Advancing Traffic Efficiency and Safety through Software Technology, Phase 2 (ATESST2)

EAST-ADL

SEVENTH FRAMEWORK PROGRAMME

EUCAR

Foreword

How can the automotive-specific language EAST-ADL contribute in solving present challenges in the development of automotive systems? Cornerstones of high quality embedded systems are mature engineering techniques using model-based development and standardized, reliable architectures. Industrial experience shows that model-based development techniques are required in the automotive domain to improve the quality and dependability of embedded systems. EAST-ADL focuses mainly on the architecture-induced complexity of automotive embedded systems. It provides an ontology for automotive electronics and software, making system models unambiguous, consistent and exchangeable. The primary scope of the EAST-ADL modeling concepts used to be the single vehicle and its embedded system. During ATESST2, the scope of the approach has been extended to cooperative active safety systems, thus covering also the interaction with systems in other vehicles and infrastructure.

The ATESST2 project team has a strong background in the automotive domain. As a consequence, the developed language – while state of the art – can be flexibly introduced and adapted in practical applications, reflecting in particular the fragmented development process between vehicle manufacturer and automotive supplier, the highly complex variability of the systems, the different automotive-relevant standards such as AUTOSAR, SysML, AADL, Marte… and the need for highly dependable systems.

The EAST-ADL has gained high visibility also in the context of the CESAR project, a flagship project of the Joint Undertaking Artemis addressing development processes for safety relevant embedded applications, comprising some 20 large enterprises in the transportation domain, and also addressing automation. In an ongoing cooperation between ATESST2 and CESAR, the first version of the CESAR reference technology platform builds on an augmentation of the EAST-ADL meta-model with the key concept of contract based design.

EAST-ADL aims to set a new standard in the automotive domain. Users of AUTOSAR, the international, standardized automotive software architecture and exchange format, will be able to apply the EAST-ADL to cover complementary information and more abstract modeling levels. In short, with EAST-ADL the ATESST2 team has provided a highly relevant model-based development technique for those who are seeking to improve industrial automotive system development.

Stefan Bunzel (Continental Systems & Technology Automotive)
Werner Damm (Universität Oldenburg)

Table of Contents

1 Thematic Overview

The inclusion of embedded system technology in vehicles has had – and is still having – a radical impact on their development, production and maintenance. Over the past decades, automotive embedded systems have evolved from single standalone computer systems, simple enough to be designed and maintained with a minimum of engineering, to distributed systems including several networks, large numbers of sensors, electric motors and points of interactions with humans. These distributed systems provide enormous opportunities for the future, but at the same time require new skills, methodologies, processes and tools.

Future systems may be distributed over several vehicles and not be contained in one. Co-operative system engineering needs is one area where methodologies needs to be extended. The ATESST2 project has a target to find principles for how the information regarding a multi-vehicle system should be defined for engineering use.

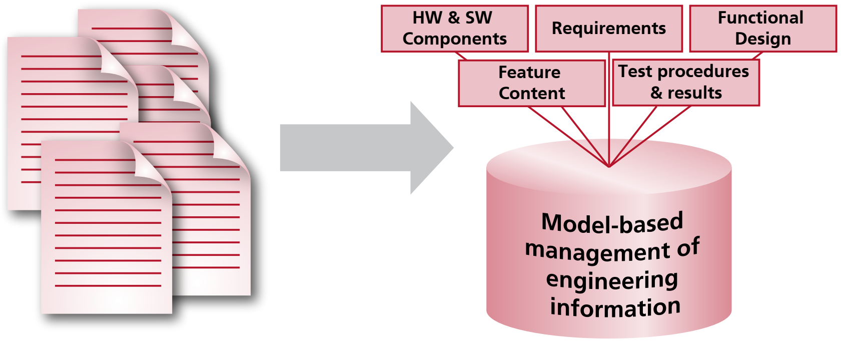

HW & SW Components
Requirements
Functional Design
Feature Content
Test procedures & results
Model-based management of engineering information

Figure 1. Embedded systems are highly complex - the extensive engineering information needs to be managed adequately.

Current methods for automotive embedded systems development lack systematic approaches and support for information management, architecting, software product lines, requirements and verification. Solutions relying on social communication and traditional text-based communication do not scale for handling advanced embedded systems. Software architectures and/or exchange format standards such as AUTOSAR offer a significant improvement of the current state of practice. However, experience tells us that advanced and complex systems also require model-based design encompassing higher levels of abstraction and multiple concerns to support cost-efficient and effective development.

In model-based development, computerized models are used to support communication, documentation, analysis and synthesis as part of system development. In such an approach, the models form the basis for the interactions between the organization's teams, information flow within and between development phases and for the design decisions made.

In both industry and research, there are strong trends toward domain-specific modeling languages. Many research efforts also address the need for managing models and integrating the plethora of models and tools that are used today in embedded systems development. The very strong potential of these approaches lies in their support for early, iterative and consistent development, and reuse.

A holistic approach to automotive embedded systems modeling needs to address several concerns, from features to implementation over structure and behavior, environment modeling and requirements to verification and validation information. Such an approach also needs to consider mapping and interoperability with existing tools and moreover, for industrial acceptance, to provide tools and be standardized. By developing EAST-ADL, the ATESST2 research project aims to provide a basis for such a holistic approach.

2 Introduction

EAST-ADL is an Architecture Description Language (ADL) initially defined in the ITEA EAST-EEA project and subsequently refined and aligned with the more recent AUTOSAR automotive standard in the FP6 and FP7 ATESST projects. EAST-ADL is an approach for describing automotive electronic systems through an information model that captures engineering information in a standardized form. Aspects covered include vehicle features, functions, requirements, variability, software components hardware components and communication.

EAST-ADL contains several abstraction levels (*see Figure 2*). The software- and electronics-based functionality of the vehicle are described at different levels of abstraction. The proposed abstraction levels and the contained elements provide a separation of concerns and an implicit style for using the modeling elements. The embedded system is complete on each abstraction level, and parts of the model are linked with various traceability relations. This makes it possible to trace an entity from feature down to components in hardware and software.

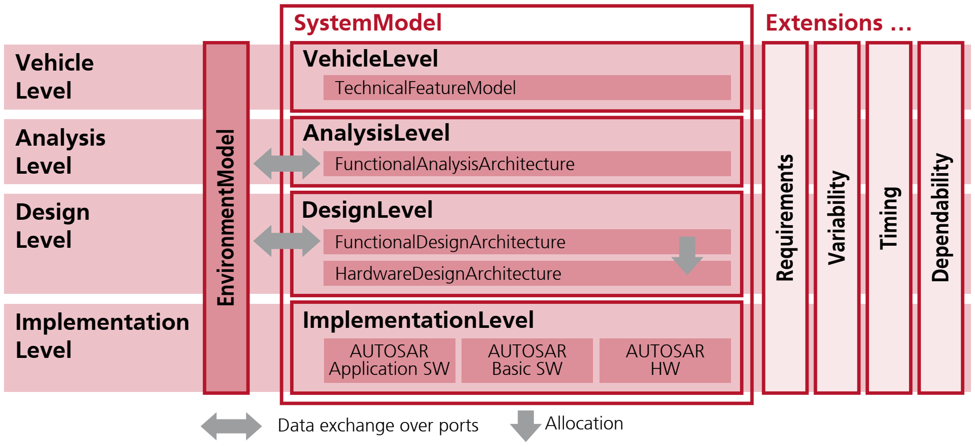


Figure 2. EAST-ADL abstraction levels and model organization.

The features in the "TechnicalFeatureModel" at the vehicle level represent the content and properties of the vehicle from top-level perspective without exposing the realization. It is possible to manage the content of each vehicle and entire product lines in a systematic manner.

A complete representation of the electronic functionality in an abstract form is modeled in the Functional Analysis Architecture (FAA). One or more entities (analysis functions) of the FAA can be combined and reused to realize features. The FAA captures the principal interfaces and behavior of the vehicle's subsystems. It allows validation and verification of the integrated system or its subsystems on a high level of abstraction. Critical issues for understanding or analysis can thus be considered, without the risk of them being obscured by implementation details.

The implementation-oriented aspects are introduced while defining the Functional Design Architecture (FDA). The features are realized here in a function architecture that takes into account efficiency, legacy and reuse, COTS availability, hardware allocation, etc. The function structure is such that one or more functions can be subsequently realized by an AUTOSAR software component (SW-C). The external interfaces of such components correspond to the interfaces of the realized functions.

The representation of the implementation, the software architecture, is not defined by EAST-ADL but by AUTOSAR. However, traceability is supported from implementation level elements (AUTOSAR) to vehicle level elements.

The Hardware Design Architecture (HDA) should be considered parallel to application development. On the design level and down, the HDA forms a natural constraint for development and the hardware and application software development needs to be iterated and performed together. There is also an indirect effect of hardware on the higher abstraction levels. Control strategies or the entire functionality may have to be revised to be implemented on a realistic hardware architecture. This reflection of implementation constraints needs to be managed in an iterative fashion.

To verify and validate a feature across all abstraction levels, using simulation or formal techniques, an environment model is needed early on. This "plant model" captures the behavior of the vehicle dynamics, driver, etc. The core part of the environment model can be the same for all abstraction levels.

After this short introduction to the EAST-ADL concepts, we go on to discuss the motivation and modeling concepts in more detail.

3 Challenges for Modeling Automotive Embedded Systems

Automotive embedded systems have evolved enormously over the past decades. The use of electronics and software in automotive products has grown exponentially. For example, today vehicles in series production contain the same amount of electronics as an aircraft did two decades ago. To satisfy customer demands and competitiveness between OEMs, innovation will further drive the significance of software-controlled automotive electronics over the next decade. It is obvious, that the vehicle's electronic architecture will continue to grow in complexity, criticality and authority.

To manage some of the challenges of automotive software, the AUTOSAR consortium has developed a standardized automotive software architecture. One of its main features is a componentization of the software architecture, to favor reuse and assist collaboration and integration aspects. The software development effort is no longer bound to a specific hardware platform or a particular provider. A standardized software architecture and methodology is a first step towards meeting the challenges connected with the development of automotive systems, often distributed over several suppliers with different responsibilities.

However, there still remains the critical issue of managing the overall engineering information to control system definition. This stage contains the most decisive steps in meeting safety challenges, controlling complexity and avoiding development errors and delays. Many stakeholders are involved here, and development is distributed over several departments and locations and involves several suppliers.

While system modeling and model-based development is the trend in the automotive industry to solve this issue, there are diverse company-specific solutions. There is no standardized approach to support system modeling of the engineering information. A federation of different modeling language initiatives is required to develop an automotive domain-specific language that is also in line with non-automotive approaches.

To support complexity and facilitate component development, an adequate organization of the system model is important. Representing the system in several "models" at different abstraction levels is a way to ensure separation of concerns and allow smooth interaction between disciplines. Supporting a functional decomposition of the system is also important to hide implementation aspects while the functional aspects are addressed.

Another challenge is the capability to use product line engineering. Today, component reuse is state of the art in the automotive industry. The organization and structuring of a product line approach, from feature selection up to decomposition into components, requires innovative and efficient techniques.

Finally, an important challenge is assessing the dependability of the application. What is needed are means for early evaluation of system architecture, in terms not only of functional properties, but also of non-functional ones (such as timing, resource, safety level, etc.). In this context, the application of the upcoming standard for functional safety (ISO DIS 26262) must be prepared by introducing new techniques and a structured development approach. An architecture description language provides means to represent the safety life-cycle information according to the requirements of the standard.

Last but not least, tool support for engineering development is organized today as a patchwork of heterogeneous tools and formalisms. A backbone environment using a standardized modeling language has to be harmonized to drive the tool market.

4 EAST-ADL Meta-Modeling Approach

This chapter outlines the modeling approach used in the ATESST2 project, which is to define a domain model for the EAST-ADL language and implement it as a UML2 profile.

4.1 EAST-ADL Domain Model

The EAST-ADL language is formally specified as a meta-model capturing domain specific (i.e. automotive) concepts. The meta-model follows guidelines originating from AUTOSAR for definition of templates. This is, concepts are represented by basic concepts of UML2 (*www.uml.org*) supplemented by the AUTOSAR template profile. The meta-model thus fits as a specification of a domain specific tool environment, and also defines an XML exchange format. This domain model represents the actual definition of the EAST-ADL language and constitutes the heart of the EAST-ADL language specification.

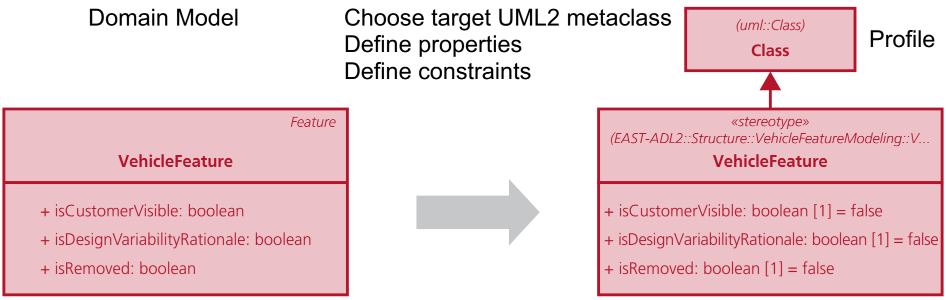

Domain Model
Choose target UML2 metaclass
Define properties
Define constraints
(uml::Class)
Class
Profile
Feature
VehicleFeature
+ isCustomerVisible: boolean
+ isDesignVariabilityRationale: boolean
+ isRemoved: boolean
«stereotype»
(EAST-ADL2::Structure::VehicleFeatureModeling::V...
VehicleFeature
+ isCustomerVisible: boolean [1] = false
+ isDesignVariabilityRationale: boolean [1] = false
+ isRemoved: boolean [1] = false

Figure 3. The domain model specifies the properties of the profile stereotype.

4.2 UML2 Profile

The EAST-ADL language is also implemented as a UML2 profile. UML profiles are standard extension mechanisms in the UML2 language, in which domain-specific concepts are provided as tags applicable to a selected subset of UML2 elements (such as classes, properties, ports, etc.) giving them different meaning and extra properties. The profile allows users to do system modeling according to the EAST-ADL semantics using off-the-shelf UML2 tools. Generally, two approaches are possible depending on the tool: either the user defines a UML2 user model and then applies "stereotypes" from the EAST-ADL profile to the elements, or the tool provides a more advanced profile support, allowing the direct creation of domain specific elements, as stereotyped UML elements, with a dedicated toolbar for instance. Constraints are also part of the profile definition; this makes it possible to constrain the rich set of modeling constructs allowed by UML2 and to validate the conformance of the model. The EAST-ADL profile is delivered as an XMI file ready for use in UML2 tools.

In the definition of the EAST-ADL profile, the general strategy has been to provide stereotype properties even for properties already populated within the UML2 superstructure. In other words, the property values that appear when defining a UML2 model are duplicated with semantic names in the stereotypes. This yields a model that is quite complete even without a profile (*see Figure 3*).

This approach is in line with the intention of UML2 that views and features of existing UML2 tools can be used readily, including for example, UML2 activity diagrams and related profiles such as SysML (*www.omgsysml.org*) and MARTE (*www.omgmarte.org*). The applied profile adds automotive semantics to this self-contained UML2 model.

5 EAST-ADL Modeling Concepts

In this section, the EAST-ADL modeling concepts are described in more detail for six areas: functional abstraction (Section *7.1*), timing modeling (Section *7.2*), requirements modeling (Section *7.3*) functional safety modeling (Section *7.4*), variability modeling (Section *7.5*) and cooperative active safety systems (Section *7.6*).

5.1 Functional Abstraction

EAST-ADL provides the means to capture the functional decomposition and behavior of the embedded system and the environment.

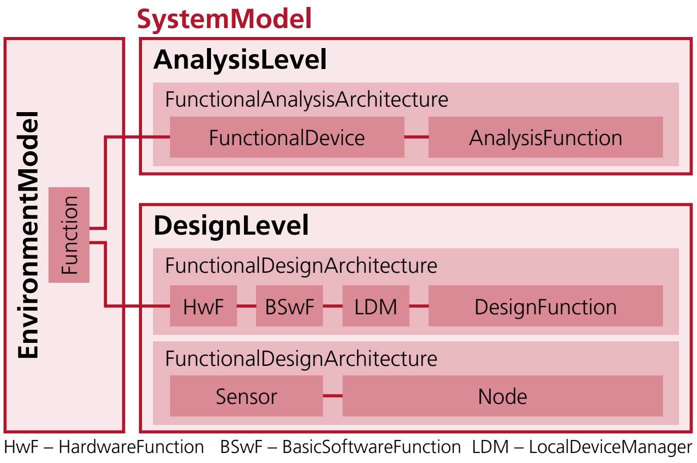


Figure 4. The "AnalysisLevel" and the "DesignLevel" and connections to the "EnvironmentModel".

At the analysis level, the "FunctionalAnalysisArchitecture" contains "Functions" that can be hierarchically composed and connected to each other. Functional devices represent sensors and actuators with their interface software and electronics, and these are connected to the environment. *Figure 4* explains the entities involved and shows how they are connected.

The “Functions” can have two types of ports, “FlowPorts” and “ClientServer” ports to represent data exchange and client-server interaction. The functions can be hierarchical, but the leaves have synchronous execution semantics, which means that they read inputs, calculate and provide outputs. They are triggered based on time or data arrival on ports. A function‘s internal behavior is typically defined by external tools and their techniques for behavioral descriptions.

The behavior of the environment is captured in the “EnvironmentModel”. The environment model also contains “Functions”, but they represent vehicle dynamics, other vehicles, road-side IT systems, etc.

The design level (*see Figure 4*) contains a more detailed functional definition of the system. “Functions” and “LocalDeviceManagers” represent application software in the Functional Design Architecture. “BasicSoftwareFunctions” are used to capture middleware behavior affecting application functionality. HardwareFunctions represents the logical behavior of hardware components and complete the logical path to the environment model with the controlled “plant” and surrounding elements. The Hardware Design Architecture represent the resource platform with ECUs, busses, sensors, actuators and I/O to which the functions are allocated. The Hardware Design Architecture also reflects the physical topology of electrical elements and connectors.

5.2 Timing Modeling

EAST-ADL provides support for model-specific engineering information, including non-functional properties that are relevant for the timing of automotive functions. Conceptually, timing information can be divided into timing requirements and timing properties, where the actual timing properties of a solution must satisfy the specified timing requirements.

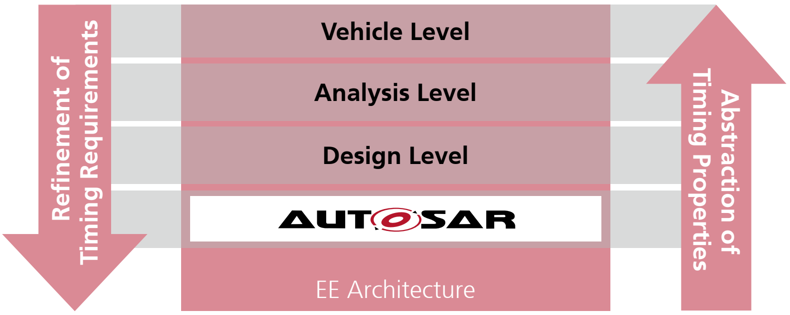

Refinement of Timing Requirements
Vehicle Level
Analysis Level
Design Level
AUTOSAR
EE Architecture
Abstraction of Timing Properties

Figure 5: Timing information as seen in the EAST-ADL system model.

Figure 5 gives an overview of how timing information is seen in the EAST-ADL system model. Note that the start of the arrows describes the origin of the timing-related engineering information, and the direction of the arrow (top-down or bottom-up) describes their inter-abstraction-level relation.

Modeling of timing requirements and properties on the functional abstraction levels of the architecture description language is done by means of the „Timing Augmented Description Language“ TADL developed by the TIMMO project. The implementation level, i.e. AUTOSAR, is addressed by the „AUTOSAR Timing Extensions“ which are introduced in AUTOSAR release 4.0. These extensions are based on TADL concepts, too.

Timing constraints are defined separately from the structural modeling and reference structural elements of the EAST-ADL. The requirements support in EAST-ADL allows for tracing from solutions as modeled in the structural model to requirements, and from verification cases to requirements. The TADL constraints fit in the requirement support as refinements of the requirements.

The fundamental concept for describing timing constraints is that of Events and Event Chains. On every level of abstraction, events can be identified, i.e. a stimulus, that causes a reaction and such a reaction leads to another observable event, i.e. a response.

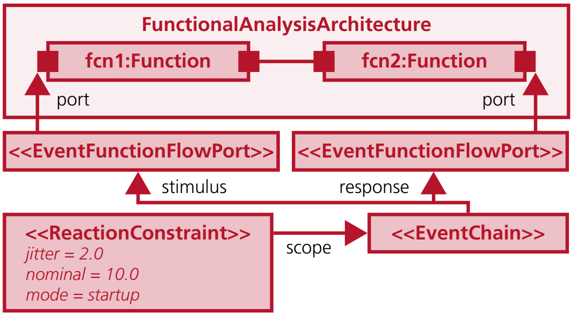


Figure 6: Event Chain with associated constraint

Timing requirements can be imposed on Event Chains, for example, specifying that the time between the occurrence of a stimulus event and the occurrence of the expected response event shall not exceed a specific amount of time – i.e. an end-to-end delay from a sensor to an actuator, or the response event shall not occur before a specific amount in time and not later than a specific amount of time after the point-in-time the stimulus event has occurred. In addition, requirements regarding the synchrony of events can be expressed as well, stating that a number of events shall occur „simultaneously“

in order to cause a reaction, or be considered as valid response of a system function. For example, in case of a passenger vehicle, its brake system shall apply the brakes simultaneously; or the exterior light system shall simultaneously turn on and off the rear- and front turn signal indicators.

Figure 6 shows a simple example of an event chain with an annotated reaction constraint. A Functional Analysis Architecture with two functions builds up the structural model. The constraint with bound attributes has been defined. This refers to an Event Chain built up by an in event (stimulus) and an out event (response) referring to structural ports.

5.3 Requirements Modeling

In order to comprehensively support the development of complex automotive systems, EAST-ADL provides means for requirements specification, i.e. for specifying the required properties of the system (at varying degrees of abstraction). Furthermore, requirements can be refined by behavioral models, they can be traced between system refinement and system decomposition levels, and they can be related to verification and validation information and activities. Another important aim of EAST-ADL is to provide means for project-specific adjustments to requirements specification structures, which are inspired by the Requirements Interchange Format (RIF, *www.automotive-his.de/rif/*).

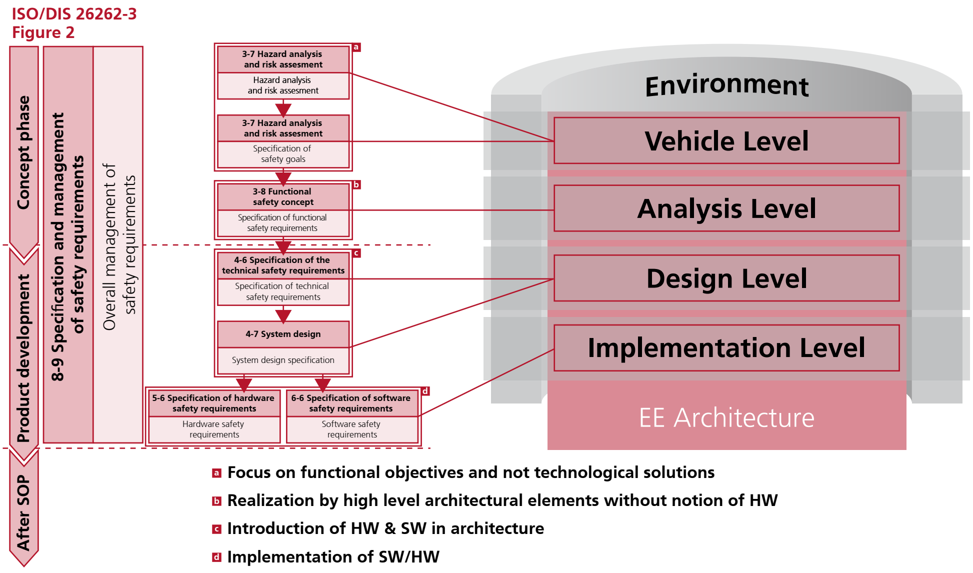
Methodically, EAST-ADL differentiates between functional requirements, which typically focus on some part of the “normal” functionality that the system has to provide (e.g. “ABS shall control brake force via wheel slip control”), and quality requirements, which typically focus on some external property of the system seen as a whole (e.g. performance, “ABS shall reduce stopping distance on snow by 40%”).

EAST-ADL offers detailed means to model artifacts of verification and validation activities and to relate these artifacts to requirements. This allows us to explicitly and continuously plan, track, update and manage important V&V activities and their impact on the system parallel to the system’s development.

5.4 Functional Safety Modeling

The overall objective of the support for functional safety modeling is to enforce explicit considerations of safety concerns throughout an architecture design process, which includes all safety related information that are necessary for developing a safety critical E/E system, in compliance with the Standard ISO 26262 (an international standard dedicated to functional safety for road vehicles).

As an overall system property, safety is concerned with anomalies (e.g. faults, errors and failures) and their consequences under certain environmental conditions. It is one particular aspect of system dependability that normally also encompasses reliability, availability, integrity, maintainability and security. Functional safety represents the part of system safety that depends on the correctness of a system in performing its intended functionality. In other words, it addresses the hazardous events of a system during its operation (e.g. component errors and their propagations).


ISO/DIS 26262-3
Figure 2
Concept phase
Product development
After SOP
8-9 Specification and management of safety requirements
Overall management of safety requirements
3-7 Hazard analysis and risk assesment
Hazard analysis and risk assesment
3-7 Hazard analysis and risk assesment
Specification of safety goals
3-8 Functional safety concept
Specification of functional safety requirements
4-6 Specification of the technical safety requirements
Specification of technical safety requirements
4-7 System design
System design specification
5-6 Specification of hardware safety requirements
Hardware safety requirements
6-6 Specification of software safety requirements
Software safety requirements
a
b
c
d
Environment
Vehicle Level
Analysis Level
Design Level
Implementation Level
EE Architecture

a Focus on functional objectives and not technological solutions

b Realization by high level architectural elements without notion of HW

c Introduction of HW & SW in architecture

d Implementation of SW/HW

Figure 7: Mapping of ISO26262 information to EAST-ADL abstraction levels

EAST-ADL facilitates safety engineering in terms of safety analysis, specification of safety requirements, and safety design. While promoting safety in general through its intrinsic architecture modeling and traceability support, EAST-ADL provides explicit support for efficient integration of functional safety activities along with the nominal architecture design and evolution.

EAST-ADL provides language-level support for the concepts defined in ISO 26262, including vehicle-level hazard analysis and risk assessment, the definition of safety goals and safety requirements, the ASIL (Automotive Safety Integrity Level) decomposition and the error propagation. The information is included in the Dependability package, as an extension of the nominal architecture model.

Following a top-down approach, the safety analysis starts at the VehicleLevel, beginning with the identification and description of the item. An item, as defined in ISO 26262, is a system or array of systems or functions that is of particular concern in regards to functional safety. Through hazard analysis and risk assessment activities, is possible to preliminarily evaluate at VehicleLevel the “safety relevance” of the item under safety analysis, to define the safety goal (top-level safety requirement) for each hazardous event (hazard evaluated in different scenarios) and to classify them in terms of ASIL. Moreover, AnalysisLevel and DesignLevel of EAST-ADL support respectively the functional safety concept and the technical safety concept definition.

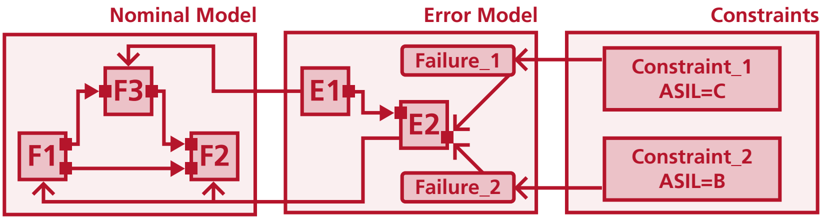


Figure 8: EAST-ADL error model as a separate architecture view extending the nominal architecture model.

EAST-ADL error modeling allows to capture detailed information about the failure behavior of the system and thus enables a safety analysis to determine whether technical safety requirements are being met. This Error Model describes the generation and propagation of failures through the system. The relationships of local error behaviors are captured by means of explicit error propagation ports and connections. Within an error model, the syntax and/or semantics of existing external formalisms can be adopted for a precise description of the error logic. The specification captures what output failures of the target architecture component are caused by what faults of this component. This, together with the error propagation links, makes it possible to perform safety simulations and analyses through external analysis tools. In an architecture specification, an error is allowed to propagate via design specific architectural relationships when such relationships also imply behavioral or operational dependencies (e.g. between software and hardware).

The error modeling is treated as a separate analytical view (*see Figure 8*). It is not embedded in a nominal architecture model but seamlessly integrated with the architecture model through the EAST-ADL meta-model. This separation of concerns in modeling is considered necessary in order to avoid some undesired effects of error modeling, e.g. relating to the comprehension and management of nominal design, reuse, and system synthesis (e.g. code generation).

Given an error model, the analysis of the causes and consequences of failure behaviors can be automated through tools. There is currently a (prototype) analysis plug-in in the Eclipse environment allowing the integration of the HiP-HOPS tool (Hierarchically Performed Hazard Origin and Propagation Studies[1]) for static safety analysis in terms of FFA, FTA, and FMEA. The analysis leverage includes fault trees from functional failures to software and hardware failures, minimal cut sets, FMEA tables for component errors and their effects on the behaviors and reliability of entire system.

In EAST-ADL, a safety requirement derived from the safety analysis has attributes specifying the hazard to be mitigated, the safety integrity level (ASIL), operation state, fault time span, emergency operation times, safety state, etc. The safety requirement is then traced to or used to derive other nominal requirements, e.g. relating to safety functions and performance.

5.5 Variability Modeling

In order to give an overview of variability management in EAST-ADL, we examine two questions:

1. In what development situations and contexts is variability management needed? Or: For what parts of the EAST-ADL is variability management support provided?
2. What are the basic modeling means used for variability modeling and to which of these development situations/contexts are they applicable?

First, variability management starts on the vehicle level, where model range features and variability is represented. At this point, the purpose of variability management is to provide a highly abstract overview of the variability in the system such as the complete system together with dependencies between these variabilities. A "variability" in this sense is a certain aspect of the complete system that changes from one variant of the complete system to another. "Abstract" here means that, for an individual variability, the idea is not to specify how the system varies with respect to this variability but only that the system shows such variability. For example, the front wiper may or may not have an automatic start. At vehicle level, the impact of this variability on the design is not defined; only the fact that such variability exists is defined by introducing an optional feature named „RainControlledWiping“. This is subsequently validated and refined during analysis and design.

One or more feature models may be defined on the vehicle level: the so-called core Technical Feature Model is used to define the complete system's variability on a global level from a technical perspective, whereas one or more optional Product Feature Models can be used to define views on this technical variability which can be tailored to a particular view-point or purpose, e.g. the end-customer perspective.

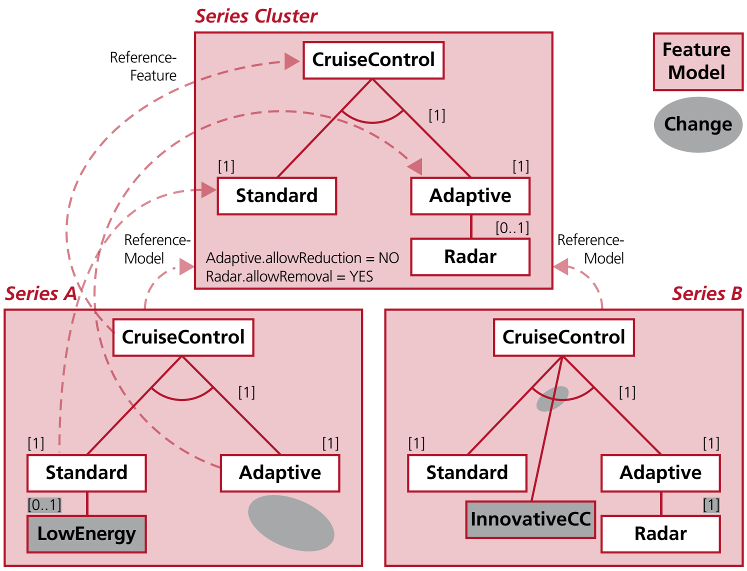


Figure 9. Reference model and referring models in the multi-level feature modeling approach.

While the details of how variability is actually realized in the system are largely suppressed at the vehicle level, they are the focus of attention when managing variability in other areas of the development process. In fact, specific variability may lead to modifications in any development artifact, such as requirements specifications and functional models. With respect to EAST-ADL, three areas must be distinguished: (1) requirements, (2) the artifacts on analysis, design and implementation level, and (3) test artifacts. Here, describing that a specific variability occurs is not sufficient; it is necessary to describe *how* each variability concept affects and modifies the corresponding artifact.
Having answered question no. 1 above, we can now turn our attention to the second question: the basic modeling means provided as support for variability management in these different situations. They are: feature modeling, configuration decision modeling and multi-level feature trees.

The purpose of feature modeling is to define the commonalities and variabilities of the product variants within the scope of a product line. Feature models are normally used on a high level of abstraction, as described above for vehicle level variability. However, in EAST-ADL, they are also used on analysis and design levels and acquire a much more concrete meaning there. Configuration decision modeling, on the other hand, is aimed at defining configuration: the configuration of a feature model f_T – i.e. the selection and de-selection of its features – is defined in terms of the configuration of another feature model f_S. A configuration decision model can thus be seen as a link from f_S to f_T that allows us to derive a configuration of f_T from any given configuration of f_S. Finally, multi-level feature trees (*see Figure 9*) are a means to strategically manage two or more separate, independent product lines. With this instrument at hand, not all variants of the complete system need to be managed within a single, extremely complex global product line. It is, instead, possible to subdivide the product line into smaller, subordinate product lines (called product sublines) without losing the possibility to manage them from a global perspective.

Variability management on the artifact level is driven by the variability captured on the vehicle level. This means that the main driver for variability and also variability instantiation is the vehicle-level feature model. Variability on the artifact level essentially consists of the definition of variation points within these artifacts. In addition, feature models can be attached to functions in order to expose the variability within these functions and hide the actual structuring, representation and binding of this variability within a function. This way, the benefits of information hiding can now be applied to the variability representation and variability binding within the containment hierarchy of functions in the EAST-ADL Functional Analysis Architecture and Functional Design Architecture (called compositional variability management).

5.6 Cooperative Active Safety Systems

The primary scope of the EAST-ADL modeling concepts used to be the single vehicle and its embedded system. Admittedly, on vehicle level the Features account also for vehicle elements beyond the embedded system such as mechanical components, but still within the single vehicle. During ATESST2, the approach to also represent the interaction with systems in other vehicles and infrastructure has been addressed. Cooperative active safety systems rely as much on modeling of adjacent vehicles and systems as on the vehicle-internal environment for its proper specification and validation.

A consequence is that the EAST-ADL modeling elements have been organized as a system model that strictly represents one vehicle and its embedded system, and an environment model where models representing adjacent vehicles or infrastructure are placed. A pattern was identified where the environment model is organized as in-vehicle, near and far environment. The in-vehicle environment represents elements of the vehicle that are not part of the electrical architecture. The near environment represents road and whether interaction that directly and physically influence the vehicle. The far environment represents vehicles and infrastructure that interact logically with the vehicle.

6 EAST-ADL Methodology

The purpose of the EAST-ADL Methodology is to give guidance on the use of the language for the construction, validation and reuse of a well-connected set of development models for automotive embedded software. The purpose of the EAST-ADL methodology is not to define or even impose a specific automotive software development process. This does not mean that such a process is considered undesirable in principle, but the range of development activities of the EAST-ADL is very large and companies usually have grown processes in these areas. Therefore, it was a conscious decision of the ATESST2 project to leave this task outside its scope.

Given the complexity of the development activities in automotive embedded software development, it is mandatory to structure the methodology so as to enable a relatively fast and easy access to the EAST-ADL language for a small kernel of essential development activities which can then be seamlessly extended to a comprehensive treatment of the language including more specialized development activities which may not necessarily be used in every development project. Hence the methodology is structured into two major components.

- The main component, the core development part, comprises a top-down description of the most central concepts of the EAST-ADL methodology:
- The analysis of external requirements resulting in the allocation to a Technical Feature Model together with the definition of necessary or intended feature configurations. In addition, for each feature a set of requirements is specified at vehicle level.
- The creation of the FunctionalAnalysisArchitecture specifying a solution of the requirements without concern about implementation restrictions of automotive series development. The analysis model is a logical representation of the system to be developed together with its environment, and the boundary of the system to this environment. All the modeling in this phase will be on a logical behavior level, i.e. it will make no distinction between HW and SW or about the implementation of communication.
- The creation of the FunctionalDesignArchitecture specifying a solution to the requirements in terms of efficient and reusable architectures, i.e. sets of (structured) HW/SW components and their interfaces, a hardware architecture, and a mapping from functional components to HW/SW components. The architecture must satisfy the constraints of a particular development project in automotive series production.
- The HW/SW implementation and configuration of the final solution. This part is mainly a reference to the concepts of AUTOSAR which provides standardized specifications at this level of automotive software development.

The core methodology is extended into a comprehensive methodology for automotive development projects by adding three additional and orthogonal activities to each of these phases:

- Specification of Requirements and corresponding V&V cases to be executed and evaluated during the corresponding integration phase. V&V cases are most typically test cases, but can also include reviews etc.
- Verification of the model on a given abstraction level to the requirements of the model at the abstraction level directly above.
- V&V activities on the model artifacts of a given level itself, i.e. peer reviews, consistency checks, check of modeling guidelines etc.

While the methodology tries to be comprehensive handling the construction phases, the integration activities are only covered inasmuch they involve V&V activities and the relation to V&V-artifacts defined in the construction phases.

The EAST-ADL methodology is further extended by adding a set of complementary loosely-coupled extensions. These extensions can of course also be combined depending on project needs. The following extensions are currently included:

- Environment Modeling: modeling of the (typically analog or discrete-analog) environment of the system to be developed.
- Safety Assurance: development of Safety-critical systems.
- Timing: detailed handling of timing requirements and properties.
- Variability Modeling: detailed handling of variability modeling.
- Behavior modeling: detailed handling of behavioral modeling.

The modeling of the methodology itself is adopting major concepts of the Software & Systems Process Engineering Meta-model SPEM (*www.omg.org/spec/SPEM/*), which means that the methodology is based on a set of elementary work tasks which are performed by a set of actors and produce a set output artifacts from a set of input artifacts. These tasks are structured into disciplines and then presented to the end user by a set of views. This leads of a highly linked network of methodological activities in which an end user can easily navigate to get information and guidance on the use of the language for particular development tasks. Technically, the EPF process framework (*www.eclipse.org/epf/*), an Eclipse plugin, has been chosen as tool support since it fits very well with the modeling style described above. In particular, it allows to publish a html model as main methodological artifact for the end user.

7 Case Studies

Different case studies have been developed during the ATESST2 project, covering different topics and different abstraction levels of the EAST-ADL. They served as assessment and exemplary application of the results of the project. The applications include: a cruise control system, a brake system, a steering system, and a car-access security system. There was a close cooperation with the HAVEit project, which deals with highly automated driving technologies. The vehicle architecture defined by HAVEit has been modeled by means of the EAST-ADL, and the cruise control model has been integrated in this architecture. Furthermore, a safety analysis was performed for the cruise control system, based on the ISO DIS 26262, and applying the methods and tools developed in ATESST2. An example on modeling style supporting cooperative systems was developed, using as an example the cruise control system extended to platoon-driving functionality.

The cruise control case study is based on an existing AUTOSAR demonstrator that had been developed prior to ATESST2. Therefore, an AUTOSAR model covering the implementation level of the EAST-ADL was already existing. The existing hardware is re-used, and a system model according to the EAST-ADL and the HAVEit architecture was developed. The same hardware is also used for the security system case study.

The brake system is a distributed brake-by-wire system with basic functionality. It was modeled in EAST-ADL on all abstraction levels, and also built as a physical setup with brake pedal, electrical brake and a single wheel. The electrical architecture is a 2-ECU topology with FlexRay communication between. The SW platform used on each ECU is based on AUTOSAR, with the ABS application implemented on top as an AUTOSAR application consisting of multiple AUTOSAR SW components. The AUTOSAR BSW was provided by the Volvo AUTOSAR Platform (VAP), which includes all major BSW components, including communication, OS, diagnostics etc. VAP is based on AUTOSAR release 3.0.

The steering system consists of an electric column lock (ECL) and electric power-assisted steering (EPAS), which replaces hydraulic power-assisted steering (HYPAS). One of the objectives of the steering system case study was to show how a more complex environment/plant model can be represented in EAST-ADL. To show the differences between the different modeling styles of EAST-ADL, and how they relate to different tools, the EPAS system was modeled in two different ways. In the first case the demonstrator was modeled using FunctionFlowPorts, and in the other case a combination of FunctionFlowPorts and the newly introduced FunctionPowerPort was created. A MATLAB/Simulink model was created in order to model the behavior of the system. The objective was to modularize the simulation blocks into different components, each represented by a Simulink subsystem.

The Security System controls the armed state of a car, and the alarms given in case of an unwanted intrusion (security alarm) or a dangerous situation, where a panic alarm is raised. The security system has to collaborate with modules providing relevant data and events derived from sensors, and modules providing access to required outputs like the InfoAndWarn and the ExteriorLight. The goal of this case study was to evaluate methods to support modeling on abstraction levels with a special focus on modeling on design and implementation level and elaborating ways to get as much benefits as possible out of the model. Important design goals here were a 1:1 mapping to the code in order to avoid redundancies, simulation and analysis capabilities for purposes like timing and scheduling analysis and generation capabilities into runtimes as AUTOSAR. Input to the demonstrator was an already existing UML model of a security system, as well as a simulation and test environment for the security system. The security system was remodeled on all abstraction levels of EAST-ADL. A component structure was added, and the security system was ported to the AUTOSAR demonstrator.

8 Related Approaches

One key aspect of the development of EAST-ADL is to benefit from existing methods and techniques and also to influence emerging approaches. Whenever possible, existing and state-of-the-art solutions were reused and integrated in the language. This favors the wide use of the language, allows the use of available tools and prepares for a sound standardization process.

Projects like AUTOSAR, TIMMO, and ISO 26262 are sources both for the alignment of domain specific challenges and for the integration of technologies and methodologies in the development of EAST-ADL.

As a future de-facto standard for automotive embedded systems, AUTOSAR addresses the needs for a process-safe integration of functions. It provides a standardized platform for the specification and execution of application software, an integration method for software components and hardware resources, and also the interchange formats that these require. While adopting AUTOSAR for the implementation level abstractions, the EAST-ADL language complements the AUTOSAR initiative by providing higher-level abstractions, analysis and lifecycle management support. In effect, it allows an AUTOSAR-compliant software architecture being extended with models relating to the design of functionality, timing and safety, the structuring and allocation of application, as well as the management of variability, requirements, traceability and verification and validation.

EAST-ADL integrates the results of TIMMO, which is an ITEA project focusing on the timing constraints and timing properties in automotive real-time systems. TIMMO has developed a formal description language and a methodology for dealing with the timing concerns on the basis of EAST-ADL1. It has been developed in a close collaboration with AUTOSAR.

The emerging international standard ISO 26262 is carefully considered in EAST-ADL. The key content includes an automotive safety lifecycle, an automotive specific approach for determining risk classes and deriving safety requirements based on ASILs (Automotive Safety Integrity Levels), and a set of requirements for validation and confirmation measures to ensure a sufficient and acceptable level of safety being achieved.

To support behavior modeling, EAST-ADL provides dedicated behavior stereotypes that facilitate the description of the relationship between behavioral and structural models and alter the UML2 semantics such that among other things, triggering policies and run-to-completion assumption hold. By clearly distinguishing between component execution and component logical computation, EAST-ADL allows the integration of behavior models from off-the-shelf tools like SCADE, ASCET, Simulink, etc., according to lifecycle stages and stakeholder needs. For continuous-time behavior (e.g., for the vehicle dynamics under control), related modeling techniques from Modelica, which combines causal modeling with object-oriented thinking, have been adopted. EAST-ADL also provides tool prototypes for model transformation to Simulink and the SPIN (Simple PROMELA Interpreter) model checker.

A further standardization effort being taken into consideration is the SAE "Architecture and Analysis Description Language" (AADL), which has its roots in the avionics domain. Compared to EAST-ADL, AADL has a more narrow scope: no explicit support is provided for variability management or requirements refinements and traceability. Specifics for automotive systems such as the networks are weakly supported. The AADL is not designed for mass-produced systems and therefore has less emphasis on optimized overall solutions e.g. by considering compact runtime systems. For the automotive domain, the clash with AUTOSAR concepts is also a problem. However, wherever applicable, AADL concepts were reused, e.g. for dependability modeling.

EAST-ADL allows the adoptions of existing formalisms for the underlying semantics and provides support for model transformation and tool interoperability with the external safety analysis techniques. In particular, HiP-HOPS and the AADL's Error Model Annex have been carefully considered in the development of EAST-ADL. They both enable the modeling of system failure behavior and allow analysis of that behavior using tools.

In ATTEST2, a tool plug-in for HiP-HOPS has been developed to support both FTA and FMEA. Other approaches to model-based safety analysis and verification that have been investigated for the development of EAST-ADL include ISSAC and its predecessor ESACS in the aerospace industries (where the goal was to develop a formal methodology and tools for the safety analysis of complex aeronautical systems), the ASSERT project (with similar goals but more focused on software intensive systems specified in AADL), the SETTA project (focusing on the use of time-triggered architectures in automotive systems), and the SAFEDOR project (which aimed to develop new practices for the safety assessment of maritime systems).

SPEEDS (Speculative and Exploratory Design in Systems Engineering) is a recent European project (FP6) aiming at providing support for modeling and analysis of complex embedded systems through the usage of formal analysis tools. EAST-ADL complements the SPEEDS approach with automotive architecture and lifecycle information. The techniques of SPEEDS have been considered in EAST-ADL for behavior modeling (i.e., with the hybrid automata variant) and for a more formal specification of requirements and constraints (i.e., with temporal logics scripts for contracts of functionality, safety, and timing).

9 Conclusions and Discussion

The introduction of information technology in the automotive industry over the past decade and the fact that electronic control units have become more and more inter-connected has led to advanced functions that were unimaginable fifteen years ago. However, with such highly advanced, extremely complex functions, appropriate design methods and tools have become crucial.

This has necessitated further research in model-based development to meet automotive needs. It is the main motivation for the definition of an architecture description language for automotive embedded systems. The EAST-ADL language has been refined by taking particular care to reuse standardized language constructs and to align the language with other existing related standardization initiatives. In addition, common standards were also applied on the language specification level. In particular, the EAST-ADL domain model has been tailored to compliance with AUTOSAR meta-modeling guidelines.

The decision to release a UML2 implementation of the language was made to allow a wider community to use the language, provide feedback and help improving the language. The profile can be used in several UML2 modeling environments that support the use of UML2 profiles. Within the ATESST2 project, an Eclipse-based open source tool platform has been developed, centered around a UML2 modeling and profiling environment called Papyrus UML (*www.papyrusuml.org*). A number of special-purpose Eclipse plug-ins have also been developed.

The main achievement of the ATESST2 project is a major refinement of the architectural description language EAST-ADL and its extension to cover cooperative active safety systems. The language complements the AUTOSAR standard with support for the early development phases and a seamless transition to the implementation level as defined by AUTOSAR. For these phases it provides a common terminology and serves as a basis for improved tool interaction and information exchange within as well as between companies.

Although the EAST-ADL has progressed during the ATESST2 project, many challenges remain in the area of model based development of automotive embedded systems. Consequently, several projects and companies will continue working on the language. For this reason, ATESST2 partners are initiating a consortium to maintain the EAST-ADL language, the EAST-ADL Association. Stakeholders that use and develop EAST-ADL tools, methods and concepts will thus have a platform to harmonize efforts and share experience even after the end of this project.

10 Selected ATESST Publications

Please find a language overview and all public deliverables of the ATESST project on the web: *www.atesst.org*

P. Cuenot, P. Frey, R. Johansson, H. Lönn, D. Servat, R. Tavakoli Kolagari, M. Törngren, M. Weber: *"Engineering Support for Automotive Embedded Systems — Beyond AUTOSAR"*, in ATZautotechnology Volume 9, April 2009.

A. Abele, R. Johansson, H. Lönn, Y. Papadopoulos, M.-O. Reiser, D. Servat, M. Törngren, M. Weber: *"The CVM Framework — A Prototype Tool for Compositional Variability Management"*, VAMOS 2010.

A. Sandberg, D. Chen, S. Torchiaro, R. Johansson, H. Lönn, L. Feng, M. Törngren, R. Tavakoli Kolagari, A. Abele: *"Model-based Safety Engineering of Interdependent Functions in Automotive Vehicles Using EAST-ADL2"*, SAFECOMP 2010.

L. Feng, D. Chen, H. Lönn, M. Törngren: *"Verifying System Behaviors in EAST-ADL2 with the SPIN Model Checker"*, IEEE International Conference on Mechatronics and Automation, August 2010.

Y. Papadopoulos, M. Walker, M.-O. Reiser, D. Servat, A. Abele, R. Johansson, H. Lönn, M. Törngren, M. Weber: *"Automatic Allocation of Safety Integrity Levels"*, 8th European Dependable Computing Conference – CARS workshop, April 2010.

P. Cuenot, P. Frey, R. Johansson, H. Lönn, Y. Papadopoulos, M.-O. Reiser, A. Sandberg, D. Servat, R. Tavakoli Kolagari, M. Törngren, M. Weber: *"The EAST-ADL Architecture Description Language for Automotive Embedded Software"*, in: H. Giese, G. Karsai, E. Lee, B. Rumpe, B. Schätz (Eds.): "Model-Based Engineering of Embedded Real-Time Systems", LNCS 6100, Springer, October 2010 (in print).

11 Contact and Further Information

The partners engaged in the ATESST2 consortium represent a balance between vehicle manufacturers, automotive suppliers and tool vendors, and universities. The partners include large corporations as well as small and medium size companies. Volvo Technology is the project coordinator.

Company	Contact	Mail
Vehicle Manufacturers		
• Carmeq	Matthias Weber	matthias.weber@carmeq.com
• Centro Ricerche Fiat	Fulvio Tagliabò	fulvio.tagliabo@crf.it
• Volvo Technology (coordinator)	Henrik Lönn	henrik.lonn@volvo.com
Automotive Suppliers		
• Continental Automotive	Friedhelm Stappert	friedhelm.stappert@ continental-corporation.com
• Mecel	Anders Sandberg	anders.sandberg@mecel.se
Tool Vendor		
• MentorGraphics	Rolf Johansson	rolf_johansson@mentor.com
Academic		
• Commissariat à l'Énergie Atomique LIST	David Servat	david.servat@cea.fr
• Kungliga Tekniska Högskolan Stockholm	Martin Törngren	martin@md.kth.se
• Technische Universität Berlin	Mark-Oliver Reiser	moreiser@cs.tu-berlin.de
• University of Hull	Yiannis Papadopoulos	Y.I.Papadopoulos@hull.ac.uk

ATESST2

ATESST project website: www.atesst.org

For more information contact: atesst-coordinator@vtec.volvo.se