# Bringing Enterprise Modeling Closer to Model-Driven Development

Iyad Zikra, Janis Stirna, Jelena Zdravkovic

# Bringing Enterprise Modeling Closer to Model-Driven Development

Iyad Zikra, Janis Stirna, Jelena Zdravkovic

Department of Computer and Systems Sciences, Stockholm University
Forum 100, SE-164 40 Kista, Sweden
{iyad, js, jelenaz}@dsv.su.se

**Abstract.** Enterprise Modeling (EM) provides the means for using models to represent organizational knowledge from different perspectives. When information systems (IS) are involved, Model-Driven Development (MDD) is an approach that focuses on the use of models as primary development artifacts. By observing that EM provides the context for high level requirements, which in turn are the input to MDD, we propose a meta-model that integrates enterprise models and requirements with design models in MDD. The meta-model defines six models that cover both organizational and IS development knowledge. Inter-model relationships ensure an integrated view of the enterprise and the supporting IS by allowing model components to be used across different models. The integrated meta-model is demonstrated through an example case study.

**Keywords:** Enterprise Modeling, Enterprise Models, Requirements, MDD, Model-Driven Development, MDE, Model-Driven Engineering.

## 1 Introduction

Enterprise Modeling (EM) aims to capture and represent organizational design in terms of business goals, processes, concepts, actors, as well as high level information system (IS) requirements by using conceptual models. Many EM techniques have emerged throughout the years, presenting different views of the enterprise and offering a wide variety of possibilities for designing, improving, re-structuring, and automating all or parts of the business in question.

Model-Driven Development (MDD) is a software development approach where models replace programming code as the primary development artifact. Models in MDD are used for describing the IS design to a level of detail that allows automatic generation of a running system. When used in the context of developing enterprise-level software, MDD has the potential of streamlining the development process. However, the input to the MDD process has to come from a higher level of abstraction, such as organizational designs and requirements, which are often represented by Enterprise Models. Most current MDD approaches implicitly assume that creating the initial MDD model is the responsibility of the modeler and provide little or no guidance for initiating development [22]. The approaches that try to

stablish a connection between preceding modeling efforts and MDD rely on mappings between, e.g. requirements and design models in MDD. They still fall short on achieving full integration and supporting the developers in all development phases.

In this paper, we present an integrated meta-model of MDD and EM, taking Enterprise Knowledge Development (EKD) [2] as an example of a specific EM approach. The aim of this research is to provide a formal connection between the two development activities, thus bridging the gap between designing the organizations and model driven development of information systems. EKD is selected as the candidate EM technique because it includes an overall model composed of inter-related sub-models for integrating different views of the organization. Many concepts in our meta-model are based on the EKD approach. The proposed meta-model is a first step towards a complete and tool-supported MDD process. Combining the high-level business-oriented EKD capabilities with the more concrete and IS-oriented MDD principles can lead to better integration between the business and information systems that support it.

The remainder of the paper is organized as follows: Sections 2 and 3 give a short background to EM and MDD. The proposed meta-model is presented in Section 4 and then demonstrated by an example in Section 5. Section 6 is a discussion of related work, while Section 7 presents concluding remarks and issues for future work.

## 2 Enterprise Knowledge Development

EM is a process where an integrated and negotiated model describing different aspects of an enterprise is created. An Enterprise Model consists of a number of related "sub-models", each describing the enterprise from a particular perspective, e.g. the purpose of the organization, business processes, entities, and structure. For example [12] proposes using the UML notation [14] to model enterprises.

In this research we have chosen EKD as the EM approach to be integrated with MDD. The following six integrated sub-models constitute EKD, each focusing on a different aspect of the enterprise:

*Goals Model (GM)*–the organization's vision and strategy; it addresses questions related to what the organization wants to achieve or to avoid and why.

*Business Rules Model (BRM)*–the business policies and rules; it addresses questions related to business rules and how they support the organization's goals.

*Concepts Model (CM)*–the business ontology and vocabulary; it addresses the things and "phenomena" covered in other sub-models.

*Business Process Model (BPM)*–the procedural aspects of business operations; questions related to business processes and how they handle information and material.

*Actors and Resources Model (ARM)*–organizational structure; addressing the responsibilities for goals and processes and how the actors are interrelated.

*Technical Component & Requirements Model (TCRM)*–IS needs; addressing questions about the business requirements for the IS and how they are related to other sub-models of the enterprise model.

The modeling components of the sub-models are related within a sub-model (intra-model relationships) as well as with components of other sub-models (inter-model

relationships). Inter-model links are particularly useful for developing a holistic view of the enterprise because they allow tracing decisions and components throughout the model. E.g. goals in the GM provide the reasons for including certain processes in the BPM and IS requirements in the TCRM. EKD strongly advocates the use of a participative approach for model development [2, 20], a technique also applicable in agile development projects [19].

Bubenko et al. [2] identify two purposes for EM: 1) developing the business: shaping the business vision and identifying the strategy and means necessary to attain it, including IS development; and 2) ensuring the quality of the business: creating a unified and shared knowledge culture, and gaining the commitment of different stakeholders. In conjunction with business development, EM often provides input to IS development.

## 3 Model-Driven Development

The focus of software development shifted from code to models as a response to the increasing complexity of software and the pressure to shorten development cycles [17]. Technology advancements enabled the transformation of models into complete and functional software, making models an essential part of the final product [5]. The emergent model-centric approaches are commonly called MDD. No unified definition exists for this new domain, which is sometimes referred to as Model-Driven Engineering (MDE) in the literature. There are however common characteristics that distinguish approaches that are labeled as model-driven [4, 8, 16, 17]. Models are the central concept in MDD; they present different views of a system and guide the development process. The structure and semantics of models are formalized in MDD as meta-models, enabling automatic creation of models. The knowledge used to create models is also formalized as transformation rules, which are based on the language of the models when the transformed models are in the same language or on the meta-models when the models are in different languages. The process for creating and managing the models, meta-models, and transformations is identified by [8] as part of MDD. Supporting tools are consistently highlighted in the MDD literature as an essential part, even though many MDD approaches have partial or no tool support. [10, 22]

The interest in MDD is growing despite the existence of many unresolved issues and the limited tool support. Creating the initial MDD model has been identified as a problem in [10] and [22]. The problem manifests itself in a gap between IS design models and higher levels of abstraction. Integration properties were proposed in [22] to establish a connection between requirements and MDD. By observing that EM is a practicable instrument for capturing high level requirements needed as the input to MDD, we propose an overarching meta-model that integrates EM and MDD. The next section describes this meta-model.

## 4 A Unifying Meta-Model

The main objective for developing the meta-model is to provide a unified platform for designing enterprise models, which are then used to derive IS models that can subsequently be used to generate a functioning and complete system using an MDD approach. The meta-model defines multiple complimentary models, offering a holistic view of the organization and enabling automatic generation of an IS that is described by the relevant models. The meta-model defines (1) models representing enterprise knowledge (enterprise-level models) based on the EKD approach, namely EKD's GM, CM, BPM, and BRM, as well as (2) system-level models, namely *Requirements Model (RM)* representing IS requirements and *IT Architecture Model (ITAM)* describing the technical components and user interfaces that are involved in the implementation of the IS.

Relationships between the enterprise-level and system-level models are formalized to support evaluation of the system models and improve traceability to their origins in the organizational design. Coarse-grained relationships give a general overview of the interactions between the models (Fig. 1). Fine-grained relationships, called inter-model relationships, relate model components across different models. They depict the use of concepts of one model in other models and present a more complete view of both the enterprise and the IS. Complete traceability is supported by the meta-model without the need to introduce additional explicit traceability links. The fine-grained inter-model relationships are further discussed in the following subsections.

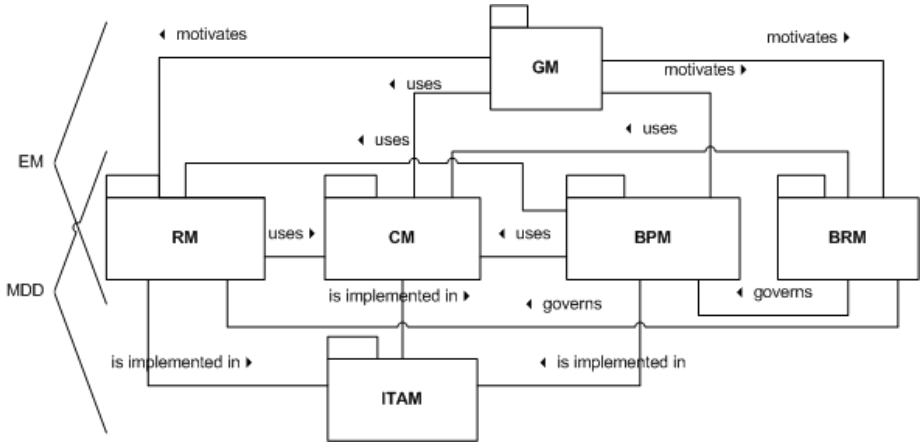


Fig. 1. Relationships between the sub-models in the meta-model

The meta-model in this paper follows a simple UML-like notation. Package symbols are used to denote models in Fig. 1, implying that concepts defined in the meta-model are spread over the six models. Furthermore, generalization links between concepts denote a general-specific relationship, while cardinalities on the relationships express the number of model component instances that can participate in a model relationship instance.

### 4.1 Common Components Model

All models described by the meta-model are derived from common components that provide the basis for other components. The Common Components Model (CCM) is shown in Fig. 2. It is not included in Fig. 1. because it is a conceptual abstraction that spans all other models. The *model component* is the topmost concept in the meta-model. Each model component has a unique identifier and a text field that allows the component to be labeled with a single name, a sentence, or a long text depending on the modeling needs. A modeling component has a *description*, which is a text that provides additional clarification for the component.

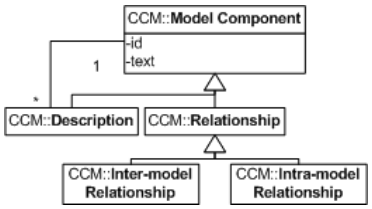


Fig. 2. The Common Components Model

The *relationship* concept is a model component that connects other model components with each other. Two distinct types of relationships are defined in the meta-model: *intra-model relationships* that link components within the same model; and *inter-model relationships* that enable components from different models to be related with each other. The inter-model relationships facilitate traceability among the models and provide mechanisms to design intersecting models.

### 4.2 Concepts Model

Concepts that are necessary to describe the static aspects of enterprises and information systems are modeled in the CM (Fig 3). They include resources and information objects that are used, processed, exchanged, produced, and stored in the organization, together with their relationships and attributes.

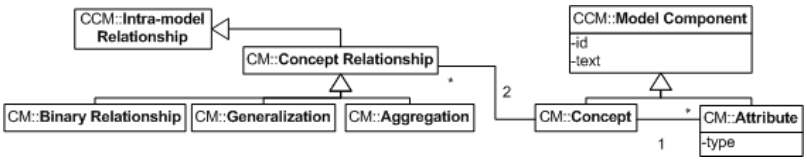


Fig. 3. The Concepts Model

A *concept* represents entities about which the enterprise stores or processes information. Concepts represent resources, information objects, or other things that are of interest to the enterprise. They are described by *attributes* that declare properties for the concepts. Concepts can be related to each other using a *concept relationship*, which can be one of three kinds: the *binary relationship*, which is a

general kind of relationship; the *generalization* relationship, which relates a general concept to a more specific one; and the *aggregation* relationship, used to indicate that a concept is composed of other concepts.

### 4.3 Goal Model

Organizational *business goals* are recorded and represented in the GM (Fig. 4). A business goal is a future state-of-affairs that the enterprise aims to attain, and through which it can grow and generate profit. An enterprise can identify potential desirable situations as *opportunities*, which highlight new possibilities or capabilities that can be transformed into actual business goals. Both opportunities and business goals are defined as types of *intentional components* because they share many properties. Moreover, modeling opportunities as intentional components allows the identification of concepts, roles, processes, and requirements; otherwise associated only with business goals. Intentional components can *support* each other, indicating that achieving one contributes to achieving the other. Intentional components can also *conflict* with each other when the realization of one challenges the realization of the other. The role which *defines* an intentional component is captured in the meta-model to provide traceability to the source of the component.

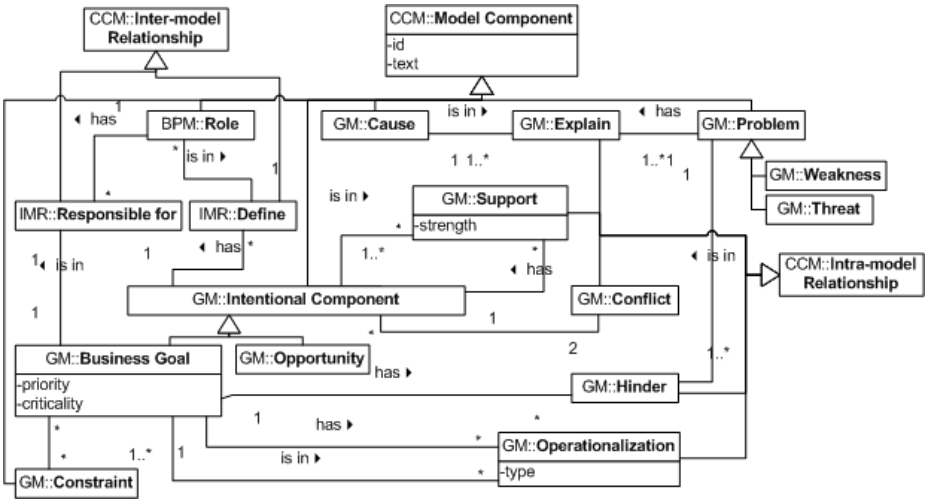


Fig. 4. The Goal Model

The *operationalization* relationship provides additional structure to the GM by allowing business goals to be decomposed into smaller, more concrete sub-goals. Decomposition can occur in one of three types, or modes: AND operationalization, indicating that the fulfillment of all sub-goals is necessary to fulfill the goal; OR operationalization, when the fulfillment of at least one sub-goal is enough to fulfill the goal; and XOR operationalization, when sub-goals are exclusive alternatives for the goal. Operationalization enables organizing the intentional components as a hierarchy.

Goals have roles that are *responsible for* them; tracking the progress of their fulfillment and making sure necessary resources are allocated for that purpose.

Achieving the business goals is usually hindered by various obstacles, and it pays to include those obstacles in the model to provide a clearer view of the organizational landscape. *Problems* that *hinder* business goals can be either internal to the enterprise, in which case they are considered *weaknesses*, or external, in which case they are modeled as *threats*. The *cause* that *explains* a problem is a useful insight when identified explicitly, and can contribute to finding suitable measures and solutions. In addition, a business goal is bound by *constraints*, which represent rules and regulations that affect how the organization operates. Constraints are always external to the organization; internal rules and regulations are described using the BRM.

### 4.4 Business Process Model

Business goals identified in the GM give rise to, or *motivate*, the design of business processes that describe activities in the enterprise needed to realize the goals. The BPM (Fig. 5) provides a view over the processes and their composition and structure. A *process* model component stands for different sizes of processes at both the business- and IS-levels, thus providing a unified dynamic view of the enterprise and its IS. The relationship between a process and its sub-processes is captured as a *composition* relationship, indicating that the sub-processes work together to accomplish the top process. The meta-model includes no limit to the number of decomposition levels, which is left to the specific needs of projects.

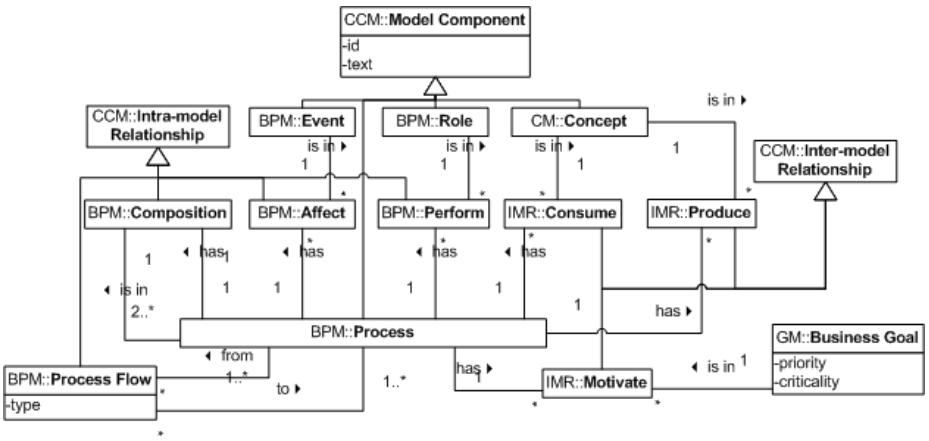


Fig. 5. The Process Model

The flow between processes is described using the *process flow* relationship, which connects processes in one execution flow. The type of the relationship indicates whether the processes are performed in parallel (AND connection), optional (OR connection), or conditional (XOR connection). Processes are affected by *events*, which are external occurrences that influence the execution of the process and cause it to deviate into certain paths, e.g. at the decision points. Concepts that are *consumed*

and *produced* are included in the BPM using inter-model relationships, denoting the inputs that guide the process execution and outputs that result from the execution.

Actors that *perform* processes in the enterprise are modeled as *roles*, which can also *provide* and be *responsible for* goals, and can be related to requirements. A role stands for a general position that is independent of the actual persons filling it, and it can represent physical persons, virtual persons, or automated systems.

### 4.5 Requirements Model

High-level requirements, also called business requirements, express the stakeholders' desires for a future IS. Business requirements are refined into more concrete IS requirements that are better understood by system designers. The line separating business requirements and system requirements is vague and hard to identify, but the decomposition is always necessary. Therefore, the meta-model for the RM (Fig. 6) includes a single *requirement* component that serves as a high-level as well as a concrete system requirement. *Decomposition* of a requirement occurs on any level, and continues depending on the judgment of the modeler and the specific needs of the project. The decomposition relationship is used to connect a parent requirement with its child requirements, which can be all necessary (AND decomposition), alternatives (OR decomposition), or exclusive alternatives (XOR decomposition). Requirements that negatively affect the realization of each other are connected using the *conflict* relationship. Also, requirements that positively affect the realization of each other are connected using the *dependency* relationship.

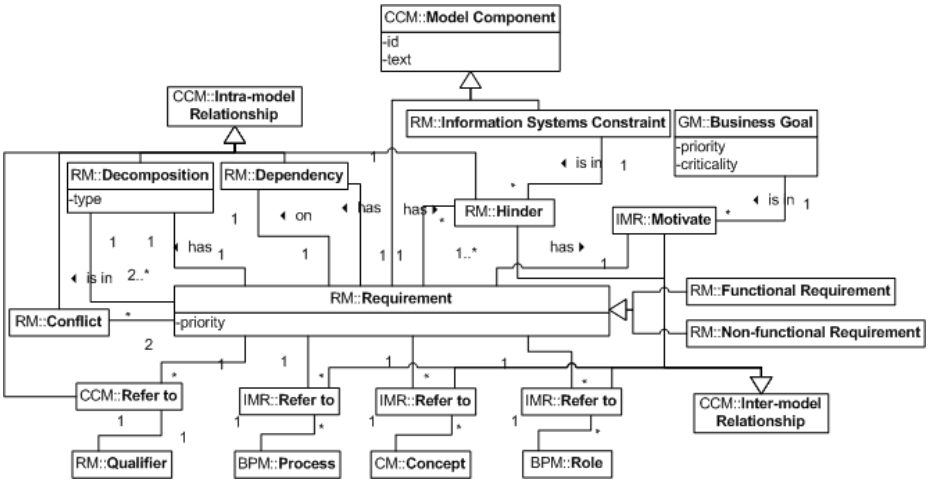


Fig. 6. The Requirements Model

Requirements at different levels of decomposition are *motivated* by business goals. However, some requirements address system related issues that are not relevant to the high-level organizational goals. When those requirements are elicited, *information systems constraints* that motivate or *hinder* them must be identified and assessed.

Requirements in our meta-model provide a central connection point between the models. In the modeling process, actors, things, and activities that are involved in a requirement are identified and the corresponding roles, concepts, and processes are linked with the requirement, respectively. Requirements have qualifiers that express additional information about certain qualities a requirement can have, e.g. performing the functionality at certain times or periodically, or the location for storing data.

### 4.6 Business Rules Model

Internal rules and regulations that govern the enterprise provide boundaries for the concepts and business processes. Business rules are often formulated together with the business goals to specify how the goals will be achieved. Concepts and business processes that are motivated by the business goals are governed by the defined business rule, and this is captured in the BRM (Fig. 7).

*Business rules* are represented in the meta-model as model components that are *motivated* by business goals. In other words, the business rules affect the fulfillment of business goals. A business rule *refers to* one or more concepts that are constrained by it. When a rule defines a necessity that needs to be guaranteed at all times by the involved concepts, it is called a *structural business rule*. When a rule addresses derived or dynamic properties that must be checked at certain points in time or when certain events occur, it is called a *behavioral business rule*. This type of business rules constraints the enterprise in terms of the change of its state, and can lead to different results depending on the triggering events.

Breaking a structural business rule produces an invalid state for the enterprise, and hence must be prevented. However, it is possible to break behavioral rules, and such breaches entail corrective action that returns the enterprise to a valid state. While structural rules *constrain* concepts, behavioral rules *constrain* processes, and can *motivate* the design of additional processes that are needed to enforce the rules.

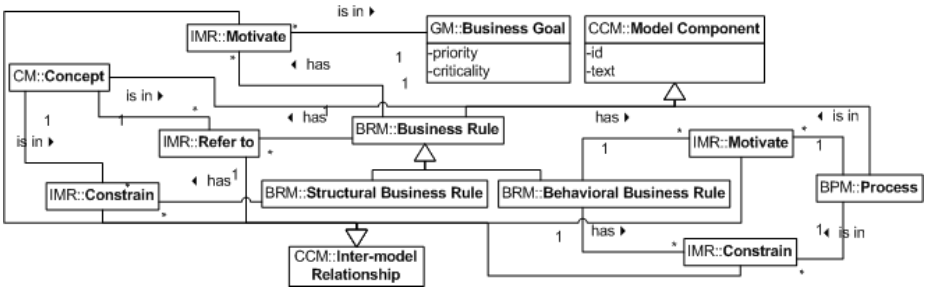


Fig. 7. The Business Rules Model

### 4.7 IT Architecture Model

Creating a complete IS model involves describing the design architecture that the final system will be based on, and how its different parts will operate conjointly. It

also involves describing how eventual users will interact with the system, and other technical details related to the MDD platform used for implementation. This information is captured by the ITAM, which is technology dependent and can exist in various forms for the same GM, CM, BPM, and RM. The ITAM must support inter-model relationships between components of the mentioned models and specific architectural components, depending on the selected implementation architecture. Those relationships highlight the motivation behind architectural design decisions, and provide additional information that can be exploited when transforming the models into an executable system.

## 5 An Example Case Study

To demonstrate the use of the proposed meta-model in an MDD setting, we use an example case inspired by the project Energy Efficiency and Risk Management in Public Buildings (EnRiMa)[1]. The overall goal of the project is to develop a decision support system assisting building managers in making smart strategic and operational decisions considering a multitude of factors, e.g. occupant needs, market prices, installed technology, environmental factors, and weather.

The aim of this example is to demonstrate the use of the proposed meta-model starting from high-level enterprise models. First, some business goals are elicited in the context of the EnRiMa project. Other models are then developed from the business goals according to the meta-model. Parts of the models are identified as candidates for IS support and hence realized as a Web services implementation.

The business goals used in this example are: 1) satisfy comfort requirements of occupants, 2) reduce $CO_2$ emissions, 3) increase energy efficiency of the building, 4) enable daily energy adjustments, and 5) balance long- and short-term plans. Fig. 8 shows the GM that describes the goals and their relationships.

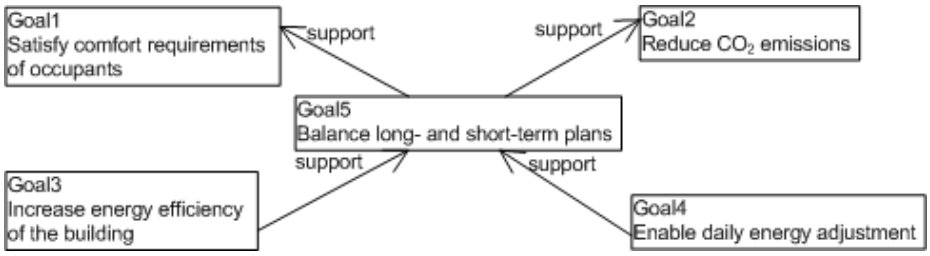


Fig. 8. Example Goal Model.

Goal 4 motivates a process whereby hourly readings of wind speed, temperature, and humidity are collected and compared with historical data to predict a short-term weather forecast. The forecast is then projected on current and prospective energy prices in the market, a step motivated by Goal 3. This process is described using a BPM (Fig. 9) that includes top-level processes and shows the different inputs and

[1] http://www.enrima-project.eu/

outputs involved. Modeling of processes can continue in a real situation by breaking down each process into its constituent parts, until a suitable level of granularity is reached. Each part is itself a process with its own inputs and outputs. Inter-model relationships allow the inclusion of business goals in the process model, providing traceability support between processes and their originating goals.

By identifying processes which merit automation, the requirements describing the information systems that support those processes can be elicited. For example, Process 4 can be implemented as Web services that gather the necessary information and calculate the adjustments to the heating system parameters. The requirements that describe the system for executing Process 4 can be formulated as: 1) the system shall retrieve current energy prices; 2) the system shall retrieve prospective energy prices; and 3) the system shall deploy a predefined set of formulae to calculate the adjustments. These requirements can be decomposed into more fine-grained ones, and, as with processes in the BPM, the decomposition can continue until a desirable level of granularity has been reached. The resulting RM is shown in Fig. 10. The business goal motivating the requirements and the process which is related to the requirements are present in the RM using inter-model relationships.

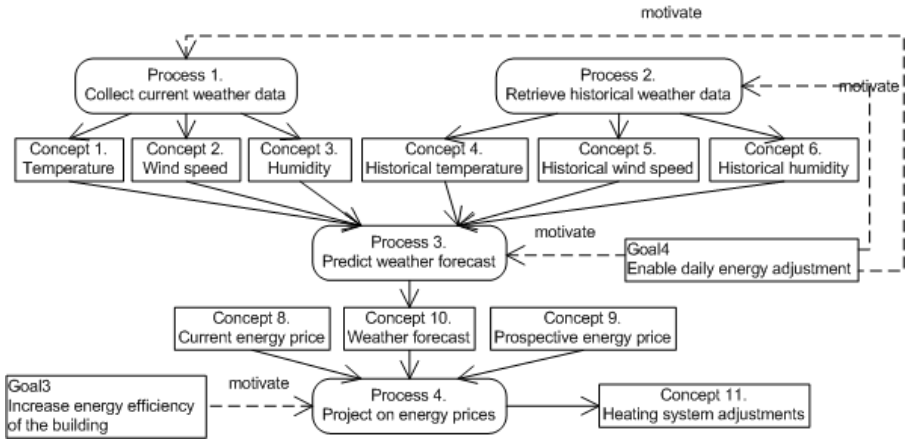


Fig. 9. Example Business Process Model.

Concepts defined in the BPM and RM are part of a larger CM that describes all relevant concepts needed for building management together with their relationships.

The information encoded in the models discussed so far is sufficient to enable an automatic generation of a service-based system that supports the identified processes and business goals. Each functional requirement in the RM in Fig. 10 is a candidate for becoming a Web service. In this example, manual intervention is needed to select suitable requirements for becoming Web services. However, mapping requirements and other modeling components will be facilitated using an ITAM which defines the relationships between eventual components of the final system. Inter-model relationships between components in the ITAM and components of other models will help formalize the transformation of different modeling components into databases, Web services, user interfaces, or any other type of IS component.

Functional Requirement 3 (see Fig. 10) is a function suitable for transformation into a Web service. Depending on the chosen granularity for modeling the processes and the requirements, the models can be used to generate simple Web services that include only one operation or more complex services that combine multiple operations. The approach provides flexibility for making such decisions, and eventual supporting tools must enable modelers to choose a suitable mapping to simple operations or complex services (using the ITAM). Moreover, the inputs and outputs of the Web service can be derived from the inter-model relationships between requirements and concepts. While these relationships show only the concepts that are related to requirement, referring to the processes that are related to the same concepts can help in identifying whether a concept is an input or an output for the Web service. An excerpt of the generated WSDL code for the Web service definition is shown in the code snippet below. The XML schema definition referred to in "concepts.xsd" corresponds to the concepts defined in the CM.

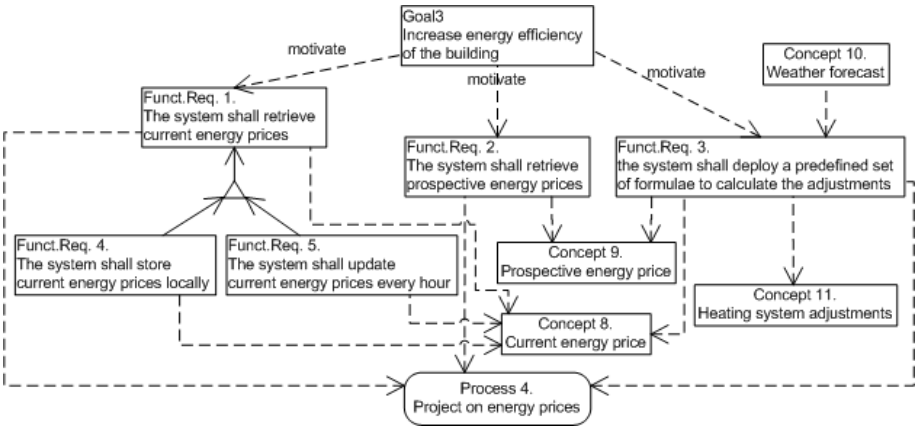


Fig. 10. Example Requirements Model.

Code snippet of the generated WSDL code defining the Web service that corresponds to Functional Requirement 3 (see Fig. 10).

```
<definitions name="CalculateHeatingSystemAdjustments"
targetNamespace="http://example.com/calcadjust.wsdl"
xmlns:xsd1="http://example.com/concepts.xsd"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  ...
  <wsdl:message name="GetAdjustmentsInput">
    <wsdl:part name="WeatherForecast"
element="xsd1:WeatherForecast"/>
    <wsdl:part name="CurrentEnergyPrice"
element="xsd1:CurrentEnergyPrice"/>
    <wsdl:part name="ProspectiveEnergyPrice"
element="xsd1:ProspectiveEnergyPrice"/>
  </wsdl:message>
```

```
  <wsdl:message name="GetAdjustmentsOutput">
    <wsdl:part name="HeatingSystemAdjustments"
element="xsd1:HeatingSystemAdjustments"/>
  </wsdl:message>

  <wsdl:portType name="AdjustmentsPortType">
    <wsdl:operation name="GetAdjustments">
      <wsdl:input message="GetAdjustmentsInput"/>
      <wsdl:output message="GetAdjustmentsInput"/>
    </wsdl:operation>
  </wsdl:portType>
  ...
  <wsdl:service name="HeatingSystemAdjustmentsService">
    <wsdl:port name="AdjustmentsPort" binding="someBinding">
      <soap:address
location="http://example.com/stockquote"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

## 6 Related Work

Lin and Sølvberg [9] present a framework for motivating process models using goal annotations. Ontologies were used to annotate process and goal models on the meta-level, establishing connections between processes and business goals that motivate them. Shahzad et al. [18] propose a generic meta-model for business processes that combines functional, behavioral, organizational, and informational perspectives. The meta-model is used, together with a generic formal process description, to create a process model repository that enables process model reuse.

Attempts to provide an integrated view of IS requirements using meta-models can be traced back to Jordan and Davis [7]. Recent meta-models for requirements management in software product families are found in Arpinen et al. [1] and Cerón et al. [3]. The meta-models include relationships between requirements and other organizational entities, such as actors and contracts. Coarse-grained relationships to other parts of the IS development lifecycle, such as system (design) models and test cases, are also present. The meta-model proposed by Goknil et al. [6] unifies common concepts in existing requirements modeling approaches. Requirements are described using their properties, types, and relationships with other requirement-related concepts, such as test cases and stakeholders. López et al. [11] treat design models as instances of semi-formal requirements models. Use case diagrams, activity diagrams, and workflow diagrams are unified on the meta- level using common modeling units.

While the approaches above include unifying meta-models for IS development, they focus on a single aspect of the final system (processes in [9] and [18], and requirements in the others). In contrast, our approach covers a wider range of organizational and IS concerns.

Pastor and Giachetti [15] present a generic process for linking i* [21] and the OO-Method—as representatives of Goal-Oriented Requirements Engineering (GORE) and

MDD, respectively. By basing the linked meta-models on EMOF [13], the authors are able to define an integration meta-model and generate model transformation rules that automatically produce MDD-compliant conceptual models from i* models. However, additional transformations are necessary to produce the final system. Our approach advocates a single-step transformation that eliminates the need for intermediate steps and helps in generating the final system directly from the models.

## 7 Conclusion

EM aims at creating a structured and unified view of an enterprise, enabling more informed and accurate decisions to be made. MDD is an approach to IS development that focuses on models as drivers of the development process and part of the final product. In this paper, we have proposed a meta-model that spans EM and MDD to give an integrated view of organizational and IS concerns. EKD was chosen as the EM approach to be integrated with MDD because it includes inter-related sub-models that cover different aspects of the enterprise. The meta-model was designed to support the integration properties suggested in [22] for bridging the gap between requirements and MDD design models. Namely, the following properties are supported:

- Static and dynamic aspects of enterprises and information systems are captured using the CM and the BPM;
- Intentional aspects are addressed in the GM, which offers a high-level view of the enterprise aims and lays the context for other models;
- Architectural aspects are captured using the ITAM, which relates components of the different models to specific implementation platform components;
- Change propagation and traceability are supported by inter-model relationships, which enable the components of one model to be included in other models, offering an integrated view of all models.

By integrating enterprise models and IS design models in a single meta-model, the need to transform models between multiple levels of abstraction—a common practice in MDD approaches—is eliminated, leaving only a single transformation step towards the implementation platform. This single-step transformation is further facilitated by the ITAM, which defines the targets that model components will be transformed into.

The meta-model was demonstrated by an example case. However, a larger and more detailed case study is necessary to show the full potential of the meta-model and to uncover the weaknesses in need for improvements. As future work, we will develop an ITAM for a specific implementation architecture, as well as a supporting tool for realizing the case study and for demonstrating the automation that stems from following the MDD principles.

## References

1. Arpinen, T., Hämäläinen, T. D., Hännikäinen, M.: Meta-Model and UML Profile for RequirementsManagement of Software and Embedded Systems. EURASIP Journal on Embedded Systems 2011, Article ID 592168, 14 pages.

2. Bubenko J.A., jr., Persson, A., Stirna, J.: D3: User guide of the Knowledge Management Approach Using Enterprise Knowledge Patterns. Royal Institute of Technology (KTH) and Stockholm University, Stockholm, Sweden (2001).
3. Cerón, R., Dueñas, J. C., Serrano, E., Capilla, R.: A Meta-model for Requirements Engineering in System Family Context for Software Process Improvement Using CMMI. In: Product Focused Software Process Improvement, LNCS 3547, pp. 291–313. Springer Berlin / Heidelberg (2005).
4. France, R., Rumpe, B.: Model-Driven Development Of Complex Software: A Research Roadmap. In: Future of Software Engineering. pp. 37–54. FOSE '07, IEEE Computer Society, Washington, DC, USA (2007). Springer Berlin / Heidelberg (2002).
5. Gašević, D., Djurić, D., Devedžić, V.: Model Driven Engineering. In: Model Driven Engineering and Ontology Development, pp. 125–155. Springer Berlin Heidelberg (2009).
6. Goknil, A., Kurtev, I., van den Berg, K.: A Metamodeling Approach for Reasoning about Requirements. In: Model Driven Architecture – Foundations and Applications, LNCS 5095, pp. 310–325. Springer Berlin / Heidelberg (2008).
7. Jordan, K. A., Davis, A. M.: Requirements Engineering Metamodel: An Integrated View of Requirements. In: Computer Software and Applications Conference, 1991, pp. 472–478. IEEE (1991).
8. Kent, S.: Model Driven Engineering. In: IFM '02, LNCS, vol. 2335, pp. 286–298.
9. Lin, Y., Sølvberg, A.: Goal Annotation of Process Models for Semantic Enrichment of Process Knowledge. In: CAiSE 2007, LNCS 4495, pp. 355–369. Springer Berlin / Heidelberg (2007).
10. Loniewski, G., Insfran, E., Abrahão, S., A Systematic Review of the Use of Requirements Engineering Techniques in Model-Driven Development. In: MODELS 2010, Part II, LNCS 6395, pp. 213–227, 2010. Springer-Verlag Berlin Heidelberg (2011).
11. López, O., Laguna, M. A., García, F. J.: Metamodeling for Requirements Reuse. In: Workshop em Engenharia de Requisitos, WER 2002, pp. 76–90. Valencia, Spain. (2002).
12. Marshall, C.: Enterprise Modeling with UML: Designing Successful Software Through Business Analysis. Addison–Wesley, Essex, UK (2000).
13. Object Management Group (OMG), Meta Object Facility (MOF) Core Specification Version 2.0, 2006.
14. Object Management Group (OMG), Unified Modeling Language (UML) 2.0, 2005.
15. Pastor, O., Giachetti, G.: Linking Goal-Oriented Requirements and Model-Driven Development. In: Intentional Perspectives on Information Systems Engineering, pp. 257–276. Springer Berlin Heidelberg (2010).
16. Schmidt, D.C.: Model-Driven Engineering. Computer 39, 25–31 (2006).
17. Selic, B.: The Pragmatics of Model-Driven Development. IEEE Software 20, 19–25 (2003).
18. Shahzad, K., Elias, M., Johannesson, P.: Towards Cross Language Process Model Reuse – A Language Independent Representation of Process Models. PoEM 2009, LNBIP 39, pp. 176–190. Springer Berlin Heidelberg (2009).
19. Stirna J., Kirikova M.: Integrating Agile Modeling with Participative Enterprise Modeling. In: EMMSAD 2008, pp. 171–184. CEUR (2008).
20. Stirna, J., Persson, A. and Sandkuhl, K.: Participative Enterprise Modeling: Experiences and Recommendations. In: CAiSE 2007, LNCS 4495, pp. 546–560. Springer Berlin / Heidelberg (2007).
21. Yu, E.S.K.: Modelling Strategic Relationships for Process Reengineering, PhD Thesis, University of Toronto, Toronto, Canada (1995).
22. Zikra, I., Stirna, J., Zdravkovic, J., Analyzing the Integration between Requirements and Models in Model Driven Development. In: EMMSAD 2011, LNBIP 81, pp. 342–356, 2011. Springer-Verlag Berlin Heidelberg 2011.