7th EUROFORUM conference 'Software in the vehicle' 3-4 May 2006, Stuttgart, Germany
Euroforum conference May 3rd 2006

AUTOSAR – Current results and preparations for exploitation

Harald Heinecke, Jürgen Bielefeld, BMW Group
Klaus-Peter Schnelle, Nico Maldener, Bosch
Helmut Fennel, Oliver Weis, Continental
Thomas Weber, Jens Ruh, DaimlerChrysler
Lennart Lundh, Tomas Sandén, Ford Motor Company
Peter Heitkämper, Robert Rimkus, General Motors
Jean Leflour, Alain Gilberg, PSA Peugeot Citroën
Ulrich Virnich, Stefan Voget, Siemens VDO
Kenji Nishikawa, Kazuhiro Kajio, Toyota Motor Corporation
Thomas Scharnhorst, Bernd Kunkel, Volkswagen

AUTOSAR Development Partnership

ABSTRACT

This communication highlights a course status of the AUTOSAR development partnership by spring 2006.

A first comprehensive set of specifications has been achieved, which is referred to as Release 2.0. Major parts of this release are now available for download.

The partnership applies a proof of concept approach based on prototype implementations. The results from the first validation step sharpen and consolidate the 1.0 specification set. Consequently this validation process is repeated for release 2.0. The resulting controlled correctness will secure a set of AUTOSAR specifications that is ready for series applications.

In addition to specifying the software infrastructure, the AUTOSAR development partnership does investigate into a new methodology as a basis for more cost-effective system development processes.

In order to effectively support the exploitation of the standard, processes for a controlled maintenance and conformance testing are in the scope of the initiative, too. Also, means for a smooth migration from proprietary solutions to the AUTOSAR standard are available to avoid unacceptable disruptive effects.

A first complete set of specification for the purpose of commercial exploitation will be made available by the end of 2006. In addition, the AUTOSAR development partnership is now preparing for the Phase 2 of its initiative in order to continue its efforts of standardization to the benefit of the automotive industries.

1. INTRODUCTION

In order to cope with the tremendous increase in functional scope of automotive electronics and thus implicitly software, a significant reduction of structural complexity by the means of an industry-wide standardized software infrastructure is mandatory.

To achieve this, in summer 2003 the AUTOSAR development partnership was formally established, with the majority of work packages starting in 2004. In addition to the 10 Core Partners, this initiative is now significantly supported by 48 Premium Members (see ref. 1 for details) with active contributors from the entire automotive E/E value system, ranging from semiconductor industries, tools and software vendors through tier 1 suppliers to the car makers themselves.

The first main focus of this initiative was the software infrastructure, e.g. to provide technical means within a layered software architecture for abstraction from hardware on the lower interface and abstraction to application layer software on the upper interface.

The efforts of this initiative are, however, not limited to software infrastructure. Instead, the focus expands also to specification of compatible functional interfaces and the design of a coherent methodology based on standardized templates and data exchange formats.

In addition, technical and procedural guidelines are being established to support the exploitation of the standard under free competition.

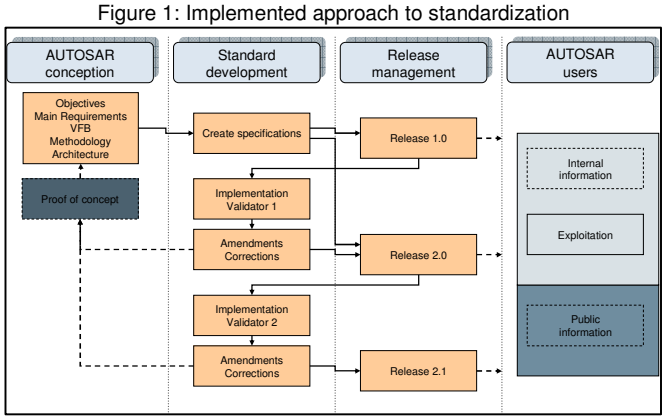
2. STATUS OF THE AUTOSAR INITIATIVE

By 2006, the AUTOSAR development partnership is in the final phase of completing a full set of specifications for the purpose of commercial exploitation. In addition to complementing the technical coverage of the standard, since mid 2005 there are significant running activities on implementation and testing. This is part of the proof of concept approach.

2.1 APROACH TO STANDARDIZATION

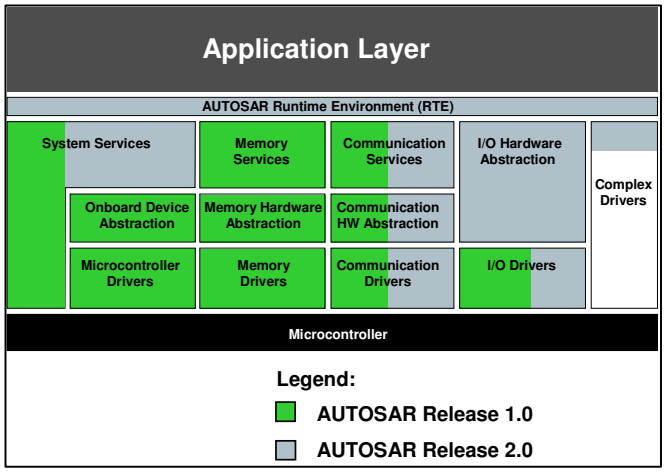
The AUTOSAR development partnership selected a phased strategy in order to manage the challenges ahead along with the tight schedule. Figure 1 illustrates the implementation of this approach.

Figure 1: Implemented approach to standardization



During the first phase the main focus was on establishing core functionality on the infrastructure level. Mid of 2005, this led to a set of specifications, which represents the AUTOSAR Release 1.0. Figure 2 shows the content of the two first releases on the infrastructure level.

Figure 2: Content of AUTOSAR Release 1.0 and 2.0



In addition to keeping control over the schedule, this approach enabled an early checkpoint to verify the overall concept and the individual module specifications.

The proof of concept approach includes a series of two validation steps as shown in Figure 3.

Figure 3: Two step approach of the validation

	Step 1 Validator 1	Step 2 Validator 2
Applications	Test Program	Test Program
Standard Software	BSW 1.0 Implementation	BSW 1.0 & BSW 2.0 + RTE Implementation
Configuration	manual proprietary configuration	Part of AUTOSAR configuration concept
Tools	proprietary configuration tool for BSW	Support for configuration concept
Hardware	Standard Evaluation Board	Standard Evaluation Board

In essence, both Validator 1 and Validator 2 are integrated prototype implementations. These validation steps are complemented by further proof of concept and demonstration activities performed by the AUTOSAR partners individually.

In conclusion, whilst the basic mandate of the AUTOSAR initiative is to deliver specifications (i.e. essentially paperwork), a proven quality assurance approach has been selected. This approach is based on two prototype implementation and integration steps, each of which enables detection and resolution of inherent errors in order to reach maturity for series line implementation prior to the final specification release.

2.2 EXPERIENCES FROM THE VALIDATOR 1

During the second half of 2005, the implementations for Validator 1 were performed, concentrating on the Basic Software (below the RTE in Figure 2). These activities were controlled by a dedicated AUTOSAR team, which prepared and synchronized the co-operation between 14 implementing parties (which were selected from the AUTOSAR partners and members) and the integrator.

A total of 31 different software modules (realized in 56 individual implementations) were implemented and subsequently integrated on two platforms (evaluation boards based on a 16-Bit and a 32-Bit microcontroller respectively). The approach was selected to verify both hardware dependent and non-dependent modules. Also, redundant implementations were applied for several modules to prove the viability of the concept and confirm the exchangeability for different module implementations.

As a result from the implementations, more than 260 change requests to the specifications were raised. In contrast, after this first specification refinement only a few conceptual changes were requested by the integration stage. This indicates that the earlier stages did effectively resolve the majority of issues, which in turn strengthens the correctness of the architectural approach.

It is clearly the main task and result of Validator 1 to confirm the AUTOSAR specifications. This is evidenced by the successful tests of the prototype implementations. Therefore in a very controlled timing, all specifications are synchronized in a high level of maturity.

The modules implemented for Validator 1 are available from the suppliers which are AUTOSAR members, strictly on the basis of the AUTOSAR mission:

Cooperate on standards – compete on implementations

2.3 AUTOSAR RELEASE 2.0

In accordance to Figure 2, Release 2.0 adds further module specifications to complement the software infrastructure. Existing module specifications were updated and amended by missing features. This increased module environment enables also to reveal issues that could not be detected before.

A controlled change process was introduced and applied to those changes resulting from the validation activities. The main improvements relate to the consistency of all specifications.

Generally, the experience gained and the lessons learned from Release 1.0 and the Validator 1 did significantly help in establishing Release 2.0, which has been completed early 2006.

Major parts of the Release 2.0 specifications are now available for free download from the AUTOSAR website. These specifications are intended for the purpose of information only. Nevertheless their availability enables early preparations for the exploitation phase. Any commercial exploitation of the specification requires an AUTOSAR membership.

2.4 THE VALIDATOR 2

The Release 2.0 specifications are currently being implemented as part of the validation approach. The efforts are distributed to 12 implementers and 1 integrator.

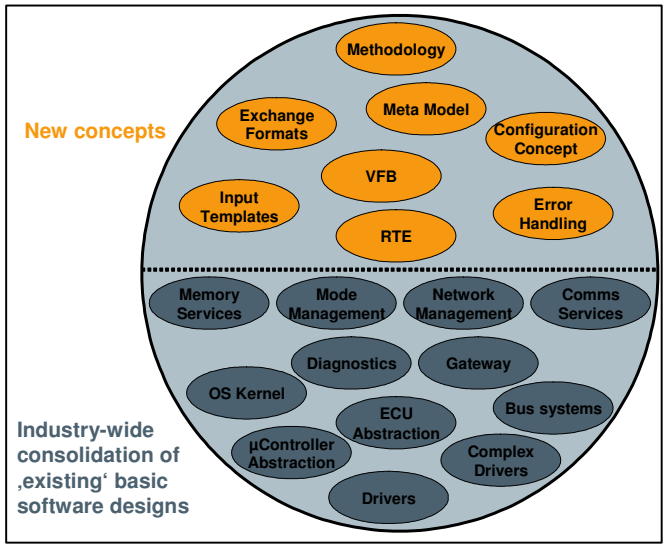
The results are intended to feed into Release 2.1, which is intended to be cleared as a basis for commercial exploitation by the end of 2006.

3. NEW TECHNICAL CONCEPTS

Release 2.0 comprises specifications for the basic software area as well as other technical areas (Figure 4).

The AUTOSAR basic software (BSW) is based on well known infrastructure elements, but does also introduce several new BSW-wide concepts, such as configuration and error handling. The major achievement is that for the first time a comprehensive (and domain spanning) infrastructure is now available that consolidates and complements the know-how and experiences from car makers and suppliers around the globe.

Figure 4: Technical scope of the AUTOSAR standard



The new concepts include:

3.1 RUNTIME ENVIRONMENT (RTE)

From an abstract point of view (i.e. at system design time), the RTE is the run-time implementation of the Virtual Functional Bus (VFB) on a specific ECU. The notion of the VFB translates to all communication mechanisms that are relevant to the application layer software.

Conclusively, the RTE abstracts the application layer from any implementation details of the basic software and from hardware aspects. With this in mind, the RTE is a novel middleware layer technology that enables transferability of application layer software components across the network.

Figure 5: Communication via RTE ports

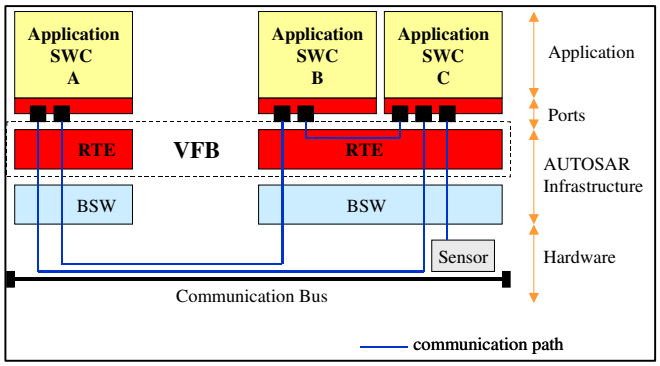


Figure 5 shows an example of the communication mechanism (here client-server based) available to application layer functions. This communication is channeled via the RTE. The communication layer in the basic software is encapsulated and not visible at the application layer.

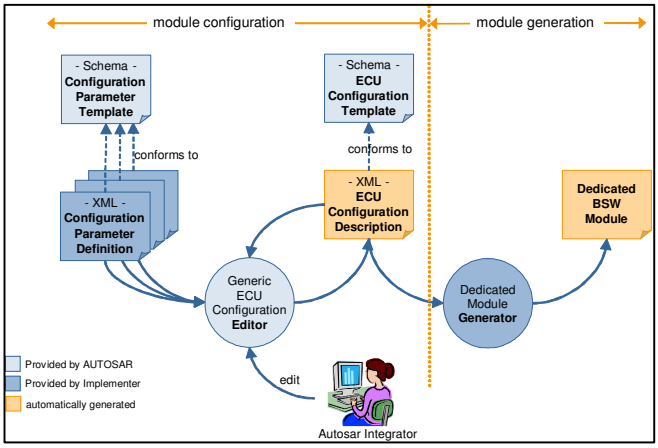
AUTOSAR defines a standardized component model consisting of both a clear programming language mapping (syntactically) and a file format for component requirement and capability description. With this component model consisting of both communication mechanisms and scheduling related concepts, the RTE allows the decomposition of the application software layer into well structured coupled components.

These components can be moved across ECUs during concept phase or reused across projects (ideally without touching the source code), thus reducing engineering and testing effort significantly together with safeguarding of quality.

3.2 EXTENDED CONFIGURATION CONCEPT

The support of multiple configuration variants enables pre-compile, link-time and post-build parameter configuration classes or a combination thereof.

Figure 6: The Generic Configuration Editor



AUTOSAR defines formats to describe the capabilities and parameters of basic software modules including XML schema definitions. Using those descriptions, a generic editor (see Figure 6) can be used to configure all software modules compiled into one ECU.

This enables editor vendors to include convenience functions and consistency checks into their products that are impossible today and will boost both engineering speed and product quality.

The AUTOSAR specifications for basic software modules allow module vendors to offer multiple modes of module configuration. A BSW module can be designed using:

- Compile-time configuration, for best runtime performance,
- Run-time configuration, for highest flexibility, or
- Link-time configuration to achieve a trade off between the two borders.

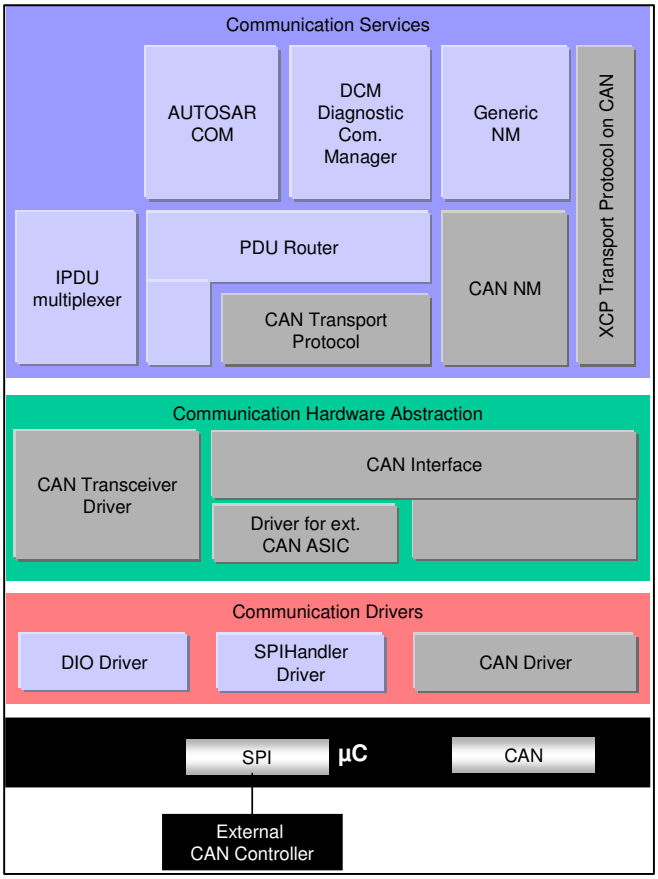
Compile time and link time configuration utilizes code generation based on module configuration description files created by the generic editor as described above.

Conclusively, AUTOSAR supports the way towards an integrated and tightly coupled tool chain for automotive software development.

3.3 ABSTRACTION INSIDE THE INFRASTRUCTURE

As a first detailed view of the architecture the CAN communication stack is shown in Figure 7 (compare to Figure 2).

Figure 7: The communication stack for CAN



The CAN communication services are a group of modules for vehicle network communications with CAN. They provide a uniform interface to the CAN network, hiding protocol and message properties from the application. AUTOSAR COM and Diagnostic Communication Manager provide uniform communication mechanisms for the application, both are bus technology independent. The PDU Router allows direct routing of Process Data Units from one bus to another. In addition, within AUTOSAR COM a signal-based gateway is included in order to route single signals from one communication system to another.

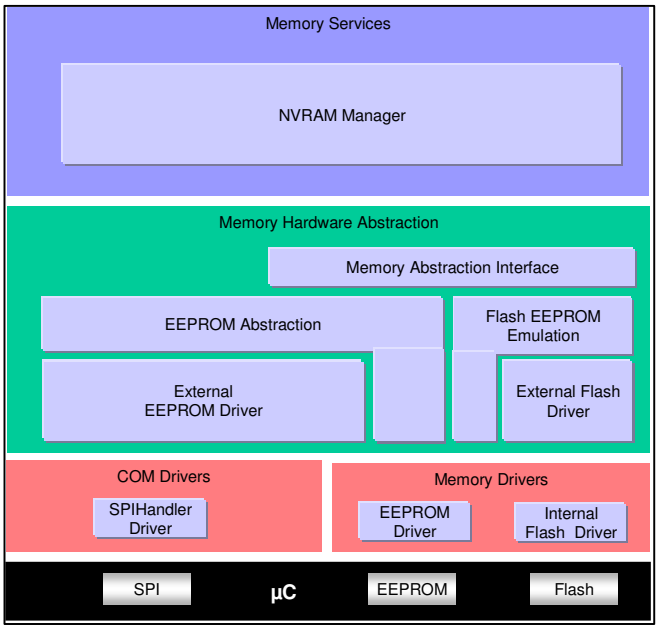
The network management is split in a bus independent and bus dependent part. If CAN is replaced by FlexRay, the Generic NM remains the same. By introducing a

CAN Interface the PDU Router does not care about the fact whether the CAN controller is a part of the embedded controller or an external device. By replacing the CAN-dependent modules by LIN- or FlexRay modules the CAN communication stack can be transformed to the LIN or FlexRay communication stack.

As a second example the memory stack of the AUTOSAR architecture is depicted in Figure 8.

The memory services given as the NVRAM-Manager provide a uniform access of the applications to non-volatile data. It abstracts memory from location and properties. Further on it provide mechanisms for non-volatile data management like saving, loading, checksum protection and verification or reliable storage. The memory hardware abstraction (below the memory services) abstracts from the location of the peripheral memory devices (on-chip or on-board) and the ECU hardware. E.g. the EEPROM interface and the flash hardware should be accessible via an equal mechanism.

Figure 8: The memory stack



The memory drivers are accessed via memory specific abstraction/emulation modules (e.g. EEPROM Abstraction). By emulating an EEPROM interface and Flash hardware units a common access via Memory Hardware Abstraction to both types of hardware is enabled.

Both examples CAN Communication stack and Memory stack show a high degree of modularization to achieve an optimum on modularization and reuse of infrastructure software for very different use cases.

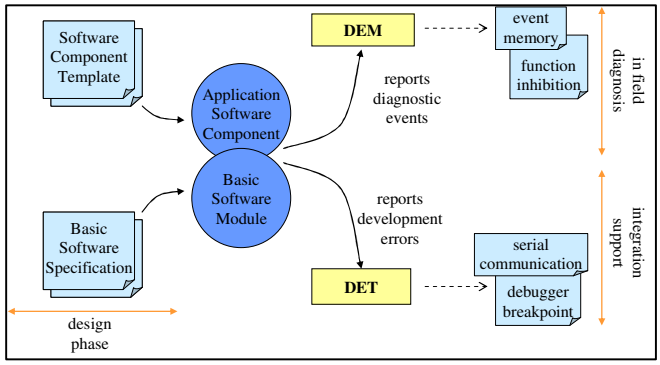
3.4 ERROR HANDLING

A consistent and unambiguous error handling supports effective communication to application layer functionality and can also be used as a means for mode management and diagnostic purposes. Use cases include broken sensors, communication errors and memory failures.

For both, basic software modules and the RTE there are mechanisms defined to propagate error information throughout the system. Therefore all API functions return error codes that are well defined in the basic software modules specifications respectively in the descriptions of the application software components.

An identified diagnostic event can be resolved directly in the client module or escalated toward upper software layers. Additionally any diagnostic event occurrences are reported to one of the two error handling facilities provided by the AUTOSAR basic software (see Figure 9).

Figure 9: Diagnosis during development and normal operation



The first facility, the development error tracer (DET) is utilized to report diagnostic events that are of interest only during the integration or implementation phase. The DET can be configured by the system integrator to report diagnostic events by any desired means and is disabled during normal operation.

Diagnostic events that shall be monitored during normal operation are reported to the diagnostic event manager (DEM) which can be configured to enter the event occurrence into the diagnostic event memory and to change the mode of operation or inhibit an arbitrary functionality of the application software.

3.5 FUNCTIONAL INTERFACES

In order to facilitate integration of application software components in a system, clear semantics of the interface are being identified in function catalogues. Descriptions of the interface that conform to the standardized AUTOSAR format are used as an input to the development process without the need to disclose any internal design, which is often sensitive to competition.

3.6 METHODOLOGY

The AUTOSAR Meta Model precisely defines the concepts used to describe a self-contained system with AUTOSAR. A direct derivation of the Meta Model are the exchange formats (based on XML ‘templates’), which are thus inherently consistent. The AUTOSAR methodology

covers all major steps of system development, starting with the input descriptions for

- Software components
- ECU resources
- System constraints

All subsequent development steps up to the generation of executable code are supported by the AUTOSAR methodology by defining exchange formats and work methods for these steps. Whilst the AUTOSAR methodology is not a complete process description itself, it can be applied as a basis for shared development processes. In analogy to the benefits of technical interface compatibility, the AUTOSAR methodology has the potential to enable synergies and improved effectiveness when applied properly in a collaborative business context.

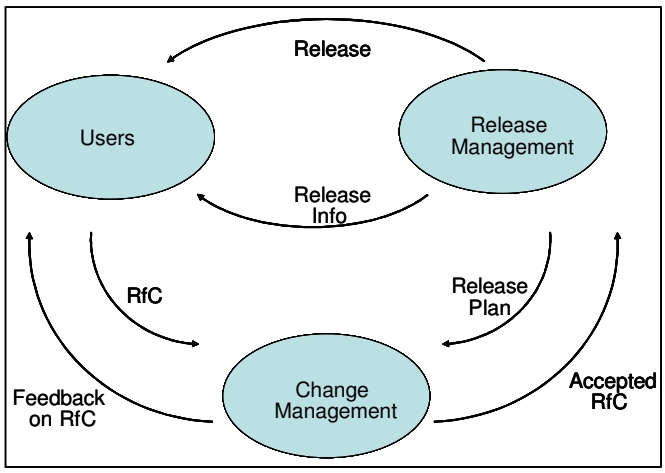
4. PREPARING FOR EXPLOITATION

In preparation for the exploitation of the AUTOSAR standard, the AUTOSAR development partnership is establishing technical and procedural guidelines.

4.1 STANDARD MAINTENANCE

In order to secure the achieved level of standardization whilst allowing for further developments and innovation, controlled Standard Maintenance procedures are mandatory. The main elements of standard maintenance are Change Management and Release Management.

Figure 10: Elements of Standard Maintenance



- **Change Management**
  Currently, changes do mainly result from the validation activities.
  The Change Manager is the direct interface for the change initiator (who raises a Request for Change – RfC) and informs on the progress of the RfC.
  Internally, a Change Control Board assesses and approves the RfC as appropriate. Where necessary this can involve expert evaluation. In analogy to the validation approach during standard development, implementation, integration and test can be used as a means of quality assurance also on the level of a single change or multiple interlinked changes.

- **Release Management.**
  In close synchronization with the Change Management, the Release Management covers issues such as release numbering and planning as well as pre-announcements and application notes for upcoming new releases of the standard.
  Any AUTOSAR release will normally be valid (that is actively supported by AUTOSAR) for a few years allowing for an overlap period when a more recent release is issued. Again, timely communication to the AUTOSAR community is a main feature of the Release Management.

4.2 CONFORMANCE TESTING

A supplier of automotive products that are developed in accordance to the AUTOSAR standard can demonstrate their standard conformance by successfully completing the conformance test process.

A quality mark for many buyers will be an attestation of conformance issued by a third party, which is referred to as a Conformance Test Agency (CTA).

The tests itself will be executed against a Conformance Test Suite (CTS), which implements the conformance test specifications (which will be available in a formal language as part of the AUTOSAR specifications).

The internals of application layer software are outside the scope of the AUTOSAR standard, which limits to specifying the lower interface only. In correspondence, the conformance specifications for application layer software will first relate to the software component descriptions (which is derived from the meta model).

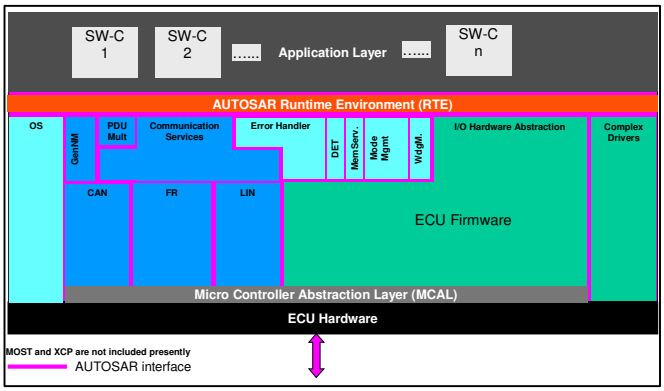
On the infrastructure level (i.e. basic software and RTE), a different test methodology is required to reduce the complexity. This is achieved by the introduction of conformance classes. For any given automotive product the relevant set of test groups/cases is specific to the actual combination of two parts:

- **Implementation Conformance Classes (ICC)**
  The ICCx provide a standardized way of packaging modules:
  - ICC1 encapsulates the entire BSW + RTE in one single cluster
  - ICC2 defines some 10 clusters each of which combines adjacent modules (see also Figure 11)
  - ICC3 separates all individual modules in own clusters (currently more than 50)

- **Functional Conformance Classes (FCC)**
  The FCCx focus on functional behavior and differentiate between mandatory and optional features:
  - FCC1-C defines the Core Feature Set that must be available

- FCC2-M defines Module Options, i.e. features inside a module, which can be optionally switched on or off
- FCC2-I defines Integrate Options, i.e. optional features, which need to be aligned over several modules/clusters. Examples for FCC2-I features are timing or memory protection and transmit confirmation

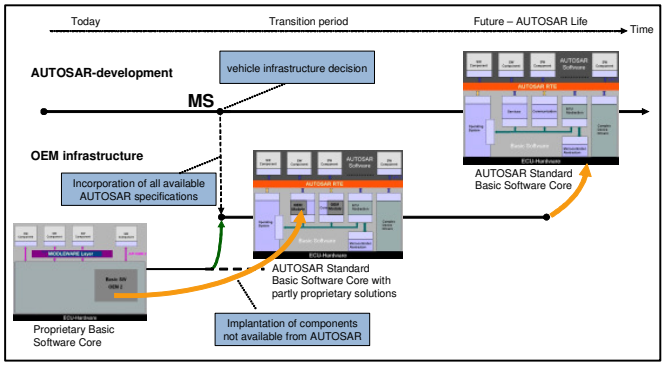
Figure 11: Implementation Cluster Conformance Class ICC 2



4.3 MIGRATION TO AUTOSAR

The introductions of the AUTOSAR standard in series development will unlikely occur in a single step. Instead, in order to secure existing assets, synchronize to vehicle development cycles and as a matter of risk minimization, a staged introduction and controlled maturation in any given development context will be preferable.

Figure 12: A migration scenario



To account for this the AUTOSAR development partnership did – at an early stage of standard creation – agree on the possibility of a staged transition from proprietary software architecture to a full AUTOSAR conform architecture via intermediate steps with appropriate implantations. A possible scenario is shown in Figure 12.

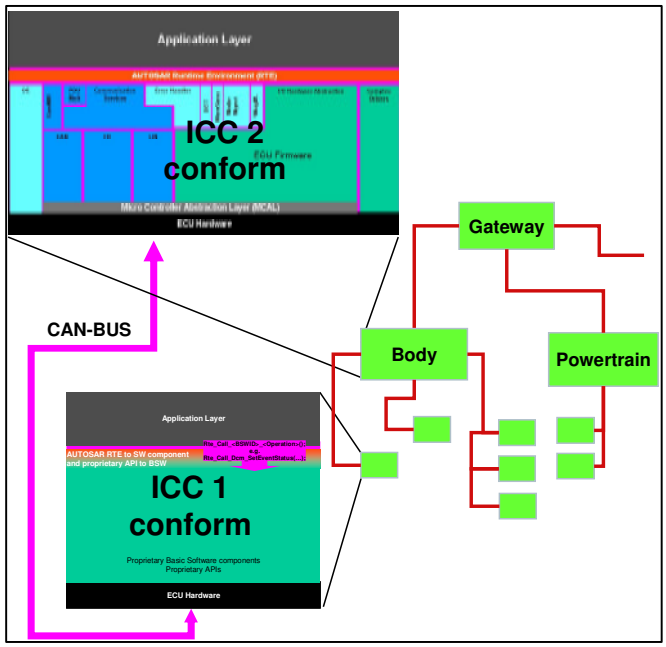
The downside of this transition approach is that the resulting intermediate architectures will still be proprietary solutions and, for example, the means of conformance testing will not or only partially apply.

Another approach to a smoother migration is enabled with the definition of ICC1. This is true on an intra- as well as inter-ECU level. In practice both aspects will correlate.

Inside an ECU, ICC1 can be used at the initial migration step. Only standard compatibility on the application layer interface given as software components and to the bus-communication is required. Nevertheless this already enables full re-usage and transferability of all functional components. It is, however, still required to migrate to full AUTOSAR architecture (i.e. ICC 2/3) otherwise the full benefits of a standardized software infrastructure cannot be used.

Owed to the compatibility of communication and network management mechanisms, an ICC1 conformant ECU can be connected to ECUs with higher conformance classes in a networked system (see Figure 13).

Figure 13: Combination of ICC 1 to ICC 2 conform ECUs in a network



Finally, an ICC1 conform basic software implementation does to some extend assimilate a Complex Device Driver (CDD). Hence, in reverse logic the CDD can also be used as a migration mechanism likewise. Despite the fact that the CDD is designed for other purposes (like complex sensor evaluations/actuator control or access to new hardware technology), there is a possibility to use this concept to migrate existent implementations without the instant need for full re-engineering.

4.4 TOOLING

The availability of powerful tools is a critical success factor for the exploitation of the AUTOSAR standard. The specifications of these tools are outside of the scope of the AUTOSAR development partnership. However, the clear exchange formats defined for the various steps in the AUTOSAR methodology do effectively enable a seamless tool chain.

In parallel to contributing to the creation of the AUTOSAR specifications, several tool vendors are currently preparing appropriate tool solutions. For example, AUTOSAR compatible tools e.g. for system architecture, network design or RTE generation do already exist – at least in evaluation versions.

In addition, the proof of concept approach is concerned with selected parts of the tool chain, too. Actually, Validator 2 features prototype implementations for a generic ECU configuration editor and two redundant RTE generation tools.

5. AUTOSAR PHASE 2

The current development agreement and the membership contracts expire by end of 2006. However, for the time period starting January 1st 2007 a continuation of the development partnership with stable Core Partners is agreed.

The current membership structure will be maintained. Accordingly, there will be revised contracts available for Premium, Development or Associate Membership or Attendees. The new contracts will be available for the signature process from 1st July 2006 on.

6. CONCLUSIONS

The AUTOSAR development partnership is in the final phase of preparing a consolidated set of specifications for the purpose of commercial exploitation. All AUTOSAR members can develop automotive applications in accordance to the released standard and market them in free competition.

The maturity of the standard specifications for series development will be effectively secured by two prototype implementation and validation steps. In addition to allowing for a controlled migration, procedural and technical support for Standard Maintenance and Conformance Testing will secure a sustained usability of the standard for automotive E/E applications.

A significant benefit is enabled by the standardized software infrastructure and AUTOSAR design technical elements of the standard. In addition the AUTOSAR methodology has the potential to enable gains in both effectiveness and efficiency of future multilateral development processes, leveraged by the availability of tools in a seamless chain.

All current members and new prospects are invited to build on these achievements and to contribute to the continued development of AUTOSAR phase 2.

ACKNOWLEDGEMENTS

The Core Partners are indebted to all AUTOSAR Premium Members for their continuous exciting contributions. Without their effort and commitment the AUTOSAR idea and realization would have been out of reach.

The authors would like to acknowledge Peter Witzgall and Tilman Ochs for the contributions in setting up this communication.

REFERENCES

1. http://www.autosar.org
   Official website of the AUTOSAR development partnership
2. Scharnhorst, Thomas et al.: AUTOSAR – Challenges and Achievements 2005, Electronic Systems for Vehicles 2005, VDI Congress, Baden-Baden, 2005
3. Heinecke, Harald et al.: AUTomotive Open System Architecture – An industry-wide initiative to manage the complexity of emerging Automotive E/E-Architectures, Convergence 2004, Detroit, 2004
4. Heinecke, Harald et al.: AUTOSAR – An industry-wide initiative to manage the complexity of emerging Automotive E/E-Architecture, Electronic Systems for Vehicles 2003, VDI Congress, Baden-Baden, 2003

CONTACT

request@autosar.org

DEFINITIONS, ACRONYMS, ABBREVIATIONS

AUTOSAR: Automotive Open Systems Architecture

BSW: Basic Software

CDD: Complex Device Driver

CTA: Conformance Test Agency

CTS: Conformance Test Suite

DEM: Diagnostic Event Manager

DET: Development Error Tracer

FCC: Functional Conformance Class

ICC: Implementation Conformance Class

RfC: Request for Change

RTE: Runtime Environment

VFB: Virtual Functional Bus