講演番号	320

文献番号	20225320

Approach to improve the efficiency of software development for Advanced Driver Assistance Systems by using Continuous Integration (CI)

Takuro Yuhara [1)] Keisuke Sato [1)] Tomonori Nambu [1)] Kazuma Sugimoto [1)]

1) Nissan Motor Co., Ltd.
560-2, Okatsukoku, Atsugi-shi, Kanagawa, 243-0192, Japan

KEY WORDS: common infrastructure, information management, Continuous Integration, ADAS [F2]

Continuous Integration (CI) is one of software development method that performs automated build and test in case of software changes, and is widely used in mobile application and open source software (OSS) development. The software of the Advanced Driver Assistance Systems (ADAS) has numerous variations due to diversifying systems, functions, and correspondence to the regulation for each region. And to meet customer needs (Time to Market), the efficient software development with keeping quality is important. In this paper, for efficiency software development, CI was applied to ADAS software, and then confirmed the effectiveness.

In applying CI, it is necessary to carry out validation required from ISO26262 (functional safety), to select appropriate functions and constants from many software variations, and to have a development scheme for collaboration with multiple software developers. To achieve these points, the CI system was constructed to automate the validation process and selection of functions and constants according to software variations, and also to unify management of all software. Fig. 1 shows CI system overview. This CI system consists of a CI Server that automatically executes software validation, a Variation Management System that manages software variation information, and a Version Control System that unifies management of all software. Its details are as below.

First, a series of automation processes (CI Pipeline) was constructed to automate software unit test and integrated test. This CI Pipeline works with a Version Control System to execute a series of software validations, triggered by software change. If validation result is failed, the error report is notified immediately to the software developers. Second, the Variation Management System that associates the information of ADAS software variation, vehicle spec and constant parameters was constructed. This management system sends the list of functions and constant parameters for target vehicle to the CI Server when the unique identifier of target vehicle is input. CI Server configures the constants and build the software with the functions based on this list. Furthermore, by managing a large number of parallelly developed software with a common Version Control System, it has become possible to collaboratively develop with multiple software developers. These were constructed on the cloud environment that can be accessed globally.

This CI system was applied to ADAS software development, and confirmed the effectiveness with a focus on the number of software issue and the detecting timing.

To confirm effect on productivity by automated validations, the total number of issues, which occurred in the software development phase, was compared before and after applying CI. As the result, the number of software issue was reduced approximately 27% (16/59 cases) and man-hours for correction were reduced approximately 23%.

In addition, confirmed how the detection timing of issues had been changed when validations were automated. Here, the change was evaluated by the reliability growth curve, which is one of the metrics of software quality evaluation. Fig.2 shows the comparison of reliability growth curve. A relative value was compared with the total number of issues detected in the software test phase and system test phase as 100%. The number of issues detected in the software test phase was increased. Therefore, the development rework in the system test phase was reduced, and the development efficiency was improved.

Finally, by using CI in this manner, software development can be streamlined, and ADAS software with many variations can be released in a timely manner while improving quality. However, the effect on productivity and the ability to release software depend largely on the complexity of the software and the processing performance of CI system. Further studies are required for improving the performance such as parallel processing of CI system for more complicated software.

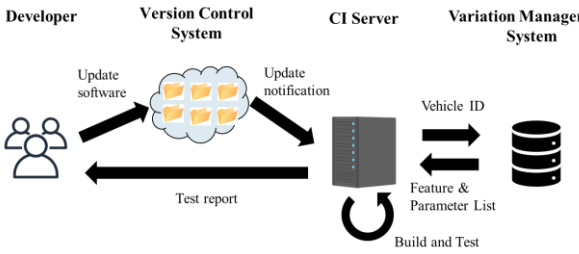

Fig.1 CI system overview

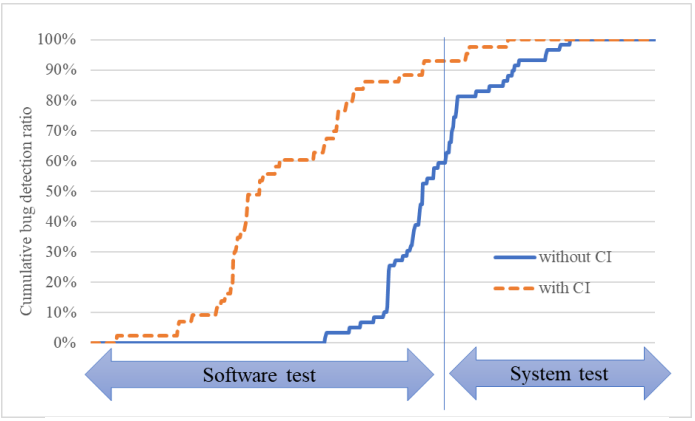

Fig.2 Comparison of software reliability growth curve