R&D Software Development Standards and their Operation

Kumi Jinzenji, Naoki Kasahara, and Takahiro Muraki

Abstract

The NTT Service Innovation Laboratories has created original software development standards for various quality levels from product incubation through to finished commercial products. These standards have been in operation for about four years and have been applied to approximately 200 software products, most of which were provided to business companies of the NTT Group. The developers and researchers in our laboratories have secured the quality and identified the risk of the products before releasing them under the responsibility of the authorized organization. This article describes our latest R&D (research and development) software development standards, including their features, operation, effects, and future developments.

Keywords: software development process, quality, risk

1. Introduction

NTT laboratories develop a variety of software products such as business applications, middleware, and information processing engines. The range of quality levels is diverse, from commercial-level high quality to low-level quality that is sufficient for product incubation. The skill levels of developers are also diverse. Moreover, these products are used in various departments of the business companies, which have different requirements and demands for software quality depending on their business objectives. These differences in our products and how they are used have sometimes resulted in problems after their release to business companies. Defects have been found when products were used in ways not anticipated by the developers, and delays have occurred in dealing with such defects. To solve these issues, it was necessary to establish rules for the organizations developing the software so that they could secure the software quality and identify the associated risk of the products and could take responsibility for implementing the rules. Accordingly, in July 2009, the NTT Software Innovation Center started addressing these issues by creating standards for software development and clarifying the rules for operation of these standards. Generally, software development standards define processes and activities necessary to continually improve product quality or productivity. Our software development standards have additional features over typical standards, with added processes that enable the software product developing organization to take responsibility for product quality and risk. The standards have undergone a few revisions in recent years, but NTT Service Innovation Laboratories settled on the new *R&D* (*Research and Development*) *Software Development Standards* in April 2013.

2. Features of the R&D software development standards

The document describing the R&D software development standards has four parts: the main text of the standards, the forms and samples section, operational guidelines, and additional documents that help readers understand the development standards. All of the processes and activities described in the main text do not necessarily always have to be performed. The processes and activities that are particularly important are marked and selected as a *mandatory* process checklist. Achieving these mandatory processes can

Table 1. Definitions of software quality classes.

Quality class	Definition with respect to suitability for introduction in a business	Example of use	Main expected users
A	Companies can introduce the software product to their businesses largely as-is, and it will be used under a very strict SLA, as with infrastructure or network services for which the service must not stop.	Infrastructure or network service in a company.	Individual consumer and cooperate customer
B	Companies can introduce the product to their businesses as-is, but it will be used in conditions where the application can be restarted to some extent, as with a software package or solution.	Package or solution in a company, disclosure of technology to a related company.	Individual consumer and cooperate customer
C	The product is used in a somewhat limited manner, as with a trial, and is introduced to companies with conditions on its functionality. Some of the architecture will require further improvement or additional testing.	Service trial conducted by company or laboratory.	Individual consumer, cooperate customer and company employees
D	Usage is extremely limited, as with a demonstration of functionality; for installation, there are conditions on functionality and architecture, and testing may require drastic revision.	Demonstration at a company	Company employees
E	No quality evaluation has been done, so it cannot be introduced in a business.	Research use	Researchers

be considered to be conforming to the development standards. The main text of the standards is used as a reference for these mandatory processes. These standards were written based on the assumption that a waterfall development[*1] process is used, but by redefining several mandatory processes, these standards can be applied to agile development[*2] processes as well.

Further, these development standards have four original features not generally seen in existing standards. The first three features were introduced in the predecessor to this research, *Essentials of Software Development for Incubation* [1], and this revision strengthens and improves the original features. The four features are described in detail in the subsections below.

2.1 Introduction of software quality classes

The quality classes defined in the R&D software development standards are listed in **Table 1**. There are five quality classes, A to E. These five quality classes can be broadly divided into three categories. First of all, Classes A and B consist of software that companies can use as-is (that is, the software can be directly introduced into a company's package or service). Further, Class A software has a strict service level agreement (SLA) to ensure the system the software is used in never stops, while Class B software permits restarts of the certain degree.

Second, in Classes C and D the software cannot be used as-is and/or needs further improvement. Class C software requires some quality issues to be improved before it can be released for business use, and Class D software needs to be totally improved or refactored.

Finally, Class E includes products of unknown quality, so this software cannot be introduced in a business in operation.

These quality classes were decided by developers, who considered how the software would be used after being released to business companies.

2.2 Quality requirement to build in and verify with a quality checklist

The R&D software development standards establish 105 items to be checked in a quality class checklist based on quality characteristics and sub-characteristics of ISO/IEC (International Organization for Standardization/International Electrotechnical Commission) 9126, the international standard for the evaluation of software quality. This makes it possible to gain a concrete understanding of work required to build in and verify quality (**Table 2**). The quality checklist provides recommended criteria for each quality class, which is based on building in and verifying quality with business applications. The recommended criteria are general information for the developers but are not mandatory because the product

*1 Waterfall development: A development method in which all functions sequentially pass through several processes to completion. In principle, the previous process is completed before proceeding to the next.

*2 Agile development: A development method in which the item being developed is divided into many small functions, which are iteratively implemented one after another, minimizing the risk due to changes in requirements.

Table 2. Quality characteristics and recommended criteria.

No.	Quality characteristics		Recommended criteria				
			Quality class				
	Quality characteristics	Sub-characteristics	A	B	C	D	E
1	Functionality	Suitability	☆	☆	☆	☆	-
2		Accuracy	☆	☆	☆	☆	-
3		Interoperability	☆	☆	☆	-	-
4		Security	☆	☆	☆	-	-
5		Functionality compliance	☆	☆	☆	-	-
6	Reliability	Maturity (validity)	☆	☆	☆	-	-
7		Maturity (fault convergence)	☆	☆	☆	-	-
8		Fault tolerance	☆	☆	☆	-	-
9		Recoverability	☆	☆	☆	-	-
10		Reliability compliance	☆	-	-	-	-
11	Usability	Understandability	☆	☆	-	-	-
12		Learnability	☆	☆	☆	-	-
13		Operability	☆	☆	☆	-	-
14		Attractiveness	-	-	-	-	-
15		Usability compliance	☆	-	-	-	-
16	Efficiency	Time behaviour	☆	☆	☆	☆	-
17		Resource utilization	☆	☆	☆	☆	-
18		Efficiency compliance	☆	-	-	-	-
19	Maintainability	Analyzability	☆	☆	☆	-	-
20		Changeability	☆	☆	-	-	-
21		Stability	☆	☆	-	-	-
22		Testability	☆	-	-	-	-
23		Maintainability compliance	-	-	-	-	-
24	Portability	Adaptability	☆	☆	☆	-	-
25		Installability	☆	☆	-	-	-
26		Co-existence	☆	☆	-	-	-
27		Replaceability	☆	☆	-	-	-
28		Portability compliance	-	-	-	-	-

Examples of conformance points for recoverability

(1) After a fault occurs, the software is able to return to the initial statebefore processing started, and perform the processing over again.

(2) Data can be recovered accurately using checkpoints or another mechanism after a fault occurs, and processing can resume within the required recovery time.

(3) The affected processes can be isolated when a fault occurs, and other processes can continue to operate.

(4) Traces, logging, dumps, or other records for analyzing the fault can be used when a fault occurs.

quality requirement depends on the characteristics of each software product and on the developers' customers. It is important that all developers share the checked result and recognize which quality characteristics are included in the development requirements.

2.3 Documentation according to quality class

The skill levels of developers at NTT laboratories are quite diverse, so the development standards provide sample documents with levels of descriptions corresponding to high-quality products of Class B and above, including a basic design document, a project planning document, and a release readiness document (the three principal documents). Some examples of items included in a project planning document are listed in **Table 3**. General-development-standard documents typically only describe items in simple, general, and broad terms, so it is difficult to create concrete and usable documents for a high-quality-class product without adequate development experience, knowledge, and skills. Thus, for the R&D software development standards, we make it possible to achieve the desired quality class regardless of the skill set available by including many possible activities, metrics, and evaluation methods. When a product with a lower quality class is developed, only items in the samples up to the target quality level need to be followed, and the rest of the items can be deleted. When we evaluated the description levels in documents within NTT laboratories, we found that the level of description seemed to correspond to the quality level better when using the Class B samples than when using the Class C samples.

2.4 Mandatory processes based on quality class

These development standards define mandatory processes according to the target quality class (**Table 4**). Mandatory processes can be broadly divided into three categories: two project reviews, creation of the three principal documents, and development management. We have made it possible to select three levels of process sets for each of these categories according to the target quality level.

Table 3. Example descriptions of verification methods and metrics in a project planning document.

Process	Functional design	Detailed design	Coding	Unit testing	Integration testing	System testing	Field testing
Verification method	Software review			Software testing			
	Review	Review	Code review	- White box test - Black box test	- Black box test - Regression test - Recovery test	- Black box test - Regression test - Time and resource efficiency test - Recovery test - Load test - Stability test - Multi-hardware test - Manual test	- Black box test - Regression test - Operational test - Non-functional tests
Metrics	- Review frequency and time - Number of errors - Number of comments	- Review frequency and time - Number of errors - Number of comments	- Review frequency and time - Number of errors - Number of comments	- Coverage - Number of test cases - Test density - Number of bugs - Bug density	- Number of test cases - Test density - Number of bugs - Bug density	- Number of test cases - Test density - Number of bugs - Bug density - Fault convergence	- Number of test cases - Test density - Number of bugs - Bug density - Fault convergence

Table 4. Mandatory processes.

Mandatory processes		Class A	Class B	Classes C and D
1	Project review	Mandatory		
2	Create three principal documents	Mandatory (level of description according to quality class)		
3	Development management	Mandatory	Partially mandatory	Not mandatory

Mandatory processes concerned with two project reviews are common to all quality classes in order to allow the software development organization to take responsibility for their project risk as well as their product risk and product quality. In addition, the first project review is done by an organization manager to confirm project baselines and decide whether or not to continue the project. The second project review is also done by an organization manager to verify the product quality and decide whether the product is ready for release. Creation of the three principal documents is also common to all quality classes, but the description details depend on each project. Furthermore, parts of the development management process can be omitted, depending on the quality class. We also define mandatory processes for existing products, which refers to software products developed before our R&D software development standards were in operation. Specifically, for *existing* products, most of the mandatory processes are activities centered on the second project review (release decision).

3. Operation of the R&D Software Development Standards

In order to avoid ending up with standards that are only a mere façade and the possibility that they will become obsolete, the R&D software development standards are operated using a Plan-Do-Check-Act (PDCA) cycle (**Fig. 1**). The core activities of this PDCA cycle are described below.

3.1 Operational rule provisions

The following two operational rules were authorized in NTT Service Innovation Laboratories when establishing the R&D software development standards.

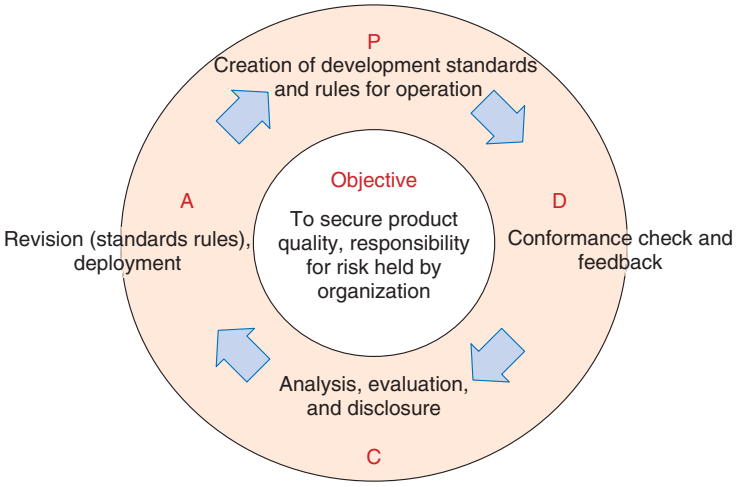

Fig. 1. PDCA cycle.

(1) The R&D software development standards shall be applied to products intended for use in the business companies and products intended for use in service trials for NTT customers, and they shall achieve one of the four quality classes from A to D.
(2) The top managers of the software developing organization shall conduct two project reviews on cases where the R&D software development standards are being applied.

Rule (1) clarifies what is subject to the development standards and the target quality class, and Rule (2) ensures that the organization is taking responsibility for the quality and risk associated with products.

In the first project review, the managers holding responsibility in the organization check the project scope, QCD (quality/cost/delivery) baseline, as well as the risk. In the second project review, they make a decision on release after checking the product quality, provision conditions, and risk.

3.2 Standards compliance check and feedback

The standards compliance check involves comparing checklists submitted by developers with evidence to see whether the mandatory processes are being achieved. Currently, this procedure is being done by the development standards operation group. The standards compliance check is done immediately after the two project reviews. The results of the check are brought back to developers by the development standards operation group in a meeting, which is used as a setting for communication between the software developer and the operation side that is also creating the development standards. These activities ensure that all projects can be monitored in the laboratories and thus avoid having standards that are mere façades.

3.3 Analysis and evaluation of the state of operation

Approximately once every six months, overall trends in the state of operation are analyzed, evaluated, and disclosed. These results are brought back so they can be used in subsequent actions of each software development project.

3.4 Revision and deployment

The development standards need to be updated when issues arise. To maintain the quality of standards documents during the updating process, reviewers selected within each organization apply the development standards. Trials are sometimes conducted to examine the effects of new policies on R&D activity. Reviews and trials have also been conducted when updating the R&D software development standards in order to check whether there are any issues with their operation or effects. Then a formal meeting is held to authorize the development standards, and they are deployed through presentations and by publishing the standards documents on a dedicated website.

4. Application results, effects, and future issues

The R&D software development standards have been in operation since April 2013, and as of the end of July 2013, they have been applied to approximately 30 cases. When the preceding development standards are included, the number of cases they have been applied to exceeds 200. We now have an understanding of the quality of software products that have been introduced into the business companies, so the objectives of this initiative have been achieved. A task for the future is to quantitatively evaluate the efficiency of operation and the validity of our strategies towards securing software quality and recognizing risk.

It is important how both the creation of software development standards and their operation are considered. We will expand the implementation of our R&D software development standards and our operational know-how to organizations within the NTT Group.

Reference

[1] K. Jinzenji and T. Hoshino, "Essentials of Software Development for Innovation," NTT Technical Journal, Vol. 23, No. 8, pp. 52–55, 2011 (in Japanese).

Kumi Jinzenji
Senior Research Engineer, Software Engineering Project, NTT Software Innovation Center.
She received the B.E. degree in electrical and electronic engineering from Sophia University, Tokyo, the M.E. degree in electronics and telecommunication engineering, and the Ph.D. degree in global informatics and telecommunication from Waseda University, Tokyo, in 1989, 1991, and 2005, respectively. She joined NTT in 1991 and was a visiting researcher at Global Informatics and Telecommunication Institute of Waseda University during 2003–2004. She was a project manager of software development for national network systems at NTT Communications from 2005 to 2007. Her current research interests are project management, software reliability, and software development processes and quality. She received the Young Researcher's Encouragement Award in 1998 from the Institute of Electronics, Information and Communication Engineers (IEICE). She is a member of IEICE and the Information Processing Society of Japan. She has been certified as a Project Management Professional (PMP) by the Project Management Institute (PMI), USA.

Naoki Kasahara
Research Engineer, Software Engineering Project, NTT Software Innovation Center.
He received the B.S. degree in mathematics from Keio University, Kanagawa, in 1991. He joined NTT in 1991. He worked at NTT Communications from 2001 to 2003 and at NTT Resonant Inc. from 2004 to 2006. He has a PMP certification from PMI, USA. He also has a Project Manager qualification from the Information-technology Promotion Agency, Japan. His interests are methodologies of project management and project risk/quality management. He is a member of PMI and the Project Management Association of Japan.

Takahiro Muraki
Research Engineer, Software Engineering Project, NTT Software Innovation Center.
He received the B.E. and M.E. degrees in electrical and electronic engineering from Gifu University, in 1989 and 1991, respectively. He joined NTT in 1991. His interests include project management and analysis of project risk and quality. He is a member of Japan Academy of Facial Studies.