Your digital version of the ATESST2 newsletter #2 2010

Click here to **download** the tool CVM, a framework for compositional variability management aligned with the EAST-ADL2.

Click here to **download** the UML Modeling tool Papyrus and the EAST-ADL2 profile

Dear Colleague,

in this ATESST2 newsletter, we will present the management of variability in EAST-ADL2 and give a short overview of its connection to the AUTOSAR variability management.

Further information about the ATESST2 project and the EAST-ADL2 language can be found at http://www.atesst.org/.

In This Issue

Final Open Workshop of the ATESST2 Project, 21 June 2010, Frankfurt

Please note in your calendar the date and place of the Final Open Workshop of the ATESST2 project, which is scheduled for

21 June 2010 in Frankfurt.

We will present the major results of the project and have a tools and demonstrator session. Participation to the workshop is free of charge and open to everyone interested.

Future newsletters will give you more detailed information about the concrete schedule and agenda.

Webinar: EAST-ADL2 Variability Management Concept Presentation on 07 April 2010, 13.00 – 14.00

You are invited to take part in a web-based overview presentation of the EAST-ADL2 variability management on Wednesday, 07 April 2010, 13.00 – 14.00 (Central European Time, CET). Below is the meeting link, please connect at 12:30 to test sound and picture.

Join the meeting.

Audio Information

Computer Audio

To use computer audio, you need speakers and a microphone, or a headset.

First-Time Users

Make sure the Office Live Meeting client is installed before the meeting:

Download the Microsoft Office Live Meeting client

Troubleshooting

Unable to join the meeting? Launch the Office Live Meeting client and join the meeting with the following information:

Meeting ID: 8534a19639c6454c96b6f6894a1c6665

Location:
meet:sip:ramin.tavakoli@volvo.com;gruu;opaque=app:conf:focus:id:8534a19639c6454c96b6f6894a1c6665

If you still cannot enter the meeting, contact support.

Notice

Office Live Meeting can be used to record meetings. By participating in this meeting, you agree that your communications may be monitored or recorded at any time during the meeting.

EAST-ADL2 Spotlight

ATESST2 defines EAST-ADL2 as a domain-specific language using meta-modeling constructs such as classes, attributes, and relationships.

The project also implements a UML2 profile which is used in UML2 tools for user modeling.

The EAST-ADL2 definition also serves as the specification for implementation in domain-specific tools.

EAST-ADL2

Variability Across Abstraction Layers

In order to understand variability management in EAST-ADL2 it is essential to distinguish two levels of concern – the Vehicle Level and the Artifact Level – and to understand the purpose and characteristics of variability management on each of these two levels.

Variability management begins on the Vehicle Level, where model range features and variability are defined. At this point, the purpose of variability management is to provide a highly abstract overview of the variability in 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, it is not specified 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. The language entity for representing such an abstract variability is called “feature”. In the example, we might thus represent the variability of the wiper by introducing an optional feature named “RainControlledWiping”. This is subsequently refined during analysis and design.

Usually this Vehicle Level variability is highly technical and contains many details not intended for end-customers. Therefore, it is preferable to have a separate view on this technically oriented variability for end-customer configuration, in which the technical details of variability can be presented differently, partly hidden and diversely packaged, for example according to marketing considerations. Therefore, at least two distinct viewpoints will often be distinguished on Vehicle Level: (1) end-customer variability and (2) technical variability.

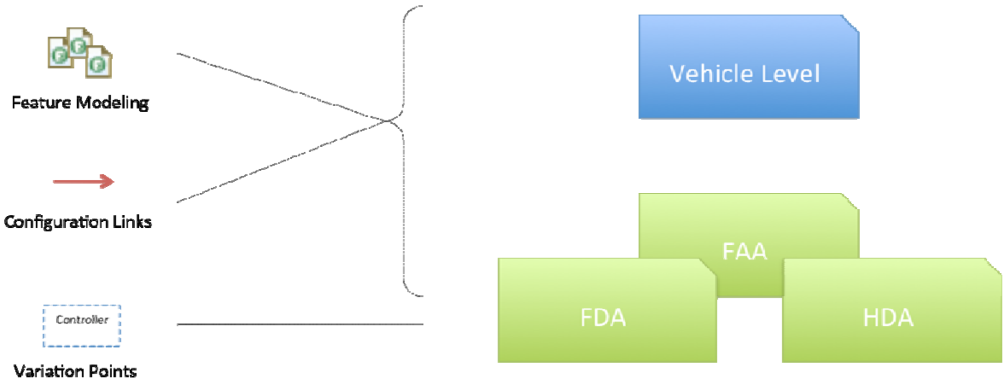

Figure 1. Variability Across Abstraction Layers

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 (cf. Figure 1). In fact, a specific variability may lead to modifications in any development artifact, such as requirements specifications and functional models. Here, describing that a specific variability occurs is not sufficient; it is necessary to describe how each variability affects and modifies the corresponding artifact, for example the internal structure of a FunctionType.

Concepts for Variability Modeling in EAST-ADL2

Variability modeling in EAST-ADL2 is based on two major techniques: feature modeling and configuration decision modeling.

The purpose of feature modeling is to define the commonalities and variabilities of the product variants within the scope of a product line. Their main purpose is to provide a basis for configuration: for every feature model a feature configuration can be defined, i.e., a selection and deselection of its features. This configuration can be complete, if the decision of selection or deselection was taken for all features, or partial. In case of a partial decision, not all decisions have been taken, yet. This way, every feature model defines a set of valid feature configurations, also called a configuration space.

Feature models are normally used on a high level of abstraction, as described above for Vehicle Level variability. In EAST-ADL2, they are also applied 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, i.e., the binding of variability. The configuration of a feature model F_t 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. Therefore, F_s is called the source feature model and F_t is called the target feature model.

In addition to these two major techniques, Artifact Level variability heavily relies on the use of another concept called variable elements, i.e., elements within the artifacts that can be marked optional and are therefore not present in all system variants. For example, FunctionPrototypes can be marked optional in this way, which means that the containing FunctionType does not contain this prototype in all system variants. EAST-ADL2 provides rules to trace the effect of such an optionality on the surrounding function architecture, based on the notion of implicit optionality. For example, connectors that go to and from an optional FunctionPrototype are marked as implicit optional.

While feature modeling is applicable to all areas of EAST-ADL2, in which variability management takes place, the other two modeling means are only valid to some areas.

Putting it all Together

Eventually, the goal is to be able to relate the variability representations on different levels of abstraction and composition such that a configuration of the complete system can be derived from a valid configuration of the root feature model, e.g., a customer feature model.

Click here to **download** a presentation with an overview of EAST-ADL2 variability management

To achieve this, FunctionTypes can be provided with a feature model that exposes the variability within this function. Such a feature model is called the public feature model of the FunctionType and can be seen as an extension to this component's interface dedicated to variability. Finally, a FunctionType can be supplied with its own configuration decision model, called internal binding, which defines how to configure contained lower level functions, depending on the configuration of the container function's public feature model.

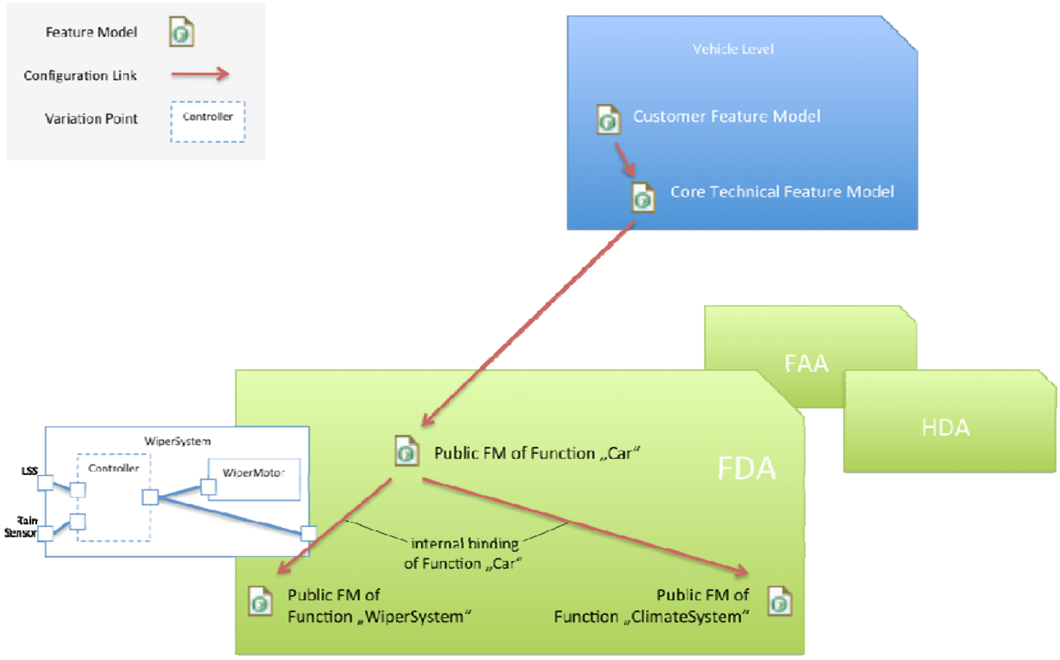

Figure 2. Configuration of feature models on different abstraction levels.

Together with configuration decision models on the Vehicle Level and another from Vehicle to Artifact Level, it is possible to link the low-level, strongly technical variations to the high-level variability representations on Vehicle Level (cf. Figure 2).

AUTOSAR Spotlight

AUTOSAR defines a software architecture platform by standardization of its infra-structure and a communication layer suitable for distributed systems.

The standard also defines description means for the execution platform including control units, network topology, I/O, and middleware and application software components.

The platform and the description means make it possible to integrate software from different suppliers on the same hardware. Reuse is favored and dependencies between application software and hardware are avoided.

Variability Management With AUTOSAR Version 4.0

AUTOSAR version 4.0 has introduced an approach for modeling variability of the software architecture on Implementation Level. The idea of this approach mainly consists of the application of four variability modeling patterns in the meta-model (metalevel 2, M2) for

- aggregation,
- association,
- attribute value, and
- classes providing property sets.

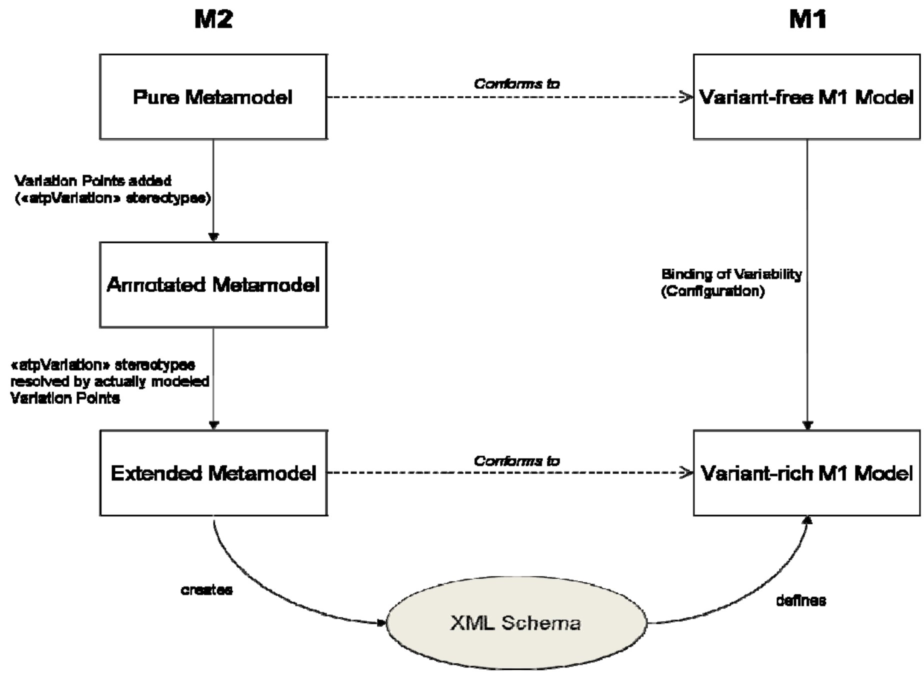

Figure 3. Variability Management With AUTOSAR Version 4.0.

AUTOSAR starts with the Pure Metamodel (cf. Figure 3).

1. The Annotated Metamodel has the same structure, but marks those locations that are variation points with a stereotype (<<atpVariation>>). AUTOSAR has defined patterns that show how to transform these locations markings into a structure that has all the information that is necessary to implement variation points.

2. An XML schema is created corresponding to the annotated metamodel. User models (metalevel 1, M1) can now be created, conforming to this schema. This is where the variability is bound (configured) and specific variants are developed.

3. (Variant-rich M1 Model) A variant-rich M1 model (which must conform to the rules defined by the Extended Metamodel) is transformed into one particular variant by binding the variation points (the variability).

4. (Variant-free M1 Model) The completely bound M1 model must adhere to the pure meta model with respect to consistency rules and semantic constraints defined in the related template specifications. Especially, the multiplicities in the bound model must conform to the multiplicities and the constraints of the pure meta model. For example, in an invariant model, ports may have either 1:n or n:1 connections. On the other hand, this rule does not apply to a variant-rich model because the variants might overlap one another, which results in m:n connections. After binding the variant-rich model and extracting one particular variant, the rules of an invariant model apply again. As another example, the interface compatibility rules can only be applied to a particular variant. Note, that the existence of PostBuild variation points implies in practice that the completely bound M1 model does not necessarily exist as an artifact. The reason is that the PostBuild variation points are bound at startup of the ECU and obviously cannot be resolved in the M1 model. However the resulting M0 model conforms to the bound M1 model.

The AUTOSAR way of modeling variability can be compared to the EAST-ADL2 artifact variability modeling (not including the feature models). All AUTOSAR variability can be expressed by EAST-ADL2 artifact variability modeling, by transforming all variable elements that belong together to an EAST-ADL2 VariationGroup.

EUCAR
EUROPEAN COUNCIL FOR AUTOMOTIVE R&D

SEVENTH FRAMEWORK PROGRAMME

The ATESST2 consortium