# Model-Driven Integration of Business Information Systems

Niels Streekmann[1], Ulrike Steffens[1], Claus Möbus[2] and Hilke Garbe[2]

[1]OFFIS
Escherweg 2
D-26121 Oldenburg
Germany
{niels.streekmann, ulrike.steffens}@offis.de

[2]Learning Environments and Knowledge Based Systems
University of Oldenburg
D-26111 Oldenburg
Germany
{claus.moebus, hilke.garbe}@uni-oldenburg.de

September 19, 2006

## Abstract

The integration of business information systems is an important task for enterprises, with an increasing demand on the level of integration. While most integration projects in the past concentrated on the integration of data used by different applications, today the focus is on the integration of the applications themselves. Furthermore there already is a tendency towards the integration of systems on the process and presentation level. In this paper we present the approach taken in the MINT project where a method for model-driven integration on the level of business processes that bases on the MDA approach of the OMG is proposed.

## 1 Introduction

Enterprises are often confronted with the task of integrating legacy systems into new system landscapes or to integrate individual systems with standard software. With existing methods this is a complex and expensive challenge. In the MINT project [1] we face this challenge by the usage of model-driven methods which are expected to reduce time and costs of the development of integration solutions. Another principle that is applied in the MINT project is the usage of patterns to structure models and simplify the transformation from models to other models or code. The contribution of this paper is a critical review of the OMG proposed MDA models and process in the context of system integration. By this, the current state of the main concepts of the MINT approach are introduced.

The paper is structured as follows. Section 2 gives an overview of the MINT approach. Sections 3, 4 and 5 describe further details of the major aspects of the approach. Section 6 describes the evaluation of the approach in the MINT project while section 7 concludes and points out future work planned to put the approach into practice.

## 2 The MINT Approach

The MINT approach provides methods for the integration of business information systems and bases on OMG's Model Driven Architecture (MDA). It supports the views on the system described in [8], namely the *computation independent model (CIM)*, the *platform independent model (PIM)* and the *platform specific model (PSM)*, to specify the application to be built. While this approach is disputed and certainly only one possible form of model-driven development methods among others, in our context the distinction between these models is considered appropriate. The exact role of these models is discussed in the sequel to this introduction.

The MINT approach focuses on the integration of software systems. In practice there are two different instances of integration on this level. On the one hand this regards the integration of modern software systems with existing legacy systems which are valuable for the enterprise and cannot be replaced at once by a new system. The other case is the integration of standard software with individual software specific for a domain or a certain enterprise. It is assumed that both cases can be addressed using the same methods described in the following. Figure 1 depicts the model levels of the MDA and the integration described above. It represents an overview of the procedure model of the MINT approach.

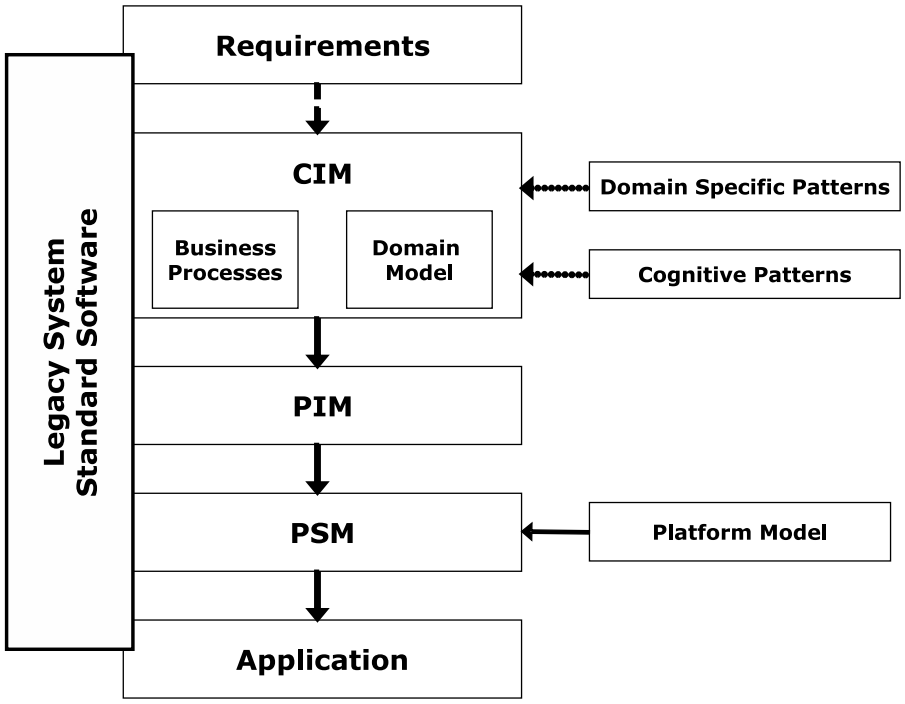


Figure 1: Procedure model of the MINT approach

The integration of systems on the level of business processes is a task that requires a lot of knowledge by domain experts. This knowledge enfolds the tasks of the domain that can be facilitated by a software system. These tasks are a combination business processes to execute the business of the enterprise and domain knowledge which includes the concepts and their correlations inside the domain.

In MDA these requirements are acquired in the CIM. The dashed line in figure 1 indicates that this is no automated, but rather a manual process that needs special methods and guidelines. In the MINT approach we use cognitive and domain-specific patterns to support this process. A more detailed description of these patterns is given in section 3.

The CIM itself is the main source of communication between domain experts and software developers. Thus it has a special role in software development that is neglected in practical approaches to MDA. In contrast the CIM has a central role in the MINT approach due to the fact that it is the initial point for our integration approach since it includes the business processes used as a basis for the integration tasks in the following model steps. The second aspect we identified as a basis for integration of different systems that resides in the CIM, is a domain model that represents the intra- or inter-organizational understanding of the domain the application operates in. The structure of the CIM is described in section 4. Since the CIM has the purpose to acquire the requirements and serve as a source for communication it should be modelled in a language that is understandable for domain experts. One way to do this is though the usage of domain-specific languages that are a widespread object of current research [6].

Opposed to the CIM the PIM is a technical model which provides the architecture of the system. In the MINT approach we examine how the PIM can be derived from the CIM and how interfaces derived from the business processes can be combined with interfaces offered by legacy systems and standard software to describe the integration of business information systems on an architectural level. This process of architecture derivation and integration of systems in the PIM is described in section 5.

The PSM specifies the realisation of the interfaces described in the PIM on a specific platform. In doing so, the interfaces that are already provided by the legacy system or standard software and their implementations have to be considered in all models. A base for this kind of consideration of integration aspects is the BALES methodology, as described in [11]. A possible platform technology for implementation of the above mentioned interfaces are web services. An example for the usage of web services in the context of integration is the application of the DUBLO pattern described in [4] where web services are used for the integration of a legacy system with a modern system architecture during the migration process from the old to the new system. The extraction of web services from systems that are not build to interoperate on this level is also part of the MINT project, but not in the focus of this paper.

## 3 Guided CIM Creation

Creating the CIM is the first task in the MDA development process. It comprises the domain specific processes and tasks and has to be developed in collaboration with domain experts. The CIM has great importance for the rest of the development process, as new requirements or changesin the CIM lead to changes in the corresponding PIM and PSM. As domain experts are often not experienced in modelling processes, they have to be supported during this task. Research in our project will show whether this can be accomplished by using cognitive patterns. "The term *cognitive patterns* refers to recurring templates that humans use during problem solving/ reasoning activities." [2] Since these patterns model natural human problem solving, it is assumed that it is easier for domain experts to express their domain knowledge using these instead of business process patterns for example. Business patterns have a different intention (e.g. workflow organization). Another benefit is mentioned by Gardner et al., who premise "that the notion of cognitive patterns, applied to organizational and system processes in business, can facilitate a deeper understanding of these processes." [2] So, the use of these patterns can result in a more complete and correct model of the domain-specific processes and tasks. Existing pattern libraries are described by Gardner et al. and Schreiber et al. [9] amongst others. In MINT these libraries have to be extended with domain-specific patterns, e.g. by adoption or discovery of new patterns. A scientific goal of the MINT approach is to identify such patterns and evaluate how they can be applied to enhance the CIM. An example is the usage of these patterns as templates in the modelling language that have to be configured with application-specific values.

## 4 CIM Structure

The structure of a CIM depends on the purpose of the system to be build. The CIM should represent all aspects of the system that are important from a domain experts point of view. Further technical details like the platforms the system runs on or details about the interfaces of the integrated systems should not be relevant on this level. Hence, as described in section 2, the languages used to model the application from a domain expert's point of view should be adapted towards their specific needs. This results from the realisation that many problems in software engineering are based on misinterpretations of requirements and misunderstandings between domain experts and software developers. MDA holds the opportunity to overcome these misunderstandings by offering views on the system in languages that are specific for every stakeholder. The PIM on the other hand should be modelled in a language understandable for software developers. Furthermore it should be usable for analyses of the system and further generation of a PSM and code. In MDA the standard modelling language is UML. This fits for technical description of the system as in the PIM and PSM, but can lead to problems in modelling the CIM, because many domain experts are not able to model their intentions in UML. A way to overcome this is the introduction of domain-specific languages (DSL) [6]. An easy way to introduce a DSL on the basis of UML is the usage of UML Profiles that restrict the usage of UML elements to simplify the language and adapt it for a certain domain.

For the description of integrated business information systems we propose a structure for the CIM that consists of a business process model and a domain model. The business process model describes the processes the system is intended to support. This model is application-specific though it may be part of a larger model that includes all business processes of the enterprise. The domain model describes the domain of the system. It is intended to provide a common model for all applications throughout an enterprise. Further details about these models and the according modelling languages are given in the following subsections.

### 4.1 Business Process Model

Business processes are a well-established in the domain of business information systems to model the tasks of such a system. Furthermore it is a good starting point for the integration of these systems, since the tasks described in a business process may be fulfilled by different systems that are already available as legacy systems or still have to be developed.

We chose UML2 activity diagrams to model the business processes, because they are well researched in this context and there is a good tool support for UML and MDA. For the modelling of business processes in UML there are extensions in the form of UML Profiles that we consider to use within the scope of the MINT approach. A profile for the detailed modelling of processes is described in [7] while [5] presents a profile that can be used to describe the processes from a business perspective. I.e. it describes different types of processes, the persons in charge of the processes, deliverables and so on.

Another goal of the MINT approach regarding modelling the CIM is the use of patterns to structure the model and describe transformations from the CIM to the PIM. For this purpose we will evaluate the cognitive and domain patterns mentioned in section 3 as well as business process and software engineering patterns. It is in the scope of currently ongoing research whether these patterns can be used as model templates or even to create new domain-specific languages that make it easier for domain experts to describe their requirements of a software system.

### 4.2 Domain Model

The domain model is a model of the application domain. I.e. it models the enterprise-wide or domain-wide concepts of the domain agreed upon by the domain experts. Such a model is of great importance for the integration of systems, because it serves as the common base for the data used in all systems. How the model is exactly used in the PIM to integrate the interfaces of different systems is described in section 5.

The domain model should not be created anew for every new application or integration project. It should rather be derived from existing organisational or domain standards and then be used for all projects within the enterprise. These standards will of course differ from domain to domain and especially for applications that integrate several systems, the CIM will contain a particular domain model for each domain that is involved and of importance for the business process that is to be implemented by the system.

## 5 Architecture Derivation and Architectural Integration

The architecture of a system and the basic architectural aspects of the integration of systems are modelled in the PIM. In practice the derivation of an architecture is a model-to-model transformation from CIM to PIM. As claimed by Gerber et al. [3] transformations are the missing link to the successful application of model-driven development. Much scientific and practical work is expended on the exploration of how such transformations should be done. However most of the work concentrates on the use of transformations to build a PSM from a PIM and how code can be generated from the PSM. I.e. the main question is how an existing architecture can be implemented on a specific platform. The focus of our work is to find a way of building that architecture from the requirements instead. Since the CIM can be seen as a structured model of the requirements in the language of domain experts, we centre on the transformation from CIM to PIM.

We propose to use business process model from the CIM and the patterns used to build the CIM as a source for the transformation. The business process tasks in the CIM will be transformed to interfaces in the PIM that provide methods to fulfil the task. As can be seen here the transformations from CIM to PIM will only supply a skeleton PIM which has to be completed by software engineers. This also covers the integration aspects that contain the mapping from interfaces derived from the CIM and interfaces derived from the systems that have to be integrated and the mapping from the domain models used in the description of the last-mentioned interfaces and the domain model from the CIM.

The mappings mentioned in the last paragraph implicate that some kind of adaptation may be needed if the interfaces, that are mapped to each other, differ in the implementation of methods or protocols, or in the naming or definition of concepts in the respective domain models. A source of our work in this area is the work of van den Heuvel [10] on adaptation regarding the BALES methodology.

After the architecture and the integration aspects have been described in the PIM, the next steps are the realisation of the architecture on a certain platform and the generation of code for this platform. These are again done by transformations, which are model-to-model transformations in the case of PIM to PSM and model-to-text transformations in case of PSM to code. The PIM to PSM transformation employs a platform model that includes detailed information about the platform. The platform we propose to use for the integration of business information systems are web services.

## 6 Evaluation

Current experience shows, that a model-driven software development approach is most successful in rather focused domains which are well-understood and where generated code can build on powerful libraries or frameworks. Therefore, in MINT we focus on the domain of coupling object-oriented business code with relations databases. Although this is a very specific aspect of system integration, it is rather frequent in practice, ranging from newly developed systems to software migration projects and legacy integration.

While model-driven approaches for coupling object-oriented business code with relational databases are to be extended in MINT, it is important to validate these approaches in contrast to other techniques for solving the same problem. Namely, persistence frameworks (such as Spring or Container-managed Persistence in Enterprise Java Beans) or using a high-level persistence interface, such as ADO.NET are alternative solutions. Therefore, the MINT project is concerned with discovering guidelines that will assist software architects in their decision on how to approach the coupling of object oriented business code with relational databases. This is a considerable step towards an engineering approach to software design, as in the current state of software design such decisions mainly base on experience or, worse, on intuition. Far too often, in software development, the benefits and, in particular the limitations of specific approaches are unclear. In contrast to this, the MINT approach is highly concerned to associate with its methods and tools general guidelines on the applicability of the approach, as it is also common in other engineering disciplines. By that, MINT will deliver a building block to the future engineering character of software design.

## 7 Conclusion and Future Work

In this position paper we introduced the MINT approach which represents a method to integrate business information systems based on business processes and domain models. The paper highlights three major tasks needed to implement this method and proposes solutions to face them. The identified tasks are:

- Creation of the computation independent model.
- Definition of the structure of such a model with respect to integration.
- The derivation of a software architecture from the CIM and the description of methods towards integration based on this architecture.

Future work on the approach will concern the usage of DSL and UML profiles to model the CIM and the transformation from CIM to PIM based on patterns and the involvement of domain-specific patterns to structure the CIM. The approach will be evaluated within the work of the MINT project regarding the functionality in selected scenarios and regarding non-functional properties in comparison to other generative integration approaches.

## 8 Acknowledgements

The work presented in this paper originates from the MINT project as proposed by Ralf Reussner, who is heading the project. MINT is supported by the *Federal Ministry of Education and Research* in the scope of the *Forschungsoffensive Software Engineering 2006*.

## References

[1] MINT project homepage. http: www.mint-projekt.de. Last visit: 13.09.2006.

[2] Karen M. Gardner, Alexander Rush, Michael K. Christ, Robert Konitzer, and Bobbin Teegarden. *Cognitive Patterns: Problem-Solving Frameworks for Object Technology (Managing Object Technology Series)*. Cambridge University Press, 1998.

[3] Anna Gerber, Michael Lawley, Kerry Raymond, Jim Steel, and Andrew Wood. Transformation: The missing link of MDA. In *ICGT*, pages 90–105, 2002.

[4] Wilhelm Hasselbring, Ralf Reussner, Holger Jaekel, Jürgen Schlegelmilch, Thorsten Teschke, and Stefan Krieghoff. The dublo architecture pattern for smooth migration of business information systems: An experience report. In *ICSE*, pages 117–126, 2004.

[5] Beate List and Birgit Korherr. A UML 2 profile for business process modelling. In *ER (Workshops)*, pages 85–96, 2005.

[6] Marjan Mernik, Jan Heering, and Anthony M. Sloane. When and how to develop domain-specific languages. *ACM Computing Surveys*, 37(4):316–344, 2005.

[7] Bernd Oestereich, Christian Weiss, Claudia Schröder, Tim Weilkiens, and Alexander Lenhard. *Objektorientierte Geschäftsprozessmodellierung mit der UML*. Dpunkt Verlag, 2003.

[8] Object Management Group (OMG). MDA guide version 1.0.1.

[9] Guus Schreiber, Hans Akkermans, Anjo Anjewierden, Robert de Hoog, Nigel Shadbolt, Walter Van de Velde, and Bob Wielinga. *Knowledge Engineering and Management. The CommonKADS Methodology*. MIT Press, 2000.

[10] Willem-Jan van den Heuvel. Matching and adaptation: Core techniques for MDA-(ADM)-driven integration of new business applications with wrapped legacy systems. In *Model-Driven Evolution of Legacy Systems*, 2004.

[11] Willem-Jan van den Heuvel, Wilhelm Hasselbring, and Mike Papazoglou. Top-down enterprise application integration with reference models. In *EFIS*, pages 11–22. IOS Press and Infix, 2000.