Meta-Modelling in Tool Support for Time-Triggered Application Development

Christian Paukovits and Wilfried Elmenreich (Faculty Mentor)
Institut für Technische Informatik
Technische Universität Wien
Email: {pauko,wil}@vmars.tuwien.ac.at

Abstract — *This paper introduces an integrated development environment based on the open-source tool GME (Generic Modeling Environment) for distributed embedded real-time systems using TTP/A as communication subsystem. The tool follows the meta-modelling design approach, and delivers an implementation of the conceptual model of TTP/A applications.*

I. Motivation

Designing and implementing a distributed embedded real-time system is a challenging task due to its inherent complexity. A possible approach to cope with complexity is to employ tools in the design and implementation process that relieve the designer from lower level issues like matching system parameters and determining communication message schedules.

We identified the need for an integrated development tool, which assists the designer from the first draft of the *application's model* to the definition of the global *communication schedule* as well as the local *job specification* and the compilation and linking of the resulting *source code*. The intentions are, on the one hand to accelerate the design and implementation phase of an embedded system's software, on the other hand to reduce the potentials of errors occuring during the whole development process.

The first goal might especially appeal to the academic research, where it is the intention to have shorter development cycles, so that we can implement several diversitive approaches to a problem fast and efficiently, when conducting research among the field of embedded systems. This matter is of interest in the industrial field, too. However, here the second goal is of upmost importance. The less errors we have to remove in source code, the less resources the debugging consumes. Thus, the application of an integrated tool will relieve a project's budget and other resources.

Existing tools like **Rational Rose**, **LabVIEW** and **Simulink** are well-known, but do not particularly support our time-triggered modelling approach. Therefore, we have adopted the open source tool GME (Generic Modeling Environment) [1] as integrated modelling tool, which supports a meta-level design approach. Although, GME can be modified in order to fit the needs of any target platform and communication technology, this case study deals with TTP/A [2] as the underlying real-time communication system and application design.

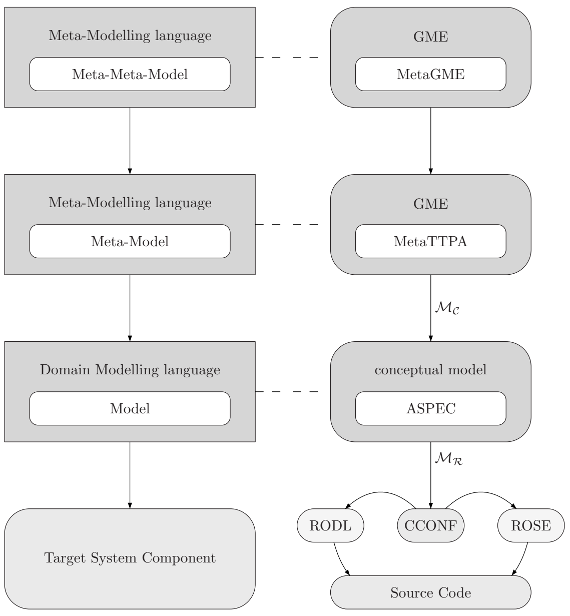

Figure 1: Meta-Modelling with GME

II. Meta-Modelling with GME

Meta-Modelling combines the strengths of *domain-specific* and *generic notational* approaches. Due to its expressiveness we can describe (meta-)models for our given **domain** of time-triggered distributed embedded real-time systems in a **generic** formalism. Fig. 1 illustrates the functioning of meta-modelling. Firstly, we have one generic model (*meta-meta-model*), which models the *meta-model* of a given domain. Such a meta-model includes the familiar *terminology* as well as *integrity constraints*, which a concrete domain model has to fulfill. Finally, we use that meta-model in order to specify the *model* a concrete target system with a domain-specific point of view.

Furthermore we see from Fig. 1, which role is taken by GME in our case study of TTP/A application development concerning the modelling of the distributed real-time embedded system. The meta-meta-model as well as the meta-model named *MetaTTPA* are expressed by means of GME, which is an UML-related notational style. Then, the *model* of the target system is described using MetaTTPA, which defines the *conceptual model* of TTP/A applications.

III. Conceptual Model

The *conceptual model* is the theoretical basis of TTP/A applications (cf. [3]). It defines a *set of integrity constraints*, a *terminology* and a *design pattern*, how TTP/A applications ought to be designed. MetaTTPA is the implementation in GME of the conceptual model.

The conceptual model defines two major entities of TTP/A applications. The **application specification (ASPEC)** is a formal description of a TTP/A application concerning *functional and temporal requirements*. This includes the *decomposition* among the TTP/A cluster, *dataflow* between jobs, and *deadlines* of dataflow transmissions and job executions. The **cluster configuration (CCONF)** gives information, what each node in the TTP/A cluster does during each TTP/A slot, for instance executing a job, or sending / receiving data on the bus. The functional and temporal requirements from the ASPEC find their manifestation in the cluster configuration.

Each application specification has to fulfill all the integrity constraints defined in the conceptual model in order to be *valid*. Cluster configurations must satisfy the functional and temporal requirements stated in the ASPEC in order to be *feasible*.

IV. Integrated Development Environment

In Fig. 2 we find the relation between ASPEC and CCONF. The linking entities are specialized plug-ins of GME, so-called *interpreters*, which support the work-flow. A *scheduler* takes a valid ASPEC and applies a scheduling algorithm in order to figure out an appropriate schedule of the application. Then, it creates a cluster configuration. Moreover, a *code generator* produces source code for RODL, ROSE, Interface Files, and job stubs from the CCONF.

The GME tool suite provides a mechanism of *constraint checking* based on OCL constraints. As a result, the integrity constraints imposed by the conceptual model and hence MetaTTPA, which is the implementation of the conceptual model, can be asserted within the GME tool suite. There is no other constraint checking tool involved. In other words, GME is able to ensure validity of application specifications by itself. However, feasibility of cluster configurations can not be checked with that built-in facility, because this is beyond the scope of OCL constraints. So, this task must be done by a scheduler, which calculates cluster configurations.

All the steps beginning from the description of the application specification till the output of source code take place within one meta-modelling tool suite: GME. We have used GME's mechanism of plug-ins to equip the generic tool with TTP/A specific schedulers and code generators. As a result, we elevate such a generic modelling tool to an integrated development environment (IDE) according to our needs for distributed embedded real-time systems using TTP/A as its underlying communication subsystem.

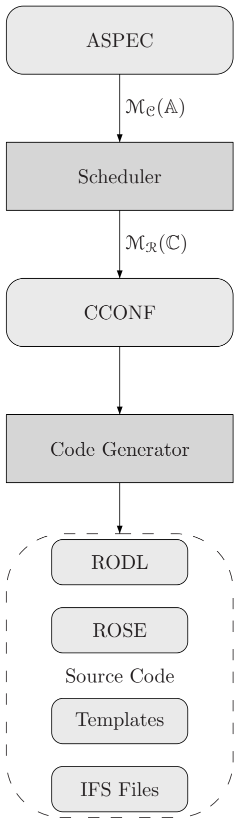
ASPEC
$\mathcal{M}_{\mathcal{C}}(\mathbb{A})$
Scheduler
$\mathcal{M}_{\mathcal{R}}(\mathbb{C})$
CCONF
Code Generator
RODL
ROSE
Source Code
Templates
IFS Files

Figure 2: Work-Flow with GME

V. Conclusions

We proposed an application of a meta-modelling tool, which assists the designer of distributed embedded real-time systems throughout the whole design work-flow. The tool is especially tailored in order to fit the needs of TTP/A application development. Therefore, it entails an implementation of the conceptual model, and offers an automatic generation of RODLs, Interface Files, and job stubs in application source code files.

Acknowledgments

The authors want to thank Susanne Kandl and Christian El Salloum for their constructive comments and ideas on this work. This work was supported by the Austrian FWF project TTCAR under contract No. P18060-N04.

References

[1] A. Ledeczi, M. Marot, A. Bakay, G. Karsai, J. Garrett, C. Thomason, G. Nordstrom, J. Sprinkle, P. Volgyesi. The Generic Modeling Environment. In *Proceedings of WISP 2001*, Budapest, Hungary, May 2001.

[2] H. Kopetz et al. Specification of the TTP/A Protocol. Research Report 61/2001, Technische Universität Wien, Institut für Technische Informatik, Treitlstr. 1-3/182-1, 1040 Vienna, Austria, 2001. Version 2.00.

[3] C. Paukovits. Applying Model-Integrated Design on Time-Triggered Application Development. Master Thesis, Technische Universität Wien, Institut für Technische Informatik, Vienna, Austria, January 2006.