

Hierarchical Accumulative Validation of Executable Control Specifications

Jared Farnsworth and Koichi Ueda
 Toyota Motor Engineering & Mfg NA Inc.

Hideaki Mizuno and Michio Yoshida
 Toyota Motor Corporation

ABSTRACT

The application of Model-Based Development (MBD) techniques for automotive control system and software development have become standard processes due to the potential for reduced development time and improved specification quality. In order to improve development productivity even further, it is imperative to introduce a systematic Verification and Validation (V&V) process to further minimize development time and human resources while ensuring control specification quality when developing large complex systems. Traditional methods for validating control specifications have been limited by control specification scale, structure and complexity as well as computational limitations restricting their application within a systematic model-based V&V process. In order to address these issues, Toyota developed Hierarchical Accumulative Validation (HAV) for systematically validating functionally structured executable control specifications. HAV can be implemented in a production vehicle development cycle to ensure specification quality while saving development time and reducing required staff. Toyota confirmed that HAV can be implemented to successfully validate executable control specifications for production control system development.

CITATION: Farnsworth, J., Ueda, K., Mizuno, H. and Yoshida, M., "Hierarchical Accumulative Validation of Executable Control Specifications," *SAE Int. J. Passeng. Cars – Electron. Electr. Syst.* 6(1):2013, doi:10.4271/2013-01-0430.

INTRODUCTION

Automotive control systems and software complexity are accelerating in order to improve vehicle performance, address sustainability, and provide increased functionality. It has become apparent that traditional methods for developing and validating control specifications are becoming insufficient to adequately address development issues. Control designers are faced with inadequate requirement analysis and test design which results in the high possibility of re-design and reevaluation later in the design cycle. Once specifications are developed, focus is largely given to the right hand side of a V-process. This can be inefficient due to the requirement for exhaustive experimentation with the actual product. Finally, as control system complexity has increased, an excessive number of tests must be generated to validate specifications. See [Figure 1](#).

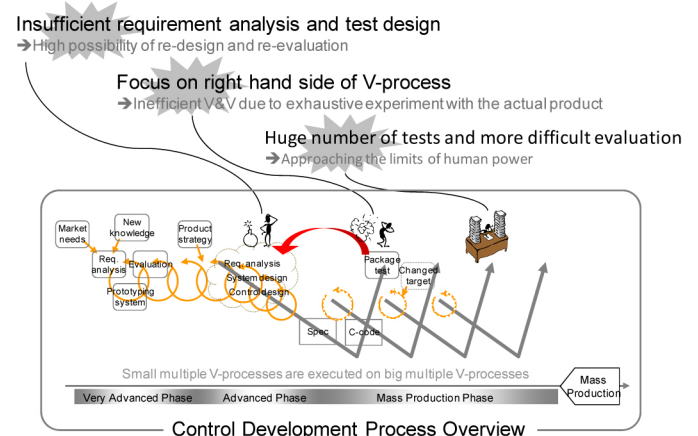


Fig.1. Traditional Controls Development Process

Automotive companies are turning to Model-Based Development (MBD) techniques to find solutions to these issues. Many approaches provide solutions to specific control applications, but are not generally applied to the entirety of a control system. In order to take full advantage of MBD to improve Fuel Cell Vehicle controls development, Toyota is implementing a production MBD process utilizing executable control specifications in the form of graphical models like Simulink® and Stateflow® for integrated multi-ECU control system and specification development. A major challenge when implementing MBD on this level is the approach to Verification and Validation (V&V) of embedded controllers within the control systems. V&V consists of processes and methods implemented to confirm the correctness of a control specification. It is imperative to introduce systematic V&V methodologies to minimize development time and human resources while ensuring control specification quality when developing large complex systems. Additionally, ISO 26262 recommends implementing formalized processes that ensure traceability.

This paper will discuss Hierarchical Accumulative Validation (HAV) developed by Toyota for systematically validating functionally structured executable control specifications as part of a model-based V&V development process as shown in Figure 2.

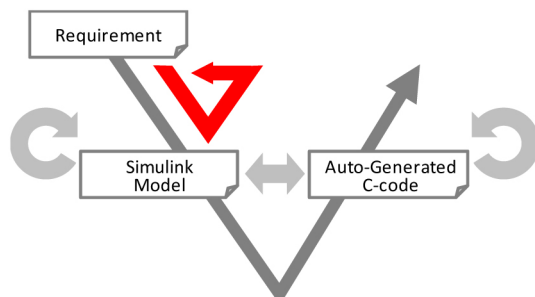


Fig.2. MBD V&V Process

Within this V&V process, control designers generate graphically based executable control specifications for an embedded system (left side). The executable specifications are validated against functional requirements during the design process. Specification design errors such as unintentional behavior, structural design errors like dead logic, divide by zero, and other bugs can be found and corrected early in the design phase. Once all embedded controllers within the ECU are validated, production c-code is automatically generated from the executable control specifications. The complete application level software can then be high-level functionally validated through Software-in-the-Loop simulations (SILS), Hardware-in-the-Loop Simulations (HILS), bench testing, and vehicle testing.

Figure 3 shows an example of a functionally structured executable control specification within an embedded controller.

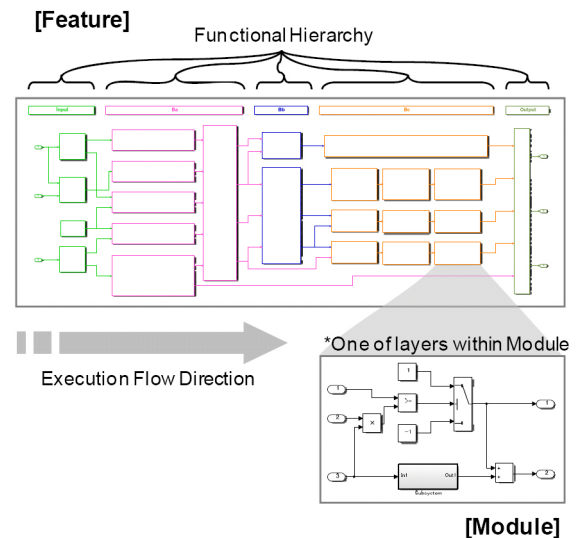


Fig.3. Functionally Structured Executable Control Specification

The feature level consists of control modules organized based on functional hierarchy (FH). Each FH performs a sub-function within the feature system. The modules and separate FH are defined as having low-level functionality such as state determination, sub-function calculations, and limits. The low-level modules combine to perform high-level functionality at the feature level. High-level functionality is defined as having physically meaningful time-series application. High-level functionality can include actuator control values, system status, and fault detection. Execution flow direction is left to right [1].

Typically, multiple control designers develop a feature specification and are individually responsible for defining functionality and building specific modules. When modules have been specified, built, visually reviewed, and checked for basic functionality, they are integrated into FH and the feature specification. Once the feature specification is built, test groups will perform V&V. Through this process control designers and test groups work in parallel creating and validating control specifications. This process is equivalent to control designers developing a traditional natural language specification document for a feature level requirement and being able to directly validate the specification. In this application, the specification is executable (i.e., it can be checked for correctness) and can be automatically converted into equivalent c-code for execution in an ECU. For example: Control designers are tasked with defining the requirements for fuel cell air system pressure control and building the specification. Air system pressure control may consist of internal state checking, target calculations, and mode selection. The overall requirement of air system pressure control can be considered a feature specification. Each of the sub-functions within the overall feature specification can be organized into FH. The FH consists of the required number of modules to complete specific sub-system requirements.

Control designers define system requirements and build the specification. Once the specification is built and control designers confirmed basic functionality based on their understanding of the system, test groups validate the executable specification against the requirements defined by the control designers. Test groups check for functionality and structural issues. If issues are found, control designers can check to see if requirements are not defined correctly, or if the specification has an error.

Several issues must be addressed to validate embedded controllers containing low-level and high-level functionality. The first issue to be considered is the process for validating a control specification for low-level structural design as well as higher-level functionality. The second issue that must be resolved is how to mitigate computational limitations during structural coverage-based test case generation for low-level validation. Finally, we had to consider how to generate physically meaningful time series test cases for high-level functional validation.

Toyota developed different methods to solve these issues for control specification validation and compliance with a necessary part of ISO 26262. These methods, processes, and tools can be implemented in a production vehicle development cycle to ensure specification quality while saving development time and reducing required staff. Toyota confirmed that HAV can be implemented to successfully validate executable control specifications for production control system development.

VALIDATION METHODOLOGY

In order to have confidence in the correctness of the control specification, validation should be conducted for low-level structural design (module & FH) as well as high-level functionality (feature). The sequence in which validation occurs is important. Low-level structural correctness should be validated before high-level functional validation to prevent unnecessary retesting. If low-level structural issues are found after high-level functional validation has been completed, retesting will be required to validate and document countermeasure effect on high-level functionality.

The validation approach taken should account for functional requirements (FR) that exist for individual modules, separate FH, across FH, and for the entire feature. Ideally, formal methods such as property proving would be applied to validate all FRs within the feature specification to achieve exhaustive coverage. Unfortunately, the state-space explosion problem due to scalability, non-linear mathematics, timer logics, lookup table control strategies, and computation limitations often prevent formal methods from being practically utilized in a production validation process. Alternative methods must then be applied to validate FRs and achieve quantifiable model coverage. Validation utilizing simulation based testing to objective model coverage metrics such as Modified Condition Decision Coverage (MCDC) have been recognized as a suitable approaches to validate

control specifications. Computational limitations can exist when attempting to achieve MCDC targets for a specification, however, simulation based testing allows for incremental and systematic validation of a control specification. Regardless of the methods used for validation, a goal for model testing is to achieve as close as possible to exhaustive coverage in order to have confidence in the correctness of a specification.

Considering executable control specification structure, development process, testing requirements, and limitations, the following assumptions can be made:

1. Low-level & High-level validation of a specification is required.
2. Once FH are structurally validated, feature functionality can be validated with test cases generated for specific functional checking.
3. Higher-level FRs should only be validated when required and validated lower-level specifications are included in the testing.
4. Targeted control specifications (FH to Feature) should be extracted from parent models in order to improve validation efficiency by mitigating practical and computational limitations related test case generation.
5. When best possible coverage is achieved for a module (MCDC), additional testing will result in systematic improvement towards exhaustive coverage (exhaustive coverage is a goal).
6. Testable FRs may only occur at greater than or equal to FH level.

These assumptions were used as the foundation of HAV. We believe that when these assumptions are applied with simulation based testing techniques, robust and systematic validation can be achieved. The assumptions can be organized into five main concepts:

Validate by Functionality

Executable control specifications are typically structured by functionality (Fig 3), providing a clear delineation of model structure and function allowing control designers to easily define and assign FRs to modules, FH, and the complete feature specification. Once the functional structure is known, a systematic approach can be taken to validating the specification. One issue to be considered is at which levels within the model can or should be validated. Generally, at module level, when FRs are translated into testable models, they are typically equivalent to the module itself and visual inspection may be sufficient. Once modules are grouped into sub-functions within a FH, the combined logic is sufficiently complex to allow for testable FRs. Stated in another way, the combined logic is complex enough that a FR can be translated into a model that is not equivalent to the combined models and can be tested. Based on these observations, it is