Attaining Higher Quality by Frontloading Approach

● Toru Shiraiwa ● Kimio Saito ● Keita Uchiyama

(Manuscript received January 12, 2007)

From the time of the mainframe era, Fujitsu has been supporting many mission-critical information systems in Japan by providing high-quality software. This tradition is also alive on today's open platforms; however, Fujitsu has had to overcome new constrains that have been introduced by the open platforms. This paper first describes Fujitsu's approach to developing high-quality middleware, which is characterized by integration of various efforts into a quality system. It then describes the frontloading approach, which aims at managing the constrains by assuring quality from the upstream development stages. In this paper, the term "frontloading" is used to indicate the efforts to assure high quality by acting on the early development stages.

1. Introduction

As a partner to its customers, Fujitsu has been responsible for the development and maintenance of many mission-critical systems that support the social infrastructure of Japan. Fujitsu is proud to provide the best products and services in large-scale mission-critical business systems. This was made possible by our ability to produce high-quality operating system (OS) and middleware products. This paper describes how the method of developing high-quality software has been changed for the open platform environments. The changes are studied in terms of the quality assurance process, development environment, and human resource development, focusing on improvements made in each area.

2. Changes in development process

This section describes the changes in the business environment and the corresponding changes in the development process, with an emphasis on quality assurance (QA).

2.1 Customers' expectations for Fujitsu middleware

Fujitsu has many customers who support the infrastructure of Japan, and they need to ensure continuous operation of their mission-critical business systems. Banking companies and stock exchanges are some examples of these customers. Once in operation, these critical systems need to provide:

1) Uninterrupted services
2) Continuous operation over very long periods

For these customers, Fujitsu's middleware must be of the highest quality that guarantee stable, uninterrupted services. For continuous operation over long periods, it is also necessary to absorb differences in the lower layer and provide a compatible operating environment for the application layer, even if the hardware or operation systems are changed. As more and more open systems are introduced into

business systems, customers are also requiring that middleware products be usable in a heterogeneous environment.

2.2 Changes in development methodology

2.2.1 Development methods for mainframes

Fujitsu has excelled in the development of high-quality software for the mainframe platform. The hardware for the mainframe platform is developed under a single architecture, making it possible to concentrate solely on a single platform. Also, external requirements such as customer needs and technology changes are stable, so it is possible to develop according to a development plan without much change. Under this environment, development is done in the waterfall style, in which the quality at each stage is ensured before proceeding to the next stage (**Figure 1**).

In this method, development is defined by a development plan document. The outputs of each stage, for example, the project proposal and functional specification, are reviewed by all related engineers to ensure the quality of design documents. These outputs are reviewed and verified so that the data that indicates product quality is within the bounds of Fujitsu's quality standards, which specify, for example, the acceptable number of bugs per unit line of code. If this data is out of standard, appropriate actions are taken before proceeding to the next stage.[1)]

Tests are done by three parties. The first party is composed of programmers and performs unit tests and component tests. The second is the dedicated test team within the development division responsible for the system test. The third party is the QA department, which is responsible for the product test (PT).

What characterizes Fujitsu's QA activities is the presence of the QA department, which is independent of the development divisions. Similar types of organizations are not common outside of Japan, and even in Japan only large vendors such as Fujitsu have such an organization. PT by the QA department is done on products that, for most vendors, are ready for shipment. During a PT, the QA department acts as an independent in-house customer and inspects various aspects of the product such as its functionality and the degree to which it solves the customer's business issues. This objective evaluation of a product in an extra round of tests serves as the final check of the overall product and contributes to high product quality.

Fujitsu's QA is part of the total quality system shown in **Figure 2**. It is Fujitsu's view that high quality can be achieved only when

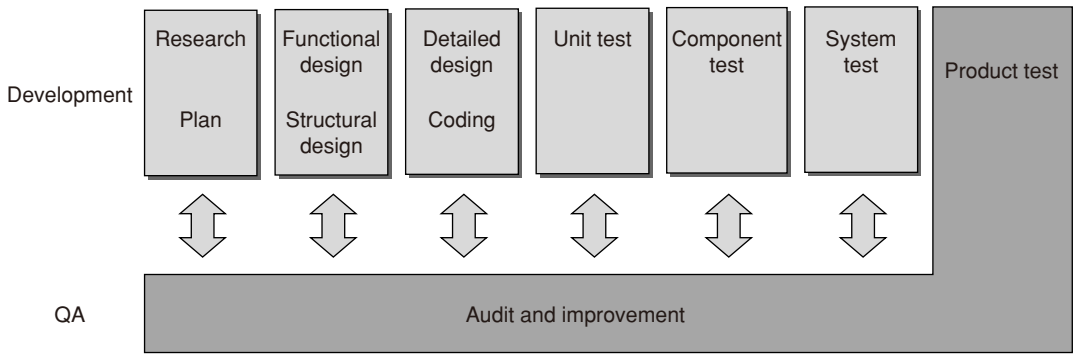


Figure 1
Typical waterfall style development process.

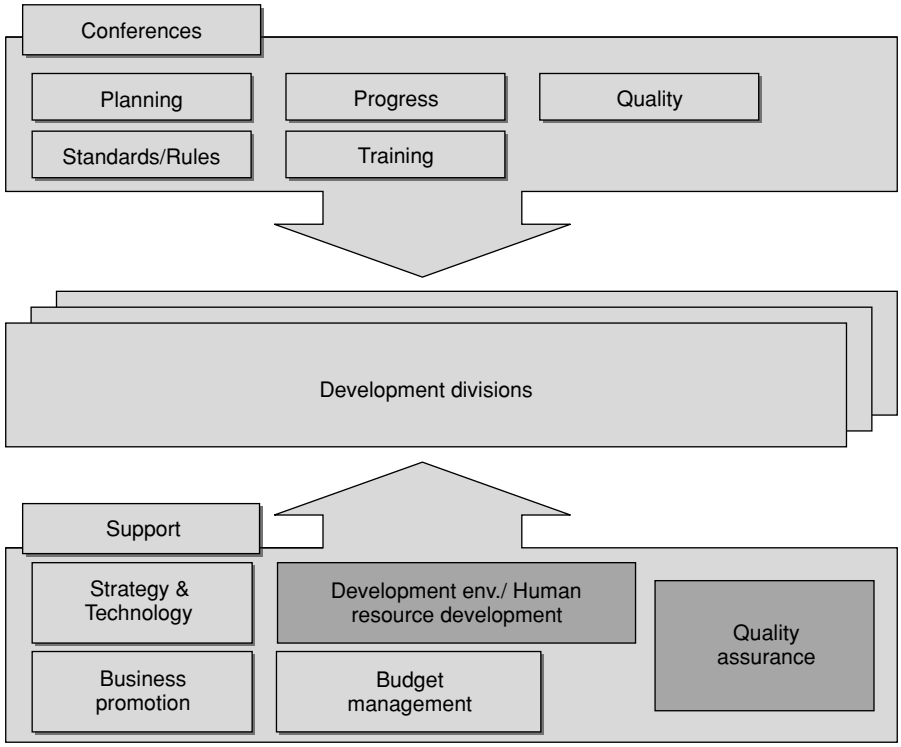


Figure 2
Total quality system.

all activities in the quality system are moving in the right direction. The importance of the development environment and its activities and also human resource development are described later.

2.2.2 New quality requirements for open platform

In the current development on the open platform, middleware needs to support hardware having various architectures. In order to provide a single feature to different platforms, several 10s of platforms, including variations in localization, will have to be supported (**Figure 3**). Testing is required on all platforms, which means that if a problem is found during testing, the tests will have to be redone for all platforms.

Because OSs in the open platform are rapidly evolving, the same OS function might behave differently in different versions, and the information provided about such differences is sometimes insufficient. There are cases where an unreported OS incompatibility caused by such a difference (between versions or even patches) has caused a customer's application to malfunction. There are also cases where an error in an incompatibility report by a vendor has caused Fujitsu middleware to malfunction.

It is important to isolate the customers from these uncertain factors, and many of Fujitsu's customers require that the middleware layer absorb these changes.

The increase in the development efforts to support multiple platforms and the presence of uncertain factors in the lower layer are two major issues that hinder high-quality development. Fujitsu's standard development process emphasizes that all issues at each stage must be resolved so each development stage can steadily improve the quality of products. However, uncertain factors caused, for example, by OS incompatibilities, could mean that

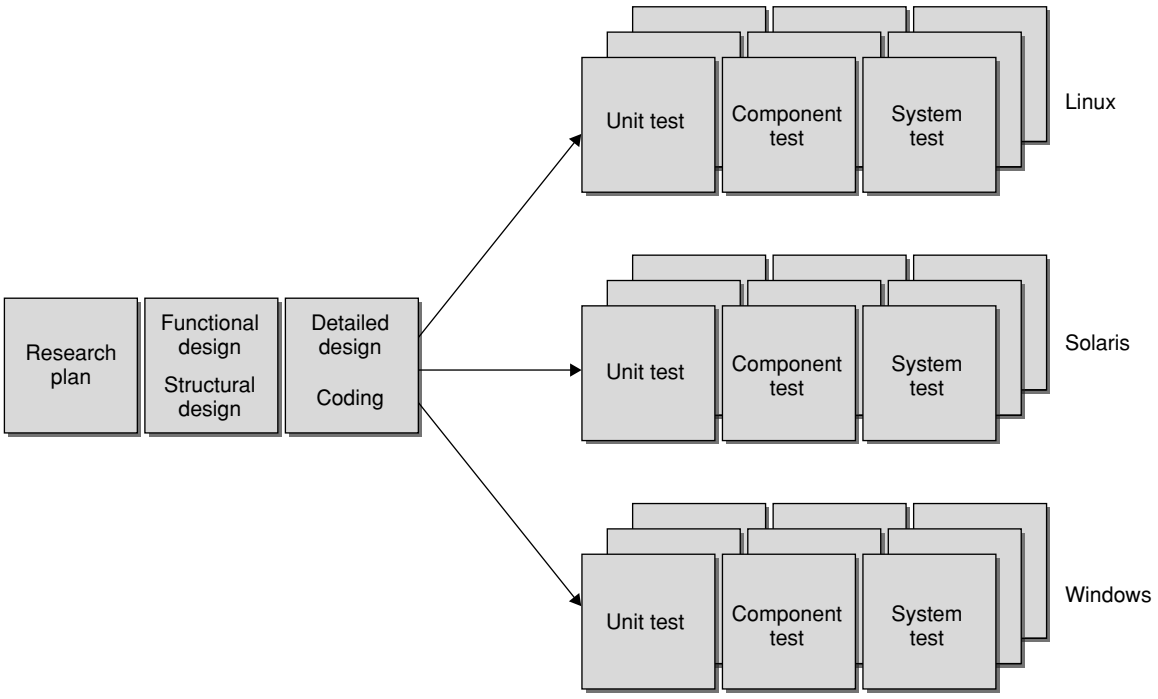


Figure 3
Over 40 platforms to be supported.

program designs that are thought to be complete may need reworking later on. We have found that simply looking at the output of each stage is not sufficient for grasping the risks of possible deviations from the standard development process and that it can lead to incorrect estimates about the final quality of a product.

In the open platform, how to achieve high-quality development by limiting the effects of uncertain factors is the foremost requirement.

3. Efforts to meet new quality requirements

In order to control the uncertain factors that have been introduced by the open platform, Fujitsu is focusing on the upstream development stages. Fujitsu's approach is to visualize the risks caused by the uncertain factors and proactively manage the risks that are identified. Through this frontloading, Fujitsu is aiming at a higher quality than in mainframe development. The next section describes our efforts in the risk management scheme based on the frontloading approach.

3.1 Controlling risks of uncertain factors at upstream stages

High-quality software is a result of a detailed plan and accurate execution of the project based on that plan. For high-quality development, it is important to isolate a project from unexpected problems as much as possible. However, the open platforms have accelerated the introduction of uncertain factors that can be caused by competing products, new technology, and market and customer requirement changes.[2)] These uncertain factors are possible risks to a project, and their effects are very difficult to estimate accurately at the beginning of a project. To proactively manage project risks, Fujitsu is employing a two-step risk management scheme to audit the development process as well as the output of each stage.

3.1.1 Planning-stage risk management

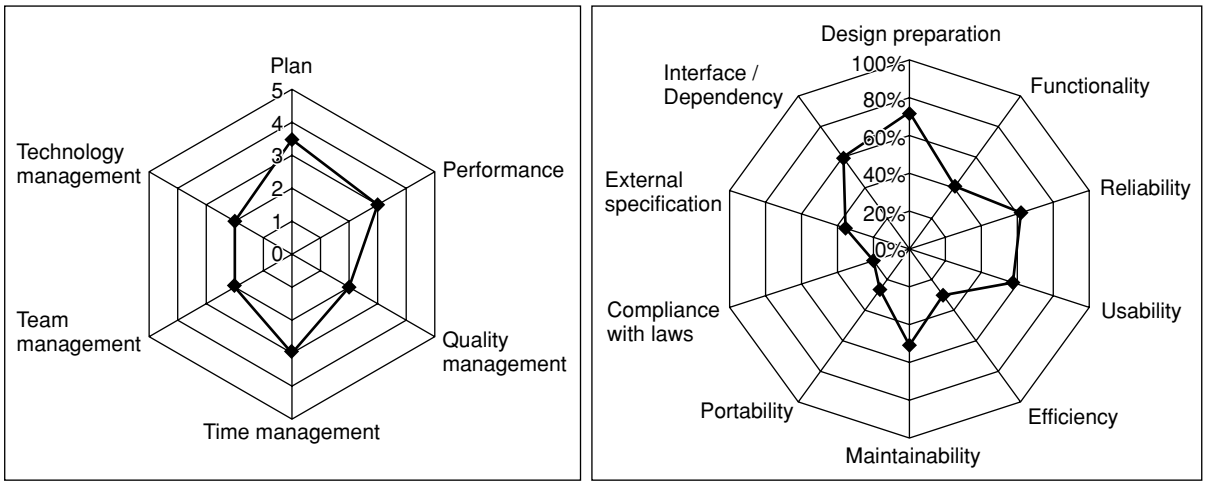
Planning-stage risk management is a method of managing risks by identifying and actively controlling them. It starts by clarifying the essential aspects of the development plan such as the project plan, project execution plan, quality objectives, and new technology management. After making sure that all essential areas are addressed, engineers look into each area in detail and study the risks that could affect the project later on. These risks are then evaluated and managed for early resolution. The steps for employing planning-stage risk control are as follows.

1) The project is checked against a checklist that covers six areas of project risk, for example, the product's characteristics, platforms to be supported, and new technology to be applied.
2) Factors that cannot be made definite are listed as risks, and their effects are evaluated. The results are shown in a radar chart [**Figure 4(a)**].
3) The action plans for reducing the risks, the persons responsible, and the date of action are recorded and followed up.
4) The results of actions and how they have reduced risks are fed back to the evaluation results.

3.1.2 Design-stage risk management

Design-stage risk management follows from the analysis of problems detected in middleware developments. A study of problems that have had major impacts on customers in the past showed that many of these problems stem from the lack of more detailed consideration at the design stage. Areas such as new technology and changing customer requirements are also causing problems in the design stage. A checklist is created to make sure that sufficient consideration is given to essential aspects at the design stage. This checklist is supplemented by know-how from the analysis of the past problems, publications, and individual check sheets used by various development teams. Similar steps as those used to design the planning-stage risk management are applied to control and reduce the risks as early as possible.

Two risk management methods are applied at different development stages and on different outputs; however, the following details are



(a) Planning-stage risk management

(b) Design-stage risk management

Figure 4
Example application of risk management.

common to both methods.

1) Evaluation is done on the detailed checklist based on know-how from previous development experiences. Examples are shown to clarify risks in the development.
2) An original risk categorization is used to visualize risks in the radar chart [**Figure 4(b)**].
3) Identified risks are traced by clarifying the actions to be taken, the schedules for the actions, and the persons who will perform them.

The risk management scheme has been applied to development projects since 2004 and has been continuously improved by feeding back its application results. Almost all of the projects that have applied the scheme have completed their development activities within the target date, proving the effectiveness of the method.

3.2 Isolation from risks of OS uncertainties

In order for the customers to operate their business systems continuously, it is important that changes in hardware and OSs are absorbed by the middleware layer and do not influence the customer applications. However, in some open systems, OS incompatibilities and incorrect information have caused malfunctions in middleware. As part of the frontloading efforts, Fujitsu has started activities on Linux systems, which are increasingly being used as mission-critical systems.

1) Proactive verification of incompatibilities
   Fujitsu has started original efforts to proactively verify there are no OS incompatibilities. The QA department owns a large number of test suites for various middleware that are automated and run against new update releases. If the results of the tests are different from those of the previous execution, Fujitsu engineers will study the cause and share the information throughout the organization.
2) Sharing trouble information
   Trouble information such as information about incompatibilities and bugs is collected by the QA department and shared throughout the organization. By quickly accumulating and sharing information from vendors, various communities, and our own development organization, early resolution of problems in the upper stream of the development is made possible.

There are many cases where middleware had to be changed to compensate for problems introduced by OSs. By frontloading the verification process, Fujitsu is proactively controlling the risks from a very early stage of development. This results in providing customers with stable operation, which enhances the value that Fujitsu's middleware products provide.

4. Development environment and human resource development

To adapt to the rapidly changing environment of open platform business, Fujitsu is improving every aspect of the quality system shown in Figure 2. The following sections describe the development environment and human resource development, the activities of which are very important in supporting the quality system's infrastructure.

4.1 Development environment

Fujitsu's resource management of source code and documentation has become complex because we now support multiple platforms and a large-scale development environment distributed over 10 locations worldwide. Off-the-shelf resource management tools cannot fully support this type of development environment.[3),4)] In addition to the efforts to attain high quality in the upstream development stages, we are also conducting activities to accurately carry out development in our complex development environments.

For example, we have created an efficient development environment and also reduced our resource management costs through a globally

distributed development environment called the Production of Advanced Software Quality (PROASQ) system (**Figure 5**).

1) Integrated and centralized management

PROASQ resides in a Fujitsu security zone called the Virtual Private Development Center (VPDC) and allows direct access from affiliated companies around the globe. The software configuration management (SCM) is based on the Enabler SCM engine, which allows the most recent resources (source code, documents) to be shared among development teams both inside and outside of Japan.

2) Process management

With PROASQ, we can realize total management of development processes. The process from the creation of a correction to its shipment is especially important for maintenance. PROASQ integrates a problem management system called Quick Quality Management (QQM) and a correction shipment/management mechanism called Updatesite MW, thereby unifying the maintenance process and realizing automated shipment of corrections.

3) Visualization of development process

PROASQ integrates standard workbenches as a workflow to make sure the standard development process is followed. This eliminates human error and deviations from standards. Through management using this workflow, visualization of the development process is also achieved and is actively used in quality assurance activities.

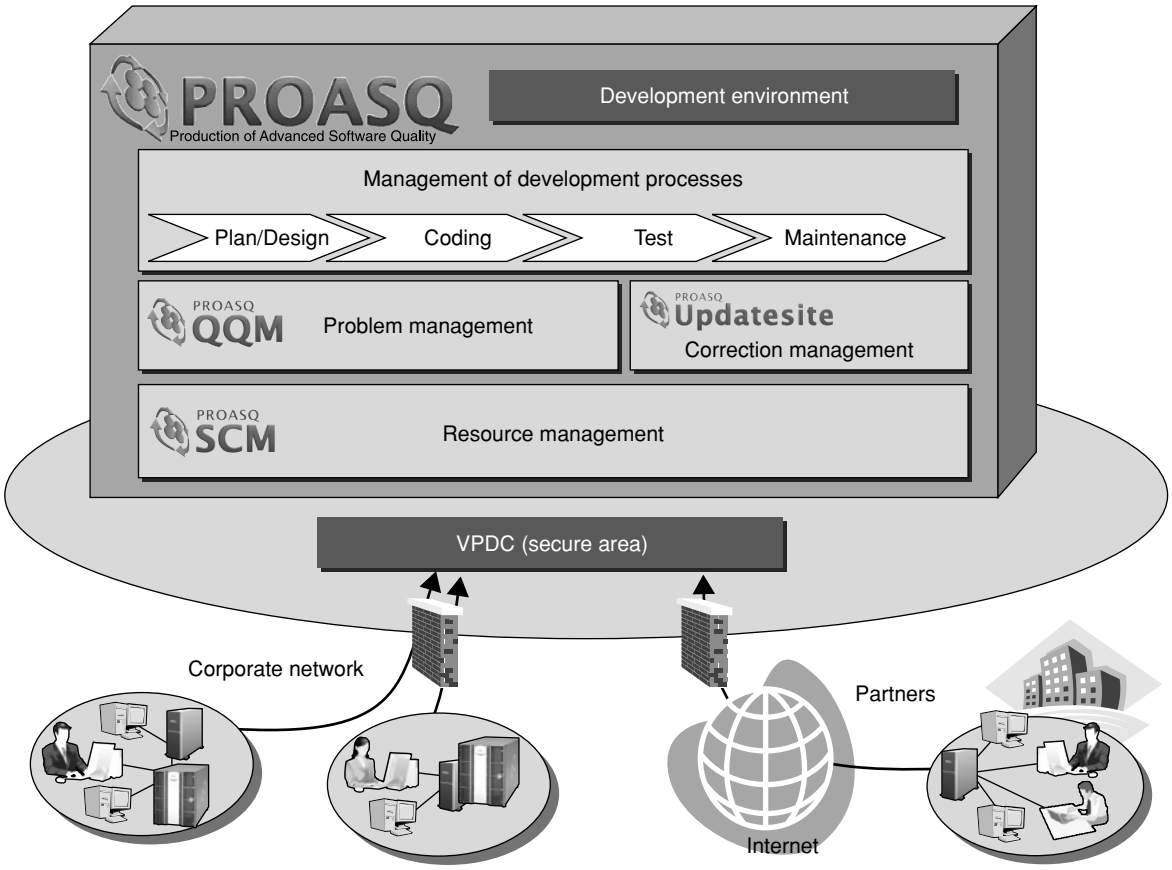


Figure 5
Globally distributed development environment “PROASQ”.

4.2 Human resource development

In the past, there were in-house experts who had either created or had a deep involvement in the creation of new technology, and their expert skills have been passed on to new engineers in an organized manner. However, in the open platform, new technology is introduced almost everyday and centralized management of technology has become almost impossible. Therefore, Fujitsu has created professional certification programs to identify the engineers who have accumulated expert skills. Certified experts will work as leaders in their jobs and participate in cross-functional communities where experts work together to solve problems and share their knowledge throughout the entire organization. Professionals in product planning, development, and product consultation are actively participating in these cross-functional activities.

5. Future actions

Through the activities mentioned above, Fujitsu has achieved a higher quality in open platform development than it has acquired in mainframe development. The next step is to make these activities common knowledge among all members of the organization and finally make them part of the organization's culture.

1) Guidelines and education

Information about individual activities such as risk management is given in various guidelines. It is important to make sure every member knows about these guidelines. By conducting Web training courses and periodic tests to verify awareness, we are establishing an environment where every development engineer is armed with the necessary information.

2) Integration of tools into development infrastructure

Risk management is currently carried out using a checklist. However, it is more effective to integrate the necessary tools into the development infrastructure so development engineers work in an environment that automatically applies the necessary guides and checks. We will continue to improve our development infrastructure so our system automatically ensures adherence to the necessary guidelines.

6. Conclusion

Fujitsu's quality system has continuously improved since the mainframe era. By applying the frontloading approach and reducing risks introduced by uncertain factors and unstable platforms, it has become possible to attain higher quality in the upstream development stages. Through these efforts, Fujitsu has achieved a level of quality that cannot be achieved by relying on testing alone.

To win in today's rapidly changing markets and quickly respond to the customers' needs, all members of a development team must be specialists in their field and have the determination to continuously improve in their work. Improving the development infrastructure to realize speedy development and creating professional communities to facilitate self-promoted improvement are part of Fujitsu's quality system, and we will continue to improve this system so we can provide our customers with the highest-quality products available in the world.

References

1) Y. Ishii: Software testing and quality assurance. (in Japanese), JUCE, 1986, p.103-132.
2) K. Yasuda: Approach and application of software quality assurance. (in Japanese), JUCE, 1995, p.339-351.
3) IPA, SEC: Visualization of IT projects. (in Japanese), Nikkei BP, 2006, p.12-58.
4) IPA, SEC: Software development data white paper. (in Japanese), Nikkei BP, 2006, p.32-115.

Toru Shiraiwa, *Fujitsu Ltd.*
Mr. Shiraiwa received the B.Sc. degree in Physics from McGill University, Montreal, Canada in 1982. He joined Fujitsu Ltd., Kawasaki, Japan in 1982, where he has been engaged in quality assurance of software products. Currently, he is responsible for the quality assurance of operation management products.

Kimio Saito, *Fujitsu Ltd.*
Mr. Saito received the B.S. degree in Information Engineering from Kougakuin University Tokyo, Japan in 1984. He joined Fujitsu Ltd., Kawasaki, Japan in 1984, where he has been engaged in quality assurance of communication software for mainframes. He has also been engaged in the creation of software development environments since 2004.

Keita Uchiyama, *Fujitsu Ltd.*
Mr. Uchiyama received the B.S. degree in Management Engineering from Tokyo Metropolitan University, Tokyo, Japan in 1991. He joined Fujitsu Ltd., Kawasaki, Japan in 1991, where he has been engaged in organization invigoration and human resources development.