

New MBD Software Development Process for Electronic Control AWD System

Satoki WATANABE* Hiroki YOSHINO*
Yosuke SHIONOYA* Takayuki SEKI*

ABSTRACT

A new model-based design (MBD) software development process was constructed for the development of electronic control all-wheel-drive (AWD) systems. This new process enhances system control design with the aim of increasing specification quality compared to the previous software development process that applies executable model-based control design specifications. The new process changes the verification process and adjusts the division of development duties between system control design and software detail design. In addition, a method for generating the related specifications was constructed, and plug-in structuring was performed for applications to increase the specification reuse rate.

The new process was applied to the development of software for a new mass-produced electronic control AWD system, and resulted in a 50% reduction in the volume of specification changes compared to the previous software development process.

1. Introduction

Recently, the complexity and scale of electronic control systems and control software mounted in vehicles is increasing. However, demands to increase the reliability of control, reduce development times, and increase development efficiency are also increasing. Therefore, the introduction of MBD specifications based on executable models is proceeding as a means of developing software and describing specifications, and model description guidelines are also being established⁽¹⁾.

A general-purpose automatic code generation tool⁽³⁾ and a data setting tool developed internally by Honda⁽⁴⁾ have been integrated with MATLAB/Simulink and Stateflow,⁽²⁾ and MBD specifications based on fixed-point data model expression have been applied to develop electronic control AWD systems from 2007 mass-production models. These MBD specifications have been applied to increase system developers' understanding of specifications, reduce inconsistency between MBD specifications and software, and facilitate software design from the specifications created by system control designers to the actual code

generation. To reduce the verification load for model and code generation, an MBD software development process has been constructed that limits the types of blocks that can be used as MBD specifications, restricts the data setting values for the parameters and I/O applied to the blocks, and specifies the data names and other description methods.

This process is expressed by a V-process as shown in Fig. 1. When creating an MBD specification model, the system control designers set fixed-point data defined within the model. However, these data settings require software design skill to perform the appropriate scaling. In addition, the process to set the data in the model involves not only making the data settings but also assembling the data descriptions and other incidental information, so generating MBD specifications required enormous amounts of time. However, these design tasks also fall under the scope of software detail design, and the division of duties between system control design and software detail design was unclear.

In addition, this process enabled system control designers to generate MBD specifications while checking the consistency of the requirement specification and the

* Automobile R&D Center

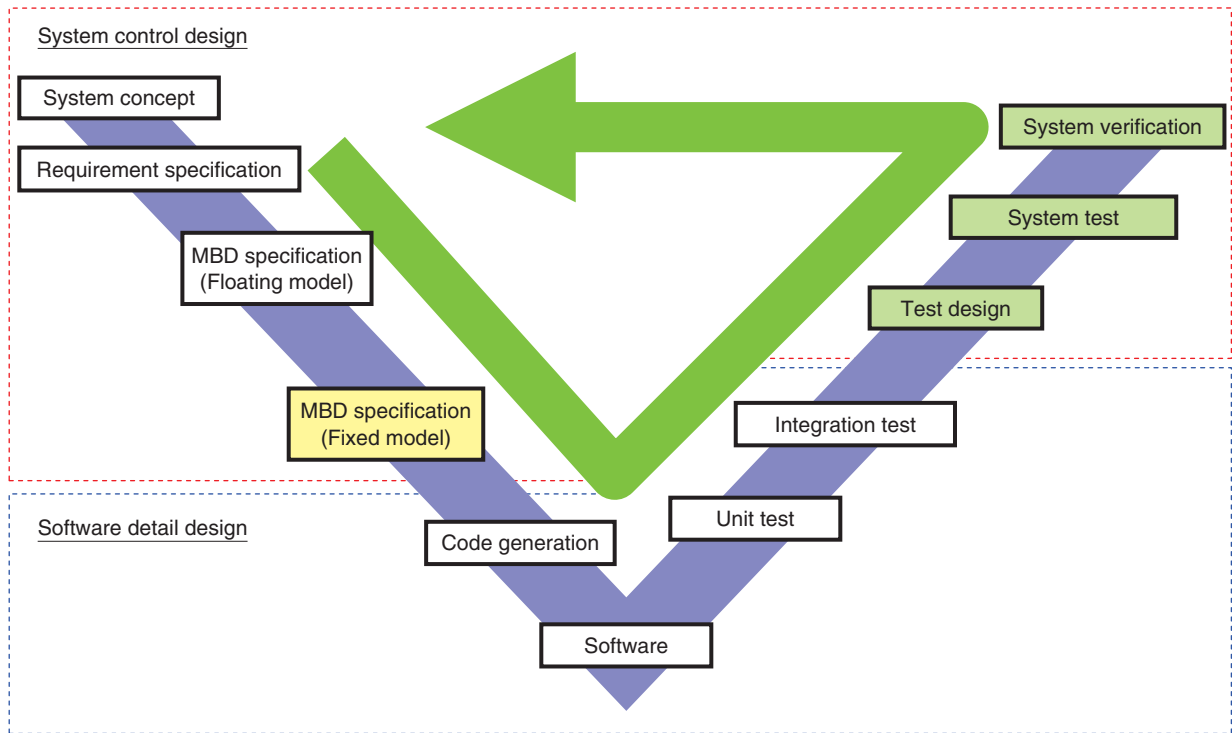


Fig. 1 V-process of previous MBD process

model, but means of confirming the correctness of the specifications were not defined, and there was also no system for managing the operation check results in the software design process of the V-process. This resulted in waste as the same tests to check operation when generating the MBD specification were performed again during system verification, which is the final software development process, and there were also cases where specification imperfections that could not be detected by the operation check when generating the MBD specification were discovered during system verification. The specification correction loop in these cases was quite large, which increased the ratio of software development due to specification changes in the latter stages of system development, and raised the issue of whether executable MBD specifications had sufficient merit for use.

2. Aims of New MBD Software Development Process

The following methods were considered for the new MBD software development process to address the issues with the previous MBD software development process and enhance specification quality.

- (1) Introduction of front loading verification using a floating-point data model
- (2) Creation of a specification format for enhanced system control design
- (3) Development of a tool for extracting MBD specification data

- (4) Plug-in structuring of applications to increase the specification reuse rate

These methods aimed to increase software development efficiency, reduce specification corrections, and reduce the ratio of the software development load in the latter stages of system development.

3. Front Loading Verification

3.1. V-process Type New MBD Software Development Process

In contrast to the previous V-process shown in Fig. 1, the new MBD software development process uses the V-process shown in Fig. 2. The changes from the previous process are the following two points. First is that front loading verification is performed before shifting from system control design to software detail design. Second is that the MBD specification generated by system control designers is expressed by a floating point data model, and the fixed-point data model and data scaling have been transferred to the software detail design area.

The aim of performing front loading verification before shifting to software detail design is to prevent outflow of imperfections with the requirement specification and MBD specification and inconsistency between these two specifications to downstream processes, and to increase development efficiency by reducing the size of the correction loop. In addition, the verification process is divided, enabling reduction of the system verification load, which is the final system development process.

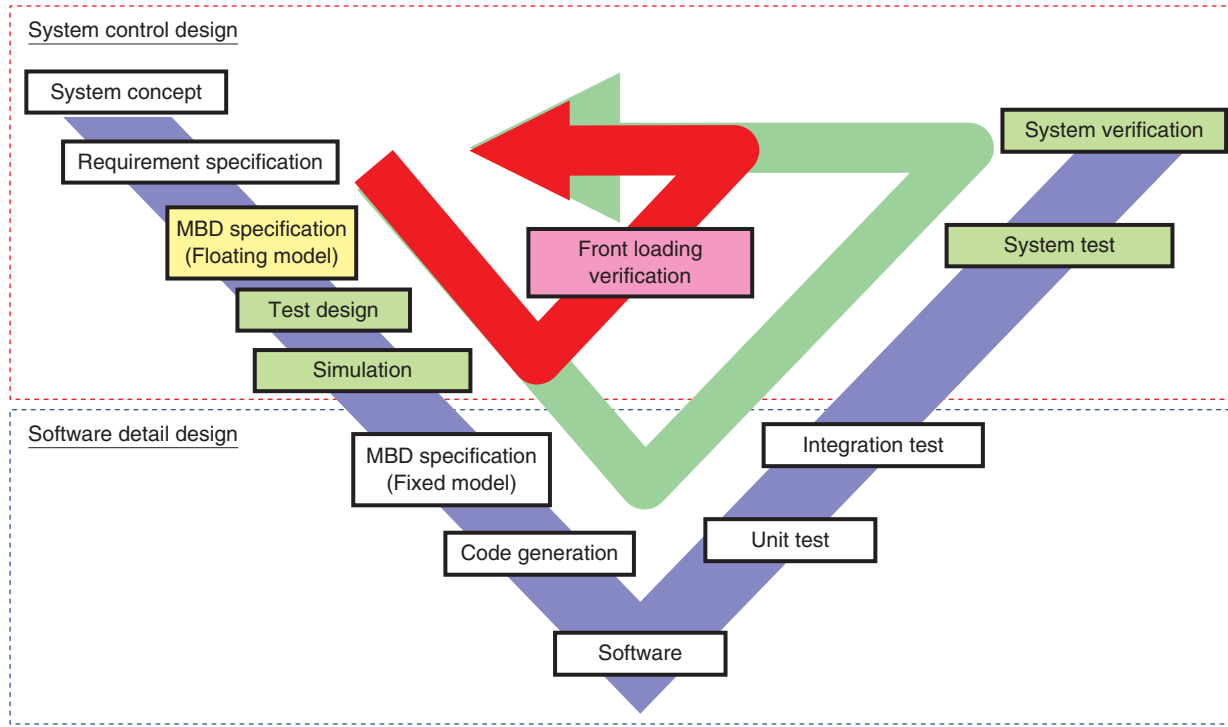


Fig. 2 V-process of new MBD process

Keeping the MBD specification as a floating-point data model when performing front loading verification reduced the data setting processes and secured efficient flow in the loop from simulation to front loading verification. This resulted in a verification process focused on system control design.

3.2. Dispersion of Verification Process

The new MBD software development process disperses the verification load by dividing the verification process into the two parts of front loading verification and system verification using an actual vehicle as shown by the V-process in Fig. 2.

Front loading verification performs simulations and theoretical verification to check the steady state control operation, make sure the requirement specification and the MBD specification are consistent, and verify the validity of the parameter design values. System verification using an actual vehicle evaluates the vehicle dynamics performance, verifies the conformance with the vehicle environment, and verifies the transient states (Fig. 3).

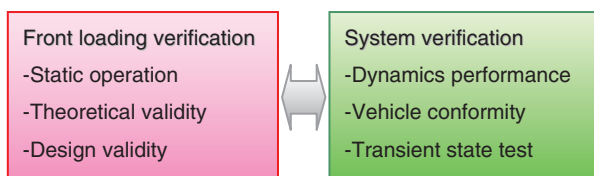


Fig. 3 Division of verification items

The new MBD software development process generates a test design specification for the front loading verification items and system verification items at the same time as the MBD specification is generated. In the front loading verification event, the various verification methods and the front loading verification results are reviewed. By reviewing front loading verification in this manner, including system verification methods using an actual vehicle, system control designers and verification approvers can deepen their understanding of system verification process test contents and check the validity of the test contents. This approach aimed to reduce the discovery of imperfect verification contents as well as imperfect specification requirements and inconsistency with the software during system verification, which is the final system development process.

In addition, in the previous process, the data introduced to the software in the initial development stages was not precisely set data, and the parameters were often adjusted while conducting tests using an actual vehicle. This meant that parameter changes sometimes affected other specification areas within the system or other systems. In the new process, specifications are verified and logical data is set in the software even in the initial development stages, which reduces parameter changes and repetitive work such as checking the effects of that software.

4. Specification Format Configuration

The new MBD software development process provides the following four formats in addition to the MBD

specification to promote consistent verification from system concept design through system verification (Fig. 4).

- (1) System concept sheet
- (2) Requirement specification
- (3) Test design specification
- (4) Test report

The features of each format are described below.

- (1) System concept sheet

This format is described together with the requirement specification of the functions to be introduced. It describes the purpose, the background and concept of the requirements, the events leading to the requirements, and other know-how.

- (2) Requirement specification

This specification clearly describes the specification requirements to the software detail design side. It excludes the background and system concept information from the system concept sheet, and describes only the purpose, requirements, and conditions of the functions to be introduced. The detailed version of this requirement specification becomes the MBD specification.

- (3) Test design specification

The verification review describes the verification methods and parameter setting methods beforehand, and

presents the verification results obtained in accordance with those methods. To facilitate checking of inconsistency between the requirements and the actual contents, the format also describes the requirement items, verification methods, viewpoint from which verification is performed, and parameter setting methods. At this time, the verification methods and parameter setting methods are divided into front loading verification items and system verification items, and are described so that the respective contents can be linked.

- (4) Test report

This format describes the front loading verification results and system verification results in a single sheet, and also describes front loading verification results that carry over to system verification, and other items. This helps prevent verification omissions between front loading verification and system verification.

5. Automatic Extraction of Data Information within Models

During system control design, the MBD specification is expressed by a floating-point data model. However, a data scaling standard is needed for software detail designers to create the software. In addition, system control designers also require scaling and other settings for certain data in order to perform appropriate control. Therefore, the following data settings are requested of software detail designers.

- a. Accuracy requirements for the final output of each application
- b. Parameter setting scaling requirements

This is accomplished by generating a specification (hereafter, “data setting requirement sheet”) that prescribes the data setting requirements for the above two items separately from the MBD specification, and providing it to software detail design as a set together with the MBD specification. Software detail design then uses this data setting requirement sheet as a reference to perform scaling for all data including data other than the required data, and creates the software.

However, if the data is described by hand entry when generating the data setting requirement sheet, data entry errors and other errors may result in inconsistent data names and data descriptions between the MBD specification and data setting requirement sheet. Therefore, specification expression errors were reduced by developing a tool that extracts data requiring data settings in the MATLAB/Simulink model, and reflects it to the data setting requirement sheet.

The automatic extraction tool used to generate the data setting requirement sheet consists of the following four elements.

- (1) MBD specification (floating-point data model)
- (2) Data information extraction program
- (3) Data request and insertion program
- (4) Data setting requirement sheet

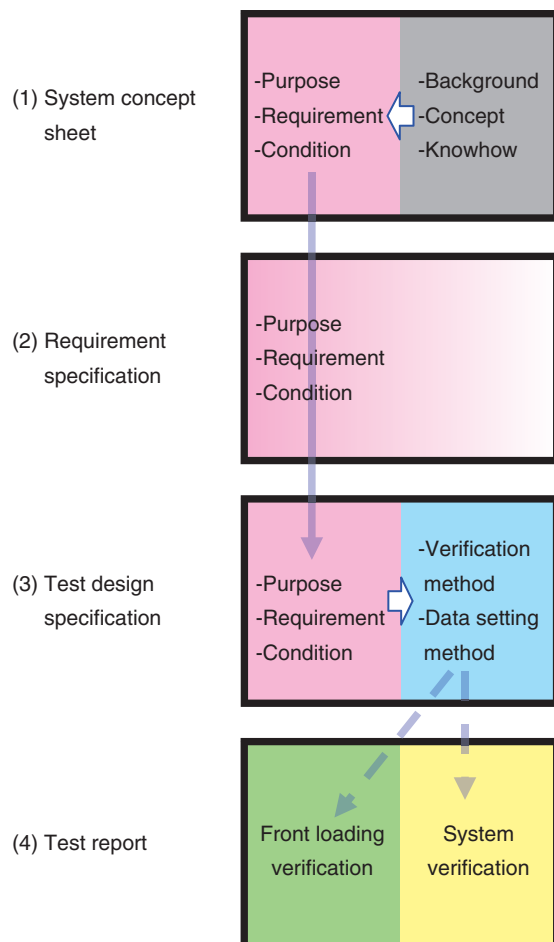


Fig. 4 Format of new process documents

The data setting requirement sheet is generated according to the flow shown in Fig. 5, and the features of this flow are described below.

(1) MBD specification (floating-point data model)

The variables and parameters to be defined in the data setting requirement sheet are expressed in model blocks within the MBD specification. In addition, the MBD specification prescribes the block applications and data names within the model in order to facilitate recognition of data attributes, enhance the legibility of the MBD specification, and increase verification efficiency. The data to be extracted for the data setting requirement sheet is also described by the MBD specification to add prefixes and set properties in order to define the data to be extracted.

(2) Data information extraction program

The data information extraction program analyzes the hierarchical structure of the model and searches using the information set in the model in (1) above as keywords. This program extracts only the data names, the data description information embedded in the blocks, and the data hierarchy from among the various information described within the model.

In addition, regarding parameter extraction, some data requires that multiple parameters be set for a single block, so the program can request the optimum number of parameters according to the applicable block.

(3) Data request and insertion program

This program is started up by the data setting requirement sheet, and the data information extraction program is executed by a request from this program. This program also inserts the data names, descriptions and data hierarchy in the list generated by the data information extraction program, into the data setting requirement sheet. In addition, this program has a function that performs data merge processing when creating a revision using existing data in the data setting requirement sheet.

This program also has a data check function that checks for duplicate data names and other errors, and a function that can extract changes in the information reflected when revising a data setting requirement sheet. These functions reduced imperfections in the data described in the MBD specification.

(4) Data setting requirement sheet

The data extracted from the model by the above program is arranged in the data setting requirement sheet, and the scaling to be performed for each piece of data and the data units are also described in order to request data settings of the software detail design.

The data is categorized beforehand and the format enables only selection of applicable data setting requirement values from lists corresponding to each category in order to reduce data entry errors and variance in data setting requirements due to multiple specification creators when describing the specification requirements.

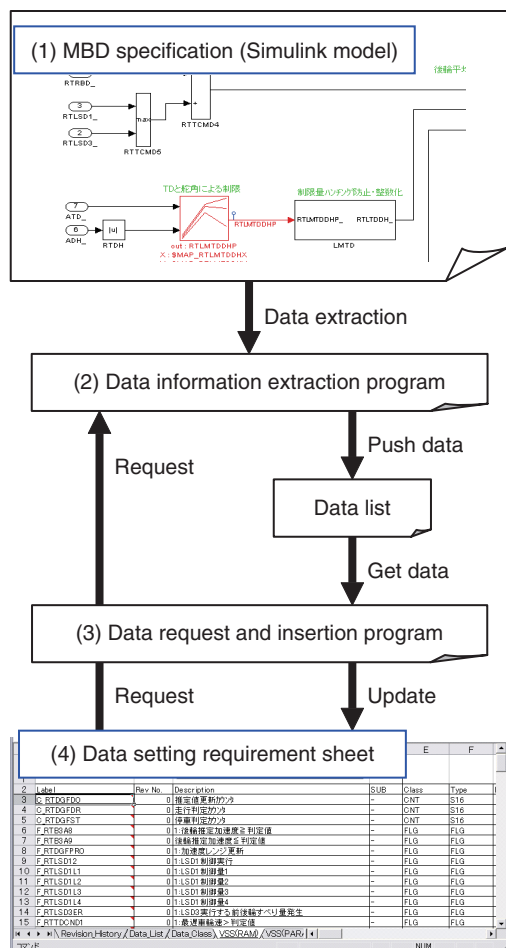


Fig. 5 Generation of data setting requirement sheet

6. Application Plug-in Structure

In the application configuration of the previous MBD software development process, the data I/O between applications exchange variables directly. As a result, when there is a change in the external output of an application, the specifications of all applications that reference those variables must be changed in consideration of the effects of the change in the reference source, even when the applications themselves do not change.

In the new MBD software development process, an application configuration was devised that reduces the need to remake specifications by reducing the effects of changes to application specifications on other applications and by increasing the specification reuse rate. This application configuration exchanges variables using a plug-in structure that enables easy replacement of applications while maintaining a software processing configuration that performs constant-time processing.

This plug-in structure features an Application Programming Interface (API) that performs physical value conversion in order to use OS data for calculations by each application, and an API for each application as shown in

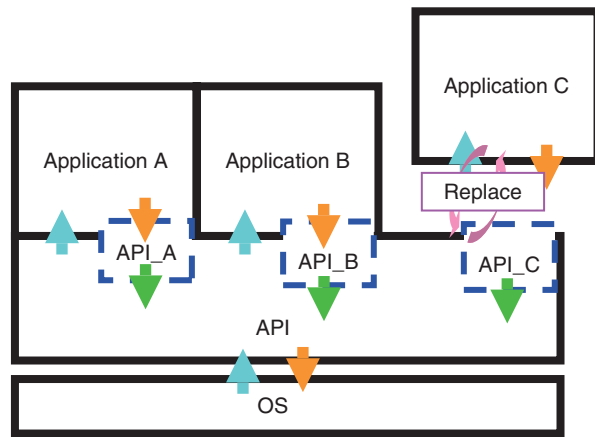


Fig. 6 Plug-in structure of applications

Fig. 6. When the output from an application is used as the input for other applications, the API of that application converts the output to a general-purpose variable to enable use as the input from the API. This enabled the API of each application to absorb changes in application outputs, which enabled the generation of software even when the referring application does not change.

For example, in addition to normal vehicle control applications, there are also applications that perform substitute value control such as control by a diagnostics unit, and it may be desired to change the substitute control values into the normal control values. In this case, instead of forcibly inserting the substitute values into the vehicle control applications, giving the vehicle control application API a function that performs processing to switch the control values enables each application to express specifications that require only the intended control. This reduced the specification changes that had to be made for various areas when specifications were added in previous development, which increased the specification reuse rate and development efficiency.

7. Evaluation of the Process

7.1. Evaluation of Specification Quality

The new MBD software development process was applied to the development of software for a new mass-production electronic control AWD system. The volume and contents of the specification adjustment sheets issued by software detail designers with respect to the specifications provided by system control designers were analyzed and compared with those when the previous MBD software development process was initially applied to new system development.

First, the ratio of the number of specification adjustment sheets issued to the total specification volume was reduced by approximately 20% compared to development that applied the previous MBD software development process.

In addition, the contents of the specification adjustment sheets were analyzed and categorized into the “Description error,” “Inconsistency between setting data” and “Imperfection with specification” items related to specification expression errors and inconsistency between the requirement specification and the MBD specification, and other items. Figure 7 shows these results. In the previous MBD software development process, specification expression errors and inconsistency between the requirement specification and the MBD specification accounted for over 80% of the total. However, this ratio dropped to 40% or less in the new MBD software development process that performs front loading verification. In addition, the new MBD software development process reduced the ratio of specification changes due to specification expression errors and inconsistency between the requirement specification and the MBD specification, as a percentage of the total specification volume, by approximately 50% compared to when applying the previous MBD software development process.

7.2. Evaluation of Software Development Load

Figure 8 shows the results of comparing the volume of issued software specifications over time during the system development schedule. In the previous MBD software development process, specification issue peaks in the latter stages of system development. In contrast, the new MBD software development process issues fewer specifications towards the end of development.

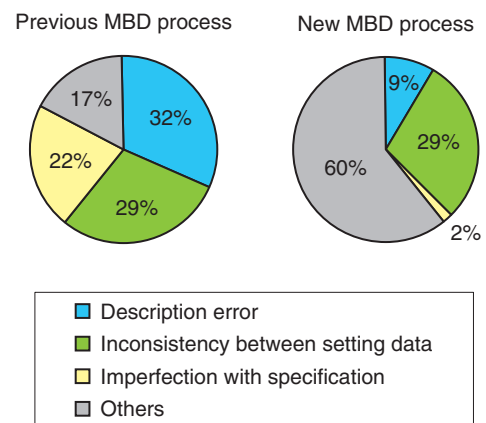


Fig. 7 Comparison of subject in requirement confirmation sheet

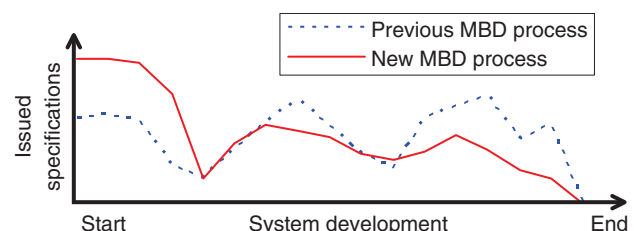


Fig. 8 Volume of issued specifications

8. Conclusion

The system control design and software detail design areas of the software development process for electronic control AWD systems were separated by expressing the MBD specification as a floating-point data model. This enabled efficient performance of front loading verification, which increased specification quality.

The new process was applied to the development of software for a new mass-produced electronic control AWD system, and reduced the volume of specification corrections due to inconsistency between the requirement specification and the MBD specification, and specification expression errors. This reduced both the specification creation process time and the verification work, which enabled reducing the ratio of the software development load in the latter stages of system development.

References

- (1) <http://jmaab.mathworks.jp>
- (2) http://www.mathworks.co.jp/products/product_listing/index.html
- (3) <http://www.dspace.com/en/inc/home/products/sw/pcgs/targetli.cfm>
- (4) Minowa, S., Hirota, T., Yokoyama, H., Ishiguro, T., Ohtsuka, S.: Autocoding of Power Plant Control Software, Honda R&D Technical Review, Vol. 16, No. 1, p. 243-248

■ Author ■



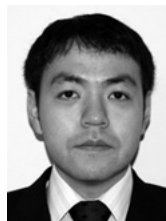
Satoki WATANABE



Hiroki YOSHINO



Yosuke SHIONOYA



Takayuki SEKI