JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Improvement of the Automobile Control Software Testing Process Using a Test Maturity Model

Jin-Wook Jang*

Abstract

The problem surrounding methods of implementing the software testing process has come under the spotlight in recent times. However, as compliance with the software testing process does not necessarily bring with it immediate economic benefits, IT companies need to pursue more aggressive efforts to improve the process, and the software industry needs to makes every effort to improve the software testing process by evaluating the Test Maturity Model integration (TMMi). Furthermore, as the software test process is only at the initial level, high-quality software cannot be guaranteed. This paper applies TMMi model to Automobile control software testing process, including test policy and strategy, test planning, test monitoring and control, test design and execution, and test environment goal. The results suggest improvement of the automobile control software testing process based on Test maturity model. As a result, this study suggest IT organization's test process improve method.

Keywords

Automobile Control Software, Risk-based Test, Software Testing Process, Test Design, Test Planning, Test Policy and Strategy, TMMi Assessment

1. Introduction

As the size and complexity of software is increasing continuously, the software industry is showing a great deal of interest in quality in relation to software product reliability and productivity. Interest in quality has spread from standard-oriented software test activity, such as the ISO/IEC 29119, British Standards Institution (BSI) and IEEE, to the importance of the software test process [1]. Test Maturity Model integration (TMMi), Test Process Improvement (TPI), Test Organization Maturity Model (TOM), and Test Improvement Model (TIM) have all been defined as software test process improvement and diagnostic models [1,2]. These concepts are being developed continuously, and the software industry is showing a lot of interest in them.

However, compliance with the software testing process does not necessarily bring about immediate economic benefits such as power cost, schedule compliance, and enhanced reusability. Therefore, IT organizations need to pursue more aggressive efforts to improve the process.

Software tests are generally taken informally or are omitted due to the short life cycle of software and the time taken to reach the market in the software industry. In order to ensure the quality of software, it

is necessary to establish a software testing program taking full advantage of the limited resources, and to apply systematic software testing techniques to the tools and methodologies of the development process.

This paper proposes a novel concept of and standard for test process improvement models destined for application to automotive control software, and infers the core area required for the testing process of automobile control software based on TMMi models. Beyond that, the test process of each core area is evaluated with a checkpoint. The merits and demerits of the test process were analyzed according to the results of the evaluation thereof, and testing process building plans for each area were proposed [2].

2. Related Work

2.1 Testing Process Model

TMMi and TPI are representative examination-oriented test process models that are widely used in the test process maturity evaluations of test execution organizations, and their usability has been proven in many different forms. Both models share various similarities in terms of the test process examination, but they also have some differences (Table 1). Below table will be helpful in improving and conducting the test process evaluation when the professionals concerned choose their evaluation model [2,3].

Table 1. TPI and TMMi

Division	TPI	TMMi
Type	Continuous model	Step-by-step model
Test methods	Associated with TMap	Separate from other test methods
SPI	No official relevance to the SPI model	Very relevant to CMMI
Test levels	Supports high level in particular	Supports all test levels
Focus	20 key areas	Each maturity level focuses on a limited number of process areas
Approach	Test engineering	Strong focus on efforts of management organization

TMMi is a 5-level Staged Evaluation Framework that describes the key elements of an effective test process. It is related to Capability Maturity Model Integration (CMMI), and is used to improve the test process. It is a test process examination model of software development. Fig. 1 shows that TMMi has a similar structure to the Staged model.

Fig. 1 shows that the TMMi model's key elements of the effective test process are divided into 5 levels. As a highly qualified model, the TMMi model has the merits of high consistency and a perfect maturity model structure. The following diagram shows the details of each level [4].

2.1.1 Level 1 initial

At TMMi level 1, testing is a chaotic, undefined process, and is often considered a part of debugging. As most organizations do not provide a stable environment to support these processes, success depends

on the competence and confidence of the people at any given organization, rather than on the use of proven processes. Tests are developed in an ad-hoc way upon completion of the coding; and testing and debugging are interwoven to eliminate bugs from the system.

Fig. 1. Test Maturity Model integration (TMMi) model.

The objective of testing at this level is to show that the software in question is capable of running without any major failures, particularly as products are often released without adequate visibility regarding quality and risks. In this field, a product does not often fulfill the needs for which it is intended, and is neither sufficiently stable nor fast enough to work with. Within testing there is a lack of resources, tools and well-educated staff. At TMMi level 1, there are no defined process areas. Maturity level 1 organizations are characterized by a tendency to overcommit and to abandon the processes in times of crisis, and by an inability to repeat their successes. Furthermore, products tend not to be released on time, while budgets tend to be exceeded and quality fails to meet expectations [5].

2.1.2 Level 2 managed

At TMMi level 2, testing becomes a managed process and is clearly separated from debugging. The process discipline reflected by maturity level 2 helps to ensure that existing practices are maintained during times of stress. However, testing is still perceived by many stakeholders simply as the project phase that follows coding. In the context of improving the test process, a company-wide or program-wide test strategy is being established, and test plans are also being developed. Within the test plan a test approach is defined, whereby the approach is based on the result of a product risk assessment. Risk management techniques are used to identify product risks based on the documented requirements. The

test plan defines what kind of testing is required, as well as when, how and by whom. Commitments are established with stakeholders and revised as and when required. Testing is monitored and controlled to ensure it is going according to plan, and appropriate actions can be taken if deviations occur. The status of products and the delivery of testing services are visible to the management. Test design techniques are applied to derive and select test cases from the specifications. However, testing may still start relatively late in the development life cycle, e.g., during the design or even during the coding phase. Testing is multi-leveled, i.e. there are unit, integration, system, and acceptance test levels; and for each identified test level there are specific testing objectives, as defined in the organization-wide or program-wide test strategy. The main objectives of testing at TMMi level 2 organizations are to verify that the product concerned satisfies the specified requirements, and to clearly differentiate the testing and debugging processes. However, many quality problems arise at this TMMi level because testing takes place quite late in the development life cycle, and defects are consequently propagated from the requirements and design to the code. Unfortunately, there are still no formal review programs with which to address this important issue; and post code execution based on testing is still regarded as a primary testing activity by many stakeholders.

The process areas at TMMi level 2 are Test Policy and Strategy, Test Planning, Test Monitoring and Control, Test Design and Execution, and Test Environment.

2.1.3 Level 3 defined

At TMMi level 3, testing is no longer simply a phase that follows coding, but rather is fully integrated into the software development life cycle and the associated milestones. Test planning is performed at an early project stage, e.g., during the requirements phase, by means of a master test plan, the development of which builds on the test planning skills acquired at level 2. The organization's set of standard processes, which is the basis for maturity level 3, is established and improved over time. A test organization and a specific test training program exist, and testing is perceived as being a profession. Test cases are gathered, stored and managed in a central database for re-use and regression testing. Basic tools support key testing activities. Organizations at this level have begun to realize the importance of including reviews in quality control, and a formal review program is being implemented, although it has not yet been linked to the dynamic testing process. Thus, reviews are taking place across the life cycle, and testing professionals are involved in reviews of the requirement specifications. The test designs at TMMi level 2 focus mainly on functionality testing, and test designs and test techniques are being expanded—depending on the business objectives—to include non-functional testing, e.g., on usability and/or reliability.

A critical distinction between TMMi maturity levels 2 and 3 concerns the scope of the relevant standards, process descriptions, and procedures. At maturity level 2, these may be quite different in each specific instance, e.g., depending on a particular project. At maturity level 3, however, these are tailored from the organization's set of standard processes to suit a particular project or organizational unit, and are therefore more consistent, except for the differences allowed by the tailoring guidelines. Another critical distinction is that at maturity level 3 these processes are typically described more rigorously than at maturity level 2. As a consequence, an organization must revisit the maturity level 2 process areas at maturity level 3.

The process areas at TMMi level 3 are Test Organization, Test Training Program, Test Life Cycle and

Integration, Non-Functional Testing, and Peer Reviews.

2.1.4 Level 4 management and measurement

In TMMi 4 organizations testing is a thoroughly defined, well-founded and measurable process. Furthermore, at maturity level 4, the IT project organizations establish quantitative objectives for product quality and process performance, and then use them as the criteria for managing them. Thus, product quality and process performance are understood in statistical terms and are managed accordingly throughout the entire life cycle.

Various measures are incorporated into the organization's measurement repository in order to support fact-based decision making; and reviews and inspections are considered an integral part of testing and used to measure document quality. Thus, static and dynamic testing approaches are integrated into one; reviews are formally used as a means to control quality gates; and product quality is evaluated using quantitative criteria for such attributes as reliability, usability and maintainability. An organization-wide test measurement program provides information and visibility regarding the test process. Testing is perceived as evaluation; and consists of all life cycle activities concerned with checking products and related work products. The process areas at TMMi level 4 are Test Measurement, Product Quality Evaluation, and Advanced Peer Reviews.

2.1.5 Level 5 optimization

On the basis of the results achieved by fulfilling all the improvement goals set at the previous maturity levels, testing is now a completely defined process that is capable of controlling both costs and testing effectiveness. At TMMi maturity level 5, an organization continually improves its processes based on a quantitative understanding of the common causes of the variations inherent in those processes. Improvements of the test process performance are carried out through an incremental and innovative process and technological improvements; and the relevant methods and techniques are optimized, with a continuous focus on fine-tuning and test process improvement.

Defect prevention and quality control are also practiced, while statistical sampling, measurement of confidence levels, trustworthiness, and reliability drive the test process forward. Among other things, "Defect Prevention" and "Quality Control" are introduced as the process areas. The test process is characterized by sampling-based quality measurements. A detailed procedure exists for selecting and evaluating the test tools, which are used to support the test process as much as possible during test design, test execution, regression testing, test case management and so forth. Process reuse is also practiced at level 5, supported by a process asset library. Testing is a process whose principal objective is to prevent defects. Thus, the process areas at level 5 are Defect Prevention, Test Process Optimization, and Quality Control.

2.2 ISO/IEC 29119 Testing Standards System

The ISO/IEC 29119 software testing standard is currently being established by the international standardization working group [4], the contents of which are shown in Fig. 2. Each of the parts is based on the international standards mentioned below: Part 1 is from BS 7925-1; Part 2 is from BS 7925-1 and IEEE 1008; Part 3 is from IEEE 829; and Part 4 is from BS 7925-2 [6,7].
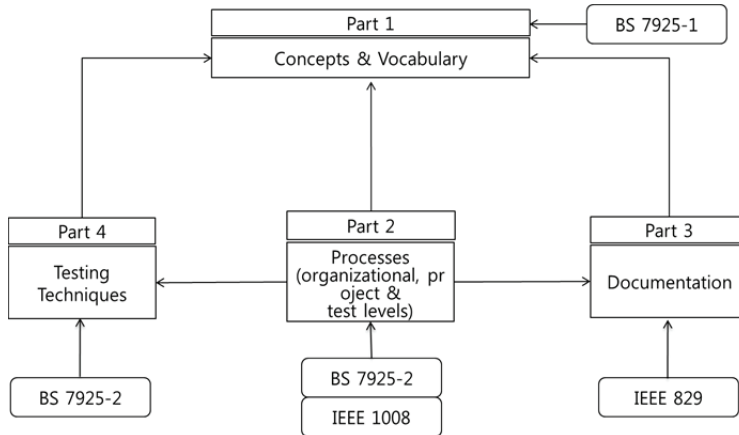
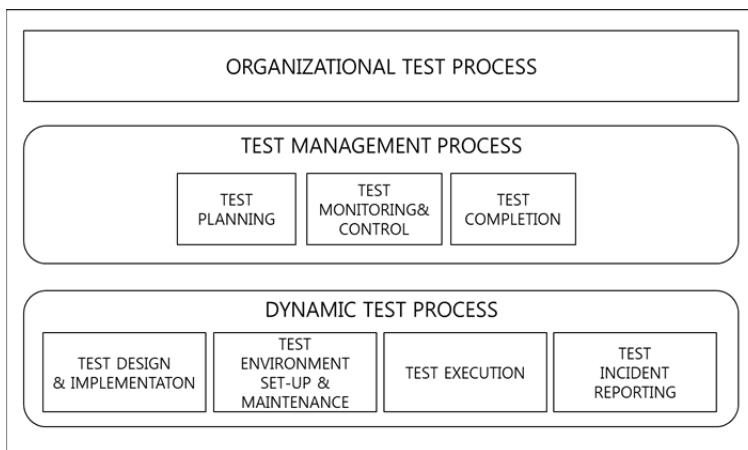**Fig 2.** Software testing standard of ISO/IEC 29119.

Fig. 3. Three layer model of test process.

Table 2. Software testing standard

Division	Contents
Part 1: Overview and term	-Definition and introduction of software testing; -Defining a relationship among testing, development, and maintenance.
Part 2: Process	-Organizational test policy process -Organizational level test strategy process -Project test managing process -Test level process
Part 3: Documentation	-Test management documentation (test policy, strategy, project completion report) -Test documentation (test plan, specification, result, defect report, test level completing report) -Test status reporting (test-level status report, test environment report)
Part 4: Test technique	-Test case design technique (static, dynamic, and non-functional test techniques) -Test measurements (coverage)

Table 2 shows the details of Part 1, Part 2, Part 3, and Part 4. Part 5 needs to be standardized through a new proposal [8-11].

Fig. 3 shows the structure and relationship of each process schematically as a 4-layer mode. These consist of a test policy and a test strategy process at the organization level, a project test management process at the project level, and a test level process at the test level. As shown in Fig. 3, these processes are organically interrelated with each other [12].

When a company devises their strategy for the project test plan, the companywide test strategy must be reflected in it. Furthermore, each level test should be linked organically with the top level project test management activities. Consequently, developer, planner, project manager, and the other concerned exchange feedback each other.

3. Test Maturity Model Integration-Based Assessment

The purpose of this study is to change our perception of software testing process improvement activities and to suggest ways of systematically improving the software testing process.

This study uses a checklist based on TMMi in order to identify and analyze a software company's level of competence regarding software testing processes, which show different scales and forms. The checklist is a structured questionnaire designed to achieve TMMi level 2. Each key area or goal reflects the following five core areas: Test Policy and Strategy, Test Plan, Test Monitoring and Control, Test Design and Execution, and Test Environment. The results of each question provide important information for the on-site examination. For this reason, four to ten respondents should be included for the maturity question, and project leaders must participate in it. The response to each questionnaire is either "Yes" or "No", and the surveyor's comments should also be recorded on the questionnaire. To ensure that the respondents fully understand all of the questions, the vocabulary used in the questionnaire should be general, and an explanation of each technical term should be provided if necessary. Table 3 shows the key areas of question research.

The testing organization's maturity and level of improvement of its business capacity will be measured, and areas for reform will be derived by applying the questionnaire.

Table 3. Software testing Key area

Key area	Contents
Test policy and strategy establishing process (KA 1)	Policy based on test purpose, strategy planning, capacity indicators, and 21 other items.
Test planning process (KA 2)	Test target list, mode of approach, estimate, planning review, and 19 other items.
Test monitoring and control process (KA 3)	Monitoring of test progress, monitoring of product quality against planning, performance management, and 11 other items.
Test design and execution (KA 4)	Test analysis design, development, implementation, incident (difference between actual results and expected results), and 18 other items.
Test environment process (KA 5)	Requirements analysis of test environment, environment setup, environment management, and 10 other items.

4. Application and Evaluation

In this paper, the software testing process and testing skills for automobile control software with the Intelligent Power Module (IPM) were applied. As the IPM was first made available in 1990, it was not necessary to design any previous actuation circuits or safety circuits, which meant it was possible to insulate the control circuit with a low-speed photo coupler.

Fig. 4 shows the IPM, an item of automobile control software that diagnoses defects and transfers data by using Controller Area Network (CAN) way communication. The Under Hood (UH) Box and the Body Control Module (BCM) use CAN way communication. The UH performs the load control and power distribution function on the subsystem and the module needed to automobile control power. The BCM is a communication module for the automobile controller.
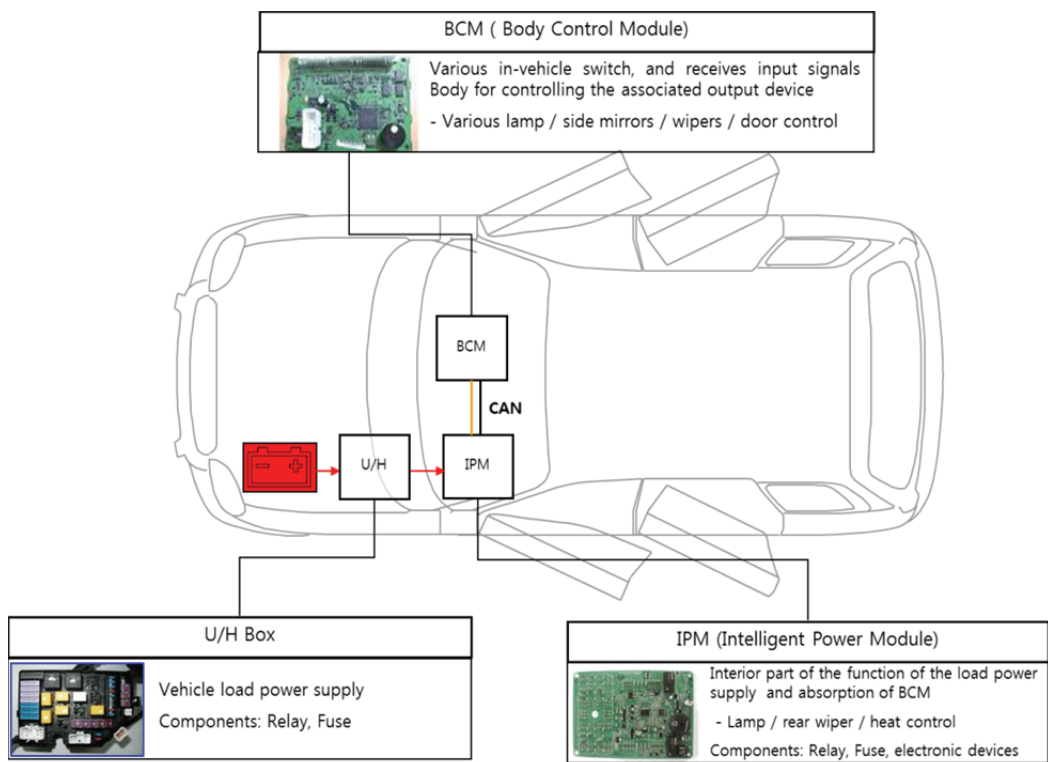

Fig. 4. Installation location of the IPM.

4.1 Establishing the Testing Policy

The testing policy in the ISO/IEC 29119 model is made to establish the rules and implementation of testing, and to define the software testing range of an organization. If the personnel of the organization concerned present the basic framework based on the test policy and manage the testing task, they can improve test process continuously [12]. Fig. 5 shows how the main stakeholders reach an agreement with the feedback obtained after reviewing the testing policy [13].
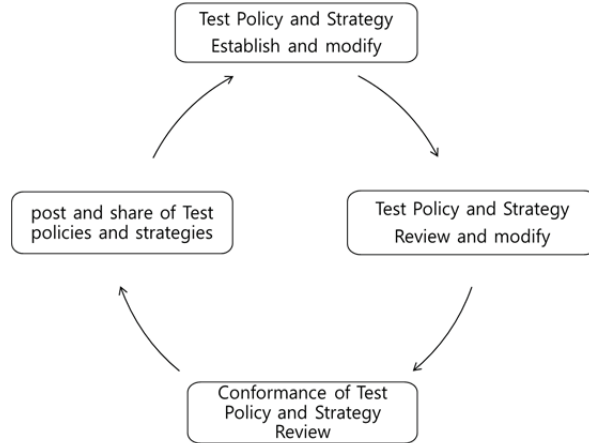
Fig. 5. Test policy process.

4.2 Establishing the Testing Strategy

The testing strategy in the ISO/IEC 29119 model suggests the basic framework so that the stakeholders can manage the test based on the test strategy. Fig. 6 shows how the main stakeholders reach an agreement with the feedback obtained after reviewing the testing strategy. The test strategy derived from the agreement becomes the basic foundation for all the testing tasks within an organization.
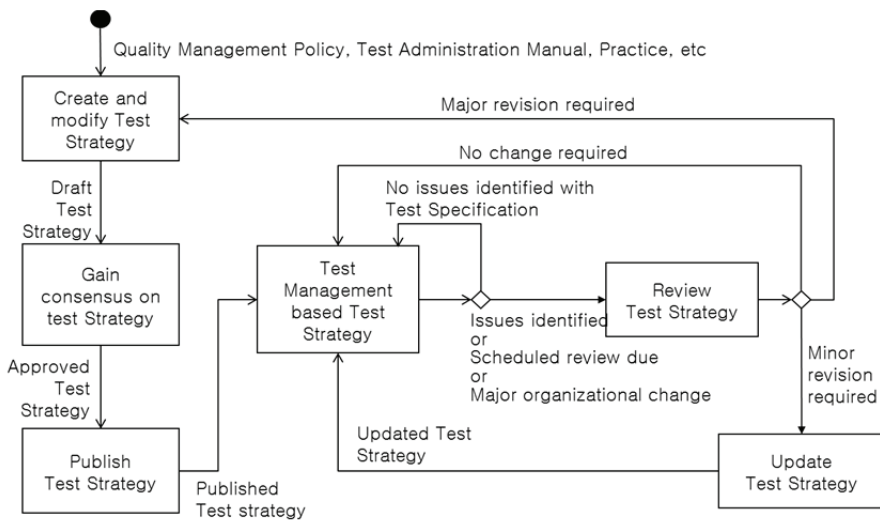
Fig. 6. Test strategy process.

4.3 Establishing the Project Test Management Process

Organizations generally have more than two level tests. The purpose of establishing the project test process is to define and manage the relationship between these level tests from a broader point of view. Fig. 7 shows that it is a process that encompasses the entire project test from project test planning to

project test completion. The overall level tests include the monitoring and control functions.

As shown in Fig. 8, the test levels of the automotive control software project were defined as hardware (HW) unit, HW integration, software (SW) unit, SW integration, HW and SW integration, HW and SW systems, and HW and SW integration testing of vehicles.

Fig. 7. Project test management process.

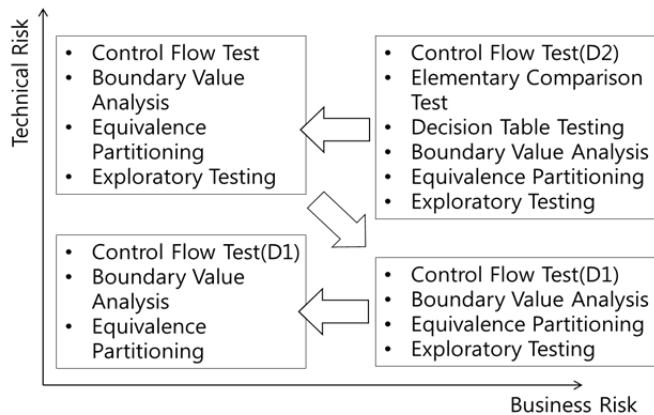**Fig. 8.** Test levels of IPM project.

Fig. 9. Application of risk-based test design techniques.

For the risk-based approach to testing, the project stakeholders derived an agreement. The risk factors and risk items were identified, and the risk scores of each stakeholder were collected. As shown in Fig. 9, a different test planning technique and completion conditions were applied in each risk area according to the priority of the results of the risk analysis. The IPM project internalized the risk-based strategy through this process. Thus, the planning techniques applied to the IPM project differed depending on the different risks; for example, a more powerful test technique was applied for areas of high risk [14].

4.4 Establishing Sleep Mode Test Case Design

In this paper, the sleep mode flow was applied to the test design. It is the most important function of the "Power Down" mode module in the IPM project because it prevents loss of power from the battery. Fig. 10 shows the control flow of the sleep mode.

The microcontroller unit (MCU) is a module that controls power and parts to promote the effective

use of power. Sleep Mode is settled upon the communication result between MCU and four modules. The four modules are indicated simply as Switch, CAN, Output, and Timer. To ensure effective testing, modified condition decision coverage (MCDC) was used during the test [15], to derive a logical test case for a high level design. Pairwise testing was used to make up for the MCDC's specificity, wherein it does not consider cases involving the conversion of more than two switches at the same time.
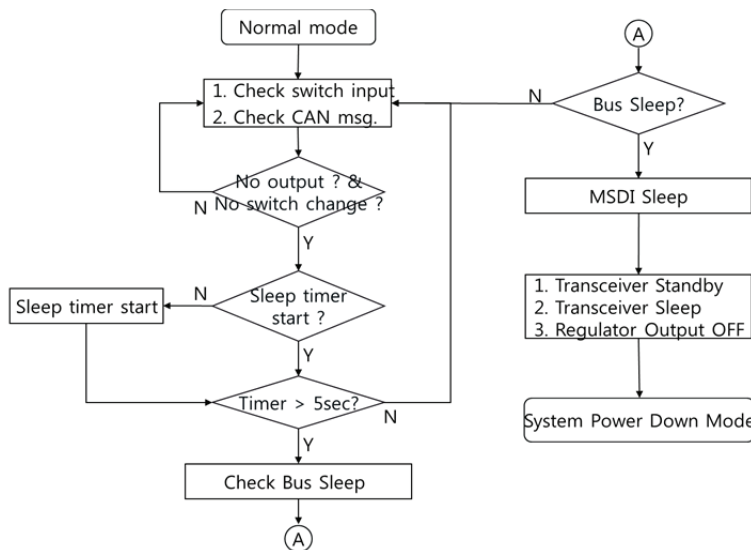

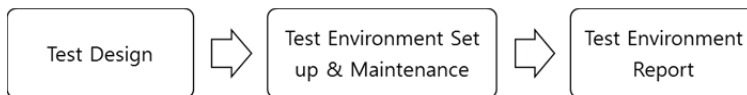**Fig. 10.** Sleep mode flow.

Fig. 11. Test Environment Process.

4.5 Establishing the Test Environment

Fig. 11 shows the Test Environment process, which consists of the hardware, instrument, simulator, software tool and other elements needed to execute the test design.

4.6 Improvement Using TMMi Assessment

Improvement strategies were derived for each of the key areas on the basis of TMMi level 2. Table 4 shows the diagnostic results. It was necessary to establish the overall test process, such as KA1 and KA2, and then connect it with the software developing process. KA1 refers to the test policy and strategy establishment process, while KA2 refers to the test plan process. Based on results of the TMMi diagnostic, improving priority on test was defined by analyzing the urgency, importance and ripple effect on each task. As shown in Table 4, the test purpose was clarified by establishing the automotive control software testing policy and a risk-based testing strategy. It is a precedence task in a short-term improvement task. In addition, software project member's ability was improved through test instruction and training, and a test design technique application was deduced, Table 5 shows MCDC coverage.

Table 4. Analysis and deduction of improvement tasks

Analysis	Term	Improvement task
	Short	1. Test strategy and policy
		2. Test planning and monitoring & control
		3. Test defect management and matrix
		4. Test design and coverage
	Mid-term	5. Test training
		6. Test and development life cycle
		7. Product quality monitoring
		8. Test environment management

Table 5. Applying and complementing of MCDC coverage

Test suite	Switch	CAN	Output	Timer	Expect results	Etc.
1	1	1	1	1	Sleep	-
2	0	1	1	1	Normal	-
3	1	0	1	1	Normal	-
4	1	1	0	1	Normal	-
5	1	1	1	0	Normal	-
6	1	1	1	0	Normal	Complementary technique
7	1	1	1	1	Normal	Complementary technique

In order to internalize the deduced improvement task within an organization, it is necessary to make the following improvements in each area. The evaluation methods and Table 4 show the priority which each participant thinks is more important, and the 3-point method in evaluation methods (1) is used on that analysis, the results of which are shown in Table 4.

$$3 - point\ method = \frac{\sum(max + (4 \times average) + min)}{6} \tag{1}$$

① Test Policy and Strategy establishment process

Although the basic framework for establishing the test policy and strategy already existed, software project members do not clearly understand the basic framework and the mutuality between the policy and the framework. In other words, it was necessary to connect the plan and the performance step as the next step in making the policy and the strategy.

② Test Planning process

Risk analysis should be reflected in the plan by analyzing the project risk rate and the function of a software product. The test items should be selected based on the priority risk, and the risk analysis should be defined how to apply on test method [16]. If the life cycle of the entire test process is not clearly defined in the test plan, then the following steps should be taken: The process from planning to execution should be planned for the life cycle of the test plan; the Test Execution Phase on Work

Breakdown Structure (WBS) should be included in the test plan; specific actions for each phase should be defined; and resource management for execution of the test plan should be accomplished. Additionally, it is essential that every stakeholder should review the test plan, and the results thereof should be documented [17].

③　Test Monitoring and Control process

To manage the test schedule and communication effectively, it is necessary to draw up a test progress report. More specifically, the following actions should be taken: the final report of each step should be reported in time; examples of the report are the project plan report, test strategy report, status report, and improvement report for tool applying; and the test work should be reviewed because it facilitates regular monitoring and control of the overall testing activities [18].

④　Test Design and Implementation

Test design and implementation were not performed at each level. However, specific requirements and test cases must be traceable [19].

⑤　Test Environment process

Requirements concerning the test environment are documented, but they are limited to the integrating test. Accordingly, it should be more specific.

TMMi-based inspection has limitation in that it does not show the maturity of each process, but rather only shows the maturity of the overall organization. Each test process can show a different level of maturity within an actual testing organization [20]. In addition, specific items, such as the checklist for TMMi inspection, are opened to the public restrictively, and each specialist evaluates it according to his or her own checklist and method. For these reasons, a standard checklist is required for TMMi inspections. In fact, there is a nonprofit organization called the TMMi Foundation that plays a leading role in resolving these problems, partly by establishing standards for inspection methods, evaluation items and evaluation processes. It also trains senior examiners and manages their qualifications.

5. Conclusion

This paper suggests that the software testing process has a correlation with the development process, and affects productivity directly. In other words, in order to apply the software testing process to the development process more easily, it needs to be more practical than any other process.

This paper deduces a testing process improvement plan suitable for practical business from the results of an analysis of a software test organization. The testing process improvement plan is divided into short-term and mid-term plans according to priority. Professional software testing knowledge and technologies are helpful in improving product quality and reliability when software testing professionals apply them to practical work, as they can secure the basis for quantitative evaluations of software quality.

The test process examination results can be helpful to an organization that uses them to establish its priorities, and then internalizes such priorities in its short-term tasks, determines the developmental direction for its mid-term tasks, and finally utilizes them in its next process improvement plan. This paper suggests a way to apply and set up the practical process for improving the quality of the automobile control software.

References

[1] E. van Veenendaal, *TMMi and ISO/IEC 29119: Friends or Foes?*. Dublin, Ireland: TMMi Foundation, 2016.

[2] E. Van Veenendaal, *Test Maturity Model integration (TMMi)*. Dublin, Ireland: TMMi Foundation, 2012.

[3] T. Koomen and M. Pol, *Test Process Improvement: A Step-by-Step Guide to Structured Testing*. London: Addison-Wesley, 1999.

[4] M. B. Chrissis, M. Konrad, and S. Shrum, *CMMI for Development: Guidelines for Process Integration and Product Improvement*, 3rd ed. London: Addison Wesley, 2007.

[5] TMMi Foundation [Online]. Available: http://www.tmmifoundation.org.

[6] E. van Veenendaal, *Standard Glossary of Terms Used in Software Testing*. Brussels, Belgium: International Software Testing Qualifications Board, 2010.

[7] *IEEE Standard for Software Test Documentation*, IEEE 829-1998, 1998.

[8] ISO/IEC 9126-1:2001, *Software engineering – Product quality – Part 1: Quality model*, 2001.

[9] *Information technology - Software process assessment - Part 9: Vocabulary*, ISO/IEC TR 15504-9:1998, 1998.

[10] IEEE 1028-1997, *IEEE Standard for Software Reviews*, 1997.

[11] ISO/IEC 12207:1995, *Information technology - Software life cycle processes*, 1995.

[12] M. Niazi, D. Wilson, and D. Zowghi, "A maturity model for the implementation of software process improvement: an empirical study," *Journal of Systems and Software*, vol. 74, no. 2, pp. 155-172, 2005.

[13] E. Jung, J. H. Hwang, J. K. Lee, and G. B. Joung, "New reliable inverter with intelligent power module," *2015 9th International Conference on Power Electronics and ECCE Asia (ICPE-ECCE Asia)*, Seoul, Korea, 2015, pp. 2730-2736.

[14] International Software Testing Qualifications Board, "Certified Tester: Advanced Level Syllabus," 2012 [Online]. Available: http://castb.org/wp-content/uploads/2013/09/advanced_syllabus_2012_test_analyst_ga_release_20121019.pdf.

[15] A. Dupuy and N. Leveson, "An empirical evaluation of the MC/DC coverage criterion on the HETE-2 satellite software," in *Proceedings of the 19th Digital Avionics Systems Conference*, Philadelphia, PA, 2000, pp. 1B6-1.

[16] ISO/IEC/IEEE 29119-2, *Software and system engineering – Software testing – Part 2: Test processes*, 2013.

[17] J. Bach, "James Bach on risk-based testing," *Software Testing and Quality Engineering*, vol. 1, no. 6, pp. 22-29, 1999.

[18] ISO/IEC/IEEE 29119 Software Testing [Online]. Available: http://www.softwaretestingstandard.org/.

[19] I. Burnstein, T. Suwanassart, and R. Carlson, "Developing a testing maturity model for software test process evaluation and improvement," in *Proceedings of International Test Conference*, Washington, DC, 1996, pp. 581-589.

[20] B. Beizer, *Software Testing Techniques*, 2nd ed. New York, NY: Van Nostrand Reinhold, 1990.

Jin-Wook Jang https://orcid.org/0000-0002-3605-9157

He received in Ph.D. of Management Engineering from Konkuk University. He worked in Korea Defense Intelligence Command of Ministry National Defense as a Computational officer and in SK communications as a PMO Manager. He is Professor of College of Liberal Arts, Anyang University. The main interests are project management, software quality, testing process, etc.