

Model-Based Development Technologies to Support Increasingly Complex Power Train Systems

Hidekazu ARAKI*
Masayuki FUNAKOSHI*

Kanako SAITO*
Kensuke YAMAMOTO*

Ryosuke OHGUCHI*

ABSTRACT

In order to support power train systems as they become increasingly complex, as well as to raise the efficiency and quality of development, it is useful to (1) clearly define requirements and verification items, (2) structure requirements and clearly define interfaces, and (3) assure the traceability of requirements.

The present research project used Systems Modeling Language, which is a semi-formal notation. By formulating the project's own original guidelines, the above three items were achieved.

Requirement models written in Systems Modeling Language are also published to suppliers as requirement specifications for the platform area of the power train electronic control unit. Therefore the tasks involved in efficiently and accurately verifying not only the validity of the requirements, but also whether or not problems exist in how they are written as requirement specifications, were defined and applied in parts to the development of 2015 model year mass-production models.

1. Introduction

Automobile power train control systems have tended to increase in scale and complexity in recent years. The following initiatives are being pursued in the automobile industry as a whole, as manufacturers approach a balance between increased development efficiency and quality.

The first, as shown in Fig. 1, is an initiative to achieve control development that results in less reworking. This approach here is to confirm the validity of control specifications at the specification design stage of the power train electronic control unit (PT-ECU) development process. After that, the work moves on to the implementation process, thus achieving control development that results in fewer returns. In order to confirm the validity of control specifications at the specification design stage, the control requirements must be formed into a hierarchy, as in Fig. 2, and the validity of the control requirements must be verified at each level.

The second is the initiative to make use of simulations to confirm the validity of the control specifications at the specification design stage. In order to implement simulations at the specification design stage before software is created, it is necessary to model the controlled objects and control algorithms. It is also essential to assure traceability among requirements, verification

items, and control algorithms, which involves ascertaining what requirements a control algorithm is tied to, and determining which verification items need to be satisfied in order to judge it valid.

The third is the initiative to design control software in a form that is in conformity with the Automotive Open System Architecture (AUTOSAR)⁽¹⁾ standard specification for automotive software. Together with the software architecture shown in Fig. 3, the interface between architectures is also specified in AUTOSAR. By designing software architectures and interfaces

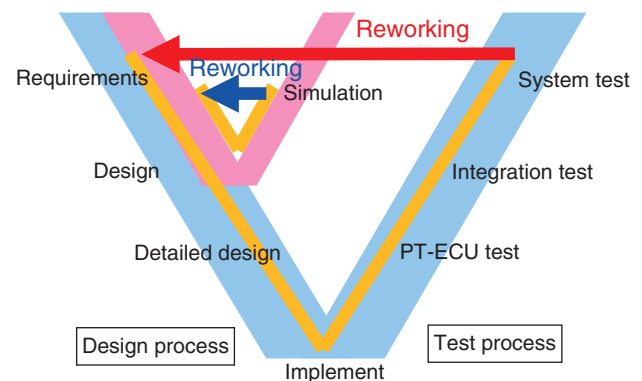


Fig. 1 V-model

* Automobile R&D Center

between architectures in forms that are in conformity with the standard, the control software can be made more reusable. However, there are differences between Honda's existing software architecture and AUTOSAR software architecture. As a result of this, when PT-ECU suppliers implement software in conformity with AUTOSAR, the large amount of time taken to bring the Honda software architecture and AUTOSAR standards into compatibility became an issue. It was reasoned that this issue could be effectively addressed by structuring requirements in light of the AUTOSAR standards and deciding on an interface between architectures.

The present research project took the initiatives described above into consideration and examined the creation of the methods described below as a means of heightening the reusability of control software in development of mass-production models from the 2016 model year on, and achieving control development with less reworking.

- (1) Methods for placing control requirements in a hierarchy and validating them using Systems Modeling Language (SysML)
- (2) Method for confirming the validity of requirements by Model in the Loop Simulation (MILS)
- (3) Method for structuring software that takes AUTOSAR standards into account

2. Construction of a Method for Placing Control Requirements into a Hierarchy and a Method for Confirming Validity of Control Requirements Using SysML

SysML is a modeling language that demonstrates its effectiveness when bringing a system to realization. Since it can be applied to the development of system specifications, for analysis and design, and to the validation and verification of a system, it is widely used today in the automotive, aerospace, telecommunications, and other such fields⁽²⁾.

In the present research, methods using SysML to place control requirements into a hierarchy and to confirm the validity of control requirements were created for use with the control requirements included in the platform area in Fig. 4.

2.1. Method for Placing Control Requirements into a Hierarchy Using SysML

In forming the hierarchy in Fig. 2, the levels were defined as follows.

Level 1 is the level where the primary control requirements of the functions that are to be achieved become clearly defined, together with the interfaces between those functions and other functions. Even if such

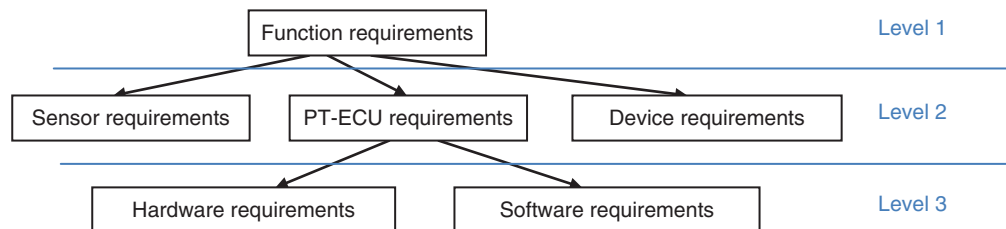
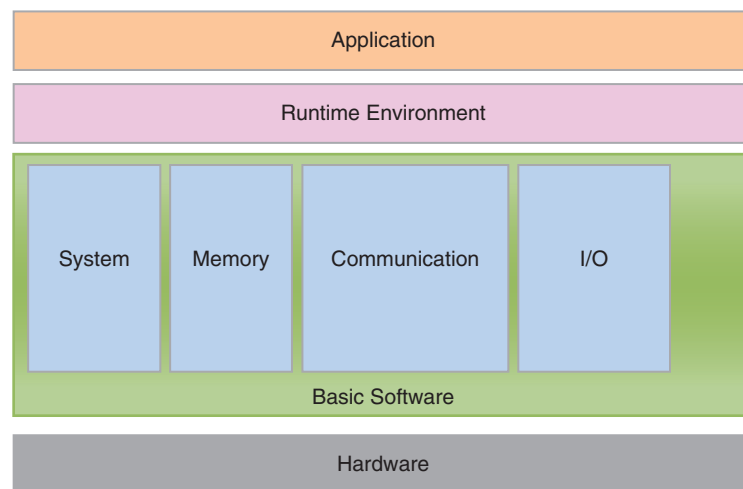


Fig. 2 Hierarchy of control requirements



I/O: Input Output

Fig. 3 AUTOSAR software architecture

component elements as the PT-ECU, sensors, and devices are reconsidered, the control requirements and interfaces at this level do not change.

Level 2 is the level where it is clearly defined what PT-ECU, sensors, and devices are to be used and how they are to detect and operate. At this level, the control requirements to be achieved by the PT-ECU, the control requirements to be achieved by the sensors, and the control requirements to be achieved by the devices are clearly defined. The interface between the PT-ECU and the sensors and devices is also clearly defined at this level.

Level 3 is the level where the control requirements to be achieved in software and the control requirements to be achieved in hardware, as well as the data that is handled in software and hardware transactions, are all finally determined.

Taking the ease of verification of each level into consideration, the structure of each level is determined so as to have the minimum number of output signals. In Fig. 5, for example, Structure A has four inputs. When each input is expressed as ON or OFF, the test pattern exists in $2^4=16$ varieties. If Structure A can be divided into Structure A-1 and Structure A-2, then the test pattern can be reduced to $2^2 \times 2=8$ varieties, making verification easier.

The control requirements that are assigned to each structure have the operation requirements for the structure stated in natural language. The seven category types shown in Table 1 were defined so that the control requirements can be isolated exhaustively.

Table 1 Category types

No.	Category type
1	Operation
2	Safety
3	Market value
4	Regulation
5	Design policy
6	Diagnosis
7	Quality

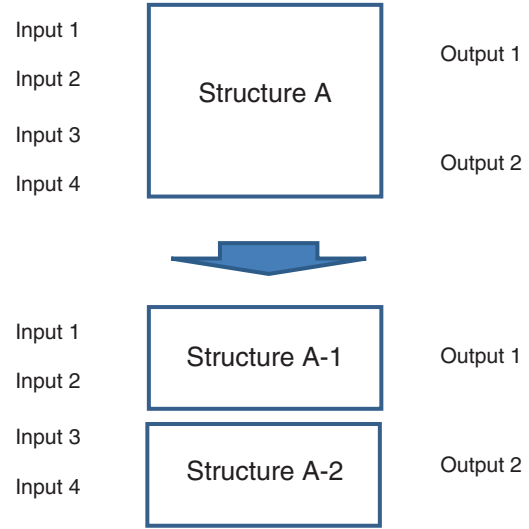


Fig. 5 Division of structure

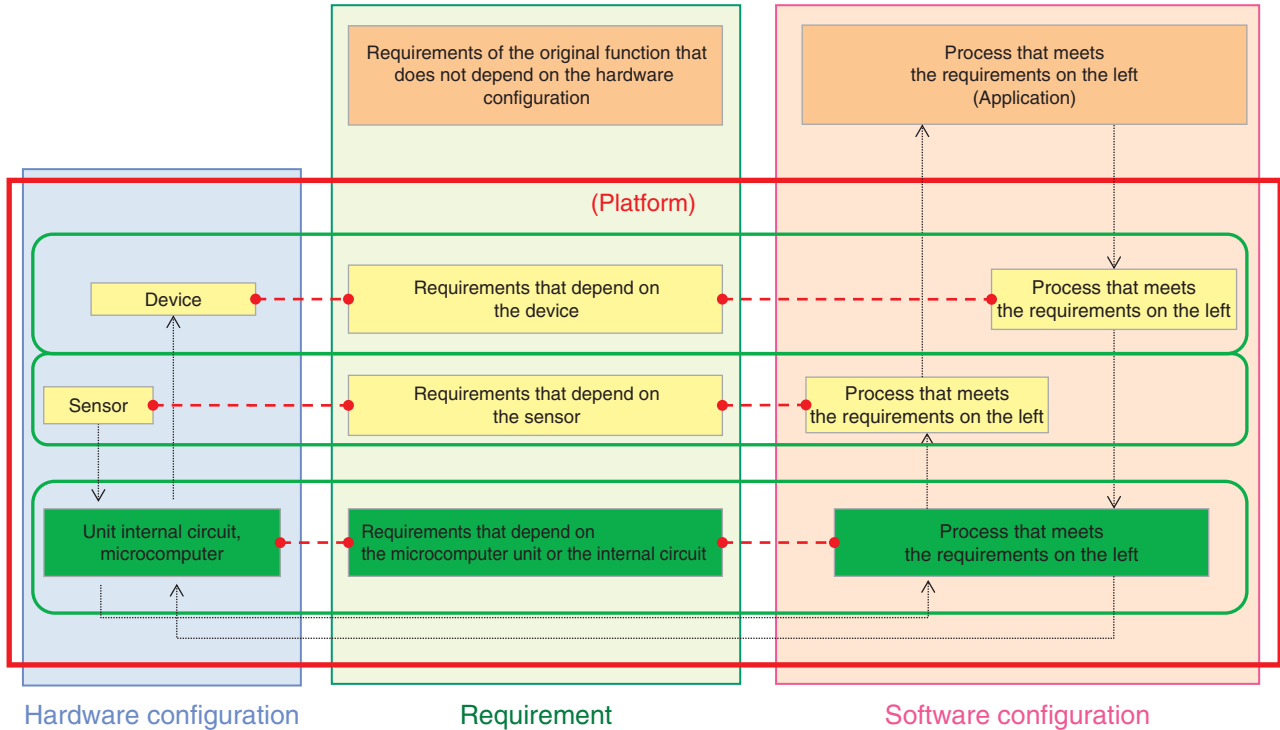


Fig. 4 Platform area

Table 2 Listed items of verification items

No.	Item	Provisions
1	Preconditions of tests	Required test conditions for verification
2	Parameters for test condition	Parameters and their values that correspond to the required test conditions for verification
3	Pre-state	State of the function before testing
4	Operating parameters and volume	Parameters that trigger transition to post-state, and the volume to operate
5	Post-state	State of function after testing
6	Measurement parameters	Measurement parameters that are subject to verification
7	Judgment criteria	Expected value of measurement parameters

2.2. Method for Confirming the Validity of Control Requirements

In order to clarify the criteria for judging whether or not control requirements are being met, one or more verification items are created for each control requirement. The verification items are used as judgment criteria for confirming the validity of the control specifications by means of the MILS or prototype at the specification design stage. In addition to that, they are used as judgment criteria for conformity verification at the ECU testing stage. Anticipating that verification will be automated in the future, the seven items shown in Table 2 were specified to be listed when creating verification items.

Given the above, the diagrams shown in sections 2.2.1 to 2.2.4 are created for each level.

2.2.1. Creation of criterional requirement diagrams

As shown in Fig. 6, the criterional requirement diagram (CRD) records the structure, control requirements, and verification items together with their respective traceability information.

2.2.2. Creation of block diagram

As shown in Fig. 7, the block diagram (BD) records

the structures included in that level together with traceability information in relation to upper-level structures. Consequently, a BD is not created for Level 1, which is the uppermost level.

2.2.3. Creation of internal block diagram

As shown in Fig. 8, the internal block diagram (IBD) records the flow of data between lower-level structures.

This approach facilitates the distinction between a variable that concludes within a function and an interface with another function during the influence analysis and safety analysis that are conducted when specifications are changed.

2.2.4. Creation of requirement diagram

As shown in Fig. 9, the requirement diagram (RD) records the traceability information in relation to control requirements included in that level and upper-level control requirements. Consequently, an RD is not created for Level 1, which is the uppermost level.

The definitions for placement into a hierarchy, category types for extraction of requirements, items listed when creating verification items, and method of writing the diagram are determined. The traceability of requirements

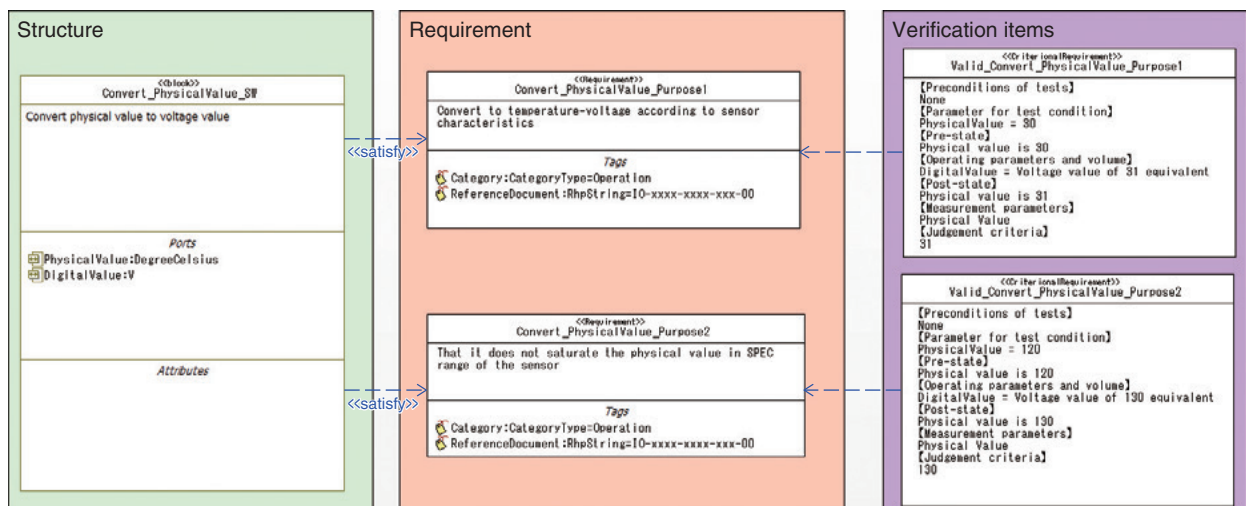


Fig. 6 Criterional requirement diagram

between levels, traceability of structures between levels, and the traceability information for structures, requirements, and verification items at each level are clearly defined. This enables validation of the control requirements.

The present research aimed to enable application of control requirement placement in a hierarchy and

confirmation of control requirement validity to 2016 model year mass-production models by using content written in SysML as system design requirement specifications. To that end, it carried out anticipatory application to 2015 model year mass-production models, identified issues, and implemented measures to address them.

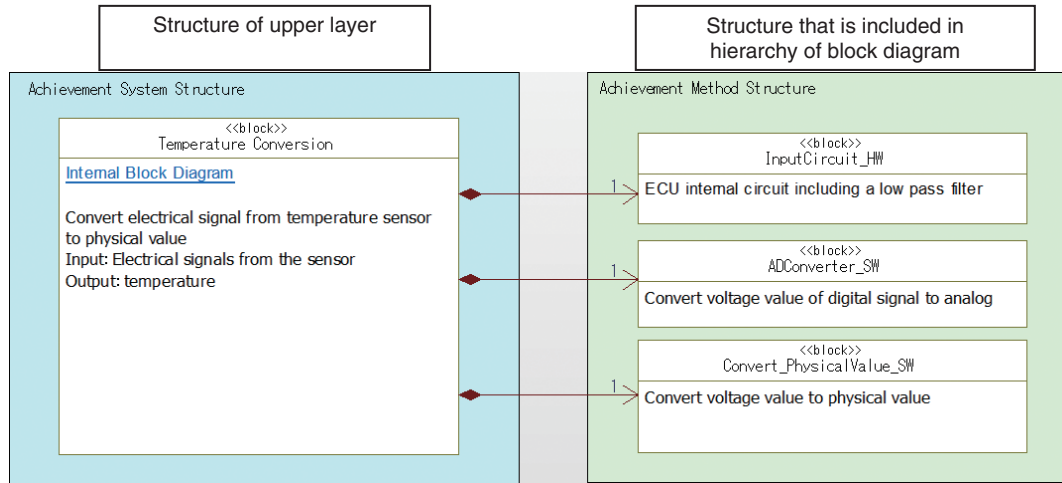


Fig. 7 Block diagram

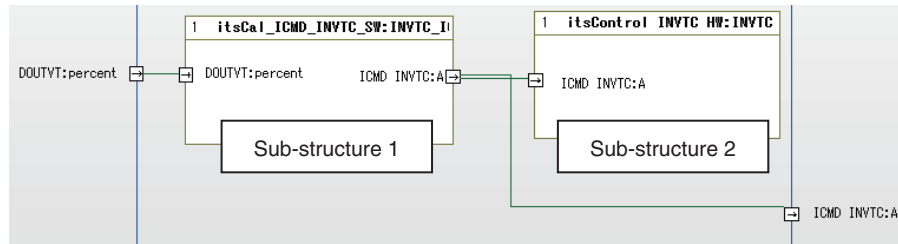


Fig. 8 Internal block diagram

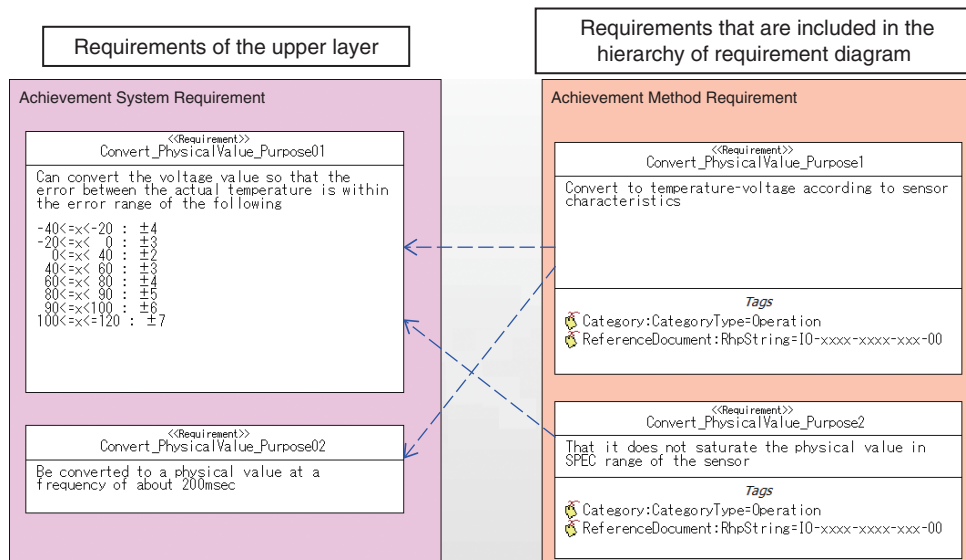


Fig. 9 Requirement diagram

3. Method for Confirming the Validity of Requirements Using MILS

In order to reduce reworking from development and to increase development efficiency, it is necessary to verify the validity of control requirements and increase the accuracy of requirements before conducting PT-ECU hardware development or software development. The method chosen for this is to implement operation verification by MILS with a Simulink model.

3.1. Creation of Simulink Model for Platform Verification

The ECU operation verification of the platform is implemented with the configuration shown in Fig. 10 in order to verify PT-ECU operation using input from sensors and output to devices.

MILS is a method of verification that models actual sensors and devices and performs desktop simulation of electronic control unit software operation.

The Simulink model shown in Fig. 11 was created for the implementation of MILS on the platform in the present research.

The plant model in the figure corresponds to sensors, devices, and so on. The controller model corresponds to control requirements. The test model corresponds to input from other systems. The input-output (I/O) model corresponds to the input-output circuits of the ECU.

3.2. Test Pattern Creation and Verification

The controller model, which is the test object, is made

up of a sensor malfunction detector, a device malfunction detector, a sensor controller, and a device controller. Each part has to be verified. Platform verification is conducted in two sections: the basic operation verification and the malfunction detection verification.

- (1) In basic operation verification, input values (target values) received from other systems are dynamically changed as shown in Fig. 12. The values output by the PT-ECU output unit relative to the input values go through verification to make certain they are valid values that match with the listed verification items.
- (2) In malfunction detection verification, malfunctions due to sensor and device open circuits, ground faults, short circuits, etc. are simulated in the model. The value of the platform malfunction detection status is then verified to make certain it is a valid value that matches with the listed verification item.

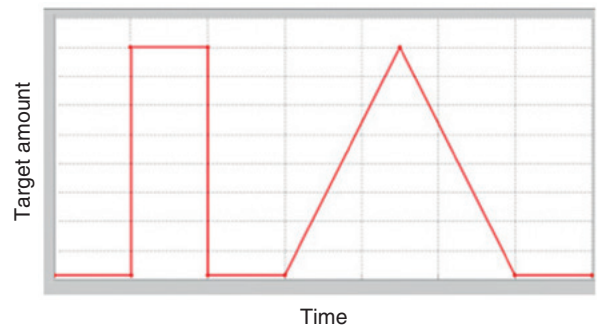


Fig. 12 Simulation test pattern

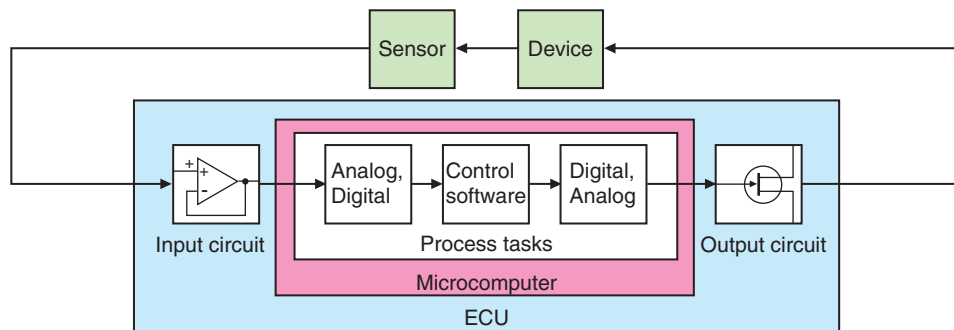


Fig. 10 Real-world environment

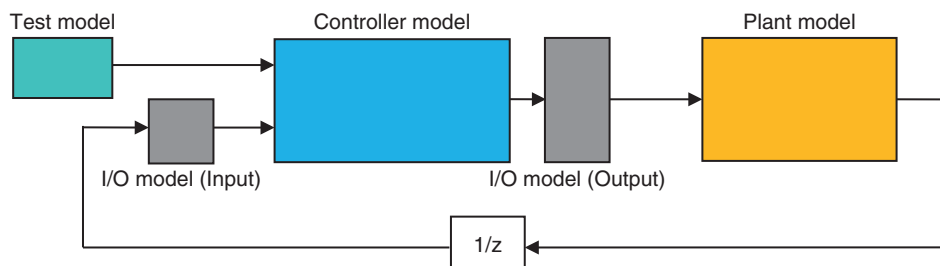


Fig. 11 Virtual environment

The validity of requirement specifications can be confirmed by unit of system function of the platform through MILS using the Simulink model.

The present research confirmed the usefulness of MILS and the issues involved by applying it to new device development in the 2015 model year. This was anticipatory application performed so that validation by MILS can be applied to the development of new devices from the 2016 model year on.

4. Method for Structuring Software with Consideration for AUTOSAR Standards

Overviews of the various parts of the AUTOSAR architecture shown in Fig. 3 are given below.

The application is PT-ECU hardware-independent software. It is positioned as an area of competition among automotive manufacturers.

The runtime environment is software that abstracts connections among sensors, devices, PT-ECU internal circuits, and microcontrollers, so that even if this hardware configuration changes, the interface between applications as well as the interface between applications and the basic software (BSW) will not change.

The BSW is software that changes the program due to changes in the microcontroller or the PT-ECU internal circuits. It is configured as internal functional modules that can be plugged in and unplugged to match the hardware configuration, so that it can be more reusable.

4.1. Platform Function Categories

In order to overcome the issue of the large amounts of time taken to bring Honda's existing software architecture and AUTOSAR standards into compatibility,

the platform functions were divided into the six groups shown in Fig. 13.

Overviews of the various groups are given below.

The scheduler in the figure performs PT-ECU task control.

The on-board diagnosis and regulation executes self-diagnosis of the on-board computer and generates data for legal compliance.

The memory performs management of non-volatile data.

The controller area network implements communication to and from other units.

The off-board tester implements communication to and from external diagnostic equipment.

The I/O implements electronic control unit input and output.

4.2. Interface that Takes AUTOSAR Standards into Account

In creating software specifications for the platform functions included in each group, the meaning of the interface published to an application is made to conform to the interface provided to applications by the functional module in AUTOSAR that is matched to each group. The interface name is generated as a combination of standard abbreviations used in the automotive industry.

The measures in section 4.1 and section 4.2 not only facilitate the plugging in and unplugging of software matched to the hardware configuration; they can also make software more reusable, for example by mitigating the burden of work on interface compatibility with applications when PT-ECU suppliers implement platform software in conformity with AUTOSAR.

In the present research, software specifications that support new architecture were created and partially

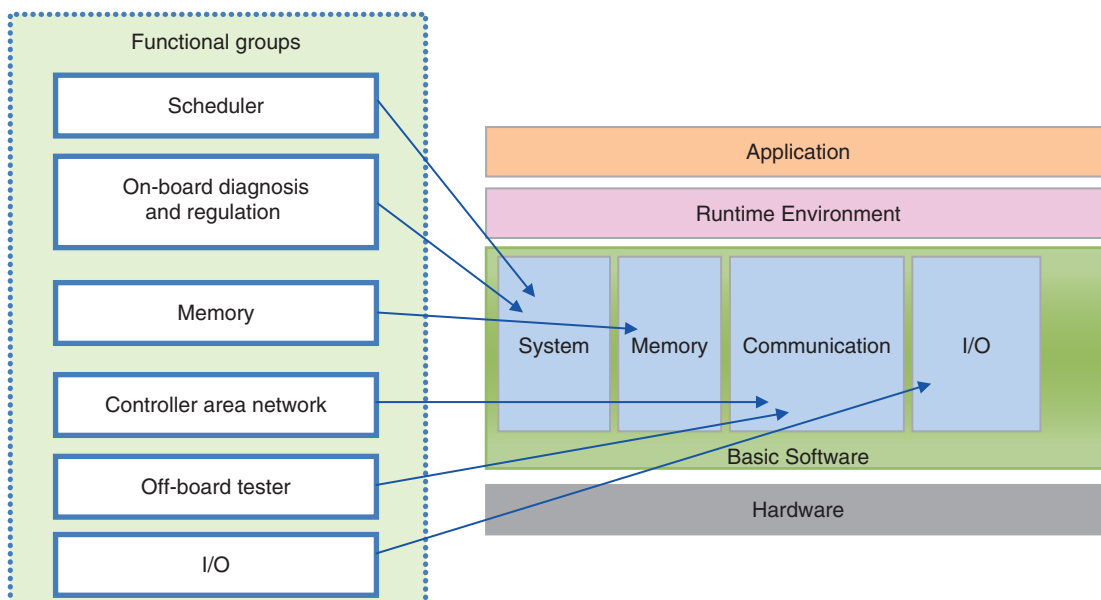


Fig. 13 Linkage of functional groups and AUTOSAR structure

applied to 2015 model year mass-production models. The objective is to make all software architecture for platform functionality transposable into new architecture in the 2016 model year mass-production models.

5. Application to Development of 2015 Model Year Mass-Production Models

5.1. Placement into a Hierarchy and Confirmation of the Validity of Control Requirements Using SysML

SysML is used to place control requirements into a hierarchy and confirm their validity, and the objective here is to apply this to the development of mass-production models. To that end, guidelines were created for the definitions of the levels described in section 2.1 and for the method of creating the diagram described in section 2.2.

At the system design level, SysML demonstrates advantages for the purpose of assuring traceability, which establishes relationships among items of information. Items of information that are similar end up being dispersed, however, and it is a characteristic of SysML that isolating just a specific type of information may not be readily achieved. This has been found to result in the issue of inability to efficiently check into information excesses and deficiencies during verification. As a result, the necessary tasks were defined before verifying system design requirement specifications as a measure to increase the efficiency of verification.

5.1.1. Information materials presented for verification

In order to efficiently display the information that is subject to verification, the creator of the specifications will create an HTML report in advance. The HTML report will extract and record items necessary for verification, such as diagrams, requirement items, component elements, and verification items.

The HTML report can have a template created in advance to specify the items of information that are necessary for verification, as well as how to display them. Doing this will allow automatic creation of reports with the same format by anyone. HTML reports can also connect related items of information using hyperlinks, which can allow verification to proceed more efficiently.

The interface report shown in Fig. 14 is one example of application to an HTML report. The report shows the input and output ports for each design block by level and in table format. Using these tables, input and output for multiple blocks that cannot be displayed simultaneously in the SysML format can be put through comprehensive verification.

The use of these HTML reports for verification of requirement specifications can reduce the amount of time spent in presenting requirement specifications to the verifier, thereby enabling more efficient verification.

5.1.2. Style check of requirement specifications

In order to raise the quality of requirement specifications,

Interface of Platform_IAT11				
This table is generated from L3 flowports				
Name	In/Out	Description	Unit	OwnerBlock
IntkAirT1B1_st	In	IAT11 status information	Integer	FI-control
IntkAir_t1B1	In	Engine intake air temperature (IAT11)	DegreeCelsius	FI-control
IAT11AD	Out	IAT11 sensor input voltage value after the AD conversion	V	IAT11_calc
IntkAirT1B1_st	Out	IAT11 status information	Integer	IAT11_calc
IntkAir_t1B1	Out	Engine intake air temperature (IAT11)	DegreeCelsius	IAT11_calc
This table is generated from L4-1 flowports				
Name	In/Out	Description	Unit	OwnerBlock
IntkAir_t1B1	In	Engine intake air temperature (IAT11)	DegreeCelsius	IAT11_DetectRangeFail
IAT11AD	Out	IAT11 sensor input voltage value after the AD conversion	V	IAT11_Conversion
IntkAir_t1B1	Out	Engine intake air temperature (IAT11)	DegreeCelsius	IAT11_Conversion
Obd10D_st	Out	IAT11 OutOfRange status information	Integer	IAT11_OBD_Common
Obd10_st	Out	IAT11 status information	Integer	IAT11_OBD_Common
This table is generated from L4-2 flowports				
Name	In/Out	Description	Unit	OwnerBlock
IntkAir_t1B1	In	Engine intake air temperature (IAT11)	DegreeCelsius	OutOfRange_Lo_Detection_SW
IntkAir_t1B1	In	Engine intake air temperature (IAT11)	DegreeCelsius	OutOfRange_Hi_Detection_SW
IAT11AD	Out	IAT11 sensor input voltage value after the AD conversion	V	IAT11_ADC_SW
IntkAir_t1B1	Out	Engine intake air temperature (IAT11)	DegreeCelsius	IAT11_PhyConv_SW
Obd10DH_st	Out	IAT11 OutOfRange High status information	Integer	OBD_Common_SW
Obd10DL_st	Out	IAT11 OutOfRange Low status information	Integer	OBD_Common_SW
Obd10H_st	Out	IAT11 Limit High status information	Integer	OBD_Common_SW
Obd10L_st	Out	IAT11 Limit Low status information	Integer	OBD_Common_SW

Fig. 14 Interface (I/F) report

it is necessary to verify that the requirement specifications have been created using documentary forms and formats that conform to the requirement analysis guidelines that have been formulated. Human visual inspection, however, may result in omissions and oversights, and so requires enormous amounts of time for verification. It was decided, therefore, to proceed using tools developed in-house, so that the person creating specifications could perform simple advance checks of the documentary forms and formats of requirement specifications and present those results (Figs. 15 and 16). Performing style checks before verification enables the written requirement specifications to be verified with contradictions and connections not envisioned in the design removed. This allows the emphasis to be placed on checking the validity of the content of requirement specifications.

With the identification of issues and the measures to address them described above, there is now a prospect that placing control requirements into a hierarchy and

confirming their validity using SysML can be applied to mass-production models in the 2016 model year.

5.2. Confirming the Validity of Requirements using MILS

In order to implement the method of confirming the validity of requirements by means of MILS, guidelines were formulated for creation of the Simulink model for platform verification described in section 3.1 and for creation and verification of the test pattern described in section 3.2.

In connection with the development of new devices scheduled for application to 2015 model year mass-production vehicles, the validity of requirements was verified by means of MILS. The following findings were obtained.

- (1) In the opening phases of development, when there is no PT-ECU hardware, software, or actual devices, MILS demonstrates its effectiveness in such applications as provisionally deciding on the control cycle, and

ProjectFileLocation D:\Platform\Platform_Brake.rpy			
Result	No	Check point	Result details
OK	1-8	Does the model have a comment named "Diagram" in the package "Index".	
Failure	1-8-1	Does the model have hyperlinks to all diagrams except internal block diagram in the description of the comment named "Diagram".	Hyperlink to diagram does not exist L4-1::L4-1_CRD_BrakeNoSwDet_ECU L4-1::L4-1_BD L4-1::L4-1_RD_Diagnosis L4-1::L4-1_RD_Operation L4-1::L4-1_CRD_BrakeNoSwDet_ECU L4-1::L4-1_CRD_BrakeNoSwErrorDet_ECU L4-1::L4-1_CRD_BrakeNoSwErrorDet_ECU L3::L3_CRD_BrakeStDet
CHK	2	Does the model have a block.	block exist.
OK	2-1	Are the names of blocks in the model all different.	
OK	2-2	Do the names of blocks in the model all start with capital letters. Also, does the model use only alphanumeric text and the underscore character.	

Fig. 15 Style check sheet

	IAT11_ADC_Purpose	IAT11_DetectRangeFail_Purpose9	IAT11_DetectElecFail_Purpose8	IAT11_DetectRangeFail_Purpose8	IAT11_DetectElecFail_Purpose7	IAT11_DetectRangeFail_Purpose6	IAT11_DetectRangeFail_Purpose5	IAT11_DetectRangeFail_Purpose4	IAT11_DetectRangeFail_Purpose2	IAT11_DetectRangeFail_Purpose1	IAT11_OBD_Common_Purpose6	IAT11_OBD_Common_Purpose5	IAT11_OBD_Common_Purpose4	IAT11_OBD_Common_Purpose3	IAT11_OBD_Common_Purpose2	IAT11_OBD_Common_Purpose1	IAT11_DetectElecFail_Purpose6	IAT11_DetectElecFail_Purpose5	IAT11_DetectElecFail_Purpose4	IAT11_DetectElecFail_Purpose3	IAT11_DetectElecFail_Purpose2	IAT11_Conversion_Purpose3	IAT11_Conversion_Purpose2	IAT11_Conversion_Purpose1	FailDetect_CommonReq	IAT11_calc_Purpose4	IAT11_calc_Purpose5	IAT11_calc_Purpose6	IAT11_calc_Purpose8	IAT11_calc_Purpose1
IAT11_calc	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
FI-control	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
_SystemDef	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Sensor	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IAT11_Conversion	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IAT11_OBD_Common	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IAT11_DetectElecFail	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IAT11_DetectRangeFail	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IAT11_PhyConv_SW	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Limit_Hi_Detection_SW	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Limit_Lo_Detection_SW	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
OBD_Common_SW	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
OutOfRange_Lo_Detection_SW	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IAT11_ADC_SW	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IAT11_InputCircuit_HW	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
OutOfRange_Hi_Detection_SW	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 16 Requirement-block connection check

ascertaining what kinds of abnormal operations could occur.

- (2) The feasibility of verification is dependent on the craftsmanship of the plant model used by MILS. Consequently, under present conditions, verification using actual hardware is essential for this decision on the control cycle, and decisions on threshold values for determining abnormalities and other such purposes. The questions of what kind of verification should be conducted with MILS and on actual hardware, respectively, will require further continued study.

These measures have made clear that MILS is useful and that there are issues involved. There are now prospects for the application of validation using MILS to the development of new devices in the 2016 model year and beyond.

5.3. Structuring Software in Consideration of AUTOSAR Standards

The policy for dividing up functions and the policy for deciding on the interface described in section 4 were followed in creating software specifications to support a new architecture for the four groups of on-board diagnosis and regulation, memory, controller area network, and off-board tester.

Software specifications were also created to support new architecture for those portions of the two groups of scheduler and input/output that had changing function requirements from before.

The above software specifications supporting new architecture were applied to mass-production models for the 2015 model year. This yielded the prospect of replacement with the new architecture in all the platform function groups in mass-production models for the 2016 model year.

6. Conclusion

The methods below were devised, and they were applied to the development of mass-production models for the 2015 model year. This enabled the preparation of arrangements to achieve control development that results in fewer returns.

- (1) A method for placing control requirements into a hierarchy using the SysML format, and a method for confirming the validity of control requirements
- (2) A method of validation by means of MILS that uses the Simulink model at the specification design stage
- (3) A method for structuring software architecture for the platform that takes AUTOSAR into consideration

Reference

- (1) <http://www.autosar.org/>
- (2) Information-technology Promotion Agency, Japan: jobun, shisutemu enjinieringu ni tsuite, moderu besu shisutemuzu enjinieringu donyu no tebiki, p. 13-36 (2013) (in Japanese)

■ Author ■



Hidekazu ARAKI



Kanako SAITO



Ryosuke OHGUCHI



Masayuki FUNAKOSHI



Kensuke YAMAMOTO