MEMOCenterNG – A full-featured modeling environment for organization modeling and model-driven software development

Jens Gulden and Prof. Dr. Ulrich Frank

University Duisburg-Essen,
Universitaetsstr. 9, 45141 Essen, Germany
{jens.gulden,ulrich.frank}@uni-duisburg-essen.de
http://www.wi-inf.uni-duisburg-essen.de/FGFrank/

Abstract. We present MEMOCENTERNG¹, an integrated, full-featured modeling environment containing 11 built-in modeling languages together with meta-modeling support for creating new languages and corresponding diagram editors. Tools for analysis, model transformation and code generation are included.

Key words: Modeling, Model-driven software development, Multi-perspective modeling, Enterprise modeling

1 A comprehensive multi-language modeling tool

Large modeling projects typically use multiple modeling languages simultaneously to express a variety of aspects of a modeled system. This holds true in diverse types of modeling projects, especially in business process modeling, which unites conceptual business aspects with technical realizations. Model-driven software development benefits from multiple interrelated modeling perspectives to gain a coherent and as complete view as possible on the system to be developed.

To foster large modeling projects, an integrated modeling environment tool is desirable to reduce efforts in combining multiple model editors operating on shared model content. Semantic integrity is an indispensable requirement for multi-perspective modeling [6], that means, when different model editors reference the same concept (e.g., a person displayed both in an organization structure diagram and in a process diagram), the model editor components need to share common information about its unique identity. The need for semantic integrity requires model editors to work on top of a common set of meta-concepts [7] which ensures semantically valid relations among model data.

With MEMOCENTERNG, we present a comprehensive modeling environment which integrates an extensible set of modeling languages on the basis of

¹ MEMOCENTERNG is named after the MEMO enterprise modeling method [6]. "NG" stands for "Next Generation", MEMOCENTERNG is the successor to an earlier MEMOCENTER prototype application.

a consolidated common meta-model API. The provided modeling languages are suitable to express knowledge about a system on three different levels of abstraction from multiple interdependent perspectives.

The available levels of abstraction provided by built-in modeling languages in MEMOCENTERNG are:

- A meta-modeling layer which allows for creating new modeling languages and corresponding diagram editors, and also internally defines other built-in modeling languages.
- 2. A set of built-in modeling languages for modeling the organizational environment of involved actors, their goals, behaviour and involved resources. This organizational abstraction layer is provided by domain-specific modeling languages of the MEMO modeling method [6].
- 3. An implementation model layer which offers general purpose modeling languages to express a software system's inner perspective in terms of, e.g., classes, attributes and relationships.

Figure 1 gives an overview on the component architecture of MEMOCENTERNG, as it evolves from applying the MEMO Meta Modeling Language (MML) for language creation.

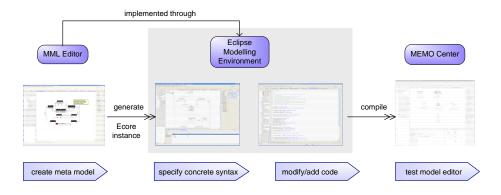


Fig. 1. MEMOCENTERNG component architecture

MEMOCENTERNG contains a total of 11 modeling languages on all three abstraction layers. This set is composed of 7 domain specific languages as proposed by the MEMO method [6], made up of Activity Diagrams, Allocation Diagrams, Organization Diagrams, Process Control Flow Diagrams, Process Decomposition Diagrams, Resource Diagrams and Strategy Diagrams. 3 implementation-level generic purpose modeling languages covering Data Flow Diagrams (DFD), Entity-Relationship-Models (ERM) and class-diagrams of the Unified Modeling Language (UML), and the MEMO Meta-Modeling Language (MML). These initially available languages can be enhanced by any number of specific modeling

languages created with the MML. Supplemented by third-party tooling components, models can be analyzed, model-transformations can be carried out, and code-generation mechanisms are available to generatively create software in a model-driven development process. Altogether, MEMOCENTERNG forms a comprehensive environment for modeling on multiple layers of abstraction from multiple perspectives, store and manage interrelated models in a common environment, and further process model data inside the same platform. In model-driven software development projects, generated artefacts can furthermore be edited, compiled and packaged within the same tool. The platform is based on the Eclipse [4] environment which can additionally be enhanced by a multitude of third-party supplementary components for software development.

By means of the included MEMO Meta Modeling Language (MML, [7]), new modeling languages can efficiently be created, and appropriate diagram editors for using the languages are automatically created from MML meta-models.

Since all generated components run in the same environment as MEMO-CENTERNG, models and generated software components may even reflectively refer to MEMOCENTERNG's models. This allows models to be integrated into software components at runtime as part of a self-referential information system architecture [8].

2 Organization modeling languages

To express the outer context of an incident to be modeled, organization modeling languages capture people's goals, their behavior, organizational structure and resources of the modeled context. Such types of models play an important role in enterprise modeling (EM, [6]), to express types of business processes that are performed in an organization. Besides business contexts in a narrow sense, any organizational setting and projects with shared goals among groups of people can generically be expressed with the semantic modeling concepts provided by these modeling languages.

MEMOCENTERNG comes pre-packaged with interrelated modeling languages of the MEMO OrgML [6] that cover organization modeling. These are in the first place the *Organization Diagram* language for modeling organizational structure, and the *Process Control Flow Language* [6], which allow to express semantically rich process model descriptions of business processes and other methodical procedures in organizations. The Process Control Flow language is enhanced by the *Process Decomposition Language* which allows for specifying static decomposition relationships among process steps, i.e., express which process steps are made out of others. Finally, the *Strategy Diagram* and *Activity Diagram* languages for expressing strategy, goals and actions from a high-level strategic view are part of the environment.

To model physical and non-tangible resources in business contexts, the *ResML* [10] is included in the set of modeling languages, accompanied by the *Allocation Diagram* language which is responsible to express the mappings between process steps and resources.

4 Jens Gulden, Ulrich Frank

Figure 2 shows an example Process Control Flow model edited in MEMO-CENTERNG.

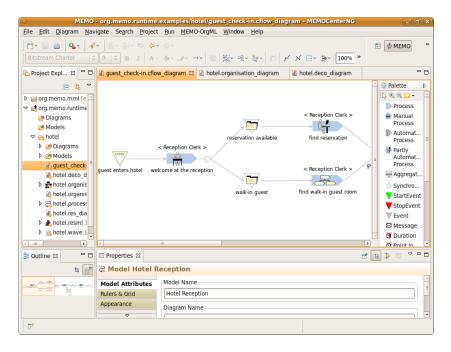


Fig. 2. Example MEMO Process Control Flow model

3 Included meta-modeling language

The MEMO Meta Modeling Language (MML, [7]) included in MEMOCENTERNG has especially been designed to efficiently enhance existing languages, and to automatically generate deployable diagram editors from meta-model descriptions of modeling languages. The MML puts special focus on interlinking multiple modeling languages. Concepts from other, previously existing MML models, can be referenced. Every concept is classified by a unique graphical symbol which indicates to which language it belongs.

Full-featured diagram editors can automatically be generated by a single mouse-click from MML models within the user-interface of MEMOCENTERNG's MML editor. These generated diagram editors are fully downwards-compatible to the Eclipse Modeling Framework (EMF, [13]) and Graphical Modeling Framework (GMF, [9]) components, which allows for applying any additional EMF / GMF technology component to MML-specified model editors and corresponding model instances, and also use model-instances for analysis, transformation and codegeneration (see sect. 5).

Figure 3 gives an example of an MML model edited in MEMOCENTERNG.

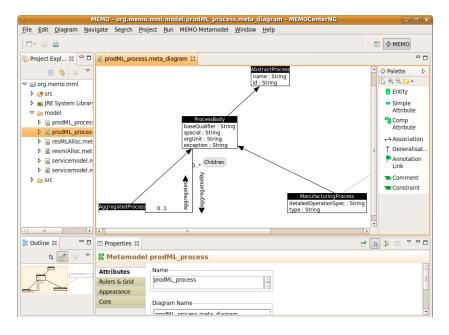


Fig. 3. Example of an MML model edited in MEMOCENTERNG

4 Implementation-level modeling languages

In order to take a traditional modeling view on a low abstraction level, MEMO-CENTERNG contains three classical formal systems modeling languages, *Entity Relationship Models* (ERM, [2]), *Data Flow Diagrams* (DFD, [3]) and object oriented class diagrams from the *Unified Modeling Language* (UML, [1]). Selected concepts of these languages can be referenced from elements in organization models to trace implementation details from a high real-world abstraction level down to technical details.

Each of the implementation-level modeling languages comes with pre-packaged analysis and code-generation functionality. The ERM model editor allows to generate a relational database schema from ERM models as a sequence of executable SQL data declaration statements, which subsequently may be executed from inside the Eclipse platform to deploy the initial database. The DFD editor comes with basic analysis capabilities, and from UML class diagrams, the source code for Java classes can be generated.

Together with the software development features of the underlying Eclipse [4] platform, MEMOCENTERNG forms a fully integrated model-driven software

development environment. An example of integrating between an organization model and an implementation model is displayed in figure 4.

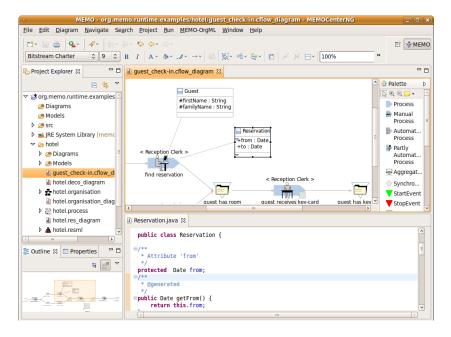


Fig. 4. Integrating between an organization model and an implementation model

5 Supplementary Tools for analysis, model-transformation and code-generation

All components of MEMOCENTERNG are based on the Eclipse Modeling Project [9] components EMF and GMF, and make use of the Ecore language [5] through the MML. As a consequence, any supplementary component that exists for the Eclipse Modeling Project can be applied in MEMOCENTERNG, too. This allows for seamlessly integrating supporting technology such as Xtend / Xpand [5], and other specific code generation languages such as Velocity [12], or the QVT [11] model transformation language. By default, the Eclipse components for Xtend / Xpand transformations are included in the MEMOCENTERNG environment.

6 Availability

A beta-version of MEMOCENTERNG is available for download at http://www.wi-inf.uni-duisburg-essen.de/FGFrank/download/memo/. Please request password information from the authors.

7 Conclusion and Outlook

We have presented a modeling tool that offers multiple modeling languages in an integrated environment, based on a meta-model supported language architecture and enriched by an easy-to-use meta-model editor for specific language enhancements.

A common language architecture ensures the semantic integration of concepts across multiple languages. By incorporating meta modeling and the creation of new modeling languages into the feature spectrum of the modeling tool, a new degree of flexibility and adaptability to future requirements part of the application. This makes MEMOCENTERNG a comprehensive, full-featured integrated modeling environment for multiple types of modeling projects, including model-driven software development approaches. Currently, the tool is successfully used for teaching purposes and in medium-sized business projects carried out in cooperation with our research group.

In further research work, we intend to develop and integrate additional modeling languages, e.g., for designing indicator systems that are integrated with models of business processes and IT resources. We also plan to extend the framework with elaborate support for method engineering by integrating model editors with corresponding process editors.

References

- Booch, G., Jacobson, I., Rumbaugh, J.: The Unified Modeling Language Reference Manual. Addison-Wesley, Reading (1999)
- Chen, P.: The Entity-Relationship Model Toward a Unified View of Data. In: ACM Transactions on Database Systems 1/1/1976 ACM-Press ISSN 0362-5915, pp. 9–36 (1976)
- 3. DeMarco, T.: Structured Analysis and System Specification. Prentice Hall, Upper Saddle River (1979)
- 4. Eclipse Foundation: Eclipse Platform. http://www.eclipse.org/ (2010-04-25)
- 5. Efftinge, S., Friese, P., Haase, A. and others: openArchitectureWare User Guide. http://www.openarchitectureware.org/pub/documentation/4.3.1/html/contents/index.html (2008)
- Frank, U., MEMO: A Tool Supported Methodology for Analyzing and (Re-)Designing Business Information Systems. In: Ege, R., Singh, M., Meyer, B. (eds.): Technology of Object-Oriented Languages and Systems, pp. 367–380 (1994)
- Frank, U.: The MEMO Meta Modeling Language (MML) and Language Architecture. ICB-Research Report No. 24, Institute for Computer Science and Business (ICB), University Duisburg-Essen (2008)
- 8. Ulrich Frank, U., Strecker, S.: Beyond ERP Systems: An Outline of Self-Referential Enterprise Systems. ICB-Research Report No. 31, Institute for Computer Science and Business (ICB), University Duisburg-Essen (2009)
- 9. Gronback, R. C.: Eclipse Modeling Project: A Domain-Specific Language (DSL) Toolkit. Addison-Wesley Longman, Amsterdam (2009)
- Jung, J.: Entwurf einer Sprache für die Modellierung von Ressourcen im Kontext der Geschftsprozessmodellierung. Logos, Berlin (2007)

Jens Gulden, Ulrich Frank

8

- 11. Object Management Group: Meta Object Facility Query / View / Transformations. $\verb|http://www.omg.org/spec/QVT/1.0| (2008)$
- 12. Apache Software Foundation: The Apache Velocity Project. http://velocity.apache.org/(2010-04-25)
- 13. Steinberg, D., Budinsky, F., Paternostro, M., Merks, E.: EMF: Eclipse Modeling Framework (2nd edition). Addison-Wesley Longman, Amsterdam (2009)