Towards Model Driven Architecture in Health Care Information System Development

Hannes Schlieter[1], Martin Burwitz[1], Martin Benedict[1], and Oliver Schönherr[1]

Dresden University of Technology, Chair of Information Systems,
esp. Systems Engineering, Dresden, Germany
{hannes.schlieter,martin.burwitz,martin.benedict,
oliver.schoenherr}@tu-dresden.de

Abstract. Failed software projects are often the result of an unsystematic transfer of business requirements to the implementation. This deficit led to the specification of the Model Driven Architecture (MDA). It claims a consistent use of conceptual models for the software development process from requirement analysis to technical specification of software. The MDA reduces the gap between the business level and the information technology (IT) level by defining a methodological framework to link these levels (Business-IT alignment). We will present the use of an MDA in health care domain. For this purpose, we show how the paradigm of MDA can be configured to implement medical application software based on a telemedical IT platform (telehealth platform). Additionally to the conceptual structure of the developed approach and the domain-specific alignment, lessons learned from the experiences gathered during design process will be formulated as assistance for similar projects and substantiated with an exemplary application.

Keywords: Model Driven Architecture, Health Care, Conceptual Modeling, Method Engineering, Clinical Pathway

1 Potentials of Model-Driven Software Development

Since the introduction of the case-based compensation model, care providers face the conflict between increasing the quality of care (I), transparency of medical services (II) and the purpose of a resource efficient care system (III) [1]. To efficiently master the challenge of the increasing complexity of medical care processes, Clinical Pathways (CP) have been established in recent years. They are specific, standardized descriptions of clinical processes for defined combinations of symptoms that are adapted to clinical conditions [2]. They are geared to the entire multidisciplinary care process of a patient type [3]. As Panella & Vanhaecht stated, CPs are more than only the structure of a care process and even more than a part of the patient record. They are "a patient-focused concept, a tool to model the care, a quality and efficiency improvement process and a product in the patient record" [4].

12th International Conference on Wirtschaftsinformatik,
March 4-6 2015, Osnabrück, Germany

Schlieter, H.; Burwitz, M.; Schönherr, O.; Benedict, M. (2015): Towards Model Driven Architecture in Health Care Information System Development, in: Thomas. O.; Teuteberg, F. (Hrsg.): Proceedings der 12. Internationalen Tagung Wirtschaftsinformatik (WI 2015), Osnabrück, S. 497-511

Likewise, the organizational and functional needs involve new challenges for the supporting IT systems. Until now, the development of Hospital Information Systems (HIS) was characterized by an evolutionary design of functional modules in the field of laboratory, radiology or picture archiving and communication systems as well as administrative systems. Thus, the typical IT ecosystem of HIS is a heterogeneous network of applications of different designs and developers [5]. Therefore, it can slowly react on new requirements that are implicated by changes of the care landscape.

The alignment ability of IT systems becomes more and more important with regard to the innovations in the areas of mobile technologies, semantic analysis, and social media techniques as well as the continuous patient empowerment. This requires an organizational ability, in order to master the balancing act between business requirements and the design of the IT system. In this context, Business-IT-Alignment is propagated by the information systems research discipline as ability of an organization, to dynamically adapt the information system according to changing functional requirements. It describes the goal to link up the business world and the IT world [6]. The Model Driven Architecture (MDA) has developed as an elementary instrument in this field. It describes a framework for model-driven design and adaptation of an IT system on the basis of a holistic business model. In contrast to the traditional software development process, MDA focuses on the consequent utilization of diagrammatic descriptions (models), to describe the contextual situation, which is successively transferred through different model layers into a technical model [7]. The strength of the MDA is to ensure that the transformation results are compliant to the architectural principles. Platform Model and Modeling Language (see sec. 2.1) act like a corset for the system development process. A further strength lies in the ability of addressing different stakeholder groups by a better fit of modeling languages and stakeholder needs. For example, icons for modeling concepts can represent common professional symbols (e.g. a test tube in context of a laboratory information system). On the one hand, this helps to adequately model the context. On the other hand, the domain expert assesses the model being a reflection of his domain and his specific language. Additionally, the strict classification of different model layers fosters the transformation of business models to technical models that are used by the software engineer to derive the specified software [8].

These strengths are the main reason to develop a specific MDA for the health care domain, especially against the background of a concrete problem context. As stated in section 3.1 in detail, there is a need for a mechanism that fosters a sustainable telehealth platform in terms of a methodology that provides an efficient way to deploy new artifacts on the platform and to ensure these artifacts are compliant to the platform properties.

Apart from a few exceptions, there are currently no approaches, which address the application of MDA in the health care domain [9, 10]. Additionally, MDA research mostly contributes to the design of technical, application-oriented models and their transformation. The research on MDA, however, takes little attention on the study of design and transformation of business models. In particular, the relevance of model-

ing for communication of different stakeholders like professionals in context of the methodological design of a MDA is less investigated.

This paper contributes to the tension between an efficient application and the business needs oriented design of applications. Therefore, it will be shown how the paradigm of MDA can be used, in order to build a methodological fundament for the systematic creation of an application system. To illustrate the design considerations and to demonstrate the method, we use the case of an IT-based workflow support for an interdisciplinary stroke care from an EU-funded project. With this article, we primarily focus on the aspect, how the domain-knowledge, modeled in the MDA, can be structured and used to contribute to the application system development.

The presented research work can be assigned to the design-oriented branch in IS research [11, 12]. Design science research (DSR) is actually more seen as a research paradigm than a specific method [13]. Its central aspect is artifact-oriented science comprising the creation and evaluation of IT artifacts [14]. The specific arrangement of a DSR procedure depends on the type of the research artifact and its maturity regarding the knowledge bases [15]. The paper is structured in DSR-typical manner like GREGOR & HEVNER explicitly propose [16].

Thus, the paper is structured as follows: Following a brief introduction to the motivation and research goals, the knowledge base and terminological foundations of MDA are laid in the second section. In the third section, the case scenario and its project background is given (sec. 3.1), the artifact description in terms of the method draft (sec. 3.2) and its evaluation (sec. 3.3) are delineated. The evaluation is conducted by an artifact demonstration using the case example from the project introduced in section 3.1 [17]. Since contributing generalizable results is limited in the field of DSR, we aggregate "lessons learned" on the basis of the design process in section 4. The paper ends with a short summary and an outlook on the further research.

2 Knowledge Base and Prior Research

2.1 Terms and Definition of the Model Driven Architecture

The concept of MDA is closely related to the standardization efforts of the Open Management Group (OMG) concerning the architecture of a model-based approach for the development of software artifacts [17]. The pillars of the specification are the three different modeling layers:

- Computation Independent Model (CIM), containing the business model
- Platform Independent Model (PIM), containing the software engineering oriented model without technology aspects
- Platform Specific Model (PSM), containing the technology-related aspects of the target platform in a model

Based on the MDA, the domain knowledge can be transferred into a model system and derived into a software solution. The business models as a result of the discourse between domain experts and modeling experts are (partially) automatically trans-

formed through the three layers (CIM → PIM → PSM). The transformation process using defined transformation rules is done with the participation of modeling experts and software engineers. As a result from these transformation steps, application code and its documentation can be generated. In addition, changes can be captured by the unique explication of the business requirements in a model system and transferred into the application system when using MDA [18, 19]. This ensures the consistency between business model and IT model at any point during development.

2.2 Layers of the Model Driven Architecture

In the further course, the three layers of MDA are described in detail. The CIM includes all information of the business model. The layer addresses the communication with the domain experts in order to describe the supported business case in a complete and comprehensible manner. For this purpose an adequate graphical representation is needed, which is understandable and intuitive from the perspective of domain experts, but also formalized to enable the semantically correct transformation.

The PIM at the next layer contains the platform-independent design specification and addresses the modeling expert (requirements engineer), who has internalized the business context of the model, as well as the software engineer, who externalizes the technical context of the model. Suitable notations for this layer are the Unified Modeling Language (UML) or the Business Process Model and Notation (BPMN), which are current standards for the modeling of software artifacts.

The PSM at the lowest layer connects the model of PIM with platform-specific information, such as concrete interface information or service definitions. Therefore, it is possible to generate several PSMs for different target platforms like Java Enterprise Edition (JEE) or .NET from the PIM. The last transformation step produces code (model-to-code-transformation). The estimates for the completeness of the generated code are differently evaluated in the literature (up to 80%) [20, 21].

3 Model-Driven Software Development in the Health Care Region of Eastern Saxony

In the following sections, a model-driven software development method, based on the MDA, is delineated, which is used to develop applications on an expandable telehealth platform. Therefore, we outline the project background (sec. 3.1), the conceptual design of the MDA and its domain specific alignment (sec. 3.2), and demonstrate its application (sec. 3.3).

3.1 Project Background

The method we describe in this paper was developed in the scope of the EU-funded infrastructural project “CCS Telehealth Ostsachsen”. Objective of this project is the development of an open platform for medical applications for the region of Eastern Saxony, in order to provide a continuous information supply in integrated medical

care processes. This should improve the availability of medical services particularly in structural weak, rural areas.

A crucial claim in this project is to assure the openness and expandability of the platform for third-party software developers. To achieve this claim, platforms should feature a high degree of modularity while using established standards. The process of development should be optimized and especially unified by a standardized set of instruments and methods helping developers to implement components for the telehealth platform. The MDA is the foundation for these instruments and methods, which enable the developer to derive application-oriented design models (PIM and PSM), workflow descriptions and architectural models from domain-oriented models (CIM). The application-oriented models can be used to derive formal syntax (e.g. executable code) or to customize existing basic platform components (e.g. master data like users or document templates).

To illustrate the MDA-based approach and the resulting platform, one of three project solutions, the aftercare of stroke patients (tele-stroke), will be considered more closely.

Scenario - the path-based stroke care: The term stroke covers several different cerebrovascular diseases (e.g. cerebral infarction). These diseases have a significant impact on the national economy (see [22]) and drastic effects to the life of the patient. Subsequent to the insult a complex therapy and extensive rehab measures, coordinated by a case management, are necessary to improve the patients' quality of life. The case management is led by a case manager. A neurologist usually carries out the treatment of the stroke patient. Owing to comorbidities that were previously present or resulting from the stroke, an interdisciplinary therapy is often required additionally [23].

Due to this interdisciplinary treatment, it is reasonable to coordinate the disease in an integrated care network. The comprehensive case management in this network can be modeled with CPs. HÜBNER ET AL. found that electronic supported CPs are underrepresented in the hospital sector [24]. This is also true for the aftercare of stroke patients. In stroke care the CP is used as a guideline for the case management. Currently, the stroke care in context of the project is organized using excel sheets, which leads to limitations in efficiency and optimization opportunities. In the course of the project, the implementation of an automatic task management based on the telehealth platform replaces this manual method.

Foundations of the task management are CPs that are specified in a business model. These pathways describe the care process of therapy and rehabilitation for the time frame of one year since the insult. The CP-model acts as a planning and controlling instrumentation for the health care professionals as well as a tool for the coordination and documentation of the treatment. Beside the process-oriented information, the model moreover contains connections to involved participants, necessary documents, physical resources, time-constraints and quality indicators.

3.2 Development of the Method

In information systems science, conceptual models as semi-formal means of description and communication, as well as methods have established as capable means for information systems design [25, 26]. Methods are defined as libraries of techniques supplemented by regulations on their users as well as their order and kind of usage, to achieve a certain goal [27]. According to the purpose of their application, modeling methods and model-driven methods are distinguished. While the former guides the construction of a model [28], the latter additionally explicates the way of model usage for the solution to a business problem [29]. The application of conceptual modeling in context of the MDA is a model-driven method aiming a systematic software development. The focus of the model-based method is the means-end-combination of model application rather than solely restricting on the design of the modeling result. Thus, different design dimensions the method intends can be derived, depending on the point of view. These dimensions result in the framework illustrated in figure 1 and explained as follows.

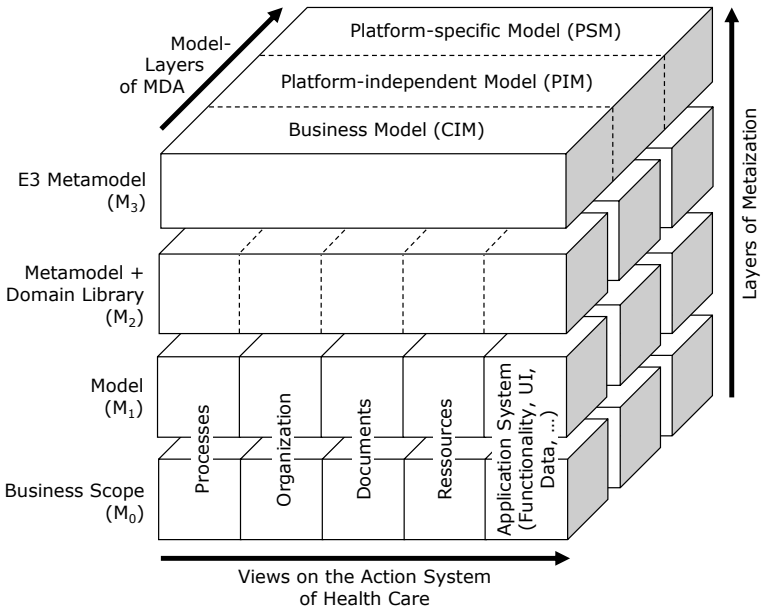

Fig. 1. Systematization of the model-based method

The MDA model layers CIM, PIM and PSM were adopted for the first structuring dimension, as this provides a clear separation of business-oriented and IT-oriented description of the model system and methodologically supports the transition between them. Concerning the content of each of the three models, the modeler is usually able to distinguish purpose-related parts of the model, to attain understanding of elements, structures and behavioral effects of a system [26]. In the field of information model-

ing, this approach of an aspect-oriented definition of views is called multi-perspectivity, where a model is known as a system of related sub-models (diagrams). Thereby, different diagrams of individual views are associated by integrative modeling elements, i.e. elements that are used in more than one view. Hence, the model is opened up by the sum of all views, their diagrams and their relations [30].

Several workshops with domain experts, considering the medical action system as well as the requirements engineering for the telehealth platform in context of the described project, finally lead to the construction of the following relevant views for modeling IS in health care domain: Processes, Organization, Documents and Resources. Within these views, we are able to provide a pure business model of health care institutions. In addition, there is a need for a fifth view that focuses the IT in health care business, esp. supporting application systems. This view of the model system is the main entry point for software development according to the MDA. It provides concepts for modeling the typical software engineering aspects, e.g. functionality, user interface, or data structures. Thereby, this view is highly associated to the business views as well. For example, use cases as representation of software functionality can be partially derived from the business processes through the identification of needed IT support in daily care. Furthermore, the business elements provide a configuration for the IT elements, e.g. the organization model configures a role-based access control while the care process configures the application for IT-guided care.

Each view is assigned by specific diagram types (presentations), which provide an adequate description of the relevant aspect of the model. Therefore, as well as to derive the necessary modeling concepts of each view and presentation, the domain context was analyzed through various interviews with domain experts and finally formalized in a meta-model, that specifies the corresponding modeling language.

The modeling language is used for model construction to represent the medical action system respectively the information system (Business Scope M_0 in figure 1) in an object model (M_1). For this purpose, the modeling language provides the structure (abstract syntax) as well as the graphical notation (concrete syntax) of the concepts (e.g. treatment step, document on CIM-layer) that is necessary to provide an adequate representation of the subject of modeling [31]. The modeling language in turn can be defined as a model the same way. This model then acts as a meta-model (M_2) in terms of the object model as this principle is called metaization [32]. For the specification of the model system and the corresponding meta-meta-model (M_3) within the proposed method, we used the E3-method [33]. Beside the definition of the necessary modeling concepts, the resulting meta-model (M_2) can be extended by domain specific libraries, esp. to facilitate the modeling on CIM layer and to provide a common foundation for various modeling projects. For example on organizational view, the basic concepts of an organigram are specialized by domain-specific roles or positions. In the case of stroke care we defined the neurologist and the case manager as specific role concepts.

The meta-model of the CIM layer additionally provides a foundation for the transformation of the model according to the concept of MDA. Thereby it has to be specified which meta-objects of the CIM layer are transformed into which meta-objects of the PIM layer and how far this transformation can be automated (analogously for the further transformation from PIM to PSM).

3.3 Demonstration - the Model Driven Path Execution

Evaluation in DSR can be differentiated into naturalistic and artificial evaluation [17]. The demonstration of artifacts is a common technique for an artificial evaluation. In the further course, we demonstrate the use of our framework according to the scenario shown in section 3.1, designing an application for the aftercare of stroke patients. With this scenario, we show the utilization of the specified MDA-approach (see sec. 3.2). The scenario relies on an abstraction of a comprehensive requirements specification, which is done in the outlined project. The focus of the example is set to the implementation of the CIM, for which the CPs and the pathway-associated documents are particularly considered. Based on this example, we outline the transformation tasks and the design of the lower layers (PIM, PSM). The developed application shall support the planning, execution and documentation of aftercare processes for stroke patients and thereby optimize them.

The first step of the MDA is to transform the context of the domain into the model scope. This involves the creation of medical business models in the CIM-layer. For the process view, CPs are used to build the business model. These CPs describe the characteristic process of the aftercare performed by the case manager (see lower left quadrant in figure 2). To model the process view, we use a domain specific extension of the flowchart [34], which is semantically explicit enough from the IT's point of view. Apart from the procedural descriptions of treatment steps and decisions, the models of CPs additionally contain the connection to involved actors, necessary documents and physical resources as well as the determination of time constraints and quality indicators. Furthermore, the CP is the integrating component for the different views. For example, the linkage between documents and treatment steps allows describing the information interchange between domain experts. The internal structure of these documents is specified in the document view of the business model.

Modeling the information flow between information nodes in the process view allows integrating the document and the process view. The meta-class `Treatment Step` and the meta-class `Clinical Document` are both generalized by the meta-class `Information Node`. Figure 2 shows a section from the CIM-metamodel. Parts of the process (left column) and the document view (right column) from the meta-layer (M_2) (upper row) and a corresponding sample for a CP of stroke aftercare (M_1) (lower row) are shown. The document view contains a sample model for the internal structure of a vastly simplified discharge report, whereby the difference solely lies in the concrete syntax for the clinical document due to the different purposes. As a result the consistency between the views is preserved.

Examples for representative document types in the health care domain are physician discharge reports, admission forms and temperature curves. Their structure is independent from the process and is maintained in the document view. The standardized information model from the Clinical Document Architecture (CDA) allows formalizing a document type structure in a model-based way [35]. Hence, we use the CDA to specify a metamodel for document models. For example the CDA describes, that a document can contain a structured body containing sections that consist of narrative texts (e.g. tables, lists etc.) and machine-readable clinical statements.

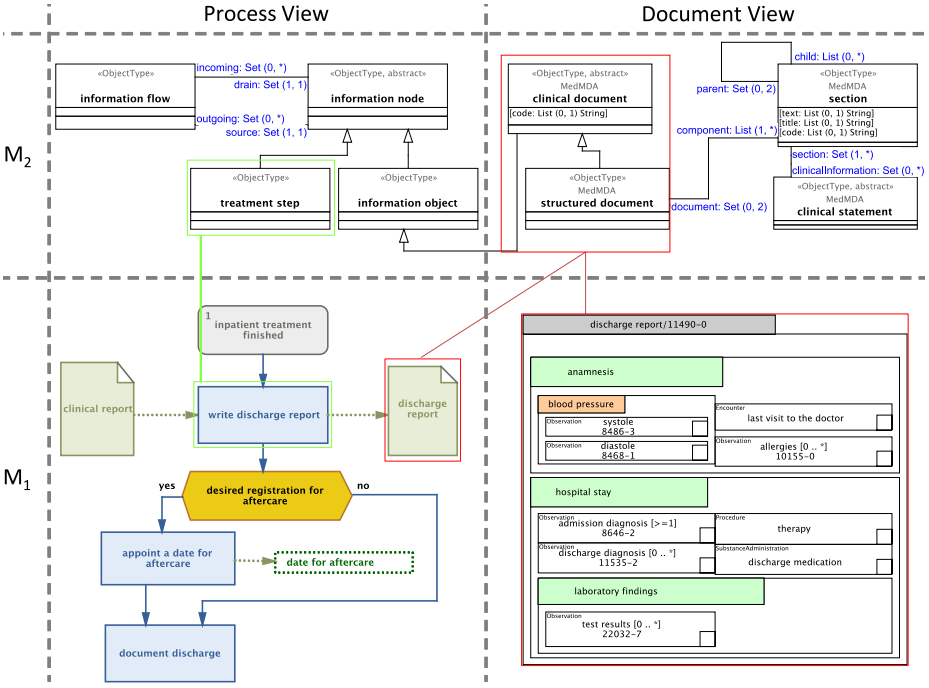

Fig. 2. Excerpt from the (meta-) model of the process and document view on the CIM-layer

The platform application must be configured based on the CP-model to support the operative care. To derive the configuration, the CP-model must be transformed along the MDA-layers. Generally, this is done by transformation rules, which are specified for the individual meta-elements of the M_2-layer. The M_2 of the PIM is specified by the metamodel of the UML-superstructure (see [7]). For example UML activity diagrams, as a notation that is generally accepted in computer science, can represent processes at the PIM-layer. The software engineer gets a technology-agnostic description of the workflow to be implemented. Static concerns on PIM-layer like the structure of a discharge report are represented by UML class diagrams. The additional semantic of the different document components can be preserved by stereotypes [7]. At the PIM-layer the integration of both views, document and process view, can be ensured using the `object node` of the activity diagram, which can be type casted with the corresponding `document class` (e.g. `discharge report`$_{\text{«ClinicalDocument»}}$) (see [7]). For the illustrating example, the transformation between CIM and PIM can be done by following transformation:

- `Clinical Pathway`$_{\text{CIM}} \rightarrow$ `Activity Chart`$_{\text{PIM}}$
- `Document Structure`$_{\text{CIM}} \rightarrow$ `Class Diagram`$_{\text{PIM}}$

We use ATLAS Transformation Language (see [36]), which is a declarative transformation language. The transformation can be done with the following rule-set, which we simplified for demonstration purposes:

```
create OUT : UML from IN : CMOD
rule{ from tstep : IN!TreatmentStep
      to act : OUT!Activity  ( name <- tstep.name,
                     edge <- tstep.informationFlow, ...)}
rule{ from docum : IN!Document
      to  rootDoc : OUT!Class  ( name <- docum.name,
                                 attribute <- ...)
          objn : OUT!ObjectNode  (
                     name <- docum.name, type <- rootDoc,
                     edge <- docum.informationFlow )
      do{ rootDoc.applyStereotype(clinicalDocument) }}
rule{ from flow : IN!InformationFlow
      to owfl : OUT!ObjectFlow ( source <- flow.source,
                                 target <- flow.target)}
```

`Source model` is an instance of the model specified on M_2 on the CIM layer. In the transformation rule it is named `CMOD`. The first rule is to transform the `Treatment Step` to an `Activity`. The second rule transforms the `Clinical Document` to instances of both meta-classes `ObjectNode` and `Class`. For the resulting class the stereotype «`ClinicalDocument`» is applied. The information flows between the activities and object nodes are transformed by the third rule. The result of this transformation rule is applied to the `edge`-attributes of the `Activity` and `ObjectNode`.

An exemplary result of a CIM-to-PIM-transformation is illustrated in figure 3. It shows an activity diagram, in which the discharge report is shown as a typed `Object Node` resulting from the action `export discharge report`. Furthermore, a hierarchic class model representing the document structure is shown. To preserve the document semantic from the CIM, the classes are annotated with stereotypes that correlate to the CDA information model as demonstrated for the Clinical Document.

An important aspect is the preservation of the integration of the views. For the example, the relationship between the `ObjectNode` (`discharge report`) and its root element `discharge report`$_{\text{«ClinicalDocument»}}$ in the class structure has to be preserved. The class `discharge report`$_{\text{«ClinicalDocument»}}$ represents the document type. The `ObjectNode` is a `TypedElement` and therefore it has the meta-attribute `type` whose range is the meta-class `Type`. The meta-class `Class` has a transitive generalization relationship to the meta-class `Type`. Ergo, all instances of the meta-class `Class` can be used as a type for `object nodes`. A CIM-to-PIM-transformation not only describes the mapping of meta-classes to other meta-classes (`Clinical Document`$_{CIM}$ → {`ObjectNode`$_{PIM}$, `Class`$_{PIM}$}), but also a transformation rule for preserving the integrity of the views, which is for-

mulated in the rule-set by the italic-marked statement: `objn : OUT!ObjectNode (name <- doc.name, type <- rootDoc)`

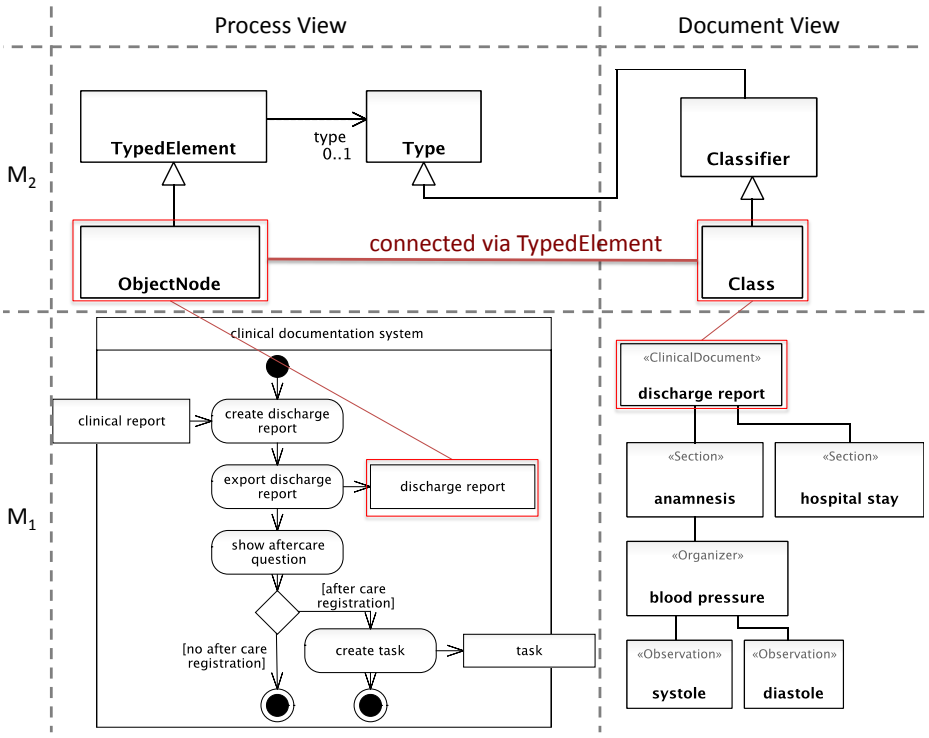

Fig. 3. Excerpt of the UML-(meta-) model of the process and document view on the PIM-layer

Finally, to describe the concrete specification of the IT at the PSM-layer, the transformation from the PIM to the PSM has to be done, which considers the target platform and technology. The set of configuration parameters that influence the transformation is specified as platform model. It specifies the fundamental system architecture and the provided basic functionality concerning realization of interfaces and standards. For the example of integrating of the CPs, the platform provides an XML-schema-definition (XSD), which also contains the mentioned concepts (e.g. documents and treatment steps). The model-based representation of XSD as part of the Platform Model and the XML instances on the PSM-layer can be done with UML profiles [37, 38]. The result of the transformation is a concrete XML instance, which conforms to the mentioned XSD. This can be used to configure the application. The structure of a discharge report will be transformed to a corresponding CDA document-level template. The hierarchic class structure can be mapped to the hierarchic CDA, where the stereotyped classes had to be mapped to the correlating CDA com-

plex types (e.g. anamnesis«Section» to an instance of <xs:complexType name="POCD_MT000040.Section">[1]). Therefore the application allows an adequate provision of necessary information and therewith it supports the business process with an optimal customized information technology support.

4 Discussing the Lessons Learned

Similar to case study research (see [39]), the validity of results in research on design artifacts and their application is limited to the research cases. Thus, this research method is limited generating generalizable statements. Nevertheless, beside the principle approach of the MDA-design, we gained lessons learned from the gathered project experiences, which can contribute to research, and facilitate similar design challenges.

Lesson 1 – Use an Adaptive Modeling Toolset. At the beginning of the project, we have considered UML-diagrams (esp. activity diagrams) as a tool for the requirements analysis with the domain experts. It became apparent that these types of diagrams are limited concerning an adequate mapping of domain requirements. There was a lack of modeling concepts that prevented the fine-grained specification of requirements, which could lead to incompatibilities between the resulting software artifact and the model, which represents the original development objectives. It is generally acknowledged that the prior evaluation of the suitability of an intended modeling language for a specific modeling approach is obligatory. However, when using such an evaluated modeling language, we have determined that lacks in semantic strength can occur later in the modeling process. Hence, we recommend selecting a meta-language and an appropriate set of tools, which allow an ad hoc extension of the modeling language. In addition to the core modeling language, libraries representing domain specific knowledge can help to assure the semantic strength of the models.

Lesson 2 – Many Notations - One Metamodel. In modeling research the structuring of models using views is an established approach to decompose the complexity of model systems [40]. This systematic approach can contribute to the design of an MDA. The cooperation of technical experts and professionals of the domain leads to issues resulting from different terminologies. Stakeholder oriented views, which are integrated and representing the usual terminology of the domain expert, can help to accumulate the overall model. In particular, they support the integration of terminologies from the information technology and the specific domain by defining joint metamodel and different notations. The notation (concrete syntax) is the instrument to represent the specific terminology.

[1] See CDA XML schema definition POCD_MT000040.xsd in the CDA specification

Lesson 3 – Use Existing Ontological Models. When designing domain specific modeling languages existing standards which describe ontological models for the domains should be considered. For example, in medical informatics standards like the CDA already conceptualize partial views of the domain for which a modeling language is specified. These conceptualizations can be used to build an abstract syntax. This ensures the alignment of the modeling language to the mode of expression of the domain on the one hand and facilitates the transformation to the information technology layers on the other hand. In the project we have considered standards like the already mentioned CDA or the Role Base Access Control (RBAC) Model (see [41]) to define a metamodel for different views (e.g. documents or roles and responsibilities of actors). The notation is aligned to the health care professionals' terminology.

Lesson 4 – Separate Software and Context View. The separation of layers in accordance to the MDA was stringently implemented in the project. Software artifacts should be obtained from the CIM. This makes it necessary to describe domain knowledge on the one hand and the non-technical but software-oriented requirements on the other hand. Such a requirement could be: "The telehealth platform must be able to print a discharge report." In the MDA guide, it is stated that the CIM "focuses on the environment [...] and the requirements for the system" [21]. Thus, it seems to be useful to dissect the CIM conceptually into a part, which describes the environment and another part, which formulates the software requirements.

5 Conclusion and Outlook

The objective of this paper was to show how a Model Driven Architecture can be used as a comprehensive approach for deriving domain knowledge to the information system. Therefore, we considered the design and application of a MDA for a health care project. We focused the design of the CIM and its transformation to the PIM. The further steps in direction to PSM are also described on the example of an IT-based CP and its corresponding documents. The experiences we gathered during the design and application of the MDA, we explicated as lessons learned. These lessons can be used as a practical assistance in similar projects.

In summary, the paper shows a way to reduce the semantic gap between the business world and the information technology world by utilizing the MDA, particularly by a view-oriented and integrating design of the CIM layer. The models of CIM are aligned to the language of the domain experts and are also specified with a high grade of formality. The integrated metamodel allows the unique identification of modeled concepts. This is obligatory for an automatic transformation. The domain experts take advantage from the integrated meta-model by getting a comprehensive and always consistent documentation of the part from their domain that is relevant to the information system.

Further research has to be done in design of additional views and their integration to the holistic approach. Concerning lesson 4, it must be analyzed how CIM layer can be dissect and how the concepts are interrelated. In the next step, the resource view

will be specified and the belonging transformers will be implemented. Furthermore, we plan to evaluate the methodology and its economic efficiency by implementing other applications for the telehealth platform using the presented the MDA-approach.

Acknowledgement

This work is part of the EFRE-fostered project „Telehealth Ostsachsen“ supported by the European Union and Free State of Saxony.

References

1. Nagel, E. ed: Das Gesundheitswesen in Deutschland: Struktur, Leistungen, Weiterentwicklung. Deutscher Ärzte Verlag (2007).
2. De Bleser, L., Depreitere, R., De Waele, K., Vanhaecht, K., Vlayen, J., Sermeus, W.: Defining pathways. J Nurs Manag. 14, 553–563 (2006).
3. Küttner, T., Roeder, N.: Definition Klinischer Behandlungspfade. Klinische Behandlungspfade. Mit Standards erfolgreicher arbeiten. 19–27. Deutscher Ärzteverlag, Köln (2007).
4. Panella, M., Vanhaecht, K.: Is there still need for confusion about pathways? Intl J Care Pathw. 14, 1–3 (2010).
5. Winter, A.F., Zimmerling, R., Bott, O.J.: Das Management von Krankenhausinformationssystemen: Eine Begriffsdefinition. Tagungsband 41. GMDS-Jahrestagung. Bonn (1996).
6. Reich, B.H., Benbasat, I.: Factors that influence the social dimension of alignment between business and information technology objectives. MIS quarterly. 81–113 (2000).
7. Object Management Group: OMG Unified Modeling LanguageTM (OMG UML), Superstructure, Ver. 2.4.1, http://www.omg.org/spec/UML/2.4.1/Superstructure/PDF/ (2011).
8. Heß, M.: Towards a Domain-Specific Method for Multi-Perspective Hospital Modelling – Motivation and Requirements. In: Brocke, J. vom, Hekkala, R., Ram, S., and Rossi, M. (eds.) Design Science at the Intersection of Physical and Virtual Design. 369–385. Springer Berlin Heidelberg (2013).
9. Blobel, B., Pharow, P.: A model driven approach for the German health telematics architectural framework and security infrastructure. INT J MED INFORM. 76, 169–175 (2007).
10. Raghupathi, W., Umar, A.: Exploring a model-driven architecture (MDA) approach to health care information systems development. INT J MED INFORM. 77, 305–314 (2008).
11. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Quarterly. 28, 75–105 (2004).
12. Baskerville, R.: What design science is not. EUR J INF SYST. 17, 441–443 (2008).
13. Iivari, J.: A paradigmatic analysis of information systems as a design science. Scandinavian Journal of Information Systems. 19, 5 (2007).
14. Hoffman, R.R., Roesler, A., Moon, B.M.: What is design in the context of human-centered computing? Intelligent Systems, IEEE. 19, 89–95 (2004).
15. Frank, U.: Towards a Pluralistic Conception of Research Methods in Information Systems Research (Arbeitsbericht). Universität Duisburg-Essen, Duisburg-Essen (2006).
16. Gregor, S., Hevner, A.R.: Positioning and Presenting Design Science Research for Maximum Impact. MIS Quarterly. 37, 337–A6 (2013).
17. Alturki, A., Gable, G.G., Bandara, W.: A design science research roadmap. Service-Oriented Perspectives in Design Science Research. pp. 107–123. Springer (2011).
18. Soley, R., Group, O.S.S.: Model driven architecture (2000). OMG white paper. 308.

19. Mellor, S.J.: MDA distilled: principles of model-driven architecture. Addison-Wesley Professional (2004).
20. Reussner, R., Hasselbring, W.: Handbuch der Softwarearchitektur. 2., überarbeitete Auflage. dpunkt.verlag, Heidelberg (2009).
21. Miller, J., Mukerji, J. eds: MDA Guide Version 1.0.1 (2003).
22. Kolominsky-Rabas, P.L., Heuschmann, P.U., Marschall, D., Emmert, M., Baltzer, N., Neundörfer, B., Schöffski, O., Krobot, K.J.: Lifetime cost of ischemic stroke in germany: Results and national projections from a population-based stroke registry the erlangen stroke project. Stroke. 37, 1179–1183 (2006).
23. Berlit, P. ed: Klinische Neurologie. Springer-Verlag, Heidelberg (2011).
24. Hübner, U., Liebe, J.D., Straede, M., Thye, J.: IT braucht Leadership. f&w führen und wirtschaften im Krankenhaus. 31, 388–391 (2014).
25. Balzert, H.: Die Entwicklung Von Software-Systemen: Prinzipien, Methoden, Sprachen, Werkzeuge. Spektrum Akademischer Verlag, Mannheim (1992).
26. Schütte, R.: Grundsätze ordnungsmäßiger Referenzmodellierung. Gabler Verlag (1998).
27. Tolvanen, J.P.: Incremental method engineering with modeling tools: theoretical principles and empirical evidence. University of Jyväskylä (1998).
28. Sinz, E.J.: Modellierung betrieblicher Informationssysteme: Gegenstand, Anforderungen und Lösungsansätze. Tagungsband Modellierung 1998. , Münster (1998).
29. Schlieter, H.: Ableitung von Klinischen Pfaden aus Medizinischen Leitlinien–Ein Modellbasierter Ansatz, (2012).
30. Ferstl, O.K., Sinz, E.J.: Grundlagen der Wirtschaftsinformatik. Oldenbourg, M. (2013).
31. Stahl, T., Völter, M.: Modellgetriebene Softwareentwicklung : Techniken, Engineering, Management. Dpunkt-Verl., Heidelberg (2005).
32. Strahringer, S.: Ein sprachbasierter Metamodellbegriff und seine Verallgemeinerung durch das Konzept des Metaisierungsprinzips. Modellierung. pp. 15–20 (1998).
33. Greiffenberg, S.: Methodenentwicklung in Wirtschaft und Verwaltung. Kovac, H. (2004).
34. Burwitz, M., Schlieter, H., Esswein, W.: Modeling Clinical Pathways-Design and Application of a Domain-Specific Modeling Language. In: Alt, A. and Franczyk, B. (eds.) Tagungsband der 11. Internationalen Tagung Wirtschaftsinformatik. , Leipzig (2013).
35. Dolin, R.H., Alschuler, L., Boyer, S., Beebe, C., Behlen, F.M., Biron, P.V., Shabo, A.: HL7 Clinical Document Architecture, Release 2.0, http://www.hl7.org/implement/standards/product_brief.cfm?product_id=7, (2005).
36. Jouault, F., Kurtev, I.: Transforming Models with ATL. In: Bruel, J.-M. (ed.) Satellite Events at the MoDELS 2005 Conference. pp. 128–138. Springer Berlin Heidelberg (2006).
37. Bernauer, M., Kappel, G., Kramler, G.: Representing XML Schema in UML – A Comparison of Approaches. In: Koch, N., Fraternali, P., and Wirsing, M. (eds.) Web Engineering. pp. 440–444. Springer Berlin Heidelberg (2004).
38. Routledge, N., Bird, L., Goodchild, A.: UML and XML schema. Aust. Comput. Sci. Commun. 24, 157–166 (2002).
39. Yin, R.K.: Applications of case study research. SAGE, Newbury Park (2003).
40. Frank, U.: Multi-perspective enterprise modeling (MEMO) conceptual framework and modeling languages. System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on. pp. 1258–1267 (2002).
41. Ferraiolo, D.F., Sandhu, R., Gavrila, S., Kuhn, D.R., Chandramouli, R.: Proposed NIST standard for role-based access control. ACM Transactions on Information and System Security (TISSEC). 4, 224–274 (2001).