Mapping requirements to a product architecture supported by a PLM system

Bruun, Hans Peter Lomholt; Hauksdóttir, Dagný; Harlou, Ulf; Mortensen, Niels Henrik

Published in:
13th International Design Conference - Design 2014

Publication date:
2014

INTERNATIONAL DESIGN CONFERENCE - DESIGN 2014
Dubrovnik - Croatia, May 19 - 22, 2014.

DESIGN 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.

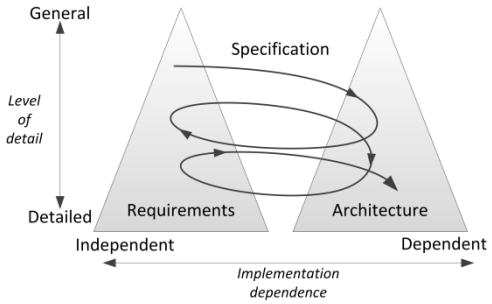

Figure 1: Twin peak model.

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.

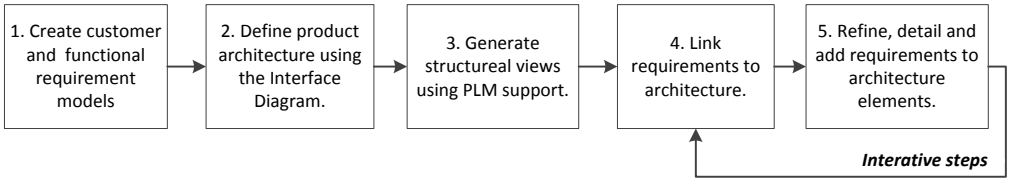

Figure 2 Steps in the approach

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.

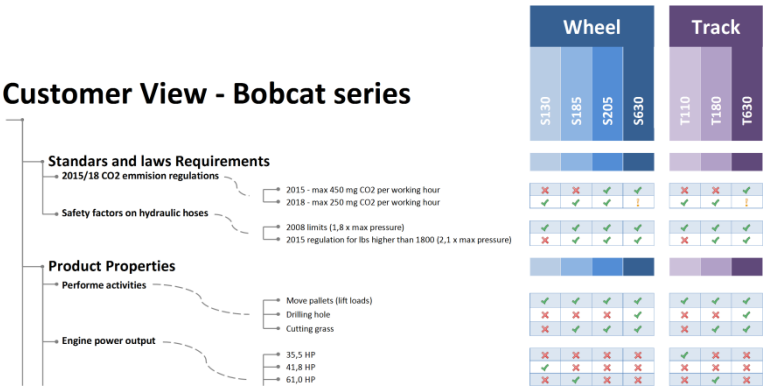

Figure 3 Excerpt from Customer view example of a family of Bobcats

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.

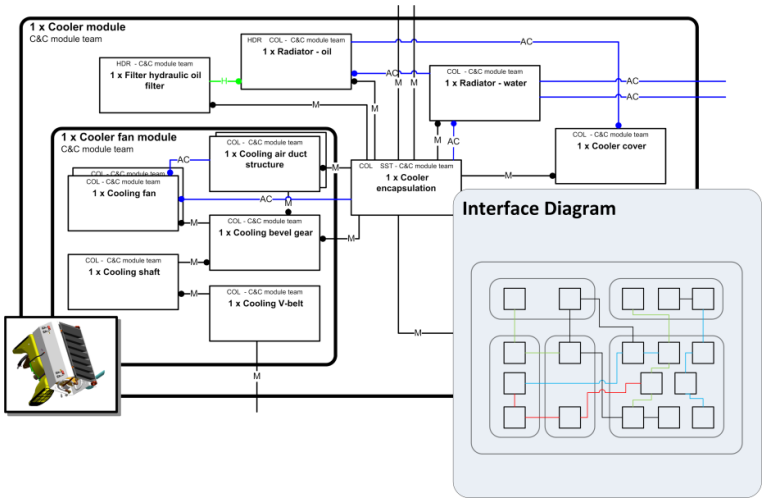

Figure 4 Excerpt from an Interface diagram of a Bobcat by the use of the Interface diagram formalism.

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.

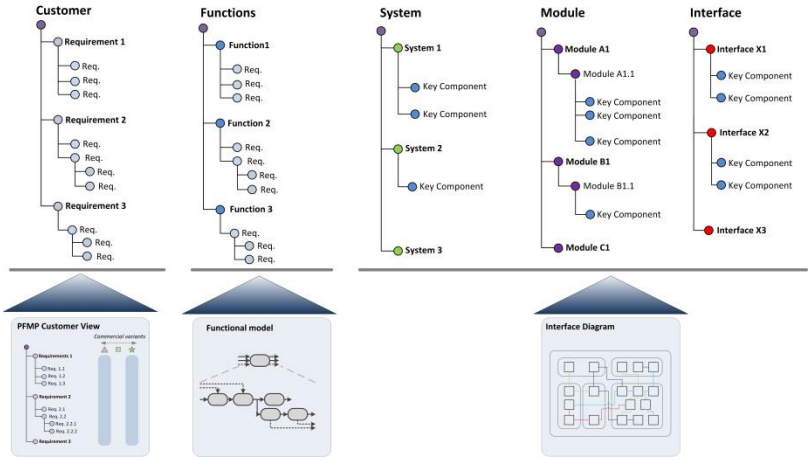

Figure 5 Loading requirements and architectural views into PLM

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.

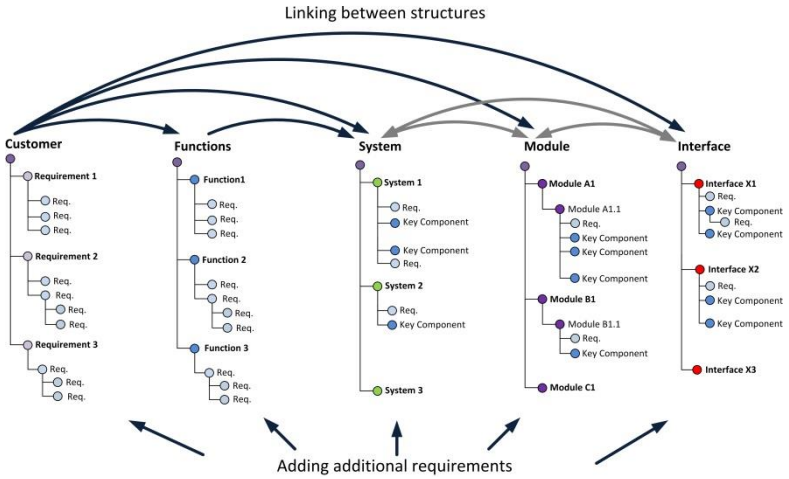

Figure 6 Mapping, detailing, and adding requirements during the development process

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