Development of Control Platform for Electric Module

1. Introduction

Various companies are developing initiatives for Connected, Autonomous, Shared & Services, and Electric (CASE) with the aim of reducing traffic accidents, providing safe and secure freedom of transportation, usefulness to users and environmental measures. NTN, in view of these trends, has developed a universal series of electric modules containing various motors and actuators so that we can contribute to the “Autonomous” and “Electric” aspect of CASE. NTN achieves this through the use of our core expertise and technologies which includes bearings and ball screws, as well as our research into the design technology of motors and control technology of vehicles. The following are descriptions related to some of the electric modules we have developed as well as their proposed uses.

1) Overview of Electric Module Series

(1) B II Series (Fig. 1)

- Electric actuator using paired gears to reduce the speed and convert rotation motion to linear motion through the use of a ball screw
- Through the use of optional components such as a planetary reducer or a reverse lock out, the driving power can be increased as well as the ability to maintain actuator position

(2) B III Series (Fig. 2)

- A compact electric actuator that combines a high output brushless DC motor, a ball screw and a linear motion mechanism within a hollow structure

In the automotive industry, the development of CASE (Connected, Autonomous, Shared & services, Electric) has proceeded. Many devices (such as cameras, sensors, actuators, motors, etc.) are used for autonomous driving and electrification. And the software for controlling those devices increase year by year. The functional safety standard ISO 26262 has been established and development to ensure safety is proceeding. However the development which is based on ISO 26262 requires larger number of people, costs, and much more skills than conventional development which is based on IATF 16949. NTN is developing electric motors, actuators, and controllers that contribute to electrification and By-wire-system. In this article, we introduce the development of control platform for NTN electric motors and actuators.

* Tatsuji INOUE* Kazuhiro AOSHIMA* Yuichi SUGIMOTO* Keigo MOTOKUBO*

* Business Promotion Department, Electric Module Products Division, Automotive Business Headquarters
(3) SP Series (Fig. 3)
- Brushless DC motor series
- For use in applications that require a thin and high torque rotational actuator. This is based on the hollow motor structure of the BII series but with a reducer in the hollow part.

![Fig. 3 Outlook of SP Series](image)

2) Examples of proposed uses of Electric Module Series
(1) Transmissions as an electronic shift selector (B II Series)
   The shift lever of both AT and CVT are usually connected to the transmission with a mechanical wire, etc. for selecting the shift position. Using the B II Series module it would be possible to electrify the shifting of the transmission.
(2) Electric hydraulic brake (B III Series)
   Applying the coaxial type B III Series with built-in ball screw to the master cylinder shaft contributes to reducing the overall size of an electric hydraulic brake system
(3) Electric variable valve timing control (SP Series)
   Space efficient electric variable valve timing mechanism can be created by incorporating the SP Series into cam shafts
(4) Electric oil pump (SP Series)
   Integrating a thin and highly efficient SP motor and controller contributes to reducing the size of an electric oil pump

With the development of the different Electric Module Series, the software needed to support each Electric Module and application also needed to be developed, which increased man-hours for development and maintenance.

Failure rate was also increased due to the complexity of the electronic system, in addition to increased risk of new failure modes due to the application of new technologies. Out of these backgrounds, the Automotive Standard for Functional Safety, ISO 26262 was issued. It became imperative to build a new process for efficient software development not only to cope with the requirements of ISO 26262 but also to address the need of software support for each application of electric module series.

2. Background of Control Platform Development

The electric module series was developed as a product line-up for applications such as chassis, engine and transmission, and are selected to meet the application specification. Various software versions have been developed for different applications by using the existing asset, which created the following issues:

- Complexity due to many changes, enhancements, and reproduction of code
- Side effect of software changes causing degradation and failure of existing functions

To address the issues of the conventional development approach of using the existing asset and to cope with ISO 26262, we decided to analyze the current situation and introduce two development methods.

1) Development of software product line
   The characteristics of the Electric Module series of products were defined by the external variability (characteristics/values of the system viewed from outside) and the internal variability (elements to compose a system). Using a variety of systems and software it was possible analyze the dependencies and differences between the Electric Module product series. As a result, the software product line was introduced, which is effective at reducing the number of man-hours by creating a core asset of software, and for improving quality by repetitive use of the core asset. This is further described in Section 3.2 in details.

2) Model Based Development
   In general, it is said that validation by simulation during the requirement definition and basic design phases reduces the man-hours of test phase and rework (Fig. 4).
Conventionally, simulation of responsiveness, etc. was conducted by spreadsheet software, which had its own limitations. In order to increase the simulation accuracy and to understand the basic characteristics of a motor at the design phase, JMAG*1 is used for motor design.

It is also required to conduct simulation of the system including controller (motor control part) and plant (JMAG-RT*2 model created from JMAG).

As ISO 26262 also suggests validation using simulation, fault injection test*3, which intentionally creates faults in the model, can be easily conducted in shorter time and at less cost than using actual machines. Validation by simulation can also be conducted at the requirement definition phase. We decided to use the Model Based Development to address the above.

We are currently conducting the development of the Control Platform using these two development methods, which is described in the following Chapter.

*1: JMAG: Simulation software made by JSOL for development of electric equipment design
*2: JMAG-RT: A high-speed and high-accuracy plant model for system-level simulation derived from the FEA model
*3: Fault injection test: Validation to review if the system turns into a safe condition as designed, by intentionally injecting faults

### 3. Control Platform Development

#### 3.1 Overview of Process

The entire process was built with compliance to the functional safety standard ISO 26262, which is subject to quality management, and is based on the internal process that conforms with IATF 16949, adding Automotive SPICE (system development process model for on-board systems) which is highly compatible with ISO 26262 (Fig. 5). The following shows the features.

- Since Automotive SPICE only defines the framework, requirements on methods and safety specified by ISO 26262 are incorporated into the process to make it compliant to the standard.
- Process definition (where work flow is specified) is made available to software developers at any time, using EPF Composer (a tool that defines the development process and converts it to HTML) via the Intranet (Fig. 6).
- The software process is separately defined for core asset development and new product development aligned with the Product Line newly introduced for efficient development of software (Section 3.2).
- The following was introduced mainly for validation of motor control in the Model Based Development (Section 3.3):
  - In the system requirement phase: MILS*4
  - In the software design phase: MILS, RCP*5

*4: MILS: Model In the Loop Simulation; Simulation that operates the control specification written in the model, combining with the plant model
*5: RCP: Raid Control Prototype; Validation of the actual machine subject to control, which incorporates the control specification written in the model into the general purpose controller

---

**Fig. 5** Foundation of the development process
3.2 Software Product Line

The Software Product Line is a method to analyze the products to be developed and to categorize certain portions into the common part and product specific part. Using this method, the common part is defined as assets to be reused and the product specific part is developed efficiently. It is effective in reducing the development period and man-hours while improving quality. In applying the Software Product Line, the scope (range of specifications that covers a product group) was defined as the existing motor series. The deliverables were defined as the software requirements, architecture, detailed design specification, Simulink® model, hand implementation code and test specification. The flow of the Software Product Line is explained in the following, along with Fig. 7.

*6: Simulink®: Block diagram environment for Model-Based Design developed by MathWorks

1) Development of core asset

(1) The common part and variable part of the product group (Fig. 8) are created by the feature model that defines the variability of functions and features which are used to generate the base of the configuration model (model to select the product properties)

(2) Reusable assets are incorporating into the deliverables (requirement specification, architecture, Simulink® model, codes) which helps develop the feature model of the common and variable parts of the product group

2) New product development

(3) The requirement specifications of the new product are classified into “common part/variable part” and “product specific part”

(4) The common/variable parts of the requirement specification of new products are selected from the configuration model

(5) The configuration model created in (4) and based on the reusable assets created in the core asset development (2) is loaded into the Binding Tool. This is then used for binding and deliverables based on the required specification of the new product

(6) The deliverables for the product specific part are newly created

(7) The deliverables to conform to the new product requirements are created by combining the deliverables extracted from the core asset (5) and the deliverables newly created for the product specific parts (6)
Tools could be created in-house in a short time, by using FeatureIDE an open source product for creating the feature model and configuration model and developing the Binding Tool.

We used a sample core asset for binding (5) and verified that the required deliverables were extracted. With this, we completed creating the mechanism of the process.

Currently, we are converting the requirement specifications and architectures of the existing products into parts to be reused. In this process, since the difference (variability) needs to be considered for design and implementation, some of them need to be modified before turning them into parts.

In the Product Line, since specifying and managing the variability that exist in the product group are important, we plan to thoroughly analyze the differences (variability) of the software platform for functional safety under development to make the design easily reusable.

### 3.3 Model Based Development

In order to verify the system to conduct simulation of the controller model, where the motor control algorithm and the controlled objects are incorporated into the plant model, the Model Based Development was applied.

In the process definition, the process was identified with system requirement review phase, software architecture design phase and simulation (MILS, RCP) in the detailed design phase (Fig. 9).

#### 3.3.1 Result of Simulation and Future Plan

As the target of simulation, in MILS, JMAG and RCP, the relative error of N-T and I-T characteristics, which are regarded as the basic characteristics of the motor, were set to ±10%.

Firstly, the motor control (vector control) algorithm was modeled reverse engineered from the existing C source code, to create the controller model by Simulink®. The vector control algorithm that was modeled was bound to the motor model (JMAG-RT) in order to conduct MILS with the general purpose controller. This control was connected to the inverter and motor to be evaluated and RCP was conducted. The comparison of N-T and I-T characteristics from MILS and RCP (Fig. 10) revealed that the I-T characteristics almost matched and the N-T characteristics were within ±10% when no load was applied but as the current increased, the N-T characteristics exceeded ±10%. Since the motor phase voltage dropped by 1.73V against the power source voltage of 12V, the voltage drop may have occurred due to the internal resistance or wiring resistance in the inverter used in RCP (Fig. 11). We plan to investigate the cause of this error by comparing the theoretical calculation and the actual measurement of the voltage drop due to internal resistance, etc. and improve the model.
3.3.2 Result of Controller Model and Future Plan

In the controller model, we are promoting the use of the core asset (reusable asset) anticipating the forthcoming auto code generation. As one subsystem meets multiple requirements, the subsystems were recreated to correspond to each requirement in the requirement specification.

Then, the switch of subsystems where made in three patterns based on “with/without” requirement. The three patterns determine the binding time depending on “when” the variability is realized for each subsystem (Fig. 12).

1) Required model elements are edited (added/deleted) by the Matlab script
2) Required model is selected by the Variant SubSystem and Configurable SubSystem
3) Processing is dynamically selected by If/Switch Case SubSystem and Switch block

Switch is applied at the design phase for 1), code generation or compiling phase for 2) and software execution phase for 3). The model unit test was defined in the process to verify the output result per SubSystem.

By conducting the simulation test per subsystem, faults can be removed before implementation with auto code generation. The model based development comes with a rich set of test tools; however, in order to eliminate variance of quality of manual tests and for efficiency purposes, a tool for automatic testing that works with different tools was developed. This enabled automatic generation of test specifications, simulation models and test results, easy retesting and recording of evidence (Fig. 13).

For applying auto code generation to the products in the coming years, we will prepare the environment and process for conducting verification of consistency between the model and codes (Back to Back testing) required by ISO 26262 (SILS, PILS).
Development of Control Platform for Electric Module

4. Conclusion

NTN will continue developing electric module products, which can be applied to various applications efficiently, taking advantage of our base technologies, and contribute to the promotion of autonomous driving and electrification, which is evolving in the automotive industry.

Photo of authors

Fig. 12 Relationship of replacement measures between the binding time and Simulink® subsystem

Fig. 13 Flow of unit test tool of automatic model