MAENAD

Grant Agreement 260057

Model-based Analysis & Engineering of Novel Architectures for Dependable Electric Vehicles

Report type Deliverable D5.2.1

Report name MAENAD Analysis Workbench

Dissemination level	**PU**
Status	**Intermediate**
Version number	**2.0**
Date of preparation	**2012-08-31**

Authors

Editor	E-mail
Carl-Johan Sjöstedt	carlj@md.kth.se

Authors	E-mail
Carl-Johan Sjöstedt	carlj@md.kth.se
Matthias Biehl	biehl@md.kth.se
Sara Tucci-Piergiovanni	sara.tucci@cea.fr
Martin Walker	martin.walker@hull.ac.uk
Frank Hagl	frank.hagl@continental-corporation.com
Henrik Lönn	henrik.lonn@volvo.com

The Consortium

Volvo Technology Corporation (S) Centro Ricerche Fiat (I)

Continental Automotive (D) Delphi/Mecel (S) 4S Group (I)

MetaCase (Fi) Pulse-AR (Fr) Systemite (SE) CEA LIST (F)

Kungliga Tekniska Högskolan (S) Technische Universität Berlin (D) University of Hull (GB)

Revision chart and history log

Version	Date	Reason
1.0	2011-05-31	First release
2.0	2012-08-31	Second release

Table of contents

1 Introduction

Deliverable D5.2.1 consists of:

- The plugins and tools that make up the MAENAD analysis workbench (MAW)
- This document providing references and short descriptions of the tools in the MAW.

The MAW consists of software developed to provide support for modelling and analysis, based on specifications of WT3.x. One objective is to validate the analysis concepts. The MAW will be made public in order to make the analysis concepts more accessible and understandable.

This intermediate release consists of the sections *Current status*, *Future plans* and *Requirements,* where the latter describes what project and language requirements that the plugin/tool fulfils. The final deliverable will also include installation/usage instructions, not available for this release.

The target of MAW is the MAENAD Modeling Workbench, D5.1.1, but with the model exchange tool, as described in section 2.6, they will also work with the tool adaptations (MetaEdit+ and System Weaver), which are developed in Work Task 5.3.

2 The components of the MAENAD Analysis Platform

Some of the plugins were developed throughout the ATESST [1], ATESST2 [1] and EDONA [2] projects, and will be updated during the MAENAD project, because of new releases of the MAENAD modelling workbench, and new releases of the profile. There will also be new plugins, for the interchange of EAST-ADL models using an exchange format, and for exchange with ModelicaML and MODELISAR FMU:s.

	Main developer	Overview
AUTOSAR Gateway	CEA	Provide an enhanced transformation from EAST-ADL2 design architecture to AUTOSAR compliant software architecture, based on the ARTOP framework.
Timing Analysis – Qompass Tool	CEA	Provides support for early-stage timing analysis of EAST-ADL models
Simulink Gateway	KTH	Provides input/output facilities with Simulink to enable simulation.
HiP-HOPS Gateway	KTH	Builds on previous results, expanding analysis capability and optimization engine of HiP-HOPS, and enhancing the feedback of FMEA/FTA in the design process.
Architecture optimization and configuration	University of Hull	Builds on CVM, the variability management plugin from ATESST2, which will be extended and interfaced with an optimization engine.
Plugin for EAST-ADL exchange format EAXML	Pulsar	Provides exchange between EAST-ADL2 models in a UML2 tool and an AUTOSAR-compliant XML exchange format
Modelica Exchange	KTH	Using ModelicaML together with EAST-ADL for analysis.
MODELISAR FMU Import	VTEC	Generate EAST-ADL FAA models from MODELISAR Functional Mockup Units.

2.1 AUTOSAR Gateway

The AUTOSAR gateway builds on results from EDONA and ATESST2 projects to provide an enhanced transformation from the EAST-ADL design architecture to AUTOSAR vehicle architecture design and initial system configuration. The gateway is based on the ARTOP framework, which is an implementation of common base functionality for AUTOSAR development tools, available free of charge for AUTOSAR members [5]. The transformation takes as input EAST-ADL functional description, hardware description and allocation information and generates a tentative AUTOSAR software architecture, a hardware topology and mapping constraints (coming from EAST-ADL allocation information). The generation of the tentative software architecture is based on mappings between EAST-ADL functions and AUTOSAR software components/runnables.

2.1.1 Current status

The first version of the AUTOSAR gateway has been released at M12. This version was based on the one developed in the EDONA project and then it had been ported to the new Papyrus MDT platform and 2.1.9 EAST-ADL profile version.

As previously planned, a new version of the AUTOSAR gateway is going to be released at M24. In this version the transformation towards the AUTOSAR implementation architecture relies on an AUTOSAR UML profile (subset of AUTOSAR centred on relevant templates: software component, system and ECU resource namely). As a result, the transformation from EAST-ADL design architecture to AUTOSAR vehicle design architecture/system configuration produces UML profiled models.

AUTOSAR gateway implements also export functionality from AUTOSAR-UML profiled models to AUTOSAR XML.

2.1.2 Future plans

Beyond M24 we will proceed to the AUTOSAR gateway assessment considering feedback from MAENAD users.

2.1.3 Requirements from WT2.1: Identifications of needs

The AUTOSAR gateway is mentioned in the following requirement:

DOW#0111: The SWC Synthesis (FDA-IL) shall be modelled in MAW-AR Gateway plugin [4].

2.2 Timing Analysis

At the beginning of the project the purpose of the Timing Analysis plug-in was centred on the idea of providing schedulability analysis of EAST-ADL models. Beyond M12, Timing Analysis for EAST-ADL models has been refined [6]. The following timing analyses have been identified as best-suited to support EAST-ADL Design level models (as documented in D3.1.1):

Early Stage Schedulability Analysis. The allocation model of EAST-ADL defines on which ECUs, functions will be executed and on which buses, communication between functions will take place. Basing on this information, the following two interesting metrics, relevant from a schedulability point of view, can be considered:

- Resource Utilization. Resource utilization is a function of (i) the function's activation rate and (ii) a time budget representing the time an execution/communication will take. Basing on utilization of single resources, other related interesting metrics can also be extracted, as load distribution and function/signals extensibility (function of processors/bus slacks).
- Interference Time: represents the waiting time to access shared resources (CPU/Bus). This delay is caused by concurrent functions/signals that are allocated to the same execution/communication node. Small interference is desirable to minimize end-to-end latency.

Schedulability analysis. Schedulability analysis is applied for the special case of linear chains of activations running on a mono-processor system and when chain rates are harmonic. The task model is generated automatically. This generation is transparent to the user. Once the task model is obtained, a response time will be computed for each end-to-end chain (thread) trough Rate Monotonic Analysis [6].

2.2.1 Current status

Let us remind that at the beginning of the MAENAD project the analysis engine should have been provided as a third-party tool. On the other hand, after EDONA and ATESST2, CEA worked on the implementation of schedulability analysis algorithms as part of its research activities (not included in the MAENAD project) in its own MARTE-based plug-in called Qompass. This timing analysis support was not completely compliant with timing analyses identified as best-suited for EAST-ADL design models (Section 2.2). Moreover, in order to directly analyse EAST-ADL models, a transformation between EAST-ADL profile models and entry models for Qompass was needed.

During year 2 the Qompass timing support has been adapted/enhanced in order to support EAST-ADL design-level timing analyses. An EAST-ADL/Qompass transformation has been developed as well. A new version of the Timing Analysis plug-in, is going to be released at M24 embedding these new features.

2.2.2 Future plans

Once the Qompass plug-in will be released, we will proceed to its assessment considering feedback from MAENAD users. This assessment will be done in link with (i) the needs and goals set for the optimization experiments and tool architecture and (ii) possible adaptations/improvements required to apply Qompass to the Brake-by-Wire case study. Planned release for a Qompass update is M30.

2.2.3 Requirements from WT2.1: Identifications of needs

The Timing Analysis plugin is mentioned in the following requirement:

DOW#0110: The Timing analysis (DL) shall be modelled in MAW-Timing plugin [4].

2.3 Simulink Gateway

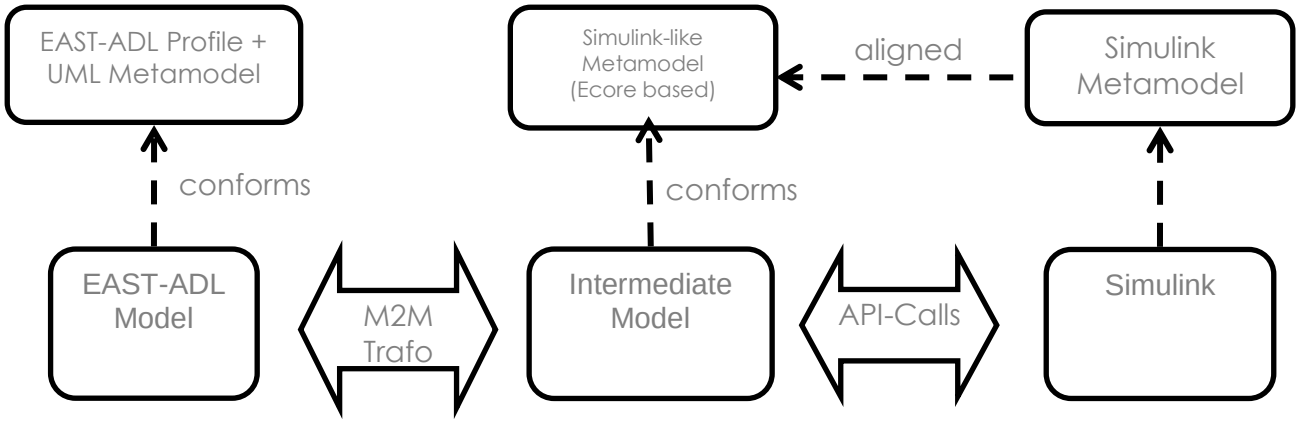

Figure 1: Overall design

The Simulink gateway builds on results from the ATESST2 project, and provides input/output facilities of models with Simulink to enable simulation. The plugin is divided in two parts (Figure 1):

- A GUI plugin to the MATLAB/Simulink environment, which aids the user in creating models that conform to the format that is needed to being able to convert it into an EAST-ADL model.
- An Eclipse plugin, which can convert between the intermediate format of Simulink models and EAST-ADL models

The MATLAB plugin exports the MATLAB/Simulink models into a custom Ecore-based format. A subset of Simulink functions is used; only library blocks of subsystems are considered. However, any Simulink model could be converted into a structure of system reference blocks, without affecting the model's simulation behaviour. The GUI plugin for Simulink mentioned above converts standard Simulink subsystems to system reference blocks, and tags them for conversion to EAST-ADL by putting them in a "FunctionTypes"-library, and assigning a unique ID, to allow bi-directional exchange and updates. To include the internal structure of a subsystem, the same pattern is repeated.

Import works the other way around, FAA FunctionTypes and FunctionPrototypes are imported to empty library blocks in the "FunctionTypes"-library, and instances of them respectively.

2.3.1 Current status

An effort has been made to port the plugin developed in the ATESST2 project to the new Papyrus MDT environment. It is possible to run the plugin, but the ATL transformation eventually crashes, for unknown reasons.

Due to lack of resources from KTH, no further development has taken place. When the modelling workbench and future tool environment for EAST-ADL is known, a new effort could be made.

Simulink exchange has been implemented in MetaEdit+, which could better serve as a demonstrator for proof-of-concept exchange between Simulink and EAST-ADL.

2.3.2 Future plans

In the iFEST project, KTH is developing a transformation between UML and Simulink using a tool integration framework. This tool adaptor is partly based on the ATESST2 Simulink plugin. The plan was to eventually merge this effort with the MAENAD Simulink tool adaptor.

Currently, KTH is prototyping of behaviour analysis with MATLAB/Simulink using the MetaEdit+ editor, and exchange mechanism.

2.3.3 Requirements from WT2.1: Identifications of needs

A2#11: A formalized meta-model of the architecture description language shall be developed. (including structural elements, behavioural description means, models of computations, and transformation rules to prototype tools and Simulink.)

DOW#0112: The Simulink import-export (FAA, FDA) shall be modelled in MAW-Simulink plugin.

2.4 HiP-HOPS Gateway

Integrating safety analysis into the development of automotive embedded systems requires translating concepts of the automotive domain to the generic safety and error analysis domain. We assume a model-based development process where automotive concepts are represented by the EAST-ADL2 architecture description language, which supports system design on multiple levels of abstraction. The concepts of the error analysis domain are represented by the safety analysis tool HiP-HOPS.

It is assumed that EAST-ADL2 models are built using the UML profile. The HiP-HOPS plugin extensively uses the concepts defined in the EAST-ADL error model, but also other language constructs from the FDA level.

We automate the translation from EAST-ADL2 to HiP-HOPS by using model transformations. We leverage the advantages of different model transformation techniques by decomposing the translation into two distinct phases, and using an appropriate technique for each phase: A phase for conceptual mapping between the domains followed by a phase for representing the output in the desired concrete syntax.

With the resulting tight integration of the safety analysis tool and the model-based development environment, the automotive safety engineer can perform the safety analysis repeatedly on refined models with minimal effort. This is compliant with the iterative design activities, which require starting the analysis after each change in the system design.

The HiP-HOPS Gateway builds on previously developed Model Transformations and Eclipse Plugin, expanding analysis capability and optimization engine of HiP-HOPS, and enhancing the feedback of FMEA/FTA in the design process.

2.4.1 Current status

The plugin is updated to support EAST-ADL2 models built using the UML profile version 2.1.10. The HiP-HOPS plugin extensively uses the concepts defined in the EAST-ADL error-model, but also other language constructs from the FDA level. Core functionality is working, including HW-SW allocation.

2.4.2 Future plans

Maintenance until September 2012:

- Update of the plugin to work with the current version of Eclipse Indigo
- Update of the plugin to work with the current version of Papyrus MDT 0.8.2
- Update of the plugin to work with the current version of the EAST-ADL Profile 2.1.10
- Update of the test model for the current version of Papyrus

2.4.3 Requirements from WT2.1: Identifications of needs

UOH#0003: The HiP-HOPS analysis tool should support any ISO 26262 or related concepts (such as ASIL decomposition) necessary to allow ISO-compatible dependability analysis of EAST-ADL models.

UOH#0004: EAST-ADL and HiP-HOPS should be able to intercommunicate by means of model transformations provided by a dependability plugin in the MAENAD Analysis Workbench (MAW). Furthermore it should be possible to import or store the results from HiP-HOPS in the Workbench

and/or the EAST-ADL model, which will require establishing some form of (perhaps XML based) interchange format.

2.5 Architecture optimization and configuration

The architectural optimization & configuration capabilities to be developed in MAENAD build upon several tools and plugins, including CVM (and the corresponding variability management plugin from ATESST2), HiP-HOPS (and its associated plugin), and timing analysis tools like MAST (and associated plugin(s)). These tools need to be interfaced with an optimization engine for fully multi-objective optimization of EAST-ADL models to be possible.

2.5.1 Current status

A proposal for an EAST-ADL optimization architecture was developed over the course of the second year of MAENAD, building upon initial ideas from the York meeting in May 2011. This architecture is shown below:

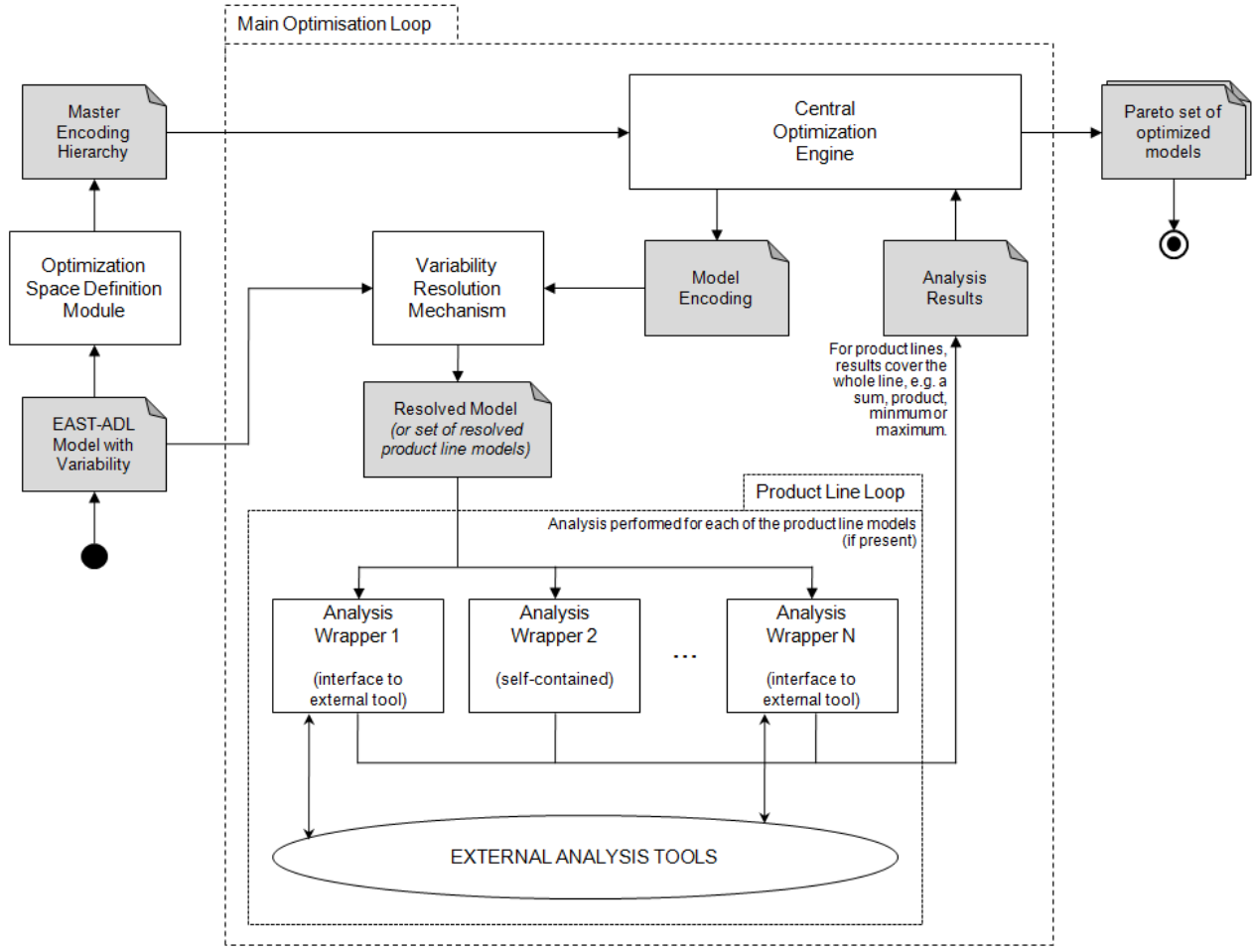

Figure 2: Optimisation Architecture

The optimisation architecture is described further in D3.2.1, but is intended to serve as a blueprint for the implementation of tool support for the optimisation process. It consists of several major elements, briefly described below:

- Optimization Space Definition Module (OSDM)

This module takes a variability-rich EAST-ADL model and generates a 'master encoding hierarchy', which is a hierarchical key to the design space represented by the model variability. In practice, this takes the form of a feature tree (with slight modifications), and can therefore be generated by variability tools like CVM.

- Central Optimization Engine (COE)

The COE is the driver of the optimization process. It is responsible for exploring the optimization design space on the basis of heuristic algorithms such as genetic algorithms. It generates an encoding for a particular design candidate, which is then resolved by the VRM (see below) and evaluated by analysis plugins to determine its relative score in each of the objectives being optimized (e.g. reliability, performance, cost, energy consumption etc.). Optimal candidates are preserved while sub-optimal designs are discarded. Once the process is complete, the COE will generate a report containing the set of optimized design candidates.

- Variability Resolution Mechanism (VRM)

The COE does not manipulate the EAST-ADL model directly. Instead, it modifies encodings (essentially feature trees), and then passes each encoding to the VRM, which is responsible for resolving the variability in the original model according to the encoding in order to produce a new model — a design candidate — that can then be analysed and evaluated.

- Analysis modules

For each objective being analysed, there needs to be a corresponding analysis module. The intention is that these analysis modules can be either external tools (such as HiP-HOPS or timing analysis tools like Qompass) or plugins written for the modelling/analysis environment (e.g. Papyrus, MetaEdit+, EPM etc.). The optimisation architecture does not necessarily interact with them directly; instead there should be a common approach, implemented by 'wrapper' objects if necessary, to present a consistent interface to the analysis modules. The hope is that this will allow new analysis modules and thus new objective types to be added (or removed) from the optimization process without requiring modification of the main optimization elements (i.e., the OSDM, COE and the VRM).

Scope has been left for product line optimisation to be added at a future date, although this will require additional complexity in the analysis wrappers, as each analysis type may have to function differently to achieve optimisation of product lines (e.g. the results of an unavailability analysis on a product line may just be the maximum probability, while the results of a cost analysis are likely to be a sum of weighted costs; the result is therefore heavily dependent on the type of analysis).

An initial prototype tool, OptiPAL, was developed by TUB and builds upon the EPM platform. It implements both the OSDM and VRM as part of the existing CVM plugin and also includes a new prototype COE. The COE is presently only a simple experimental version and does not implement the full genetic algorithm for optimization yet, but it does allow for generation of different encodings and thus allows the main optimization loop to take place. Analysis is provided by an OptiPAL cost analysis plugin (which currently only does simple cost summations) and a bridge to the HiP-HOPS safety analysis tool for dependability analysis via FTA.

OptiPAL is primarily intended as a proof of concept and as a way of testing out the optimization concepts on test models, to provide feedback to facilitate further development of the optimization architecture concept. Should it prove successful, however, it may evolve into a more fully functional optimization tool.

2.5.2 Future plans

The development of OptiPAL should greatly facilitate XGO work in the third year of MAENAD, enabling us to test the optimization concept on test models and use the feedback received to develop the concepts further. One of the main tasks ahead is the generation of those test models, both smaller, simpler models to test individual ideas and problems, and larger models such as the brake-by-wire model, which is intended to be extended with optimization variability. This has been a goal for some time but was hampered by a lack of tool support until now.

Should the experiments prove fruitful, the aim is to upgrade and extend OptiPAL further. The first step will be to implement full genetic algorithms in the COE module, to allow a better assessment of the practicality of the optimization process and produce more meaningful results in the form of a Pareto set of optimized designs. As tool support for other analyses matures, it may also be possible to create new interfaces to other tools or additional OptiPAL-based analysis plugins, e.g. for other FEV-related objectives such as cable length, battery life, or energy consumption etc.

Finally, the goal is to investigate the feasibility of product line optimization further. This would take the form of an 'inner optimization loop' and would mean leaving some variability unresolved in the encoding from the COE to allow the different parts of the product line to be iterated through exhaustively and analysed individually. The results of these analyses would then be combined in an analysis-specific manner by the appropriate analysis wrapper and returned as the results of the evaluation for that product line. Product line optimization would likely be a topic investigated towards the latter half of the third year, as it depends upon the standard optimization process being developed successfully.

2.5.3 Requirements from WT2.1: Identifications of needs

VTEC#UC006: A model of the validator with timing, dependability and cost annotations as well as design space, variability and take rate annotations is defined and exported to EAXML. An optimization tool computes the optimal design for the defined product line. The resulting optimized model is recorded in the model (design space variability removed) and exported in the EAXML file.

UOH#0002: The EAST-ADL error model should fully support automatic optimisation, e.g. through rules that specify a 1:1 mapping from nominal to error models.

UOH#0005: To support multi-objective optimisation, there must be a standardised way of passing design candidates to analysis tools/plugins and receiving results in a given format.

2.6 Plugin for EAST-ADL exchange format EAXML

The model exchange plugin provides exchange between EAST-ADL2 models in a UML2 tool and an AUTOSAR-compliant XML exchange format: EAXML.

2.6.1 Current status

Development has been postponed because of Pulse-AR departure. Pulse-AR was supposed to be the main developer for these plug-in because of its expertise in the serialization of ARXML. Clear decisions about a new planning for the development of this plug-in will be taken once it will be clear how the departure of Pulse-AR will be managed in the project. For the moment a tentative planning has been considered, as detailed in the Section below.

2.6.2 Future plans

Let us remind that CEA was supposed to collaborate with Pulse-AR to the investigation of the following restricted scenario: export to EAXML an existing EAST-ADL profile-based model from Papyrus; import an EAXML model as a new EAST-ADL profile-based UML model. For this, CEA/Pulse-AR suggested to make use of the investigations that will be made for the AR Gateway reengineering (see above 2.1). As already pointed out, this work demands a connection between an AR profiled UML model and an ARXML model that is planned for month 30. Adaptation of this work could be made to support the scenario during second half of year 3.

From this simple scenario, other contributors would define extensions as needed.

2.6.3 Requirements from WT2.1: Identifications of needs

EAXML is mentioned in VTEC#UC001 through VTEC#UC008 [4], which describes scenarios where 1) models are exchanged from the modelling workbench and the external tools of WT5.3; 2) the analysis tools of WT5.2 are used in link with the external tools of WT5.3, independently from the modelling workbench. Such scenarios rely on exchange based on EAXML files produced/read by the various tool chain participants.

2.7 Modelica Exchange

Modelica is a language for modelling and simulation of dynamical systems, and could be used for various analyses of EAST-ADL models, e.g. modelling of plant models or timing constraints.

2.7.1 Current status

There is a plugin for Papyrus MDT for ModelicaML, using the UML stereotype mechanism. By assigning both an EAST-ADL FunctionType and a ModelicaML stereotype to a class, Modelica behaviour can be assigned to EAST-ADL FunctionTypes.

Another possibility is to use the Functional Mockup Interface to exchange Modelica models with EAST-ADL. A prototype plugin for getting structural information from Functional Mockup Units and transforming it into EAXML has been developed by VTEC.

2.7.2 Future plans

One way of co-use of EAST-ADL and ModelicaML is using the EAST-ADL structural model in combination with the ModelicaML dynamic model. The structural parts could be mapped more or less directly 1:1 into ModelicaML, the dynamic parts can be modelled by ModelicaML as the behavioural parts of a FunctionType. The FunctionType in this case should be seen as an AUTOSAR runnable.

ModelicaML will be used in order to do verification of TADL timing constraints in the context of the TIMMO2USE project by Continental during the fall of 2011 and spring 2012. Based on these efforts, a plug-in for TADL timing constraints within MAENAD might be an option.

A summary on how to co-use EAST-ADL and ModelicaML could be developed, in form of an alignment/mapping of ModelicaML and EAST-ADL stereotypes and possibly an automatic conversion.

2.7.3 Requirements from WT2.1: Identifications of needs

Modelica exchange is mentioned in e.g.:

CON#0009: Annotate SysML/Modelica models with EAST-ADL stereotypes. On base of a defined mapping between SysML and EAST-ADL, the SysML model of the profile and mode selection logic shall be annotated with EAST-ADL stereotypes. Structural as well as behavioural elements shall be annotated with EAST-ADL stereotypes CON#0012: EAST-ADL supports the definition of timing of constraints by the inclusion of the TADL language. A verification of the TADL constraints shall be possible. It is an option to verify TADL constraints either by the use of timing analysis techniques as provided by languages as MARTE or AADL or model simulation techniques as provided by Modelica. Within the scope of the project, it has to be evaluated, if the verification of timing constraints on base of these techniques is possible and samples shall be given.

CON#0014: VV Case Development, including fault injection and verification of model constraints in a Modelica simulation environment.

CON#0021: Virtual integration is an important use case during development within the ID4EV project. It is obvious that the physical demonstrator will not be available for a long time and the SW modules must be integrated in a virtual environment. The Modelica simulation environment is very well suited for integration C-code into the simulation environment.

2.8 MODELISAR FMU Import

The MODELISAR project [5] has defined Function Mockup Units to exchange and integrate simulation blocks from different modelling tools.

2.8.1 Current status

There is an Eclipse plugin that imports the Function Mockup Unit specification, called Function Mockup Interface (FMI). The Function Mockup Interface defines the input and output variables of each unit and also the data types of these variables. Based on this information, an AnalysisFunctionType with the corresponding interface is defined in EAXML.

2.8.2 Future plans

The current plugin has flaws concerning the formatting of the EAXML file, which needs to be corrected. Further, additional options for the import can be foreseen. Currently, only AnalysisFunction is supported, but any specialization of the FunctionType is a candidate for import. Further, the FunctionBehavior construct is currently not created and populated with FMU information, such as the path to the FMU file.

A more extensive potential addition concerns export. Export in the form of FMU generation is probably not appropriate since EAST-ADL is not primarily a behavioural definition language. However, export of the FMU:s linked to FunctionBehaviors to a simulation engine would be useful. This would concern creating S-functions in Simulink according to the connected FunctionPrototypes or to configure a Simulation manager to run the executables according to execution and connection information defined in the EAST-ADL model.

2.8.3 Requirements from WT2.1: Identifications of needs

MODELISAR FMU import is not explicitly mentioned in the requirements, although it relates to behavioural simulation in general.

DOW#0012 O2-2: Behavioural Simulation of EAST-ADL2 models

4SG#0003: Perform behavioural Simulation of EAST-ADL2 models according to performance evaluation standards

4SG#0004: Perform behavioural Simulation of EAST-ADL2 models according to standards covering communication with infrastructures

4SG#0019: The project shall enable to perform behavioural simulation according to ISO 8715: Electric road vehicles - Road operating characteristics

There are more similar requirements on behaviour simulations, see e.g.

4SG#0020, 4SG#0021, 4SG#0022, 4SG#0023, 4SG#0024, 4SG#0025, 4SG#0026

3 References

[1] www.atesst.org, accessed 2011-05-26

[2] www.edona.fr, accessed 2011-05-26

[3] www.artop.org, accessed 2011-05-26

[4] MAENAD: Deliverable D2.1.1, draft version 0.5

[5] www.modelisar.org, accessed 2011-05-26

[6] MAENAD: Deliverable D3.1.1