varBPM
A Product Line for Creating Business Process Model Variants

Andreas Daniel Sinnhofer, Peter Pühringer and Christian Kreiner
Institute for Technical Informatics, Graz University of Technology, Austria
{*a.sinnhofer, christian.kreiner*}*@tugraz.at, p.puehringer@inode.at*

Keywords: Software Product Lines, Feature Oriented Modelling, Business Processes, Tool Integration.

Abstract: Business processes have proven to be essential for organisations to be highly flexible and competitive in today's market. To manage the life-cycle from modelling such business processes over the execution and the maintenance, Business Process Management Tools are used in the industry. In many cases, different business processes do only vary in few points. This leads to the situation that new business process variants are formed through copy or clone of previous solutions leading to a high number of instantiated process templates. However, this means that changes to a template affects many processes, where all of them need to be manually updated, which can lead to a considerable amount of work and money for a bigger company. In this paper, we will present a framework for the integration of business process modelling tools and software product line engineering tools to provide a systematic way to reuse and trace process variations of whole process families.

1 INTRODUCTION

Business Process (BP) oriented organisations are known to perform better regarding highly flexible demands of the market and fast production cycles (McCormack and Johnson (2000); Hammer and Champy (1993); Valena et al. (2013); Willaert et al. (2007)). These goals are achieved through the introduction of a management process, where business processes are modelled, analysed and optimised in iterative ways. Nowadays, the business process management is also coupled with a workflow management, providing the ability to integrate the responsible participants into the process and to monitor the correct execution of the business process in each process step. To administer the rising requirements, so called business process management tools are used (BPM-Tools) which cover process modelling, optimization and execution. In combination with an Enterprise-Resource-Planning (ERP) system, the data of the real process can be integrated into the management process.

In many cases, business processes do only vary in some points, which leads to the situation, that new process variants are created through a copy and clone of old solutions (often called as templates). As a result, such templates are instantiated in many various processes which makes the propagation of process improvements time and cost intensive for a bigger company. Also the consistency of the documentation of this huge number of process variants is a challenging task.

Software Product Line Engineering (SPLE) techniques have been successfully applied for almost any domain, providing a technique for the systematic reuse of domain artefacts. Although the topic of product line techniques in the domain of business process modelling is not new (e.g. Gimenes et al. (2008); Rosa et al. (2008); Fantinato et al. (2012); Derguech (2010)) only little work is found for the issues related to the correct configuration of whole process families (e.g. Hallerbach et al. (2009a,b)), the integration into existing toolchains and the reuse throughout various production plants. Thus, our approach is focused on developing a framework for the integration of a SPLE Tool and a BPM Tool, to provide a generic way to generate process variants of whole process families. In particular, we use the SPLE Tool for a systematically reuse of expert knowledge in form of valid process variations, designed in an appropriated BPM Tool. The integrity of the process variations is secured by the capabilities of the BPM Tool and a rich constraint checking in the SPLE Tool. Furthermore, our proposed approach enables the abilities to automatically trace all process variants for an automatic propagation of changes and process improvements and the systematic integration into the capabilities of the BPM Tools such as documentation generation, workflow engines, process optimisation tools, etc.

Daniel Sinnhofer A., PÃijhringer P. and Kreiner C.
varBPM - A Product Line for Creating Business Process Model Variants.
DOI: 10.5220/0005886901840191
In *Proceedings of the Fifth International Symposium on Business Modeling and Software Design (BMSD 2015)*, pages 184-191
ISBN: 978-989-758-111-3

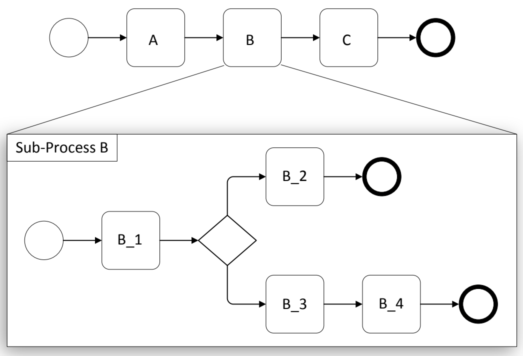

Figure 1: Principal structure of a business process according to Österle (1995) which is used for our approach. Starting with an abstract description of the process, the tasks are further described in sub-processes until a complete work description is reached (microsopic level).

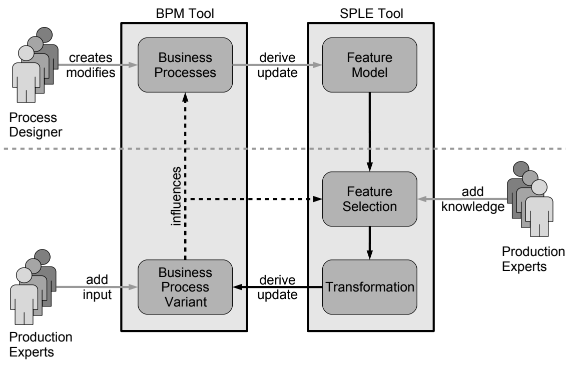

Figure 2: Overall conceptual design. The upper side of the Figure describes the domain engineering part and the lower side of the Figure the application engineering part.

This work is structured in the following way: Section 2 gives an overview over the concepts of tool integration and the design paradigm for business processes which is needed for our framework. Section 3 presents the conceptual design of the framework and states construction rules of the according feature models and some design rules for the BPM Tool. In section 4 we will introduce our case study regarding some metrics and implementation details. Section 5 summarizes the related work and finally section 6 concludes this work and gives an overview of open issues.

2 BACKGROUND

2.1 Tool Integration

According to the work of Karsai et al. (2005), two possible patterns exists for tool integration. The first approach is named ”Integration based on integrated models” and is based on the idea of a common data model which is shared between each participating tool. This means that each tool needs two model converters, one for the conversion of the native data model into the common data model and one for the opposite direction. The data is shared over a so called integrated model server where each tool can publish or consume data. For obvious reasons this approach is meaningful applicable if each participating tool has a similar data model. A drawback of this approach is that it does not scale very good with the number of connected tools.

The second pattern is named ”Integration based on process flows” and is based on the idea of a point to point message based communication. Each participating tool registers itself at a backplane providing information about what data is shared and what data is intended to be consumed. As a result, this approach scales better with the number of participants since potentially fewer model transformations are needed at which each model can be better optimized regarding the communicating tools. This approach is used in those situations, where the data is processed in a specific sequence.

For our framework both patterns would be possible. Due to the fact, that the number of participants is small (in most circumstances there is only one SPLE Tool and one BPM Tool) and since the representation of business processes is very similar throughout various tools, the first approach is more applicable.

2.2 Business Processes

A business process can be seen as a sequence of tasks/sub-processes which needs to be executed in a specific way to produce a specific output which is of value to the costumer (Hammer and Champy (1993)). According to Österle (1995) the process design on the macroscopic level (high degree of abstraction) is split up into sub-processes until the microscopic level is reached. This level is reached, when all tasks are detailed enough, so that the process employees can use it as work instructions.

In other words, a complete business process is designed in layers, where the top layer is a highly abstracted description of the overall process, while the production steps are further refined on the lower levels. As a result, the lowest level is highly dependable on the concrete product and production environment, providing many details for the employees. In fact the top layers are – mostly – independent from the concrete plant and the supply chain and could be interchanged throughout the production plants, whereas the lower levels (the refinements) of the processes would need to be reconsidered. Figure 1 gives an overview of such a structure. Variability of such a process structure can either be expressed through a variable structure of a process/sub-process (e.g. adding/removing nodes in a sequence) or by replacing the process refinement with different processes. The current version of our developed prototype focuses on the second method but the framework is not limited to it.

2.3 Informal: Feature Model

A feature is defined by Kang et al. (1990) as a *"prominent or distinctive user-visible aspect, quality, or characteristic of a software system or system"*. In context of a Software Product Line, a feature model is a model which defines all these features and explicitly states their relationships, dependencies and additional restrictions between each other. It enables the ability to visually represent the variable parts of a system and the options available for all products of a product line.

Table 1: Needed process information within the SPLE Tool.

property name	description
id	A unique id to identify the process/task
category id	A unique id of the category of the process
display name	A human readable and understandable name of the process/task
children	A list of ids which references the processes/tasks of the process itself (empty if the microscopic level is reached)
additional data	A list of additional data which is needed for the concrete instantiation of the process. E.g. data for a workflow engine, the responsible workers, etc. This data can be provided through the BPM Tool (almost static) or can be design variable in the SPLE Tool.

3 VARIABILITY FRAMEWORK

3.1 Conceptual Design

The overall conceptual design is based on a feature oriented domain modelling approach and is displayed in Figure 2. It is intended, that the domain experts (process designer) design process templates in the according BPM Tool, providing also all needed information for e.g. a workflow engine. Based on these templates and the abstract process model (the top level description of the process) a feature model is partially automatically created/updated with the guidance of the domain experts. This is done by identification of the variation points and the linkage of the according variations on every process level. Each variation can contain additional variability by either defining new variation points where further refinement can be linked to or by a variable process structure. Additional constraints regarding the possible combination of features are intended to be modelled in the SPLE Tool, but are not limited to it. In application engineering the domain experts (not necessarily a process designer, but someone who knows the current needs of the production) adds his knowledge and selects the needed features. The found description is then automatically transformed in a real business process which can be executed by the workers. During the process execution, loads of data is generated regarding the performance and efficiency of the process. Thus, it is possible that some additional information is added to the derived processes which leads to a possible influence of the according business process templates or a possible influence of the feature selection. This flowback mechanism is an important task and needs to be considered for the maintenance and the evolution throughout the lifecycle of a process.

For illustration a short example for the flowback mechanism is given: Task B of the process displayed in Figure 1 could be dependent on the logistic chain of a supplier of a specific part or material. If during the execution of the process the supplier is changed, it is also likely that the overall control of the logistic chain is changed due to the fact that the newly integrated supplier can only deliver goods in a specific way. If the process of the logistic chain was not already modelled, then the process designer would need to create a new process variant and would need to update the existing feature models first. After this is done, our proposed toolchain needs to update the feature selection of the respective process in the SPLE Tool. This automatically updates the structure of the overall process variant, leading to an almost on the fly update of the complete process workflow especially when the process variation was already modelled.

Summarizing this concept means that the SPLE Tool is responsible for the following points: It needs to assist the domain experts (process designer) during

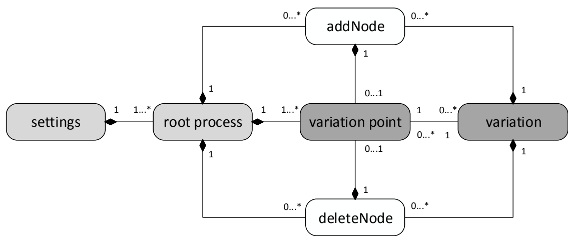

Figure 3: Feature model construction rules. The white parts cover the variability on the process level itself and the dark grey parts cover the variability between each layer of the overall process structure.

the (partially) generation of the feature models and during the selection and creation of the concrete product variant. Furthermore, it has to keep track of all generated process variants to automatically apply process template improvements or changes to the overall process structure and to flow back information added during the execution in the BPM Tool. To do so, the tool needs information about the properties of the process displayed in Table 1. The BPM Tool is responsible for the creation of semantically correct process variants and to provide capabilities which are of value for the developing company e.g. automatic documentation generation, workflow engine, etc. For obvious reasons, each Tool must have rich import/export capabilities or the ability to integrate user defined plug-ins to extend the functionality.

3.2 Type Model

To model the variability within the SPLE Tool in a structured way, the feature model should support the following type models:

- **settings**: Is a data model, containing information for the tool adaptors to identify the right datasets (e.g. identifier of the database of the BPM Tool from which the process structure is imported; login settings, etc.)
- **root process**: A process model of the top level process and therefore an abstract description of the overall process sequence. It consists of various nodes where some of them deal as variation points.
- **variation point**: A node in a process where at least one variation can be linked to.
- **variation**: A process model for a task or a subprocess which can be linked to a number of variation points. If it is a process, it can also contain variation points or a variable process structure (addition/deletion of nodes).
- **addNode**: Adds a node (task or process model) at a specific location of the process structure. This added node can also be a variation point for further refinements. The addition of the node can be dependent on the feature selection.
- **deleteNode**: Deletes a node (task or process model) at a specific location of the process structure. The deletion of the node can be dependent on the feature selection and hence it is also possible to link further refinements to this node.

The construction rules for this type model can be seen in Figure 3. The settings node is only instantiated once in such a model. The root process is somehow very similar to a variation, but with the difference that it must contain variation points (to prohibit a ”Blob” anti pattern [Brown et al. (1998)]).

3.3 Design Rules for Business Processes

Aforementioned, the processes should be designed as stated by Österle (1995). Secondly we have noticed, that almost every bigger BPM Tool supports the assignment of specific group identifiers to groups of processes, providing a more structural design of the processes. Thereby it is possible to automatically map specific groups of processes to a specific variation point. This leads to the situation that new variations can be automatically detected and can be advocated for an integration into existing feature models. Furthermore, the structuring in groups of processes enables the ability to introduce a constraint check so that variation points are limited to specific groups of processes. This increases the assurance in the creation of semantically valid processes.

4 INDUSTRIAL CASE STUDY: VARBPM

In this section an overview over our industrial case study is given, which describes the domain of our industry partner and the developed toolchain.

4.1 Industrial Project Partner

Our project partner Magna Cosma[1] is an international company in the metal stamping and assembly industry – specialised on class-A car body panels and closure parts (e.g. doors) – with several plants all over the globe. The implemented business processes

[1]http://www.magna.com/de/kompetenzen/karosserie-fahrwerksysteme

are mostly controlled by an SAP infrastructure and are designed with the BPM-Tool Aeneis[2]. Although some plants are specialised on the same production parts, almost every plant develops and maintains their own business processes, which makes it difficult to compare processes, mark bottlenecks, optimise the processes and publish the changes to other plants.

4.2 Tool Integration

As mentioned before, the tool integration of the SPLE Tool (pure::variants[3]) and the BPM Tool (Aeneis) is based on the Pattern "Integration based on Integrated Models". The integrated data model contains the relevant data enumerated in Table 1 where all fields are of type String respectively an array of Strings for the children and data field. This means that the native data model of the SPLE Tool is the integrated data model and hence only the BPM tool adaptors need to implement a data conversion. To support an update mechanism without sending the complete process, each published dataset can be assigned to a specific type indicating what should happen with this dataset. Possible types are:

- **New**: Indicating that this dataset was not published before and hence it should be integrated directly just as is.
- **Update**: Contains the id of the according dataset and the data which shall be updated. Non specified attributes are not affected.
- **Remove**: Contains the id of the according dataset which should be deleted out of the system. Linked variations of such a process are not affected.

The developed tool adapters are also applicable to get notified when data is added/updated so that such changes are processed almost immediately. If this mechanism is not supported by the participating tool connector, an operator needs to trigger this update mechanism manually. In the current development, the SPLE Tool needs to check manually for updated data since this task is intended to be supported by an domain expert, whereas the BPM Tool uses the benefits of the immediate notification system. The communication of the tools is done by an XML based file exchange and due to some consistency issues the communication is only possible if both tools are running. I.e. deriving process variants is only possible if the BPM Tool is running too.

[2]http://www.intellior.ag
[3]http://www.pure-systems.com

4.3 pure::variants

pure::variants is a feature oriented domain modelling tool and is based on Eclipse. As such, it can easily be extended based on java plug-in development. During the implementation of this project, five different plug-ins where developed:

- An import plug-in, which is capable of importing the process structure - including the definition of variation points and the according variations - and converting it into a feature model compliant to the construction rules displayed in Figure 3 without the white parts.
- An extension to the internal model compare engine so that different versions of created feature models can be compared with each other.
- An update mechanism to automatically search for deleted / added variations or updated process structures, providing graphical assistance for the domain expert.
- An extension to the internal model transformation engine so that the feature selection is automatically converted into a business process in the common data model; This process is then delivered to the attached BPM Tools, so that a native version of the process can be created/updated and executed.
- Additions to the internal model check engine to model and create only valid processes (e.g. checks related to the feature selection, the consistency of the feature model, etc.).

To keep track of all generated business process variants, a list including all ids of the processes is stored and maintained in a file located in the same directory as the variant description model (feature selection), but hidden from the user perspective. This list is automatically updated when a process variant is deleted in the BPM Tool or created with the SPLE Tool. pure::variants also provides a framework for the comparison of different models, which enables the ability to compare different process variants in an efficient way.

4.4 Studied Use Cases and Results

In the list below, use cases can be found which we investigated during the development of our approach regarding the performance of our toolchain (time saving). The according results are displayed in Figure 5, where the white bars are related to the manual case and the grey bars to our approach. Although the varBPM approach automatically integrated each derived process into the capabilities of the BPM Tool

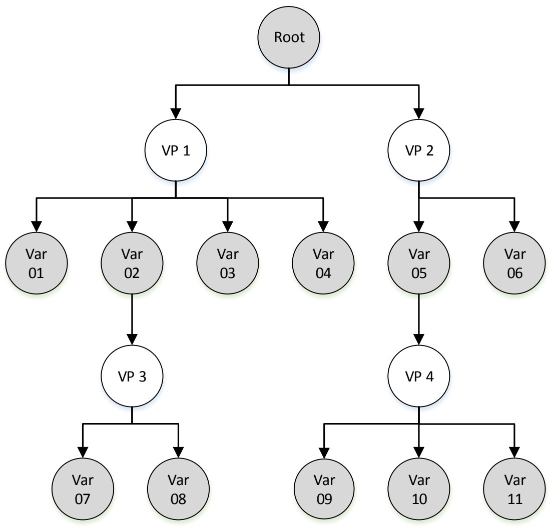

Figure 4: The variability structure of the process for the evaluation examples.

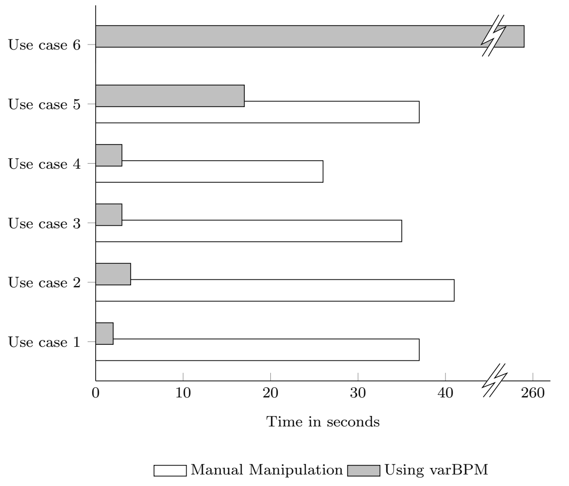

Figure 5: The results of the evaluation use cases. The grey parts are the time spent using the varBPM approach and the white parts states the time taken if a manual manipulation is used.

(documentation generation, workflow engine), the manual approach only reflects the time used for the creation of the bare process structure. The used process structure is part of a bigger process used for an on-demand manufacturing of spare parts for different car manufacturer.

Use case 1: For this use case, the time was measured that a domain expert needs to create a new process variant manually or using an existing feature model. The process setup consisted of four variation points (two top-level variation points and two variation points on lower levels) where a total number of twenty-seven different process variants were possible (for illustration, the variability structure can be seen in Figure 4). The number of other processes in the database of the BPM tool was considered to be low (20 other processes). The experiment was repeated with different experts and different process setups (the overall variability structure was kept the same but the process structures changed). The results of this use case are divided with the number of variation points to get a rough estimate for the time saving per variation point.

Use case 2: This use case is an addition to the first one, with the difference of a high number of other processes in the database (200 processes).

Use case 3: Is related to the topic of maintenance. In this scenario a process template was changed and all process variants should be updated. The domain expert was told that there is a number of six variants he needs to update providing only the name and the id of the changed process template. The size of the database was limited to 50 processes. The change to the template was a deletion/addition of a node. The results displayed in Figure 5 are normed to the update of one process.

Use case 4: This use case is an addition to the previous case, but with the difference that the domain expert was now told how the process variants where called (assuming that the domain expert was responsible for the creation of the process variants and exactly knows the variants).

Use case 5: In this situation, a new process variation was created by a process designer and the domain expert should now derive a new product variant out of it (in fact, a new variation for the Variation Point ”VP 3” was developed). It is assumed that the processes are designed according to our proposed design rules. The number of other processes in the database was considered as low (20 processes). This use case also gives a good estimate on how much overhead is produced, to update an existing feature model.

Use case 6: In this case, a new feature model was developed according to the variability structure displayed in Figure 4 to generate a metric on how much overhead is produces for the initial creation of the feature model. To get a rough estimate on the overhead per variation point, this number is divided by four.

The results of ”Use case 2” were surprising, since our developed approach performs worse in relation to the previous scenario. The reason for this is that the project now consists of ten times more processes, leading to a more time demanding search for the right processes. For humans, it was still quite easy to find the right processes since they were organised in a clear ”human understandable” manner. As a result the increasing number of processes do not have a high influence to a manual manipulation if the processes are structured in a clear and meaningful way. Otherwise the time would increase more significantly.

To get a rough estimate of the break-even point, the following equation is used:

$$p \approx \frac{\text{Overall Overhead}}{\text{Average time saving}} = \frac{259}{4 \cdot 29} \approx 2.2 \qquad (1)$$

The overall overhead is the time spent to create the feature model ("Use case 6") and the average time saving is calculated using the average time saving per variation point multiplied by the number of variation points. Interestingly, when software product line techniques are applied to pure software systems, the break-even point is also located at around three systems (according to Pohl et al. (2005)).

4.5 Restrictions

Depending on the API of the used BPM Tool, your approach can be limited in terms of the available features. This means that if the BPM Tool only provides access to the basic process structure, our framework is limited to the creation of derived processes without the ability to automatically integrate the models into the capabilities of the BPM Tool (e.g. workflow engine).

5 RELATED WORK

As stated in the survey of Fantinato et al. (2012), major challenges in the field of business process variability modelling are related to the reaction time of process changes and of the creation and selection of the right business process variants, which are also main topics in our approach.

Derguech (2010) presents a framework for the systematic reuse of process models. In contrast to our approach, it captures the variability of the process model at the business goal level and describes how to integrate new goals/sub-goals into the existing data structure. The variability of the process is not addressed in his work.

Gimenes et al. (2008) presents a feature based approach to support e-contract negotiation based on web-services (WS). A meta-model for WS-contract representation is given and a way is shown how to integrate the variability of these contracts into the business processes to enable a process automation. It does not address the variability of the process itself but enables the ability to reuse business processes for different e-contract negotiations.

While our approach reduces the overall process complexity by splitting up the process into layers with increasing details, the PROVOP project (Hallerbach et al. (2009a,b) and Reichert et al. (2014)) focuses on the concept, that variants are derived from a basic process definition through well-defined change operations (ranging from the deletion, addition, moving of model elements or the adaptation of an element attribute). In fact, the basic process expresses all possible variants at once, leading to a big process model.

The work of Gottschalk et al. (2007) presents an approach for the automated configuration of workflow models within a workflow modelling language. The term workflow model is used for the specification of a business process which enables the execution of it in an enterprise and workflow management system. The approach focuses on the activation or deactivation of actions and thus is comparable to the PROVOP project for the workflow model domain.

Rosa et al. (2008) extends the configurable process modelling notation developed from Gottschalk et al. (2007) with notions of roles and objects providing a way to address not only the variability of the control-flow of a workflow model but also of the related resources and responsibilities.

The work of Leitner and Kreiner (2010) addresses the process variability through a bottom up approach by examining the possible configurations through the scan of the according ERP System (SAP). In contrast to this approach, we focus on an top down method to abstract the complexity of the underlying ERP System.

The Common Variability Language (CVL Haugen et al. (2013)) is a language for specifying and resolving variability independent from the domain of the application. It facilitates the specification and resolution of variability over any instance of any language defined using a MOF-based meta-model. A CVL based variability modelling and a BPM model with an appropriate model transformation could lead to similar results as presented in our work.

6 CONCLUSION AND OUTLOOK

The reuse of business process models is an important step for an industrial company to survive in a competitive market. With our work we have proposed a way to combine the benefits of software product line engineering techniques with the capabilities of a business process modelling tool to provide a framework for the systematic reuse of business processes. With the proposed design rules, our approach results in an automatic detection and propagation of new and/or changed business process variations. On the other hand it leads to an automatic integration of new assembled process variants into the BPM capabilities such as an automatic integration into a workflow engine, integration of responsibilities and resources, etc.

Our developed framework covers the variability of the process in two different ways: Through the linkage of different process variations to variation points and through a variable process structure (deletion / addition of nodes) in each layer. Due to the fact that our developed framework is in an early stage of usage, further research efforts would address the collection and evaluation of data regarding the evolution and maintenance of the process models. In this context, an integration of Six Sigma[4] into our framework is aimed to provide a complete framework from modelling and improving process models. Additionally the customization of the ERP system of the underlying system (in this case SAP) is an interesting topic, providing a complete framework for the topics of process modelling, execution and maintenance including the planing of the resources of the concrete production facility.

ACKNOWLEDGEMENT

We want to gratefully thank Magna Cosma for sponsoring this project, Danilo Beuche from pure::systems and Intellior AG for their support.

REFERENCES

Brown, W. J., Malveau, R. C., McCormick, H. W. S., and Mowbray, T. J. (1998). *AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis: Refactoring Software, Architecture and Projects in Crisis*. John Wiley & Sons.

Derguech, W. (2010). Towards a Framework for Business Process Models Reuse. *In The CAiSE Doctoral Consortium*.

Fantinato, M., Toledo, M. B. F. d., Thom, L. H., Gimenes, I. M. d. S., Rocha, R. d. S., and Garcia, D. Z. G. (2012). A survey on reuse in the business process management domain. *International Journal of Business Process Integration and Management*.

Gimenes, I., Fantinato, M., and Toledo, M. (2008). A Product Line for Business Process Management. *Software Product Line Conference, International*, pages 265–274.

Gottschalk, F., van der Aalst, W. M. P., Jansen-Vullers, M. H., and Rosa, M. L. (2007). Configurable Workflow Models. *International Journal of Cooperative Information Systems*.

Hallerbach, A., Bauer, T., and Reichert, M. (2009a). Guaranteeing Soundness of Configurable Process Variants in Provop. In *Commerce and Enterprise Computing, 2009. CEC '09. IEEE Conference on*, pages 98–105. IEEE.

Hallerbach, A., Bauer, T., and Reichert, M. (2009b). Issues in modeling process variants with Provop. In Ardagna, D., Mecella, M., and Yang, J., editors, *Business Process Management Workshops*, volume 17 of *Lecture Notes in Business Information Processing*, pages 56–67. Springer Berlin Heidelberg.

Hammer, M. and Champy, J. (1993). *Reengineering the Corporation - A Manifesto For Business Revolution*. Harper Business.

Haugen, O., Wasowski, A., and Czarnecki, K. (2013). Cvl: Common variability language. In *Proceedings of the 17th International Software Product Line Conference*, SPLC '13.

Kang, K., Cohen, S., Hess, J., Novak, W., and Peterson, A. (1990). Feature-oriented domain analysis (foda) feasibility study.

Karsai, G., Lang, A., and Neema, S. (2005). Design patterns for open tool integration. *Software & Systems Modeling*, pages 157–170.

Leitner, A. and Kreiner, C. (2010). Managing erp configuration variants: An experience report. In *Proceedings of the 2010 Workshop on Knowledge-Oriented Product Line Engineering*, KOPLE '10, pages 2:1–2:6.

McCormack, K. P. and Johnson, W. C. (2000). *Business Process Orientation: Gaining the E-Business Competitive Advantage*. Saint Lucie Press.

Österle, H. (1995). *Business Engineering - Prozess- und Systementwicklung*. Springer-Verlag.

Pohl, K., Böckle, G., and Linden, F. J. v. d. (2005). *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer.

Reichert, M., Hallerbach, A., and Bauer, T. (2014). Lifecycle Support for Business Process Variants. In Jan vom Brocke and Michael Rosemann, editor, *Handbook on Business Process Management 1*. Springer.

Rosa, M. L., Dumas, M., ter Hofstede, A. H. M., Mendling, J., and Gottschalk, F. (2008). Beyond control-flow: Extending business process configuration to roles and objects. In Li, Q., Spaccapietra, S., and Yu, E., editors, *27th International Conference on Conceptual Modeling (ER 2008)*, pages 199–215, Barcelona, Spain. Springer.

Valena, G., Alves, C., Alves, V., and Niu, N. (2013). A Systematic Mapping Study on Business Process Variability. *International Journal of Computer Science & Information Technology (IJCSIT)*.

Willaert, P., Van Den Bergh, J., Willems, J., and Deschoolmeester, D. (2007). *The Process-Oriented Organisation: A Holistic View - Developing a Framework for Business Process Orientation Maturity*. Springer.

[4]Six Sigma is a set of techniques and tools for a systematic management of process improvements based on quality management methods, including some statistical methods.