelope represented by the discrete concept variants.

8. **Determine layout of secondary design characteristics and performance properties using various scaling principles**

Until now, the design only contains the defining design characteristics and performance properties. The secondary ones are now to be balanced within the design envelope to make a good fit in balanced design and performance steps. These secondary design characteristics and performance properties describe the sub-solutions. They do not interfere with primary performance, as the task is to optimize the whole architecture and product family. There is no value in varying these sub-solutions unnecessarily across the architecture.

Link f: Certain sub-solutions might have close ties to the production equipment and desired future volume scalability. Therefore, it should be ensured sub-solutions are aligned with the critical choices of production technology and volume scaling principles.

9. **Verify market envelope and communicate preferred design range to customers**

To close the loop and harvest benefits of the preparation of the architecture, the next step is to verify the design and resulting market envelope to make sure that it fits with the needs of potential future customers. If this can be verified the next challenge is to communicate the preferred design range to customers. The preferred design range is the market translation of the architecture variants prepared in the lead design and discrete concept designs.

Link g: As the market envelope reflects an optimized scaling range of the designs ("inside" the envelope), it is of critical importance to communicate the preferred range of designs to customers, as there might not be a good product match if the customer falls "outside" the market envelope.

Nomenclature: Green means "inside", Yellow means "inside with compromises/adaptions", Red means that larger changes to the architecture has to be made in order to offer a competitive product.

The exact sequence of the 9 steps varies slightly between individual cases, but the sequence presented here was the one used during the case study.

7 Case

The framework was applied in an engineering oriented OEM company serving the global energy industry with performance critical components used for critical energy generation processes. The company has a global production footprint enabling the OEM to serve a number of customers having only regional production facility of their own and to help them expand their market reach to win larger orders on a global scale. The case company is anonymized as a result of competitive reasons and in order to be able to report more interesting details than a public case allows for.

7.1 Situation

In the wake of the global financial crisis the customers of the OEM company have experienced financial problems with financing their energy solutions. Their financial shortcomings put pressure on their OEMs, as they cannot co-finance R&D activities to the extent they have done earlier, and cannot commit

themselves to larger production volumes as the business is becoming increasingly project oriented. The OEM is experiencing severe price cuts for their high volume products forcing them to find new growth segments, where their engineering expertise can be valued in terms of a higher price level.

7.2 Framework

The first engagement with the company was an assessment project to evaluate the potential of architecture based product development in the company. As the products of the company cannot be modularized in the traditional sense by visible and physical structural decoupling, certain skepticism was expressed by many stakeholders in the company. However, the result of the assessment project was to apply the framework to a new promising development project, where a few potential large lead customers had shown interest.

As specified in ***Step 1***, the potential customers constituted a future growth segment for the OEM, making it an appropriate business area to apply the framework to.

The next step, ***Step 2***, was to extract and interpret the key requirements from the most likely future customers – including the potential lead customers. The Customer View was used to create an overview of which requirements were common to all customers and which vary between them, and the most important market parameters were isolated.

In ***Step 3***, taking the starting point in the Customer View, the defining design characteristics and performance parameters were separated out in order to create early concept designs that fit directly to the market parameters. Thus it was ensured that the primary requirements of efficiency, load limits and preparation for scalability was taken into account during these very early stages of designing. Fulfilling these was directly associated with the fluid dynamic nature of the products. This made the fluid dynamic trade-offs and design considerations primary for defining design characteristics and performance properties.

Figure 2 exemplifies defining design characteristics and their relations to performance properties for an engine manifold. The performance properties of flow, pressure, and strength are realized in a complex

interplay between the inlet/outlet size, the tube wall thickness, the diameter of the tube, and multiple bending radiuses of the tubes.

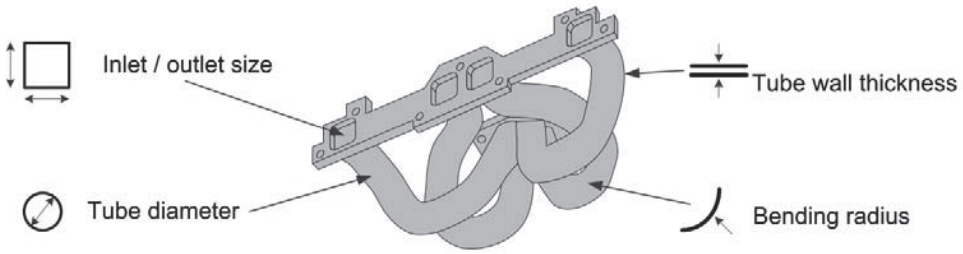


Figure 2 – Identified performance critical design characteristics for an engine manifold

The performance critical design characteristics were used in ***Step 4*** as input to determining the geometrical and performance scaling principles of the production equipment.

Figure 3 exemplifies the concept of scaling principles. The defining design characteristics of the engine manifold have been identified i.e. the defining structure and its attributes based upon desired performance properties. Some geometry is fixed for all variants in the design envelope (red areas) in order to support modularity in the production equipment, while others are flexible (green areas). The performance properties are related to flow, pressure, and strength. In order to meet the requirements to performance properties in the design envelope, the design characteristics have been determined in principles and ranges of scaling i.e. tube diameter, inlet/outlet size, bending radius in tubes, and the tube wall thickness. The relations between design characteristics and performance properties have been analytically verified, and even if no physical parts yet exist, the design is fully scalable within the defined ranges.

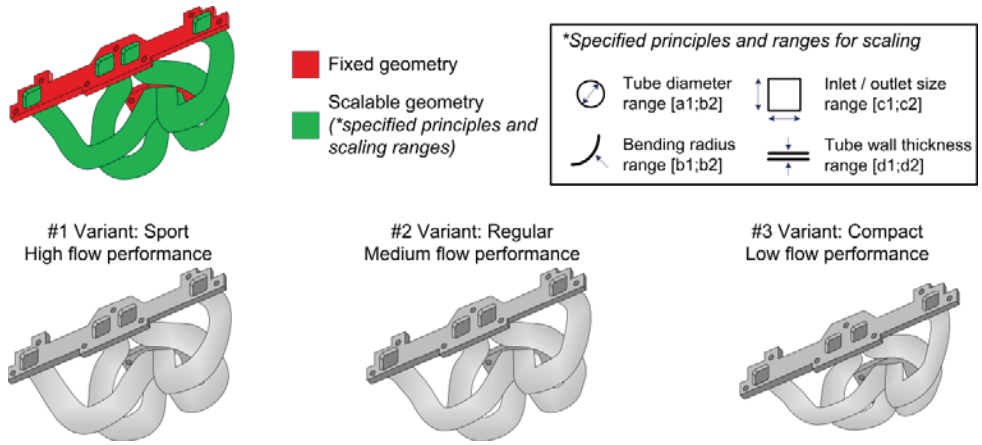


Figure 3 – Engine manifold used as example for illustrating scaling principles and scaling ranges

Modular thinking was incorporated early and applied to the production equipment to ensure future reuse of most the costly and lead time defining production equipment.

The next step, ***Step 5***, was to scope the architecture from a market point of view based on the work from Steps 1-4, as it was not possible to make a one-fits-all design. Clear decisions were made regarding which performance ranges to support, and which not to support. As the customers of the OEM use different technologies, several decisions were made to include support for certain ones while excluding the support of others (e.g. different support system technologies).

In ***Step 6***, the lead variant was identified together with three other discrete concept variants – one variant with lower specification and two with higher specifications. In total now four architecture variants. The lead variant was intended for a large customer. The discrete concept variants were aligned with the preferred scaling principles (from Step 4) and targeted to fulfill market parameters in the surrounding areas of the chosen ranges (from Step 5). Throughout the rest of the architecture design process, the concept variants were used as modifiable instantiations of the architecture to put structure to the design envelope. These concept variants do not limit the future design of variants to match these exact discrete specifications, but they prepare the architecture and the scaling principles applied to fulfill any requests for variants inside the design envelope of the architecture.

Step 7 was the inclusion of production equipment to include the actual lead variant while preparing for the future scaling of the discrete concept variants. Modularization was applied to ensure the preparation towards the required scalability of future variants by decoupling the variable equipment from the costly and lead time defining equipment.

In ***Step 8***, the sub-solutions were included now taking all the secondary design characteristics and performance properties into account. For instance, the steps between sub-solutions do not have to follow the same steps as the four architecture variants – the sharing of sub-solutions could be independent hereof, and in certain cases, only two variants of sub-solutions were chosen for the four variants in scope. An example of this was layers of strengthening material that were completely shared within the common parts of the architecture variants decoupling the variance needed to other more flexible areas of the design.

In ***Step 9***, the architecture including the design and market envelopes was communicated to customers. This approach represented a large shift of paradigm for the OEM company, enabling a much more qualified early dialogue, as the OEM could now enter specific design discussions with customers *before* having started an actual customer project with them. This has so far enabled the OEM to influence and impact design decisions of their customers – a side effect that only very few company representatives believed possible when the project started.

7.3 Results

An extensive validation effort was initiated after the framework was applied. All participants in the architecture design process were interviewed and their best estimate of their resource requirements (in hours) and lead time (in days) was collected. The investigation showed that

- An architecture variant (succeeding the lead variant) can be developed using:
    - R&D resources: From index 100 to 30
- Lead time from:

- First customer dialogue to finished design: From index 100 to 58
- Finished design to prototype delivery. From index 100 to 50

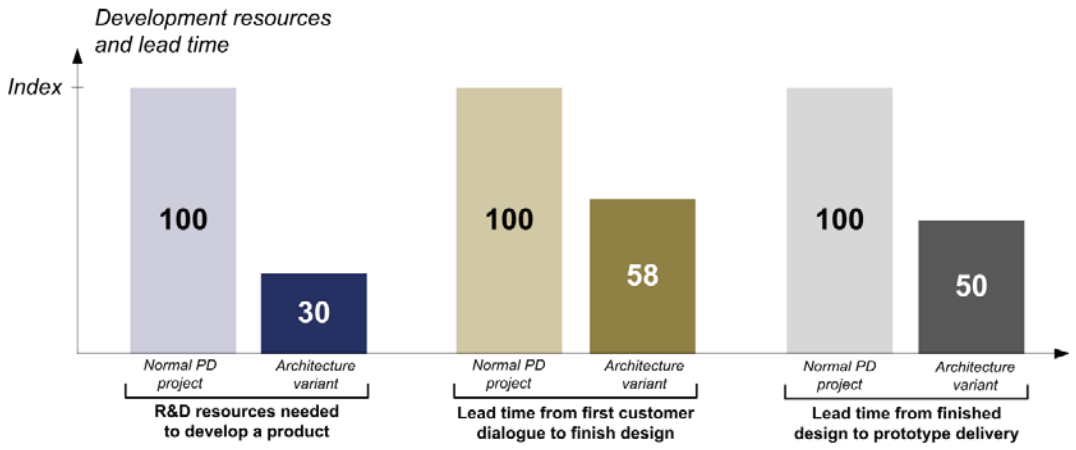


Figure 4 – Savings in design lead time and R&D resources for one variant

The index 100 is reflecting a traditional customer project where a unique and dedicated design is developed from scratch.

This estimate is conservative and includes all resources relevant – including project management itself, while taking critical path and normal project uncertainties into account.

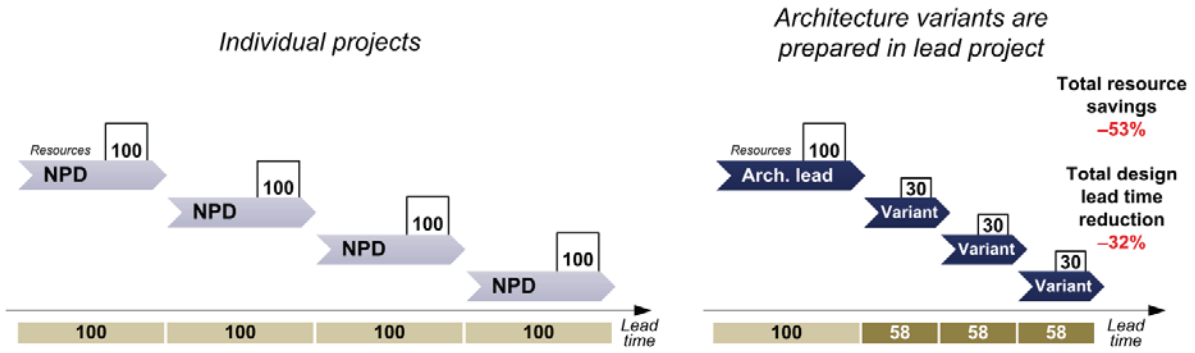


Figure 5 – Total savings in design lead time and resource consumption for four variants

Figure 5 illustrates the savings in design lead time (–32%) and R&D resource consumption (–53%) by using the new architecture approach instead of the company's previous new product development (NPD)

process. The architecture project covered four variants and the savings are calculated on this premise. The OEM's ability of supplying new solutions at a faster pace has increased, without increasing R&D resources in the same step. The R&D efficiency has actually improved by reducing the resources needed to develop four variants by as much as 53%. The savings compared to normal projects will moreover increase when more variants are developed inside the already specified design envelope.

The dramatically improved responsiveness towards customer requests inside the architecture has been a game changer for the OEM company. Customers are now even sharing their own roadmaps with the OEM company in order to ensure closer collaboration and make sure that their product requests falls inside the scope of the further development of the architecture. In other words, the benefits from requesting new products inside the envelope of the architecture are obvious enough to enable the early customer interaction as needed.

Today, the OEM company is working with marketing their architecture in order to harvest the full potential, and recent conclusions are that customers find their approach proactive and constructive. Also the organizational anchoring of the architecture thinking patterns, methods etc. utilized in this project is also a derived activity as are the implications to portfolio management, roadmapping, product and production technology etc.

8 Discussion and reflection

8.1 Case

Reducing time-to-market by almost half and saving 53% on R&D resources is a drastic improvement on current performance. However, limitations of the case results do exist.

The framework only covers the development of a lead variant and preparation for derived architecture variants. The actual execution of architecture variants was estimated through interviews with all relevant stakeholders, but the actual execution of these are in the making at the time of writing. However, the

projected realization time, on which basis 70% of the savings were derived, was accepted and found credible by the OEM management. This covers the internal validation of the results.

External validation is more difficult. The researchers have undertaken many architecture projects for OEMs developing mechanical solutions for large customers, which provide certain evidence that this OEM company is comparable to many other OEM companies. Their challenges are similar to the challenges experienced in many other OEM companies. The OEM is a global player, capable of attracting many skilled employees with an annual turnover in the range of EUR 0.5-1billion, and there are no indications that the leap of performance achieved in the case study, could be explained with lack of professionalism or a competence level below average.

The reliability of the results are therefore of course contingent upon many contextual factors of the company where it will be applied. Also the mere presence and attendance of the researchers in kick-off meeting, review meetings and evaluation meetings is impossible to isolate from the results – as of course, a certain competence within the field is necessary in order to create such positive results.

8.2 Theory evaluation

As reported in the case the framework proved useful and solved the challenge of identifying an architecture for the OEM case company. The theoretical gap mentioned in section 4.1 has therefore been challenged by the framework proposed, which is integrating the PFMP-based methodology of concurrent design of market, product and production aspects merged with the function-oriented focus on performance properties to identify coherent features, performance and production scalability. The framework's inclusion of behavioral aspects of architectures across market, product and production domains, namely what the architecture enables the company *to do* in terms of preparation and responsiveness towards future launches (instead of limiting the focus to what the architecture *is*) has also been an important parameter differentiating the framework from previous works, and proved useful in highlighting the relevance of the work in the industrial setting of the case study. This angle of attack

seemed a powerful response to the traditional skepticism which can be found by practitioners who doubt the industrial relevance of such a framework.

8.3 Further works

The framework applied here may be altered to fit the exact needs of other engineering companies. For example, the focus on the lead customer and lead design could be carried out earlier than described in this paper. Also, as high performance mechanical products might experience the need for very different optimization loops, the generic and general inclusion of these is difficult and therefore left out of this work. However, the transferability of the results presented here is generally assessed to be good, as long as industrial practitioners can mobilize the necessary driving force to ensure the architecture work is progressing and is aligned among the major stakeholders.

An improvement area to focus further works on is the creation of quantitative trade-off models to help support designers in critical decision making of where to apply dedicated design and where to allocate efforts for incorporating scalability into the architecture. The aim would be to minimize the costs of complexity while improving time-to-market and the responsiveness towards customer requests. Some early models were applied in this and earlier case studies, but it was out of scope to generalize these for this publication.

Another improvement could be a general assessment model to apply for another case company experiencing similar challenges. The model should support the assessment of the level of readiness to profit from the development of an architecture to serve the customers instead of always proposing dedicated designs. A result could be a maturity model that prescribed which areas to focus a pre-project on before developing the actual architecture in a lead customer project.

9 Conclusion

This paper has presented a framework including a step-wise approach to develop an architecture, particularly suited for OEMs developing mechanically and highly integrated performance products, where

traditional modularization is not enough to achieve reuse, scalability, reduce time-to-market while improving R&D resource utilization. The framework and its approach takes it starting point in bridging the few but defining design characteristics and performance parameters of the products with the market parameters that are critical to achieving competitiveness within a target segment. By systematically implementing scalability for main- and sub-solutions an architecture for the product and production setup is developed. This ensures an increased responsiveness towards customer requests and a case study shows promising and significant reductions in time-to-market (almost 50%) and savings in R&D resources used per customer project to develop derived product variants based on the architecture (70%).

The authors would like to thank the case company for sharing challenges, resources and competences with the research team.

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

10 References

Andreasen, M M, Hein, L. (Eds.), (1987). Integrated Product Development. Springer, Berlin

Bickerstaffe, S., (2012). Comment. Automotive Engineer. 37

Claesson, A., (2006). A configurable component framework supporting platform-based product development. Doktorsavh. Chalmers. Tek. Hogsk., 1-159

Du, X., Jiao, J., Tseng, M.M., (2001). Architecture of Product Family: Fundamentals and Methodology. Concurrent Engineering. 9, 309-325

Ericsson, A, Erixon, G. (Eds.), (1999). Controlling Design Variants - Modular Product Platforms. Society of Manufacturing Engineers, Dearborn, Michigan

Gedell, S., Johannesson, H., (2013). Design rationale and system description aspects in product platform design: Focusing reuse in the design lifecycle phase. Concurrent engineering, research and applications. 21, 39-53

Gershenson, Prasad, Zhang, (2003). Product modularity: definitions and benefits. J. Eng. Des. (UK). 14, 295-313

Guo, F., Gershenson, J.K., (2007). Discovering Relationships Between Modularity and Cost. Journal of Intelligent Manufacturing. 18, 143-157

Hansen, C.L., Mortensen, N.H., Hvam, L., (2012). On the Market Aspect of Product Program Design: Towards a Definition of an Architecture of the Market. 12th International Design Conference - Design 2012

Harlou, U., 2006. Developing Product Families Based on Architectures - Contribution to a Theory of Product Families. Department of Mechanical Engineering, Technical University of Denmark, Kgs. Lyngby

Haug, A., Hvam, L., Mortensen, N.H., (2011). The impact of product configurators on lead times in engineering-oriented companies. AIEDAM. 25, 197-206

Hölttä-Otto, K., de Weck, O., (2007). Degree of Modularity in Engineering Systems and Products with Technical and Business Constraints. Concurrent engineering, research and applications. 15, 113-125

Hubka, V., Eder, W.E., (1988). Theory of technical systems: a total concept theory for engineering design. Berlin and New York, Springer-Verlag, 1988, 291 p. 1

Jiao, J., Tseng, M.M., (1999). A Methodology of Developing Product Family Architecture for Mass Customization. Journal of Intelligent Manufacturing. 10, 3-20

Joergensen, K., (1992). Videnskabelige arbejdsparadigmer (In Danish). Institute for Production, Aalborg University of Technology, Denmark

Ko, Y., Kuo, P., (2010). Modeling Concurrent Design Method for Product Variety. Concurrent Engineering. 18, 207-217

Krause, D., Eilmus, S., Jonas, H., (2013). Developing Modular Product Families with Perspectives for the Product Program, in: Smart Product Engineering. Springer, pp. 543-552

Latour, B., (1986). Visualization and cognition. Knowledge and society. 6, 1-40

Levandowski, C.E., Corin-Stig, D., Bergsjö, D., Forslund, A., Högman, U., Söderberg, R., Johannesson, H., (2013). An integrated approach to technology platform and product platform development. Concurrent engineering, research and applications. 21, 65-83

Meyer, M H, Lehnerd, A.P. (Eds.), (1997). The Power of Product Platforms - Building Value and Cost Leadership. The Free Press, New York

Mortensen, N.H., Hvam, L., Haug, A., Boelskifte, P., Hansen, C.L., (2010). Making Product Customization Profitable. International journal of industrial engineering. 17, 25-35

Otto, K N, Wood, K.L. (Eds.), (2001). Product design: techniques in reverse engineering and new product development. Tsinghua University Press

Pahl, G, Beitz, W., Wallace, K. (Eds.), (1996). Engineering Design - A systematic approach. Springer, London

Pimmler, T.U., Eppinger, S.D., (1994). Integration Analysis of Product Decompositions. American Society of Mechanical Engineers, Design Engineering Division. 68, 343-351

Prasad, (1999). Enabling principles of concurrency and simultaneity in concurrent engineering. (AI EDAM) Artif. Intell. Eng. Des. Anal. Manuf. (UK) Artificial intelligence for engineering design, analysis and manufacturing. 13, 185-204

Prasad, B (Ed.), (1996). Concurrent engineering fundamentals. Prentice Hall PTR

Steward, D.V., (1981). The design structure system: A method for managing the design of complex systems. IEEE Trans. Eng. Manage. EM-28, 71-74

Ulrich, K., (1995). The Role of Product Architecture in the Manufacturing Firm. Research Policy. 24, 419-440

Ulrich, K, Eppinger, S. (Eds.), (2000). Product Design and Development. McGraw-Hill, Boston

Zoia, D.E., (2013). Global Suppliers Favor German OEMs, Survey Finds. Ward's Auto World. 49

Paper C

Bruun, H.P.L., Mortensen, N.H., Harlou, U.,
PLM support for development of modular product families,
Conference proceedings of ICED 2013, Seoul, South Korea.

ICED13
INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED13
19-22 AUGUST 2013, SUNGKYUNKWAN UNIVERSITY, SEOUL, KOREA

PLM SUPPORT FOR DEVELOPMENT OF MODULAR PRODUCT FAMILIES

Hans Peter Lomholt BRUUN (1), Niels Henrik MORTENSEN (1), Ulf HARLOU (2)
1: Technical University of Denmark, Denmark; 2: Center for Product Customization, Denmark

ABSTRACT
Most modern manufacturing companies use a PLM/PDM system for documenting and managing product data. Companies use their PLM/PDM system for management of CAD files, documents, and drawings, but they do not take advantage of the full potential of the system to support modularisation. The objective of this research is to develop an approach for improving the role of PLM/PDM systems as supporting tools for developing modular product families. The approach is based on a visual product architecture model; representing a product family seen from a functional system perspective and a physical modular perspective. By means of a software program, product structures visual modelled can be imported to a PLM system, forming so called upper structures. Data associativity between upper structures in the PLM system and CAD models is described, as well as other types of associated product information. The key result of the research is the approach of using companies' PLM systems to build up and define product structures that support the activities of creating modular product families.

Keywords: product families, product lifecycle management, product architecture, modularity, interface management

Contact:
Hans Peter Lomholt Bruun
Technical University of Denmark
Mechanical Engineering
Lyngby
2800
Denmark
hplb@mek.dtu.dk

1. INTRODUCTION

The manufacturing industry has in decades been an intensive user of data-systems to drive quality and efficiency, adopting information technology (IT), and automation for design, production, and distributing products. It is hence not new that manufacturing companies use IT systems to manage the product lifecycle, including computer aided-design (CAD), engineering, manufacturing and product management tools (Manyika, 2011). Product Data Management (PDM) emerged in the late 1980s as manufacturing companies recognised a need to keep track of ever growing volumes of design files generated in CAD systems. PDM can in many ways be seen as a subset of modern PLM systems (Product Lifecycle Management) (Sääksvuori & Immonen, 2008), and is in this paper treated as so. The main idea behind PLM systems is that companies can create more value by integrating data from multiple systems. This is done by obtaining synergies of all available product related data, and to eliminate redundant data existing in different system environments. PLM is often described as an integrated, information-driven strategy of managing the whole life cycle of a product starting from generating an idea, concept description, business analyses, product design and solution architecture, technical implementation, and product testing, to the entrance to the market, service, maintenance, product improvement, and recycling (Stark, 2007). A concurrent initiative which the manufacturing industry has picked up in the last decades is the initiative of multi-product development/development of product families. The approach, described by many authors, strives to create variety at low cost by utilising commonalities or synergies between products and/or processes (Ericsson & Erixon, 1999; Harlou, 2006; Hölttä-Otto & de Weck, 2007; Martin, 2002). Product families can be defined as the products that share a common platform but have specific features and functionality required by different customers. While a subset of a platform is defined as the subsystems and interfaces common to all products in the product family, the central aspect of modularisation has to be addressed. The outcome of modularisation can be seen as the encapsulation of complexity by breaking down systems to manageable parts i.e. a modular structure. The activities of handling product development, and the related production development, are influenced by the degree by which the components and modules are coupled. Moreover development activities are influenced by the degree by which the components are re-used or pre-used in a number of products. The main hypothesis of this paper is that proper application of modern PLM systems can enhance the success of modularisation. This can be achieved because a PLM system is a strong tool not only for managing product data, but also for documenting modular structures, supporting concurrent engineering, and managing the variants of systems, modules, and interfaces within a product family.

1.1 Research Motivation

There seems to be some key reasons for developing and describing an approach for using PLM systems to support the development of modular product families. The challenge that companies face is, among others, the lack of integration between module data only present in the PLM/PDM system and the information about the modules that is captured in CAD. Some of the consequences of not having properly documentation and not having aligned data when developing modular architectures are:

- Difficult to recognise and manage modules and interfaces.
- Difficult for designers to find and re-use existing modules.
- Unclear what is standard and what is customised for a specific project or product.
- Difficult to carry out proper monitoring of relational properties and design progress during a project (e.g. cost of modules).

The research question which the authors seek to answer in this paper is: *How can PLM systems support the development of modular products?* The research belongs to the area of designing i.e. the aim of the research is to improve the design activity, and the effect on practice, directly or indirectly, is addressed in the descriptions of results.

2. ADJACENT FIELDS OF RESEARCH

In order to clarify aspects relevant to this research, the following section is recording briefly some main contributors to this area. The themes of the contributions include:

2.1 The introduction of enterprise PLM systems in companies

PDM technology is intensively used in industry and today its application is mainly focused on particular product lifecycle phases, e.g., development, prototyping, or production (Abramovici, 2007; Sääksvuori & Immonen, 2008; Stark, 2011). In recent years PDM vendors and integrators have found a multitude of acronyms, e.g., PDM Link, Team Center PDM, Collaborative Product Development (cPDM), 3D Product Lifecycle Management (3D-PLM), or Virtual Product Development (VPDM). In reality acronyms and descriptions are converging to PLM. PLM is the extension of PDM towards a comprehensive approach for product related information and knowledge management within an enterprise. This includes planning and controlling of processes that are required for managing data, documents and enterprise resources throughout the entire product lifecycle (Abramovici, 2007).

2.2 Modularisation

Modularisation is used as a foundation for identifying and developing product architectures and standard designs (Harlou, 2006). The sub-systems in modular product architectures are referred to as modules. One of the most common explanations of a module is provided by Baldwin and Clark (2000), who define a module as an unit, whose structural elements are powerfully connected among themselves, and relatively weakly connected to elements in other units. This explanation can be used to describe the essence of modularisation, because the split between generic and variable characteristics is possible due to the level of integration between the elements and units. Several authors describe approaches for developing modular products, which can assist in the development process (Erixon, 1998; Krause & Eilmus, 2011; Steward, 1981) .

2.3 Product architecture

In order to account for the relations between the meetings encountered by a product through its life cycle phases, structures can be defined for every life cycle phases, which are to be taken into account during development (Andreasen, Hansen, & Mortensen, 1996). The alignment of structures of the life cycle phases may be outlined as architectures. One definition of architecture is that it is a "purposefully aligned structure of systems" (Andreasen, Mortensen, & Harlou, 2004). The product architecture defines the basic building blocks of the product. Both in terms of what they are able to do, and what their interfaces with the surroundings and the rest of the device are (Ulrich, Eppinger, & Goyal, 2011). To sum up, the product architecture holds the information on how many commercial variants the product family consists of, how many components the products consists of, how these components work together, how they are built and assembled, how they are used, and how they are disassembled.

2.4 Interface management

Interface management deals with the issue of component integration (Sundgren, 2003). Component integration is the process of identifying all functional and physical characteristics of interacting entities from different organisations. Furthermore it is to ensure that proposed changes to characteristics are assessed and approved before implementation. Interfaces can be understood as linkages shared among components of a given product architecture. Depending on the level of analysis, a component can be a part, a module, a sub-system, or system (Mikkola, 2001).

2.5 Model based definition

Model-based definition (MBD) is a new strategy of PLM based on CAD model's transition, from simple gatherers of geometrical data, to comprehensive sources of information for the overall product lifecycle. With MBD, most of the data related to a product are structured inside native CAD models, instead of being scattered in different forms through the PLM database (Alemanni, Destefanis, & Vezzetti, 2011).

2.6 Conclusion

All of the above described research areas are related to the approach and model described in this paper. The approach is grounded in an enterprise PLM system that is customised to support the functionality of handling a product seen from a functional system viewpoint and a physical modularisation viewpoint. The approach encompasses a representation of a visual product architecture to model the product components, their arrangement, and the interfaces among them. The PLM system can be used

as an operational interface management system ensuring that components, modules, or sub-systems are compatible. The approach cannot be characterised as a MBD, because data is not structured inside CAD models. It can instead be characterised as an architecture based definition, in which product data is structured according to systems, modules, and their interfaces.

3. APPROACH FOR PLM SUPPORT FOR DEVELOPING MODULAR PRODUCT FAMILIES

The assumption behind this research is that visual product architectures in a beneficial way can assist the development of modular product families, because they model the product system in an explicit way. This creates overview of the products in the product family, and provides a basis for creating modules that are interchangeable when relevant. Most companies are already using a PLM/PDM system to manage product data in the development process, and there is a need for expanding the utilisation of these IT systems to also support the creation of modular product families. This section is presenting the approach of realising this. The product model formalism is based on the prior work of Harlou, but has been enhanced and expanded to cover a multiple structures view. The functionality of loading the IFD modelled structures into a PLM system, and the functionalities made possible in the PLM system, belongs also to the results of this research.

3.1 High level approach steps

The first step in the approach is to establish an architecture representation of a product family by means of the Interface diagram formalism (IFD) (Bruun & Mortensen, 2012a; Bruun & Mortensen, 2012b). Concurrent with this, a platform is set up in the companies PLM system in order to prepare the alignment with the objects defined in the IFD. In this research study PTC's Windchill 9.1 was used for the approach. In collaboration with the software vendor, functionalities were added to the commercial version for loading product structures defined in the IFD. By means of a small written software program the modelled structures from the IFD, made in Microsoft Visio, were retrieved and output was generated in XML-format, which could be imported to the PLM system. This upload is only performed one time, and further updates and alignment between the IFD and the PLM system is done manually with support in comparative reports. A comparative report is automatically generated and compares the structures from the IFD with the upper structures in the PLM system, outlining differences between them. In the user interface of the PLM system it is possible to link product information files to the structures. The structures become placeholders for information as CAD, design specifications, interface descriptions, purchase specifications, drawings, diagrams, test protocols etc. The structures are denoted Upper structures in the PLM system.

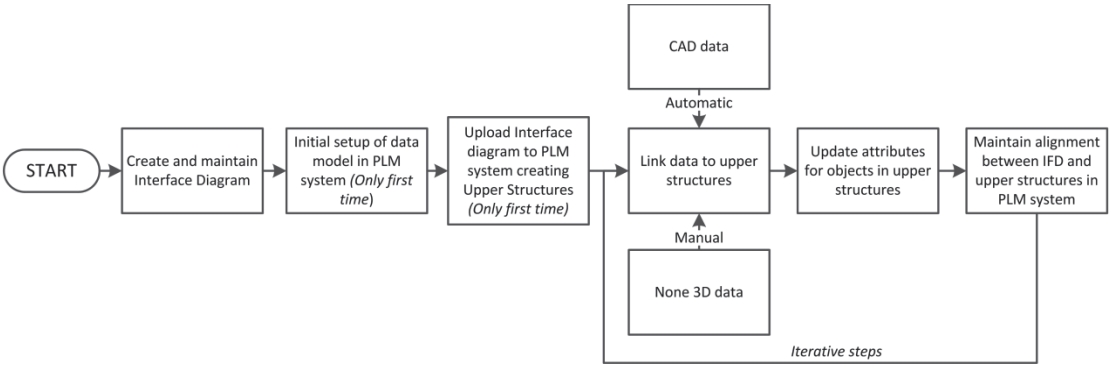


Figure 1. Simplified process model for bringing modularisation into a PLM system

Attributes like cost for objects in the PLM system can be assigned independent of the existence of CAD data for the object. This enables that a 'likely cost' can be monitored very early in the design phase. During the development process the IFD is updated while the design is matured and decisions on solutions are incorporated. These updates have to be reflected in the PLM system, and is done be comparing the structures modelled in the IFD with the structures established in the PLM system. As mentioned, this process is done manually by the support in comparative reports that outline which objects have to be added or removed from the upper structures in the PLM system.

3.2 Design management by IFD

The approach of using PLM system functionalities for supporting development of modular product families is made possible by the means of the IFD. An IFD is a visual product architecture representation, capturing structural characteristics of a product family, mostly combining the aspect of mapping between technology domains, and mainly a mapping between physical structures. The model puts emphasis on managing interfaces between components in the model, hence the chosen name of the modelling tool. The formalism has its basis in the Generic organ diagram presented in the work of Harlou (2006). Because of secrecy issues for the company involved, the formalism is illustrated and described by using an example of an IFD for a product family of Bobcats. The model puts emphasis on handling the product family seen from different viewpoints. The main viewpoint is a system perspective i.e. the perspective that deals with the product's main functions or its related lifecycle. Figure 2 (left side) shows the architecture of two of the systems in the Bobcat; the hydraulic system and drive train system. The two systems are physically allocated to different spatial sections of the Bobcat, connected by physical interfaces as hydraulic hoses, electrical wires, transmissions belts etc.

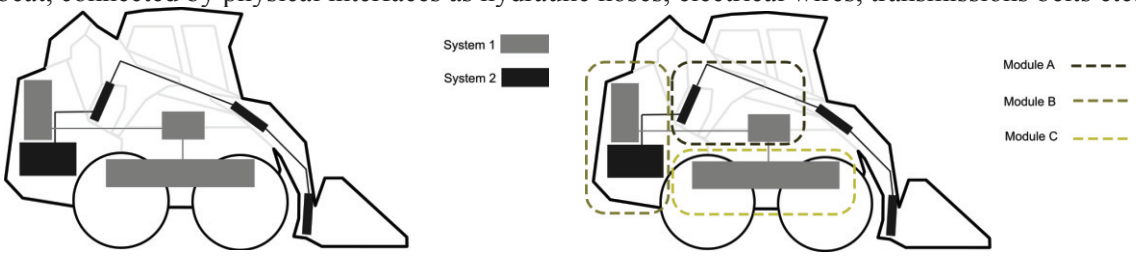


Figure 2. System structure (left) and a Module structure (right) of a bobcat showing entities belonging to systems and modules and the relations between them

The second viewpoint, handled in the IFD, is a modular viewpoint in which systems are split and physically joined components are encapsulated into modules. Modules can consist of elements belonging to different systems i.e. developed by different system teams. It is therefore crucial both to integrate systems in modules, but also to handle interfaces created along the module boundaries splitting systems. Figure 2 (Right side) shows the architecture in which the two systems are split in three modules: Base frame module, Engine module, and Hydraulic module. When monitoring the IFD in Visio, it is possible to turn on layers containing specific systems and the interfaces belonging to it. Moreover it is possible to monitor interfaces to other systems. The modelling formalism has been described in prior published work by the authors, but will in short be introduced here. The IFD is modelled by means of blocks and lines in the software program Microsoft Visio. The IFD is normally printed on large blue prints in order to get the overview of the product family it represents. The IFD is used as a boundary object between different developers. Figure 3 is a symbolic representation of an IFD. In order for the reader to quickly understand the structure of the product, it is suitable to model the diagram so the layout is established as a cross section of the actual product. In that way the physical layout of the product is possible to recognise in the diagram. The diagram can be read following the interfaces. For products that process or transforms objects, this gives a logical reading direction, for other products it is up to the reader to find a suitable flow in the model. The main elements of the diagram formalism are objects denoted Interface components (IF components). The purpose of the IF components is to decompose the product family into systems and encapsulate the building blocks into modules. IF components have different characteristics and are modelled in different ways. Each IF component belongs to a product system. To avoid confusion, systems and their interaction must be clearly defined. This is done by choosing the relevant interaction as a basis for determining the system boundary. There is no fixed list of systems to be included in the development. Systems can be modelled and thereby control important properties of the final product. Modules are modelled by arranging IF components inside boxes with a thick black boundary and rounded corners. Modules can contain smaller modules, but they do not overlap as it is clearly defined to which module any element in a product belongs.

The structure of the IFD appears as the relations i.e. interfaces are added to the diagram. An interface, among two IF components, represents a relation, e.g. a physical connection, energy transportation, information flow, or flow of material. The purpose of working with interfaces is to ensure responsibility for the components interaction and to ensure that components are interchangeable, when

relevant. An interface between two IF components holds a definition on which is the master and the slave of the interface. This enables a responsible for an IF component to monitor whether he owns the right to change or modify the related interface. There exist a number of interface classes and the list can be extended in order to support the context in which a product belongs. Optional interfaces can be modelled on the diagram to show affected relations between entities or systems if the interface is established.

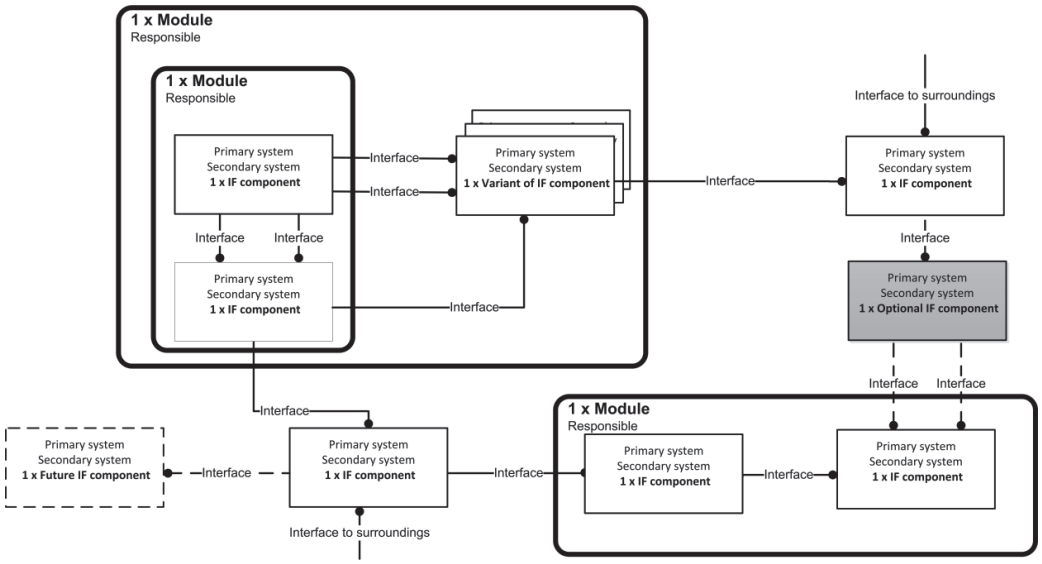


Figure 3. Symbolic representation of a generic IFD

3.3 PLM support

In this research case, the company's PLM system had features typical for most enterprise PLM systems on the market today (Sääksvuori & Immonen, 2008). The features of the system included: Item management, product structure management, user privilege management, maintenance of the state and status of items, information retrieval, change management, configuration management, workflow management, document management, backup management, history/system log, file vault etc. The PLM system was implemented in parallel with the company's latest product development project, in which the PLM system became the comprehensive source for information belonging to the new product family. The studied company had a great focus on modularisation and standardisation of their products. Because of the complexity of the company's products each containing more than 40.000 unique parts, there was seen potential in developing modules that could be shared by different products. The process of modularisation was handled in many levels in the company, and was supported by a number of methods and tools. By using the IFD from the early design phases and maintaining it during the entire design process, different module concepts could be tested in a fast and simple way. Modularisation supported in the IFD and documented in the PLM system, created some effects that are listed here:

- The definition and development of a product family's system structure: Systems (e.g. cooling or hydraulic) are defined in a bill-of-material seen from a functional point of view. This means that required behavioural properties of a system can be targeted in development, by designing system in their totality. For example can 3D visualisations of the systems be generated in the PLM system, even though they are realised by means of several modules, each containing several system elements.
- The definition and development of a product family's modular structure: Modules are defined in a bill-of-material seen from a modular point of view. A module is created because of different driving forces for modularization, and because a module is composed by components designed by different systems, it is important to design the relations between system and modules, so that the products both fulfill requirements of behavioral properties and requirements to physical realisation in production. 3D visualisations and

reporting of relational properties such as cost and weight of each module, can be created by using PLM system functionalities.

- The definition and development of a product family's interface structure: Interfaces are clearly defined in both a system and a module context. Interfaces can be defined in a structure representing different interface types, and it is clearly defined which components and modules are related to each interface, and which system holds the responsibility of an interface. The template of Interface documents was developed to support the management and the integrity of the modular architecture. Interface documents have the nature of design specifications that can be linked to the interface structure in the PLM system.

For CAD models so called WT parts (Windchill Technology parts) are created automatically when creating CAD objects (assemblies and parts) inside the PLM environment. WT parts are objects that can be linked to upper structures in the PLM system. This is done in order to be able to impose the CAD models according to the chosen structure of interaction (system, module, and interface). Non CAD data is linked to upper structures in the same way as for CAD data. This organises data in a way that is identifiable with the product architecture modelled in the IFD.

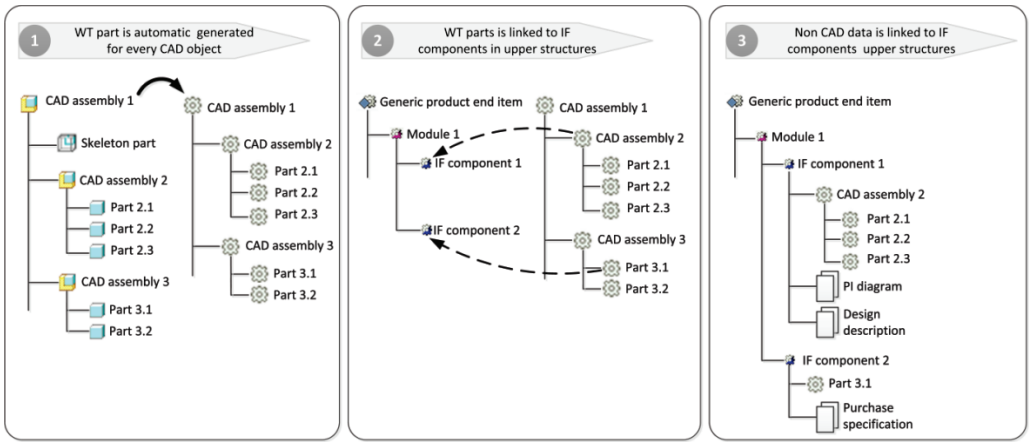


Figure 4. Data Associativity in the PLM system

Visualisation of products seen from different perspectives is seen as a strength also in the conceptual phase in order to enhance concurrent engineering, see implications of introducing module boundaries, and in general when focusing on integration of different systems. Figure 5 shows an example from the PLM system in which a module of a front loader and a hydraulic system are loaded in a web-browser. Because CAD models are linked to IF components represented in both a system and a module structure, the CAD models are imposed in proportion to the chosen structure.

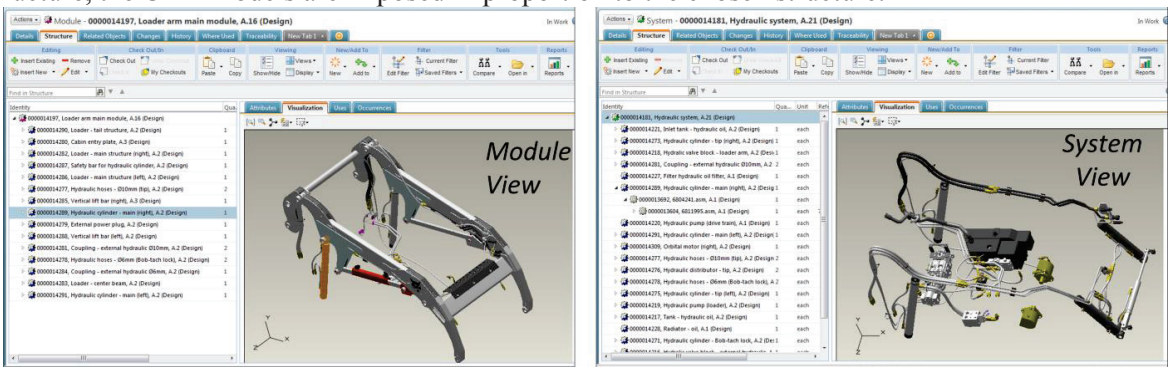


Figure 5. Visualisation of module and system view

Reporting capabilities in the PLM system is used to monitor design progress on a weekly basis. The numbers of new, modified, and approved parts are indications on the progress of the design. Furthermore a report on the cost development is generated each week.

Every component in the system is allocated with an initial target cost attribute, and gradually direct cost is added to all components as they are developed or purchased. By means of simple roll up mechanisms cost are added up from the top product level and down to the smallest components. If a

cost figure is missing on a component, a target cost steps in. Target cost numbers are provided by financial controllers in collaboration with the relevant developers and purchasers. Cost reporting is in that way a strong tool for supporting a design according to budget, because difference between budget and actual cost is discovered quickly.

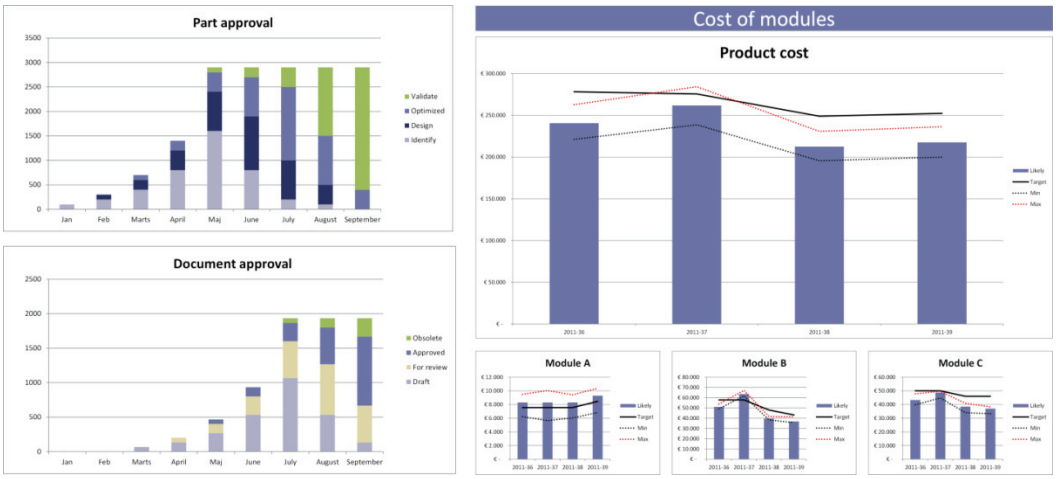


Figure 6. Example of reports showing design progress and cost development generated from the PLM system on a weekly basis

4. DISCUSSION

This contribution should be regarded as outlining a suggested approach for supporting the activities of developing modular product families. The approach has focus on the abilities in modern PLM systems. The results presented here do not represent a complete framework to describe the development of modular product families, but serve as a contribution of importance to the framework of authors in the area of practical application of supporting tools. A strength in this approach is that it can be used in companies developing products belonging to a diversity of products and industry. Even if strict definitions differ, the fundamental principles of modular design are common: break systems into modules, ensure modules can interchange with each other, and provide well-defined interfaces. Using a visual product architecture model in combination with a PLM system in the design process, support development on a conceptual high level, and still supports management of information on a very detailed level. Support functionalities are: definition of functional system structures, physical module structures, and the interfaces between entities, management of interfaces, and reporting of design progress and cost development. Any products could potentially be modularised and their information could be managed in PLM systems, but to gain more easily benefit from the advantages of PLM systems, companies must be selective in choosing which products to support with this approach. The strength in this approach is its ability to address complex products with a large number of shared components, components belonging to different technology domains, and/or components developed by different organisational teams. To apply this approach for simple products with few components could be categorised as “killing butterflies with muskets”.

5. CONCLUSION

This paper has presented an approach for using a visual product architecture model in combination with a PLM system for supporting the activities of developing modular product families. The initial hypothesis has been tested success. The research question has been answered by illustrating the results of the research. One can ask: What is the validity of such a model and an approach? Our proposals shall be meaningful to practitioners, they shall be easy to integrate into their practice and support their reasoning; this is where the validity shall be found. The validity should be found in the feedback through unstructured interviews in the company, reporting the ease of application and the beneficial PLM support in designing modules and maintaining an interface integrity of the product family. The approach is seen as applicable for development of complex products as e.g. members of a product family, with a large number of components, and in less degree as an approach to choose, when

developing simple products with few components and simple technology. The approach has been implemented in an industrial setting and the first results by doing this, have been described. The most important results are listed in terms of the experienced effects on the modular design process in the company:

- Clearly defined systems - an enabler for managing the design process and the responsibility within each system.
- Clearly defined modules - an enabler for re-use of modules.
- Clearly defined interfaces - an enabler for re-use of modules.
- More accurate cost reporting because of automated reporting based on modules.
- More accurate design progress reporting because of automated reporting based on modules

The boundaries of this PLM approach are that it focuses on establishing and managing data of modular product architecture. It is an approach that aims at supporting the engineers working on developing modular product structures by managing module documentation and information on interfaces between modules. The approach is not an extensive PLM approach, supporting all stages of a product's lifecycle, e.g. as recycling. Many preconditions and prerequisites exist for successful implementation of modular product architecture based development, which have not been described in detail in this paper. Conditions as e.g. a modern IT-infrastructure, organisational ownership, sufficient resources/competences and high-level anchoring of the initiatives, are all seen as conditions that to some extent have to be in place in companies. Regarding further work, detailing and elaboration of the interface management process could be mentioned, as well as more testing and refinement of the general approach.

REFERENCES

Abramovici, M. (2007). Future trends in product lifecycle management (PLM). *The Future of Product Development,* , 665-674.

Alemanni, M., Destefanis, F., & Vezzetti, E. (2011). Model-based definition design in the product lifecycle management scenario. *The International Journal of Advanced Manufacturing Technology, 52*(1), 1-14.

Andreasen, M., Hansen, C., & Mortensen, N. H. (1996). The structuring of products and product programmes. *Proceedings of WDK2 1996,* , 4, 28.

Andreasen, M., Mortensen, N., & Harlou, U. (2004). Multi product Development–New models and concepts. *15. Symposium" Design for X", Neukirchen, 2004,* 75-86.

Baldwin, C. Y., & Clark, K. B. (2000). *Design rules, volume 1: The power of modularity* mIt Press.

Bruun, H. P. L., & Mortensen, N. H. (2012a). Modelling and using product architectures in mechatronic product development. *Norddesign 2012, Aalborg, Denmark,*

Bruun, H. P. L., & Mortensen, N. H. (2012b). Visual product architecture modelling for structuring data in a PLM system. *PLM12, Montreal, Canada,*

Ericsson, A., & Erixon, G. (1999). *Controlling design variants modular product platforms*. Dearborn, Michigan: Society of Manufacturing Engineers.

Erixon, G. (1998). *MFD–Modular Function Deployment, A Systematic Method and Procedure for Company Supportive Product Modularisation,*

Harlou, U. (2006). *Developing product families based on architectures contribution to a theory of product families*. Lyngby: Department of Mechanical Engineering, Technical University of Denmark.

Hölttä-Otto, K., & de Weck, O. (2007). Degree of modularity in engineering systems and products with technical and business constraints. *Concurrent Engineering, 15*(2), 113-125.

Krause, D., & Eilmus, S. (2011). Methodical support for the development of modular product families. *The Future of Design Methodology,* , 35.

Manyika, J., et. al. (2011). *Big data: The next frontier for innovation, competition, and productivity.* ().McKinsey Global Institute.

Martin, M. V. V. (2002). Design for variety: Developing standardized and modularized product platform architectures. *Research in Engineering Design, 13*(4), 213-235.

Mikkola, J. H. (2001). Modularity and interface management of product architectures. *Management of Engineering and Technology, 2001. PICMET'01. Portland International Conference On,* 599-609.

Sääksvuori, A., & Immonen, A. (2008). *Product lifecycle management* Springer Verlag.

Stark, J. (2007). *Global product: Strategy, product lifecycle management and the billion customer question* Springer Verlag.
Stark, J. (2011). Product lifecycle management. *Product Lifecycle Management,* , 1-16.
Steward, D. V. (1981). The design structure system: A method for managing the design of complex systems. *IEEE Transactions on Engineering Management,* (3), 71-74.
Sundgren, N. (2003). Introducing interface management in new product family development. *Journal of Product Innovation Management, 16*(1), 40-51.
Ulrich, K. T., Eppinger, S. D., & Goyal, A. (2011). *Product design and development* Irwin/McGraw-Hill.

Paper D

Bruun, H.P.L., Mortensen, N.H., Harlou, U., Wörösch, M., Proschowsky, M.,
PLM system support for modular product development,
Journal of Computers in Industry, In 2nd review 2014.

PLM system support for modular product development

1. Abstract

A modular design strategy both enables, but also demands, parallelism in design activities and collaboration between a diversity of disciplines in companies, which often involves supporting computer-based tools for enhancing interaction, design management, and communication. Product Data Management (PDM) and Product Lifecycle Management (PLM) systems offer support by automating and managing some of the operational complexity of modular design activities. PLM system tools are used for handling a variety of product definitions, to manage workflow of development activities, and to measure relational properties such as cost and performance. Companies often use a PLM tool for management of CAD files, documents, and drawings, but they do not take advantage of the full potential of the PLM system to support the development activities of modular product designs. The key result of this paper is the description of an empirical tested approach using a visual product architecture representation in combination with a PLM system to support the development of a product family of complex products. The results from the study encompass new PLM capabilities for handling multiple product structures, visualising multiple architectural views on products, controlling interfaces, and quantifying and communicating the status and progress of product-related resources.

Keywords: *Product lifecycle management (PLM), architecture based development, product models, modularisation, interface management.*

2. Introduction

In the wake of the financial crisis companies are facing increasing pressure to develop solutions faster and at lower cost. The changed circumstances put great pressure on manufacturing companies, requiring them to increase their R&D efficiency by providing new competitive solutions faster; and at the same time cut costs by improving product quality and productivity in production. Modularisation, appropriately applied, can serve as a means to provide the variety needed from a customer point of view and at the same time reuse sub-solutions across different products to improve time-to-market and maintain predictable product quality [1]. However, the complexity of man-made product systems has in recent years grown to an unprecedented level. Many companies do not succeed in developing modular architectures, as modules are not easy to identify in products where key functionality is distributed across the product structure [2]. In this situation computer-based supporting tools have become a necessity in order to handle complexity in all aspects of the product design. One strategy for handling complex product systems is the introduction of PLM system tools, to obtain comprehensible product representations without neglecting relevant aspect and dependency types [3, 4]. The most prominent idea behind PLM systems, or the top-down perspective on new product development, is that companies can create more value by integrating data from multiple systems in order to obtain synergies of all available product-related data and to eliminate redundant data existing in different system environments. In that way PLM systems aim to support collaborative work within product design processes in order to integrate partners and all associated knowledge efficiently [5-7]. A secondary view on PLM technology is related to a bottom-up perspective. It considers that knowledge of available tools can allow appropriate solutions to be found for company-specific problems. Both perspectives are considered equally important when approaching PLM [8, 9]. However, many manufacturing companies deploy PLM systems in an ineffective way, merely for documenting and managing product data as CAD files, product related documents, and drawings. The result is that firms might find themselves quite far from the expected operational or strategic outcomes from PLM tools [10, 11, 11, 12].

This paper is based on the assumption that the definition of a visual architecture representation and the operational handling of it in a PLM system can enable companies to overcome the challenging situation of identifying modules when developing product families. The paper describes an approach

for deploying PLM system tools to support the development of modular products, providing functionalities for defining and evaluating modules and their interfaces in a product family context. The approach is based on a visual architecture description, expressing a product family's structures (how it is built up) seen from different viewpoints. Multi-viewpoint product modelling for different purposes has been the focus for several researches, e.g. multi-viewpoint models for various engineering specialist involved in a complex mechanical design [13], multiple domain modelling for uncovering dependencies of mechatronic products [14], and modelling of viewpoint relationships between products, manufacturing processes and resources used to produce them [15] .

An architecture viewpoint here is defined as a work product establishing the conventions for the construction, interpretation and use, to frame specific system concerns [16]. The architecture description can be used for configuration, i.e. identification of modules and module variations, and basic rules for interaction and interfaces between modules. This again allows grouping of variants according to commonality and application areas.

The visual architecture description, however, has some drawbacks due to the large amount of information that has to be visually represented and updated during development. The challenges by using visual product models for developing modular product designs are linked to the difficulties in handling large amounts of information in a dynamic way. Some of the challenges are, among others:

- Difficult to integrate the development over several departments, totalling thousands of employees.
- Difficult to recognise and manage modules and interfaces because of product data scattered in both CAD systems and other architecture descriptions.
- Difficult for designers to find and re-use existing modules.
- Difficult to carry out monitoring of design progress during a project because of the distributed development.
- Difficult to manage interfaces i.e. designing interfaces and controlling interfaces for change.
- Difficult to see the implications of Engineering Change (EC) of already released structures.

The contribution of this paper is to describe the integration between a visual architecture description and product definitions in a PLM system environment. This is done in order to have the best of both worlds: Utilise visual architecture descriptions to create overview, improve communication and collaboration, and to support the creation of modular product structures; and to utilise a PLM system to manage and integrate product information from the architecture descriptions effectively.

The approach is the result of several action research-based studies, and has recently been applied in another case study with promising results, which is presented here. The empirical base for this study is a large Danish manufacturer of complex products related to the renewable energy sector. The PLM tool applied to the described approach is created by PTC®, and marketed under the name *PTC Windchill PDMLink 9.1*. The approach has also been implemented in Siemens TeamCenter®, but this case will not be included in this paper.

The paper is structured as follows: First, a brief introduction to the theoretical basis and the concepts relevant to this research. Second, the suggested approach and its methods and models are described. Then the case is described and the results by implementing the approach are presented. A discussion extracts the principles, relationships, and generalisations from the results. Finally, a conclusion sums up the experience of implementing and testing the approach in a case study and the suggestions on future work.

3. Theoretical foundation and related work

The literature relevant to this paper has its basis in the Theory of Technical Systems [17] and Theory of Domains [16]. The theories' primary role is the support of function and property reasoning. A secondary role of the theories is to support concepts for product modelling which again suggests wide application in architecture thinking. In accordance with Theory of Technical Systems, this article will primarily reserve the word *structure* for how individual products are built up and *architecture* will be reserved for describing how a product family is built up including the future derivative products.

Literature addresses a variety of concepts related to architecture thinking that in short will be described: product platform, product architecture, modularity, and product family. To expand the research base, the present status of PLM tools in industry has also been considered.

3.1 Product platform

The basic thinking pattern behind the later described architecture framework has strong relations to the concepts of product platforms. Platform thinking, the process of identifying and exploiting commonalities among a firm's offerings, target markets, and the processes for creating and delivering offerings, appears to be a successful strategy for creating variety at low costs. The commercial exploitation of a product platform is perhaps better known as mass customisation. Mass customisation is originally a manufacturing strategy aiming at satisfying individual customer requirements while keeping manufacturing cost and delivery times close to those of mass-produced products [18, 19]. A product platform has been defined by Meyer et. al [1] as a set of subsystems and interfaces that form a common structure from which a stream of related products can be efficiently developed and produced. Product platforms are therefore specifically designed to serve one specific group of related products. A product architecture is not essentially developed with this limitation in mind. Baldwin and Clark [20], page 77, define three aspects of the underlying logic of a product platform: (1) its modular architecture; (2) the interfaces (the scheme by which the modules interact and communicate); and (3) the design rules (that the modules conform to). The three aspects will be explained next, because they are important to the approach presented in this article of PLM support of modular product development.

3.2 Modular architecture

An important principle of architectures is the concept of modularity, i.e. the concept of decomposing a system into independent parts or modules that can be treated as logical units. This enables concurrent development of modules and forms the basis for configuration of product families [21, 22]. Modularity has been defined as the relationship between a product's functional and physical structures such that: there is a one-to-one or many-to-one correspondence between the functional and physical structures—and unintended interactions between modules are minimised [21]. Although exact definitions vary on modular architectures, the fundamental ideas are common throughout: Break systems into discrete modules; ensure modules can interchange with each other; and provide well-defined interfaces. Modular architectures thus fully specify component interfaces and therefore limit successive component development. The approaches for designing modular architectures are multiple. Some focus on the technical aspects of architectures [20, 21, 23, 24], while the economic aspects are dealt with by others [1, 2, 25, 26]. A modular design strategy both enables, but also demands, parallelism in design activities and collaboration between a diversity of disciplines in companies. This again involves supporting methods and tools for enhancing interaction, design management and communication. Architecture descriptions are used by the parties that create, utilize and manage product systems to improve communication and co-operation, enabling them to work in an integrated, coherent fashion. Architecture frameworks and architecture description languages are being created as assets that codify the conventions and common practices of architecting and the description of architectures within different communities and domains of application [27]. The product architecture defines the basic physical building blocks of the product in terms of what they do and what their interfaces with the surroundings and the rest of the device are [23].

3.3 Interfaces

In this paper, we address the part domain and use the term "interface" synonymously with part interface, being the connection between subsystems or components of a product. The subsequently explained *System structure* is based upon a functional product view where elements are grouped by their functional identity i.e. a system delivers one or more active effects. Several authors have classified interfaces in the part domain. Pimmler and Eppinger [28] proposed a taxonomy of interactions between components, namely spatial, energy, information, and material types. Interface management in the part domain deals with the issue of interface design and interface control [29, 30]. Interface design is about the creation, form, arrangement, and configuration of interfaces, whereas interface control focuses on ensuring compatibility of interfaces during the design phase. Ensuring

interface compatibility is the process of identifying all functional and physical characteristics of interacting entities from different organisations, and ensuring that proposed changes to these characteristics are assessed and approved before implementation. Interface management is an important aspect of the suggested approach of using PLM capabilities in product family development.

3.4 Design rules that modules conform to

Configuration is mainly the task of choosing between various variants for subsystems, while respecting certain design rules and constraints [31]. The encapsulation in modules is governed by the desired effects, and the clustering will depend on the drivers for a platform approach, i.e. whether product customisation, upgrading, or outsourcing etc. is the main reason for the grouping to take place [26]. Consequently, the module design has a strong relation to the strategic reasons behind the layout of an architecture and which products have to be derived from it. Modularisation is the task of deciding which characteristics to group and which characteristics to separate [20, 32-34]. Deciding on the common and variable proportions of a product family has strong ties with the product design and production capabilities, because they largely determine which attributes are easy to decouple, which again has an impact on the module creation [19, 35, 36]. The PLM system used in this approach has capabilities to monitor the impact of creating specific modules and thereby offers a proactive exploration of the optimal modular architecture for a product family.

3.5 Enterprise PDM/PLM systems

The functionalities of enterprise PDM tools have evolved in the last decades and today PDM is in many cases seen as a subset of modern PLM system tools [9]. Product Data Management (PDM) technology is intensively used in industry and today its application is mainly focused on particular product lifecycle phases, e.g., development, prototyping or production [5, 9, 37]. In recent years, PDM vendors and integrators have found a multitude of acronyms, e.g., PDMLink, TeamCenter PDM, Collaborative Product Development (cPDM), 3D Product Lifecycle Management (3D-PLM), and Virtual Product Development (VPDM). In reality, acronyms and descriptions are converging to Product Lifecycle Management (PLM). PLM is the extension of PDM towards a comprehensive approach for product-related information and its management within an enterprise. PLM refers to the management of the product definitions, e.g. product data within product development, but in the essence of the definition, product data of the whole lifecycle of the product. This includes planning and controlling of processes that are required for managing data, documents and enterprise resources throughout the entire product lifecycle [37]. The environment of PLM systems can be characterised by the co-existence of various independent tools, each based on its own specific product model [38]. This has been acknowledged by others and proposals of frameworks for supporting the full range of PLM information needing to form a common product definition have been described e.g. by Sudarsan [39].

3.6 Summarising on state of the art

The related work points out major methodological concepts related to the discipline of developing product families. However, there seems to be a gap in developing product families supported by PLM systems tools. Companies often use PLM systems for management of CAD files, documents, and drawings, and to access various independent computer systems holding their own product definitions. The coexistence of these independent tools with their own specific models of the product or product family brings redundancy in the digital product documentation. The strength of a PLM system derives from its shared product definition in which dependencies and relations can be both monitored and manipulated from a central entry point. The risk of having a support system without access to the content of product models is that important dependencies are overseen in the development process. There is therefore a need for making prescriptive studies of using PLM systems for supporting a common product definition in order to create the optimal balancing of a modular architecture.

4. Research methodology

The approach presented in this article is the result of several action research-based studies conducted with manufacturing companies. These studies have matured the approach to its current state and subsequently it has been tested in a case study with promising results. Engaging in the practical setting of the case study, different types of inquiries were in practice. During the analysis phase of the study,

the inquiries were rather exploratory and diagnostically based, helping the researchers to understand the situation and assess the applicability of the approach. Moving on to the synthesis phase, the inquiries changed to having more of a confronting character and being directly prescriptive.

As one of the aims of the research is to bridge information from different engineering domains, there was a need to create a boundary object enabling the different competences to interact, exchange ideas, and understand each other's work challenges [40]. Thus visualisation has been used to create such a boundary object to facilitate collective alignment among engineering professionals. From the early stages of the project the concept architecture has been illustrated on a large A0 poster, allowing professionals with different backgrounds to gather around a large poster and make review meetings efficient by taking advantage of the optical consistency such a visualisation represents. The working method enables participants to lay aside their daily working habits and see the challenges in the project as the 'same type'.

5. Approach for using PLM to support modular development

As stated in the introduction, the approach is grounded in an architecture modelling framework developed at DTU [41]. The architecture framework is a best practice methodology that prescribes the alignment between a company's products and their production, marketing, sales, distribution and the company's involvement in delivering service and actively participating in re-use, recovery and disposal. The relation between the architecture framework and the PLM support approach is highlighted in figure 1 (*Scope of the approach*).The assumptions behind the framework will in short be explained in the following.

The framework covers four perspectives on a product system: The market, the product family (how it is built), the production, and the roadmap for deployment. The purpose of modelling the market architecture is to bring precision into decision making concerning the choice of which segments to cover or not cover and what properties are needed in order to do so across different business areas with different applications. A clearly defined market architecture is able to guide and control the engineering efforts towards profitability by scoping product family design. The modelling of product architectures covers the basic structural elements of a product family and the behavioural functional abilities. In other words, the aim of these modelling techniques is not solely to describe what the product architecture is, but also what the product architecture is able to do. Depending on the size of the product architecture development project, the associated production system will need an update, a modification or a complete redesign. The production system is designed coherently, as the product architecture matures and passes from concept to detailed design. Finally, the behavioural aspects of the market-, product- and production architecture are considered in the "architectures'" future launch preparedness. This is a function of the architecture, explaining what the architecture is able to do. This ability is modelled by visualising the launches, derivative products and specific product updates.

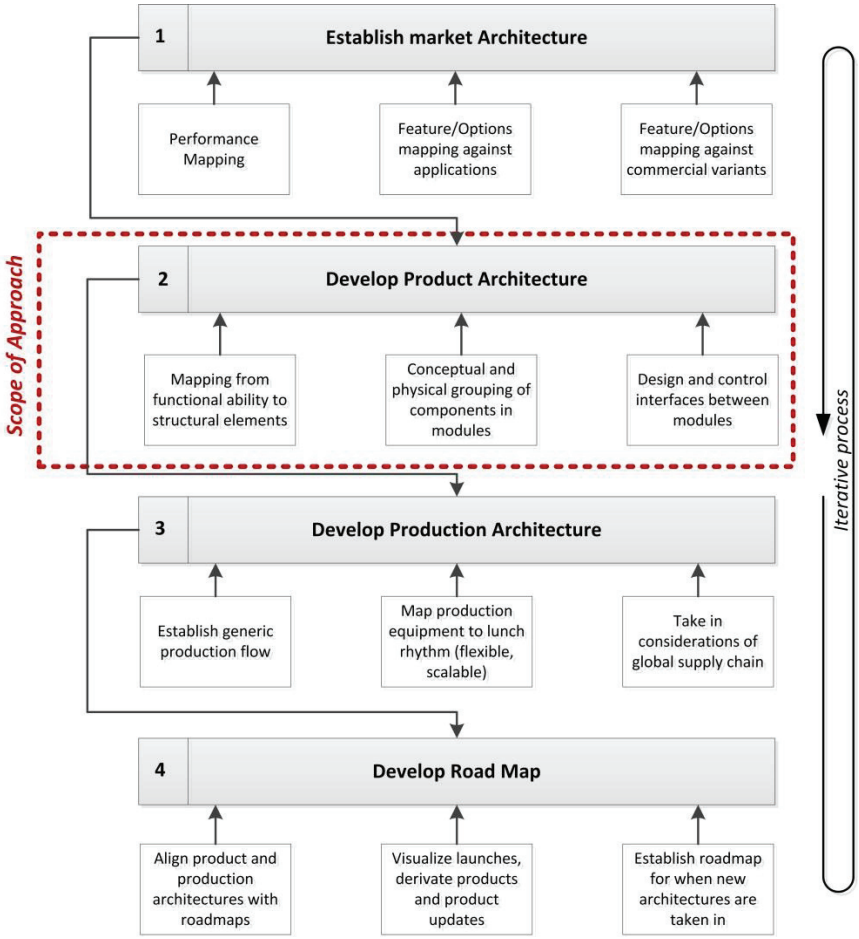


FIGURE 1. FRAMEWORK FOR ARCHITECTURE-BASED PRODUCT FAMILY DEVELOPMENT, BASED ON WORK BY [41]

The approach is striving to support the development of the product architecture i.e. the mapping from functional ability to structural elements, the conceptual and physical grouping of components in modules, and the design and control of interfaces between modules. The steps of the approach of using PLM systems for supporting development of modular designs is outlined in figure 2. Each step will be covered in the subsequent sections.

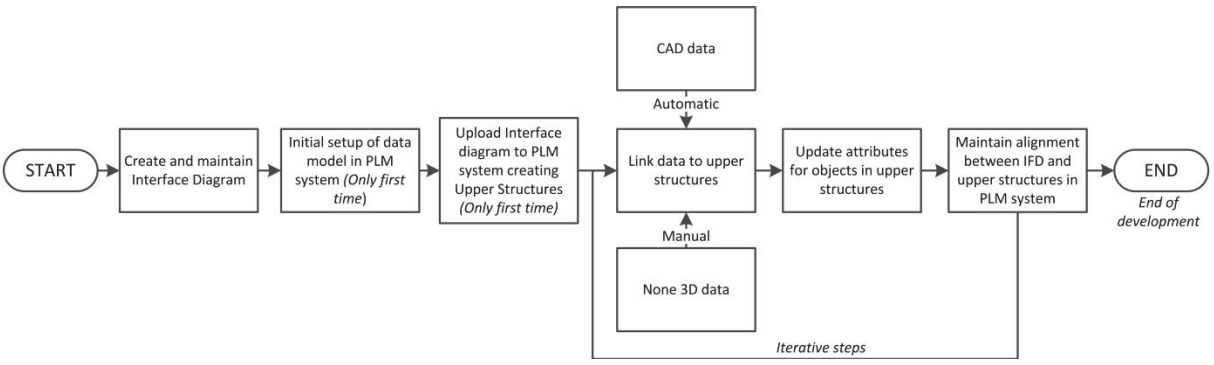


FIGURE 2. STEPS IN THE DESCRIBED PLM APPROACH

5.1 Create and maintain Interface diagram

As outlined in figure 2, the approach begins by establishing a visual product architecture representation named *Interface diagram.* An Interface diagram is an architecture representation capturing structural characteristics of a product family, combining the aspect of mapping between functional ability and structural elements domains. The model puts emphasis on managing interfaces between components in the model, hence the chosen name of the modelling tool. The formalism is

denoted *Interface diagram* which has its basis in the *Generic organ diagram* presented in the work of Harlou [42].

The formalism is illustrated and described by using an example of an Interface diagram conducted for a product family of *Bobcats*. The model puts emphasis on handling the product family seen from different viewpoints. The main viewpoint is a systems perspective i.e. perspectives that deal with the product's main functions or a related lifecycle. Figure 3 shows the viewpoint of two of the systems in the Bobcat: the hydraulic system (System 2) and drive train system (System 1). The two systems are physically allocated to different sections of the Bobcat and are connected by interfaces such as hydraulic hoses, electrical wires, transmission belts etc.

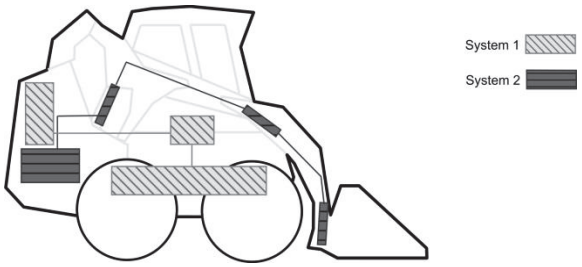


FIGURE 3. SYSTEMS VIEWPOINT OF A BOBCAT SHOWING ENTITIES BELONGING TO SYSTEMS AND THE RELATIONS BETWEEN THEM

As seen above, the components included in the system can be spread throughout the complete vehicle. The purpose of focusing on system development is to support functionality in components that are spread across multiple modules. Systems are often characterised by one or more of the following:

- Deliver important functionality, e.g. steering, braking, loading etc.
- Is a complex or new technology, e.g. hydraulics, cooling etc.
- Organisation – alignment with organisational structure and/or affecting organisational structure.

The objective of the system design is to focus the design on functionality and performance. An example of a system in the Bobcat could be the hydraulic system. It consists of a hydraulic pump, driven by the combustion engine, and actuators to manoeuvre the lifting arm and shovel; valves, filters, pipes, hoses, oil reservoir etc. All parts have to be taken into account when designing and dimensioning the system to meet the required performance properties. The same component can, from a functional perspective, be a part of multiple systems e.g. the combustion engine which is both a part of the hydraulic system and the drive system.

Because of initiatives for modularisation (carry over, serviceability of modules, outsourcing of design and production etc.) it is appropriate to manufacture the Bobcat in modules. The second viewpoint handled in the Interface diagram is a modular viewpoint in which systems are split and physically joined components are encapsulated into modules. Modules can consist of elements belonging to different systems, i.e. developed by different system teams. It is therefore crucial both to integrate systems in modules and to handle interfaces created along the module boundaries splitting systems. Figure 4 shows the conceptual architecture in which the two systems are split into three modules—Base frame module, Engine module, and Hydraulic module.

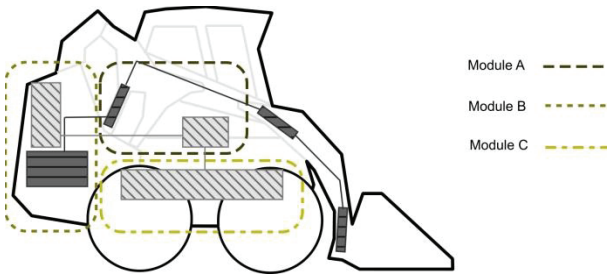


FIGURE 4. MODULE VIEWPOINT OF A BOBCAT SHOWING ENTITIES BELONGING TO MODULES AND RELATIONS BETWEEN SYSTEMS AND MODULES

As an example, the Hydraulic module is a physical sub-assembly consisting of the hydraulic pump, the oil reservoir, hosing, and structural components such as a frame, clamps, and screws.

The modelling formalism of the Interface diagram has been described in previous work [43], but will in short be explained here. The Interface diagram is modelled by means of blocks and lines in a graphical modelling software tool, MS Visio. The tool is suited for object oriented modelling and structuring different perspectives of architectures in layers. The Interface diagram is normally printed on large A0 blueprints in order to get the overview of the architecture it represents, and is used as a boundary object between different developers (see section *Research methodology*). Figure 5 is a symbolic representation of an Interface diagram. In order for the reader to quickly understand the structure of the product, it is suitable to model the diagram so the layout is established as a cross-section of the actual product. In that way, the physical layout of the product can be recognised in the diagram. The diagram is read by following the interfaces. For products that process or transform objects, this gives a logical reading direction; for other products it is up to the reader to find a suitable flow in the model. The main elements of the diagram formalism are functional objects denoted Key components. The purpose of the Key components is to decompose the systems and modules into smaller building blocks. Key components can have different characteristics and are thus modelled in different ways. Each Key component belongs to one or more product system. To avoid confusion, systems and their interaction must be clearly defined. This is done by choosing the relevant interaction as a basis for determining the system boundary. There is no fixed list of systems to be included in the development. Systems can be modelled and thereby control important properties of the final product. Modules are modelled by arranging Key components inside boxes with a thick black boundary and rounded corners. All modules are assemblies of physically joined components forming one bill of material (with possible multiple levels). Modules can contain smaller modules, but they do not overlap as it is clearly defined to which module any element in a product belongs.

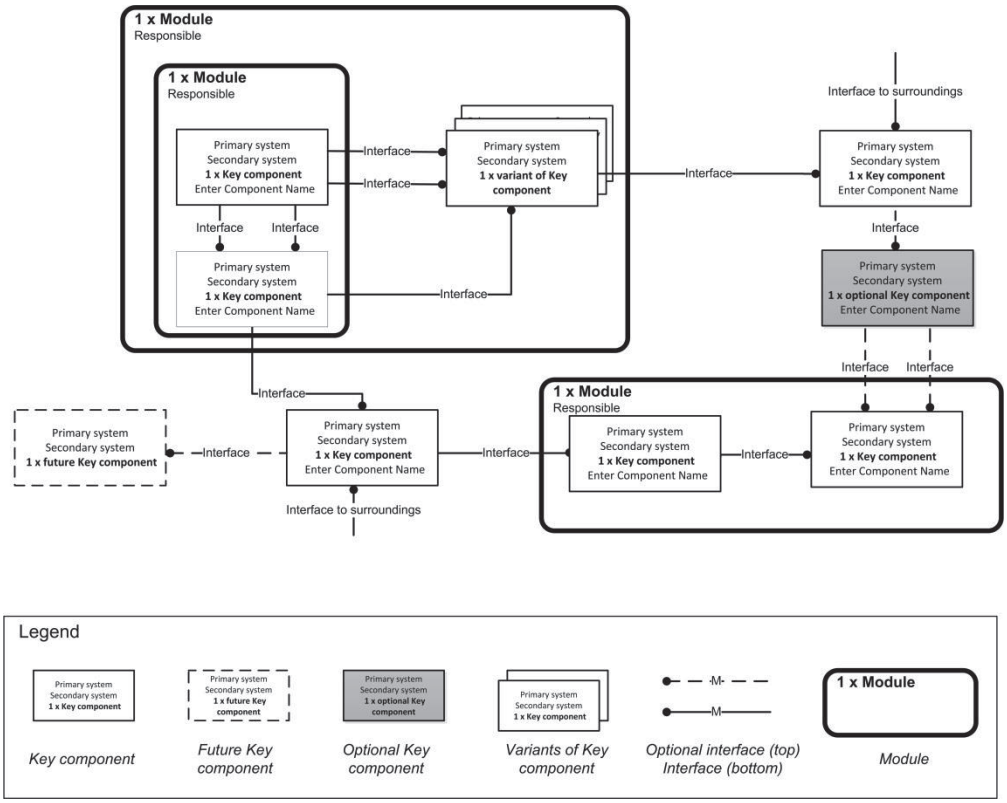


FIGURE 5. SYMBOLIC REPRESENTATION OF A GENERIC INTERFACE DIAGRAM [44]

The structure of the Interface diagram appears as the relations, i.e. interfaces are added to the diagram. The interfaces between Key components are drawn with lines that represent a relation. An interface among two Key components represents a physical relation, e.g. physical connection, energy transportation, information flow or flow of material. The purpose of working with interfaces is to

ensure responsibility for the components' interaction and to ensure that components are interchangeable, when relevant. An interface between two Key components holds a definition on which one is the master and which one is the slave of the interface. This enables responsibility for a Key component to monitor whether he or she owns the right to change or modify the related interface. There exist a number of interface classes and the list can be extended to support the context of a product. Examples of interface classes are: mechanical, spatial, cooling air, cooling liquid, electrical measurement, electrical signal, electric grid etc. The dot at the end of the line indicates the responsibility of the interface. Optional interfaces can be modelled on the diagram to show affected relations between entities or systems if the interface is established. The UML (Unified Modelling Language) class diagram notation technique has been utilised for conceptual defining the structures of entities and their relations modelled in the Interface diagram (see Figure 6).

5.2 Setup platform in PLM system

In collaboration with the vendor, the PLM system has been prepared to handle multiple structure views. The primary task was to set up the data model. The object types and their relations are illustrated in Figure 6 which is a UML class diagram. A dotted line in the figure illustrates the split between *Upper structures* defined in the Interface diagram, and *Lower structures*— that is all other data linked to objects in the Upper structures. The difference will be explained in the section of linking data to Upper structures. The objects have been implemented as soft types of the type *WTPart*. For the interfaces, each classification of the interface has been implemented separately. Furthermore, a number of attributes were added to each object to contain non-structural data i.e. ownership, optionality, variance etc. A new product folder and sub folders for each object type were created to hold all objects and ease managing access rights.

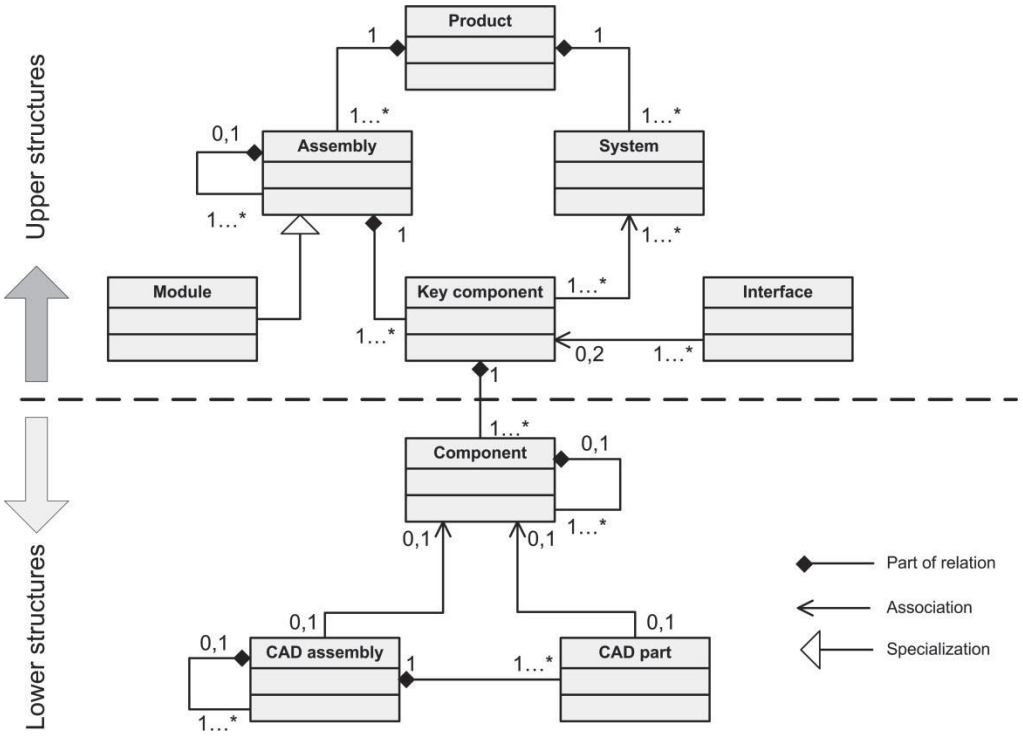


FIGURE 6. UML CLASS STRUCTURE/DIAGRAM OF DATA STRUCTURE IN PLM SYSTEM

The objects have been carefully defined to provide the functionality of multiple structure views, yet able to handle the interdependencies as: systems split by modules, modules containing elements belonging to different systems, interfaces between different system elements etc. The PLM objects are listed and described below.

Product: The product level deals with the complete product family, e.g. the whole product system. A product consists of assemblies and systems.

System: The system elements are grouped and labelled by their functional identity, related lifecycle or could be aligned with an organisational structure, and applied for articulating a structure of systems. A system is seen as a means of a product, which realises its function and carries its properties. The relations between system elements are constituted by part interfaces which deliver active effects (input/output) based upon physical, chemical or biological phenomena. System elements can be differentiated substructures localised spatially in the larger product structure. The purpose of system development is to support functionality in components that are spread across multiple modules.

Assembly: A product can be decomposed into physical assemblies. An assembly contains physically joined Key components forming one bill of material (with possible multiple levels). An assembly can consist of other assemblies, modules and Key components. In contrast for systems, assemblies cannot overlap, meaning that any given Key component, assembly or module can only be a child of one other object. It is worth noticing that components cannot be assigned to assemblies. Likewise, systems cannot be associated with assemblies.

Module: A module is a special type of assembly, following the same rules as just described. The difference between an assembly and a module is the purpose of a module regarding changeability, serviceability and reuse. The module must have well-defined interfaces i.e. assembly relations. The system structure sets the requirements for the module structure, but the module structure and the module functionalities are also determined by materialisation and assembly arrangement.

Key component: Key components are the smallest entities known from the Interface diagram. The purpose of the Key components is to decompose the systems and modules into smaller building blocks. The Key component is the constitutional element of assemblies and modules and it consists of components, CAD-assemblies and CAD-parts. Interfaces are assigned to Key components, and Key components only. Each Key component is assigned one primary system and optionally one or more secondary systems.

Interface: An interface among two Key components represents a physical relation, e.g. physical connection, energy transportation, information flow or flow of material. The purpose of working with interfaces is to ensure responsibility for the components' interaction and to ensure that components are interchangeable, when relevant. An interface connects to two Key components. One of the Key components is the master and the other is the slave. The master Key component has the ownership of the interface, i.e. specification, design, documentation, cost, etc. Interfaces can be of different kinds (e.g. cooling, internal power, etc.) and Key components can have multiple interfaces attached to it.

CAD assembly: The CAD-assembly is the assembly defined in a CAD-system. The CAD-assembly is the constitutional element of Key components and components. It can consist of CAD-assemblies, CAD-parts or components (with no CAD data).

CAD part: The CAD parts are the parts defined in a CAD-system. CAD-parts can belong to a CAD-assembly. The CAD-parts are the constitutional elements of CAD-assemblies and components.

Component: Components are the smallest elements considered in the development process. The objective of the component level is to develop components that suit the requirements of the systems and modules that they support or are part of. Components can consist of other components, CAD assemblies and CAD parts. Components are the constitutional elements of the Key components.

Four structures are needed to accommodate the product structures of a product family. The four structures are implemented as view variants of the same top-level object. The structure views are described below.

- **Design structure:** This view contains the complete CAD assembly structure. This structure is driven from the CAD application and is not altered when building up the other structure views. The CAD objects from the Design structure are linked to the Components

- **Module structure**: This view contains all Modules, Assemblies, Key Components and Components. This structure is showing a 100% BOM, meaning every component is shown exactly once.
- **System structure:** This view contains all Systems, Key Components and Components. This structure is overloaded, because as Key components can be associated with multiple systems, they will exist multiple times in the structure. If a filter is applied and only primary system relations are shown, it will not be an overloaded BOM.
- **Interface structure:** This view contains all interface classes, Interfaces, Key Components and Components. The structure is overloaded, as one Key component can have multiple interfaces and therefore will show up multiple times in the structure. In the relation between the interface and the two Key components, an attribute shows which one is the master and slave, respectively.

The Design structure is made manually by adding the top CAD assembly to the design view of the product node. The other three structures are created automatically, based on input from the Interface diagram, as described in the following section.

5.3 Upload Interface diagram to PLM system

The process of getting all the information from the Interface diagram and into the PLM system is basically divided into two steps: exporting from Interface diagram and importing into PLM. An interchange format based on XML was defined to contain all information from the Interface diagram and was used to contain the data between export and import (see figure 7).

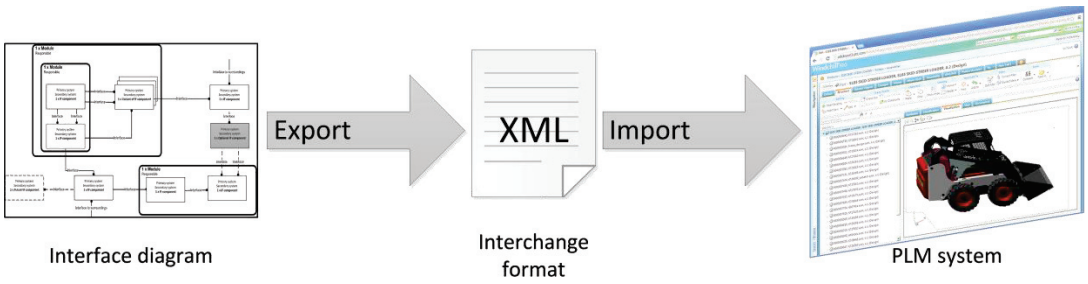


FIGURE 7. ILLUSTRATION OF DATA TRANSFER FROM INTERFACE DIAGRAM TO PLM SYSTEM

5.3.1 Interchange format

The interchange format was designed to contain all information available in the Interface diagram and is a solid way to reach the attributes and structures from other applications, including a PLM system. The format is not limited to getting Interface diagram data into PLM, but is currently also used for various reporting, structure-compare tools, structure-modelling tools etc. The format reflects the module structure and is hence hierarchical, human readable and contains no redundant data. This is possible as the module viewpoint is a 100% structure where all objects exist exactly once — opposite to the systems viewpoint. If the lower structure (CAD data) was to be included, the hierarchical data structure would likewise contain redundant data, as the same assemblies are used in multiple Key Components. The hierarchical format enables quick visualisation and meaningful reading/editing in a simple text editor. The format is illustrated in figure 8.

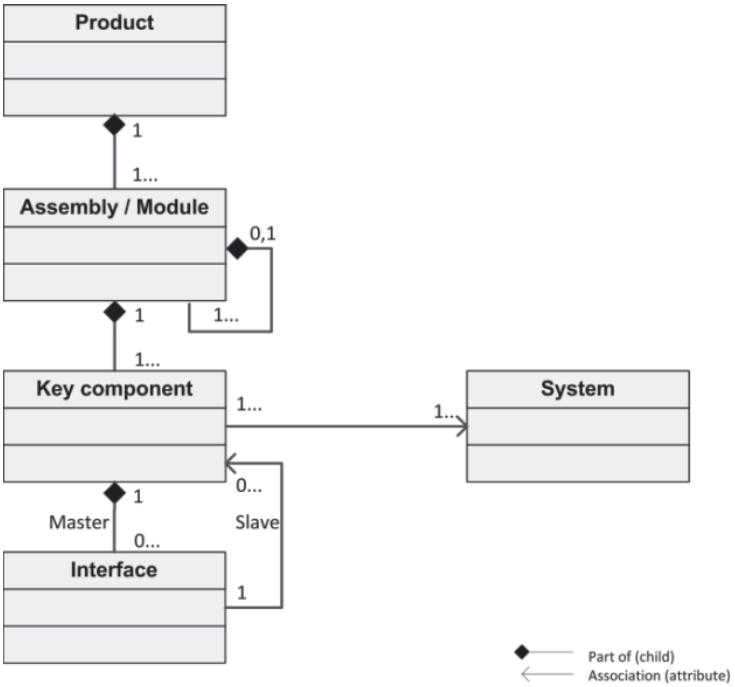


FIGURE 8. HIERARCHICAL FORMAT OF STRUCTURE

The top element is of the type *Product*, and holds the complete product structure. Apart from empty elements containing different attributes, it contains *Assembly* and *Module* elements. These can contain *Assembly*, *Module* and finally *Key Component* elements. *Key Component* elements have an empty element for holding system relations as a comma-separated string — the first being the primary system. *Key Component* elements contain the *Interface* elements for which they are the master. The *Interface* elements have an empty element for holding the ID of the slave *Key Component*. For all elements it is true that business attributes, like name and responsibility, are implemented as empty XML elements and technical attributes, like ID and source application, are implemented as XML attributes.

5.3.2 Export Interface diagram to interchange format (XML)

Visio has native support for Visual Basic for Applications (VBA), which is a scripting language supporting the Visio object model, meaning it has built-in functions for working directly on the objects defined in the Interface diagram. The export function was implemented as a macro in VBA in an independent Microsoft Visio document. The macro starts by locating the Visio application instance and the Interface diagram document and sheet. If successful, it loops through all objects and stores them in an array. If some objects already exist in the PLM system, a file with IDs can be loaded and objects are populated with the IDs. Also, Excel sheets containing BOMs of Key Components can be loaded and added to the structure. The parent/child relationship between objects is defined visually: an object surrounding another object is the parent of that object. This relationship is not explicit to Visio, thus logic is needed to identify the relationship. This is done based on the position and size of each object and is populated by the array holding all the objects. The order of objects sharing the same parent is likewise not explicit to Visio. Some PLM systems order objects alphabetically, in which case the order is trivial; but for systems supporting a custom order, the order is kept as close to the Interface diagram as possible. Interface diagrams are often structured with a goal to match the physical product as much as possible to support people in getting overview and locating objects, and that overview we want to maintain in a later visualisation of the structure in e.g. a PLM system. This is achieved by ordering objects based on their position, the first being the uppermost left object, implemented as the sum of the x and y components in ascending order.

5.3.3 Import interchange format (XML) to PLM system

The utilised PLM system offers multiple methods for automation. The chosen technology is named Windchill® Info*Engine®, which is a scripting language that works directly on the implemented data model. Consultants from PTC® supported this activity.

The script runs in three steps: 1) transform interchange format, 2) create new objects and 3) create new links. Firstly, the script traverses the XML document and transforms it into two new data load files; a file containing all the objects that need to be created, including all the object attributes, and a file containing all the links that need to be created, including quantity. These files are in a standard PTC® XML format. Secondly, the script loads the newly created object file and creates all the objects; Modules, Assemblies, Key Components and Interfaces. It also populates the objects with attributes, such as name, responsibility, whether it is optional or has variants. The processing time is in the range of two minutes for an Interface diagram with 400 Key Components. Thirdly, the script loads the link file and creates all the relations between objects. A part of this process is to locate the newly-created objects and match them with the IDs from the Interface diagram.

When the script is done, objects need to be manually linked to the top-level object. All system objects need to be linked to the system view variant of the top-level object. All top-level module and assembly objects need to be linked to the module view variant of the top-level object. All interface class objects need to be linked to the interface view variant of the top-level object. When this is done, all the structures are complete. This manual process is reasonably fast but could probably be automated in the future. The script was developed to only enable a one-time load from the Interface diagram into the PLM system, but later experience shows that quite long into the design process, managing the upper product structures is done most efficiently on the Interface diagram, thus a method for continuous alignment of the structures in the PLM system is needed.

5.4 Link data to upper structures

Instead of being an environment with focus mainly on the administration of files generated from specific tools like CAD, the PLM system now contains a master product model. The system contains the comprehensive representation of the architecture for the product family by means of the modelled product structures from the Interface diagram (System, Module, and Interface). The structures in the PLM system are named *Upper structures*, to underline that they constitute the master product model. The structures, representing different perspectives of the product family, can be handled individually in the PLM system without the need for redundant data. The object class Key component is present in all three structures, and the objects can be associated with both geometric data (CAD models) and non-geometric product data characteristics and properties (described in documents and by attributes). People responsible for designing systems and modules are allowed to enrich, create, manipulate, and/or delete data belonging to their specific designs, and every member on the project is able to view information from all elements of the product model.

5.5 Update and maintain Interface diagram and data in PLM system

As the Interface diagram is the master model of the product family and changes are incorporated directly onto a blueprint of it at team-meetings, changes have to be incorporated into the PLM system in order for data structures to reflect the present status. We created a simple software tool that compares the product structures from the Interface diagram model with the respective ones in the PLM system. The tool reports any differences between the structures, and thereby enables manual update of the product structures in the PLM system i.e. creating, deleting, or moving objects. Possible documentation already linked to objects that have to be deleted, has to be handled individually; with the purpose of determining if it can be deleted or has to be reassigned to another object.

6. Case

6.1 Situation

The approach was developed and applied in a large manufacturing company serving the global energy industry with energy generation products. The case company is anonymised for competitive reasons and in order to be able to report more interesting details than a public case allows for.

In the wake of the global financial crisis the customers of the company have experienced financial problems with financing their energy solutions. This has led to price competition in the market, while the company and competitors fight for the limited orders. The external pressure on delivery time has increased, which again has put internal pressure on bringing down development time. In order to accommodate these challenges, the company has focused on a strategy of modular product family development instead of single product development. The modular design strategy demands parallelism in design activities and collaboration between a diversity of disciplines in companies, which again involves supporting computer-based tools for enhancing interaction, design management and communication. The deployment of a PLM tool has been seen as an important facilitator for achieving success with the modular design strategy.

6.2 Description of the approach in use

The purpose of the initial interaction with the company was to support its initiative in architecture-based product development and to support the implementation of the PLM system *PTC Windchill PDMLink*. The outcome of the research was prescriptive in nature i.e. descriptions of the intended support; what it is and how it works.

6.2.1 Visual architecture representation

The first step of the approach was to model an Interface diagram of the new product family. The Interface diagram is a dynamic tool which is updated and refined during the beginning-of-life phases for a product family i.e. concept, design, and realisation. The technique for modelling the Interface diagram was based on interviewing the persons with the insight to model systems in their totality. No single person in the company had the required insight to draw the diagram. Therefore several domain experts had to be involved in giving input to the diagram. Experience from the case company showed that it took a number of months to create a diagram with the required detail to represent the product family. Certain scepticism was expressed among large parts of the development organisation. The scepticism was related to the expected extra workload to update a visual representation of the product architecture. However, the opinion turned when the tool quickly enabled designers, module owners and other stakeholders to get an overview of the design status and to evaluate consequences for redesign and upgrade activities. The designers could identify affected parts and modules and who to contact by looking at the Interface diagram. The sequence in the creation of the Interface diagram was to begin with a functional modelling technique (Systems viewpoint in the diagram) and then to create a superimposed modular structure (Module viewpoint in the diagram). Both structures were afterwards developed and maintained in parallel. The diagram provided all project members with a holistic overview of the product system's present status, which enhanced the parallelism among system development activities. The interface diagram enabled a proactive approach for developing and evaluating alternative modular architectural concepts. This could be achieved because modules could be evaluated when the freedom to design was high and the product structures were still to be fixed.

6.2.2 PLM capabilities

The next step of preparing the PLM system to support the upload of the structures from the Interface diagram was carried out in collaboration with the software vendor. No customisation of the PLM system was needed to handle import of the interchange format or represent multiple structures. Export of the modelled structures in the Interface diagram was created outside the PLM environment. Because the PLM system was a completely standard system, there was no extra work for the vendor in supporting use in the implementation phase.

When the Upper structures from the Interface diagram were imported to the system, the actual linking of data was put into operation. As stated earlier, the main purpose of the Interface diagram is to support the design by controlling interfaces. The interfaces were visually documented within the Interface diagram and further described in detail in the PLM system's interface structure. The approach of using the Interface diagram focuses on ensuring compatibility of interfaces during the design phase. Interface descriptions are comparable to *Interface Control Documents* (ICD's) [30]. When a system had complex development and modification processes, agreements between different

working teams on interfaces could not easily be achieved. Therefore, interface descriptions were developed by the teams that establish the requirements to which the interfaces should be designed and developed. These descriptions are not proposed to show the detailed data that would normally appear on design drawings. Instead, they show parameters, characteristics, or configurations that are necessary to ensure that the produced system will operate as desired. The purpose of using interface descriptions is to formalise the description of the connectivity between two systems. The typical information that is described in an interface description is document purpose, the description of the interface, relevant block diagrams, interface functions, their performance, their configuration, and essential features.

In the company, the PLM system was now representing a consistent product model containing multiple structural descriptions, and the previous master role of the CAD-system in the development project changed significantly. The CAD-system now acted as a sub-system of the PLM system, which administrated geometrical descriptions of the products. It offered direct access to parts of the geometry by means of standardised procedures that could be addressed and utilised by the PLM system. The different structures of the product model point to the geometry (assemblies or parts) while CAD-data is linked to objects in the Upper structures (Key components). In that way CAD-models could be monitored differently, without having redundant models in the CAD-system. This is illustrated in figure 9. Different products of the family could be configured by the PLM system, or at single product level representing a system, module, or interface structure. That made it possible for teams responsible for different designs in different life cycle phases to have simultaneous access to the current state of the CAD design— and therefore be able to identify conflicts in the design of systems and modules.

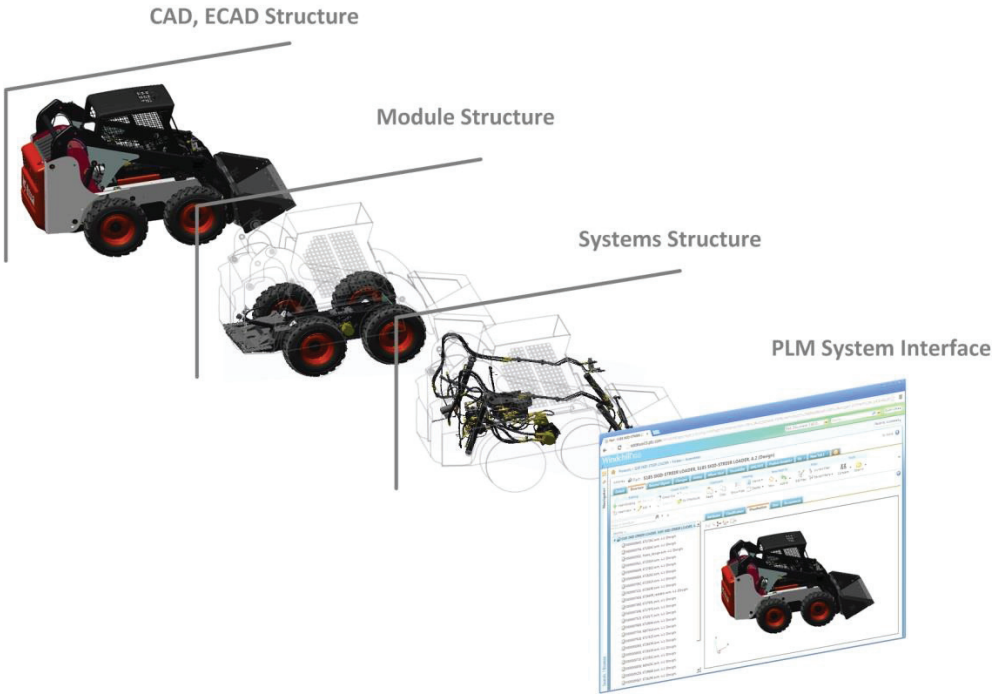


FIGURE 9. VISUAL CAPABILITIES FOR SEEING INTERACTIONS BETWEEN SYSTEMS AND MODULAR STRUCTURES

The PLM system provided continuous computer support because there was no fundamental difference between ‘early’ and ‘late’ phases. The integration of non-geometric characteristics and properties in the system allows for a representation of requirements from the beginning; that is, at a time when perhaps no characteristics are known at all. The three main viewpoint with their characteristics are:

- Systems viewpoint: Systems (e.g. cooling or hydraulic) are defined in a structure representing the bill of material seen from a systems point of view, which can, for example, enable 3D visualisation of the systems and automated reporting of the cost and weight of each system.

- Module viewpoint: Modules are defined in a structure representing the bill of material seen from a module point of view, which enables 3D visualisation and automated reporting of factors such as the cost and weight of each module.
- Interface viewpoint: Interfaces are clearly defined in a structure representing different interface types, and it unambiguously defines which components and modules are related to each interface, and which system holds the responsibility.

For the company, the approach expanded the capabilities of PLM beyond the handling of mainly structural data and information i.e. characteristics and dependencies between them. It was able to support the control and management of the design process itself. The product model in the PLM system evolved during the project, and the processes were managed by means of already built-in work flow management capabilities. A part of the approach was to develop reporting capabilities (see figure 10). Reporting functionality was established in the PLM system and was used to monitor design progress on a weekly basis. The numbers of new, modified, and approved parts were used as indicators of the progress of the design. This resolution level enabled early detection of design conflicts e.g. the risk of not being on time with designs. Furthermore, a report on the product cost was generated weekly.

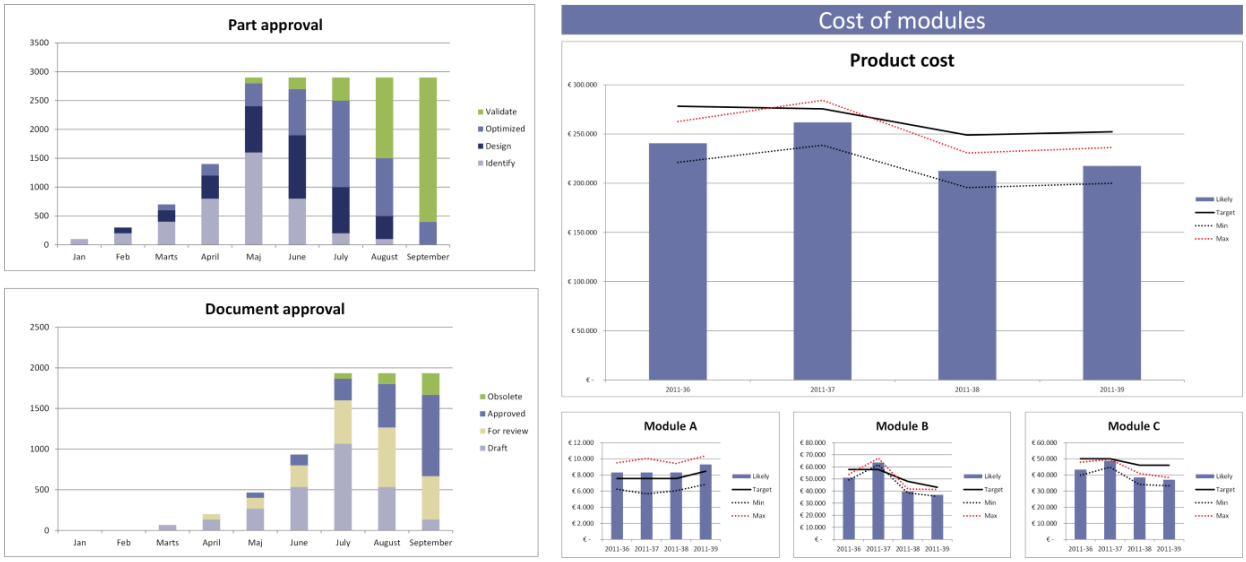


FIGURE 10. EXAMPLES OF REPORTING SHOWING DESIGN PROGRESS AND COST DEVELOPMENT ON A WEEKLY BASIS

Every component in the systems was allocated an initial target cost attribute, and gradually direct cost was added to all components as they were developed or purchased. By means of simple roll-up mechanisms, cost was added from the top product level and down to the smallest components. If a cost element was missing on a component, a target cost stepped in. Target cost numbers were provided by financial controllers in collaboration with the relevant developers and purchasers. Cost reporting became in that way a strong tool for supporting a design according to budget because derivations between budget and actual cost were discovered quickly.

7. Results

The R&D resources saved and the reduction of lead time have not been quantified in a structured way due to two factors. One, the project has not been completed yet. Two, it was difficult to compare this project with others because the project also included development of radical new technologies. The team responsible for maintaining the Interface diagram in the company made the quantitative estimation of a 25% reduction of lead time in early development. The estimate was based on comparing the ongoing project's lead time from G0 to G3 (stage-gates in the company's development model), with the budgeted lead time based on the lead times for past development project in the

company. This reduction was mainly achieved because of an improvement in concurrency of development activities.

However, the implementation of the approach of using a visual architecture description in combination with a PLM system showed some promising qualitative results. By interviewing the members of the management team and the 14 different design teams, the following statements could be reported as results of the implemented approach:

Management

- Design readiness much clearer earlier in the projects.
- More transparent cost deviations.
- Design progress more accurate to measure on a weekly basis.

Designers

- Very little extra work when thinking in architectures for a family compared to single product development.
- All engineering teams have access to the same single source of information.
- Easier to develop and evaluate module concepts in the early design phases.
- More effective reviews.

Today, the company is working with their PLM system as a central tool to support development activities. Also the organisational anchoring of the architecture thinking patterns, methods etc. utilised in this project is also a derived activity as are the implications for portfolio management, road mapping, product and production technology etc.

8. Discussion and reflection

This section discusses the effects of using the computer-based approach and for which types of projects it appears to be advantageous. For the company investigated, the implications of having one master product model instead of having product definitions spread in multiple IT tools are furthermore addressed. The described approach should be seen as contributing to the research area of architecture-based development i.e. the development of product families. The approach is constituted by methods and tools for creating support in product development and in particular in the area of product definitions.

8.1 Case

One case study does not justify wide generalisations or provide a general proof of the suggested benefits. In this research setting, the findings, however, support the formulated hypothesis of creating support when developing modular designs. Even if strict definitions differ, the fundamental principles of modular design are common: break systems into modules, ensure modules can interchange with each other, and provide well-defined interfaces. Results from the case study show that using a visual product architecture model in combination with a PLM system provides the support for developing on a high conceptual level, but still being able to manage information on a very detailed level.

A strength of this approach is that it can be used in companies developing products for a variety of customers and industries. The approach establishes a common product definition of the product family, creating a better basis for developing modules. Making modular designs by decomposing and separating technologies into functional groupings also simplifies the process of the design. If designers had to work with many thousands of parts they would face challenges in handling the complexity of the development tasks. By decomposing a product into building blocks (modules), designers can focus on them separately, and more easily see how these fit together to contribute to the working of the whole.

A weakness to the approach is that it is somewhat resource demanding to implement. Even though design engineers reported that very little extra work was used in the architecture-based development, it is not without effort to decompose a product into functional building blocks. It costs something — mental effort at the very least. So it only pays to divide a product into modules if they are used repeatedly in a number of products, or if they comply with another argument for module creation. Another cost is the implementation of a PLM system, training in using the tool, and in learning the methods constituting the approach of using it. For the case study this has required resources from the company, the software vendor, and external PLM consultants to implement, maintain, and perform training. An actual quantification of the resource demand has not been performed as the project is yet to be completed.

A general observation is offered on the implementation of the approach in the company: In the beginning of the project the approach met some internal scepticism from designers. Creating a common product definition in a PLM system somewhat moves power from the designers in the classical engineering domains to the module managers. They are no longer free to design solutions in a decentralised manner; designs now have to conform to module and interface definitions established in the PLM systems.

8.2 Future work

The modular product definition supported by a PLM system has produced promising results in the case study. The work is however still in its early stages, and more research remains to be done. Three major avenues for future research have been identified.

- First, more experience from applying the approach has to be gathered, in order to estimate the value of using it. The aspects to investigate should cover the approach's usability, its applicability, and its usefulness. Such studies could compare product families within and across industries, and link them to performance measures of interest i.e. development time, cost, revenues, quality, etc.
- Second, the study of the application effects of the approach over time should be done. Successive generations of a product family supported by the PLM approach could be described and measured to investigate the aspects of architecture evolution.
- Third, detailing of the version and revisions to the control of structures should be performed— in order to support engineering change management in a more structured way.

8.3 Conclusion

This paper has presented an approach for using a visual product architecture representation in combination with a PLM system for supporting the activities of developing modular product families. The empirical findings in one case do not justify wide generalisations or provide final proof of the suggested benefits. Each product family is used within a different context and with different requirements. As a consequence, different product family approaches exist, each with its own properties. Many factors — a lot of which are of a non-technical nature — determine whether it will be successful in a particular situation. Many preconditions and prerequisites exist for successful implementation of modular product architecture-based development, which have not been described in detail in this paper. Conditions such as e.g. an established IT infrastructure, organisational ownership, sufficient resources/competences, and high-level anchoring and support of the initiatives, are all seen as conditions that have to be in place in companies. The findings, however, carry the formulated hypothesis of creating support when developing modular designs. The findings show that the approach is applicable for development of complex products e.g. members of a product family, with a large number of components, and to a lesser degree as an approach to choose, when developing simple products with few components and simple technology. The approach has been implemented in an industrial setting and the first results by doing this have been described. The most important results are listed in terms of the experienced effects on the modular design process in the company:

- Clearly defined functional systems; an enabler for managing the design process and the responsibility within each system to meet desired properties of the final product. System development supports functionality in components that are spread across multiple modules.
- Clearly defined modules; an enabler for reuse and other reasons for grouping components in modules.
- Clearly defined interfaces; an enabler for reuse of modules and the general overview of interchangeability of modules. Clear allocation of responsibility for interfaces.
- Accurate cost reporting because of automated reporting based on modules.
- Accurate design progress reporting because of automated reporting based on module status.

The main contribution of this research is the distinction between structural and functional contents of architectures, and the ability to handle this distinction when developing product families. By actively using this distinction it is possible to improve modular design, because modularity is based on both characteristics and properties of products. The implemented approach has shown that the benefits are especially dominant in the related CAD system, to represent CAD models in both a functional and physical grouping. One of the objectives for using the Interface diagram in combination with a PLM system is to enable parallel activities in different life phases. Parallelism in design activities requires carefully decomposing products into discrete modules and specifying interfaces to ensure that the results of the parallel activities go hand in hand.

Last, it is important to note that any products could potentially be modularised and their information could be managed in PLM systems, but to benefit more easily from the advantages of PLM systems, companies must be selective in choosing which products to support with this approach. The strength of this approach is its ability to address complex products with a large number of components, components belonging to different technical domains, and/or components developed by different organisational teams. To apply this approach to products with few components could be characterised as using a far too complex tool supporting the design of a simple product or as "killing butterflies with muskets".

8.4 Acknowledgements

The authors would like to thank the case company for sharing challenges, resources and competences with the research team.

9. References

[1] Meyer, M.H., and Lehnerd, A.P., 1997, "The ¤power of product platforms building value and cost leadership," The Free Press, New York, pp. 267.

[2] Hölttä-Otto, K., and de Weck, O., 2007, "Degree of Modularity in Engineering Systems and Products with Technical and Business Constraints," Concurrent Engineering, **15**(2) pp. 113-125.

[3] Krause, D., and Eilmus, S., 2011, "Methodical Support for the Development of Modular Product Families," The Future of Design Methodology, pp. 35.

[4] Rogers, C., 2012, "1 Platform, 4 Million Cars," Automotive News, **86**(6511).

[5] Stark, J., 2011, "Product Lifecycle Management," Product Lifecycle Management, pp. 1-16.

[6] Bergsjö, D., Malmqvist, J., and Ström, M., 2006, "Architectures for mechatronic product data integration in PLM systems," Proceedings of the 9th International Design Conference DESIGN 2006, Anonymous pp. 1065-1076.

[7] Gecevska, V., Chiabert, P., Anisic, Z., 2010, "Product Lifecycle Management through Innovative and Competitive Business Environment," Journal of Industrial Engineering and Management, **3**(2) pp. 323-336.

[8] Sudarsan, R., Fenves, S. J., Sriram, R. D., 2005, "A Product Information Modeling Framework for Product Lifecycle Management," Computer-Aided Design, **37**(13) pp. 1399-1411.

[9] Sääksvuori, A., and Immonen, A., 2008, "Product lifecycle management," Springer Verlag.

[10] Cantamessa, M., Montagna, F., and Neirotti, P., 2012, "An Empirical Analysis of the PLM Implementation Effects in the Aerospace Industry," Computers in Industry, **63**(3) pp. 243-251.

[11] Chungoora, N., Young, R. I., Gunendran, G., Palmer, C., Usman, A., Anjum, N. A., Cutting-Decelle, A., Harding, J. A., Case, K., 2013, "A Model-Driven Ontology Approach for Manufacturing System Interoperability and Knowledge Sharing," Computers in Industry, **64**(4) pp. 392-401.

[12] Li, S., Chen, J., Yen, D. C., Lin, Y., 2013, "Investigation on Auditing Principles and Rules for PDM/PLM System Implementation," Computers in Industry, **64**(6) pp. 741-753.

[13] Kugathasan, P., and McMahon, C., 2001, "Multiple Viewpoint Models for Automotive Body-in-White Design," International Journal of Production Research, **39**(8) pp. 1689-1705.

[14] Lindemann, U., Maurer, M., and Braun, T., 2009, "Structural complexity management," Springer.

[15] Gunendran, A. G., and Young, R., 2010, "Methods for the Capture of Manufacture Best Practice in Product Lifecycle Management," International Journal of Production Research, **48**(20) pp. 5885-5904.

[16] Andreasen, M. M., 1980, "
Syntesemetoder på Systemgrundlag – Bidrag Til En Konstruktionsteori (in Danish) " .

[17] Hubka, V., and Eder, W. E., 1988, "Theory of Technical Systems: A Total Concept Theory for Engineering Design," Berlin and New York, Springer-Verlag, 1988, 291 p., **1**.

[18] Pine, I.,B.J., 1999, "Mass Customization: The New Frontier in Business Competition (Paperback)," Harvard Business School Press Books, .

[19] Pine, I., Joseph, B., and Victor, B., 1993, "Making Mass Customization Work," Harvard Business Review, **71**(5) pp. 108-117.

[20] Baldwin, C.Y., and Clark, K.B., 2000, "Design rules, Volume 1: The power of modularity," mIt Press, .

[21] Ulrich, K., and Tung, K., 1991, "Fundamentals of Product Modularity," American Society of Mechanical Engineers, Design Engineering Division (Publication) DE, **39**pp. 73-79.

[22] Ulrich, K., 1995, "The Role of Product Architecture in the Manufacturing Firm," Research Policy, **24**(3) pp. 419-440.

[23] Ulrich, K.T., Eppinger, S.D., and Goyal, A., 2011, "Product design and development," Irwin/McGraw-Hill, .

[24] Pahl, G., Beitz, W., and Wallace, K., 1996, "Engineering design A systematic approach," Springer, London, pp. 544.

[25] Fellini, R., 2005, "Platform Selection Under Performance Bounds in Optimal Design of Product Families," Journal of Mechanical Design, **127**(4) pp. 524-535.

[26] Erixon, G., 1998, MFD–Modular Function Deployment, A Systematic Method and Procedure for Company Supportive Product Modularisation, .

[27] Anonymous 2012, "Swedish Standards Institute: Systems and Software Engineering - Architecture Description (ISO/I EC/IEEE 42010:2011, IDT)," .

[28] Pimmler, T. U., and Eppinger, S. D., 1994, "Integration Analysis of Product Decompositions," American Society of Mechanical Engineers, Design Engineering Division (Publication) DE, **68**pp. 343-351.

[29] Sundgren, N., 2003, "Introducing Interface Management in New Product Family Development," Journal of Product Innovation Management, **16**(1) pp. 40-51.

[30] Rahmani, K., and Thomson, V., 2011, "Managing Subsystem Interfaces of Complex Products," International Journal of Product Lifecycle Management, **5**(1) pp. 73-83.

[31] Pedersen, R., 2009, "Product Platform Modeling," Technical University of Denmark, Department of,Mechanical Engineering; DTU, Lyngby, .

[32] Gershenson, J. K., Prasad, G. J., and Zhang, Y., 2004, "Product Modularity: Measures and Design Methods," Journal of Engineering Design, **15**(1) pp. 33-51.

[33] Jiao, J., and Tseng, M. M., 2000, "Fundamentals of Product Family Architecture," Integrated Manufacturing Systems, **11**(7) pp. 469-483.

[34] Krause, D., Eilmus, S., and Jonas, H., 2013, "Smart Product Engineering,"Springer, pp. 543-552.

[35] Simpson, T. W., Maier, J. R., and Mistree, F., 2001, "Product Platform Design: Method and Application," Research in Engineering Design - Theory, Applications, and Concurrent Engineering, **13**(1) pp. 2-22.

[36] Tiihonen, J., and Soininen, T., 1997, "Product Configurators: Information System Support for Configurable Products," Helsinki University of Technology, .

[37] Abramovici, M., 2007, "Future Trends in Product Lifecycle Management (PLM)," The Future of Product Development, pp. 665-674.

[38] Weber, C., Werner, H., and Deubel, T., 2003, "A Different View on Product Data Management/Product Life-Cycle Management and its Future Potentials," Journal of Engineering Design, **14**(4) pp. 447-464.

[39] Sudarsan, R., Fenves, S. J., Sriram, R. D., 2005, "A Product Information Modeling Framework for Product Lifecycle Management," Computer-Aided Design, **37**(13) pp. 1399-1411.

[40] Latour, B., 1986, "Visualization and Cognition," Knowledge and Society, **6**pp. 1-40.

[41] Hansen, C. L., Hvam, L., and Mortensen, N. H., 2011, "Proactive modeling of product and production architectures," Proceedings of ICED 11, Copenhagen, Denmark, Anonymous .

[42] Harlou, U., 2006, "Developing product families based on architectures contribution to a theory of product families," Department of Mechanical Engineering, Technical University of Denmark, Lyngby, pp. 173 sider.

[43] Bruun, H. P. L., and Mortensen, N. H., 2012, "Modelling and using Product Architectures in Mechatronic Product Development," Norddesign 2012, Aalborg, Denmark, .

[44] Bruun, H. P. L., and Mortensen, N. H., 2012, "Visual Product Architecture Modelling for Structuring Data in a PLM System," PLM12, Montreal, Canada.

Paper E

Bruun, H.P.L., Hauksdóttir, D., Harlou, U., Mortensen, N.H.,
Mapping requirements to a product architecture supported by a PLM system,
Conference proceedings of Design Conference 2014, Dubrovnik, Croatia.

INTERNATIONAL DESIGN CONFERENCE - DESIGN 2014
Dubrovnik - Croatia, May 19 - 22, 2014.

MAPPING REQUIREMENTS TO A PRODUCT ARCHITECTURE SUPPORTED BY A PLM SYSTEM

H.P.L. Bruun, D. Hauksdóttir, U. Harlou, N.H. Mortensen

Keywords: Requirements, Product architecture, PLM system support

1. Introduction

Engineering design in the modern global and competitive business environment is under ever increasing pressure to perform better in terms of productivity, quality and high value output customised for specific market segments. One approach to improve engineering design performance is through reusing previous knowledge. If a product domain is mature it is likely that a new product will have considerable overlap with previous product variants. This creates the opportunity to harvest benefits from creating a high-quality reusable knowledge that can be used between development projects. The practice of exploiting commonality to take advantage of economies of scale and scope, while targeting a variety of market applications, is generally referred to as Product Families Engineering (PFE) or Product Line Engineering (PLE). A product family is a group of related products that are derived from a common set of design elements to satisfy a variety of market applications where the common elements constitute the product platform [Meyer and Lehnerd 1997]. There are many advantages to PFE most of which stem from increased commonality among the set of products. All work elements used in the product development are possible elements for reuse. It can be assumed that reusing knowledge at previous development process steps will subsequently lead to reuse in the following process steps. Therefore ideally, identifying and reusing front-end knowledge such as market information and customer needs should be the optimal origin of information reuse. . Requirements management (RM) emphasises on capturing what the product should do without describing how it should do it. This emphasis on the product domain improves understanding a planned product and encourages the discovery of the true stakeholder needs, therefore preventing premature solution selection. One of the main goals of RM is to ensure good understanding between the development organization and stakeholders. Inadequate RM is generally considered one of the major causes for product failures in product development project [Goldin et al. 2010; Jones 1995; McConnell 1996]. One of the main reasons for companies facing a challenge in managing requirements is that requirements are defined in the beginning of a development project when desirable properties are abstract. When the product system is synthesised, requirements often have to be detailed and modified in order to reflect the solution domain. This interplay between problem and solution domains is at the heart of any engineering design activity [Chen et al. 2013]. Successful development of a platform and deployment of a product family requires a successful transformation of information between the disciplines.

The purpose of this paper is to support the interplay between the problem (requirement) domain and the solution (architecture) domain of a product family. An approach for mapping requirements to architectural views of a product family is presented. In the approach 5 views are suggested, 2 requirement model views, a customer view and a functional view, and 3 views describing the product architecture; functional system, physical module and interface view. The customer requirements are mapped to the product architecture on a conceptual level. Based on the architecture design further requirements are identified. The architectural models therefore contain design information as well as

requirements, enabling the designers to specify requirements originating from architectural decisions simultaneously as those decisions are taken and continue to work with and elaborate on requirements thought the design process. The paper furthermore describes how the suggested approach can be accomplished by using a Product Lifecycle Management (PLM) tool.

2. Why is it beneficial to map requirements to the product architecture?

An early understanding of requirements and choice of architecture is key to managing product development for complex systems and projects. Typically the effectiveness of a solution is determined with respect to a defined problem, however, the nature of the problem and its scope could depend on what solutions already exists or what solutions are plausible and cost-effective [Chen et al. 2013]. In reality, the architecture can constrain designers from meeting particular requirements, and the choice of requirements can influence the architecture that designers select or develop. Recent models suggest that instead of doing requirements only at the early phases, requirements definition and design are interactive activities, handled simultaneously though the development life-cycle [Nuseibeh 2001]. The key objectives for mapping requirements to elements of the product architecture are:

- To realise how the architecture constrains and enables requirements.
- To better evaluate the cost of implementing new requirements.
- To identify and rationalise reuse of components for a new product variant.
- To enable modular design:
    - By enabling an optimisation of modularity in the product architecture in regards to requirements.
    - By enabling an implementation of new or changed requirements by only redesigning the affected systems/modules.
- To enable tests of modules and functional systems and therefore, identifying incompliances earlier.
- To identify constraints that the architecture and selected solutions compose on the design.
- To support focus on requirements compliance during design, by enabling designers to have a overview of the requirement affecting the module they work on.
- Better traceability for how customer requirements are realised in design leading to a better understanding of how customer value is established.

Requirements are information intense. For complex systems the requirement specification can contain hundreds of requirements and be presented in documents of considerable length. It can therefore be difficult to get a sufficient overview. It is important to consider the receiver and user of the requirement specification. If it is not possible to provide a view of the requirement relevant for a specific part of the system, it will be difficult to sort out and identify the relevant information. This decreases the value of the requirement specification and threatens product quality. A method for systematically addressing the right requirements during design is needed.
In a modern engineering environment, both requirements and architecture are important aspects of modelling the system, and architecture becomes a critical aspect of describing the problem. Architecture serves two roles; it defines and it constrains [McGovern et al. 2004]. It defines the system components, their interfaces and interactions. Existing systems, infrastructure and capabilities provide a rich base from which to create new capabilities but also introduce a set of complex constraints. As systems become more technically complex, the architecture will have a more dominant role in defining the problem. Architecture is no longer solely a downstream development activity; it is also an upfront activity, a key consideration in defining the problem [Cole 2006]. While specifying a solution, designers are also constraining, and therefore imposing additional requirements. Architecture at one level must support requirements at that level, but since generates constraint to the solution space, it drives requirements at lower levels [Cole 2006]. Researchers and practitioners are struggling to develop methods that allow rapid development in a competitive market, combined with the improved analysis and planning that is necessary to produce high-quality systems within tight time and budget

constraints. A more robust and realistic development process allows both requirements engineers and system architects to work concurrently and iteratively to describe the intended system [Nuseibeh 2001].

3. State of the art

This section briefly reports significant contributions in the research areas of representing product architectures, requirements management and the transformation between the two.

3.1. Representation of products architectures

Product architecture is the scheme by which the functions of the product is mapped towards the physical parts, thus defining the product architecture as the arrangement of functional elements, the mapping from functional elements to physical parts and the specification of interfaces among these [Ulrich 95]. The conceptualisation of a product system expressed in a product architecture model assists the understanding of the product system's essence and key properties pertaining to its behaviour, composition and evolution [IEEE 2011]. Architecture descriptions are used by the members that create, utilise and manage product systems to improve communication and co-operation; enabling them to work in an integrated, coherent fashion. The formulation of an architecture model involves considerations from several perspectives generating different views, e.g. formulated in the PFMP framework [Harlou 2006]. The customer view describes the quality properties that are valuable and meaningful to the customers. The technical or engineering view reveals how functionality is provided and what technology has been applied. The physical view describes how the product design is realised by the physical components. In addition, to enable access supply chain considerations, a supplementary representation of possible production layouts, a production view, is needed [Mortensen et al. 2008]. In order to incorporate the different views, generic modelling notations have to be applied that enable the representation of commonality, alternative variety, and ranges [Jiao and Tseng 2000; Harlou 2006].

3.2. Requirement management and product architecture

A requirement specifies something that the product must do or a quality that the product must have [Robersson and Robertsson 2012]. Product quality is a degree of excellence regarding the ability of a system to provide a desired combination of quality characteristics [Niemelä and Immonen 2006]. Poor product definition can result in specification creep, time-to-market delays and insufficient product quality. RM proposes methods to cope with the requirements at the early phases of the development life-cycle and presents concepts of identifying, collecting and allocating "*system functions, attributes, interfaces, and verification methods that a system must meet including customer, derived (internal), and specialty engineering needs*" [Stevens and Martin 1995, p.11]. It can be said that in modern approaches RM issues are engineered, involving tools, modelling, database design, data handling etc. [Alexander and Beus-Dukic 2009]. The twin peaks model, see Figure 1, present an emphasis on the equal status of requirements and architecture.

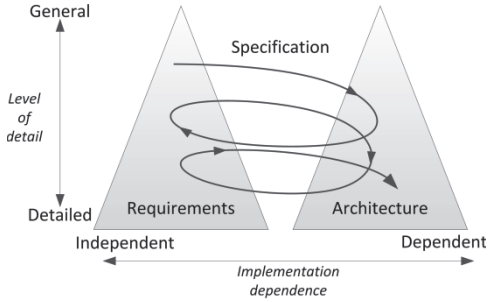


Figure 1: Twin peak model.

The two are intertwined, and it is important to understand how the architecture constrains and enables requirements. At the same time a separation of the problem specification structure and solution specification structure is necessary [Nuseibeh 2001]. Although the twin peaks provides a conceptual

differences and relationship between requirements and design, the *process* of moving between the problem and the solution domain is not as well recognised [Goedicke et al. 1996]. Matrix-based modelling techniques help to classify the relationship between different design elements and therefore support the process of moving from the problem to the solution domain. Through *Quality Function Deployment (QFD)* and *Axiomatic Design (AD)* method designers can use a series of inter-domain matrixes [Malmquist 2002] to transfer customer requirements into specific product attributes, engineering characteristics, possible design solutions and manufacturing activities [Akao 1990; Suh 2001]. Both methods provide guidelines to systematically make design decisions and reuse design elements based on customer requirements with the objective to design customer satisfaction and quality assurance [Jin and Lu 1998].

4. Requirements management and product data management tool support

To implement any kind of a systematic approach to requirements reuse having an appropriate tool support to provide the necessary mechanisms is required [Toval et al. 2008]. RM tools are either specifically intended for managing requirements only, or to support the entire systems engineering process. Analysis of some of the most popular current RM tools has revealed a lack of automated support for some necessary reuse features e.g. for sufficiently managing variability [Toval et al. 2008]. Product Data Management (PDM) is the use of software or other tools to track and control data related to a product domain. PDM technology is intensively used in industry and today its application is mainly focused on particular product lifecycle phases, e.g., development, prototyping or production [Abramovici 2007; Stark 2011; Sääksvuori & Immonen 2008]. The functionalities of enterprise PDM tools have evolved in the last decades and today PDM is in many cases seen as a subset of modern PLM system tools [Sääksvuori & Immonen 2008]. PLM is the extension of PDM towards a comprehensive approach for product-related information and its management within an enterprise. PDM/PLM system tools also holds functionalities for requirement management, but have a broader scope that include data vaults that control access to files, and there are formal, workflow-supported, processes for executing engineering changes. It is, however, well known that one of the most severe limitations of current PDM/PLM systems towards realising such visions is the lack of dedicated functionality for RM [Sääksvuori & Immonen 2008]. PDM/PLM systems handle requirements in documents, which again require additional work when creating a requirement structure of objects [Malmquist 2001].

5. Concept for mapping requirements to architectural views of a product family

The suggested approach for mapping requirements to architectural views of a product family is presented in this section. In the approach 5 structural views are suggested, 2 requirement model views, a customer view and a functional view, and 3 views (encompassed in 1 visual model) describing the product architecture; system, module and interface view. The approach includes using a standard PLM system for operational mapping between requirements and the architectural views of a product system and enabling continues detailing of requirements and product architecture during development. The approach has been developed and tested in a research study. A mobile power loader (Bobcat) is used to illustrate and to exemplify. The steps in the approach for mapping requirements to architectural views of a product family can be seen in figure 2.

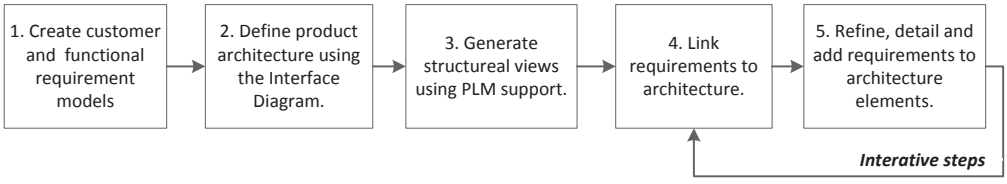


Figure 2 Steps in the approach

The approach suggests two visual models which are utilised for capturing and defining requirements to a product system. The first is the *Customer view* [Harlou 2006] describing the requirements coming

from the customer domain. The second is a *Function view,* e.g. [Pahl et al. 1996], analysing and detailing functional requirements. The reason for having two models for handling requirements is to take into account the different characteristics of static and functional qualities. The customer view presents characteristics and performance levels, and how they differentiate between the product variants while the functional view is meant to analyse and detail the expected actions of the product. The product system's material entities are represented by a visual product architecture model, *The Interface diagram* [Bruun & Mortensen 2012b], supporting the mapping from functional to physical characteristics, decomposition of the product system into manageable design units, and the creation and management of interfaces and interactions between design units. The Interface diagram holds three architectural views on a product family, a functional structured system view, a physical encapsulated module view, and an interface view. When the visual models have been defined they are generated in structural models using PLM support. Links are then inserted to map the requirement elements to the product architecture. Finally, requirements and architecture are continuously detailed and refined in an interactive manner as the product family based design process proceeds.

5.1. Creating customer and functional requirement models

Before identifying any form two requirements models are established. A customer requirement model, presents specific quality attributes that the product must meet and the variability of the product family from the viewpoint of the customer. The functional requirements model describes the product's ability to do something or be used for something, i.e. deliver an active effect.

Creating the customer requirement model

The PFMP customer view is a model used for specifying requirements to product families and ranges of products, highlighting the product differentiation from a customer point of view. The Customer view formalism has its origin from the object-oriented paradigm and system modelling, and was introduced at DTU 1999 [Mortensen 1999]. The formalism is constituted by two types of elements including requirement classes and additional attributes, and two types of structures denoted as part-of-structure and kind-of-structure.

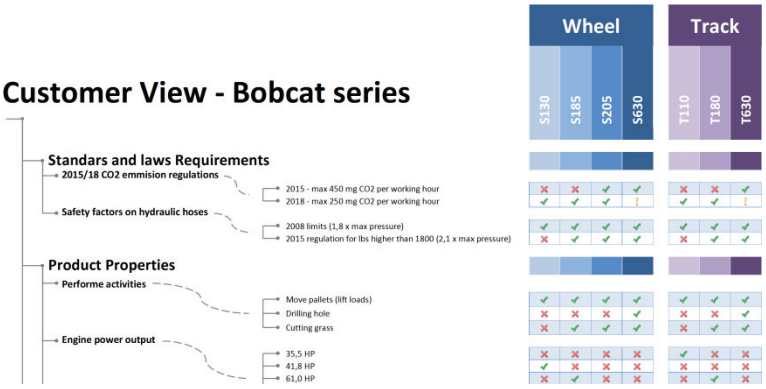


Figure 3 Excerpt from Customer view example of a family of Bobcats

Classes are groups of requirements that share a common denominator. The part-of-structure describes the hierarchical structure of the class of requirements. The kind-of-structure describes the associated variance of the part-of-structure and together this gives the total variants of requirements to the product system. The approach is performed for a family of products, i.e. a family of commercial product variants. The overview of the commercial variants is combined with the customer view. Here the mapping of features and options are done towards these variants. Figure 3 shows an excerpt of a visual overview of the customer view. Examples of requirements in PFMP customer view:

- *Expected life time (5 years, 10 years) => Product 1 and 3 = 5 years, Product 2 = 10 years*
- *Move/Lift pallet application => Product 1 and 2, not product 3*

Creating a functional requirements model

Functional requirements describe an action that the product shall do. To identify the required functions of a system a functional model providing a graphical representation of the transformation of energy,

material or information flow as they pass through the system, can be created. The conceptual functional model should identify the basic individual functions required to accomplish the overall functionality of the product [Hutcheson et. al. 2007]. By analysing the relationships between the inputs and output of the functions, more concrete functional requirements can be identified. At this point we no longer have only a black box view of the product, but still no form specific solutions are included. IDEF0 is an example of a well-known formal functional modelling methodology [www.idef.com]. Causality between functions and means, capable of realising the functions, was first pointed out by Hubka [Hubka & Eder 1988]. In a functional view the functions and means constitute a hierarchy, because any function/means need helping functionalities. The Functions/Means-pattern may support the synthesis activity. Examples of functional requirements:

- *The machine shall vertically lift the material*
- *The machine shall show signals about state vulnerable parts*

5.2. Defining the product architecture using the interface diagram

The Interface Diagram is a visual design tool for decomposing a product system into functional sub-systems, modules, components, and interfaces. The tool has been described in additional work by the authors [Bruun & Mortensen 2012a], and will not be described in detail here. Application of the tool for analysis and synthesis is a proposal for supporting the engineering process when developing complex product systems. One of the objectives for deploying the Interface diagram is that it should provide the designer with an aid to decompose, characterise, and compose product systems. Another equally important objective is to enable a computer-based tool to support this in interplay with the Interface diagram. The Interface diagram puts emphasis on managing technical interfaces between the modelled entities, hence the chosen name of the modelling tool. The tool puts emphasis on handling a product system seen from different viewpoints. The main viewpoint is a sub-systems perspective, i.e. a perspective that deals with the product's main functions or one of its related lifecycles. The main sub-systems can be identified by recognizing groups of related functions in the functional model described in section 5.2. The second viewpoint is a modular viewpoint in which physically joined components are encapsulated into modules. Modules can consist of components belonging to different systems, i.e. developed by different engineering teams. It is therefore necessary to integrate system components into modules.

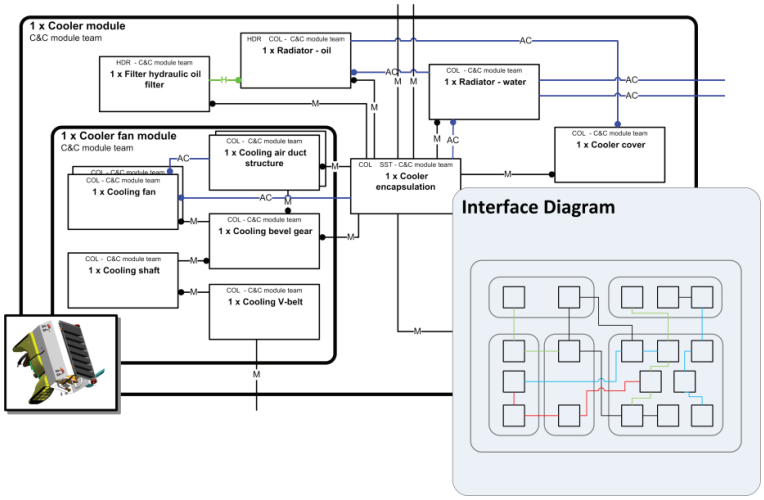


Figure 4 Excerpt from an Interface diagram of a Bobcat by the use of the Interface diagram formalism.

The Interface diagram is modelled by means of blocks and lines in the software program Microsoft Visio. The Interface diagram is printed on large blue prints in order to get the overview of the product system and to allow for stakeholders to write and draw directly on the modelled structures during meetings. Figure 4 illustrates an excerpt of an interface diagram. The main elements of the diagram formalism are objects denoted *Key-components*. The purpose of the Key-components is to decompose the product system into smaller building blocks. Modules are modelled by arranging Key-components

inside boxes with a thick black boundary and rounded corners. The relations between Key-components are drawn with lines which represent an interface or an interaction. Interfaces and interactions are linkages shared among components, modules, and sub-systems. Interfaces between two Key-components represent physical relations, e.g. constituting physical connections. Interactions between two Key-components represent the transfer of material, energy (forces, movement), and information. The interaction may be transferred via an interface, but can also be indirect.

5.3. Generating structural views using PLM support

The visual, analytical models are next transformed into structural views, where the relationships between the architecture elements will be established. In collaboration with the software vendor (*PTC®)* a method for loading the objects from the Interface diagram (Key components and their system, module and interface relationships) to the utilised PLM system (*Windchill PDMLink 10.2*). The process of getting all the information from the Customer view, Function view, and Interface diagram and into the PLM system is basically divided into two steps: exporting from the models and importing into PLM. An interchange format based on XML was defined to contain all information from the Interface diagram and the Customer view and was used to contain the data between export and import. The function view model was created in an Unified Modelling Language (UML) system environment, *Enterprise Architect 10,* and could be exported directly to the PLM system.

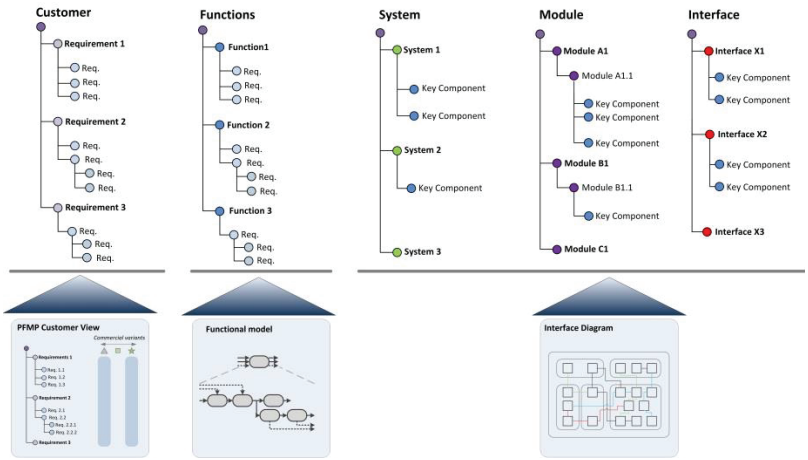


Figure 5 Loading requirements and architectural views into PLM

With both requirements structures and the architecture definition defined in the PLM system, it was possible to start establishing linkages between objects.

5.4. Link requirements to architecture elements

Requirements are mapped to the product architecture on a conceptual level. Relations are established with a standard functionality in the PLM system called MPSE (*Manufacturing Product Structure Explorer*), which works by simple drag and drop operations for creating relations. Examples of requirement mapped to the architectural views, and characteristics of the views, are described in the following sub-sections.

Mapping customer requirements to functional requirements

Although the customer performance requirements and the functional requirements are analysed and presented in different models, it is important to consider and understand that there is a tight coupling between the two. For functionality to provide the expected effects in a sufficient manner it often must meet dependent performance attributes. The performance requirements were therefore mapped to the functional requirements.

- *Operating weight (2610 lbs.) > mapped to > the machine shall lift a load of material*
- *Travel speed (3,5 mph) > mapped to > The machine shall be able to move forward*

The first example illustrates that the machine is required to provide the functionality of lifting a load of material, but the function does not meet its quality target unless the lifting functionality can manage lifting a weight sufficient for the intended operation.

Mapping requirements to sub-systems
The purpose of sub-systems is to support the development of functionality in components that are spread across multiple modules. Systems are often characterised by one or more of the following: Deliver important functionality, e.g. steering, braking, loading etc.; is a complex or new technology, e.g. hydraulics, cooling etc.; alignment with organisational structure; requires the involvement of many different technical disciplines. The requirements identified in the functional model are mapped to the corresponding sub-system. Examples of requirements mapped to sub-systems:

- *Hydraulic pressure (30bar) > mapped to > the machine shall provide hydraulic pressure > mapped to > Hydraulic system and Power System*
- *The machine shall be able to respond to signals from sensors > mapped to > Control and conditioning system*

Mapping requirements to modules
Modules organise and group product components in the best physical arrangement that will optimally support the product lifecycle; some examples can be manufacturability, encapsulation of complexity, upgradability either for current program or aftermarket needs and serviceability. Examples on requirements mapped to modules:

- *Width of bucket (68in) > mapped to > Front equipment module*
- *Seat type (Suspension seat) > mapped to > Cabin module*

Mapping requirements to interfaces
In modularisation it is the interface that ensures decoupling, and thereby enables reuse, sharing, substitution and serviceability. The purpose of working with interfaces is to specify and design any logical or physical relationship required to integrate the boundaries between systems or between systems and their environment. In a module context, an interface is important from a physical assembly perspective. Examples of interface classes are: Mechanical, spatial, cooling air, cooling liquid, electrical measurement, electrical signal, electric grid etc. Examples of requirements mapped to interfaces:

- *Bolt flange (30 BCD, M12x1,5, (8.8)) > mapped to > Interface.Mechanical.245(Hydraulic shaft – hydraulic pump)*
- *Flow (500 l/h) > mapped to > Interface.Hydraulic.456(Hydraulic pipe – Hydraulic manifold)*

5.5. Refine, detail and add requirements to architecture elements

Based on the architecture design further requirements are identified. The elements in the architectural models therefore contain design information as well as requirements, enabling the designers to specify requirements originating from architectural decisions simultaneously as those decisions are taken. As form specific solutions that solve the desired functionality of the system are identified, more detailed, form-specific functional models can be created. These models further refine the functionality of the chosen solution, and should result in an identification of additional requirements and modification of existing requirements to reflect the new functionality and flows. The traceability between the models in the PLM system greatly supports this interactive system design work. When a requirement is associated to a Key component, all structures where the Key component coexists (system, module, and/or interface), can see it. Furthermore functionality for monitoring ‘where used’ can be utilised for seeing *parent-relations* for Key components and their requirements, as well as *children-relations* for systems and modules to requirements.
Requirements are linked from the customer model to the architectural structure of systems and modules, as customer requirements affect how functional effects and structural characteristics are realised, see Figure 6. Requirements from the functional model are linked to the system view, as it is possible to identify relations to the already functional based system view.

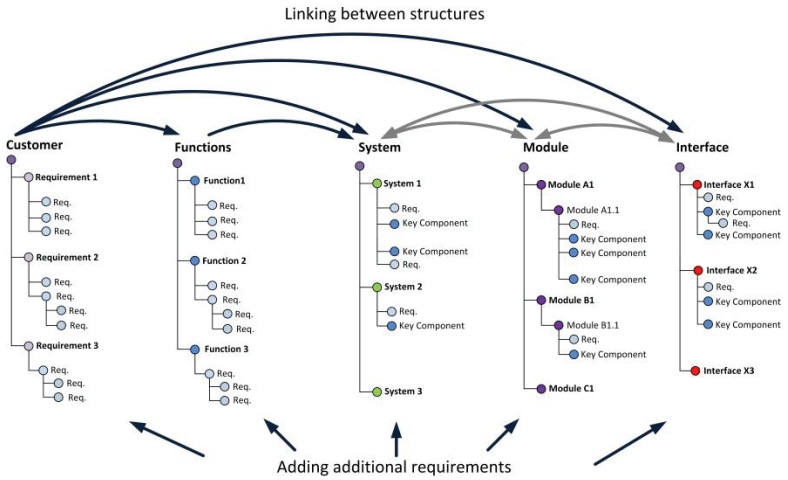


Figure 6 Mapping, detailing, and adding requirements during the development process

In the early phases of NPD, no or few requirements are linked to the interface structure, as it determines as a result of the architecture realisation and composition. Additional requirements between the architectural views are established and can be bidirectional, i.e. point to design elements between two views. The most important relationships are captured in the beginning of a design project, but as the architecture is realised, also detailed relations are defined between the views. In the PLM system it is possible to view each model requirements coming from the other models, e.g. engineers are able to view customer requirements affecting the design of the architectural element they are working on.

6. Conclusion

This paper has presented an approach for mapping requirements to a product architecture for the application in product family development. The paper contributes to the process of moving from the problem domain to the solution domain and addresses the real life industrial challenge of simultaneously working on requirements and architectures. The requirement and architectural models used in the approach difference between quality properties and functional actions of the product family and take into account the special characteristics of functional elements. The method supports an initial definition of a limited set of solution neutral customer requirements as well as capturing design driven requirements. Customers can view requirements only relevant for their interest, while designers can view requirements affecting specific modules, sub-systems, components and interfaces. As the architecture is matured and detailed during development additional requirements to chosen solutions are identified and documented in the architectural models. Customer requirements can furthermore be re-evaluated and adjusted as understanding of architectural elements evolves. The result is an interactive approach to work on requirements and architectures. The traceability and continues evolvement of requirements is a crucial aspect of all product development projects. By using a PLM system for representing the product family defining architecture and for mapping requirements to systems, modules and interfaces, it is possible to trace requirements in a straight forward way. Using a PLM system in the approach provides a design team with full traceability from requirements to architecture elements and from architecture elements to requirements. The approach has been validated for functionality, but still remains to be tested in an industrial setting. This work is in progress in collaboration with the software vendor and a large manufacturing company developing products to the construction industry.

7. References

Abramovici, M., "Future trends in product lifecycle management (PLM)". The future of product development, 2007, pp. 665-674.

Akao, Y., "Quality Function Deployment, QFD - Integrating Customer Requirements into Product Design". Portland, Productivity Press, 1990.

Alexander, I., Beus-Dukic, L., "Discovering Requirements – How to Specify Products and Services", John Wiley & Sons Ltd, Chichester, England, 2009.

Bruun, H. P. L., & Mortensen, N. H., "Modelling and using product architectures in mechatronic product development", Norddesign 2012, 2012a,

Bruun, H. P. L., Mortensen, N.H., "Visual product architecture modelling for structuring data in a PLM system", Product Lifecycle Management 2012, 2012b.

Chen, L., Babar,M.A., Nuseibeh, B., "Characterizing Architecturally Significant Requirements", IEEE Computer Society, 2013, pp.38-45.

Cole, R., "The Changing Role of Requirements and Architecture in Systems Engineering", Proceedings of the 2006 IEEE/SMC International Conference on System of Systems Engineering Los Angeles, CA, USA, 2006.

Goedicke, M., Nuseibeh, B. "The Process Road between Requirements and Design", Integrated Design and Process Technology, 1, 1996, pp. 176-177.

Goldin, L., Matalon-Beck, M., Lapid-Maoz, J., "Reuse of Requirements Reduces Time to Market", SwSTE2010: IEEE International Conference on Software Science, Technology, and Engineering , 2010 , pp 55-60.

Harlou, U., "Developing Product Families Based on Architectures Contribution to a Theory of Product Families", Lyngby: Department of Mechanical Engineering, Technical University of Denmark, 2006.

Hubka, V., Eder, W. E., "Theory of technical systems: a total concept theory for engineering design", Computer-Aided Design, 22, 1988, pp. 254.

Hutcheson, R.S., McAdams, D.A., Stone, R.B. and Tumer, I.Y. "Function-Based Systems Engineering (FUSE)" International Conference on Engineering and Design, ICED'7, 28-29 August 2007, Paris France.

IEEE. ISO/IEC 42010 Systems and software engineering — Architecture description, 2011.

Jiao, J., & Tseng, M. M. (2000). Fundamentals of product family architecture. Integrated Manufacturing Systems, 11(7), 469-483.

Jin, Y., Lu, S., "Toward a Better Understanding of Engineering Design Models", in Grabaowski,1998, pp. 73-90.

Jones, C., "Patterns of Large Software Systems - Failure and Success", Computer, 28, 1995, pp. 86-87.

Malmqvist, J., "Implementing requirements management: A task for specialized software tools or PDM systems?" Systems Engineering, 4, 2001, pp. 49-57.

Malmqvist, J., "A Classification on Matrix based Methods for Product Modeling", Proceedings of DESIGN 2002: Proceedings of the 7th international design conference, 2002, pp. 203-201.

McConnell, S., Rapid development : taming wild software schedules, Microsoft Press Redmond, 1996.

McGovern, J., et al., "A Practical Guide to Enterprise Architecture", Prentice-Hall, 2004.

Meyer, M.H., Lehnerd, A.P., "The power of product platforms: building value and cost leadership", Free Press, New York, 1997.

Mortensen, N. H., "Design Modelling in a Designer's Workbench Contribution to a Design Language", Kgs. Lyngby: Department of Control and Engineering Design, Technical University of Denmark, 1999.

Niemelä, E., Immonen, A., "Capturing quality requirements of product family architecture", Information and Software Technology, 49, 2006, pp. 1107–1120.

Nuseibeh,B.,"Weaving Together Requirements and Architectures", Software Management, 2001, pp.115-11.

Pahl, G., Beitz, W., & Wallace, K. (1996). Engineering Design A Systematic Approach. London: Springer.

Robersson, S., Robertsson, J., "Mastering the Requirements Process, 3rd edition", pp. 9-10. Addison-Wesley, London, England, 2012.

Sääksvuori, A., Immonen, A., "Product Lifecycle Management", Springer Verlag, 2008.

Stark, J., " Product Lifecycle Management: 21st century paradigm for product realisation", Springer, 2011, pp. 1-16.

Stevens, R., and Martin, J. (1995) What is Requirements Management? Proceedings of the Fifth Annual International Symposium of the INCOSE, Volume 2, pp. 13-18.

Suh, N.P., "Axiomatic design advances and applications". New York, Oxford University Press, 2001.

Toval, A., Moros, B., Nicolás, J., Lasheras, J., "Eight key issues for an effective reuse-based requirements process", International Journal of Computer Systems Science & Engineering, 6, 2008, pp. 373-385.

Ulrich, Karl. "The role of product architecture in the manufacturing firm." Research policy 24.3 (1995): 419-440.

Web: http://www.idef.com/idef0.htm

DTU Mechanical Engineering
Section of Engineering Design and Product Development
Technical University of Denmark

Produktionstorvet, Bld. 426
DK- 2800 Kgs. Lyngby
Denmark
Phone (+45) 4525 6263
Fax (+45) 4593 1577
www.mek.dtu.dk
ISBN: 978-87-7475-397-1

DCAMM
Danish Center for Applied Mathematics and Mechanics

Nils Koppels Allé, Bld. 404
DK-2800 Kgs. Lyngby
Denmark
Phone (+45) 4525 4250
Fax (+45) 4593 1475
www.dcamm.dk
ISSN: 0903-1685