AUTOSAR ECU Development Process Using DaVinci and MICROSAR from Vector

English Translation of a Japanese Technical Article from Mitsubishi Motors Corporation

AUTOSAR is a group paving the way for the standardization of software platforms across Electronic Control Units (ECU). Its activities have gained momentum in recent years. In order to find out how AUTOSAR could best be applied to ECUs, we – Mitsubishi Motors – joined forces with our supplier, Mitsubishi Electric, to test the development process and evaluate the tool chains relevant to the implementation of AUTOSAR software components on ECUs. For evaluation we used the EV-ECU of the i-MiEV electric car. Here is an overview of the project.

AUTOSAR

As the number of on-board computers (ECUs) in automobiles increase along with their functionality, there has been an increase in software implementation. As a result, the Europe-based AUTOSAR has been gaining attention as the leading platform for the standardization of basic software specifications. This standardization covers basic software modules e.g. for I/O access, CAN communication and the operating system. Furthermore, the standardization includes XML formats for exchanging configuration data.

Preparing for AUTOSAR

As it is not yet clear how AUTOSAR will affect the existing ECU development processes and development environments of automobile OEMs and suppliers in Japan, implementing AUTOSAR without careful thought may cause confusion and disruption. This is why we joined forces with Mitsubishi Electric, our ECU supplier, and Vector Japan Co., Ltd. to test the AUTOSAR ECU development process and verify the development environment (tools) as a preparation for implementing AUTOSAR software components. This project also covers evaluation of software reusability with AUTOSAR.

AUTOSAR Evaluation Overview

For this project, we evaluated the EV-ECU, the main control ECU for the i-MiEV electric car. We decided to extract and use a part of Mitsubishi Motor's model-based software from the EV-ECU. To carry out the evaluation, the extracted software was divided and assigned optimally to two ECUs using microcontroller evaluation boards **(Figure 1)**. The AUTOSAR ECUs designed for the purpose of this

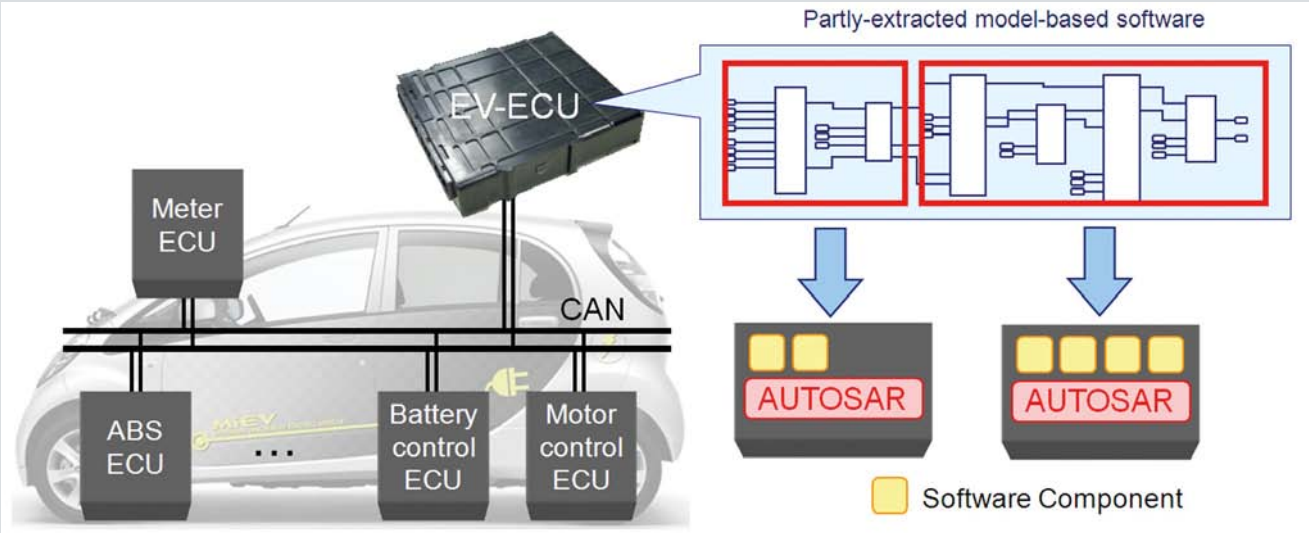

Figure 1: Division of ECU

evaluation and the EV-ECU were then connected using a CAN communication network. **(Figure 2, "Local CAN")**.
To evaluate software reusability, the software assigned to one of the two AUTOSAR ECUs was divided again and applied to another AUTOSAR ECU **(Figure 7)**. This implementation process was compared to the implementation process of commonly reused software, to evaluate software reusability with AUTOSAR.

ECU Development Process Testing

Here is a summary of the tests we carried out with a focus on the ECU development process with AUTOSAR.

> Differences from legacy ECU development processes
> Division of roles between automobile OEMs and suppliers
> Tool chains
> Data definition file format

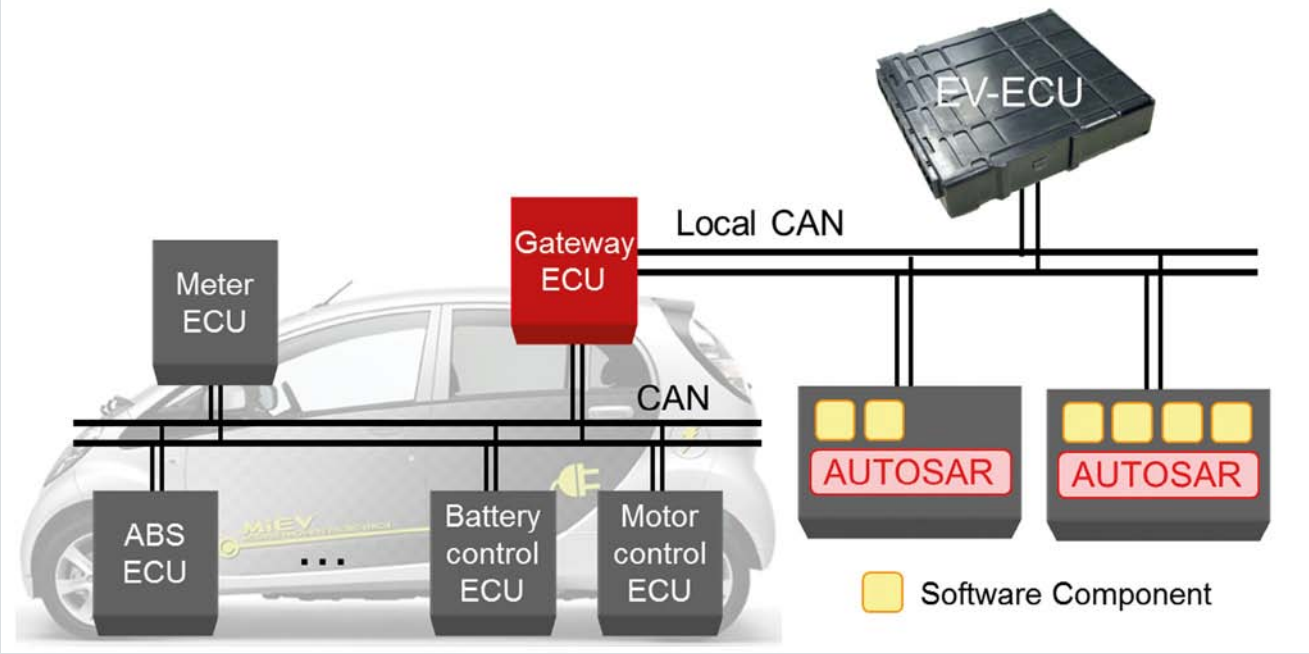

Figure 2: Overview of vehicle network

This project evaluated the processes listed below. We have summarized the details of the workflow and testing criteria, as well as the results for each process. The development and implementation environment can be seen in **table 1**.

> Vehicle system design
> Vehicle network design
> ECU application software design
> ECU software system design
> ECU software detailed design
> Coding/ Implementation

Vehicle System Design

A certain system requirement specification was created for the evaluation project. We designed a system focusing on coordination between the EV-ECU and the extracted software. The evaluation included the allocation of functions to each ECU, the network configuration and choosing suitable AUTOSAR BSW. The system requirement specification has been revised according to the evaluation results. The allocation of functions to each ECU was realized by assigning software components (SWC) using the AUTOSAR method. It depends on the OEM and the project whether to define the SWC and BSW configuration during the vehicle system design, or during ECU design. Since we were dealing with a small-scale system and with BSW provided by the specific software vendor, we chose the former option.

Design tools	
AUTOSAR Evaluation Bundle [Vector]	
Network design	Network Designer
RTE design	DaVinci Developer
OS/BSW design (Configuration)	DaVinci Configurator Pro GENy
Model-based development tool [Mathworks]	
Model design	Simulink®
Code generation	Real-Time Workshop® Embedded Coder™
Other development environment	
Compiler [IAR]	IAR compiler
Microcontroller [Renesas]	R32C
Microcontroller evaluation board [Renesas]	Evaluation board for R32C

Table 1: Development and implementation environment used for evaluation

Vehicle Network Design

On the vehicle network of an i-MiEV, a gateway ECU was set up where the EV-ECU was located, and a local CAN communication network was constructed using the modified EV-ECU and the AUTOSAR ECUs **(Figure 2)**.

We used Network Designer for the CAN communication network design **(Figure 3)**. The CAN communication data was exported as a system description file in ARXML format. As far as the communication network design is concerned, the design method remains unchanged except for the tool and data transfer file format. In addition to the ARXML format, Network Designer also offers various other formats such as DBC and FIBEX which could be used instead of the AUTOSAR format. As network design also takes the entire vehicle into consideration, the process is carried out by the OEM.

ECU Application Software Design

We edited the software extracted from EV-ECU using Simulink®. According to the software model's inputs and outputs, we defined the AUTOSAR compliant ports/interfaces and runnable entities.

In terms of software reusability, the SWC input/output port and data configuration require caution, especially with regard to the collection of data and configuration of the port. If the data does not match other SWC port configurations when reusing SWC, the port configuration will have to be changed **(Figure 4)**. On the other hand, creating a port for each data makes the SWC configuration too complicated and is therefore not recommended. For SWC ports in line to be reused, it is important to review the port configuration beforehand.[1]

Regarding the interaction of tools, it should be taken into consideration that several tools might change the Software Component Description file (SWC Description).[2]

[1] In the latest AUTOSAR release 4.0, these problems are reduced since it is possible to explicitly define the mapping between port data.
[2] Therefore, the tools need to support a roundtrip and must not overwrite the data created by another tool.

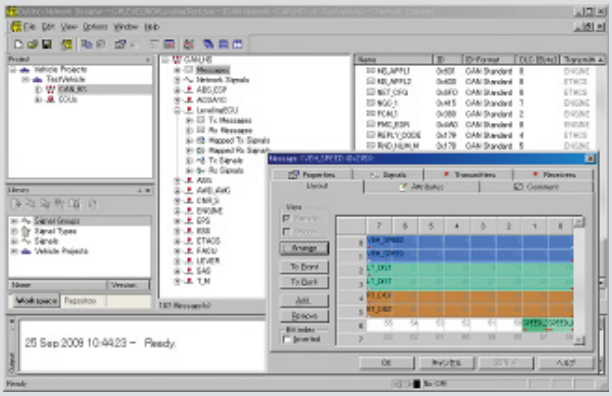

Figure 3: Network Designer™

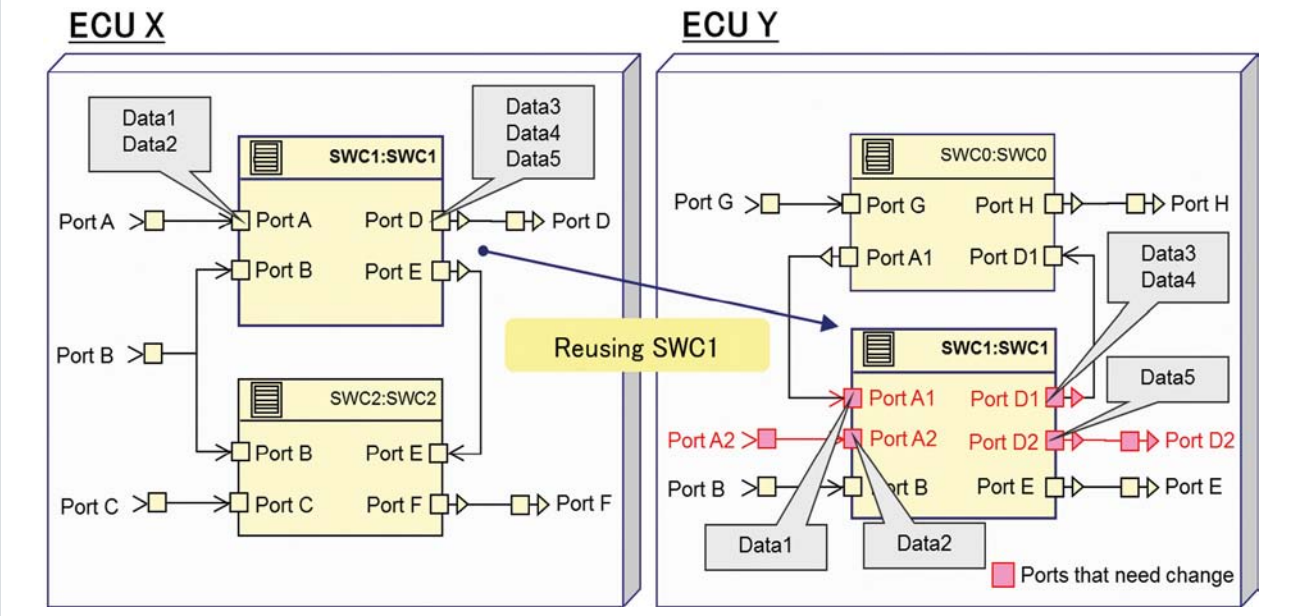

Figure 4: Points to note in terms of SW-C reusability

The model's software source code and the SWC Description file were automatically generated using Real-Time Workshop® Embedded Coder™. In this evaluation project, the ECU application design was carried out by Mitsubishi Motors. For series projects the division of roles should be discussed before design.

ECU Software System Design

The ECU software system design involves the allocation of SWC to ECU and designing the ECU input/output data. We used DaVinci Developer™ **(Figure 5)** for this purpose.
First, the SWC Description file generated from Simulink® was imported into DaVinci Developer and the SWCs were allocated to each ECU. Next, the System Description file generated from the network design was imported and the CAN communication data and SWC input/output data were mapped. DaVinci Developer features an automatic data mapping function, which is extremely useful when dealing with large amount of data. The data configured by DaVinci Developer for each ECU is exported as an "ECU Extract of System Configuration" file in ARXML format.
As the network design is frequently being updated with regard to the ECU development process, it is important for the Runtime Environment (RTE) design tool that the System Description can be simply updated and that the content of such updates can be easily verified.
The ECU software system design was carried out by Mitsubishi Motors. For small-scale development projects (such as closed projects with ECU units), it may be efficient for the supplier to carry out the design. However when designing software partially or designing a cooperative control with multiple ECUs, it is better for the OEM to do the design. By mapping the communication and SWC data, the OEM can verify the consistency of the data in advance. This is useful, as inconsistency of data can lead to unnecessary work between the OEM and supplier.

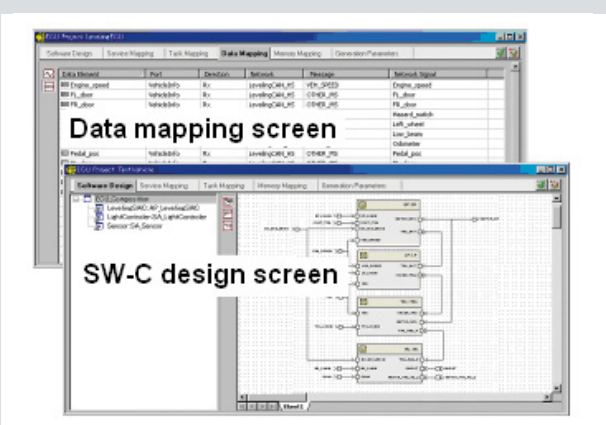

Figure 5: DaVinci Developer™

ECU Software Detailed Design

ECU software detailed design involves the configuration of ECU device drivers and design of OS and ECU basic functions (CAN communication, error diagnostic functions, etc). The design uses the ARXML-format "ECU Extract of System Configuration" files gene-rated above.

Figure 6: Vehicle testing

Specific design details include the allocation of Runnable Entity to OS tasks and the configuration of the relevant BSW parameters. We carried out the configuration in this case using DaVinci Developer, GENy and DaVinci Configurator Pro. The ARXML-format "ECU Configuration Description" file describes the specific configuration of the basic software modules and is shared between the involved tools. As ECU software detailed design is dependent on the hardware configuration, it is carried out by the supplier.

Coding/Implementation

The configuration code for the basic software modules is automatically generated by DaVinci Configurator Pro and GENy. This code is compiled along with the static code of the basic software into the executable code, which is then written to ROM. We used an IAR compiler for compiling. In the past, the codes were generated manually by hand coding. In contrast, AUTOSAR enables the usage of configuration tools and automatic code generators for coding. As is the case with legacy methods, coding/ implementation is carried out by the supplier.

Functional Testing

We equipped an i-MiEV with the AUTOSAR ECUs we designed and ran vehicle tests **(Figure 6)**. As the functions formerly processed by one ECU are now distributed to several ECUs, there was some delay in communication between the ECUs. Nevertheless, we were able to onfirm that the car was controlled correctly by the AUTOSAR ECUs.

Testing Reusability

We then tested the reusability of SWCs. By assigning the SWCs to an additional ECU (they were assigned to two ECUs during ECU development process testing) and changing the assignment pattern **(Figure 7)**, we estimated the process and man-hours needed in porting the SWCs. We were then able to compare the legacy design and the design according to the AUTOSAR method.
Here are the tasks involved in porting the SWC:

> Redesigning the network
> Assigning SWCs to ECUs
> ECU software detailed design

Redesigning the network is necessary for both legacy and AUTOSAR methods and the processes do not differ by much. The application software and SWC interface require no modification in porting the SWCs. Assigning SWCs to

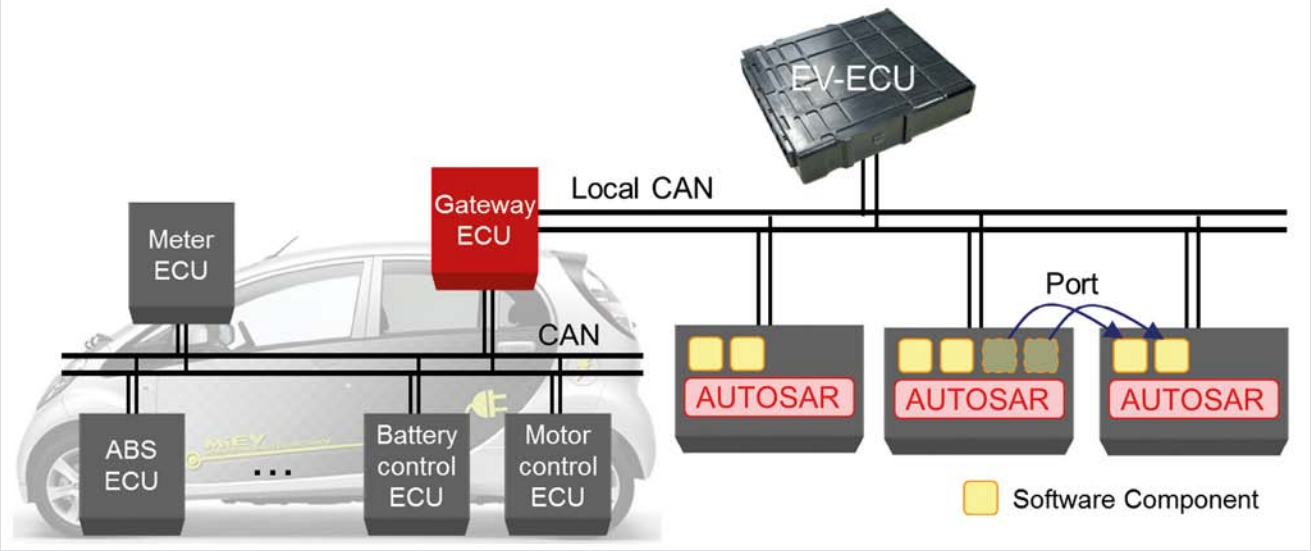

Figure 7: Evaluating reusability and ease of porting

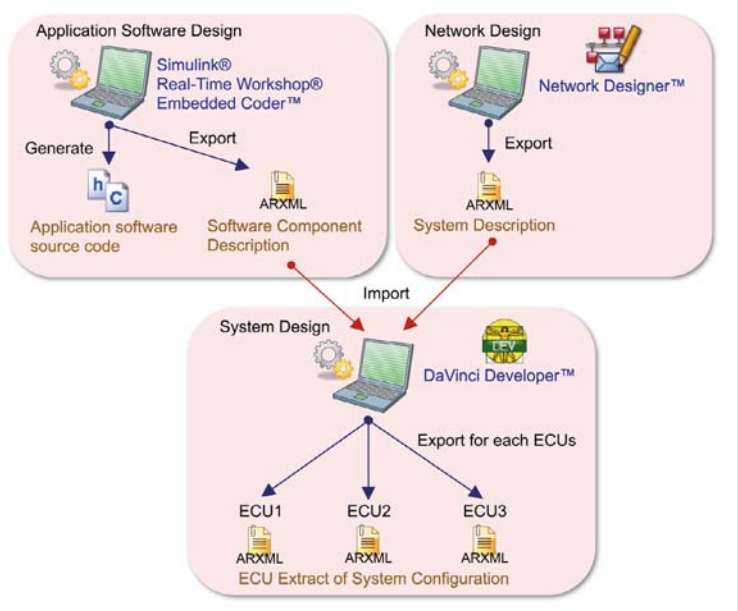

Figure 8: ECU development process with AUTOSAR (Part 1)

ECUs and data mapping with CAN communication data are both easy to do by using the tools. This has not been the case until now: adapting application software interfaces usually took much time and effort.

Regarding the detailed design of ECU software, it is possible to reduce the configuration settings when carrying over parts of the existing configurations. As a result, there is a great potential for reducing the time and effort that usually goes into adapting interfaces and designing middleware.

Evaluating Tool Chains

As tool chains influence the quality of the software greatly, it is important to discuss issues thoroughly with the supplier. Considering the tool dependence, it makes sense to use tools from the same vendor throughout the entire development process. The tool chains evaluated in this project can be seen in **figures 8 and 9**. The pink areas show the work carried out by the OEM and the light blue areas indicate the work carried out by the supplier.

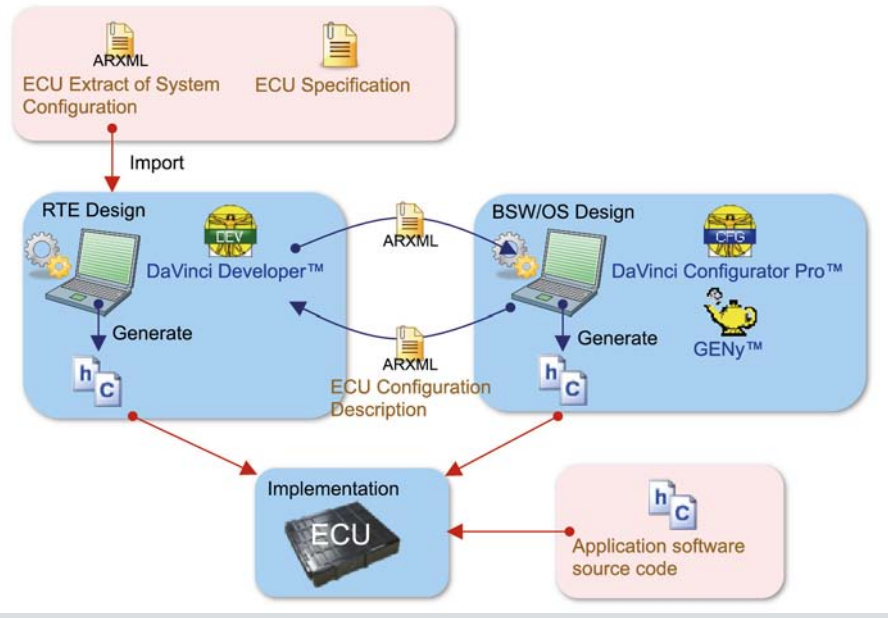

Figure 9: ECU development process with AUTOSAR (Part 2)

From network design to AUTOSAR RTE/BSW/OS detailed design and code generation, Vector offers tools encompassing the entire process. As the tool chain is pre-established, there is consistency and uniformity in the data throughout the process, increasing reliability in development.
The tools could be improved by expanding the compatibility of data exchanged with the most commonly used tools from other companies. As model-based development with AUTOSAR has proven to be a valid development method for the future, we hope this issue will be prioritized.
Also, GUI-based software design is efficient in the design phase, but it is not suitable for comparing and verifying the design data. An additional function to export the data designed by each tool in table format would be desirable. The process of testing the communication signal's configuration parameters and the transmission and reception of the signal was very complicated. This process would be much more efficient if the data could be exported in table format.[3]

Summary

By testing the ECU development process using AUTOSAR, we were able to gain an understanding of the workflow and division of roles. We were also able to clarify the necessary mutual consultation between OEMs and suppliers for each process. With AUTOSAR, the OEM can take over the designing process usually associated with the supplier side. Furthermore, there is room for optimization in overlapping areas such as ECU software system design, where both OEM and supplier should be more flexible in the future.
Regarding reusability, we found that this goal can be easier reached with AUTOSAR than with legacy methods. However, we focused solely on the sender-receiver port format for the SWC input/output, therefore the reusability for server-client port formats still needs to be evaluated.
Through the evaluation project, we were able to establish the technical details to some extent. In the future, we will actively exchange information and opinions with suppliers and tool software vendors to put in place a framework for the introduction of AUTOSAR at Mitsubishi Motors, from the introduction process to implementation.

Afterword

We received great support for this evaluation project. Compilers were kindly provided by IAR Systems, and microcontrollers by Renesas Electronics Corporation. Though we were not able to realize their offer for this evaluation, an offer from Fujitsu Semiconductor Limited for collaboration was greatly appreciated. And of course, Mitsubishi Electric's cooperation and Vector Japan's tool software were invaluable. We'd like to take this opportunity to thank everyone for their support.

Translation of a publication in the Japanese Vector Journal

Image rights:
All figures: Mitsubishi Motors Corporation

Literature:
www.autosar.org

Links:
Homepage Vector: www.vector.com
Information on Vector's AUTOSAR products:
www.vector.com/autosar

Author: Yuichi Kamei
Mitsubishi Motors Corporation
Electronic Control System Development
Electronics Engineering Dept.
Development Engineering Office

[3] In the latest generation of the tool chain, Vector provides according functions to compare the design data.