

High Confidence Powertrain Control Software Development

Hakan Yazarel, Tomoyuki Kaga, Ken Butts

Toyota Motor Engineering & Manufacturing North America

To reduce fuel consumption and emissions, improve vehicle performance, and meet higher customer expectations, the powertrain control system is becoming more and more complex. As a result of the increased complexity, the software has a structure composed of thousands of variables and tuning parameters with highly interacting functions. The software is becoming very difficult to test and the cost for securing the highest quality is increasing exponentially.

Model-Based Development (MBD) has been promoted for several years in this context. One of the biggest and important challenges in model-based development is **Verification and Validation (V&V)**. Conventional V&V processes of powertrain control software rely on exhaustive experimental testing. However, the complexity, large scale, and hybrid nature (multiple modes of operation) of the system make conventional V&V process extremely challenging and time consuming. Therefore, novel methodologies which detect potential defects in the early design stages are needed.

Toyota uses Matlab®¹, Simulink® and Stateflow® for specifying the software executables. As part of the MBD effort, the legacy control algorithms written in C code are converted into executable specifications [1]. The aim is to: 1) enable development in the virtual world using plant models, 2) improve maintainability, and 3) facilitate significant system re-design when necessary. While these software specification models are effective for simulation and automatic code generation, we need new tools to analyze them for functional correctness.

Challenges and Research Needs

There are many challenges and research needs to be addressed for realizing advanced V&V in the MBD context. We list a few of them in the following:

Complexity: The biggest challenge facing MBD and V&V is the complexity of the system. The software consists of thousands of variables and tuning parameters with highly interacting functions arranged in a "Spaghetti" type software structure. These characteristics make the software very difficult to verify. New results addressing the complexity, hybrid nature, and scale of the system are needed.

Architecture: Defining levels of abstractions and perhaps modeling languages for specifying these abstraction levels are needed [2]. For example, a Simulink model visually provides hierarchies. However, it has all the details that a C-code has. A high level software architecture abstraction that captures the necessary properties to integrate the software modules into a design is needed. For example, these properties might include component interactions, real-time communication schedules, data access rules,

¹ Matlab, Simulink, and Stateflow are registered trademarks of the MathWorks, Inc, Natick, MA.

and explicit and implicit dependency structure. This will enable easy replacement, composition, and evaluation of new software modules.

Hierarchical verification methods are promising to overcome the complexity problem for module level, feature level and system level verification. First is the verification at the lower levels to confirm modules and features meet their specifications. Next is the verification at the system level where we check that the interactions and behavior of modules and features (as captured by the architecture) meet their specifications.

Ensuring semantics-preservation between architecture abstraction layers during the development process is a key challenge to develop high confidence control software. As we move down the abstraction layers, there are more details added to the design. The less abstract implementation must preserve the semantic properties of the high level model. If it is not preserved, the gap and its effect on robustness need to be analyzed.

Software Integration: As new algorithms are developed using Simulink, existing legacy code will be integrated into the design later in the process,. The integration can be time consuming and error prone if the software module written by an engineer is not as formal as required to use automated development tools. To provide reliability and quality guarantees, sufficient formalization of the software modules is necessary. Tools that semi-automatically extract the information necessary to formalize the code are needed.

V&V Tool Chain: The powertrain control software has evolved by adding new functionalities over many years. As new functionalities are added, the number of tests required to keep the reliability and quality grows exponentially. Instead of a “final check” test-type validation, comprehensive V&V tools are needed in every step of the design process. We need to expose defects at the logic design, implementation and system integration stages.

Roadmap

Toyota has started promoting model based development and advanced V&V activities for high confidence power train control software. Our aim is to improve our development capability by addressing the challenges listed above. Additionally, a roadmap should include defining standards for automotive software V&V activities starting with the definition of V&V for automotive control software itself. See [3] for a hardware development and verification standard.

References

- [1] Koichi Ueda, “*Converting Legacy Embedded Control Software to Executable Specifications*” MathWorks International Automotive Conference 2006.
- [2] <http://aadl.info/>
- [3] IEEE Std 1800-2005 IEEE Standard for SystemVerilog: Unified Hardware Design, Specification, and Verification Language