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

IMPLEMENTING ARCHITECTURE IN INDUSTRY

M. Kreimeyer

Keywords: product architecture, framework, design process, value

1. Introduction

Product architecture is an accepted approach to design products aligned with customer needs while making best use of design principles such as modularity and commonality [Robertson and Ulrich 1998]. As such, product architecture design is a key step in the design of new and the adaptation of existing products [Ulrich and Eppinger 2012].
Commercial vehicle design has specific needs with its comparatively high complexity, i.e. a wide variety of products and comparatively small production volumes; hence, a well-targeted modular design of a commercial vehicle is a business need to enable profitability. Commercial vehicle manufacturers historically have designed their products in this manner.
This paper explains the introduction of a formalized product architecture design process at a major German commercial vehicle manufacturer. As such, the paper represents a case study from a practitioner's point of view. It reviews the progress of implementation and the steps made towards today's level of implementation. To do so, the paper makes use of an 'architecture framework', which was used to systematize the effort of setting up the new organizational function and monitor the progress towards the goals as described by the framework.

1.1 Problem description: Product architecture implementation in vehicle design

The organizational function "product architecture" was created initially as part of a major reorganization at the company. This reorganization was driven by the evolution of markets the company caters to: Originally, the company had focused on European premium markets, and with the company's growth, international markets came more and more into focus. These markets are harder to understand for an engineering centre located in Germany, and, therefore, a more market-focused product management division was installed to collect requirements from the markets and channel them into the engineering design process.

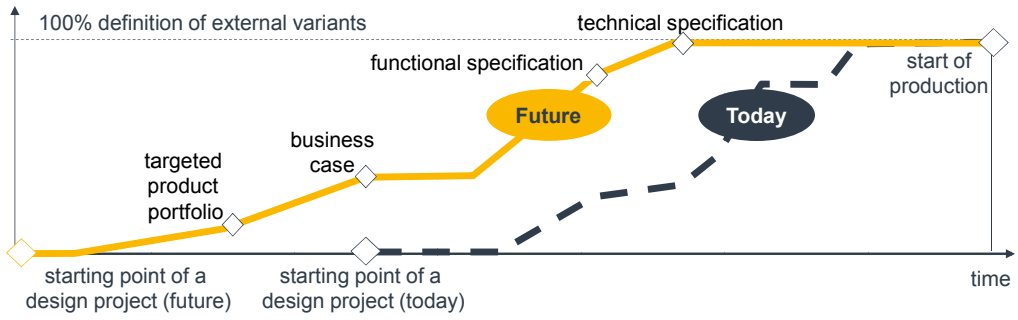

Figure 1. Suspected value of product architecture design at the start of implementation

At the same time, the engineering design process template (i.e. the standard procedure that each design project follows) was adapted to cater for what the company considered “frontloading”, i.e. a more focused concept design and validation phase (including all variants needed) to counteract delays in finalization of designs and their ramp up in production. This frontloading was reflected in a process that, as an input, received a functional specification and replied to it with a technical specification to conclude the concept design phase and have a clear gate to transit to the industrialization of a concept (in German: “Lastenheft” and “Pflichtenheft”). Such procedures are common practice with external contractors (as, e.g., DIN standard 69901 prescribes), which respond to a request for a quote (i.e. the functional specification of what is needed) with a certain bid (i.e. a technical specification of how they intend to solve the problem).
In this initial situation, the precise scope and value of product architecture were rather vague; at the time, the intended scope and value was initially believed as shown in Figure 1. The goal was the early determination of the feasibility of a product design, the alignment with the market and the early identification of the properties of the design, specifically the necessary variants and their return on investment with regard to the functional specification through the targeted product portfolio.
With this background, the problem at hand was to answer three questions: “What is product architecture for the company?”, “What value does it have for the company?”, and “How can the progress be illustrated and tracked?”. The initial scope was described through an initial architecture framework, which is explained in the next section. This framework, however, has evolved through the experiences made during its implementation and, partly, through changing boundary conditions. Hence, the focus of this paper is to identify, additionally, the evolution and reasons for it with regard to the questions above.

1.2 Research approach and methodology

As a research approach, the actual work done at the company was protocolled and reviewed periodically with outside experts from academia and consulting to ensure a thorough and neutral reflection of the observations made. Figure 2 illustrates this approach. While driven and managed as an industrial approach, both scientific and industrial reviews were run regularly over a period of about three years to ensure that the focus was not lost and that both industrial and academic best practices were considered. This was done through a series of students writing their master theses in this context and through regular workshops with researchers from various institutions.
As part of the initialization of this research, an intial framework was set up to generate a reference about the expected observations and to obtain a clear picture of the scope of architecture design within the specific context of the company. This intial framework was set up based on the state of the art from academia, as explained in [Plaikner et al. 2012].
This framework was then reviewed with several industrial partners both from other OEMs and from consulting. The OEM reviews were done as workshops on the exchange of experiences made in product architecture management with other companies of a similar scope but not being competitors in commercial vehicle design (especially farm equipment and construction machinery, about twice yearly). Academic reviews were done as part of two research projects that were run in parallel every few months.
All workshops were run with all members of the architecture department (about ten persons today). At each step, the results were verified internally with management to confirm the progress and to ensure the overall direction. To do so, the “users” of the architecture process, i.e. design project managers and senior engineers, were part of these reviews, too, to provide their feedback on the relevance of the new process and the underlying methodologies.
This was done to improve, above all, the implementation process at the company, but also to identify best practices and scientific principles. Also, the progress was measured to ensure the overall goal was reached, adding to the value that the newly installed process was to deliver.

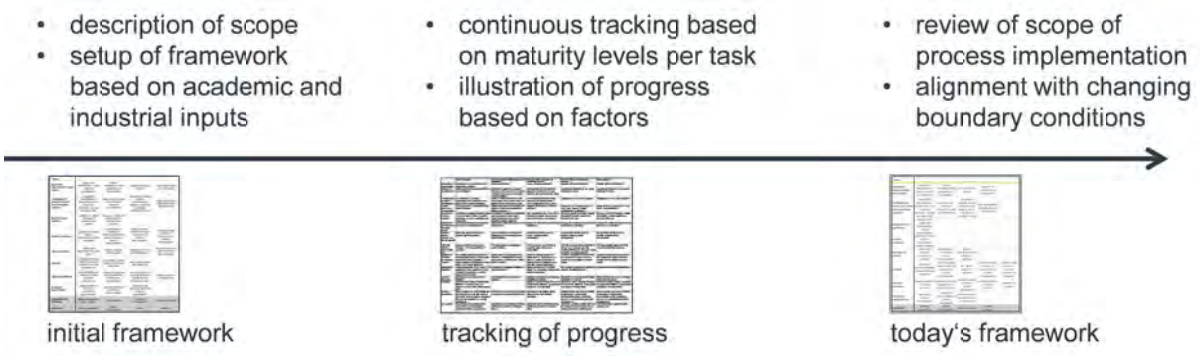

Figure 2. Overall procedure

1.3 Structure of this paper

As a start, the state of the art in both academia and industry is reviewed to deduce the core tasks of architecture from these perspectives. In the following chapter, the initial setup of architecture at the company is regarded, and based on a reflection of how architecture was implemented step by step, an updated framework is presented. Based thereon, the value of architecture design is reflected to contrast the initial expectations shown in Figure 1, followed by a summary and a conclusion.

2. Architecture – state of the art

As a context to this research, a small state of the art is given. It lists, in addition to the common references from academia, industrial "best practices" that were used in the implementation process.

2.1 Architecture definitions and characteristics in research

Literature provides many definitions for architecture, which have evolved mostly over the past 20 years, starting with [Ulrich 1995] stating that product architecture has three main characteristics:

- the arrangement of functional elements or functional structure
- the mapping of functional elements to physical elements
- the specification of the interfaces among interacting physical elements

In other words, a product architecture consists of both a functional structure and a physical structure. Other authors (for a comprehensive literature review, see esp. [Jose and Tollenaere 2005]) have followed this definition, relativizing it in more abstract terms as a configuration between components of the product and the tasks that each component should do or as the "scheme", by which the product's functionality is allocated to the physical structure, which is segmented into the "physical building blocks" and their "interactions" [Ulrich and Eppinger 2012].

However, variant design is not directly included in these definitions, but is rather brought in through the definitions on "product platforms", which are considered to be closely related to product architecture: "A product platform is a set of architecture, common modules and interfaces from which a stream of derivative products can be efficiently developed and launched." [Friedel 2011]. More generally, literature, e.g. [Robertson and Ulrich 1998] defines a platform as the "collection of assets that are shared by a set of products", i.e. they extend the notion on the physical building blocks and their difference and commonality across a range of products, also stating that the standardization or parts alone does not result in a platform. As the latter definition relates closely to the needs of commercial vehicle portfolios (a mapping of components and functions, incorporating different variants), it is this definition that is followed in this research.

Authors furthermore differentiate a number of characteristics of architectures: above all, modular and integral architectures as the two poles of architecture design, between which there is a continuum of possible blends [Ulrich and Eppinger 2012]. [Ulrich and Tang 2005] introduced six different kinds of modularity for architectures: sharing components; swapping components; cut-to-fit; mixed components and modules; bus architectures; and sectional architectures. Schuh et al. [2007] concretize this by a procedural model that includes working with requirements, setting up functions and

properties of a product, relating the physical components, and the standardization of components as the different domains of "doing architecture". More generally, the "architecture process" therefore can be seen as converting a desired behaviour (given by requirements and/or functions) to a solution (given by components), not just for one integral product but a product portfolio, based on modularization to achieve commonality among the different variants within this portfolio.
This is confirmed when looking at the value of product architecture. While architecture is based on a trade-off between distinctiveness and commonality [Robertson and Ulrich 1998], authors point out that platforms or modular architectures help to leverage different markets and product behaviours (also regarded as performance levels or performance steps) [Simpson et al. 2001]. However, the trade-off also implies that when designing the behaviour of a product portfolio, the actual decision making to obtain the right trade-offs is an essential part of this process [Robertson and Ulrich 1998].
In summary (see Table 1), the following aspects are considered "part of architecture":

Table 1. Characteristics of product architecture

Characteristic	Description, keywords
Behaviour	Requirements, functions, characteristics, product behaviour
Components	Chunks, physical decomposition, building blocks
Mapping	The actual "architecture", scheme
Architecture approach	Integral architecture, modular architecture, strategy
Variants	Product portfolio coverage, markets, technical and market variants
Commonality	Standardization or parts, modules, modularization
Decision making	Trade-offs, distinctiveness, commonality

2.2 Architecture in industry – automotive design

In general, information on product architecture in industry, such as automotive industry, is hard to obtain, as the architecture and modularization strategies are generally considered a key strategy to most businesses. A few examples, however, can be given in the following paragraphs.

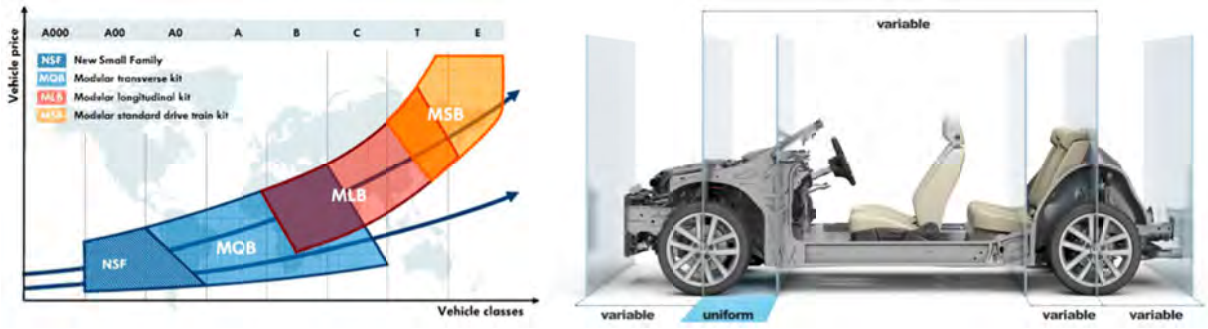

Figure 3. Volkswagen "assembly kits" [VW 2013a] and MQB kit [VW 2013b]

Scania CV, a major Swedish commercial vehicle manufacturer, uses a modular system approach to design and build a wide range of vehicles based on a common product architecture. This architecture approach is based on three principles [Gustavsson and Eklund 2013]:

- Standardized interfaces between components
- Well-adjusted interval steps between performance classes
- Same customer-need pattern = same solution

With that as a basis, the Scania design process is a continuous update process towards a technical portfolio of components. This implies that no platform or specific models exist and that requirements are formulated for each equipment to serve the whole product range (or a part thereof) instead of regarding requirements towards individual models oft he product range [Gustavsson and Eklund 2013].
Opposing that, passenger vehicle industry usually works with platforms that serve as a common standard and as a basis for several models of the product range that are derived individually from such

a platform. Volvo, for example, uses three platforms that all current models are derived from; in turn, design is done towards a model based on one of the platforms, and requirements are set for a model [Gustavsson and Eklund 2013]. Volkswagen similarly uses a number of "assembly kits", which can roughly be compared platforms enriched by common modules across all platforms (in German, they are called "Modularer Baukasten", i.e. "modular kit"). Currently (Figure 3, left hand side), there are four assembly kits available, each of which standardizes certain components and their topology (Figure 3, right hand side) as well as modules that are shared with other assembly kits.
BMW similarly develops product lines, which serve as a starting point to derive models based on model-specific requirements; however, such architectures are based on a standardized product structure [Hablhuber 2012] to facilitate the commonality across models and to facilitate the communication of all involved stakeholders planning the usages of components utilized across more than one model.
In summary (see Table 2), a few aspects can be added to those already shown in Table 1.

Table 2. Characteristics of product architecture

Characteristic	Description, keywords
Standards	Positioning of components, usage of components and modules
Interfaces	Standardization, modularization, designation of standardized modules
Design approach	Platform, portfolio-wide modules, individual models, common parts
Product structure	Product decomposition, common language, common "cut" of product
Component usage	Designation and planning for reuse, usage and variant description
Requirements approach	Focus on models, on markets / market segments, on equipment

3. The evolution of architecture during its implementation

To plan for a complete implementation of architecture design, it was necessary to formulate the concrete goals of what architecture design should be about and measure its progress. To this end, a company-specific "architecture framework" was designed initially. It was based on the state of the art and on industrial practice, as summed up in Table 2. More specifically, the framework was meant to help ensure that architecture design at the company would be complete, consistent and correct. The framework was especially intended to project a vision for the needed models, methods and tools and to measure the progress of installing an architecture function within the existing design department. Furthermore, it was meant to help communicate the idea of architecture.
Therefore, the framework was set up as a table that contained the different "topics" that architecture design at the company was to be about, and for each topic provide the "tasks" of the architecture process as well as certain measurement critera that would help understand how complete the implementation of each topic was.

3.1 The initial framework

As a starting point, a framework was desgined [Plaikner et al. 2012]. It was, essentially, adapted as a simplified version of the Zachman Framework [Matthes 2011], as it offered both a certain flexibility that would support a later scalability and a representation intuitive enough that it could be used within the company without much explanantion.
The framework, in its initial state, was filled through literature research from both academic and industrial references (Table 2 provides a highly aggregated summary thereof). It thereby served as a starting point for the following company-internal discussions. Based on these findings, a series of workshops and discussions with in the company were run to adapt these findings to the company boundary conditions, the company-internal nomenclature and to the needs of the process, i.e. certain tasks were focused while others were dropped. These discussions accompanied the design of the actual architecture process and its methodology (see e.g., [Kreimeyer et al. 2011]), which were led at the level of the company's top management, with a reprensentative group of project managers from different vehicle design projects and with the staff of the architecture design group. As a part of this, the framework was refined through a series of workshops consulting companies and academic partners,

who accompanied the implementation project. Figure 4 shows details the framework as it was consolidated initially, and it lists the topics and tasks of archicture design at the start of the implementation of the architecture design process at the company.

Topics				
Specbook, requirements, spec codes	support the identification of spec codes and combinatorics	ensure completeness and consistency of combinatorics	visualize external variants	build codes based on spec codes
verification of return-on-invest and production volumes	support the identification of planned return-on-invest and production volume per spec code	map return-on-invest and production volume to combinatorics	consolidate external variants (combinatorics) based on return-on-invest and production volume	align consolidated external variants with specbook
functions and systems	update and extend catalogue of functions and systems	map spec codes and requirements to functions and systems		
product structure	update and extend product decompsition where necessary	provide generic mapping of functions/systems to upper structure of product decomposition	identify/manage variant drivers per component of product decomposition	allocate concept solutions to components of product decomposition
carry-over-parts	identify and designate carry-over parts	identify new and derived parts	assess cost of new and derived parts	allocate carry-over parts in product decomposition
variants	compute necessary variants based on variant drivers and combinatorics	consolidate variants based on technical restrictions	align variant design with specbook	
concept solutions	help identifying and detailing concept solutions (alternatives)	review and evaluate concept solutions and ensure completeness	ensure selection of concept solutions and document decision process	identify long-term testing needs
technical specification	generate draft of technical specification documentation	generate technical specification documentation	compare initial specbook and final technical specification	
organizational interfaces	target management (cost, weight)	bill of material	innovation management	package and DMU
principles	enable quantitation	ensure completeness	review	optimize

Figure 4. Initial architecture framework [Plaikner et al. 2012]

	historic situation	first prototypical architecture process in a project	first project fully supported by architecture process	first full rollout of architecture process	future projects
generation or requirements	decentralized, no specific process for requirements collection	central specifications	centrally collected requirements	centrally collected requirements	centrally collected requirements
structure of specifications	unstructured list of requirements with no specific template	reworked by engineering to generate initial spec code structure as a basis for a transition to the sales codes (called "order possibilities")	unstructured list that only late in the project was transferred into a more structured spec book using spec codes	requirements structured by spec codes in formalized manner	requirements structured by spec codes in formalized manner
combinatorics as part of specifications	no combinatorics available; combinatorics only derived late in the process from technical constraints	partially at total vehicle level for main variant drivers such as wheel bases etc. towards the end of the concept phase	no combinatorics designated during project start but generated during concept design phase for core areas	combinatorics of spec codes available	combinatorics of spec codes available
change process on specifications	no change process, communication via emails and phone	change management not formalized but driven by centralized communication through one person	simpplified change process, driven by one person	change process for requirements and spec codes; however, not fully operational for combinatorics	change process for requirements, spec codes, and combinatorics
planning of external variants	no dedicated planning for variants; design of external variants driven by available technology, based on suggestions from engineering	no centralized technical documentation; early agreement on sales codes between engineering and sales to designate available equipment	early agreement on sales codes but no designation of variants during concept phase; instead, focus on prototypes	necessary variants for markets centrally documented; partially supported by forecast of take-rates	necessary variants for markets centrally documented; partially supported by forecast of take-rates
handover of specifications to project teams	no dedicated handover of requirements but based on individual engineer	managed by architecture group based on formal process	managed by architecture group based on formal process	managed by architecture group based on formal process (VMI logics)	managed by architecture group based on formal process (VMI logics)
common product structure for bill of material	historically grown structure of 2D drawings and bill of material	based on initial draft of MAN product structure and refined using passenger vehicle templates	documentation of BoM based on centrally managed product decomposition	documentation of BoM based on centrally managed product decomposition	documentation of BoM based on centrally managed product decomposition
CAD data structure aligned with bom	structure of bill of material and of drawings / CAD responsibilities not consistent	CAD data structure not aligned with product structure	CAD data structure oriented but not completely aligned with product decomposition	CAD data structure aligned with BoM for all new parts; common parts with existing portfolio (carry over parts) remain in historic, non-aligned structure	CAD data structure aligned with BoM based on product decomposition
planning of technical (component) variants	no centralized planning with regard to existing building blocks to obtain overall modular kit; driven by individual organizational units with interest to lower effort at each organizational unit	isolated efforts by engineering design disciplines, not integrated to form any given modules or building blocks within product structure	variants not centrally planned but planning process supported for core areas (e.g. seating) at the level of engineering teams; mapping to sales codes after concept design phase	centralized designation of variant drivers per component to deduce necessary variants and track fulfilment of variant targets	centralized designation of variant drivers per component to deduce necessary variants and track fulfilment of variant targets
planning of interfaces	central planning with large projects, but relativized by later adaptations and not controlled systematically (except for industry-wide standards)	no planning or designation of interfaces at total vehicle level	planning of vehicle cross sectional interfaces by engineering, represented in product decomposition	first concept of management of interfaces available, not implemented yet	management of selected main interfaces
concept decisions	no template, most decisions taken intuitively	no standardized decision process or template	standardized decision process including vehicle lifecycle aspects	standardized decision process including vehicle lifecycle aspects	standardized decision process including vehicle lifecycle aspects
reporting	no systematic tracking of requirements fulfillment, of design targets or of progress of design efforts during concept phase	?	reporting by project manager on progress of requirements fulfillment, design targets and progress of design efforts	reporting by project manager on progress of requirements fulfillment, design targets and progress of design efforts	reporting by project manager on progress of requirements fulfillment, design targets and progress of design efforts
process descriptoin	process template for project initialization and concept phase very little enforced and mostly based on optional milestones that were not supported by a specific methodology	central process description, agreed upon by project team but not finally enforced	central process description, refined where necessary with available methodology	refined company-wide process template incorporating a central product decomposition and the architecture process as a compulsatory part of the design process	refined company-wide process template incorporating a central product decomposition and the architecture process as a compulsatory part of the design process
tool support	decentralized documentation using spreadsheets and presentations, structured in folders on project drive	spreadsheet-based centralized tracking of information relevant to architecture process	spreadsheet-based centralized tracking of information relevant to architecture process	office tools including complex data base for change processes; variant contexts defined in variant tree authoring tools	PLM environment, coupled to variant authoring tools

Figure 5. Evolutionary steps of architecture implementation and tasks therein

As shown in Figure 4, architecture at the company consists of tasks ordered by "topics" along rows; besides the main topics, the organizational interfaces and the principles of architecture were made part of the framework to ensure both were integrated when building up each of the topics. The model of architecture behind the framework followed two strands of thought: A classic systems engineering one overlaid with a description of variants the underlying combinatorics. As a systems engineering model, the approach to first collect requirements, map them to functions, find solution principles for them, and embody them as components formed the main topics. These were paired with an early description of combinatorics (i.e. what equipment should be made available for a vehicle in combination with what other equipment) for requirements; to this end, "spec codes" were made available as a concept [Karrer-Müller et al. 2013], i.e. containers that each describe a potential equipment (and its properties) of a vehicle that could be selected or deselected by a customer (e.g. a spare wheel, a fuel tank volume, a sunroof) to represent the voice of the customer and the portfolio perspective in requirements management. For these spec codes, a forecast of take-rates was added, too, to ensure that in complex variant designs a focus on the most important variants would be possible.

3.2 Impact of the implementation of architecture onto the framework

During the implementation of architecture over a period of approximately three years, the initial architecture framework evolved to the current state that is shown in Figure 6. The evolutionary steps that have led to this result are listed in Figure 5; each column represents an implementation step that had (or had not) a certain impact on each topic, as listed in the rows. The evolutionary steps were mostly oriented on large vehicle projects that served as try-out grounds for the concept of architecture available when such a project started (i.e. the maturity of the process description, the available methods, and tools).
In short, the following aspects drove the evolution:

- Description of combinatorics: As a necessary basis for variant planning, the detailing, visualization and reviewing of combinatorics provides a basis for technical variant descriptions, e.g. as variant trees or tables
- Product structure: Initially, the company did not have a set product structure that was centrally managed; considerable effort was, therefore, put into this as a service that the architecture group provides to the company, and the tasks to manage it have impacted the architecture process, too.
- CAD data structures: As most engineers are still concerned with mechanical design, the way the product is decomposed / set up is reflected in how CAD data is structured. To make architecture come alive, therefore, a close alignment between CAD methodologies and the company's product structure was implemented.
- Concept decisions: It turned out that a formalized decision-making process was a well-accepted methodology early during the introduction of the architecture process. It was, in a second step, extended to include decision maps to plan for individual decisions.

3.3 The updated framework

The updated framework was built about three years after the initial one, and reflects the results and experiences made from the design projects that were supported in the meanwhile. It is shown in Figure 6. It contains 8 tasks with a total of 26 sub-tasks; 11 of these are identical to the initial framework, and most others have changed only slightly. The focus on functional modelling was dropped, although it is being taken up again by a different department in the company with a focus that is less on concept design and more on adding functional requirements. The focus on CAD data structures, although important to architecture, was not made part of the architecture framework, as it is only impacted by the architecture process but not a principal output thereof.

Topics					
Specbook, requirements, spec codes	support the identification of spec codes and combinatorics	ensure completeness and consistency of combinatorics	visualize external variants	support the identification of requirements	
verification of return-on-invest and production volumes	support the identification of planned return-on-invest and production volume per spec code	map return-on-invest and production volume to combinatorics	consolidate external variants (combinatorics) based on return-on-invest and production volume	align consolidated external variants with specbook	
assignment of specbook	allocate spec codes and requirements to product decomposition				
product structure	update and extend product decompsition where necessary				
modular kit vehicle	identify and analyse surrounding variants	allocate carry-over-parts to product decomposition	support modularization and definition of interfaces		
variants	deduce and provide necessary variants to design engineers	identify/manage variant drivers per component of product decomposition	align variant design with specbook	allocate concept solutions to components of product decomposition	plan take rates based on production volumes from specbook
concept solutions	support generation of concept solutions	enable review and assessment of concept solutions	allocate concept solutions to product decomposition	prepare decisions on concept solutions	prepare final codes
reporting / transparency	reporting on product decomposition	prepare and conduct release of product architecture	compare product architecture with initial specbook		
principles	generate transparency	foster communication	prepare decisions	manage complexity	

Figure 6. Current architecture framework

3.4 The value of architecture today

As part of the setup of the current framework, the initial value was reassessed to have a clearer picture of the value of architecture today. While the actual curve has evolved very little to today's depiction, as shown in Figure 7, the motif behind it has. Initially, the value was placed particularly on generating the right variants, i.e. the focus was put on planning the actual architecture. Based on the operational model introduced, with architects facilitating the process and managing the transfer of information across the whole design organization, the focus has shifted to generating transparency about the progress of concept design, rather than doing a part of the concept design per se. This can be recognized in the evolution of the framework, too, where tasks are designated more as "support" rather than "do". Above all, it shows in the change of principles, which have not yet reached a final state, and another step of its evolution is expected. At the same time, the principles (bottom row of Figure 6) reflect the actual value delivered by the architecture process: at any given point of the concept design

process, there is transparency on how many variants are described and how many are still missing, as well as what properties (weight, cost,…) they each have (principle 1). To obtain these variants, architects ensure that responsibilities are clear within a design project (Principle 2: Who designs what? Who takes care of what requirements? What information is available on that?) and they push decisions in a structured manner (principle 3: What needs to be decided? In what sequence? What has been decided already?). By this, the overall complexity is managed (principle 4).

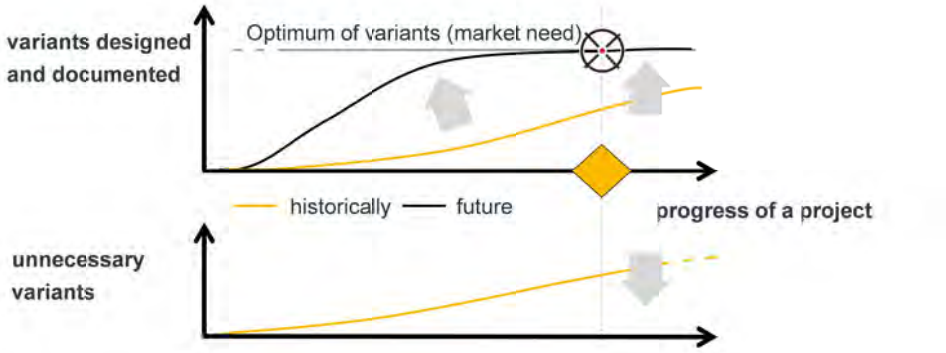

Figure 7. Value of architecture from today's point of view

As such a rather abstract description is hard to understand for many engineering staff, internally, the picture shown in Figure 7 is used, addressing four aspects:

- "We obtain **early knowledge** (transparency) about project results during the concept phase.": Based on a planned target on what variants are necessary (the upper graph in Figure 7, showing the "optimum of variants") and the monitoring of the progress of design, the actual gap as well as the fulfilment of the design targets given by the requirements can be analysed.
- "We obtain a well-reviewed and clear **task description** on what needs to be designed.": With the company's complex variant portfolio and the broad modular kit that vehicles are set up from (about 10^{46} different vehicles can currently be configured from a portfolio of about 100,000 building blocks), engineers find it hard to ensure they are doing what is expected of them; therefore, a clear task description that shows what variants and what variant contexts as well as the requirements to the design solution provide a well-received value to engineers.
- "We **reduce the synthesis of variants** that are likely not sold.": With the broad variant portfolio, often engineers design a solution to fit every possible variant context, even though the customer might not need it. Therefore, vehicle configurations that are never sold can occur and need to be avoided; this also prevents costly deletion processes.
- "We **focus the design effort** on the market needs.": With broad markets to be served, engineers need to be sure that they now about the requirements and the priorities they need to tailor their design efforts to; the architects ensure that the right information is available to the different engineers to help with this.

3.5 Using the framework in practice

In practice, the framework is mostly applied to review the progress of implementation based on a simplified CMMI scheme. This way, quality of implementation and progress description as well as the repeatability of the process can be easily monitored. For each topic, the (sub-)process, the templates including according data models, the responsibilities, available tools, training materials and examples are described. Instead of the five CMMI levels (see, e.g., [Chrissis et al. 2006]), three levels are used ("concept available", "prototypical process available", "process implemented").
The application and practise show that the framework yields the following advantages:

- It is easy to illustrate the scope of implementation by explaining the individual topics and tasks of the architecture process, as those are more concrete and closer to the rather abstract concept of product architecture management per se. Thus, especially discussions with new partners, e.g. new projects that are supported by the archtitecture process, are facilitated.

- For each topic, indicators about the progress were set up, as Figure 5 illustrates; in a current effort, these are being consolidated with the topics of the framework to better track the progress of implementation for each topic. While it would have been better to do this initially, many measurement criteria were not clear initially, as many boundary conditions in the company changed during the implementation period.
- The framework helps especially with regard to managing process improvement efforts within the company and with inviting tenders to gather external support for such bidding wars with different external consultants and software vendors, as the scope of a certain effort can be described well and consistently with the rest of the architecture process.

To facilitate establishing and embedding all this in the company, responsibilities within the architecture department are organized accordingly, with responsibilities designated per topic; internal documentation is organized in a similar manner, as is the company-internal training portal. To ensure the consistency of all topics and related concepts, a process responsibility exists in parallel to draw upon the topics and build the actual design process. In a design project, a product architect then draws upon this process description that is set up from the different topics; for complex design problems, the responsible persons for the method in focus (e.g. concept decisions) serves as a coach for the architect in a project. This way, the experience is transferred into better-suited methods that remain aligned with the needs of the design process.

4. Conclusion

Realistically, it must be said that the introduction of architecture is not yet finished; about another two years will be necessary to complete all tools, templates and training materials in a way that they are simple and clear enough to be used by all engineers working in the concept design phase. This can be read from the maturity assigned to each task; especially the tasks that relate to the verification of the return-on-investment are still at a conceptual level, while e.g. the management of the specbook and the product structure have already been implemented in the company's PLM environment and, thus, the related processes are considered "implemented".

Similarly, staff at the company finds the explanations and illustrations of architecture still abstract and hard to access; this is, in part, due to the workload of their daily business, and partly because their formal training has given them little chance to build a more abstract understanding of management- and process-related issues. Therefore, more communication is being focused on this in the future, especially to obtain more examples and use them as individual illustrations specific to the different target groups.

However, the transition from historically grown product documentation to a clear and consistent product structure turns out to be a very complex undertaking, especially considering the complex interdependencies between components across the whole vehicle portfolio. Therefore, a stable basis of product documentation will be a next step before future aspects of architecture (such as specific vehicle standards) can be systematically introduced.

A factor that helped during the implementation was that all structures introduced (the product structure, the requriements structure,…) were colored. This today helps engineers speak one language: when people talk of "yellow processes", they refer to everything related to requirements management and the specbook, as the requriements structure is the "yellow structure". This also reflects in the framework, which uses the same categories.

The architecture process today, in its essence, is a focused systems engineering approach including variant design; the management of modularity across different design project has not impacted the architecture framework yet and will provide for a future evolution when the above three aspects have been implemented fully.

References

Chrissis, M., Konrad, M., Shrum, S., "CMMI", Addison-Wesley, Amsterdam 2006.

Friedel, A., "Investigating the Management of Uncertainty in Product Platform Lifecycles", Master Thesis, MIT Cambridge, MA, 2011.

Gustavsson, H., Eklund, U., "Architecting Automotive Product Lines: Industrial Practice", Science of Computer Programming, Vol. 78, No. 12, 2013, pp. 2347–2359.
Halbhuber, T., "Komplexitätsbeherrschung in der Konfiguration von der Entwicklung bis zur Produktion", InMediasP Workshop Variantenmanagement, Stuttgart, 14.3.2012.
Jose, A., Tollenaere, M., "Modular and platform methods for product family design - literature analysis", Journal of Intelligent Manufacturing, Vol. 16, No. 3, 2005, pp. 371–390.
Karrer-Müller, E., Deubzer, F., Kreimeyer, M., "Lastenheftabbarbeitung in den frühen Phasen der Entwicklung", VDI-Berichte Nr. 2186, VDI, Düsseldorf, 2013, pp. 285-294.
Kreimeyer, M., Kindsmiller, H., Landsherr, T., Heintze, W., "Systematische Planung der Produktarchitektur von Nutzfahrzeugen in den frühen Phasen der Entwicklung", VDI Nutzfahrzeuge Steyr, 2011.
Matthes, D., "Enterprise Architecture Frameworks Kompendium", Springer, Berlin 2011.
Robertson, D., Ulrich, K., "Planning for product platforms", Sloan Management Review, Vol. 39, No. 4, 1998, pp. 19–31.
Schuh, G., Arnosch, J., Nußbaum, C., "Produktarchitekturen richtig gestalten: Ein Weg zum variantenoptimierten Produktprogramm", Industrie Management, Vol. 23 , No. 6, 2007.
Simpson, T., Maier, J., Mistree, F., "Product platform design: method and application", Research in Engineering Design, Vol. 13, No. 1, 2001, pp. 2–22.
Ulrich, K., "The role of product architecture in the manufacturing firm", Research Policy, Vol. 24, No. 3, 1995, pp. 419–441.
Ulrich, K., Eppinger, S., "Product design and development", McGraw-Hill/Irwin New York, NY, 2012.
Ulrich, K., Tang, K., "Fundamentals of product modularity", Issues in Design/Manufacture Integration, DE-39, 1991, pp. 73–79.
VW, (online), Available at: <http://www.vwvortex.com/features/technical-features/vwvortex-feature-a-first-look-at-the-new-golf-7-mqb-architecture/>, Last viewed on 8.12.2013, 2013a.
VW, (online), Available at: <http://www.vwvortex.com/features/technical-features/golf-7-technicalpreview/>, Last viewed on 8.12.2013, 2013b

Dr. Matthias Kreimeyer, Product Architect
MAN Truck & Bus AG
Dachauer Str. 667, 80995 Munich, Germany
Telephone: +49 89 1580 63958
Email: matthias.kreimeyer@man.eu