Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 17 (2014) 338 – 344

www.elsevier.com/locate/procedia

Variety Management in Manufacturing. Proceedings of the 47th CIRP Conference on Manufacturing Systems

# Managing product and production variety – a language workbench approach

Amir Hossein Ebrahimi[a*], Pierre E. C. Johansson[b], Kristofer Bengtsson[c], Knut Åkesson[a]

*"aDepartment of Signals and Systems, Chalmers University of Technology, SE-41296 Göteborg, Sweden"*
*"bVolvo Group Truck Operations, SE-40508 Göteborg, Sweden"*
*"cSekvensa AB, SE-44351 Lerum, Sweden"*

* Corresponding author. Tel.: "+46 31 772 17 54" ; fax: +46 31 772 17 54. *E-mail address:* amir.ebrahimi@chalmers.se

**Abstract**

Product platforms are commonly used in industries with complex products and high competition like the car and truck industry to allow a customer to order a product that satisfy its unique needs. A consequence of product variety is that manufacturing and assembly processes need to deal with this variety as well. If the variety is low and changes of the product occur infrequently then the variety may be handled by designing the production system for a small set of typical products. But as the variety increases and changes become frequent the necessity for integrated product and production information model is high, to partially solve this problem Product Life Cycle Management (PLM) systems aim at providing an integrated model to all categories of users, e.g. product designers, product preparation engineers, line builders and shop-floor workers. All users need to access the information in the platform and refine and modify the information to reflect new knowledge that has been acquired. Today, most often multiple systems are used where some systems may store information in a structured way but often unstructured text documents are also used. This easily results in redundant information models and automated analysis is not feasible or not event possible because of issues regarding cohesion and traceability of information. The contribution in this paper is to discuss how a new type of tool for building domain specific languages and editors using language workbench approach can be used to support the different user categories in their tasks working with variability of a product and production system while at the same time provide cohesion and traceability of information.


Selection and peer-review under responsibility of the International Scientific Committee of "The 47th CIRP Conference on Manufacturing Systems" in the person of the Conference Chair Professor Hoda ElMaraghy"

"Keywords: product life cycle management, mass customization, product platform, process platform, knowledge managment, variability"

## 1. Introduction

Product Life Cycle Management (PLM) [1] strategies aim to integrate information throughout a products life cycle from the imagination of the product to the design and realization of the product and information about the use and recycling of the product. A PLM system supports everyone in the organization, from purchasing, to product designers, to production preparation engineers, to shop-floor workers. In [2] requirements for PLM systems, in a workshop hosted by NIST (National Institute of Standards and Technology), are presented. The notion of *cohesion* and *traceability* of


Selection and peer-review under responsibility of the International Scientific Committee of "The 47th CIRP Conference on Manufacturing Systems" in the person of the Conference Chair Professor Hoda ElMaraghy"
doi:10.1016/j.procir.2014.01.100

information is reported as being crucial for improved data management of PLM systems.

This paper is part of an ongoing EU FP7 project Know4Car [*] which aims at developing a knowledge-based collaborative platform of the design and deployment of manufacturing systems [3]. In this paper we discuss how a new type of tool called *Language Workbenches* can support cohesion and traceability of information in such a knowledge based system. We specifically focus on managing variety of the product and the production system.

Handling variability within the automotive industry [4] is a vital aspect. In today's competitive market the diversity in customer's need have resulted in a high level of variability in the products which have to be manufactured. As a result in the manufacturing industry there has been a shift from single products to product families and product platforms [5]. Within each product family there is a high degree of variability. Hence the type of information which the manufacturing industry has to work with is tightly coupled with variability. This variability has affected both the design of the product and the production process. It can be said that variability has affected the whole process platform [6], where the term process platform is used as a conceptual structure of producing a family of customer specific products.

When ordering a car or a truck, a customer might have it customized to meet the individual needs. While the car industry typically has a well-defined set of options that the customer can choose from, a truck is used in a wide range of environments and used to carry different type of loads. Consequently individual trucks from the same product family might have very different physical characteristics. Product platforms define a set of components and configuration rules that model how these components can be combined together accordingly. The product variability has consequences for the manufacturing and assembly processes [7], because they have to adapt to the characteristics of each unique order. If product variability is low, the production process might be adapted to solve each possible variant. However, in this paper products with high variability are considered, where it is not feasible to exhaustively enumerate all possible variants. This is typically the situation in the automotive and truck industry.

Building a new or modifying an existing manufacturing or assembly line is a costly and time-consuming process that needs to be supported by efficient tools. Various user roles like product designers, production preparation engineers, line builders and shop floor workers need support and tools to share knowledge in an efficient manner.

Variability during product design and the design and operation of the production systems influences many stakeholders. Product designers define and develop engineering solutions for product platforms and specify requirements and constraints in the platform e.g. allowed combinations of the components in the platform. Production preparation engineers together with line builders have to consider all possible product configurations that might be ordered by a customer, and need to develop a production system that is able to handle all possible variants of the product. The variability has consequences for both the distribution of operations between assembly cells, but also on the distribution of operations between shop-floor workers in the same cell. The variability of the product platform increases the complexity of analyzing the consequences of new or added information for the production system.

In [2] cohesion is defined to be knowledge of the interrelationships that exist between data and traceability to be the knowledge of origin or basis for believing in certain data. Different user categories that work with the information and knowledge for the product and production system prefer to represent the information in different ways, for example, the product designers and preparation engineers prefer to represent the parts and subassemblies in different ways. This difference in representation is because for the product design the focus is on the structure of the product while the assembly order is of importance to the product preparation engineer. Since the design and development of a product and the production systems will go through several cycles it is time consuming and error-prone to have multiple representation of the same information in the system. In [2] the properties, *associatively across views* and *logical consistency* are identified as two important properties of systems supporting cohesion and traceability. Associatively across views means the absence of knowledge that two conceptualization are used for the same purpose and also the absence of knowledge that two references refer to the same thing. Logical consistency is defined to be *type awareness*, *interpretation constraints* and *wellformedness conditions of the information*. Another important property is the traceability of the origin of belief, where an assumption that was true at some point might no longer be true and thus invalidating certain decisions that were made previously. Problems regarding cohesion and traceability of information will result in an inefficient process working with the information in the product and production system.

Domain specific languages (DSLs) [8, 9] are for end-users of a system, who are supposed to be domain experts but not experts on computer languages. DSLs might be textual or graphical or a hybrid of the two. DSLs support a concise and domain-specific notation for working with information within the domain. DSLs also benefit from many of advantages with general purpose programming languages as having a well-defined syntax and semantics. This allows DSLs to support the automated analysis and transformation of data. Importantly, DSLs provide extensive support for handling the conceptual and associative gap that was crucial in PLM systems, but also for dealing with logical consistency of the data. This is because in a DSL it is straightforward to make a distinction between a types and instances of parts. Also a DSL can easily enforce types and logical constraints that guarantee that the information is consistent and does not contain contradictions. In addition DSLs excel at modelling associations between information something which is useful for supporting traceability of information and decisions.

Language workbenches [8, 10] are tools that support development and use of DSLs. While most existing tools focus on pure textual languages, the two language

[*] www.know4car.eu

workbenches from Meta Programming Systems (MPS) system from Jet brains [11] and the Domain Workbench from Intentional Software [12] also support the viewing and editing using graphical representations.

The contribution in this paper is a study of how language workbenches might be suitable for supporting coherence and traceability of information while working with variability of product and production systems. A proof-of-concept tool is under development as part of an ongoing project. The tool is based on the Domain Workbench from Intentional Software. We show with some examples, how the tool can be used within the domain of variability management, we also discuss strength and weaknesses with a language workbench approach.

The outline of the paper is as follows. In section 2 the different roles working with knowledge variability through the development of product and production systems are defined. In section 3 the approach is discussed. Finally, in the last section the paper conclusions and possible future work is discussed.

## 2. The user roles and how they work variability knowledge during development of product and production systems

Many user roles need to handle variability in their daily work. As each role will manipulate the information in some way, adding some of their respective experience to the information, it will be suitable from this point onwards to talk about knowledge [13] instead of information.

In this paper there we have a case included which exemplifies the concepts of each section. In Table 1 we introduce some of the roles that work with managing information in a system modeling variability of product and production systems.

Table 1. An example of roles working with development of a product and production platform and examples of contributed or required knowledge.

| Role | Contributed / Required Knowledge |
|---|---|
| Customer | Requirement specification |
| Product designer | Feature Diagram (Defining a Family of products) |
| | Bill-of-material |
| Production preparation engineer / Line builder | Precedence constraints on the assembly sequence |
| | Tolerance models |
| | Expected throughput |
| | Number of assembly cells |
| | Assembly line balancing |
| | Assembly Sequence |
| | Matching of operations to cells (Gantt charts) |
| Shop floor workers | Work instructions (mapping of operations to an individual worker within a cell) |
| | Work distribution within a cell |
| | Feasibility of work instructions |

Although the various roles, all contribute to the knowledge but the format of this contribution is very different. Each role has their own preference and requirements when it comes to the representation of knowledge.

Currently the information is represented in multiple sources, which are very syntactically diverse. Some are informal like text documents or rich text documents; others are semiformal like spreadsheets, while there are also more formal documents like CAD drawings and Bill-Of-Materials (BOM) that might be stored in structured databases. Although these document types are very different from each other, they still describe the product platform, but from different views and levels of abstraction.

The main problem regarding knowledge management is the lack of a central model to capture the process platform model. The lack of such a central model means that it is the engineers who have to play the central role. When someone wants certain knowledge from one specific document to be used in another document; it is the engineer who has to act as the translator between these two documents types.

Each user will add knowledge to documents, knowledge which often is based on other sources of knowledge. But as these documents are all in different formats, there is always a constant need for translation of information between these documents. Currently this translation is done by the engineer. This manual editing and translation can be extremely error prone, and time consuming.

Also there are important documents that can be complicated to create. As an example consider the role of the shop-floor worker in a manufacturing company. This company is working on a product platform with variability. It is important for the shop-floor worker to be able to give feedback to the product designers and production preparation engineers on for example assembly complications that show up during industrialization of a new line. Another example of such knowledge can be when the shop-floor worker has some change they want to make to the sequence of operations which are to be carried out in the assembly of the product. This might be due to that an optional feature was selected for this specific instance and then the provided assembly sequence is no longer feasible. Another problem is the lack of conceptual traceability between these document types. Although these documents are very different, they still describe the concept of the product platform from different views and levels of abstraction. So naturally there will be conceptual threads between the knowledge in these documents. Currently it is very hard to keep track of these conceptual threads and in rare cases where this traceability is kept it will be both hard and time consuming to keep it updated.

Let us return to the example with the shop-floor worker. The information that he/she needs from the process platform is work instructions. These work instructions will guide the shop-floor worker by defining the sequence of operations to perform. If the shop-floor worker realizes that the stated sequence of operations is not working, or it can be improved in some way, it needs to report in some form. There need to be a standard protocol for the shop-floor worker to contribute this new knowledge, back to the for example production preparation engineer. What also needs to be kept in mind is

the fact that although this knowledge is useful for the production preparation engineer but other roles will also benefit from it, for example the product designers. Currently the feedback of this kind of information is often either done manually, e.g. with a phone call, or through a more formal report which is in most cases a textual document. But none of these two types of feedback are transparent for other roles in the process platform other than the initial receiver of the information. Thus to allow automated analysis and propagation of knowledge back from the shop-floor operator to the product designers and production preparation engineers to provide the shop-floor worker with a way to input the new knowledge into the system. The more support the worker has in filing as detailed and accurate report the easier it will be for others to handle the knowledge appropriately.

## 3. A Language Workbench approach to manage product and production system variability

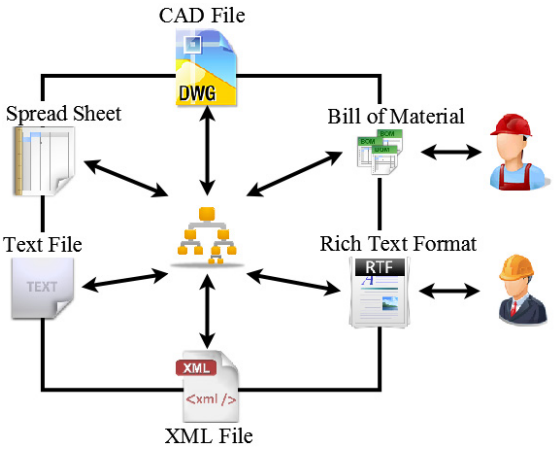



Fig. 1. A shared conceptual model of the domain. From this view different projections for different the roles can be automatically created.

In [14] the term *Language Workbench* is used to describe a class of tools that support the definition and use of DSLs. Ontologies also support the development of DSLs. Ontologies are defined [15] as follows, “Ontology is a formal, explicit specification of a shared conceptualization”. Thus, existing ontologies are useful when defining a DSL. In the Know4Car project, ontology for modeling of products, processes and resources has been developed [3]. Ontologies for variability management have been defined for modeling variability in software product lines [16], and in [17] the authors develop a UML-based, similar to an ontology, representation of a variability for a generic product , process and routing structure.

In this work the focus is on using a language workbench for describing the variability of product and production systems. While this is an ongoing work we are implementing a proof-of concept using the Domain Workbench (DW) from Intentional Software [12] and will in this paper discuss the strength and weakness with such an approach. Within the DW it is possible to define multiple domains that can be viewed and edited in multiple projects. For our domain this means that we have different DSLs for different aspect of the system. However, an important feature of the DW is that it is possible combine different DSL in an application. For example, product variability might be modeled using feature diagrams while relations between sequences of operations (precedence constraints) might expressed using so called sequence of operations diagrams [18]. Fig 1 shows the proposed architecture of a language workbench approach.

As the first step in language workbench approach a conceptual view of the domain is designed. The advantage of having this central conceptual view is two folds. Firstly, it is with the help of this conceptual view knowledge from different parts of the process platform is related to each other, hence giving a cohesive conceptual view of the overall system. Secondly, this conceptual view help in defining the needed DSLs that will make the roles in the process platform involved in the process of knowledge management. It is with the help of these DSLs that different user roles will be involved in the process of knowledge management, while working with their own preferred editable projections. In order to reach this conceptual view of modeling of feature diagrams and the relation to class and object diagram we have implemented a domain specific language which has many similarities with Clafer [19]. Clafer is a meta-modeling language which has been used in software product lines to model variability but is also appropriate to model variability of product and production systems. Software is also an essential part of a car or truck and which components of the software that will be used depend on how the car or truck is configured. For mechatronic product that has variability we would like to model the relations between physical components and software components. To the authors best knowledge existing PLM systems have no dedicated support for modeling the relations between feature diagrams, and the class and object diagrams of the software architecture. The flexibility of DSLs and the DW allow us to support the modeling of powerful meta-modeling languages like Clafer.

In the next step, with the help of a language workbench, editable projections of the conceptual view can be developed. An editable projection is a specific representation of the central conceptual view used by any of the user roles, which can act both as an output and an input from and to this central conceptual model. To put it in simple words an editable projection is a mapping between the entities in the central conceptual view and the specific representation which it plans to produce. There are two bases on which editable projections can be defined. An editable projection is defined for each document type which is used in the process platform. Generally editable projections will be defined based on the needs of the roles in the process platform. These editable projections will act mainly as a form of feedback from the various roles to the knowledge management framework.

It is important to mention that the use of language workbench tools will maintain the data topology of the overall system, meaning that the current physical data storages are kept intact. The main contribution of language workbench tools is to add a conceptual level over the whole process platform while providing support for associatively across views and logical consistency.

### *3.1. Advantages and disadvantages of using Language workbenches*

In section 2 some problems were mentioned which was the result of the current data architecture in the industry. Here the aim is to see how the introduction of a language workbench solution can help with these problems.

Firstly, the design of a conceptual view of the process platform will act as a central model between the various forms of documentation and knowledge presented from different roles in the system. This central model will act as the conceptual glue that will structure all the parts into a more uniform and harmonious structure, hence the resulting system will be a much more cohesive system. Secondly, it is with the help of this conceptual view that the traceability of a concept among various document types can be tracked. This is due to that in a language workbench the same concept can be used in several different languages. In the DW it is also possible to define equivalence relations between concepts in different domains. It is with the help of this updated traceability that cascading updates in various levels of knowledge can be handled in a more cost effective way. Thirdly, multiple DSLs, which are defined as a direct result of the central conceptual model, will help providing uniform languages through which all the roles within the process platform can communicate and contribute their knowledge to the overall knowledge management framework. This will provide an environment in which all the roles can give their possible feedback to other stakeholders involved in this process. Finally, with the use of tools provided by the language workbench, editable projections can be defined for each different document type, to capture the needed knowledge within these documents. These projections can also act as the needed editable views which will be needed by the various roles in the process platform.

All the items mentioned above will be counted as advantages of introducing a language workbench approach.

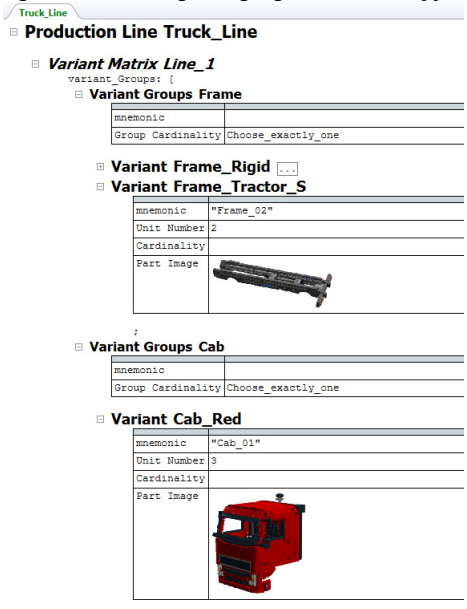


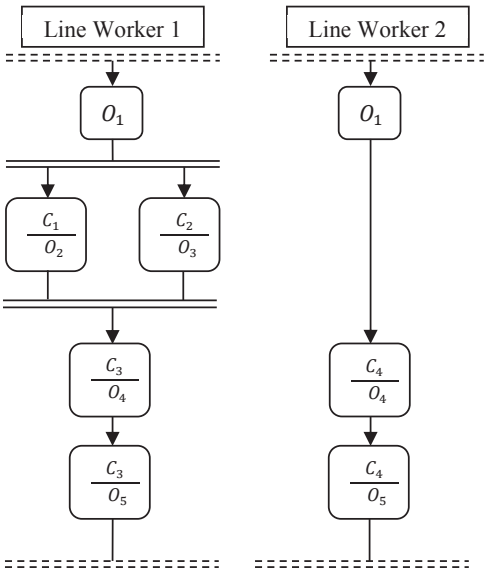

Fig. 2. Example of a representation of a feature diagram, in this case represented as table within the Domain Workbench.

But the introduction of language workbenches can have their own weaknesses alongside the advantages which were mentioned. Firstly, although the central conceptual view has many advantages but to reach such a complete view in the industry a process of maturity is needed. It is only through this process of maturity that the full potential power of such a central view can be realized. In order to reach such a maturity level time and resources are needed in each specific industry. Secondly, the process of finding the mapping between the central conceptual view and each user role specific representation can also be a time-consuming process. It is our belief that these two processes can be initialized and matured. The advantages of reaching state will much greater than the time and resources which will be needed for this process.

To see how editable projections provided by the language workbench tools can help to solve some of the mentioned problems in the previous section let us revisit the example case. A product designer work with a model of the variability of the product, this might be in the form of a feature diagram. This model is a combination of variants and variant values and the constraints which are used in the product assembly. Fig 2 shows the projection in which the product designer has defined a family of trucks and some initial constraint. This editable projection uses a rich-text format to define a product family, including all of its variability points and the different options each variability point has. Another important part of this projection is the constraint set on the product family. These constraints, defined using first order logic, can be defined for the whole product family and for each specific sub feature of the family. All the instances need to satisfy all the specified constraint rules. In the example case as part of the given feature diagram projection, Fig 2, an example of a constraint rule can be seen. This constraint rule is defined to restrict the whole feature hierarchy, in contrast to sub-feature specific constraints. And in this example the constraint rule is in the simplest form of first order logic in which the cab type is constrained by the choice of frame. Fig 3 introduces another type of editable projection which is the operation sequence projection, as defined in [18], that can be used by the shop-floor workers. In this example case, two workers are working on the left and the right hand side of a truck. For each worker in the cell a sequence of operations is defined, of which some operations are bounded by a specific condition, e.g. $O_2$ , $O_3$, $O_4$, or $O_5$. Some operations need to be carried out by both the workers, e.g. $O_1$. While others are only carried out by the worker on one side of the truck, e.g. $O_4$ or $O_5$. Some operations can only be carried out when a specific condition is fulfilled. The projection also has the capability to show parallel operations which need to be carried out by the workers, e.g. $O_2$ or $O_3$. These projections can currently be automatically generated base on product assembly specifications from the proof of concept tool.

Another result of having a central conceptual model of the process platform is the ability to analyse the model. The model consists of a feature hierarchy and a set of constraints which are used to define valid combinations of the features. An initial use of this model can be to create the different instances of the model. The advantage of creating the instances is three fold. Firstly, sometimes the listing of the

concrete instances would be advantageous in itself. Secondly, by analysing which instances have or have not been created one can reason about the constraint set of the system (whether the system is over or under constrained). Thirdly, it would allow for the creation of partial views [20]. Partial views are the result of partial instantiation of variables. These types of views are useful for some roles in the manufacturing industry e.g. shop-floor workers.




| Operation | Description |
|---|---|
| $O_1$ | Mount driveline |
| $O_2$ | Mount fifth wheel |
| $O_3$ | Mount grey side cover |
| $O_4$ | Mount accessories plate |
| $O_5$ | Mount fuel tank |

| Condition | Description |
|---|---|
| $C_1$ | If frame == frame rigid |
| $C_2$ | If frame == frame tractor S |
| $C_3$ | If left accessories position == used |
| $C_4$ | If right accessories position == used |

Fig. 3. Operation Sequence projection

## 4. Conclusion and future work

In this paper variability of process platforms is discussed. It is discussed that as a part of handling this variability there needs to be a specific solution for knowledge management within process platforms. The knowledge within the manufacturing industry is accumulated from the different roles active in the process platform. These different roles also require this knowledge with their own personalized presentations and at different levels of abstraction. Some potential problems facing process platforms are discussed like the lack of a central conceptual model, or the lack of traceability between the items in these personalized presentations. In an effort to solve these potential problems the use of tools based on the technology of language workbenches is considered. Possible ways of addressing the mentioned problems are discussed using tools provided by language workbench, namely some example editable projections which are defined on the needs of different roles. While implementing a full proof-of-concept tool using a language workbench is an ongoing work we show with an example how the tool will be used to view and editing information that are of importance for variability management of product and production systems.

As the future work of this project the central conceptual model should be extended to include more of the existing knowledge within the process platform. Suitable editable projections should be developed so more roles can interact with this central model. The method's applicability should be tested through close collaboration with industrial use cases. The central conceptual model should be expanded to include both requirement specifications from one side and generated code for both the product and production system on the other side. In this way the framework will cover the whole process of knowledge management from requirement specification to embedded software, code for programmable logic controllers and generated work instructions for the shop-floor workers.

## Acknowledgements

This work was partially supported by the Know4Car project, grant agreement number 284602, funded by the EC Seventh Framework Programme theme FoF-ICT-2011.7.4.

## References

[1] J. Stark, Product lifecycle management, Springer, 2011.
[2] P. Denno, T. Thurman, Requirements on information technology for product lifecycle management, International journal of product development, 2 (2005) 109-122.
[3] G. Pintzos, L. Rentzos, K. Efthymiou, N. Papakostas, G. Chryssolouris, A Knowledge Based Collaborative Platform for the Design and Deployment of Manufacturing Systems, in: Product Lifecycle Management for Society, Springer, 2013, pp. 268-276.
[4] S. Thiel, A. Hein, Modelling and using product line variability in automotive systems, Software, IEEE, 19 (2002) 66-72.
[5] J. Jiao, T. Simpson, Z. Siddique, Product family design and platform-based product development: a state-of-the-art review, J Intell Manuf, 18 (2007) 5-29.
[6] L. Zhang, J. Jiao, S. Pokharel, Process platform and production configuration for integrated manufacturing and service, in: Industrial Informatics, 2006 IEEE International Conference on, IEEE, 2006, pp. 498-503.
[7] S.J. Hu, J. Ko, L. Weyand, H.A. ElMaraghy, T.K. Lien, Y. Koren, H. Bley, G. Chryssolouris, N. Nasr, M. Shpitalni, Assembly system design and operations for product variety, CIRP Annals - Manufacturing Technology, 60 (2011) 715-733.
[8] M. Fowler, Domain-specific languages, Pearson Education, 2010.
[9] M. Mernik, J. Heering, A.M. Sloane, When and how to develop domain-specific languages, ACM Comput. Surv., 37 (2005) 316-344.
[10] M. Pfeiffer, J. Pichler, A comparison of tool support for textual domain-specific languages, in: Proceedings of the 8th OOPSLA Workshop on Domain-Specific Modeling, 2008, pp. 1-7.
[11] B. Becker, Re-Thinking the Stage-Gate Process–A Reply to the Critics, Management Roundtable Inc, Waltham, MA, (2006).
[12] C. Simonyi, M. Christerson, S. Clifford, Intentional software, in: ACM SIGPLAN Notices, ACM, 2006, pp. 451-464.
[13] I. Galandere-Zile, V. Vinogradova, Where is the border between an information system and a knowledge management system?, Managing Global

Transitions, 3 (2005) 179-196.
[14] M. Fowler, Language Workbenches: The Killer-App for Domain Specific Languages?, in, 2005.
[15] R. Studer, V.R. Benjamins, D. Fensel, Knowledge engineering: principles and methods, Data & knowledge engineering, 25 (1998) 161-197.
[16] T. Asikainen, T. Männistö, T. Soininen, Kumbang: A domain ontology for modelling variability in software product families, Advanced Engineering Informatics, 21 (2007) 23-40.
[17] L. Zhang, J. Jiao, P.T. Helo, Process platform representation based on Unified Modelling Language, International journal of production research, 45 (2007) 323-350.
[18] B. Lennartson, K. Bengtsson, C. Yuan, K. Andersson, M. Fabian, P. Falkman, K. Akesson, Sequence planning for integrated product, process and automation design, Automation Science and Engineering, IEEE Transactions on, 7 (2010) 791-802.
[19] K. Bąk, K. Czarnecki, A. Wąsowski, Feature and meta-models in Clafer: mixed, specialized, and coupled, in: Software Language Engineering, Springer, 2011, pp. 102-122.
[20] A. Voronov, K. Åkesson, F. Ekstedt, Enumeration of valid partial configurations, in: Proceedings of Workshop on Configuration, IJCAI 2011, 2011, pp. 25-31.