# Software support for the Computation Independent Modelling in the MDA context

Andreas Martin, Peter Loos

Institute for Information Systems (IWi)
at the German Research Center for Artificial Intelligence (DFKI),
Stuhlsatzenhausweg 3,
66123 Saarbruecken, Germany
{Andreas.Martin|Peter.Loos}@iwi.dfki.de

**Abstract.** Common MDA development tools concentrate on the Platform Independent Level (PIM) as starting point for the software development process. Here, we propose a new language to support the CIM modelling one abstraction step above PIM and present a GMF based editor supporting the MDA process on a higher, more business oriented level than the most used UML.

**Keywords:** Business processes, business process software support, Model Driven Architecture (MDA), Computation Independent Modelling (CIM), Graphical Modelling Framework (GMF)

## 1 Model driven architecture

The idea of MDA [OMG01], [MSUW04] is the translation of information models via different steps into finally executable code. According to the definition of MDA the process of creating software starts with information models on computation independent level (CIM) and transforms them into models on a platform independent level (PIM). These models on PIM level are enriched and then transformed into platform specific models (PSM) which after the last transformation result in executable source code [FeLo03], [OMG01]. By a separation of concerns through creating CIMs and PIMs before PSMs you reach a kind of interdependency from platforms, languages and systems. The transformation starts on a highly abstract level and gets more concrete with each step down. ERP systems which support business processes are a very good example how to use MDA. Domain experts can be much more included in the software development process to bridge the gap between real requirements and the understanding of them by a software engineer.

## 2 Conceptual modelling language for MDA-support of business processes

A special language is needed to integrate domain experts which are involved in a business process and their knowledge into the software development. It has to be a trade off between the “non-IT-oriented” people who are the target users of a new system and the people who implement it. So, there are two main requirements:
The language has to be as simple and commonly understandable as necessary and simultaneously as concrete as possible with the intention to automatically create programs.

In order to create such a common language different established languages and notations were explored. For example the Business Modelling Notation (BPM), the Business Process Execution Language (BPEL) and the Event driven Process Chain (EPC) have been analysed. It turned out that each has its advantages and disadvantages. EPCs are a multi-purpose standard for a high level design of application systems or organisational structures, but they are not focused on transformation into an IT system. BPEL is a machine readable, textual format without an easy-going way to sketch a process in a graphical way. BPMN on the other hand is a graphical notation for describing business processes with a weak semantic and no strong formal definition, but, based on our research, it offers the best starting point for creating the new language.

So, the BPMN meta model was used as a basis and enriched by different items, partly gathered from other languages as well as newly created. For example we introduced the concept of business rules, but it showed that, on conceptual design level, we only need decision rules and constraint rules [ScWe05].

Beyond the business rule view we added three other views, namely a process view, a data view and an organizational view. The most extensively represented one is the process view, which integrates the other views. It consists of eight parts each of which describes a particular element of the meta model. Structuring objects are the top-level class the model contains. The control flow section introduces lane elements, which are connected to each other by flow objects. The connections section has two further types of connections between objects in a model. The annotated elements part of the meta model shows model objects that are used to enrich activity objects with relevant information. The activity section tells which elements are representing actions in a model. The events section describes different types of events which describe what kind of triggers could be used in a model. Gateways explain how decisions of different types could be integrated into a VIDE CIM model. The enumerations section is the last section and it gives an overview of the complex types used in three different classes. Data view and organizational view introduce interfaces to two further business process analysis scopes. Figure 1 shows the graphical meta model of the proposed CIM language.

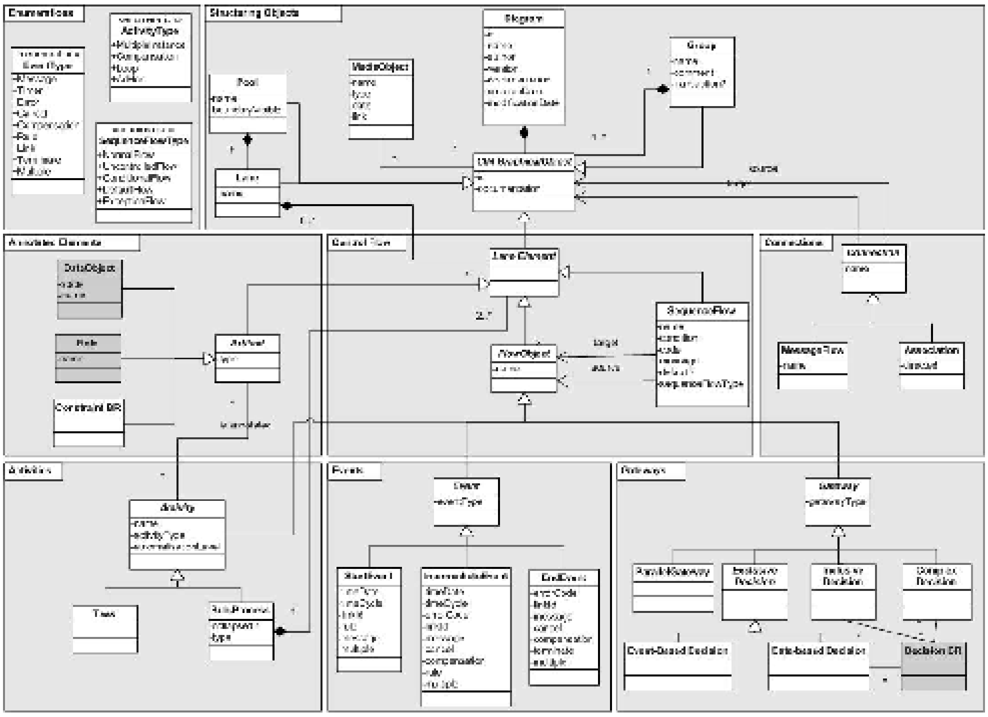

Figure 1: CIM meta model

A completely new item is the media object. It represents unstructured data that can be attached to every element in the diagram. Media objects could be hand-written texts, recorded audio data of interviews or videos enriching the background knowledge of the model element they are annotated to. So, an analyst or programmer has always the possibility to reconsider the origin of a software requirement.

## 3 The Eclipse Graphical Modelling Framework (GMF)

The Graphical Modelling Framework GMF [GMF08] is part of the Eclipse project and based on the Eclipse Modelling Framework (EMF) and the Graphical Editing Framework (GEF). With the help of GMF it is possible to define the meta model of a language and a syntactical representation and a bridge between them. Then you are able to generate a model editor tool out of the definitions.

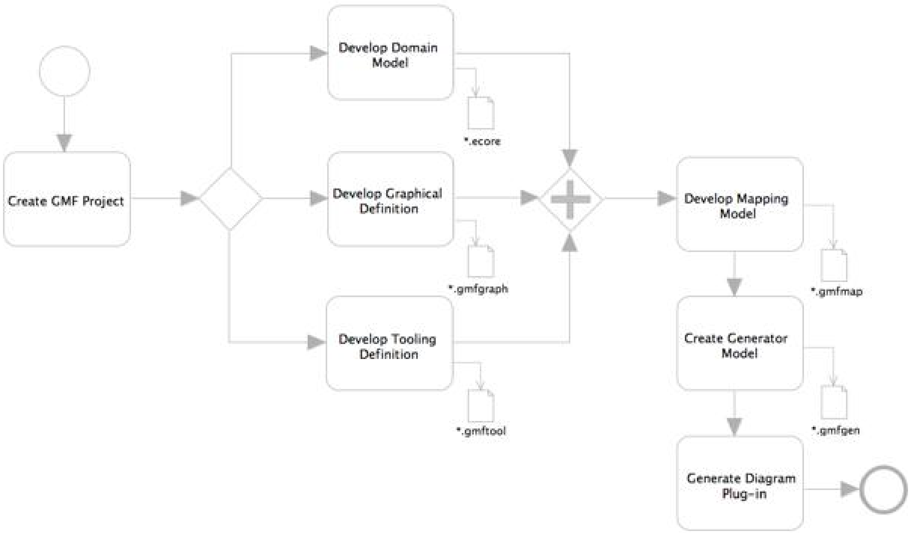

Figure 2: GMF components [GMF08a]

Figure 2 illustrates the main components and models used during GMF-based development. First a domain model and graphical definition is developed, optionally you can define some tooling information for toolbars or menus.

When the appropriate mappings are defined, it is possible to add some implementation details for the generation phase inside a generator model. At the end an editor plug-in based on the generator model can be automatically created. The editor will bridge the notation and domain model when a user is working with a diagram, and also provides mechanisms for the persistence and synchronization of both.

## 4 CIM Editor

Based on the meta model in chapter 2 and the symbols of the Business process modelling notation (BPMN) we created a software tool to support the CIM level modelling in the MDA context. The editor offers four kinds of model views: Process view, data view, organizational view and business rules view, which can be all interconnected (see Figure 3).

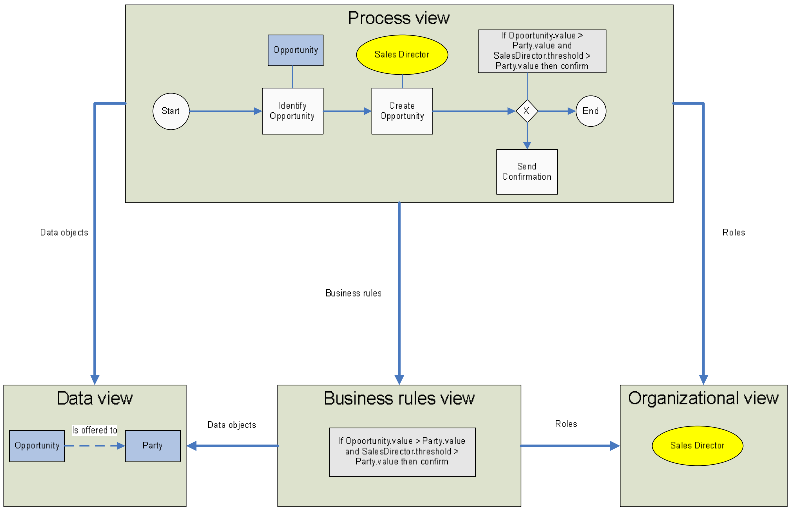

Figure 3: Editor views

## 4.1 Process view

The CIM editor follows the classical GMF editor design. In Figure 4 you can see the main process editor window. It is divided into different parts known from Eclipse, in particular there are the package explorer, the main editor window, the palette with model elements and some additional views at the bottom of the window.

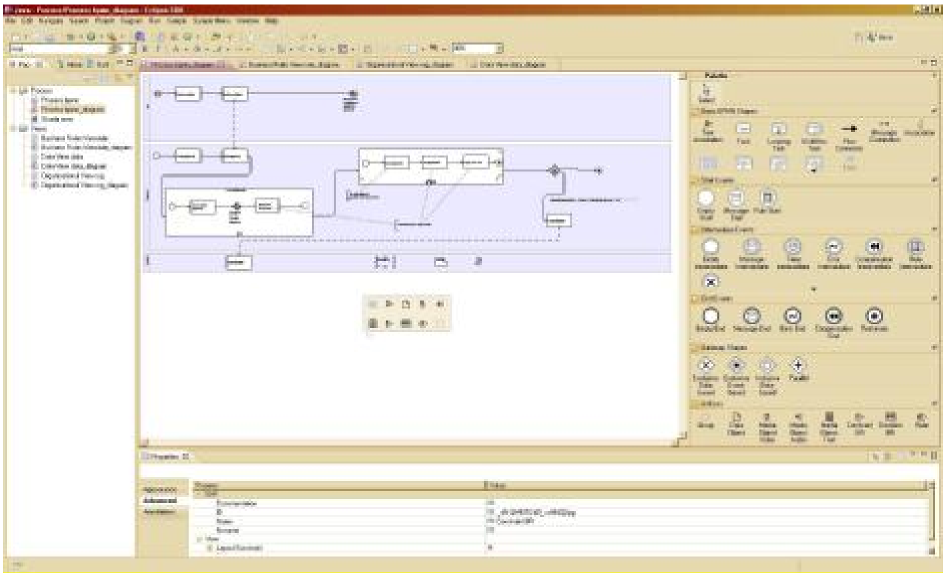

Figure 4 Main process editor window

Inside the package explorer on the left side you can create new models or open existing files. All kind of diagram types like process models, business rule definitions and data or organizational view are shown there. Also attached files like video movies or office documents are included in the file tree to work with inside the models.

Central in the middle of the window the main editor window is located. This is the main work area to build the models. All opened files are shown as tabs on top of the section. So you can change easily between different models and work for example parallel on a workflow and the structure of the data elements. At the right side of window there is always the palette. All model elements you can add to your model are presented there. To do this drag the wished symbol to the main editor window and drop it into your model. In case of the process editor there are all standard BPMN elements and the additional language elements.

At least there is the properties view at the bottom of the window. If you choose an element in the main editor window, all corresponding textual information is shown and editable in this view. A right-click on the editor pane will bring up the diagram context menu. There, there are the functionalities you can utilize to conveniently work with the diagram, e.g. selecting, zooming or arranging.

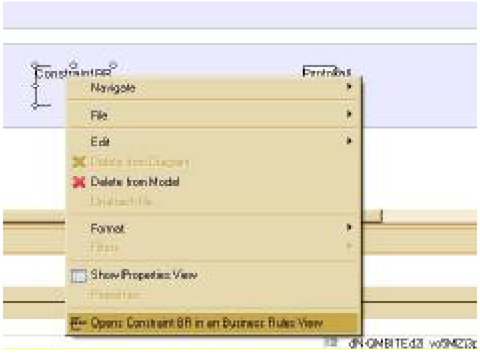

Figure 5 Process editor – Business rule context menu

A right-click on certain model elements offers some navigation functions. So, if you choose a business rule object inside a process the specific context menu in Figure 5 is shown. Apart from standard functions like edit, delete or show properties view there is a possibility to open this rule in its rule view. In the newly opened view you can see the rule's constructs. You decide whether you want to edit it in the rule editor and then go back to the process editor to continue modelling.

Additionally there is a feature that you can utilize in your CIM model that allows you to bring different media files in connection with CIM objects. You can attach binary files to every object in the model. It suffices to drag the needed file into the editor pane and place it on the target object. If you make a right-click on such an attached objects it brings up a context menu with a specific function to open the attached object. So, if you choose the default editor option from the context menu to open the attached file, according to your system properties the default application will be started, for example MS Word for *.doc- files or MS Media Player for *.wmv files(see Figure 6).

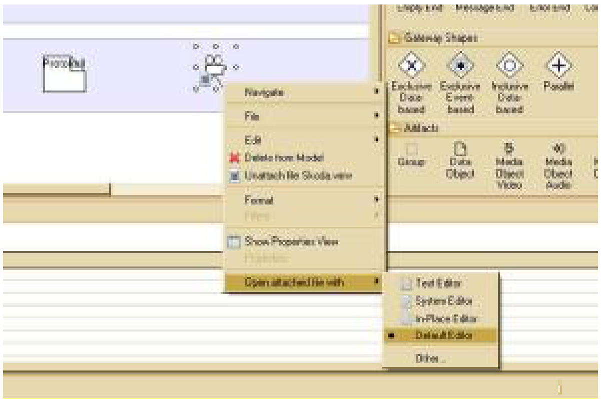

Figure 6 Attachments

### 4.2 Business rule view

The business rule editor consists mainly of the editor pane and the palette to the right of the editor pane (see Figure 7). A business rule is represented through a box, in which you can insert keywords like IF, THEN, ELSE, logical connectors like AND, OR, XOR, terms and verbs. You can also use data objects and attributes defined previously in the data model by inserting them through “Create shortcut…” context menu function in the editor pane.

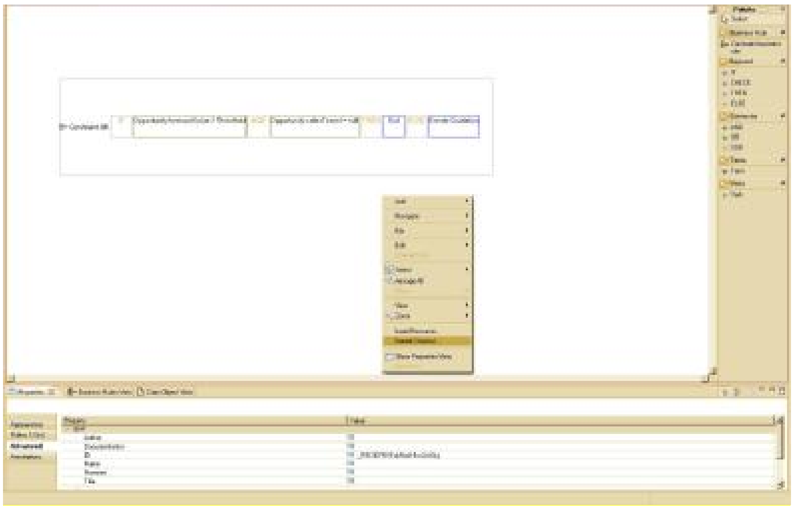

Figure 7 Business rule editor

A context menu function “Create shortcut…” in the editor allows for shortcuts to other editors. It can be a data object, an attribute of a data object or a construct from organizational model.

## 4.3 Organizational view

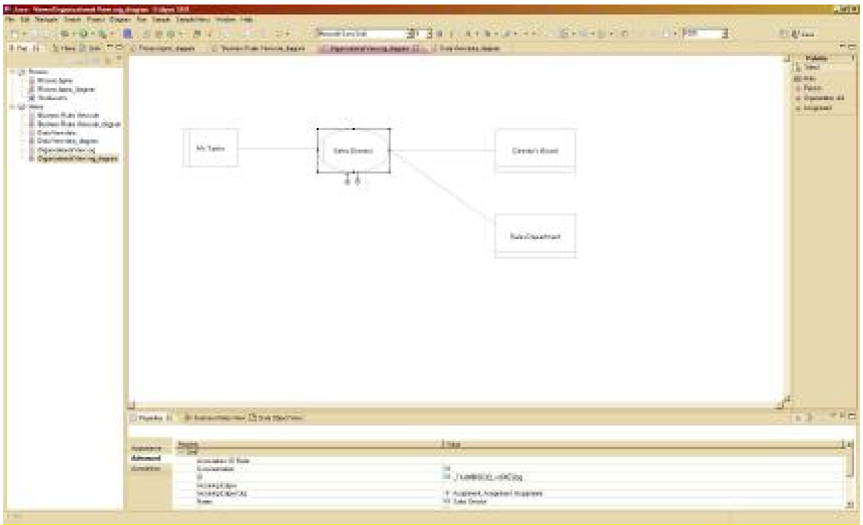

Figure 8 Role editor overview

The organizational hierarchy editor consists mainly of the editor pane and the palette to the right of the editor pane (see Figure 8). An organizational diagram is represented through four constructs: role, organizational unit, person and assignment. A Role is a position that a person can take in an organizational unit. An organizational unit is a department or a part in an organization, possibly hierarchically ordered and a person represents a physical person who is working in an organization and performs a certain role in there.

## 4.4 Data view

Data model editor consists mainly of the editor pane and the palette to the right of the editor pane (see Figure 9). A data diagram is represented through three constructs: data object, attribute and relationship. A data object is an entity describing a fact model of the model domain which this model covers. A simple attribute describes its object's property in very simple informal way and a relationship makes logical connection between two data objects.

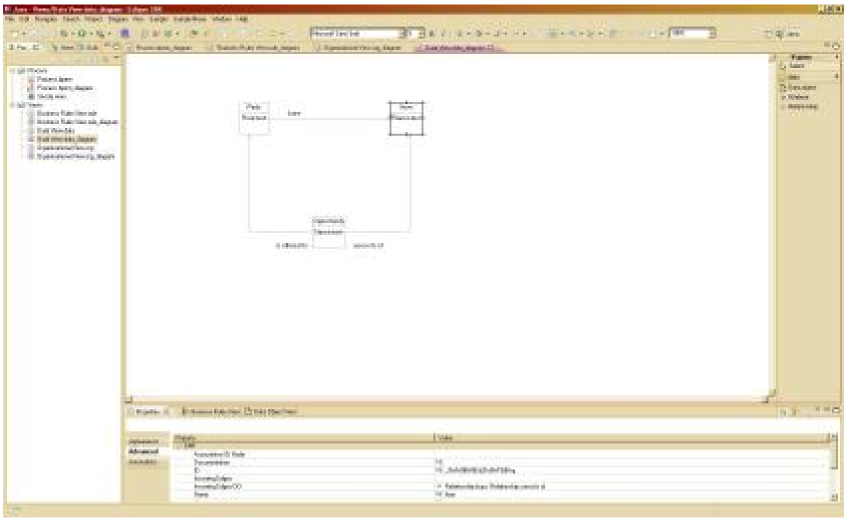

Figure 9 Data editor overview

## 5 Conclusion and outlook

A possible way to create software systems to support flexible business processes in industrial enterprises is the MDA approach. Currently software systems are designed without appropriate consideration of user requirements. Therefore domain users and their knowledge should directly be involved in the software design process. So, models on the CIM level should be used as the starting point for Model Driven Architecture approaches. This can lead to a model transformation process that starts on conceptual level and leads to generated models on PIM and PSM level.

But actually suitable conceptual modelling languages that are seamlessly integrated into a MDA approach are still missing. Hence the authors proposed a conceptual modelling language which is based on and still compliant to BPMN. To evaluate the practicability and usability a modelling tool based on GMF was implemented.

Further research in the area of MDA approaches starting on CIM level is needed in the model transformation from CIM to PIM level to have a better integration into existing PIM tools.

## References

[FeLo03] Fettke, P.; Loos, P.: Model Driven Architecture (MDA). In: Wirtschaftsinformatik, Bd. 45, 2003, Nr. 5, pp. 555-559.

[Fra03] Frankel, D.: Model Driven Architecture. Applying MDA to Enterprise Computing, Wiley & Sons, 2003.

[Fra05] Frankel, D.: Toward a Business Process Platform. In: MDA Journal, July 2005, pp. 1-7, 2005.

[GMF08] Graphical Modelling Framework (GMF), http://www.eclipse.org/modeling/gmf/

[GMF08a] GMF Tutorial, http://wiki.eclipse.org/GMF_Tutorial

[MSUW04] Stephen J. Mellor, Kendall Scott, Axel Uhl, Dirk Weise: MDA Distilled: Principles of Model-Driven Architecture, Addison Wesley, 2004.

[OMG01] Object Management Group: Model Driven Architecture (MDA) http://www.omg.org/docs/ormsc/01-07-01.pdf, 2001.

[ScWe05] Scheer, A.-W.; Werth, D.: Geschäftsprozessmanagement und Geschäftsregeln. In A.-W. Scheer (Hrsg.): Arbeitsberichte des Instituts für Wirtschaftsinformatik, Heft Nr. 183, Saarbrücken, 2005.