SAE International™

AUTOSAR as a Key Enabler for Collaborative Product Development

2010-01-2341
Published
10/19/2010

Mark Jensen and Tony Mascolo
Vector CANtech, Inc.

ABSTRACT

Whether it be in highly visible features like fascinating new infotainment systems or hidden behind the scenes in complex new hybrid powertrain controls, in-vehicle software is rapidly changing the way the automotive industry engages its vehicle-buying customers. In every application where a compelling new electronic solution is emerging, it is enabled by the convergence of in-vehicle software developed by different collaborating partners. As more and more component suppliers, vehicle OEMs, and technology vendors enter into collaborative software development projects with each other, a new set of technical and business challenges are showing collaborative software development to be a very distinctive proposition than traditional stand-alone development. This paper will highlight key AUTOSAR concepts that provide an open-standard framework for collaborative development of in-vehicle software that provides natural boundaries for both the technical and business interfaces between partners and their software.

INTRODUCTION

Automotive electronics and in-vehicle software are fertile ground for innovation. Where innovative and compelling new electronic product content is emerging, it is enabled by the convergence of in-vehicle software developed by different collaborating partners, including key technology companies from the consumer electronics, telecommunications, information technology, and defense industries.

In traditional stand-alone software development, all software developers work together in a single team under a single leadership structure to deliver a product that does not interact with other products from other organizations. The challenges of these teams are well-documented by vast bodies of work in both academia and industry. There are a variety of software development philosophies and technologies that successfully address these challenges. However, this independent stand-alone software development scenario is almost non-existent in today's automotive industry.

With almost no exceptions, software development in today's automotive industry involves multiple development teams working in different companies, in various geographic locations with unaligned processes to deliver software that directly interacts with software from other teams. Most modern vehicles contain software from dozens of such development teams, compiled from millions upon millions of lines of source, and distributed over dozens of networked electronic controls. This pervasive interaction between the vast collection of software components in the vehicle means that collaboration is not a special case at all, but rather a norm that is already in place across the entire industry - even if it is not formally recognized in most cases. With the realization that collaboration is already an integral part of the way the automotive electronics industry does business comes the requirement to master it. Collaborative software development comes in several forms. Each form of collaborative development faces challenges that have yet to be effectively and consistently mastered and managed by the automotive industry.

AUTOSAR is an emerging standard software framework designed especially to meet the needs of automotive electronic systems. AUTOSAR is developed by a voluntary consortium of companies from the automotive electronics industry. The AUTOSAR consortium specifies the AUTOSAR software framework in a collection of documents that all consortium member companies can utilize. While the AUTOSAR consortium strives to achieve a multitude of goals, enabling efficient collaborative software development is a central theme and objective.

AUTOSAR offers significant opportunities for the automotive electronics industry to address the challenges of collaborative software development and even master it. With AUTOSAR, it is possible to enable collaborative product development through a common context that all industry participants can understand and work within. Software and software development processes in the AUTOSAR context are inherently collaborative in nature and implicitly eliminate many of the challenges facing less-structured approaches to collaborative development.

This paper will examine three points -

• Software is the single biggest and key enabler for collaborative product development in automotive electronics.

• Traditional ad-hoc forms of collaborative software development face serious challenges to efficiency, productivity, and severely limit the value of collaboration.

• AUTOSAR provides a context that implicitly enables the collaborative development of automotive electronics products and eliminates many of the challenges facing traditional collaborative development efforts.

SOFTWARE AS THE KEY ENABLER TO COLLABORATIVE DEVELOPMENT OF AUTOMOTIVE ELECTRONICS

Proof of the central role and value of software in product innovation is easy to ascertain from some of the current high-profile innovations in the automotive industry -

• Adaptive cruise control starts with a typical cruise control system and adds a radar system. Software implementing algorithms for pattern recognition and drivetrain management binds the two systems together.

• Hybrid propulsion starts with a traditional internal combustion engine and adds an electric motor. Controls software coordinates the use of the two power plants to deliver an optimal combination of performance and operating efficiency.

• Popular new infotainment systems start with a traditional audio system and add a microphone, a GPS receiver, a Bluetooth wireless transceiver, and a serial data port like USB. Voice recognition software, digital maps, navigation software, MP3 playback software and an overarching software application glue all of the hardware components together to deliver integrated entertainment, telephony, and navigation capabilities though a single intuitive and consumer-friendly interface the vehicle-buying customer sees as a single system.

• Connected vehicle technologies will start with traditional vehicle electronics and add WiFi or other wireless connectivity to transform the entire vehicle into a single node in an ad-hoc network. Vehicle nodes and smart infrastructure nodes will discover each other and interact through software-enabled services that each node can provide or use.

In each of these examples, two, three, or more partners are jointly and yet independently working to create new and innovative vehicle electronic systems and solutions that none could effectively develop and deliver on their own. In all of these examples, in-vehicle software is the key enabler that joins these complementary technologies and partners together. As software continues to expand in scope and importance in the automotive electronics domain, collaborative software development is critical to innovation.

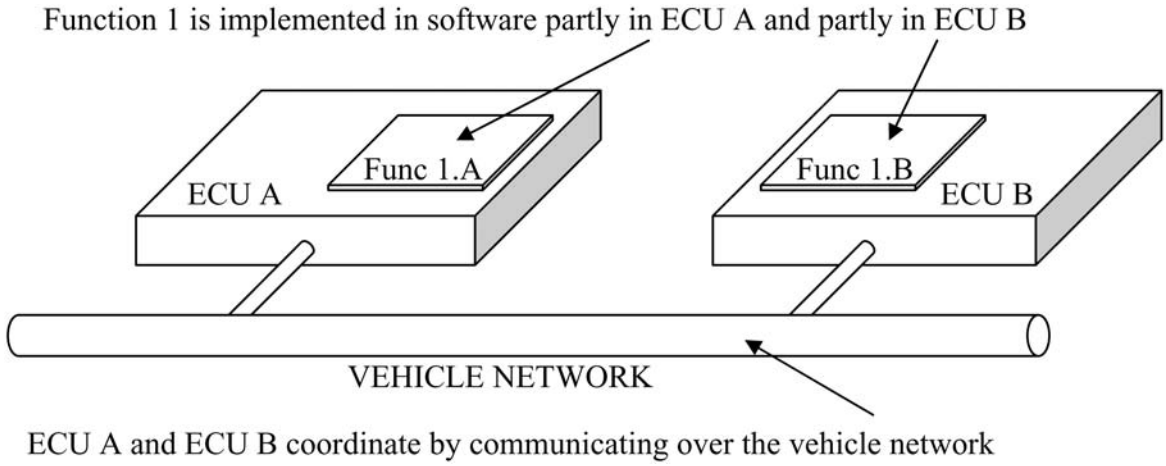
But high-profile futuristic features are not the only place collaboration takes place in the development of vehicle electronic systems. Software-based collaboration takes place at multiple levels in the vehicle.

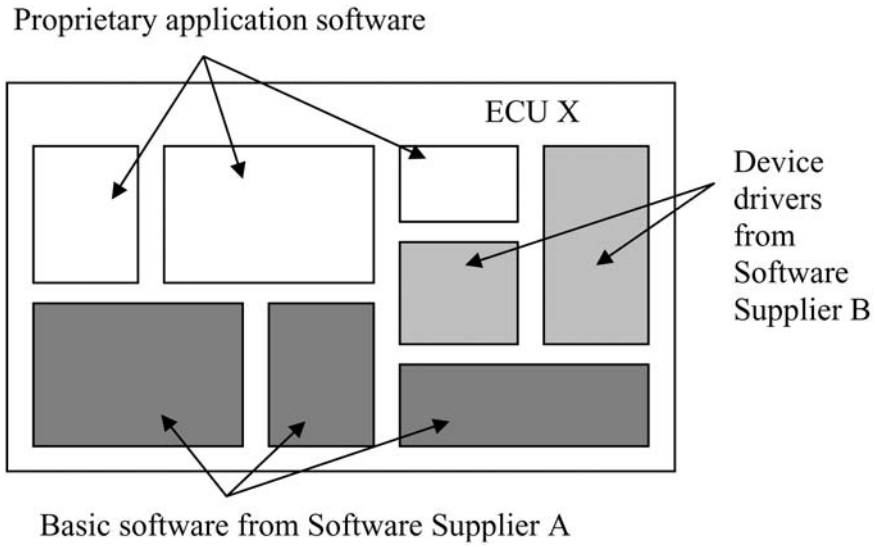
VEHICLE SYSTEM LEVEL

Electronic control units (ECUs) for traditional vehicle functions like engine control, transmission control, anti-lock braking, airbag control, central door locking, and lighting still require collaboration, but of a different kind than the direct collaborative development examples described in the previous section. The behaviors of the individual ECUs in the vehicle are coordinated through interaction over in-vehicle communication networks. While the developers of these traditional electronic control units do not necessarily directly collaborate, the vehicle OEM acts as a collaboration agent that works to seamlessly align the network interfaces between the ECUs and integrate them into cohesive systems. Like the explicitly collaborative efforts seen in hybrid propulsion and new infotainment systems, this looser and informal approach to collaboration is also enabled by software. These systems are bound together by the software stacks that implement the communication protocols and data exchange mechanisms that all participants on the network capitalize on to interact and perform in a coordinated way at the vehicle level.

COMPONENT LEVEL

At the component level, collaboration also occurs through software, but at an even lower level inside of individual ECUs. As electronic control functions for sophisticated behavioral algorithms, complex I/O drivers, power management, network communication, flash memory drivers, and diagnostic monitors become more and more specialized, there is a natural progression toward having multiple software partners with different areas of specialization provide software components that work together as a single software system inside the ECU. In this form of collaboration, it is the ECU manufacturer that acts as the collaboration agent that seamlessly aligns the interfaces between the software components and integrates them into the ECU system. Where





an ECU's functional behavior is considered a core competency or a competitive advantage for the vehicle OEM (as is frequently true with powertrain controls and central body/gateway controls), the ECU manufacturer is sourced to provide the ECU hardware and basic software while the vehicle OEM provides its own application software.

So, whether it is in game-changing new innovations, networked vehicle systems, or inside the ECU, software is the single biggest enabler for collaborative product development in the world of automotive electronics.

CHALLENGES TO COLLABORATIVE SOFTWARE DEVELOPMENT

Collaborative development does not just happen on its own. Collaborating partners must have a framework that allows their respective contributions to come together and provide value greater than the sum of the parts. This framework for collaboration must effectively and efficiently harmonize these aspects of the software development process:

- software design
- software re-use
- implementation specification
- inter-organizational communication

An exploration of each of these points will demonstrate the challenges that must be addressed in order for collaborative software development to not just succeed, but to become a robust and repeatable process that consistently delivers higher value than any collaborating partner could produce for itself at a similar cost.

SOFTWARE DESIGN

Design for assembly is a well-known concept on the mechanical side of the automotive industry. Design for

assembly challenges designers of mechanical parts to look beyond the functional requirements and physical performance of their part and to consider how that part will be physically placed and fastened into the vehicle assembly. A good design for assembly can significantly improve production efficiency, reduce operating costs, and improve the bottom line for the vehicle manufacturer. In order to succeed with a design for assembly initiative, the vehicle manufacturer must orchestrate informal collaborations at many points along the product development process and across the complete supply chain.

Collaborative software development activities share many common themes with the development of mechanical systems. The software counterpart to design for assembly is "design for integration". Design for integration asks software designers to look beyond the functional requirements and behavioral performance of their software and to consider how the software will be integrated with the other software in the vehicle system. A good design for integration approach must address several challenges in order to deliver optimal value.

The time and effort that goes into designing new algorithms that deliver innovative functionality is a necessary investment of human resources and as much art as it is science. However, given the huge impact this innovation can have on market success, this investment of resources to design new functional algorithms offers an opportunity for a huge return on investment that makes it a high-value-added activity. The key to maximizing this return on investment is finding a design approach that allows the creative new software to be easily and consistently integrated with all of the other software going into the collaboration.

Stepping back and taking a broader view, software from one partner that was not designed to interact with the software from other partners creates integration challenges that drive non-value-added rework for every partner. Defining system-specific interfaces so that all software in the system can interact with the rest of the software in the system is non-value-added work. This concept applies to the source code interfaces between software components inside a single ECU as well as to the network communication interfaces between ECUs in a vehicle system.

This scenario represents a serious challenge to the efficiency of collaborative product development and leads to a strong need for a standardized design for integration approach that the automotive electronics industry can align on. A standard design for integration approach that all partners are already using would allow all software to be quickly and easily inserted into a standardized system framework and would eliminate the need for non-value-added definition of system-specific interfaces.

SOFTWARE RE-USE

As described above, many new and innovative systems are often a new combination of existing technologies. A "design for re-use" approach can realize the highest possible value from the existing technologies brought into the collaboration by each partner. The key to maximizing this value is finding a design approach that allows existing software from all partners, both standardized and pre-standardization, to be easily and consistently re-used with all of the other software going in the collaboration.

Just as a design for integration approach asks software designers to look beyond the bounds of their own software parts, a design for re-use approach requires partners to look beyond the bounds of the current development project. A good design for re-use approach must address several challenges in order to deliver optimal value.

Looking to the past, a design for re-use approach must facilitate the integration of existing legacy software into the new system. Software that was not designed to interact with the software from other partners creates integration challenges that often limit re-use and drive non-value-added rework for every partner. Modifying the designs of existing software parts and designing new software parts so that all software components in the system are explicitly aware of each other and can interface to each other exactly as needed is non-value-added work. Further, such a here-and-now perspective serves only the current collaboration and does nothing to avoid repeating such reworks in future collaborations with different partners.

Looking to the future, a design for re-use approach must ensure that software from the current project will be inherently re-usable in future collaboration projects with different partners. This approach must include all of the legacy software repackaged for the current project as well as all new software developed for the current project. This approach must apply to the source code interfaces between software components inside a single ECU as well as to the network communication interfaces between ECUs in a vehicle system.

This scenario represents a serious challenge to the efficiency of collaborative product development and leads to a strong need for a standardized design for re-use approach that the automotive electronics industry can align on. A standard design for re-use approach that all partners are already using would allow existing software parts from previous projects to be quickly and easily inserted into new systems and would eliminate the need for non-value-added reworks of existing software that implements carry-over functionality needed in the current collaboration and also in future collaborations with different partners.

IMPLEMENTATION SPECIFICATION

Sufficient and accurate specifications for requirements and implementation are key enablers to any software development effort. Not only do these specifications define the product as the vehicle-buying customer will perceive it, they also define the way the product will be implemented.

Where the product is new or innovative, there is little or no existing specification material that can be re-used, making this effort unavoidable. The investment made in this creative development of specification provides a high return on investment as it will have a significant influence on customer acceptance and the market success of the product. In a collaborative development effort, there is often a general consensus on the targets before the specification development even begins. The collaborating partners already have a common virtual concept of the product they want to bring to market.

The time and effort that goes into specifying the software implementation is another matter. The software implementation will not be visible to the end user and the vehicle-buying customer will place no value on it. There are multiple reasonable and viable options for implementation strategy. Which strategy is selected has little impact on the success of the product. (Of course a poor or inappropriate implementation strategy can have disastrous effects, but for the sake of this discussion let's assume that is not a serious risk and all options given serious consideration are reasonably good ones.) As a result, any investment and effort made in the development of a new implementation strategy is potentially wasted and the re-use of an existing strategy offers the highest return on investment.

While the question of specifying an implementation strategy based on re-use may seem like a simple one, in a collaborative scenario the question becomes much more complex. Typically, each collaborating partner has based their previous work on a different implementation specification. In such scenarios re-use becomes a mixed proposition as selecting one strategy over another provides an opportunity for re-use by one partner, but means a potential complete re-design for the other partners. Negotiations on such decisions include both business and technical considerations and can consume time and resources while adding little or no value for the vehicle-buying customer or even for most of the collaborating development partners.

This scenario represents a serious challenge to the efficiency of collaborative product development and leads to a strong need for a standardized implementation specification that the automotive in-vehicle software community can align on. A standard software implementation specification that all partners are already using would eliminate the need to negotiate or make tradeoffs and would save all of the time and effort that goes into making such decisions and would eliminate most of the effort faced by partners forced to change existing implementations in a move to a new implementation specification.

INTER-ORGANIZATIONAL COMMUNICATION

Solid inter-organizational communication between partners is a key to success in any collaboration. Communication between collaborating organizations occurs at multiple levels and may be between humans or between machines. Effective and efficient communication channels are essential at all interfaces.

Human-to-human communication regarding software is filled with technical details expressed in specialized terminologies. When these detailed technical terms evolve organically inside an organization and a process, they become a local language. Fluency in a local language often requires not just deep technical knowledge, but years of experience in the local organization.

Information sent in the language of one organization may not translate well into the language of a partner organization. While human diligence can significantly reduce the risks of miscommunication due to differences in local technical languages, this approach is not perfect, does not produce consistent results, is effort intensive and adds no value to the development process or the product.

Machine-to-machine communication can be equally challenging. The software development world is filled with a variety of PC-based software development tools from numerous technology vendors. While many of these tools support similar activities, each may have their own data model with different perspectives on the data and distinctive file formats. Collaborating partners using different software development tools from various vendors face problems of not just file format compatibility, but perhaps even significant differences in the way data is organized, navigated, and manipulated.

Manual conversion of data from one tool to another is not a viable approach because of the effort-intensive and error-prone nature of such work. Automated solutions that can re-structure and translate data between specific tools can overcome machine-to-machine communication hurdles, but are costly to develop, add no value to the development process or the product, and must be replaced in future collaborations with different partners that depend on yet another tool.

These human-to-human and machine-to-machine scenarios represent a serious challenge to the efficiency and cost optimization of collaborative product development. These

challenges lead to a strong need for standardized inter-organizational communication. Standard terminology, standard development contexts, standard data models, and standard computer file formats would enable more direct, more effective, and more efficient communication between organizations with an improved return on investment.

AUTOSAR AS A KEY ENABLER TO COLLABORATIVE SOFTWARE DEVELOPMENT

The rapid evolution, growth, and complexity of software in automotive electronic systems have become a significant challenge for even the most capable organizations to manage. In response, many automotive industry participants - OEM and supply chain alike - have joined in a collaborative project to develop a set of standards that are intended to address these common challenges and to collectively satisfy a variety of mutually-desirable objectives. This collaborative project is called AUTOSAR - AUTomotive Open System Architecture. See www.autosar.org [1] for more information on the AUTOSAR consortium and project objectives.

In order for the consortium to keep itself on track with all stated objectives, AUTOSAR defines a list of main requirements that the consortium's work products must utilize and conform to. Some of the AUTOSAR requirements target collaborative development explicitly. Other AUTOSAR requirements indirectly guide AUTOSAR toward the support of collaborative development. For the purposes of this paper, an investigation of the AUTOSAR requirements, project objectives [2], and technical strategy will serve as evidence that AUTOSAR is a key enabler for collaborative software development.

One AUTOSAR Main Requirement stands apart from the others as testimony to the suitability of AUTOSAR as enabler for collaborative software development -

· AUTOSAR Main Requirement 300 - AUTOSAR shall support work-share in large inter-company development groups

Collaboration could well be defined as “work-share in large inter-company development groups.” Main Requirement 300 ensures that AUTOSAR is built from the ground up with collaborative development in mind.

How Main Requirement 300 is met is better understood by considering how other AUTOSAR requirements are aligned with the collaborative product development concepts that permeate the automotive electronics industry and directly target the challenges to collaborative software development that are described by this paper:

• standardized design for integration and software re-use

• standardized implementation specification

• standardized inter-organizational communication

An exploration of each of these enablers will demonstrate how AUTOSAR requirements drive the AUTOSAR standard to not only enable collaborative software development, but make it a robust and repeatable process that consistently delivers the value every partner aims to create through collaboration.

STANDARDIZED DESIGN FOR INTEGRATION AND SOFTWARE RE-USE

A standardized design for integration and software re-use approach is necessary to enable collaborative product development. AUTOSAR is a standardized design for integration and software re-use approach that allows different collaborating partners to work nearly independently and in parallel and still arrive at the system integration phase of the project with software that can be quickly and reliably integrated with the software contributions from the other partners.

Evidence of the suitability of the AUTOSAR standard as a design for integration and software re-use approach for collaborative software development is seen in the following AUTOSAR requirements:

· AUTOSAR Main Requirement 80 - AUTOSAR shall ease the reusability of software and its concepts and implementations

As described earlier, many collaboratively developed products are a union of existing products whose combined implementation provides value higher than that of the sum of the individual parts. The AUTOSAR specification is designed to ensure that existing software can be re-used in a new project like a collaboration based on the union of existing products.

· AUTOSAR Main Requirement 160 - AUTOSAR shall provide a functional interface view of the entire system

Software integration is all about aligning interfaces between software components. The AUTOSAR functional interface view of the entire system ensures that all interfaces between all software components are comprehended and pre-aligned. In a collaborative development scenario, this means that the interfaces for every software component provided by every partner are pre-aligned and ready for integration.

· AUTOSAR Main Requirement 70 - AUTOSAR shall provide complete interfaces to application software and basic software modules

Main Requirement 70 ensures the AUTOSAR specification covers how all software components, including those supporting the ECU's unique application functionality, are connected to each other. As described earlier in this paper, it is the joining of existing software functionality that is a key enabler to collaborative product development. AUTOSAR directly addresses this need by defining how all application software, including new innovative application functionality, is integrated together.

· AUTOSAR Main Requirement 50 - AUTOSAR shall support inter- and intra-ECU-communication mechanisms with high reliability

The integrated results of any collaboration can only be as good as the performance of the interfaces between the software contributions from each partner. A good design approach ensures that the interfaces for each software component not only meet the performance needs of the software component, but also the performance needs of the software components on the other side of the interfaces. The AUTOSAR standard ensures that each of these interfaces will support the exchange of information needed by all software components at both the component level and at the vehicle system level.

· AUTOSAR Main Requirement 120 - AUTOSAR shall provide means to clearly check the conformance of an AUTOSAR-implementation with its AUTOSAR-specification

Software integration is a two-step process. The first step is to bring software components together and connect them through their common interfaces. The second step is checking the integrated software system for compliance with the specification and testing the system performance against functional requirements. Having a means to consistently and reliably perform such checks and tests is essential to a successful integration.

• AUTOSAR Main Requirement 130 - AUTOSAR shall provide an abstraction of the application software from hardware

• AUTOSAR Main Requirement 140 - AUTOSAR shall provide an independency of application software from in-vehicle communication technologies

• AUTOSAR Main Requirement 141 - AUTOSAR should provide an independency of application software from operating systems

• AUTOSAR Main Requirement 150 - AUTOSAR shall provide mechanisms and methods to encapsulate application software from the infrastructure

There is a natural expectation for collaboration between suppliers of basic AUTOSAR software and creators of application software. AUTOSAR structures software in such a way that it can be integrated into not just a specific target system, but into any target system. When working in the context of a single specific system, it is easy to allow application software to become inexorably entwined with the specific system infrastructure at hand. Main requirements 130, 140, 141, and 150 ensure that AUTOSAR design concepts keep application software and local system infrastructure from becoming inseparable and promote portability (re-use) of software from not just one system to another but from one collaboration to another.

• AUTOSAR Main Requirement 190 - AUTOSAR shall provide interoperability with legacy software

• AUTOSAR Main Requirement 210 - AUTOSAR shall provide means to integrate AUTOSAR ECU's in non-AUTOSAR networks

Unless AUTOSAR-based implementations become ubiquitous, there will always be a need to re-use and integrate non-AUTOSAR-based software with AUTOSAR-based software. This is especially true in a collaboration scenario where one partner has already adopted an AUTOSAR implementation strategy while another partner has not.

As described here, many of the AUTOSAR Main Requirements drive the AUTOSAR standard to provide a design for integration approach is aligned with collaborative software development needs and meets the challenges related to design for integration described by this paper.

STANDARIZED IMPLEMENTATION SPECIFICATION

A standardized implementation specification is required to enable collaborative product development. AUTOSAR is a standardized specification for an implementation framework. This framework is generic and does not explicitly define the implementation of any single ECU. Rather the framework describes in detail an implementation super-set of functionality that covers all foreseeable automotive software control system needs. From this framework, chunks of specification can be extracted, configured into a concrete design for a specific system, and built to by all partners in a project.

Evidence of the suitability of the AUTOSAR standard as an implementation specification for collaborative software development is provided in the following AUTOSAR requirements:

· AUTOSAR Main Requirement 160 - AUTOSAR shall provide a functional interface view of the entire system

As noted above, the AUTOSAR standard specifies a complete software implementation framework. This structure

covers the entire system and defines the boundaries between all software components. This coverage is essential to enable re-use of the specification of discrete software components across different projects with different partners.

· AUTOSAR Main Requirement 320 - Definitions of relations between SW components are exhaustive and formal

The AUTOSAR standard provides a complete implementation specification [1]. Comprised of over 100 documents, the standard details both common functional requirements shared by all ECUs in a vehicle system as well as the software architecture for complete ECU systems and networked multi-ECU systems. This breadth of scope supports collaboration both inside a single ECU as well as between ECUs in a system or network.

· AUTOSAR Project Objective 4 - Implementation and standardization of basic system functions as an OEM wide Standard Core Solution

For basic functionality common across all ECUs in a system or network, PO4 requires AUTOSAR to not only provide a framework, but an exhaustive list of specific software components that address all basic ECU functionality required to support the ECU application. The AUTOSAR standard provides specifications for more than 60 such basic software components covering operating system, operating state management, network communication, hardware and I/O abstraction, and timers. Such an implementation specification completely eliminates the need to create any new specifications for basic common functionality, leaving the collaborating partners free to focus on the creative specification of the new application-specific functionality the collaboration is intended to deliver.

· AUTOSAR Main Requirement 70 - AUTOSAR shall provide complete interfaces to application software and basic software modules

While Project Objective 4 ensures that all basic (non-application) software is explicitly defined, Main Requirement 70 ensures the specification covers how all software components, including those supporting the ECU's unique application functionality, are connected to each other. The AUTOSAR standard provides specifications for more than 300 interfaces to connect software components. This comprehensive list of interfaces assures all collaborating partners that the interfaces needed are available and ready for re-use.

· AUTOSAR Main Requirement 50 - AUTOSAR shall support inter- and intra-ECU-communication mechanisms with high reliability

Main Requirement 50 drives AUTOSAR to look beyond the bounds of a single ECU. For collaborations at the vehicle system level, inter-ECU communications to support functions that span multiple ECUs are supported by the standard.

· AUTOSAR Main Requirement 110 - AUTOSAR shall provide a software architecture that is applicable across different functional domains

Main Requirement 110 ensures that the AUTOSAR specification is suitable for use in a variety of applications. This broad application support is important for collaborative developments that are creating innovative new functionality that does not necessarily fit into any traditional category.

· AUTOSAR Main Requirement 100 - All AUTOSAR standard software functions shall be standardized across OEM- and Supplier

Collaborative software development is frequently done by OEM-supplier partnerships. The AUTOSAR specification is designed to be valid and common for both OEMs and suppliers.

· AUTOSAR Main Requirement 120 - AUTOSAR shall provide means to clearly check the conformance of an AUTOSAR-implementation with its AUTOSAR-specification

A specification is only as valuable as it is verifiable. In collaborative development projects, implementation issues most often occur at the boundaries where the software of one partner meets the software of another. When these implementation issues arise, the AUTOSAR specification defines a means to check each component and see which partner(s) are out of spec and need to make adjustments.

· AUTOSAR Main Requirement 190 - AUTOSAR shall provide interoperability with legacy software

Technical standards tend to gain acceptance incrementally over a period of years. Until a standard specification reaches universal acceptance and application, there will be non-standard legacy software in the mix that was created before the adoption of the standard. The AUTOSAR specification supports the ability to incorporate pre-AUTOSAR legacy software into an AUTOSAR implementation. This is an important benefit for collaborating partners that want to re-use pre-AUTOSAR software in a new AUTOSAR-based collaboration project.

· AUTOSAR Main Requirement 270 - Releases of AUTOSAR shall be forward and backward compatible

As a standard evolves and new versions are released for use, the concept of legacy software grows to include not just pre-

standard implementations, but also standard implementations based on earlier versions of the current standard. The forward/backward compatibility means that as the AUTOSAR specification evolves, newer versions will not preclude the re-use of software built to previous versions of the specification.

Clearly, many of the AUTOSAR Main Requirements drive the AUTOSAR standard to include a standardized implementation specification that is aligned with collaborative software development needs and meets the challenges related to implementation specification described by this paper.

STANDARDIZED INTER-ORGANIZATIONAL COMMUNICATION

Standardized inter-organizational communications are essential in a collaborative development project. Collaborating partners must share a common dialect, protocol, and context to facilitate efficient and effective human-to-human and machine-to-machine communication. AUTOSAR provides standardized communication for both humans and machines to allow different collaborating partners to communicate effectively in any context.

Evidence of the suitability of the AUTOSAR standard as a basis for inter-organization communication for collaborative software is seen in the following aspects of the AUTOSAR standard:

· AUTOSAR Main Requirement 320 - Definitions of relations between SW components are exhaustive and formal

AUTOSAR provides not only an exhaustive text specification of all relevant concepts, but also a glossary that defines over 200 technical terms. This documentation provides a powerful standard lexicon sufficient for all partners in any collaboration. This standardized dialect and system specification eliminate the need for local dialects that can inhibit human-to-human communication between partnering organizations.

• AUTOSAR Main Requirement 100 - All AUTOSAR standard software functions shall be standardized across OEM- and Supplier

• AUTOSAR Main Requirement 110 - AUTOSAR shall provide a software architecture that is applicable across different functional domains

Main Requirement 100 and Main Requirement 110 ensure that the AUTOSAR lexicon is sufficient to support the needs of OEMs and suppliers alike across different functional domains separated along natural boundaries in vehicle functionality like powertrain, chassis, body, and infotainment.

· AUTOSAR Main Requirement 370 - AUTOSAR method shall provide a predefinition of typical roles in work-share method

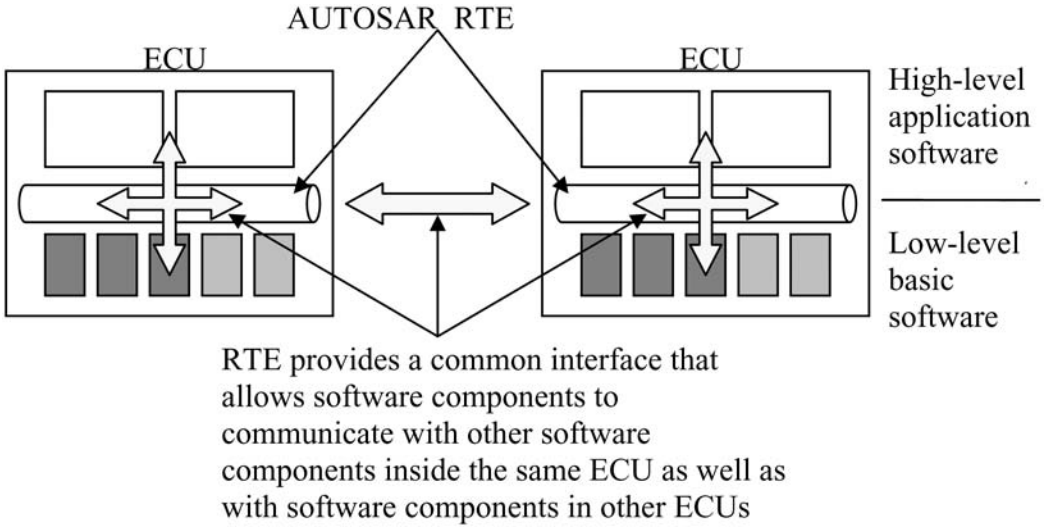
AUTOSAR not only provides a lexicon of terms for the relevant software development concepts, it also provides a concept for a working environment that includes both a development process framework and descriptions of the roles that humans take when working in the AUTOSAR environment. AUTOSAR documentation describes a methodology for developing AUTOSAR-based software. Templates are provided for system architecture, system design, and software component descriptions. The development framework and descriptions for roles create a standard protocol for how collaborating partners should work together.

· AUTOSAR Main Requirement 90 - Tool-chains, which are developed or adapted to AUTOSAR, must be compatible with the AUTOSAR methodology

In a collaborative development project, machine-to-machine communication plays just as important a role as human-to-human communication. Machine-to-machine communication occurs in the form of electronic data exchange between different PC-based software development tools. The AUTOSAR standard provides everything needed to ensure different software development tools from different vendors can effectively exchange data without human conversion.

The AUTOSAR standard provides a data model that captures the entire software architecture. The AUTOSAR standard includes an XML electronic file format that can capture all AUTOSAR data in a machine readable/writeable form. The AUTOSAR development framework and user roles point to natural use cases common to the application of different software development tools. In combination, the AUTOSAR data model, XML file format, development framework, and user roles create an opportunity for different tool vendors to provide commercial off-the-shelf development tools that are largely interoperable right out of the box. This open market approach to tools allows collaborating partners to select each other without concern over differences in tools and without need to change tools in order to work with new partners.

The AUTOSAR consortium includes prominent members from the tools sector. Tool vendors are now providing new AUTOSAR-compliant tools to the industry as well as adding AUTOSAR support to new versions of existing tools. The AUTOSAR-enabled interoperability of these tools is visible in the collaborative agreements that vendors of complementary tools are forming to deliver multi-tool solutions that address the broader needs of their common automotive engineering customers.



Finally, the AUTOSAR consortium provides tutorial companion documents that help software developers reach a common understanding of how to go about their work in an AUTOSAR environment. These aspects of AUTOSAR bring the standard to life and enable standardized inter-organizational communication between potential partners - even before they might consider engaging each other in a collaborative development project.

AUTOSAR TECHNICAL STRATEGY

As can be seen above, the AUTOSAR requirements ensure the AUTOSAR framework enables collaborative software development. A look at the AUTOSAR technical strategy shows how these requirements have shaped the framework and what this collaborative environment looks like.

As noted earlier, collaboration takes place at both the vehicle system level and the component level. At the vehicle system level, the vehicle OEM acts as a collaboration agent that works to seamlessly align the communication interfaces between the ECUs and integrate them into cohesive networked systems. At the component level, the ECU manufacturer (and in some cases also the vehicle OEM) acts as the collaboration agent that seamlessly aligns the interfaces between the software components inside the ECU and integrates them into the ECU system.

Collaboration at both the vehicle system level and at the component level is facilitated by a central implementation concept in AUTOSAR called the Run Time Environment (RTE). The RTE acts as a virtual bus that manages all interactions between application software components. Application software components communicate through the RTE to other software components, whether these other software components are inside the same ECU or in another ECU connected by a communication network.

By providing a standard interface that software components interact through, the RTE abstracts away the location of other software components. Software components that interface through the RTE are highly portable and can be readily relocated to other ECUs in the networked vehicle system as well as re-used in other ECUs for different vehicle development projects.

SUMMARY/CONCLUSIONS

As can be seen in many aspects of the development of automotive electronics, in-vehicle software is the single biggest and key enabler for collaborative product development in automotive electronics. Collaborative product development is enabled by software that binds complementary technologies to create new and innovative electronic functionality. Collaboration is achieved at both the vehicle systems level and at the component level through vast amounts of software delivered by dozens of contributing development partners.

Traditional ad-hoc forms of collaborative software development face serious challenges to process efficiency, productivity, and severely limit the value of collaboration. These challenges lead to the need for a standardized design for integration approach, a standardized design for re-use approach, a standardized implementation specification, and standardized inter-organizational communication. AUTOSAR project objectives and main requirements directly address these challenges and have lead to a standardized development framework that enables collaborative software development in several significant ways.

The AUTOSAR framework provides a standardized design for integration and software re-use approach that streamlines the process of aligning and interconnecting the software contributions from each partner in the collaboration as well as

ensuring new software developed in one collaboration is inherently suited for re-use in future collaborations with different partners. The AUTOSAR framework allows software developers to use a proven design that eliminates the non-value-added effort that would otherwise be needed to define software interfaces along the boundaries between the software contributions coming from each partner. The AUTOSAR framework facilitates the re-use of existing pre-standard software while ensuring AUTOSAR-compliant software components will always be ready for re-use in future collaborations.

The AUTOSAR framework provides a standardized implementation specification that predefines the software architecture both inside the ECU and at the vehicle network level. This approach allows software developers to focus on the implementation of new functionality without having to create a specification for how the generic application-independent parts of the software will be structured.

The AUTOSAR framework provides a standardized language for inter-organizational communications that facilitates both human-to-human communication and machine-to-machine communication. This approach allows all partners to work in a common context that does not require any partner to adapt their organization to the language or development tools of other partners.

The end result is that AUTOSAR delivers an optimized software framework that dramatically reduces non-value added activities and leaves collaborating partners free to focus on the business of innovation and bringing their combined products to market as efficiently and reliably as possible both now and in the future.

ACKNOWLEDGEMENTS

Thank you to Robert Miller and Ralf Fritz, of Vector CANtech, Inc., for their wisdom in the brainstorming sessions this paper evolved from and for their review of the manuscript.

REFERENCES

1. www.autosar.org

2. Main Requirements; Document ID 054: AUTOSAR_RS_Main

CONTACT INFORMATION

Mark Jensen - Product Line Manager, Process Tools
Vector CANtech, Inc.
39500 Orchard Hill Place
Novi, MI 48375
mark.jensen@vector.com

Tony Mascolo - President
Vector CANtech, Inc.
39500 Orchard Hill Place
Novi, MI 48375
tony.mascolo@vector.com

DEFINITIONS/ABBREVIATIONS

AUTOSAR
AUTomotive Open System ARchitecture

ECU
electronic control unit

RTE
run-time environment

The Engineering Meetings Board has approved this paper for publication. It has successfully completed SAE's peer review process under the supervision of the session organizer. This process requires a minimum of three (3) reviews by industry experts.

ISSN 0148-7191

doi:10.4271/2010-01-2341

SAE Customer Service:
Tel: 877-606-7323 (inside USA and Canada)
Tel: 724-776-4970 (outside USA)
Fax: 724-776-0790
Email: CustomerService@sae.org
SAE Web Address: http://www.sae.org
Printed in USA