How the concepts of the Automotive standard "AUTOSAR" are realized in new seamless tool-chains

Stefan Voget, P Favrais

How the concepts of the Automotive standard "AUTOSAR" are realized in new seamless tool-chains

Dr. Stefan Voget[1], P. Favrais[2]

1: Continental Engineering Services GmbH, Siemensstraße 12, 93055 Regensburg
2: Continental Automotive France SAS, Avenue Paul Ourliac B.P. 83649, 31036 Toulouse

ABSTRACT: In this paper we present concepts leading to a new seamless tool-chain in the automotive industry. The actual tool market is mainly influenced by the AUTOSAR standard and an open user group approach that published and maintains an AUTOSAR Tool platform (ARTOP).

Keywords: AUTOSAR, AUTOSAR methodology, meta-model, seamless tool-chain, ARTOP, EAST/ADL, timing modeling

1. Introduction

Since six years the AUTOSAR standard [1] is under development within the automotive industry. No sub-domain of the vehicle is excluded. Today, series developments based on this standard run in interior, power-train as well as in chassis. Although their requirements are quite different, AUTOSAR allows reuse of infrastructure software between the sub-domains.

In none of the projects the development starts at zero. Existing solutions have to be adapted to the AUTOSAR approach. The reality shows, that there is not the one and only path to migrate existing solutions to AUTOSAR. But, several "best practices" can already be derived by the experiences done in all automotive domains as the standard is in series development since more than three years.
AUTOSAR does not only define new infrastructure software, but also the processes and tool-chains are influenced by the standard.

In this article we will motivate the importance of AUTOSAR for the tool chains in series development projects. This is due to the AUTOSAR methodology [2][3] which shifts more and more complexity from an implementation oriented process to a configuration based process. This results in a shift of main tasks in a development process more to the left part of the V-cycle.
As a consequence, the configuration based process enables the introduction and use of more automatism in the software development. The new character of the AUTOSAR methodology enables and requires new tools that provide the automatisms. Therefore, AUTOSAR currently has a deep influence on the tool developments for the automotive industry. More and more tool vendors arise in the market with tools that support the AUTOSAR methodology.

Mainly two alternative approaches for the development of these tools can be observed. Some vendors enhance their existing tools with export and import capabilities to and from the AUTOSAR format but do not adapt the workflow regarding the configuration process. This is mainly reasoned in the fact that most of the series projects are still non-AUTOSAR projects. The comment given above regarding migration to AUTOSAR is also valid for the tool industry. Other vendors use the standard as an opportunity to design new tool capabilities. This enables new vendors to establish them self in the automotive industry.

A lot of new tool innovations that realize the concepts around the AUTOSAR system methodology are on the way. One platform for these innovations is ARTOP (**A**UTOSA**R** **TO**ol **P**latform) [4][5][7][9]. The so called ARTOP user group provides common base functionality used for design and configuration of AUTOSAR compliant systems and electronic control units (ECU). It is an Eclipse-based AUTOSAR tool infrastructure platform (See www.artop.org) [6].

Based on this platform we will show how the AUTOSAR concepts like system development, system configuration, timing analysis and code generation can be brought together into a seamless tool-chain. Some outlook about extensions into functional modelling – that is not defined by the AUTOSAR methodology itself – is presented, too.

2. AUTOSAR

The main concept of the AUTOSAR approach is to separate application and infrastructure software. On application level the AUTOSAR software components (SW-C) are introduced. A SW-C may cover a small, reusable but also a complex automotive functionality.

Software components are connected with each other through well defined ports and interfaces. In terms of implementation, the AUTOSAR software component

is independent from the infrastructure, i.e. it is independent from the type of the microcontroller and the type of electronic control units (ECU) the SW-C is running on.

The separation between function and infrastructure is important for the reusability of software components in different ECU's and is achieved at design time through the Virtual Functional Bus (VFB). The Virtual Functional Bus takes care of the communication between different components and between software components and the hardware. The VFB realizes the goal of being able to relocate software components and allows a virtual integration of AUTOSAR software components in an early development phase. AUTOSAR provides a methodology and proposes the use of tools for this purpose.

2.1 AUTOSAR Methodology

The AUTOSAR Methodology is a booklet to support the exchange of model data in early development steps of a series project. It defines activities and work products. But, it is neither a process description nor a business model and therefore does not predefine a strict order in which the activities should be accomplished.

In the first step, system information is collected that will be used for a configuration of the system. These inputs are formal descriptions of software components, ECU hardware resources and system topology.

The first important step, the system configuration, takes these descriptions and performs a mapping of the software components to one or more ECU's. With the help of the VFB principle the application software had been modelled independent from the concrete hardware until this step. The system configuration step marks the changeover from a hardware independent SW development to a hardware related SW configuration.
To complete this step, a second mapping is performed. It is the mapping of signals to bus frames which results in the communication matrix.

If the system configuration step is finalized, one knows the software components that are allocated to a single ECU. Now, the further steps of the methodology operate on a single ECU. To be able to concentrate on a single ECU, the subset of information that is relevant for the further implementation of that ECU is extracted from the system configuration description.

Additionally, necessary information for the implementation is added in the ECU configuration activity. The output of this step is the ECU Configuration Description containing all information concerning one electronic control unit. Using this description executable software for this ECU can be generated.

This step includes the generation of code, compiling code and connecting everything into an executable.
The description of an AUTOSAR system is performed using appropriate templates. Those templates define the AUTOSAR meta-model and allow a formal description of an AUTOSAR system. Based on these descriptions the AUTOSAR Methodology generates an ECU executable.

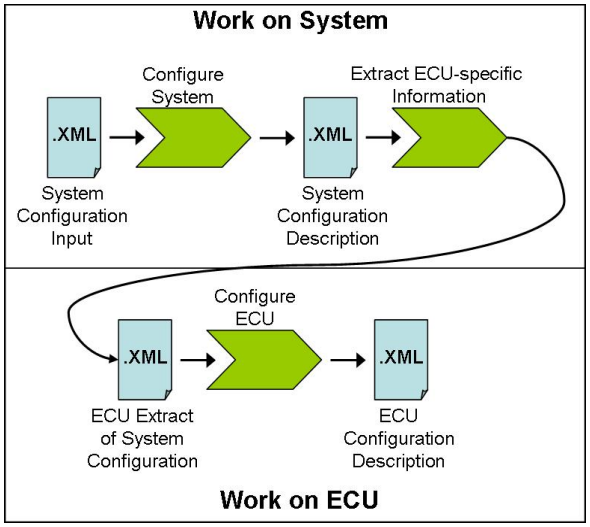

Figure 1 AUTOSAR methodology

2.2. Challenges in AUTOSAR Methodology

SW architectures with several thousands of networked software components lead to a high complexity of today's distributed embedded systems in cars. To improve the distributed development, AUTOSAR reduces the usage of different data formats. Now, models are exchanged between the development partners based on a standardized exchange format.

But, the AUTOSAR meta-model is very complex. It is a new terminology with complex relationships between the defined objects. So far, only limited knowledge about how to realize the methodology in series projects can be found in the industry.

The models will be exchanged and reused between suppliers and OEMs during the life cycle of a software development. However, the AUTOSAR methodology does not define who is doing what and when in a software development. Procedures and the sharing of the different roles and tasks have to

be identified and contracted between the OEM and their suppliers.

The standard itself does not give support in the modelling process and just partial support in the data consistency. Concurrent configuration / modelling and parallel development are not considered by AUTOSAR but are a must for the usability of a tool-chain.
Furthermore, until now several AUTOSAR versions are in use, that are incompatible with each other.
The templates defined by AUTOSAR improve the interoperability of tools but do not completely solve the problem of usage of too many different tools. As each step of the methodology leads to a result, which is saved in an exchange format, the change of a tool can lead to loss of information and to errors.
Additionally to the AUTOSAR data, very often the tools need to store presentation data for the tool itself. As this is not foreseen in AUTOSAR, exporters, importers and translators are still needed.

2.3 AUTOSAR Use Case: exchange of models

In this section we want to consider an example for how an OEM and a supplier may collaborate during the development of a network of ECUs. It is assumed that the supplier develops several ECUs that run within the network of ECUs in a vehicle. The OEM is the integrator for the whole system. The steps are presented in the form of a use case description.

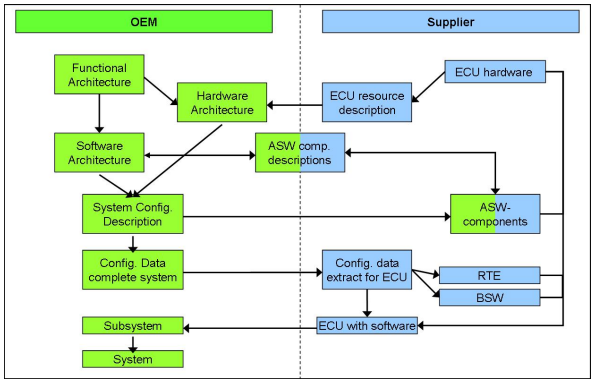

Figure 2: Exchange of models between OEM and supplier

Title of Use Case:
A subsystem, consisting of several ECUs (integrated hardware and software), is sold by one supplier to an OEM. The OEM integrates the subsystem into its vehicle network of ECUs. The remaining ECUs may come from different suppliers.

Pre-condition:
A subsystem, consisting of several ECUs equipped with AUTOSAR software, is developed by the supplier.

Post-condition:
A subsystem of ECUs is adapted according to the customer's specific wishes and is sold to the customer.

Description:
1) The supplier delivers the input information for the system configuration step.
The supplier delivers the ECU resource description for the hardware used in the subsystem to the OEM. The OEM adds this description to the ones used in the whole vehicle such that he gets a complete set of ECU resource descriptions for all hardware used in the dedicated vehicle.
The supplier delivers the application software (ASW) component descriptions for the ASW components of the subsystem to the OEM. The OEM adds these descriptions to his software architecture for the whole vehicle.
The supplier delivers the system constraints for the ASW components used in the subsystem to the OEM. The OEM integrates these system constraints into the system architecture.

2) OEM runs the system configuration.
The OEM integrates all three templates – the ECU resource description, the system template and the ASW component template. Now, the OEM runs through the system configuration step. This step is an iterative one. A first mapping of ASW components to the ECU hardware may lead to the need to adapt the system architecture as well as the software architecture. This leads to changes on the side of the OEM as well as on the supplier's side. So, a strong relationship between OEM and the involved suppliers is necessary.
The OEM extracts the configuration descriptions for the dedicated subsystem of ECUs that will be implemented by the considered supplier. The OEM delivers this configuration data to the supplier.

3) Supplier implements subsystem.
The supplier adapts its subsystem, generates the RTE and configures the BSW for each ECU. After the integration and testing on the ECU as well as on the subsystem level, the supplier delivers the integrated subsystem to the OEM.

4) OEM integrates system.
The OEM integrates the subsystem into the whole electronic/electric system. This includes the necessary integration and testing steps.

The use case shows an intensive interaction between OEM and suppliers. This is particularly

necessary when considering early integration on the vehicle function bus level. Such a new concept breaks traditional OEM – supplier relationships. As mentioned for the development processes as well as for the introduction of new OEM – supplier collaboration processes needs time and are difficult to establish. But the success of the whole standard depends on the success of the processes demanded by the standard.

2.4 Effects on tool landscape

The importance of exchange of models leads directly to the conclusion that the interoperability of tools is a key factor for enabling the AUTOSAR methodology in reality.
But, some tools use proprietary internal data formats and support AUTOSAR via Import / Export features. Although the data can be imported or exported to AUTOSAR, the merge of the exported data is complex as the proprietary formats usually do not follow the workflow of the AUTOSAR methodology.
Additionally to the AUTOSAR data, very often such tools need to store presentation data for the tool itself which leads to a mix of AUTOSAR data with non-AUTOSAR data in files.

Last but not least, a fact that should not be neglected is that the tools use different base technology. The users spend a lot of time in learning the tool rather than in using the tool. Moreover, tools based on different technology cannot be easily integrated in a seamless workflow.

3. ARTOP

A common framework, which enables the development of a continuous, lossless tool chain, is helpful to overcome the challenges presented in the previous chapter.

ARTOP (AUTOSAR Tool Platform) is such a framework jointly developed by the ARTOP user group. The ARTOP user group is a group of users of the AUTOSAR standard with a special interest in AUTOSAR tools. The organization of this group is comparable to the Eclipse Foundation. The ARTOP user group had been founded by BMW Car-IT, Continental AG, PSA and Geensys. Until now further partners joined. ARTOP is in general open to all AUTOSAR members.

The ARTOP user group provides an infrastructure platform for the development of tools used for the design and configuration of AUTOSAR systems. ARTOP implements the non-competitive base functionalities usually needed by all AUTOSAR tools.

The core component of ARTOP is the AUTOSAR meta-model implementation. It supports all available AUTOSAR meta-models like 2.0, 2.1, 3.0, 3.1 but also the newest 4.0. Furthermore, ARTOP encloses AUTOSAR XML schema conformant serialization, rule-based validation, model re-factoring, workspace management, example editors and further utilities.

Through ARTOP the interoperability of different tools that are used to support the methodology can be improved and commercial tools with better quality can be developed in less time, since only key functionalities have to be implemented.

Since the first release, published in 2008, more than 200 users had shown interest in ARTOP. In 2009 the first tools based on ARTOP arrived in the market. This shows the success of ARTOP. It seems that the automotive tool market awaited such an approach.

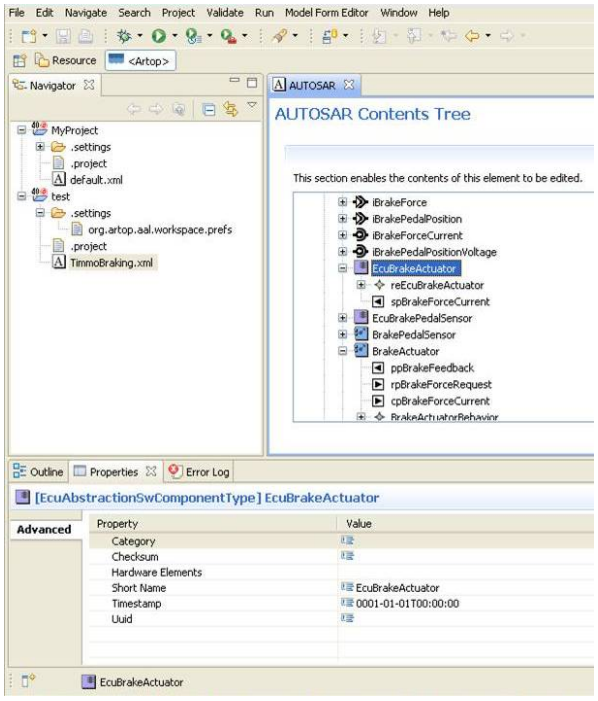

Figure 3 ARTOP surface with example tree editor

4. Eclipse

ARTOP is an Eclipse-based infrastructure platform, i.e. it is build on top of Eclipse and uses Eclipse technologies like the Eclipse Modelling Framework (EMF). Mainly, the technologies provided from Eclipse are taken by ARTOP and are applied to the AUTOSAR meta-model specifics.

It is not the goal of ARTOP to develop new, generic tool utilities. Eclipse already provides a modelling

framework and code generator technologies to develop tools based on structured data models.
For the purpose of this paper it is sufficient to highlight three helpful technologies provided by Eclipse that are attractive for the development of AUTOSAR tool-chains.

1. The Plug-In mechanism from Eclipse provides well defined interfaces to plug different functionalities together to an enriched tooling. As Plug-in code is loaded only if necessary this mechanism enables well performed reaction times of the tools on user input.
2. The Extension Points mechanism contributes functionality to a specific plug-in, which defined the extension point. This enables the users of tools to extend the functionality in an easy manner. Adaptations to the needs of the user can be operated by him. This is an often whished feature in the Automotive industry.
3. The Wizards utilities enable to guide the user through a defined sequence of steps to fulfil a specific task Eclipse provides support to easily create wizards.

Recently, an automotive working group had been founded in Eclipse. This group wants to define an Eclipse target platform for the automotive industry. This target platform should be used to develop tools for the whole development process lifecycle. AUTOSAR influences parts of this development process. Therefore, ARTOP is of highly interest for the working group. Due to restrictions given by the AUTOSAR contracts, the AUTOSAR dependent parts of ARTOP cannot be handled by the Eclipse automotive workgroup itself. But the AUTOSAR independent parts are taken over by the Eclipse working group. ARTOP is the first concrete use case for the workgroup.

5. ARTOP enables seamless tool chain

This chapter will present some challenges in the existing tool landscape and how ARTOP supports solutions.

5.1 AUTOSAR has boundaries

Having a look to the AUTOSAR methodology one recognizes that it does not cover the complete development process lifecycle. The influence of the methodology begins somewhere in the middle of the system architecture modelling step and ends directly before the compilation step. Topics like requirements management, hardware development or built management are not related with AUTOSAR.

One cannot expect that ARTOP provides a platform to enable a seamless tool chain starting with a requirements step up to an executable built step. But, as AUTOSAR influences directly a central part of the development process lifecycle, ARTOP is a good base for harmonization of the tool landscapes. It provides useful features to extend the platform itself – on editor level as well as on meta-model level. The extension point mechanism from Eclipse is heavily used.

5.2 Meta-model structure in-homogeneous

The structure of the AUTOSAR meta-model differs very much between the "work on system" part and the "work on ECU" part (**Figure 1**). This reflects the differences in the use cases to be covered on both sides.

The tool market follows these borders. Normally a tool works either on system level or on ECU level but not on both. The interoperability between both sides are usually done file based. ARTOP enables tool chains that do not require from the user to start several tools, to export from the one tool and to import to another tool. Everything is shifted from the file level to the meta-model level that enables a better data management consistency.

5.3 Concurrent modelling

In addition, in a seamless tool chain it is also important to consider the support of concurrent modelling, configuration and implementation. So, it is a key aspect that ARTOP enables the split of models such that several developers can work on different parts of the model independently from each other. In addition, the support of concurrent modelling also needs the support for successive iteration loops.
The integration of the work products delivered by the different developers need to be facilitated by advanced comparison and merge features. This is necessary to detect

- integration conflicts
- unresolved references
- incompatible versions
- missing parts of the models

Such workflow oriented support is not content of ARTOP itself. It has rather to be implemented by the commercial tools that are implemented on top of ARTOP.

5.4 Limitations of data format harmonization

The normative aspect of AUTOSAR presents an answer to the usage of different data format for

describing the same things. Nevertheless, also with AUTOSAR this has still some limitations.

- Mainly all today's projects are not 100% AUTOSAR, e.g. some projects implement only basic software with AUTOSAR but model the application software without AUTOSAR.
- As there are several releases of AUTOSAR, there is not the one and only data format. Transformations between the releases are still needed.

As ARTOP supports all published AUTOSAR meta-model releases, transformation algorithms can be implemented on top of the platform easily.

6. Case Study – CESSAR-CT

On the one side the arguments for an approach like ARTOP are obvious. It works and one gets something for free.
On the other side one may ask the question about why tool vendors should use ARTOP, as they may give sensitive parts of a tool development into an open user group.

The development of an AUTOSAR configuration tooling within Continental Engineering Services (CES) can be taken as a case study for the successful usage of ARTOP. Together with the AUTOSAR standard basic software – also implemented and put on the market - these software products enable the engineering activities of CES.

For the engineering business it is of main importance to have a flexible, extensible and customizable tool. Normally, these are not the attributes one finds in the tools available on the market. These are usually oriented on a fixed end user use case and not usable for an engineering expert that develops and adapts something for the customer.

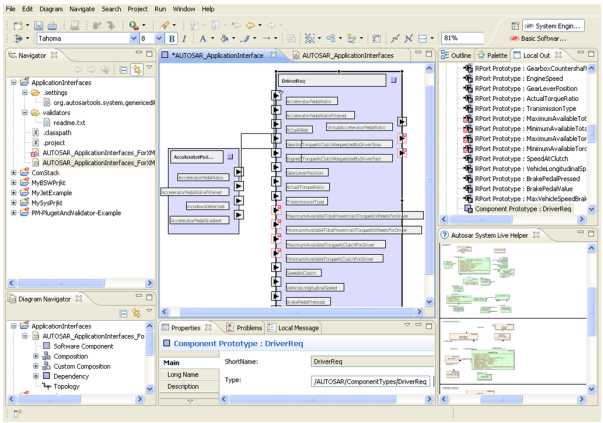

Figure 4 CESSAR-CT showing a SW-C editor

The main user roles supported by the tool are the "basic software developer" and the "ECU integrator".

Therefore, the focus is on the ECU part. But, with AUTOSAR also these use cases need information from the system part of the meta-model. E.g., Figure 4 presents a SW-C editor. It is a typical example for a situation, in which users normally have to open several tools in parallel. As CESSAR-CT is based on ARTOP the user gets a seamless AUTOSAR tooling.

It had been shown that the extensibility mechanisms provided by ARTOP are of main importance for CESSAR-CT. CESSAR-CT provides a Pluget mechanism such that the engineering expert or also the end user can extend the tool. Code generators of different technologies can be integrated and a form editor enables the easy extensibility and customization of the UI.

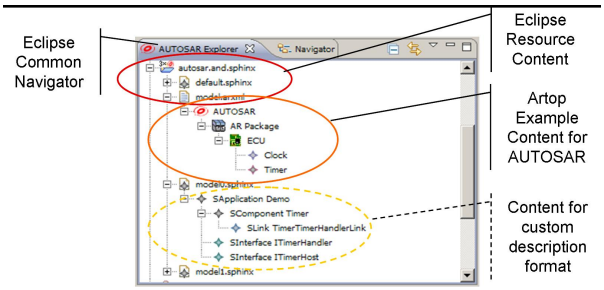

Figure 5 One editor based on several layers

Figure 5 shows a typical example about how the end product, the commercial tool, is created. A basic explorer is already provided by Eclipse. ARTOP adds AUTOSAR standardized elements. Last but not least, the tool supplier adds the use case specific features. This last step enables the usability of the tool for the end user.

As Continental were not a tool vendor before it had been a challenge to act as a founding member of the user group ARTOP. After nearly two years with ARTOP it is accepted, that the approach enabled CES to develop a tool that can be sold on the market.

7. Experiences

Experiences done within Continental Automotive have shown the importance of the seamless integration of the AUTOSAR tools together with the build tool chain. Indeed, with AUTOSAR architecture, it becomes natural that a project is an integration of many AUTOSAR modules (IO, COM, RTE...) developed by different companies. So, the first challenge to the project team, before making the ECU software running, is to make the development

tool chain running on a robust manner and with good performances so that it can be deployed to the complete project team.
If an AUTOSAR module or a set of modules (e.g. PORT, ADC, DIO...) is delivered by an external company, the delivery contains the XML configuration files, the related code generators and - sometime - specific tool editors. This means that the tool chain becomes very heterogeneous and more and more complexity is shifted to this area.

The following problems have to be considered:
Regarding the editors, the complete workbench becomes very heavy and not user-friendly because too many different tools must be started to edit the different modules. As a consequence, the working-performance is becoming really bad.
Moreover, as there are different tools, it also implies very often different workspaces or projects for the XML configuration files.
Regarding this problem ARTOP framework will bring advantage if it is largely deployed in the future. Thus, instead of delivering a complete standalone tool, we could imagine that the companies who deliver its modules also deliver the ARTOP plug-in for the edition.

Concerning the code generator, they could have been implemented on different platforms, but need to be executed from a unique build environment at the project level. For instance, it is important that:

- The code generator can be executed in command line mode so that they can be launched from the build.
- There is a way to check the dependencies between the generated .h/.c files and the input XML configuration files. Indeed this is important to not regenerate and then recompile more that necessary the .h/.c files. Else, it is a considerable loose of time for the project end-users to wait for regeneration and compilation. On the other side, it is important to launch the code generators affected by the modification of parameters in any of the input XML configuration files.

The AUTOSAR configuration model is very large and complex. On the side of the configuration tool one of the challenges was to develop editors which hide the model complexity and assist the user in its configuration work. Indeed, for the very first project, we started to make the configuration using generic editors. For starting, such generic editors were a good solution because it provides very quickly the capability to edit any parts of the whole AUTOSAR model with the same editor. However, the drawback of this editor is that it does not hide the AUTOSAR model complexity.

Indeed, these editors show the configuration model in a raw way (keeping the Containers / Parameters structure), which do not necessarily correspond to the user functional view. Moreover in the AUTOSAR model a quite big quantity of parameters have dependencies between each other, or are sometimes duplicated in different AUTOSAR model parts.

Also that does not include that each project and customer have its own requirements. For example, very often - in a project for an identical AUTOSAR model - the configuration is done by several users having different roles (e.g.: Basic SW developer, Basic SW integrator, Application SW engineer, and customer). Also the editor should assist the user in avoiding overwriting configuration parameters which he is not responsible for or making inconsistencies by detecting and modifying parameter with dependencies.

Of course, to improve as much as possible these generic editors, on the top of them we implemented many additional accessories (e.g.: validators, additional consistency checker and advanced tool tips, plugets...), but still in the project, the time spent in doing the configuration was too long.

Therefore we have introduced on the top of CESSAR-CT a Form Composer framework which enables the users to implement easily specific editors. The Form Composer is a powerful and lightweight declarative UI framework based on XML as markup language. Using the XWT (XML markup language), the users are able to build their own editors and to bind any UI controls to the AUTOSAR model by specifying its fully qualified name. When more complex operations need to be performed, of course Java can be used.

Indeed, as AUTOSAR is very large, complex and still moving, the tool development effort can not only be covered by a tool team. So, with this framework, we give the chance to any users (e.g.: AUTOSAR function specialist, project user members...) to develop their customized editors.
Moreover, as these Form Editors are placed on the top of the tool framework, they are very easy to deploy at the project level and do not need any update of the tool framework itself.

8. Outlook

The next phase in the development of AUTOSAR tool-chains will be driven by the new concepts of AUTOSAR release 4.0 that has been published end of 2009. The methodology and the meta-model had been enhanced in this release. "Functional safety"

and "timing extensions" are only two concepts with major influence that have been incorporated.
ARTOP will take these new concepts into account and provide platform support such that the idea of ARTOP can also be realized in future.

9. References

[1] www.autosar.org; webpage of the AUTOSAR partnership.
[2] AUTOSAR Methodology; see [1].
[3] AUTOSAR Technical Overview; see [1].
[4] www.artop.org; webpage of the ARTOP User Group.
[5] M. Rudorfer, S. Voget, S. Eberle; Artop Whitepaper; see [4].
[6] www.eclipse.org; webpage of the Eclipse foundation.
[7] H. Heinecke, M. Rudorfer, P. Hoser, C. Ainhauser, O. Scheickl; Enabling of AUTOSAR system design using Eclipse-based tooling; ERTS2008; Toulouse; 2008.
[8] S. Eberle, S. Voget, et al.; Artop – a shared platform for AUTOSAR tool development; Eclipse Embedded Day; 2009.
[9] M. Rudorfer, S. Voget, et al.; Artop – an ecosystem approach for collaborative AUTOSAR tool development; ERTS 2010; Toulouse; 2010.