Advancing Traffic Efficiency and Safety through Software Technology (ATESST)

Information Society Technologies

Safety

EUCAR

1 Foreword

An *Architecture Description Language* (ADL) for automotive, software-intensive systems – why does it matter? Experience has shown that the comprehensive development of automotive systems needs more attention.

How can the automotive-specific language *EAST-ADL2* help? 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. The focus of EAST-ADL2 developed in ATESST is to adequately meet the engineer's needs regarding practical methods and engineering information management. EAST-ADL2 focuses mainly on the architecture-induced complexity of automotive embedded systems. It is therefore the ontology for automotive electronics and software, making system models unambiguous, consistent and exchangeable.

The ATESST project team has a strong background in the automotive domain, and both industrial and academic partners have, in developing the ADL, drawn on their extensive experience with automotive systems. As a consequence, the developed language – while state of the art – can be flexibly introduced and adapted in practical applications; it is specific to the automotive domain, 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 ATESST project is developing not only the language and a backbone for the accompanying set of practical methods, but also a set of Eclipse-based plug-in tools and a comprehensive, realistic example of the language's use, demonstrating its suitability for down-to-earth industrial scenarios.

EAST-ADL2 aims to set a new standard in the automotive domain. AUTOSAR, the international, standardized software architecture and exchange format for automotive components, could easily be extended by EAST-ADL2 to cover more abstract modeling levels.

In short, with EAST-ADL2 the ATESST team has provided a highly relevant model-based development technique for those who are seeking to improve industrial automotive system development.

Daniel Svensson (Volvo Cars) and Stefan Voget (Continental)

Table of Contents

3 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.

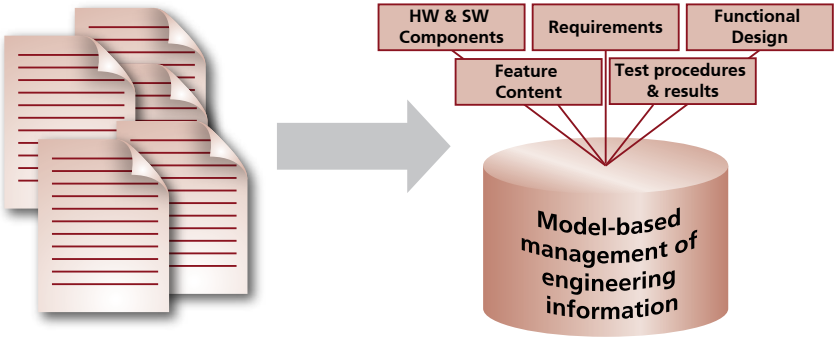

Figure 1. Embedded systems are highly complex - the extensive engineering information needs to be managed adequately.

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-ADL2, the ATESST research project aims to provide a basis for such a holistic approach.

4 Introduction

EAST-ADL2 is an Architecture Description Language (ADL) initially defined in the EAST-EEA project (*http://www.east-eea.net/*). It is being refined and aligned with the more recent AUTOSAR automotive standard in the ATESST project. EAST-ADL2 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-ADL2 contains several abstraction levels (see *Figure 2*). The software- and electronics-based features of the vehicle are described at different levels of abstraction.

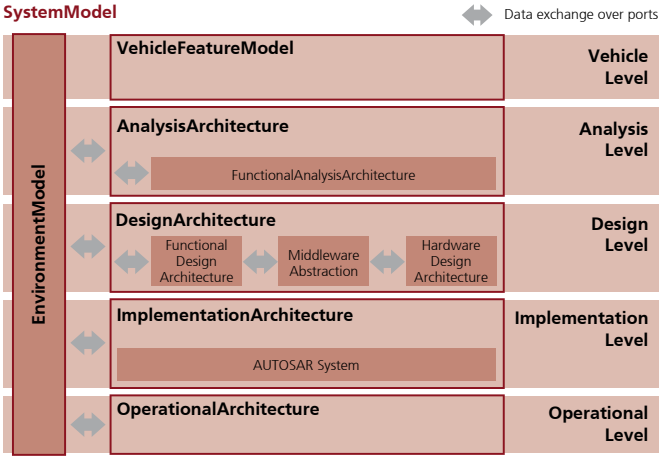

Figure 2. EAST-ADL2 abstraction levels and model organization.

The proposed abstraction levels and the contained elements provide 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.

The features in the Vehicle Feature Model (VFM) at the vehicle level represent the

content and properties of the vehicle from an external 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 on an abstract level 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 their 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-ADL2 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 entire functions 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-ADL2 concepts, we go on to discuss the motivation and modeling concepts in more detail.

5 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 electronics as an aircraft did two decades ago. To satisfy customer demands and competitiveness between OEMs, innovation will drive the increase in electronics parts to 40% of a vehicle's value by 2015 *(see Figure 3).*

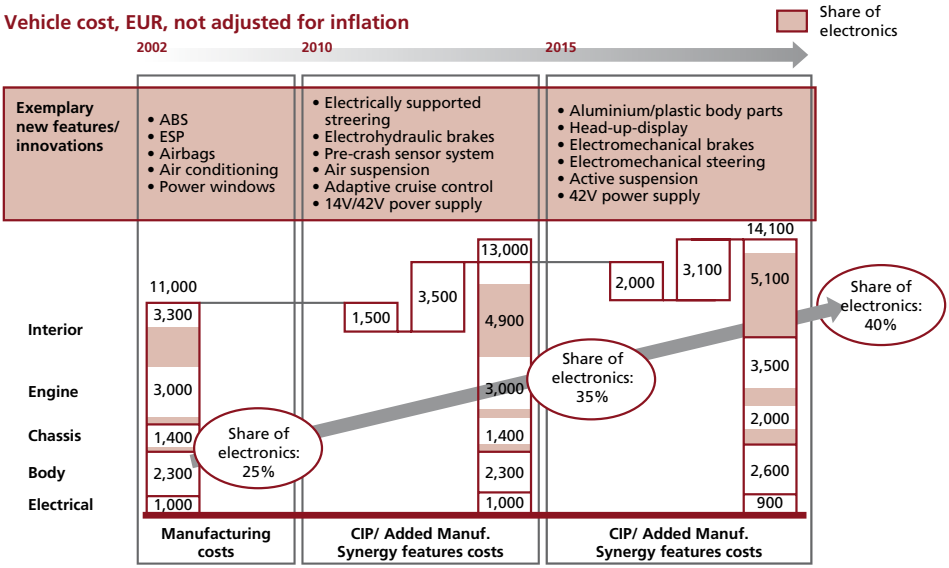

Figure 3. Share of electronics in product costs, source: "Auto Catalog, Pass 2000, Mc Kinsey".

It is obvious, that the complexity of vehicle's electronic architecture will continue to grow, bearing in mind that failure rates, at a high level today, have to be controlled and reduced.

To improve dependability and manage the complexity of automotive products while keeping track of development costs, 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.

A standardized software architecture and methodology is a first step toward 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. Many stakeholders are involved here, and development is distributed over several OEM 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 improving 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 WD26262) must be prepared by introducing new techniques and a structured development approach.

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.

6 EAST-ADL2 Modeling Approach

This is an outline of the modeling approach used in the ATESST project to define a domain model and a profile for EAST-ADL2, also showing how this is used by a modeler in architecture-driven development.

6.1 Overview of the Profile Approach

The meta-model and profile of EAST-ADL2 are defined in two steps. A domain model is defined, capturing the domain-specific modeling needs. This captures the concepts and their relations defined in the language. The EAST-ADL2 meta-model is defined using the Object Management Group's Meta Object Facility. This is equivalent to the basic concepts of UML2 (www.uml.org), such as classes, compositions and associations.

To provide EAST-ADL2 to a wider community, a UML2 profile is defined. The profile allows users to do system modeling according to the EAST-ADL2 semantics using off-the-shelf UML2 tools. Generally, two approaches are possible with the tool: either the user defines a UML2 user model and then applies "stereotypes" from the EAST-ADL2 profile to the elements, or the profile is used as a guide to which domain elements are available for modeling using a toolbar for example. Predefined domain-language attributes are thus available for each stereotyped element. Constraints are also a 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 domain model and profile are documented in the EAST-ADL2 specification and the profile is also delivered as an XMI file ready for use in UML2 tools.

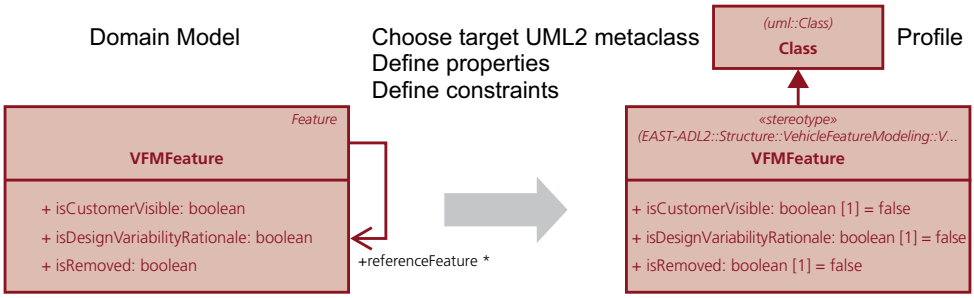

Figure 4. The domain model specifies the attributes for the profile stereotype.

In the definition of the EAST-ADL2 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 4*).

6.2 Meta-modeling and Instances

The elements defined by the user on the M1 level are instances of the elements defined at the M2 level, which include stereotypes that provide templates for domain-specific semantics and notation.

The user model is defined as a combination of UML2 elements and EAST-ADL2 stereotypes, which means that the model is close to the intention of UML2 and that views and features of 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 (see *Figure 5*).

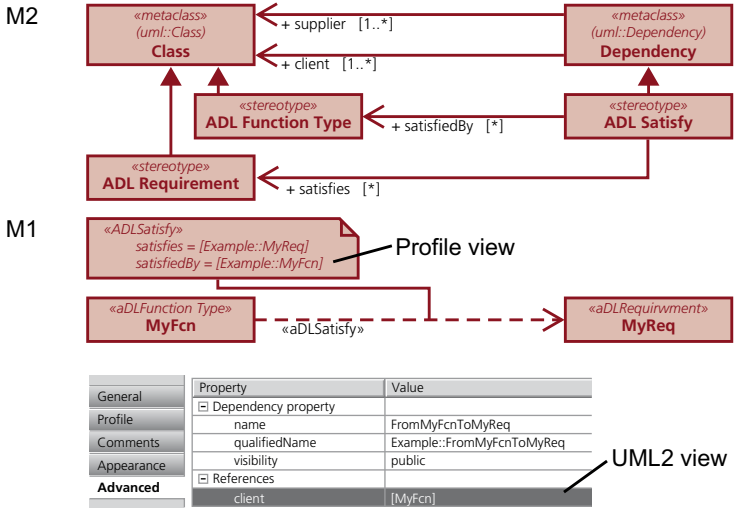

Figure 5. The UML2 meta-levels (M1 and M2) and EAST-ADL2.

7 EAST-ADL2 Modeling Concepts

In this section, the EAST-ADL2 modeling concepts are described in more detail for five areas: functional abstraction (Section *7.1*), timing modeling (Section *7.2*), requirements modeling (Section *7.3*) functional safety modeling (Section *7.4*), and variability modeling (Section *7.5*).

7.1 Functional Abstraction

EAST-ADL2 provides the means to capture the functional decomposition and behavior of the embedded system and the environment.

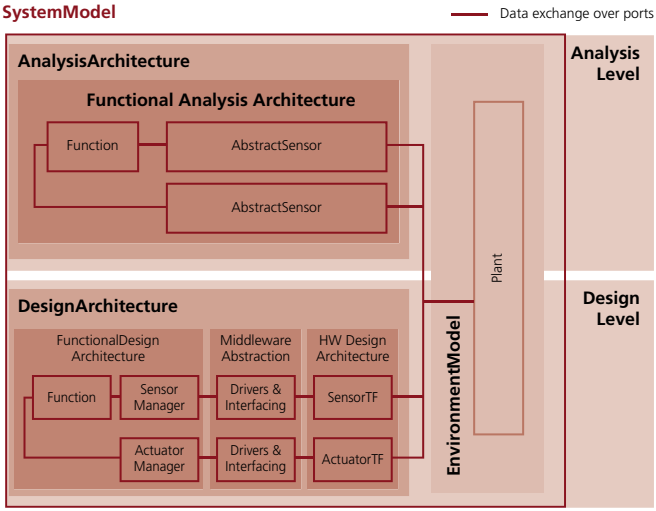

Figure 6. The "AnalysisArchitecture" and the "DesignArchitecture" and connections to the "EnvironmentModel".

At the analysis level, the "AnalysisArchitecture" contains "ADLFunctions" 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 6* explains the entities involved and shows how they are connected.

The "ADLFunctions" 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.

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

The design level (see *Figure 6*) contains a more detailed functional definition of the system. "ADLFunctions" and "LocalDeviceManagers" represent application software in the Functional Design Architecture. "ADLFunctions" in the Middleware Abstraction are used to capture middleware behavior affecting application functionality. The Hardware Design Architecture serves two purposes: sensors, actuators and I/O represent the interface to the environment. ECUs and busses represent the computation platform to which application functions are allocated. The objective is to describe a representation of the vehicle hardware architecture from an abstract perspective, and to serve as an allocation target for functions. The hardware entities, in particular sensors and actuators, may also have a behavioral definition to allow end-to-end analysis of the functions.

7.2 Timing Modeling

EAST-ADL2 provides support to 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.

Figure 7 gives an overview of how timing information is seen in the EAST-ADL2 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.

EAST-ADL2 currently focuses on modeling of timing requirements on the functional abstraction levels of the architecture description language. The implementation level, i.e. AUTOSAR, is currently not explicitly considered, but it is expected that the information can be modeled in a similar way. The same holds for timing properties on both the functional abstraction levels and the implementation level.

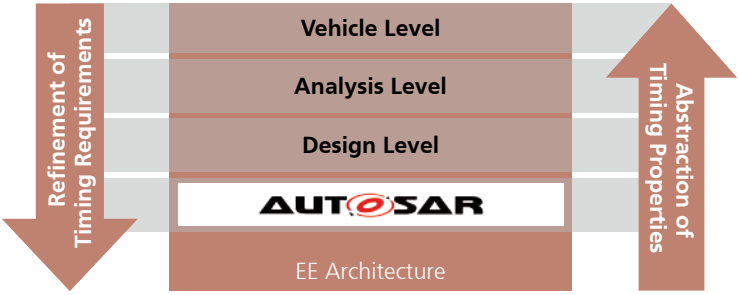

Figure 7: Timing information as seen in the EAST-ADL2 system model.

Timing information on the functional abstraction levels is perceived as follows:

Timing requirements for a function can be captured on logical abstraction levels where no concrete hardware is yet available. This allows the specification of general timing requirements such as end-to-end delays from sensors to actuators regardless of how the final solution is built. This reflects the notion that a purely logical functional specification is not concerned with its technical realization, i.e. how many ECUs or bus systems are ultimately involved. What matters from the functional perspective are the recurring end-to-end delays of a control application, which need to keep pace with the real plant. Specifying timing requirements on the implementation level might be both too late in the development process and rather difficult because of language complexity (e.g. AUTOSAR) and the number of details on this level.

Timing properties are characteristics of a solution, e.g. actual response times, and should be reflected in the functional abstraction levels.

In EAST-ADL2, timing requirements are divided into various kinds of delays (or latencies) for single time-consuming modeling entities as well as specific requirements for temporal synchronization of input or output data. The delays are either end-to-end delays, which are subject to segmentation along the functional decomposition track (e.g. end-to-end delay for a top-level function), or the delays form part of an end-to-end timing chain, and thus constitute segments of such an end-to-end timing chain. Furthermore, delays can be sub-classified as being induced either by data transformation performed by an "ADLFunction" or data transmission via an "ADLConnector".

7.3 Requirements Modeling

In order to comprehensively support the development of complex automotive systems, EAST-ADL2 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-ADL2 is to provide means for project-specific adjustments to requirements specification structures, which are inspired by the Requirements Interchange Format (RIF, http://www.automotive-his.de/rif/doku.php?id=welcomeeng).

EAST-ADL2 does not start from scratch but closely aligns its requirements concepts with SysML 1.1. However, extensions and adjustments are made to these proposals based on the needs of the automotive application domain.

Methodically, EAST-ADL2 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. "ABS shall have an MTTF of 10,000 hours").

EAST-ADL2 offers detailed means to explicitly model central 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.

7.4 Functional Safety Modeling

The overall objective of the support for functional safety and requirements modeling is to enforce explicit considerations of requirements and safety concerns throughout an architecture design process and also to enable early quality investigation and feedback.

As an overall system property, safety is concerned with abnormalities (e.g. faults, errors and failures) and their consequences under certain given environmental conditions. It is one particular aspect of system dependability that normally also encompasses reliability, availability and security. "Functional safety represents the part of safety that depends on the

correctness of a system operating in its context"[1]. In other words, it addresses the hazardous abnormalities of a system during its operation (e.g. component errors and their propagations).

EAST-ADL2 facilitates safety engineering in terms of safety analysis, specification of safety requirements, and safety design. While supporting the safety design through its intrinsic architecture modeling and traceability support, EAST-ADL2 allows the integration of safety analysis and safety requirements by explicitly capturing the abnormalities and hazards of concern.

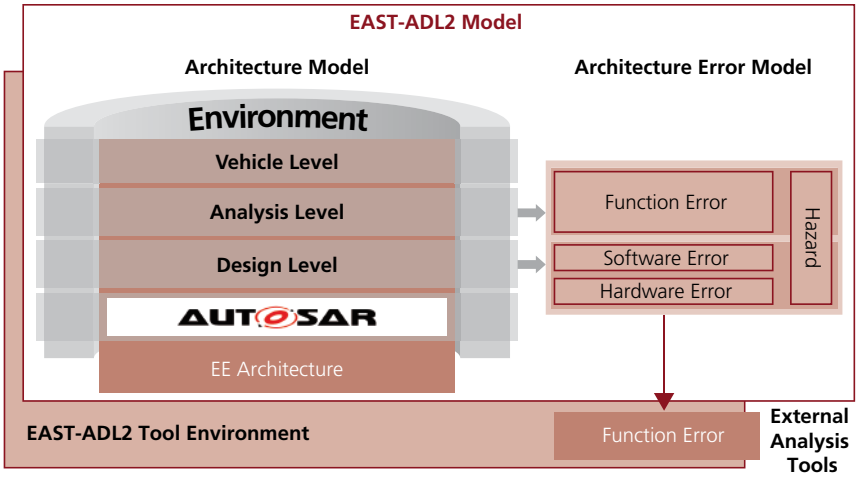

Figure 8: EAST-ADL2 error model as a separate architecture view extending the normal architecture model and providing analysis leverage through external tools.

To this end, the EAST-ADL2 error modeling package extends a nominal architecture model, typically at the analysis and design levels, by information on failure semantics and error propagations. The failure semantics can be provided in terms of logical or temporal expressions, depending on the analysis techniques and tools of interest. Such analytical information, together with environmental conditions, forms the basis for identifying the likely hazards, reasoning about their causes and consequences, and using them to derive safety requirements. The relationships of local error behaviors are captured by means

[1] *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 0: Functional safety and IEC 61508. International Electrotechnical Commission © 2005, IEC, Geneva, Switzerland.*

of explicit error propagation ports and connections. Owing to these artifacts, EAST-ADL2 allows advanced error propagation properties, such as the logical and temporal relationships of source and target errors, the conditions of propagations, and the synchronizations of propagation paths. Hazards or hazardous events are characterized by attributes for severity, exposure and controllability. A hazardous event may be further detailed, e.g. by use cases, sequence or activity diagrams. 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-ADL2 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 external 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[2]) 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-ADL2, a safety requirement derived from the safety analysis has attributes specifying the hazard to be mitigated, the safety integrity level (ASIL/SIL), 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.

7.5 Variability Modeling

In order to give an overview of variability management in EAST-ADL2, we examine two questions:

1. In what development situations and contexts is variability management needed? Or: For what parts of the EAST-ADL2 is variability management support provided?

[2] *Papadopoulos, Y., McDermid, J.A.: Hierarchically Performed Hazard Origin and Propagation Studies. SAFECOMP (1999) pp. 139-152.*

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 Feature Level, where model range features and variability is viewed. 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.

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-ADL2, 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, product 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-ADL2, they are also used on analysis and design levels and acquire a much more concrete meaning there. Product decision modeling, on the other hand, is aimed at defining configuration. The configuration of a feature model f_a – i.e. the selection and de-selection of its features – is defined in terms of the configuration of another feature

model f_b. A product decision model can thus be seen as a link from f_b to f_a that allows us to derive a configuration of f_a from a given configuration of f_b. 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 VFM. This means that the main driver for variability and also variability instantiation is the Vehicle Feature Model. Variability on the artifact level essentially consists of the use of variation points and simple feature models to expose the variability of functions.

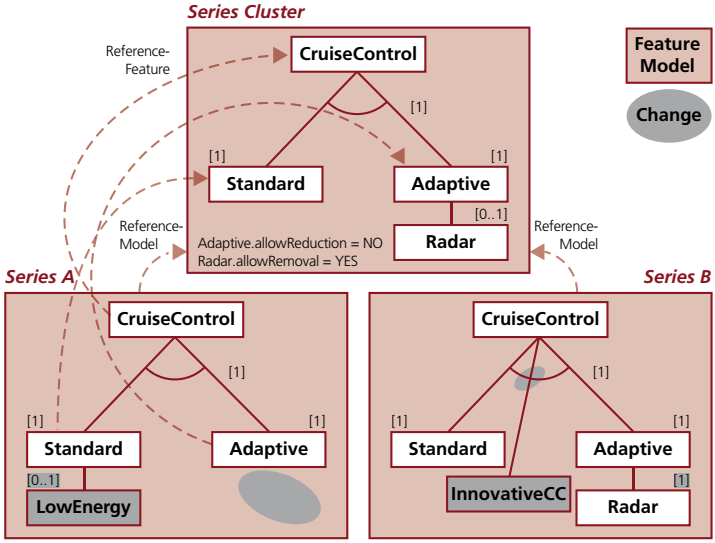

Figure 9. Reference model and referring models in the multi-level feature modeling approach.

8 Related Approaches

One key aspect of the refinement of EAST-ADL1 to EAST-ADL2 is to benefit from existing approaches and languages. Whenever possible, concepts 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.

The AUTOSAR platform is a future de-facto standard for automotive embedded systems. It defines a set of middleware components that provides a standardized platform for application software and also means to model platform and application software. In EAST-ADL2, the entities corresponding to software and hardware components are taken from the AUTOSAR standard but put in a context where system modeling concepts can be used. AUTOSAR-compliant software architecture can thus be modeled with support for e.g. variability, requirements, traceability and verification and validation. In this sense, the EAST-ADL2 language complements the AUTOSAR initiative by providing higher-level abstraction and analysis.

EAST-ADL2 is constructed in line with UML2 because EAST-ADL2 is designed as a UML2 profile, i.e. a specialization of the UML2 language for a specific application domain. This can be viewed as an implementation of EAST-ADL2 in UML2, thus allowing for the use of various available UML2 modeling tools. SysML concepts are reused to manage requirements and plant modeling constructs. MARTE is a recently adopted UML2 profile for "Modeling and Analysis of Real-Time and Embedded Systems". It fills several gaps of UML2 for modeling both hardware and software aspects of RTES and performing quantitative analysis of their characteristics. The MARTE timing constructs are reused in EAST-ADL2. Further harmonization is carried on with a view to releasing the EAST-ADL2 profile as an annex to the subsequent version of MARTE. Some of the ATESST partners are in charge of MARTE finalization.

Another standardization effort has also been taken into consideration: the SAE "Architecture and Analysis Description Language" (AADL). Compared to EAST-ADL2, 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 run-time systems. For the automotive domain, the clash with AUTOSAR concepts is also a problem. However, wherever applicable, AADL concepts were reused, e.g. for error behavior modeling.

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 interconnected 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. Based on initial work done in the EAST-EEA project, the EAST-ADL1 language is being refined by taking particular care to reuse standardized language constructs and to align EAST-ADL2 with other existing related standardization initiatives.

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 ATESST project, an open source UML2 modeling and profiling environment based on Eclipse, called Papyrus, has been developed. A number of Eclipse Plug-ins have also been developed. Dedicated enhanced support of the language was developed for the Papyrus UML modeling tool, which is freely available (http://www.papyrusuml.org). The tool was customized to facilitate its use for UML2 and to show the capabilities offered by the use of EAST-ADL2 concepts, e.g. support for feature and variability modeling or initial support for import and export to the Simulink platform.

The main achievement of the ATESST project is the standardization of the architectural description language EAST-ADL2. If EAST-ADL2 becomes a de-facto standard for the automotive industry, complementing the AUTOSAR standard, it will introduce a common terminology and serve as a basis for improved tool interaction and information exchange between/within companies. Being a UML2 solution, it allows open usage and standardization of results.

10 Selected ATESST Publications

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

F. Törner, D. Chen, R. Johansson, H. Lönn, M. Törngren: "Supporting an Automotive Safety Case Through Systematic Model-based Development - the EAST-ADL2 Approach", *Proceedings of the SAE World Congress*, April 2008.

C.-J. Sjöstedt, J. Shi, M. Törngren, D. Servat, D. Chen, V. Ahlsten, H. Lönn: "Mapping Simulink to UML in the Design of Embedded Systems: Investigating Scenarios and Structural and Behavioral Mapping", Invited paper. To be published in *OMER 4 Post Workshop Proceedings*, April 2008.

P. Cuenot, D. Chen, S. Gérard, H. Lönn, M.-O. Reiser, D. Servat, C.-J. Sjöstedt, R. Tavakoli Kolagari, M. Törngren, M. Weber: "Managing Complexity of Automotive Electronics Using the EAST-ADL", *Proceedings of the 2nd International UML&AADL Workshop* (UML&AADL'2007) at the 12th International Conference on Engineering of Complex Computer Systems (ICECCS), IEEE Computer Society, September 2007. pp. 353-358.

P. Cuenot, D. Chen, S. Gérard, H. Lönn, M.-O. Reiser, D. Servat, R. Tavakoli Kolagari, M. Törngren, M. Weber: "Towards Improving Dependability of Automotive Systems by Using the EAST-ADL Architecture Description Language", Book Chapter in *Architecting Dependable Systems IV*, Lecture Notes in Computer Science, edited by R. de Lemos, C. Gacek; A. Romanovsky, Springer, August 2007. pp. 39-65.

C.-J. Sjöstedt, D. Chen, P. Cuenot, P. Frey, R. Johansson, H. Lönn, D. Servat, M. Törngren: "Developing Dependable Automotive Embedded Systems using the EAST-ADL: representing continuous time systems in SysML", *Proceedings of the 1st International Workshop on Equation-Based Object-Oriented Languages and Tools* (EOOLT2007), Forschungsberichte der Falkutät IV - Elektrotechnik und Informatik, Bericht Nr. 2007-11, July 2007. pp. 25-36.

D. Chen, M. Törngren, J. Shin, H. Lönn, S. Gérard, M. Strömberg, K.-E. Årzén: "Model Integration in the development of Embedded Control Systems: a characterization of current research efforts", *Proceedings of the IEEE International Symposium on Computer-Aided Control Systems Design*, IEEE Computer Society, October 2006.

11 Contact and Further Information

The partners engaged in the ATESST 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		
• Daimler	Jörg Donandt	joerg.donandt@daimler.com
• Carmeq	Matthias Weber	matthias.weber@carmeq.com
• Volvo Cars	Hans Alminger	halminge@volvocars.com
• Volvo Technology (coordinator)	Henrik Lönn	henrik.lonn@volvo.com
Automotive Suppliers		
• Continental	Philippe Cuenot	philippe.cuenot@continental-corporation.com
• Mecel	Pontus Jansson	pontus.jansson@mecel.se
Tool Vendors		
• ETAS	Patrick Frey	patrick.frey@etas.de
• 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	Ramin Tavakoli Kolagari	tavakoli@cs.tu-berlin.de

ATESST

ATESST project website: http://www.atesst.org

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

DAIMLER

Continental

VOLVO

Mecel

Mentor Graphics

ETAS

carmeq
software & systems

cea list

VOLVO

berlin

KTH
VETENSKAP
OCH KONST
ROYAL INSTITUTE
OF TECHNOLOGY