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

METHOD TO INTEGRATE MODULAR PRODUCT ARCHITECTURE INFORMATION INTO STANDARD IT-SYSTEMS

M. Heilemann, S. J. Culley, M. Schlueter and V. Lindemer

Keywords: modularisation, product architecture, product structure

1. Introduction

Modular product architectures are used to derive high and changeable product variance under low internal complexity and cost [Ulrich 1995]. Researchers have developed a whole range of techniques to establish modular product architectures [Heilemann et al. 2012]. What all these techniques have in common is that they end with establishing the product architecture in the concept phase of product development. However, it is also vital to consequently embody, detail and realize the product architecture concept over a prolonged period. A particularly under-researched area is the representation of product architecture information during embodiment, detailed design and the product architecture lifecycle [Arnoscht 2011], [Heilemann et al. 2012], [LaLande 2013]. Thus, this paper will concentrate on formal representation of the product architecture in standard IT-Systems of companies and the provision of product architecture information to design engineers during the product architecture lifecycle. Within this focus, it is the aim of this paper to present a method for the integration of modular product architecture information into standard IT-Systems.
This paper has been divided into five parts. Chapter 2 reviews the state of the art. Requirements for the integration of modularisation into IT-Systems are derived in Chapter 3. Chapter 4 describes how these requirements can be fulfilled with a newly developed method. Chapter 5 describes how the method was validated and what the output of validation was. Chapter 6 gives a brief conclusion of this paper.

2. Modular product architecture design: State of the art

Product architecture is defined as “composition of a product from a number of component products” [Erens and Verhulst 1997]. According to the definition, each product has an architecture. The architecture can either be integral, modular or possess a certain degree of modularity. A modular architecture embodies one-to-one mapping between functional and physical elements. Another characteristic of modular product architectures is that a design change in one module does not require a design change in another module, assuming that the product still works as intended [Ulrich 1995].
A modular architecture that is designed across a wider range of products can be used to establish a modular system. This is a set of predefined module variants which follow the same product architecture rules to derive final product variants. Product architecture rules are the guiding principles like standardised interfaces, fixed module boundaries, same module arrangement and similarity in function that make module variants interchangeable and reusable for a wide range of products.

2.1 IT-Integration of product architectures

Modelling of product architecture information is frequently done by using graph-based or matrix-based approaches. For instance, Pimmler and Eppinger [1994] take use of design structure matrices

(DSM) to model the product architecture whereas Erixon [1998] uses QFD-matrices. Stone et al. [2000] use functional flow structures to model modules within a product portfolio and Martin and Ishii [1996] use variant trees to represent product variety in relation to the manufacturing sequence. Other information like product specifications or market requirements that is used for product architecture design is shown in a reference model by Kissel et al. [2012].
It is recommended by other researchers to store product architecture information centrally in standard IT-Systems like PLM and ERP. For this, Kreimeyer [2012] presents a four-layered model (three layers plus property and configuration parameters) to handle the product structure in a standard PLM system. The approach adopts state of the art methods to the specific needs of a heavy and light vehicle manufacturer. In the model, the product portfolio layer is connected to the generic product structure of different vehicle systems. Under consideration of the attached property and configuration layer, the right component variants for a specific product variant are identified. The fourth layer is the solution space with concrete embodied design elements.
Supporting engineers in developing interchangeable modules in PDM is the goal of the Interface Diagram Formalism (IFD) [Bruun et al. 2013]. The IFD is a generic view on a product family that represents the interface structure, the system structure and the module structure together with their organizational responsibilities. The system structure of the product family represents the bill-of-material from a functional point of view. The modular structure represents the product breakdown from a module driver perspective. As the name implies, the main element of the IFD framework is the interface structure. By describing ownership and interfaces in detail between components, modules and systems, the elements of the product architecture get interchangeable and manageable by organizational units. The developed framework is supported by integrating the modelled IFD into a PLM environment. This helps to visualize the module and the system view on the product family. In addition, the integration into PLM systems allows for adding attributes like cost information in order to generate more sophisticated reports about the modularisation design process [Bruun et al. 2013].

2.2 Summary

This state of the art section provides the principles on which this work is based on. In order to make a critical assessment of state of the art and to develop an IT-Integration method that is applicable in industry, it is first necessary to collect detailed requirements from both, industry and literature.

3. Requirements for IT-Integration of modular product architectures

The requirements for modularisation transition and the gaps in current approaches were collected in three different studies. First, requirements were collected from the literature. Second, requirements were collected during process and document analysis in the course of modularisation projects. Third, semi-structured interviews and workshops were conducted.

3.1 Requirements from literature

Recent studies undertaking IT-Integration of modularisation found it as necessary to represent the generic product structure of the product range from which the specific structure can be derived [Kissel et al. 2012], and information that ensures that the “product architecture is collision-free for the intended variant models” [Kreimeyer 2012]. Another recent study by Bruun et al. [2013] involved further requirements which are recognizing modules and interfaces, facilitating the search for existing modules for reuse purposes, identifying standard or customized elements of a specific project and carrying out proper monitoring of relational properties.

3.2 Process and document analysis

During participant-observational field-research in industry with a major global manufacturer including benchmark partners, different alternatives how the transition from single product development toward modular system development can be made were identified. These alternatives were analysed and taken to derive particular requirements for IT-Integration of modularisation.

All alternatives have in common are that it must be ensured that all developed module variants contribute to a working modular system from which new products can be derived by combining module variants. It is assumed that this can only be realized if the rules of the modular system are known and met. In order to achieve this, following key requirements have to be fulfilled:

- Newly or changed module variants and interfaces have to contribute to product specifications of all products in scope of the modular system and not just to single products. Thus, this information must be updated and transparent at the point of use.
- Newly or changed module variants and interfaces have to follow the rules of the overall product architecture. By following them, they are reusable and interchangeable. Thus, this information must be updated and transparent at the point of use.

By comparing the detailed design with product architecture rules, it can be shown if the module variants are collision free, interchangeable, according to its reuse plans and still contribute to defined product specifications. The requirements mentioned in this section are conclusions of the process and document analysis. Detailed results are not presented within this paper.

3.3 Semi-structured interviews

Semi-structured interviews were conducted with engineering managers and engineers at a global manufacturer transitioning toward modularisation. The people involved were mainly from the engineering department involved in transitioning toward modularisation, but also engineers and managers from other departments participated. After collecting the requirements, they were checked and prioritised in a workshop with 15 engineering and IT-experts. The requirements were not only checked for validity, but also for realisability. The requirements are as follows:

- Elements (e.g. interfaces, modules, module variants) of the modular system have to be represented
- Ownership of modular system elements has to be represented
- Relation between elements of the modular system has to be represented
- Elements of the modular system have to be checked against violation of product architecture rules
- Elements of the modular system are searchable for reuse purposes
- Data maintenance and naming has to be standardized across all development sites
- Product architecture documents that are valid for several elements have to be stored centrally and linked to the respective product architecture elements

Boundary conditions agreed by the participants are: a) appropriate effort for data maintainers, b) integration into existing core standard software, and c) integrated CAX chain (i.e. CAD, PDM, and ERP).

3.4 Summary

In order to develop a satisfying method for IT-Integration of modularisation, the requirements above were consolidated and condensed. This was necessary because the requirements collected in the three studies were partially duplicates, not on the same level, and sometimes already related to a possible solution. In the end, following solution-neutral requirements have to be fulfilled:

- Product architecture elements that already exist have to be easily found
- Module variants with the same functionality need to be identified
- The product architecture has to be protected or fixed in order to achieve long-term stability
- Artefacts can be checked against product architecture rules and product portfolio specifications at the point of use. (For the purpose of this work, ‘product portfolio’ is defined as the range of products that is defined during product architecture concept phase. It is possible that this range comprises several different product families.)
- Monitoring of the product architecture in terms of checking it against plans and goals must be facilitated

Given the findings of the requirements collection study, all the previously mentioned approaches from the state of the art section suffer from limitations. First, they only have focused either on spreadsheets,

concept visualization, separate software tools, configuration, or the focus is on the integration into PLM systems instead on a general model for the whole CAX process chain and on established core IT-Systems. Second, the existing approaches mainly focus on modules and interfaces and fail to resolve the trade-off between high external product variety and low internal complexity of the overall modular system.

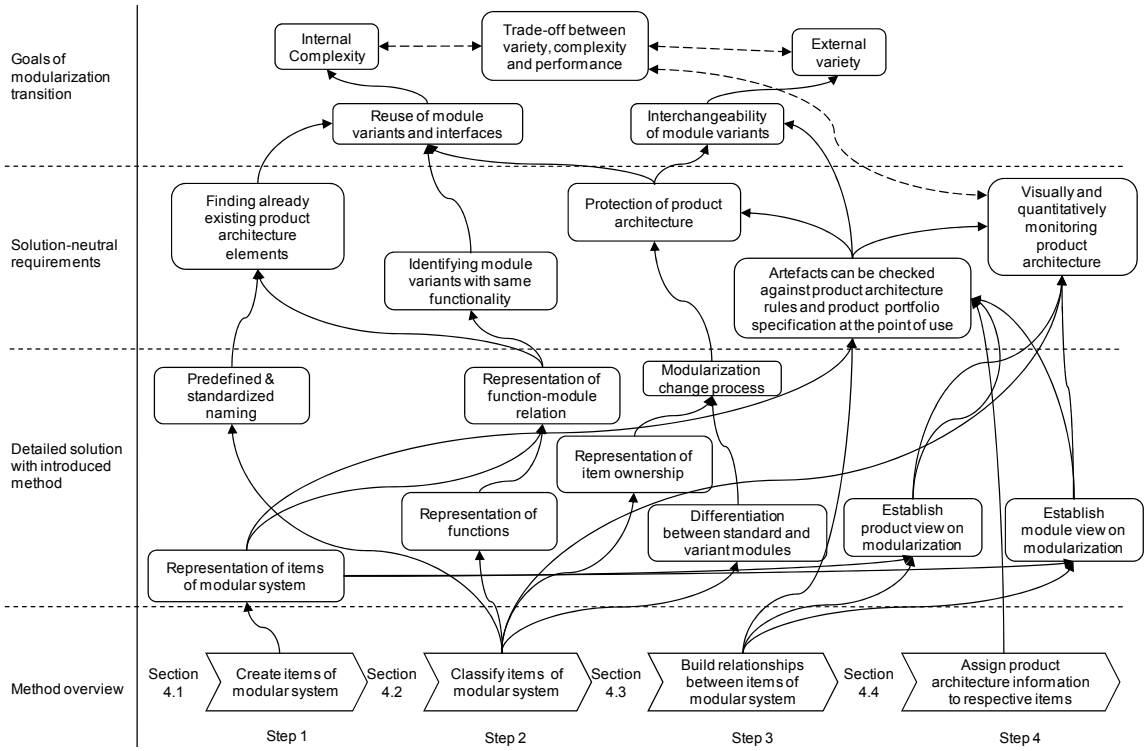

Figure 1. Cause-effect diagram of the IT-Integration method

4. Method for IT-Integration of modular product architectures

To deal with the requirements mentioned above and to overcome the gap of existing approaches, a method for the integration of modular product structures into IT-Systems is proposed. It is the goal of the method to create an information model in standard IT-Systems that helps engineers to transit toward modular system development with stable product architectures. “Stable” in this sense means that the underlying logic of the product architecture is met by all developed elements during the lifecycle of the architecture. The method consists of four interconnected steps. First, items of the modular system are created in the IT-System. Second, the items are classified according to the identified requirements of modularisation. Third, relationships are built to show how the items interconnect. Fourth, product architecture information (e.g. data specifying functions, interfaces, modules and products) is stored centrally by assigning it to the respective items. Figure 1 shows how the method is designed to fulfil the requirements set by this study and in addition how to achieve the goals of modularisation.

4.1 Step 1: Create items of modular system

The first step of the method starts with creating the items of the modular system. The items are the labelled boxes in bold type in Figure 2 (e.g. “Item: Modular System – Overall View”). For easier reference, the items are numbered with characters from A to H in Figure 2.

The items are used as reference and information carrier for Step 2 to Step 4. Usually, the product structure in standard IT-Systems consists of following items: products, assemblies, and components. This means that in the current way a *product* is made of *assemblies* or *components*. In turn, assemblies consist of other assemblies and components. After transitioning toward modularisation, a *product portfolio* is composed of *module variants* complying with the architecture of the underlying *modular*

system. In order to get a view not only on single products, but on the product portfolio (product view) and on the modular product architecture (module view), following items are introduced:

Modular System – Overall View (A), Modular System – Product View (B), Modular System – Module View (C), Products (D), Module Variants (E), Modules (F), Assemblies (G), Components (H) See Section 4.5

It depends on the individual needs and common processes of the company applying this method in which IT-System the items are created. For instance, it seems quite clear that components are created within CAD and then transferred to PLM and ERP. A modular system item can be created as plain item in PLM. On the other hand it can also be created in CAD with information about geometric positions for all relevant modules before it is transferred to PLM.

4.2 Step 2: Classify items of modular system

In order to make information about the items of the modular system explicit, the items are classified with descriptive properties, attributes, or characteristics (for examples, see below). This "Classification" is assigned to the respective item from Step 1. The predefined and unique entry options for classification values are predetermined during product architecture concept phase and assigned to the respective elements during modular system design.

For *reuse purposes* and to identify module variants with the same functionality, the standardized characteristics "Item_Name" and "Characteristic 1 … Characteristic N" are introduced. The "Item_Name" gives the item a company-wide agreed and standardized name like "Servo_Pneumatic_Positioning_Module". "Characteristic 1" to "Characteristic N" are used to uniquely identify the functionality or characteristics of the respective item. For instance, the "Servo_Pneumatic_Positioning_Module" can be uniquely identified by its functional characteristics "Stroke", "Output_Signal", and "Piston_Diameter" and the respective predefined values. By setting predefined values, it is not possible that module variants with functionality that is out of scope of the modular system is designed and built into products. This is particularly valid for intentionally excluded module variants with low profit contribution.

For the *purpose of protecting* the product architecture from making changes that violate against product architecture rules, two characteristics are assigned to the respective module variant: the "Item_Owner" and the "Variety_Level". The item owner is the responsible person for the item and decides how the item is handled. For instance, the change process for introducing a module variant or changing an interface passes the item owner because this is a designated role that has the overview about the impact on the product architecture. The "Variety_Level" of the item has an influence on the change process as well. It may have similar values such as "Standard" or "Variety". The item has to be particularly protected if it is "Standard". If it is a "Variety" item, it can be easily changed to generate variety.

The system classification depends on two factors: a) where the company usually does its classification and b) where the change process to create and change items is handled. The required functionality of the standard IT-System can be provided by both PDM and ERP systems.

4.3 Step 3: Build relationships between items of modular system

How different information and design artefacts interconnect is obvious if development projects are undertaken for a small range of products. However, the task gets more complicated if a large product portfolio is to be derived from the same modular system. Therefore, relevant items are linked to each other and set into a certain product architecture hierarchy. This means also that items are linked to the respective information of their master item. Practically, this has a few beneficial effects: it shows the relation of a master item to a slave item (e.g. the relation of a product to a product portfolio), and it shows which product architecture rules have to be considered by a design engineer (e.g. while designing a "Module Variant", the rules of the master items "Module" and "Modular System" have to be considered).

The items of Step 1 have to be related to each other by following the associative UML notation of Figure 2. The different associations are numbered from 1 to 9. For instance, a module can have one to infinite module variants (1..*) while a module variant always belongs to one certain module (1..1).

4.4 Step 4: Assign product architecture information to respective items

Product architecture information is established during the product architecture concept phase and kept more or less informally in project folders and the like. If this information shall be updated constantly, it is necessary to provide this information centrally in Standard IT-Systems in a formal manner. By assigning product architecture information from the product architecture concept phase to the interconnected items during design, items can be checked against product architecture rules and specifications of the whole product portfolio. This means also that needed information can always be traced back to its origin from where it is needed. There is another point in assigning data to the respective item. By taking the plans from the product architecture concept phase, the actual design of the modular system can be checked against these plans.
It becomes obvious that it is of advantage to store information centrally at the master item. In this case, the information is free of duplicates, automatically connected and traceable for slave items. For instance, the boundaries for a module item are stored centrally with the module item but they are equally valid and available to all module variants that must adhere to the boundaries and interfaces of the master module.
In the case of ERP and PDM Systems, it is possible to directly assign files in any format to the items. If items are taken directly from CAD-Systems, they already contain the required information in the form of 3D-CAD data.

4.5 Information model for modular systems

The result of processing Step 1 to Step 4 of the method is an information model that represents information about the modular system. It consists of following related elements: items, classification, relationships and assigned product architecture information. Following points give an overview about the introduced elements of the modular system (see Figure 2):

Item: Modular System – Overall View (A):
a) Purpose: The purpose of this item is to act as information carrier and as reference for all information that is valid for the overall modular system. This means that it is used to connect the product view on the product portfolio with the module view on all modules and module variants of the modular system. Therefore, this item is used on all matters that handle the trade-off between product variety (i.e. product management perspective) and internal complexity (engineering perspective). It is assumed that the two different views are contradicting and, thus, need special treatment.
b) Classification information: In the classification section, it is necessary to name the modular system and to assign an item owner. The item owner is handling the trade-off between the product management view on the modular system and the engineering view. It is also possible to assign those functional characteristics to this item that must be fulfilled by the product portfolio. In turn, these functional characteristics can later be broken down to module level.
c) Additional information: This item contains attached data that is relevant for all products and all modules of the modular system. For instance, this could be a matrix relating the modules and module variants of the modular system to the products where they are planned to be used.
d) Relation information: In the IT-System, this item is set into relation with the item "Modular System – Module View" (association 2) and with the item "Modular System – Product View" (association 1). This means that there is a product view on the modular system and a module view. In terms of hierarchy, this is the highest item in the product architecture information model.

Item: Modular System – Product View (B):
a) Purpose: The purpose of this item is to act as information carrier and as reference for all information that is valid for the product portfolio that is derived from the modular system. This item accommodates the requirement that modules are not only developed for single products but for a broad range of current and future products, even across product families and brands.
b) Classification information: In the classification section, it is necessary to name the modular system to which the item belongs to and to assign an item owner. The item owner is the role in charge for the products that have to be derived from the modular system. It is usually a product management role

with the goal to have high variety derived from the modular system. It is also possible to assign those functional characteristics to this item that must be fulfilled by the product portfolio.
c) Additional information: This item contains attached data that is relevant for all products of the portfolio. For instance, this could be the product portfolio requirement specification, a product roadmap or the overview of all functional characteristics and their values.
d) Relation information: In the IT-System, this item is set into relation with all effectively designed products of the modular system (association 3). Hence, it becomes clear which products belong together and to which central specifications they have to adhere.

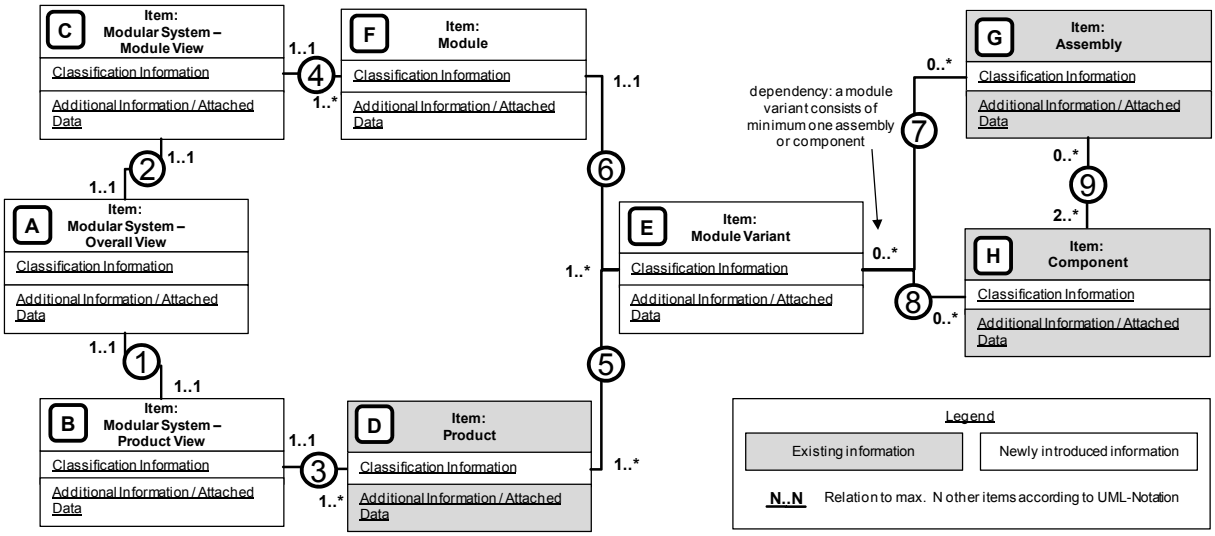

Figure 2. Information model for IT-Integration of modularisation (for an example, see Figure 3)

Item: Modular System – Module View (C):
a) Purpose: It is the purpose of this item to establish a module view on the modular system connecting all modules and centrally providing information for all modules. This item accommodates the requirement that modules are not only built for single products, but also for the modular system with the intention to easily derive further products in future.
b) Classification information: In the classification section, it is necessary to name the modular system the item belongs to and to assign an item owner. The item owner is the role in charge for the modules that have to feed the modular system. It is usually an engineering role with the goal to have low complexity in the modular system. It is also possible to assign those functional characteristics to this item that must be fulfilled by the modules.
c) Additional information: This item contains attached data that is relevant for all modules in scope. For instance, this could be a module roadmap, an overview of all planned modules, the functional structure the modules have to fulfil or the manufacturing sequence in which the modules are manufactured.
d) Relation to other items: This item is set into relation with all modules of the modular system (association 4).

Item: Product (D):
Product items are not treated differently as in current IT-Systems. Thus, the classification section of this item can be left blank or filled with ordinary classification data for product functionalities. The difference of this item is that in a modular architecture the product item is not related directly to assemblies and components but to module variants (association 5).

Item: Module Variant (E):
a) Purpose: It is the purpose of this item to realize the carefully drafted modular architecture of the products and the modular system.

b) Classification information: In the classification section, the item is named with a standardized name, the owner of the module is assigned, the variety level is assigned and the distinctive characteristics are filled out with their values (see respective relation to Item: Module (F)).
c) Additional information: This item contains attached data that is similar to the data of ordinary assemblies (e.g. drawings or part lists).
d) Relation information: On the one hand, the module variant has to contribute to the product characteristics of the final product. Logically, it is set into relation with respective products (association 5). On the other hand, the module variant has to be designed for the modular system in order to derive products from it in future. For this reason, more characteristics have to be met than those of the current product. This is why the module variant is also connected to its abstract master module item (see Item: Module (F)) that gives an overview and contains rules for a stable modular system (association 6).

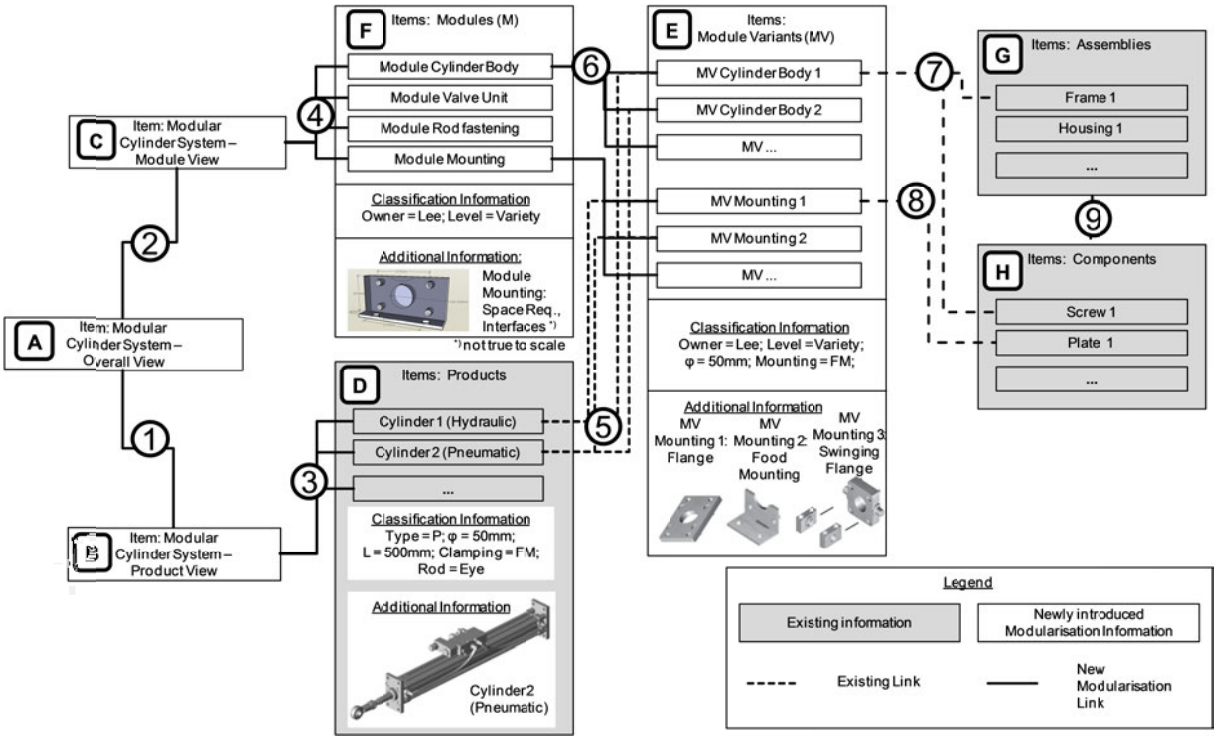

Figure 3. Example of the information model based on a pneumatic cylinder (CAD images based on the publicly available e-business portal of Bosch Rexroth AG [2014])

Item: Module (F):
a) Purpose: This is an abstract item that has the purpose to centrally store information about respective module variants and to provide an overview about the connected module variants of a certain category. For example, a "Linear_Drive" module is related to the module variants "Linear_Drive_20N", "Linear_Drive_30N" and "Linear_Drive_45N". For reasons of interchangeability and reuse, all module variants have to follow the same rules of the module master item.
b) Classification information: In the classification section, the standardized module name (same like respective Item: Module Variant (E)), the module owner and the variety level have to be assigned to the item. The module owner is the role in charge that has the complete overview of the module and its connected module variants. For example, if a module variant shall be changed, the item owner has to decide whether the change is in line with the module rules or whether the module rules have to be changed as well. The characteristic "Variety_Level" sets the hurdle for the engineering change process. If a module is classified as "Standard", the change process will be more restricted than for a "Variety" module. The item owner is also the person who defines functional "Characteristic 1" to "Characteristic N", assigns it to the module item and determines which values for each characteristic have to be fulfilled by the respective module variants (see Chapter 4.2).

c) Additional information: The attached data section centrally contains information that is valid for all module variants of the respective module. This comprises interface specifications, interface drawings, geometric module boundaries (e.g. in 2D or 3D CAD format), or written module specifications.
d) Relation information: To realize the relation between modules and module variants, association 6 is installed in the IT-System.

Item: Assembly (G) and Item Component (H):
These items are not treated differently as in current IT-Systems. Therefore, the classification section of this item can be left blank or it can be used for other purposes. In case of modular products, these items are not directly related to products but to administrated module variants (associations 7, 8, 9).
Figure 3 shows an illustrative pneumatic cylinder example for the information model for IT-Integration of modularisation.

5. Validating the method

The method was validated within a modularisation project at a major global manufacturer. For confidentiality reasons, no details about the company or implementation examples can be published.
However validation was conducted in several steps. First, the drafted method was discussed with benchmark partners from other organizations and with IT-experts. Second, the method was applied on a separate database and on a test server and afterwards discussed with potential applicants and mangers. After amending the method based on feedback, the method was launched in a pilot project. For this, design engineers who are involved in the project were trained and the final method was again discussed within an expert group. Lastly, the method or rather the information model for IT-Integration was partially implemented within the collaborative modularisation project. The project took 17 months and involved 20 engineers, managers and software specialists from four sites.
The validation phase showed that the method is applicable in the field. Moreover, it showed that the requirements for IT-Integration are well selected and the introduced method supports to fulfil these requirements. It is indicated that this will support to better achieve the goals of modularisation transition.
Before conducting the validation project there were concerns about the additional effort it takes to handle product architecture information in IT-Systems. However, effort for the introduction, classification and linking of the information model was tested to be considerably low as this has only to be done once at the beginning of the project and can be assisted by reuse/copy-and-paste functionality within PLM and ERP.
The validation phase uncovered an issue in use that needs further consideration. The method aimed to establish the same modular architecture within the CAX process chain, i.e. in CAD, PLM and ERP. However, a single situation was identified where it was not appropriate to establish the same view on the product architecture between design and manufacturing. In consequence, there are three different possibilities how to handle this point: a) the method is only applied on a single leading system (i.e. CAD, PLM or ERP), b) it is made clear and ensured that all company functions have the same view on the modular architecture, or c) the method has to be extended in future research in order to accommodate different views of the product architecture. In this context, there is another point that needs further consideration. If there is not the same structure between PLM and ERP, it is not necessary to have the modularisation items installed additionally in ERP. These items are necessary as information carrier in PDM (e.g. 3D CAD data) and to generate a visual overview about the product architecture. The question was raised how the method can be conducted within ERP without introducing additional items. This point will have to be analyzed in future studies as well.

6. Conclusions

This paper presents a method that supports companies in the transition toward stable modular systems by making product architecture information explicit in standard IT-Systems during embodiment, detailed design and the product architecture lifecycle. This has been validated in a series of workshops, discussions and an implementation project over a 17 month period. The main achievements of the method are that it a) broadens the view of engineers from single products to a

wide range of products, b) it supports engineers in designing module variants that comply with the overall modular system and can be reused in a wide range of products, c) it shows how product architecture information can be directly and explicitly provided across products and module variants, and d) it links into established and core company IT systems and CAX process chains seamlessly.
Even though that the findings of the study were closely compared to literature, benchmark cases, expert opinion and theoretically validated, the limitation of this study is that the evaluationary findings come from a singly applied case study. Thus, it is suggested that a study similar to this one should be carried out within another company or industry context. This will also help to clarify the issue with different product structures between PLM and ERP systems that was raised during validation. However, the ongoing use of the approach will add substance to its value.
While the IT-Integration method introduced provides a sound base to computationally capture product architecture information in a structured way, further investigation is needed on how this information can be analyzed, presented and condensed for monitoring of product architectures.

Acknowledgement

The authors want to thank all contributors of this study. They are particularly grateful to the Bosch Group and its collaborating organisations as well as to the University of Bath for support, guidance and valuable insights.

References

Arnoscht, J., "Beherrschung von Komplexität bei der Gestaltung von Baukastensystemen", Thesis (PhD), University of Aachen, 2011.

Bosch Rexroth AG, "eBusiness bei Rexroth - Kataloge und Konfiguratoren", (Online), Available from: <http://www.boschrexroth.com>, Accessed 17 March 2014, 2014.

Bruun, H. P. L., Mortensen, N. H., Harlou, U., "PLM support for development of modular product families", In Proceedings of the 19th International Conference on Engineering Design (ICED13), Seoul, Korea, 2013, pp. 379–388.

Erens, F., Verhulst, K., "Architectures for product families". Computers in Industry 33, 1997, pp. 165–178.

Erixon, G., "A method for product modularisation", Thesis (PhD), Royal Institute of Technology, Stockholm, Sweden, 1998.

Heilemann, M., Schlueter, M., Culley, S. J., Haase, H. J., "Methodologies toward product architecture improvement in theory and practice", In Proceedings of 12th International Design Conference - DESIGN 2012, Cavtat, Dubrovnik, Croatia, 2012.

Kissel, M., Bradford, N., Kreimeyer, M., Lindemann, U., "Product structure management as the backbone of engineering design: exploration of a reference model". In Proceedings of 12th International Design Conference - DESIGN 2012, Cavtat, Dubrovnik, Croatia, 2012.

Kreimeyer, M., "A product model to support PLM-based variant planning and management". In Proceedings of 12th International Design Conference - DESIGN 2012, Cavtat, Dubrovnik, Croatia, 2013.

LaLande, M., "Overcoming barriers to a successful vehicle modularity strategy", SAE Technical Paper, SAE International, Warrendale, 2013.

Martin, M. V., Ishii, K., "Design for variety: A methodology for understanding the costs of product proliferation", In Proceedings of the 1996 ASME Design Engineering Technical Conferences and Computers in Engineering Conference, Irvine, California, 1996.

Pimmler, T. U., Eppinger, S. D., "Integration analysis of product decompositions". In ASME Design Theory and Methodology Conference, Minneapolis, 1994.

Stone,R. B., Wood K. L., Crawford, R. H., "A heuristic method for identifying modules for product architectures", In Design Studies, Vol. 21, No. 1, 2000, pp. 5–31.

Ulrich, K., "The role of product architecture in the manufacturing firm", Research Policy, Vol. 24, 1995, pp. 419–440.

Markus Heilemann
Department of Mechanical Engineering
University of Bath
Email: M.Heilemann@bath.ac.uk