INTERNATIONAL DESIGN CONFERENCE - DESIGN 2006
Dubrovnik - Croatia, May 15 - 18, 2006.

DESIGN 2006

PRODUCT STRUCTURING FOR CROSS-*X* PDM

M. Vielhaber, H. Burr and M. Eigner

Keywords: distributed engineering, product structure, engineering objects, product data management, product lifecycle management

1. Introduction

"Beginning a discussion on how to structure the bill of material is a good way to start a fight in a bar" [Garwood 1997]. Yet solving this issue is an essential milestone on the way towards efficient IT support for today's and tomorrow's design and engineering processes.

In line with the demand for shorter times to market, automobile manufacturers are forced to cut lead-times to the quick, while, at the same time, continually increasing the complexity of their products. This trend is driving new concepts and strategies for product and process development. Challenges arise from consistently condensed and interlinked processes in cross-domain, cross-business unit and cross-enterprise (i.e. cross-x) engineering networks. And management of product data and the organisation and control of information flows both play a pivotal role.

One major building block in any strategy to cope with these challenges is hence to develop a powerful, efficient and flexible data management backbone for all IT applications supporting the product creation process. Trends are clearly pointing towards a modular product data management (PDM) concept based on application-geared application data management (ADM) components interlinked by a streamlined and standardised communication backbone. The buzzword of service-oriented architecture (SOA) is penetrating the world of engineering IT.

With the IT implementation, however, being the easier part of such a solution, the development of a holistic product structuring concept across the different domains, business units and companies seems to be the greater challenge. Application-specific data models and structuring preferences have to be integrated, configuration and variant management requirements have to be considered, and highly unstable system concepts of IT suppliers have to be adapted and consolidated.

This paper presents and discusses steps towards such a product structuring concept, which can serve as a methodical foundation for future engineering processes and for a future cross-x PDM system concept. Next, chapter 2 provides an overview of the status quo in automotive engineering with chapter 3 introducing a solution concept for product structuring in a cross-x PDM environment. In this context, engineering objects will be introduced as universal carriers of engineering information. Finally, chapter 4 gives an outlook at relevant future work in research and practical application.

2. The Status Quo

The background for the research topic of this paper is spanned by the current situation and trends in automotive engineering as well as by current and upcoming methods and system concepts for product data management and product structuring.

2.1 Trends in automotive engineering

2.1.1 Cross-x complexity

The product creation process in the automotive industry is complex. Not only is this true for the products themselves, i.e., the automobiles, which are continually growing in complexity to meet market demands: carmakers are also confronted with a need to integrate a soaring number of electrical, electronic, and software components in what used to be a chiefly mechanical system landscape. In addition, an explosion in the number of variants per model due to the trend toward mass customisation, which strives to tailor-make stock cars to specification, continues to put pressure on the industry; see figure 1.

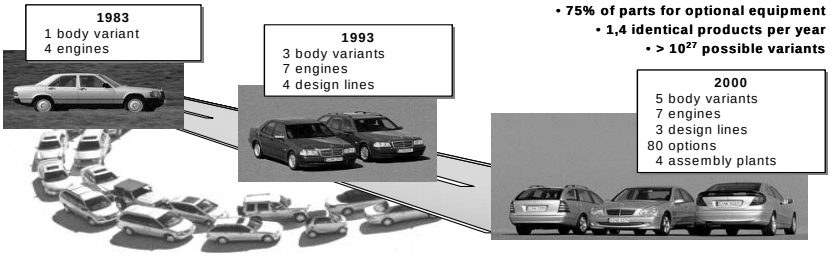


Figure 1. Growing complexity in automotive engineering [Burr 2003]

On the other hand, there are a multitude of organisational aspects that significantly impact product development. For example, experts across a variety of domains both within a wide range of intra-company departments and from external organisations are typically involved in the development of a single passenger car model. Thus the effective and efficient exchange of information between all these people and the systems involved in the overall process is paramount. And the requirements and needs of the individuals making up this complex need to be transparent in order to prevent a loss of information, duplicated tasks, and redundant data.

In state-of-the-art development, product and process activities are now geared to run largely in parallel, striving to reap the benefits seen in simultaneous or concurrent engineering. The call for minimised development times makes it imperative that tasks be structured efficiently and the workflow coordinated effectively. This therefore drives a continuing process of change, which affects the process landscape. In addition, the endeavour to cost- and time-optimise the iteration loops occurring during product creation is pushing the general vogue towards front-loaded development, i.e., shifting of development steps to the early phases of the product development process.

Apart from these cross-domain aspects, collaboration in flexible and fast-changing cross-business unit and cross-enterprise engineering networks is becoming the standard for future engineering. Large parts of the engineering work are outsourced to engineering suppliers, putting the focus on inter-company communication and exchange.

2.1.2 Contributions of engineering IT

To cope with these challenges, engineering IT systems for computer-aided anything (CAx) are spreading throughout the product creation process, continuously increasing in functionality and covering areas formerly executed without dedicated system support. Yet, taking advantage of these enabling tools has led to rocketing heterogeneity of IT systems in the automotive industry: the system landscape now often resembles a jumble of stepping stones leading through the development path. While these systems attempt to provide optimum support in the areas they are tailor-made for, due to the high degree of specialisation necessary they tend to represent only partial solutions. In fact, many of these tools are proprietary in nature or, if bought off-the-shelf, have been customised to a large extent, thus generally necessitating a great deal of effort for maintenance and extension. What is more, data exchange between these islands is difficult if not sometimes impossible to achieve. The obvious remedy to this problem is a move to integrated system concepts. In any such concept, product data management plays the key role on the way to a sustainable solution.

2.2 Product data management

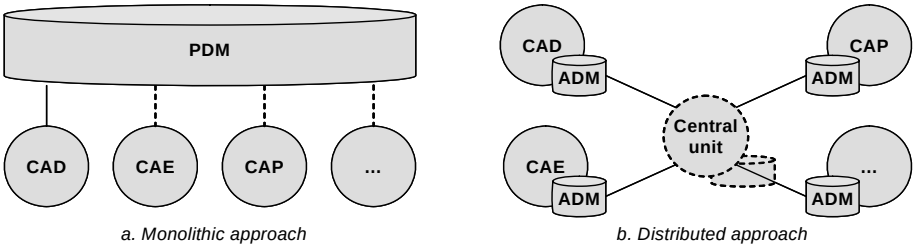
One major building block in any strategy to cope with the described challenges is to develop a powerful, efficient, and flexible data management backbone for all IT applications supporting the product creation process. Based on the growing complexity of the engineering process, the requirements set for product data management are, however, growing as well. Both the quantity and quality of the data to be managed are increasing.

PDM systems started in the 1980s as simple computer-aided design (CAD) file managers, replacing file- and map-based data storage by managing multi-user access, versions, and simple configurations of designs. And even today, PDM systems remain largely part-oriented, thereby discarding important assembly relations and other non-physical information.

On the system side, PDM systems were implemented as single databases with a proprietary interface to the geometry-supplying CAD system; see figure 2a.

Over the years, the CAD systems and the data to be managed have become more and more sophisticated. Nowadays, not only items - mostly single parts - and their locations have to be stored: to support a more assembly-oriented design process, CAD systems now build complete multi-level assemblies including various kinds of links between the components involved. In addition to simple parent/child links in the assembly structure, it is especially inter-part links as created during contextual design and assembly information such as assembly connections or assembly-level fits and tolerances that pose high demands on data management.

Apart from the CAD systems, other CAx applications such as CAE for computer-aided engineering and simulation and CAP for computer-aided planning have started to generate a multitude of data, which organisations try to manage by extending the grown monolithic PDM applications.



a. Monolithic approach *b. Distributed approach*

Figure 2. Approaches to product data management

Though driven by the trends described, the monolithic approach to product data management as depicted in figure 2a seems to have reached its limits. The single data backbone, which attempts to manage the ever growing quantity and quality of data, has become sluggish, inefficient, and no longer capable of supporting the managed applications in a satisfactory manner. Thus, a trend towards more modular data management concepts can be observed; see figure 2b.

Here, each application area is supported by a local data management system, often referred to as team data management (TDM) or, as in the figure, as application data management (ADM) system. However, since the various application areas have to exchange the data they create (e.g., product data generated by CAD are needed in virtually any follow-up application), these local ADM systems have to be interlinked.

For this purpose, different solutions are conceivable. One approach would be merely a communication network, built either from bilateral interfaces or, in a more standardised form, by a so-called enterprise service bus. Another approach would be a central data management unit, that controls and coordinates the communication between the local ADM systems and, through a central database, potentially collects and stores all the data generated and needed by the applications connected. Carrying this to an extreme, the central unit can even be blown up to the size and complexity of the monolithic PDM backbone as shown in figure 2a, again. It adds the approach of local ADM systems, whose size and content, however, largely depend on the role chosen for the central data management unit.

To summarise, both figures can be interpreted as extremes of a similar system approach: a backbone built from a central coordination unit and a communication network, both of which may vary in size

and content, and local elements consisting of optional local ADM solutions and dedicated application or ADM specific adapters or interfaces.
Current developments in automotive engineering clearly point towards such a modular product data management concept. One popular IT-concept to achieve this modularisation is the service-oriented architecture (SOA), which is based on the granularisation of system functions into (web) services communicating through standardised protocols (e.g. XML, PLM services). Using this concept of an enterprise-wide PDM backbone and application-specific local ADM solutions, cross discipline/ domain, cross-business unit, and cross-enterprise engineering can be established; see figure 3.

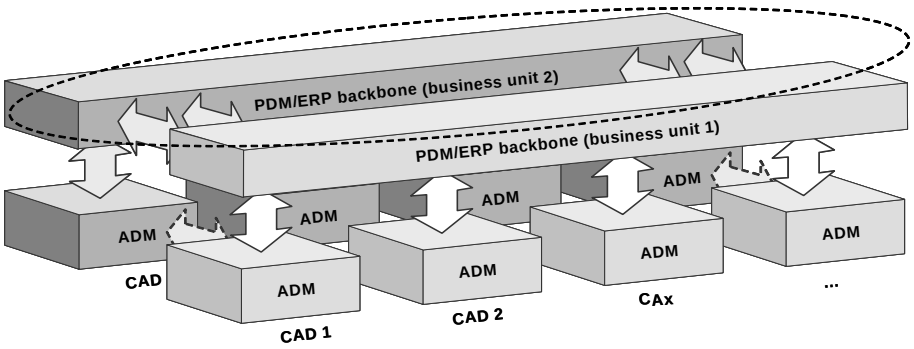


Figure 3. Cross-x product data management [Vielhaber 2005]

The remaining issue to be addressed is that of the role and character, size and content of the central unit. These questions are not easily answered. To approach a solution, further investigations are necessary. A basic rule defines the backbone as "as strong as required, but as lean as possible". Potential contents are set out below:

- User management to allow for consistent access rights across all application domains
- A number generator to allow for enterprise-wide unique and unambiguous numbering
- A directory service to reference data locations across all distributed databases
- A workflow engine for coordinated release management
- Configuration management functions
- A central instance of a reference product structure.

All these points have to be addressed before a distributed product data management concept can be put into practice. In this context, this paper focuses on a product structuring concept that is not merely suitable for the needs of automotive engineering but can also be implemented in a system layout as described. To be determined is which parts of the product structure will reside on the individual data management layers and what each of the system components should look like.

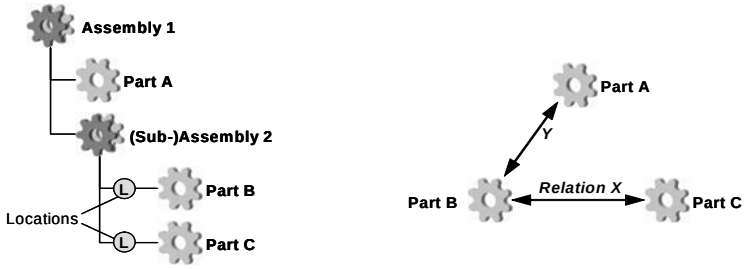
2.3 Product structuring

Product structuring is one of the most crucial factors in organising the development and production of technical products. On the one end, product structures are the foundation for designers to organise the results of their work, whereas, on the other end, they drive logistics, representing the basis for the building of the real products.
Product structuring as referred to in this paper consists of two main parts. First, it determines the types of objects that form the building blocks for the product structure. Second, it describes the way these objects are arranged to form the final structure.

2.3.1 Product structure objects

Both the engineering practices in place today and state-of-the-art engineering IT systems deal with parts as their central data building blocks, each of them describing one single part as a whole.
One level higher than part objects, assembly objects describe the joining of components to an assembly. These components can be single parts or assemblies themselves. This enables hierarchical,

multi-level assembly structures to be built. Suppliers of engineering IT systems tend to realise part and assembly objects using one common object type.
On a level lower than part objects, features form the basic building blocks from which the parts are put together. Another level lower, features are collections of basic geometries.
Today's PDM systems generally deal with structures from the part level upwards: beyond the assembly objects they may feature further structuring objects to accomplish this. They do not manage information below the part level, which is the main domain of the creating CAD systems. Hence features are currently not managed in PDM systems as such, but just as content of the single part objects.
Other information not managed as such in PDM systems is assembly information. Assembly information is used in the context of this paper as a term for all the information describing an assembly beyond the pure parent/child structure, e.g., assembly connections, assembly-level dimensions and tolerances, assembly features, or constraints. Assembly information is what makes a product more than merely a bunch of parts; yet it is today not managed adequately - often it is even not managed at all. It is therefore crucial for a future product structuring concept to consider assembly information to the same extent as part information, or even beyond.



a. Hierarchical approach b. Relational approach

Figure 4. Product structuring approaches [Vielhaber 2005]

2.3.2 Structuring concepts

As described, current PDM systems mainly feature part and assembly hierarchies to build product structures. Figure 4a portrays such an approach. In addition to parent/child relationships, this structure may also contain transformation matrices to describe the location of each part relative to the reference location of the one level-up assembly and thus absolute to the other parts within the assembly.
A further way of describing assemblies does not focus on hierarchical parent/child relations, but uses direct inter-part relations instead. This approach is referred to as relational product structuring; it is shown in figure 4b. However, current engineering IT systems do not support such an approach. Even in the case of geometrical constraints, which describe the relative location of parts by using such relations, the storage of this information is done by transforming it to absolute location matrices, which again are stored within the hierarchical structure, as depicted in figure 4a.
Both structuring concepts do not sufficiently allow for the management of assembly information, as set out above. A future product structuring concept will have to take this shortcoming into account.

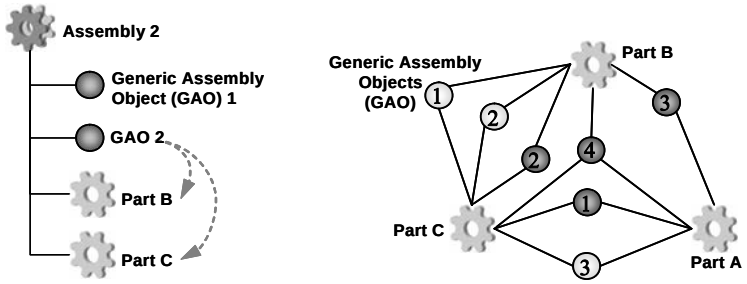
3. Product structuring in a cross-x PDM environment

In the following, a concept is described to bring the methodical requirements for product structuring within the automotive engineering process together with the upcoming cross-x system environment for product data management.
This is done in three steps. First, enhancements to the described building blocks of an application- and data management-spanning data model are presented. Second, a shell model is introduced to allow real cross-domain concurrent engineering on the same engineering objects. Third, a system of parallel product structures distributed on the different components of the system layout is described.

A further chapter indicates how configuration and variant management can be established within such a distributed data management environment.

3.1 Building blocks for a data model

In a first step, the data model for the relevant applications and data management systems has to be enhanced to be able to also optimally handle assembly-relational information. To achieve this, [Vielhaber 2004] introduced generic assembly objects (GAO) as universal carriers of assembly information. These information objects are managed on the same level and in the same way as the geometry-describing part objects. They contain relation-specific information together with references to the part objects they put into relation.



a. Hierarchical approach *b. Relational approach*

Figure 5. Product structuring with generic assembly objects [Vielhaber 2005]

With this concept, both hierarchical (figure 5a) and relational (figure 5b) product structures could be empowered to be capable of representing complete product descriptions consisting of both part and assembly information. A relational product structure enhanced by generic assembly objects leads to a structural network in which the components are interlinked by all different kinds of assembly information objects, e.g., geometrical constraints or assembly connections. Within such a structural network, the full information - even of different product structure views - can be stored. It can thereby serve as a neutral product structure from which applications may derive application-specific structures or views merely by filtering and using the generic assembly objects that form their relevant structure. The hierarchical structure in figure 5a can serve in a similar way, as it also contains the complete information, simply arranging it in a special, derived hierarchy.

This enhanced data model has to become a common basis for both the relevant application and data management systems. Figure 6 demonstrates this taking the example of a weld design in both CAD and PDM (including ADM) applications. Part objects, generic assembly objects, and assemblies form the common data model foundation for both system sides. With this, a foundation is given for a close integration and for full data management support for the product-defining CAD area.

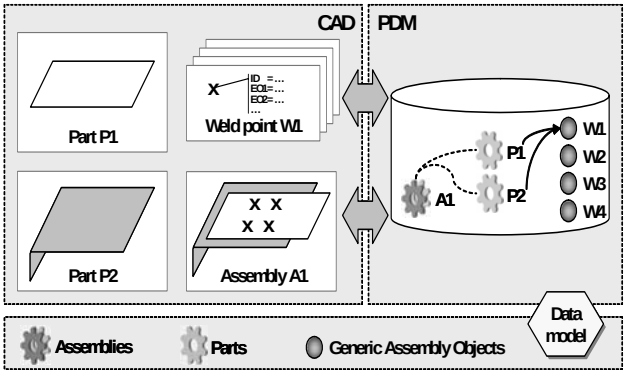


Figure 6. Integrated CAD/PDM data model [Vielhaber 2005]

As pointed out, many of today's PDM systems feature a common data object for the mapping of both part and assembly objects. Enhancing this philosophy by including all kinds of engineering information higher and lower than the part level, from geometry and features to parts, generic assembly objects and assemblies, would lead to one universal engineering data object type. Such objects are proposed to be called engineering objects (EO) [Zimmermann 2005].
The method of engineering objects can be used as common, federated and integrated "super-view" on a semantic level of information modelling of engineering data across the full product lifecycle. From the physical (and pragmatically) viewpoint the data can be stored in different and distributed applications, fileservers and data vaults. The repository contains the relevant meta data and is able to hyperlink different physical data structures and applications.
An engineering object is defined by:

- Identifier, name, administrative/logistic information
- User (observer, that means the designer, engineer, customer, etc.)
- The purpose/the intended usage of this Engineering object; this kind of information depends on the point of view of the user/observer
- *"Description of EO"*=: {set of properties}, from the actors/observers/users point of view
- *"Representation"* (that may be a NURBS CAD model, or also screen format)
- *"Methods"* = functions and constrains related to the EO, e.g. insert, delete, copy

In short notation: *EO*=: | *Representation (a, b, c, …), Properties (α, β, χ, ...), Methods (1, 2, 3, …)* |.

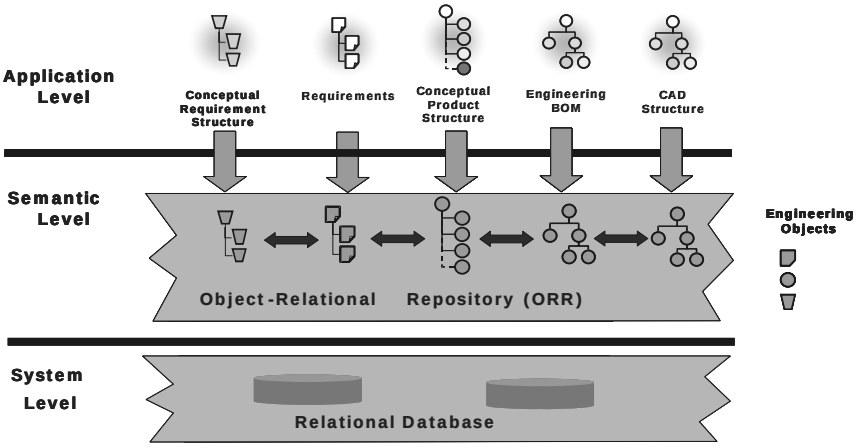


Figure 7. Transferring engineering objects onto an object relational repository

Using this strategy many different partial models (e.g. requirement structure, conceptual product structure and CAD models) are transferred to an object-relational repository. There the dependencies between the structures are mapped and maintained; see figure 7 [Eigner 2005].
Hence the concept of engineering objects is well suited to build up a lean PDM/ERP backbone as a virtual container and integrator of all the different and domain-specific information across the product lifecycle.

3.2 Shell model for domain-spanning data

Concurrent engineering is one of the most prominent paradigms in automotive engineering, as it is a key to shorter development times and hence shorter times to market. In a distributed data management environment, it has to be possible for different domains to work together collaboratively and in parallel on same data objects.
To achieve this, a shell model for the management of domain-spanning product information was developed. The model allows information to be subsequently added to the same objects from different domains, so that, along the different domains involved, the maturity of the product is increased to its

final state. [Burr 2006] discusses the shell model to a deeper extent and using the example of CAD- and CAP-spanning product and process data.

3.3 Structure system

Based on the introduction of generic assembly objects and engineering objects as well as of the shell model for domain-spanning information, a concept for the use of product structures within the distributed data management environment is now presented. This concept builds on a structure system consisting of application-specific working structures and one central reference master structure; see figure 8.

With each application being supported by its own ADM system for managing the overall quantity and quality of application data, the working structures reside in just these ADM systems. The user has the necessary freedom for structuring this structure as desired, and for inputting all the information that is required to best support the application-specific work. The only thing to be assured on the ADM side is that the contents of this working structure that may be relevant to be shared with any downstream domain are published to the outside. For the CAD domain example, this refers to geometry information within part objects and to assembly information within generic assembly objects.

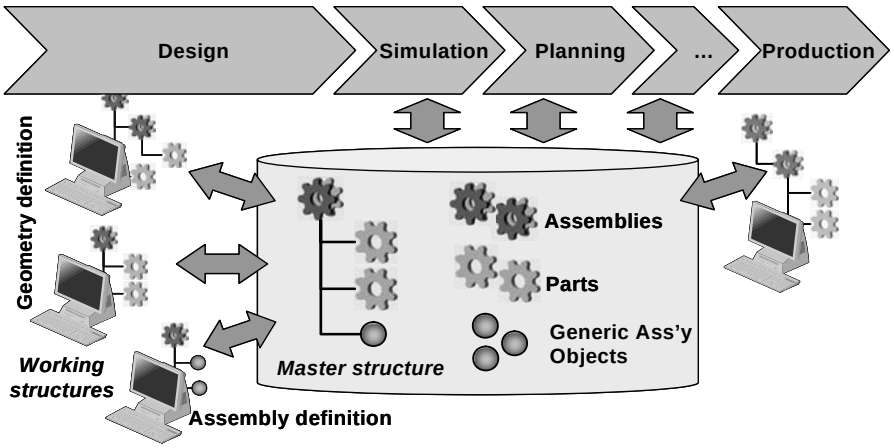


Figure 8. PDM structure system

The central reference master structure resides within the central unit introduced in figure 2b or, more precisely, within the PDM backbone system. As long as it is provided with complete information in such a way that all information concerning more than one application domain is included in an exchangeable format, its exact (hierarchical) structure is only secondary, as was explained in the discussion above.

The authors therefore propose using just a flat, one-level structure, that, if also filled with complete relational assembly information, is equivalent to a structural network as shown in figure 5b. If domain-spanning information is required to be modelled according to the shell model, this model structure will also have to be considered within the reference structure.

The communication between the ADM and the backbone system then has to assure that all information objects required from one ADM to another are transferred to the backbone system and kept consistent. This refers, first of all, to the meta objects on both the part and assembly levels and secondly to the attached data objects in a format defined for data exchange. This format is generally not a native format, instead being a lean and standardised exchange format such as JT for geometry data.

Any follow-up application domain requiring information from the backbone system has to subscribe to the desired information objects and incorporate them into its own defined product structure. Following this concept, the discussion set out at the beginning of this paper about the “right” product

structure is by-passed, with each application domain enabled to select its optimal application-specific structure.
For the IT realisation of such a communication, various kinds of web services have proven to be adequate solutions. For this purpose, it again seems desirable to choose a standardised communication format, e.g., STEP PLM services [Feltes 2005].

3.4 Configuration in a distributed PDM environment

With the huge number of product versions and variants to be handled in automotive engineering, configuration management is one of the most crucial aspects for automotive product data management. Many data management concepts have proven to work fine in unconfigured environments, while failing to work out for complex configured products such as automobiles. For the concept developed in this paper, it has therefore to be decided in which of the different data management layers configuration should occur.

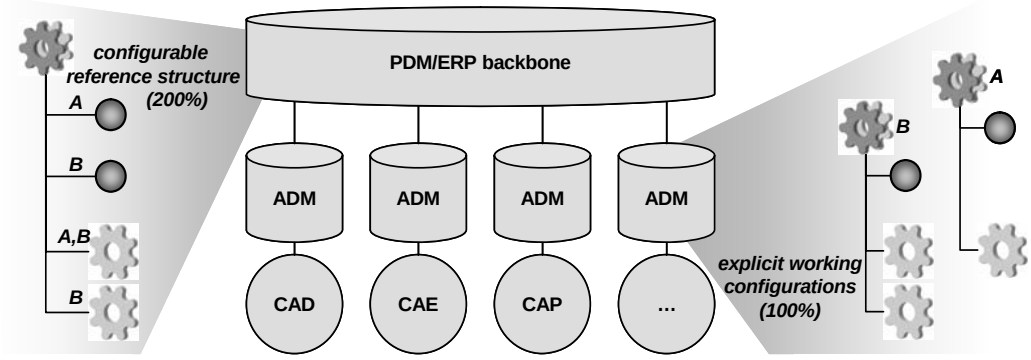


Figure 9. Configuration in a distributed PDM environment

At the latest, configuration is called for when it comes to the final costumer-specific product instance: the single personalised car. It is therefore clear that ERP systems applied in logistics and production have to be the final configuration masters.
As these ERP systems are provided with product data from the product data master, the reference structure within the backbone system, it also makes sense to mirror the configuration from the ERP side to the PDM backbone side, as well. Configuration is therefore a strong argument for automotive manufactures to move the PDM backbone functionality close to the ERP side, or even to integrate these two system blocks, as shown schematically in figure 9.
At the other end, application systems tend to deal with more or less explicit, buildable car configurations. For example, CAD designers generally design one (or a few similar) configurations at a time, digital mock-ups are made for single configurations, and crash simulations are performed for buildable cars, also. It therefore seems logical to keep configuration within the ADM systems to a minimum or, if applicable, to avoid configuration within the ADM area at all. The other option would be to synchronise configuration mechanisms between the PDM/ERP backbone side and the ADM side. This would be quite a complex exercise, as the ADM system is generally designed by the application's system supplier, whereas the backbone system is designed by the ERP or another neutral supplier.
Figure 9 summarises the system and configuration layout presented.

4. Summary and Conclusions

This paper introduced a concept for product structuring in a heterogeneous cross-domain, cross-enterprise product data management environment. The concept presented was laid out according to the requirements of automotive manufacturers and their complex, configuration-intensive products. It results in a powerful, flexible, and adaptable PDM layout which empowers engineering methods to optimally support the product creation process.

Future research work will still have to be done to finalise details of the concept, e.g. the application and data management spanning data modeling based on engineering objects, or the shell model for domain-spanning information. Standardisation activities in the relevant areas have to be supported.
Implementation work will have to be done especially on the PDM systems' side, whose suppliers are currently not yet ready to sufficiently support the requirements of future engineering processes.

References

Burr, H., Deubel, T., Vielhaber, M., Haasis, S., Weber, C., "Challenges for CAx and EDM in an international automotive company", proceedings of the ICED 03 - International Conference on Engineering Design, The Design Society, Stockholm, 2003, pp. 309-310 (executive summary), paper no. 1506 (CD-Rom).

Burr, H., Vielhaber, M., Weber, C., "Information management for the digital factory - bridging the gap between engineering design and digital planning", proceedings of the DESIGN 2006 - International Design Conference, The Design Society, Dubrovnik, 2006.

Eigner, M., Zagel, M., Weidlich, R., "The Conceptual product structure as backbone of the early product development process", proceedings of the ProSTEP iViP Science Days 2005, Berlin, 2005, pp. 62-71.

Feltes, M., Lämmer, L., Vettermann, S., "PLM services - a new OMG standard to push collaborative engineering", proceedings of ICE 2005 - 11th International Conference on Concurrent Enterprising, München, 2005.

Garwood, D., "Bills of material - structured for excellence", 6th edition, Dogwood Publishing Company Marietta, 1997.

Vielhaber, M., Burr, H., Deubel, T., Weber, C., Haasis, S., "Assembly oriented design in automotive engineering", proceedings of DESIGN 2004 - International Design Conference, The Design Society, Dubrovnik, 2004, pp. 539-546.

Vielhaber, M., "Assembly oriented design", PhD thesis, Saarland University, Saarbrücken, 2005.

Zimmermann, J. U., "Informational integration of product development software in the automotive industry - The ULEO approach", PhD thesis, University of Twente, 2005.

Dr. Michael Vielhaber
DaimlerChrysler AG
Research - IT for Engineering
P.O.-Box 2360, 89013 Ulm, Germany
Tel.: +49-731-505 4815
Fax.: +49-731-505 4400
Email: michael.vielhaber@daimlerchrysler.com