



SOFTWARE DEVELOPMENT STANDARD

Mar. 22 , 2019

Japan Aerospace Exploration Agency

The official version of this standard is written in Japanese. This English version is issued for convenience of English speakers. If there is any difference between Japanese version and English one, the former has precedence.

This is an English translation of JERG-0-049B.

If there is anything ambiguous in this document, the original document (the Japanese version) shall be used to clarification.

Disclaimer

The information contained herein is for general informational purposes only. JAXA makes no warranty, express or implied, including as to the accuracy, usefulness or timeliness of any information herein. JAXA will not be liable for any losses relating to the use of the information.

Published by
Japan Aerospace Exploration Agency
Safety and Mission Assurance Department
2-1-1 Sengen Tsukuba-shi, Ibaraki 305-8505, Japan

JERG-0-049B

SOFTWARE DEVELOPMENT STANDARD

1. JAXA JERG-0-049B is hereby established.
2. For any questions concerning this standard, contact the Safety and Mission Assurance Department of the Japan Aerospace Exploration Agency (JAXA).

Establishment: Safety and Mission Assurance Department Director
March 22, 2019

Table of Contents

1 Scope	1
2 References	1
2.1 Informative references	1
3 Terms, definitions and abbreviated terms	1
4 Organization of this standard	6
5 Primary life cycle processes	8
5.1 Not used.....	8
5.2 Not used.....	8
5.3 Development process	8
5.3.1 Process implementation	9
5.3.2 Items to be applied to all processes.....	10
5.3.3 Computer system requirements analysis.....	10
5.3.4 Computer system architectural design.....	10
5.3.5 Software requirements analysis	11
5.3.6 Software design	13
5.3.7 Not used.....	15
5.3.8 Software coding and testing	15
5.3.9 Not used.....	16
5.3.10 Software integration.....	16
5.3.11 Software integration test	17
5.3.12 Software installation into target platforms (embedding)	18
5.3.13 Computer system integration and computer system integration test	19
5.3.14 Supply and introduction of software product	21
5.3.15 Software acceptance.....	22
5.4 Operation process	22
5.4.1 Process implementation	22
5.4.2 Operational testing	23
5.4.3 Operation of computer system including software	23
5.4.4 Operation results management	24
5.4.5 Customer and user support	24
5.5 Maintenance process	24
5.5.1 Process implementation	25
5.5.2 Problem identification and modification analysis	25
5.5.3 Modification implementation.....	26
5.5.4 Software reprogramming.....	26
5.5.5 Perform logistics support.....	26
5.5.6 Manage results of maintenance and logistics.....	26
5.5.7 Transition.....	26

5.5.8	Software disposal	28
6	Supporting life cycle process	30
6.1	Documentation process.....	30
6.1.1	Process implementation	30
6.1.2	Development	30
6.1.3	Production.....	30
6.1.4	Maintenance/Revision/Disposal.....	31
6.2	Configuration management process.....	31
6.2.1	Process implementation	31
6.2.2	Configuration identification.....	32
6.2.3	Configuration change control	32
6.2.4	Record of configuration change status	32
6.2.5	Evaluation of configuration change status	32
6.2.6	Release management and delivery.....	32
6.3	Quality assurance process.....	32
6.3.1	Process implementation	33
6.3.2	Product and service quality assurance	34
6.3.3	Process assurance	34
6.3.4	Assurance of quality system.....	34
6.3.5	Manage quality assurance records	35
6.4	Verification process	35
6.4.1	Process implementation	35
6.4.2	Verification	36
6.5	Validation process	37
6.5.1	Process implementation	38
6.5.2	Validation	38
6.5.3	Manage the validation results	39
6.6	Joint review process	39
6.6.1	Process implementation	39
6.6.2	Project management reviews	39
6.6.3	Technical reviews	39
6.7	Assessment process.....	40
6.7.1	Process implementation	40
6.7.2	Assessment implementation.....	40
6.8	Problem resolution process.....	41
6.8.1	Process implementation	41
6.8.2	Problem resolution	41
6.8.3	Prevention	42
6.8.4	Problem trend analysis	42

Appendix43

1 Scope

This standard applies to the activities relating to the development, operation, and maintenance of software for satellites, probe, launch vehicle and ground systems, and the activities needed for system development and related support. When this standard is applied, embodying and tailoring may be performed in accordance with the characteristics of specific projects and other factors.

In principle, this document does not define the categories of personnel who implement processes. These personnel shall be defined by individual contracts or other agreements to which this standard is applied.

2 References

2.1 Informative references

- (1) ISO/IEC 12207:2017 Systems and software engineering -- Software life cycle processes
- (2) JIS X0160:1996 Software life cycle processes
- (3) ISO/IEC 14764:2006 Software Life Cycle Processes - Maintenance
- (4) JIS X0161:2008 Software Engineering - Software life cycle processes - Maintenance
- (5) ISO/IEC 15504-2:2003 Information technology - Process assessment - Part 2: Performing an assessment
- (6) Software Life Cycle Processes- Japan Common Frame 2007 (Japanese) (Copyright IPA/SEC 2009)

3 Terms, definitions and abbreviated terms

Term	Definition
Acceptance Inspection and Acceptance Testing	Acceptance inspection and acceptance testing are actions to evaluate compatibility with requirements at the time of acquiring software products. Inspection is an action to confirm the compatibility of a product, in accordance with evaluation criteria based on either requirements specifications or a predetermined value, by visually checking quantities and test results. Test comprises analysis, evaluation and checking of the functionality and capabilities of the software in order to obtain evaluation results and data required for use in inspection.
Activity	Activity is a component of a process and a set of strong correlative tasks.
Adaptive maintenance	The modification of a software product, performed after delivery, to keep a software product usable in a changed or changing environment NOTE—Adaptive maintenance provides enhancements necessary to accommodate changes in the environment in which a software product must operate. These changes are those that must be made to keep pace with the changing environment. For example, the operating system might be upgraded and some changes may be made to accommodate the new operating system. (ISO/IEC 14764 : 2006)
Architecture	Fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution (ISO/IEC/IEEE 12207:2017)
Assessment	Assessment is a process to evaluate the strengths and the weaknesses of a subject process and to identify opportunities to improve the process for some predetermined purpose.

Term	Definition
Computer system	<p>Computer system is the entire system consisting of sets of software, platforms, and hardware, including the platform and hardware that are able to execute the targeted software for development. Although the definition of what a computer system contains is arbitrary, the definition shall be unique for software products that are subjected for development.</p> <p>As a computer system may be defined in various ways, from a one-chip microcomputer to multiple general-purpose computers connected to a network.</p>
Configuration management	<p>Configuration management is an action to define the configuration items, i.e. computer systems or projects, to record changes of content and to manage such aspects as their storage, handling, and distribution.</p> <p>If the software consists of multiple modules, then not only must the software version be managed, but the version of each software module must also be managed. In addition, configuration management items require software consistent modules, requirements specifications, operation manuals, and so on.</p>
Corrective maintenance	<p>The reactive modification of a software product performed after delivery to correct discovered problems</p> <p>NOTE—The modification repairs the software product to satisfy requirements</p> <p>(ISO/IEC 14764 : 2006)</p>
COTS	<p>COTS is the abbreviation of Commercial Off-The-Shelf. It has already been developed and is available in the commercial marketplace.</p>
Enabling system	<p>System that supports a system - of - interest during its life cycle stages but does not necessarily contribute directly to its function during operation</p> <p>(ISO/IEC/IEEE 12207:2017)</p> <p>Note 1 For example, when a system-of-interest enters the production stage, a production-enabling system is required. When the target system is a spacecraft installation software, if speaking throughout the life cycle, the enabling system means such as an integrated development environment, emulator, or simulator.</p> <p>Note 2 Each enabling system has a life cycle of its own. This Standard is applicable to each enabling system when it is treated as a system-of-interest.</p>
Identifier	<p>Identifier is a short line of numbers, letters, and symbols, which is appended to each item of output and input to enable each item to be identified and classified by item type. Identifiers shall consist of not only a set of numbers, but also a set of letters and symbols. In addition, it is not always necessary identifiers be serial numbers.</p> <p>Appending a unique identifier to each item is convenient for requirements management and for traceability.</p>
Incident	<p>Anomalous or unexpected event, set of events, condition, or situation at any time during the life cycle of a project, product, service, or system</p> <p>(ISO/IEC/IEEE 12207:2017)</p>
Independent Verification and Validation: IV&V	<p>IV&V are the verification and the validation that are performed by the organization independent of the software development organization. With regard to independence, financial, technical and management viewpoints shall be considered.</p>
Input	<p>Input is information needed to implement an activity.</p>
Integrity	<p>Integrity is defined as the following properties in this standard:</p> <p>(1) Software component is complete with no deficiencies.</p> <p>(2) Software component is at an appropriate version.</p>
Interoperability	<p>A behavior including a successive messages exchanged among multiple objects to accomplish a specified objective</p>

Term	Definition
Knowledge asset	Generic name of assets including reusable software item, reusable code library, reference architecture, design element (such as architecture or design pattern), process, criteria, or domain knowledge related another technical information(such as training material) and lesson learned
Model based technology	A development method, an abstraction of system/software, using a model representing target system/software with one view point or one abstraction level
Non-functional requirements	Non-functional requirements are all requirements except functional requirements, such as performance, safety, and reliability.
Operation	Operation is an action to carry out missions for the achievement of a purpose by means of an appropriate computer system. An operation utilizes the computer system from beginning to end, and includes monitoring and maintenance functions, and so on.
Output	Output is information that is transformed from input by performing activities.
Perfective maintenance	The modification of a software product after delivery to detect and correct latent faults in the software product before they are manifested as failures NOTE—Perfective maintenance provides enhancements for users, improvement of program documentation, and recoding to improve software performance, maintainability, or other software attributes (ISO/IEC 14764 : 2006)
Preventive maintenance	The modification of a software product after delivery to detect and correct latent faults in the software product before they become operational faults (ISO/IEC 14764 : 2006)
Problem	Difficulty, uncertainty, or otherwise realized and undesirable event, set of events, condition, or situation that requires investigation and corrective action (ISO/IEC/IEEE 12207:2017)
Process	Process is a set of interrelated or interacting activities to transform input to output.
Project	Project is a time-limited endeavor to be implemented by means of specified resources and a temporary organization, with the purpose of fulfilling the project's mission.
Requirement	Requirement is one of a set of functions and performance targets requested for computer system or software and they may be also included such as not embodied and not detailed enough, or ambiguity in expression and vague expectations.

Term	Definition
Requirements specification	<p>Requirements specification is defined as the description of functions and performance required for computer system or software, embodied and specifically defined, and which also consider feasibility.</p> <p>In principle, the specified requirements specifications shall be verifiable as both feasible and mutually consistent.</p> <p>However, on the characteristic of adopted development process and required functions and performance, if the requirements specifications representative format is not a feasible verification format, the verification of feasibility of the requirements specification shall be complemented by the following methods:</p> <p>(1) It shall include the planning of agreement procedure with computer system users that the requirements specifications are satisfactory, and software verification plan shall include the planning of agreement procedure.</p> <p>(2) It shall include the test specifications enough to verify the requirements in the verification plan.</p> <p>In principle, any restrictions, laws, rules, and a project policy shall be included in the requirements specifications.</p>
Risk	<p>Risk is defined as the degree of danger attaching to a system's safety and surrounding projects. It includes assessment of undesirable outcomes which may occur as a result.</p>
Software	<p>Software is a set of computer system configuration items comprising commands and data, which are executed or processed by a CPU to fulfill functions and capabilities defined in the software requirements specifications. If it is a set of commands and data which are implemented or managed by a CPU, it is categorized as software, and the software development process standards apply to it. However, for the driver of firmware, OS, and middleware, appropriate development processes are applied, in accordance with the characteristics and therefore may be removed from the software development process standard. For example, hardware and its integrated driver development shall be considered as such a case.</p>
Software integration	<p>Integrate the software components step by step and construct the software that operates in the target platform</p>
Software life cycle	<p>Software life cycle is the period from the beginning of the requirements analysis phase until the termination of use of the software.</p>
Software products	<p>Software products are the set of software, source code, and related documentation.</p>
Software test specifications	<p>Software test specifications are defined as the descriptions of test conditions and expected results, expressed unambiguously, to prove that software meets the requirements specifications.</p> <p>If the requirements specifications are represented in a verifiable format, they may be treated as appropriate software test specifications.</p>
Software under test	<p>Software under test is software that is being subjected to testing and inspection.</p>
Software user's manual	<p>Software user's manual is the set of information a user needs in order to use software. It includes operation unit manuals, computer system operation manuals, and work operation manuals.</p>
Software verification plan	<p>Software verification plan is a documentation of the scope, content, method, environment (such as test equipment) and schedule pertaining to the verification of software development.</p> <p>A validation plan may be included.</p>
Strategy	<p>Policies to consider in planning an execution plan aimed at effective utilization of specific resources and time to achieve organizational business objectives or project missions.</p>

Term	Definition
Tailoring	Tailoring is defined as the activity to change processes defined in this standard in order to meet the project's particular characteristics and to establish an appropriate framework for each system development project.
Tasks	Tasks are components of activities corresponding to each stage of work.
Test plannability	Test plannability is defined as the aspect of test specification descriptions that indicates the possibility of testing and planning using the appropriate development phases and test environment for target test items.
The stability (maturity) of software requirements specifications	The stability (maturity) of software requirements specification is defined as the index which shows the possibility of specification change is small because of the software requirements specification be extracted and analyzed sufficiently. The definition of the index, and how it is evaluated, are arbitrary. Generally, provision for essential or refined changes to software requirements specifications affect process cost, delivery date and quality which let the software requirements specifications inputs. It is hoped that the index which is used to evaluate this shall be selected based upon the stability and maturity of software requirements specifications.
Traceability	Traceability is defined as the property which shows the correspondence with higher level documents.
Validation	Validation is a process. It uses objective evidence to confirm that the requirements which define an intended use or application have been met. Whenever all requirements have been met, a validated status is achieved. The process of validation can be carried out under realistic use conditions or within a simulated use environment. [ISO9000]
Verification	Verification is a process. It uses objective evidence to confirm that specified requirements have been met. Whenever specified requirements have been met, a verified status is achieved. [ISO9000]
Waiver	Refers to the case of accepting a system or component item as is despite a nonconformance to a requirement of the configuration identification document which occurred after the start of its fabrication or accepting it after repaired by an approved method. (JMR-006)

4 Organization of this standard

This standard categorizes the software life cycle into three primary life cycle processes and eight supporting life cycle processes, and defines these processes. The definition of these processes is shown in Figure 4.1 and Table 4.1.

The primary life cycle processes are processes in a software life cycle directly related to the development of target software, and consist of processes implemented during development, operation, or maintenance. The supporting life cycle processes are processes that indirectly affect the software life cycle process, with reference specifically to the development of object software, and act to support a primary life cycle process and are called by other processes, as necessary.

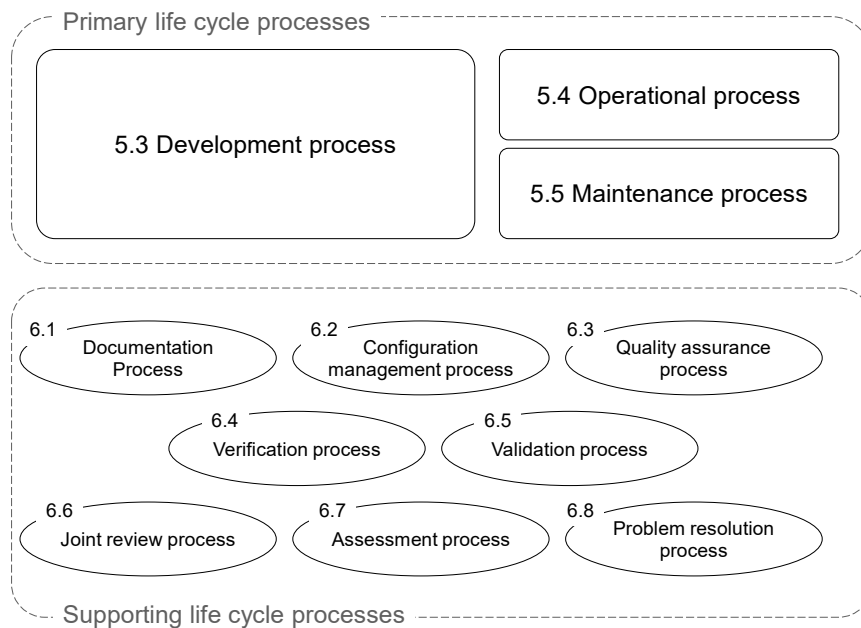


Figure 4.1 - Structure of the standard

The processes may be implemented in an order different from their clause number order in this document. Also, note that there will be cases where identical activities are described in multiple processes. For example, activities relating to software verification may be described as part of the development process, which is a primary life cycle process. These are also activities relating to the verification process, which is a supporting life cycle process.

The classification of processes is better understood as a classification according to the different viewpoints. This standard aims to define processes from various viewpoints, so that all of the required contents is covered completely (duplication is allowed). Therefore, this standard assumes that it will be applied after appropriate concretizing and tailoring have been performed regarding the relevant processes.

Table 4.1 Process list

Process		Description
Primary life cycle processes	5.1 Not used	
	5.2 Not used	
	5.3 Development	Process to be implemented from the viewpoint of development. Requirements analysis, design, coding and testing, installation on target platforms (embedding), supply, introduction, acceptance, and so on.
	5.4 Operation	Process to be implemented from the viewpoint of operation. Drafting plans and rules for operations, operational testing, operation, user support, and so on.
	5.5 Maintenance	Process to be implemented from the viewpoint of maintenance. Drafting plans and rules for maintenance, problem identification, modification, retirement, and so on.
Supporting life cycle processes	6.1 Documentation	Process regarding the record of the outcomes of individual processes.
	6.2 Configuration management	Process regarding the management of software and documents.
	6.3 Quality assurance	Process regarding the confirmation that the process meets the requirements of this standard and processes are managed according to the plan.
	6.4 Verification	Process regarding the confirmation that specified requirements have been fulfilled, based on the provision of objective evidence.
	6.5 Validation	Process regarding the confirmation that the requirements for a specific intended use or application have been fulfilled, based on the provision of objective evidence.
	6.6 Joint review	Process regarding a joint review means it is conducted by multiple personnel with different viewpoints.
	6.7 Assessment	Process regarding the confirmation of the executing status and identifying the items which need to be improved.
	6.8 Problem resolution	Process regarding resolving problems which occur during implementation.

5 Primary life cycle processes

This clause defines the following primary life cycle processes:

- (1) Not used
- (2) Not used
- (3) Development process
- (4) Operation process
- (5) Maintenance process

5.1 Not used

5.2 Not used

5.3 Development process

The development process is a collective process of defined activities, inputs, and outputs comprising the following processes:

- (1) computer system requirements analysis process;
- (2) computer system architectural design process;
- (3) software requirements analysis process;
- (4) software design process;
- (5) software coding and testing process;
- (6) software integration process;
- (7) software integration test process;
- (8) software installation into target platforms (embedding) process;
- (9) computer system integration and computer system integration test process;
- (10) supply and introduction of software product process;
- (11) software acceptance process.

These processes and activities may be implemented in a different order from what is described in this document. However, the overall configuration of the processes shall be defined as well as the management method of the entire process, so that appropriate process management is performed. In performing the processes, the use of the model based technologies shall be considered, as necessary.

5.3.1 Process implementation

When software development is started, activities that meet the following requirements shall be performed:

- (1) Define a development strategy (*1)
- (2) Based on the development strategy, the software development plan including the following information shall be established to cover:
 - (a) Purpose and constraints of the software development
 - (b) Scope of computer system
 - (c) Identification of target software
 - (d) Definition of the identification of software development processes and their relationship (operation process, maintenance process, and so on are considered)
 - (e) Definition of software development processes and activities(*2)
 - (f) Activities including roles, authorities, and responsibilities in each individual development process implementation management plan
 - (g) Review plan
 - (h) Preparation of development related documents' structure, and relationships of input and output in each development process
 - (i) Documentation plan, including development department and schedule
 - (j) Appropriate work allocations and methods including work plans (schedules, milestones) and achievement criteria, and progress management for each work
 - (k) Environment to be used for software development and verification (simulator, real hardware, test environment, and so on, enabling system or service) and their acquisition or access
 - (l) Management plan for COTS and knowledge assets used. The following shall be included:
 - (i) Identification of COTS items and knowledge assets
 - (ii) Definition of the quality assurance process regarding COTS items and knowledge assets
 - (m) Evaluation plan of compatibility with computer system
- (3) Software development plan shall be documented and approved

*1 In software coding and testing, the following points shall be considered as a development strategy

- (i) following development policies and rules relating to the software coding and testing
 - i suitable safety, security, privacy, and policy for environment activity
 - ii programming and coding standard
 - iii unit testing policy
- (ii) in case of software reuse, conformation method of the applicability to the computer system and the safety of the acquisition route
- (iii) performing of the software integration, peer review, and walkthrough review
- (iv) in case change management is conducted by not using any tool, how the configuration management is performed during software coding and testing

*2 Including activities regarding computer system with software installed.

5.3.1.1 Output

- (1) Software development plan

5.3.2 Items to be applied to all processes

Activities that meet the following requirements shall be performed throughout the entire development process:

- (1) The software development plan shall be updated and managed in accordance with the development status.
- (2) The progress of software development shall be monitored. It shall be reported to administrators as necessary.

5.3.2.1 Input

- (1) Software development plan

5.3.2.2 Output

- (1) Software development plan (updated)
- (2) Software development progress report

5.3.3 Computer system requirements analysis**5.3.3.1 Activity**

Activities that meet the following requirements shall be performed with respect to the computer system requirements analysis:

- (1) Requirements extraction

The functional boundary of the developed computer system shall be clarified. The requirements for computer system to be developed, the operational concept shall be analyzed, and operational scenarios shall be documented. The state transition (including operation mode) necessary for the developed computer system shall be identified.

- (2) Requirements specification development

Feasibility and consistency shall be confirmed, based on the operational scenarios, and the requirements specifications for computer system shall be defined.

The rationale for the requirements specifications for the computer system shall be clarified, and the traceability of requirements for the computer system shall be evaluated and maintained.

5.3.3.2 Input

- (1) Requirements for the computer system
- (2) Operational concept

5.3.3.3 Output

- (1) Operational scenarios
- (2) Requirements specifications for the computer system
- (3) Evaluation results of the traceability of the requirements specifications and requirements for the computer system

5.3.4 Computer system architectural design**5.3.4.1 Activity**

Activities that meet the following requirements shall be performed for the computer system architectural design:

- (1) Computer system architecture shall be designed based on the requirements specifications for the

computer system and the operational scenarios. Configuration items and their various categories (hardware, firmware, software, and operational) shall be clarified.

- (2) Requirements pertaining to the requirements specifications for the computer system shall be allocated among the individual configuration items of the system.
- (3) Feasibility of software items in fulfilling their allocated requirements shall be evaluated.
- (4) Rationale for the design and preconditions (e.g. operational assumptions) for the computer system architectural design specifications shall be identified, and an appropriate evaluation shall be performed.
- (5) Traceability of the computer system architectural design specifications relative to higher level requirements, such as the requirements specifications for the computer system, shall be evaluated.
- (6) Interface requirements for the software shall be extracted.
- (7) Set the evaluation criteria for a computer system architectural design based on the requirement specification and operational scenario for the computer system. The computer system architecture design shall be evaluated by that. Additionally, the computer system architectural design selection rational shall be recorded.

5.3.4.2 Input

- (1) Operational scenarios
- (2) Requirements specifications for the computer system

5.3.4.3 Output

- (1) Computer system architectural design specifications
- (2) Requirements for the software, including operational scenarios after the analysis
- (3) Interface requirements
- (4) Evaluation results of traceability relative to architectural design specifications with the computer system and requirements specifications for the computer system

5.3.5 Software requirements analysis

5.3.5.1 Activity

The following activities shall be performed for the software requirements analysis:

- (1) Software requirements specifications shall be developed, based upon the analysis of the computer system architectural design specifications, interface requirements, and requirements for software including non-functional requirements.
- (2) The required software state transition (including operation mode) shall be identified.
- (3) Identifiers shall be included in the individual software requirements specifications.
- (4) Specifications for data and databases to be handled by the software shall be included in the software requirements specifications.
- (5) Specifications for failure detection and handling functions shall be included in the software requirements specifications.
- (6) Risks, software severeness, and specifications for important quality characteristics shall be included in the software requirement specifications.
- (7) User interface (in case having), information provided to users, and specifications for user trainings shall be included in the software requirement specifications.
- (8) In case of a software transition to an on going system, specifications for the satisfaction of the

transition condition shall be included in the software requirement specifications.

- (9) Interface requirements shall be analyzed, and interface specifications shall then be developed. Agreement with the relevant parties regarding the interface specifications shall be made based on a common understanding and interpretation of the contents.
- (10) Traceability and consistency of the software requirements specifications relative to the computer system architectural design specifications and interface requirements shall be analyzed and documented.
- (11) Rationale of individual requirement of the software requirements specifications shall be clarified, and their feasibility shall be evaluated.
- (12) If COTS or reused software is used, compliance with the software requirements specifications and its applicability with the computer system architectural design specifications shall be analyzed.
- (13) Operational assumptions and constraints regarding software requirements specifications shall be extracted.
- (14) In addition to the feedback of the requirement analysis contents to appropriate stakeholders, developed software requirement specifications shall be reviewed and approved by the stakeholders.
- (15) Problems, inadequacies, and inconsistencies included in the software requirement specifications shall be identified and planned to resolve them.
- (16) Verifiability of the individual requirements of the software requirements and interface specifications shall be evaluated, and a software verification plan, including the validation method, shall be established.
- (17) Software verification coverage pertaining to software function, performance, and operational scenarios in the verification plan shall be evaluated, and test plannability regarding the software requirements specifications and interface specifications shall be evaluated.
- (18) With regard to the software verification plan, whether the test is affected by the behavioral difference between the test environment and the real hardware, or whether verification is performed by review, analysis and so on, without testing, the evaluation that shows the adequate identification and verification methods shall be included.

5.3.5.2 Measurement

In terms of evaluating stability and quality levels of software requirements specifications, the following measurement shall be performed for software requirements analysis:

- (1) The definition of the data to be collected for evaluating the stability (maturity) of the software requirements specifications and their evaluation methods shall be defined. This measurement result is used as input for the project management, and is also referenced data for the post-project.
- (2) Collection and evaluation of data defined in (1) above shall be planned.
- (3) Collection and evaluation of data defined in (1) above shall be performed, and the results recorded.

5.3.5.3 Input

- (1) Computer system architectural design specifications
- (2) Requirements for software, including operational scenarios after the analysis
- (3) Interface requirements

5.3.5.4 Output

- (1) Software requirements specifications
- (2) Interface specifications
- (3) Software requirements specification traceability and consistency evaluation results
- (4) Software requirements specification rationale and their feasibility evaluation results
- (5) COTS or reused software applicability evaluation results
- (6) Operational assumptions and constraints
- (7) Software verification plan, including validation plan
- (8) Verification coverage and test plannability for the software verification plan evaluation results
- (9) Software requirements stability (maturity) evaluation results

5.3.5.5 Review

For each output, a software requirements review shall be performed. Items to be reviewed shall be chosen based on 5.3.5.4 and shall be defined in a plan document such as the software development plan document. Appropriate follow-up of action items shall be performed in accordance with due dates, follow-up status, degree of influence, and so on. If a review is performed, it shall be documented in a technical review record after its completion. In addition, quantitative data such as the reviewers' positions, their review time, the number of questions, and their comments shall be recorded, and the quality of the review shall be evaluated.

5.3.6 Software design

In this standard, software architectural design and detailed design are not separated especially. However, in an actual development process, the software architectural design and the detailed design phases may be separated as necessary.

5.3.6.1 Activity

Activities that meet the following requirements shall be performed for the software design:

Software architectural design

- (1) The design guidelines (software architecture requirement, etc.) and the design characteristics to be considered shall be selected, and the design shall be performed by considering priorities.
- (2) Functional decomposition and module partitioning shall be performed based on the software requirements specifications and the relationships between the modules comprising the functions, and the structures between modules and themselves shall be clarified, so that an appropriate software architectural design is performed.
- (3) Software architectural design shall include the design and distribution of non-functional requirements (processing time requirements, requirements of resources such as memory, and so on) and be defined as software requirements.
- (4) Interface specifications shall be detailed in accordance with the considerations of the boundaries and interrelations, and the decomposition of the software functions and modules. Agreement with the relevant parties as to the interface specifications shall be arrived at based on a common understanding and common interpretation of the contents.

Software detailed design

- (5) Each individual module shall be designed in accordance with the decomposition of functions and modules, and a software detailed design shall be performed.

- (6) Interface specifications shall be detailed in accordance with the considerations of the boundaries and interrelations, and the design of the module. Agreement with the relevant parties on the interface specifications shall be made based on a common understanding and interpretation of the contents.

Common to software architectural and detailed designs

- (7) The needed design methods or organizationally maintained past knowledge shall be identified, prepared and acquired.
- (8) Traceability and consistency of the software design with the software requirements specifications, interface specifications, and with the necessary related documents shall be analyzed, documented and maintained.
- (9) Operational assumptions and constraints regarding the software design shall be identified.
- (10) As necessary, with regard to the individual software design, the design rationale shall be clarified and its feasibility and testability (including test case) evaluated.
- (11) If COTS or reused software is used, its applicability with the software design shall be analyzed.
- (12) Software test plans and specifications shall be established in accordance with the software verification plan.
 - (a) For software test specifications, the following shall be considered:
 - (i) Operational scenarios
 - (ii) Interface specifications
 - (iii) Maximum load for assumed scenarios
 - (iv) Coverage for software requirements specifications and software design specifications
 - (v) Anomalies, such as exceptions and failures
 - (vi) Applicability of COTS or reused software items with computer system
- (13) If new operational assumptions and constraints arise or are identified, they shall be updated.

5.3.6.2 Measurement

The following measurement shall be performed for software design in order to evaluate the progress risk of the software design:

- (1) Definition of the data to be collected and of the evaluation methods for progress management and risk evaluation of software design shall be defined.
- (2) Collection and evaluation of data defined in (1) above shall be planned.
- (3) Collection and evaluation of data defined in (1) above shall be performed, and the results recorded.

5.3.6.3 Input

- (1) Software requirements specifications
- (2) Interface specifications
- (3) Operational assumptions and constraints
- (4) COTS or reused software applicability evaluation results
- (5) Software verification plan

5.3.6.4 Output

- (1) Software architectural and detailed design specifications
- (2) Interface specifications (updated)

- (3) Software design traceability and consistency evaluation results
- (4) Software design rationales and their feasibility evaluation results
- (5) COTS or reused software applicability evaluation results (updated)
- (6) Operational assumptions and constraints (updated)
- (7) Software test plan
- (8) Software test specifications

5.3.6.5 Review

For each output, a software design review shall be performed. Items to be reviewed shall be chosen based on 5.3.6.4 and shall be defined in a document such as the software development plan document. Appropriate follow-up on action items shall be performed in accordance with due dates, follow-up status, degree of influence, and so on. If a review is performed, it shall be documented in a technical review record after its completion. In addition, quantitative data such as the reviewers' positions, their review time, the number of questions, and their comments shall be recorded, and the quality of the review shall be evaluated.

5.3.7 Not used

5.3.8 Software coding and testing

5.3.8.1 Activity

Activities that meet the following shall be performed for the software coding and testing:

- (1) Source codes shall be developed based on constraints, the software design specifications and interface specifications, and reviewed.
- (2) Definitive implementation guidelines for error handling shall be considered.
- (3) Source code shall be developed based on the defined coding standard.
- (4) Unit testing specifications shall be developed in accordance with a software verification plan and a software test plan.
- (5) Unit testing shall be performed in accordance with the unit testing specifications, and the test results shall be recorded in a format that it allows determination of pass or failure.
- (6) For unit testing, the criteria for the test coverage for source code shall be established and the test shall be performed so that these criteria are satisfied. The branch of source code shall be well covered at least.
- (7) Static analysis shall be performed with a source code checking tool or equivalent, and the source quality shall be evaluated.
- (8) The correspondence between the naming convention (variable name, function name, etc.) in source codes and the design specifications shall be considered.
- (9) If new operational assumptions and constraints arise or are identified, they shall be updated.

5.3.8.2 Measurement

The following measurement shall be performed for software coding and unit test in order to evaluate the quality:

- (1) The definition of data to be collected for evaluating source code quality, and the evaluation method, shall be defined.
- (2) Collection and evaluation of data defined in (1) above shall be planned.
- (3) Collection and evaluation of data defined in (1) above shall be performed and the results recorded.
- (4) Results of the evaluation in (3) above shall be reported periodically, or for each milestone.

5.3.8.3 Input

- (1) Software design specifications
- (2) Interface specifications
- (3) Operation assumptions and constraints
- (4) Software verification plan

5.3.8.4 Output

- (1) Source code
- (2) Operation assumptions and constraints (updated)
- (3) Unit testing specifications
- (4) Unit testing record
- (5) Traceability analysis record
- (6) Source code quality evaluation results

5.3.8.5 Review

For each output, a software coding and testing review shall be performed. Appropriate follow-up shall be performed with regard to action items in accordance with due dates, follow-up status, degree of influence, and so on. If a review is performed, it shall be documented in a technical review record after its completion. In addition, quantitative data such as the reviewers' positions, their review time, the number of questions, and their comments shall be recorded, and the quality of the review shall be evaluated.

5.3.9 Not used**5.3.10 Software integration****5.3.10.1 Activity**

Activities that meet the following shall be performed for the software integration:

- (1) Based on objectives, the integration criteria and verification points for software functions (operations of correct interfaces and completeness) shall be selected and defined.
- (2) The integration constraints included in the system/software requirements, architecture, and design shall be identified.
- (3) Software shall be integrated based on the software design specifications, and the baseline shall be determined after software integration.
- (4) Debug information collected during software integration shall be recorded and, as necessary, related processes such as the problem resolution (refer to 6.8) and configuration management (refer to 6.2) shall be implemented.

5.3.10.2 Measurement

The following measurement shall be performed for software integration for quality evaluation:

- (1) For debug information collected during software integration, the definition of data to be collected for evaluating software products quality and its evaluation method shall be defined.
- (2) Collection and evaluation of data defined in (1) above shall be planned.
- (3) Collection and evaluation of data defined in (1) above shall be performed and the results recorded.
- (4) Results of evaluation in (3) above shall be reported periodically, or at every milestone.

5.3.10.3 Input

- (1) Source code (unit)
- (2) Operation assumptions and constraints
- (3) Software design specifications

5.3.10.4 Output

- (1) Source code (integrated)
- (2) Software (integrated)
- (3) Identified results of constraints for the integration

5.3.11 Software integration test**5.3.11.1 Activity**

Activities that meet the following shall be performed for the software integration test:

- (1) Test preparation
 - (a) As the result of software coding, testing, and software integration, the software test specifications shall be updated as necessary.
 - (b) For software test, the following shall be considered:
 - (i) Operational scenarios
 - (ii) Interface specifications
 - (iii) Maximum load for assumed scenarios
 - (iv) Coverage for software requirements specifications and software design specifications
 - (v) Anomalies, such as exceptions and failures
 - (vi) Applicability of COTS or reused software items with computer system
 - (c) Software integration test procedures shall be documented in accordance with the software verification plan, the software test plan, and the software test specifications.
 - (d) If new operational assumptions and constraints arise or are identified, they shall be updated.
 - (e) Test specifications and procedures shall be checked.
- (2) Implementation of tests
 - (a) The tests shall be performed in accordance with the software integration test procedure.
 - (b) During the software integration test, quick reviews of the test results shall be performed as necessary, and judgment shall be made about whether the test shall be continued or not.
 - (c) With regard to the software integration test, information about the test environment, test data, configuration and version of the software under test shall be recorded to ensure the reproducibility of test conditions.
 - (d) Test results shall be recorded and stored appropriately, and they shall be presented as necessary.
 - (e) If software or software integration test specifications need to be revised during a software integration test, the effectiveness of the activities such as the joint reviews, verifications, and validations performed formerly shall be evaluated and tests shall be performed again as necessary.

5.3.11.2 Measurement

For software integration tests, the following measurement shall be performed in order to evaluate the quality:

- (1) Collection of quality metrics data
 - (a) Failures found during software integration tests shall be recorded, together with related information such as test cases.
- (2) Quality metrics data setting

If quality metrics other than (1) above are set, collected, and evaluated, the following shall be implemented:

 - (a) Metrics for quality evaluation during tests shall be set
 - (b) Identified data shall be collected
 - (c) Analysis evaluation method for the identified data shall be defined
 - (d) Analysis and evaluation of identified data shall be performed
- (3) Result of data analysis and evaluation shall be reported periodically or for each milestone

5.3.11.3 Input

- (1) Interface specifications
- (2) Software requirements specifications
- (3) Software design specifications
- (4) Software verification plan
- (5) Software test plan
- (6) Software test specifications
- (7) Operation assumptions and constraints
- (8) Source code (integrated)
- (9) Software (integrated)
- (10) Operational scenarios

5.3.11.4 Output

- (1) Software integration test procedure
- (2) Software integration test record, including pass or failure judgment results
- (3) Operation assumptions and constraints (updated)
- (4) Source code (tested)
- (5) Software (tested)
- (6) Software test specifications (updated)

5.3.11.5 Review

For each output, a software test review shall be performed. Items to be reviewed shall be chosen based on 5.3.11.4 and shall be defined in the relevant plan document, such as the software development plan. Appropriate follow-up on action items shall be performed in accordance with due dates, follow-up status, degree of influence, and so on. If a review is performed, it shall be documented in a technical review record after its completion. In addition, quantitative data, such as the reviewers' positions, their review time, the number of questions, and their comments shall be recorded, and the quality of the review shall be evaluated.

5.3.12 Software installation into target platforms (embedding)

5.3.12.1 Activity

Activities that meet the following shall be performed for the software installation into target platforms

(embedding):

- (1) Software shall be prepared in a form that allows installation (embedding) into the target platform, and the configuration management information (filename, version information, and so on) of the software shall be acquired.
- (2) Configuration management information of software to be released shall be prepared, to include the installation plan, schedule and procedure into the target platform, the installation result check procedure, and the operational assumptions and constraints.
In principle, in planning an installation, the following shall be considered
 - (a) Identification of the enabling system/service needed for the installation and the acquisition of the access right of them
 - (b) Set an advanced verification for the installation
 - (c) Provision and support for the software or the installation service (provide supports for the acquirer)
- (3) Before installation, a check of the installation preparation status (whether operational conditions/constraints have been resolved, the advanced verification has been done, and so on) shall be performed. After that, software shall be installed into the target platform in accordance with the installation (embedding) procedure. However, this can be omitted if the software has already been installed into the target platform.
- (4) A check that the software has been properly installed shall be performed, in accordance with the results check procedure and confirmed results.

5.3.12.2 Input

- (1) Software
- (2) Operation assumption and constraints

5.3.12.3 Output

- (1) Software prepared in a form that allows installation (embedding) into the target platform
- (2) Configuration management information
- (3) Installation plan, schedule, and procedure
- (4) Computer system installed (embedding) software
- (5) Installation result check procedure, confirmed result

5.3.13 Computer system integration and computer system integration test

5.3.13.1 Activity

Activities that meet the following shall be performed for the computer system integration and computer system integration test:

- (1) Test preparation
 - (a) Computer system integration shall be performed with the identification of the integration constraints included in the system/software requirements, architecture and designs.
 - (b) Based on objectives, the computer system integration criteria and verification points for software functions (operations of correct interfaces and completeness) shall be selected and defined.
 - (c) Computer system integration test specifications and procedures shall be documented in accordance with the software verification plan.

- (d) With regard to the computer system integration test specifications, the following viewpoints shall be considered:
 - (i) Operational scenarios
 - (ii) Software requirements specifications
 - (iii) Maximum load
 - (iv) Coverage regarding requirements specifications for a computer system and computer system architectural design specifications
 - (v) Anomalies, such as exceptions and failures
 - (vi) Applicability of COTS or reused software items with the computer system
- (e) Incidents and problems found during test preparation (test procedure checks, and so on) shall be recorded and managed.
- (2) Implementation of tests
 - (a) The tests shall be performed in accordance with the computer system integration test procedure.
 - (b) As necessary, based on the operational scenarios, tests shall be performed by using tools such as simulators, and the verification coverage regarding operational scenarios shall be checked.
 - (c) With regard to the computer system integration test, information about the test environment, test data, configuration, and version of the software under test shall be recorded to ensure the reproducibility of test conditions.
 - (d) Test results shall be recorded and stored appropriately.
 - (e) When software or computer system integration test specifications need to be revised during a computer system integration test, the performance effectiveness of the activities such as the joint reviews, verification, validation, tests, and others shall be evaluated and performed again, if necessary.

5.3.13.2 Measurement

The following measurement shall be performed for computer system integration and computer system integration tests in order to evaluate the quality:

- (1) Collection of quality metrics data
 - (a) Failures found during the computer system integration tests shall be recorded, together with related information such as information about test cases.
 - (b) Incidents and problems found during the test preparation (test procedure checks, and so on) shall be recorded, together with related information such as information about test cases.
- (2) Quality metrics data definition

If a quality metrics other than (1) above is defined, collected, and evaluated, the following shall be implemented:

 - (a) The metrics for quality evaluation during computer system integration tests shall be defined.
 - (b) Identified data shall be collected.
 - (c) The analysis evaluation method for the identified data shall be defined.
 - (d) Analysis and evaluation of identified data shall be performed.
- (3) The result of data analysis and evaluation shall be reported periodically, or for each milestone.

5.3.13.3 Input

- (1) Requirements specifications for computer system

- (2) Computer system architectural design specifications
- (3) Software requirements specifications
- (4) Software verification plan
- (5) Operational assumptions and constraints
- (6) Operational scenarios
- (7) Computer system installed software

5.3.13.4 Output

- (1) Computer system integration test specifications
- (2) Computer system integration test procedure
- (3) Computer system integration test record, including pass or failure judgment results
- (4) Operational assumptions and constraints (updated)
- (5) Constraint identification results for the integration (after updated)

5.3.14 Supply and introduction of software product

5.3.14.1 Activity

Activities that meet the following requirements shall be performed for the supply and introduction of software product:

- (1) Documentation of manuals
 - Software user's manuals shall be documented.
- (2) Preparation for supply
 - The software that is ready to be supplied shall be confirmed, and the confirmation results shall be recorded.
- (3) Establishment of an introduction plan
 - The introduction of software shall be planned, including replacement of the existing system and temporary parallel operation, and a procedure shall be documented. For the introduction plan and procedure, not only installation (embedding) of the software into the target platform, but also work for introducing computer system into the actual operational environment, shall be considered.
- (4) Implementation of introduction and recording of results
 - Software shall be installed (embedded) into the target platform in accordance with the introduction plan and procedure, and the computer system shall be introduced to the actual operational environment. The results of the introduction shall be recorded.

5.3.14.2 Input

- (1) Software requirements specifications
- (2) Software design specifications
- (3) Source code or software
- (4) Installation (embedding) procedure

5.3.14.3 Output

- (1) Software user's manual
- (2) Source code or software
- (3) Confirmation results that software is ready to be supplied

- (4) Installation (embedding) procedure
- (5) Introduction plan
- (6) Introduction procedure
- (7) Record of introduction results

5.3.15 Software acceptance

5.3.15.1 Activity

Activities that meet the following shall be performed for the software acceptance:

- (1) With regard to acceptance inspection and acceptance testing, plans shall be established, subsequently, specifications and procedures shall be documented. If this is to be substituted by tests performed by the suppliers, the acquirers' approval shall be needed regarding the content of those tests.
- (2) Acceptance inspection and acceptance testing shall be performed in accordance with the plans, the specifications and the procedures described in (1). Records of the acceptance inspection and acceptance testing shall be maintained.
- (3) A review shall be performed regarding confirmation of the results that the software to be acquired is ready to be supplied.

5.3.15.2 Input

- (1) Software user's manual
- (2) Source code or software
- (3) Configuration management information
- (4) Record of confirmation that software is ready to be supplied
- (5) Installation (embedding) procedure

5.3.15.3 Output

- (1) Acceptance inspection and testing plan
- (2) Acceptance inspection and testing specifications
- (3) Acceptance inspection and testing procedure
- (4) Acceptance inspection and testing record

5.4 Operation process

The purpose of the Operation process is to operate a computer system including software in an intended environment and to provide supports to customers and users.

This process consists of the following activities

- (1) Process implementation
- (2) Operational testing
- (3) Operation of the computer system including software
- (4) Management of operational results
- (5) Customer and user support

5.4.1 Process implementation

5.4.1.1 Define an operation strategy

- (1) An operation strategy shall be defined. In principle, the following shall be considered in the

strategy.

- (a) Expected each criteria(capacity, occupancy rate, response, safety, etc.) between service installation, periodic operation and disposal
- (b) Software or computer system release criteria and schedule considering the correction to maintain the current service
- (c) Method to implement each operation mode(regular operation/preparation stage/operation test/assumed occurrence of disasters and troubles)
- (d) Operation metrics to evaluate performance level

5.4.1.2 Plan an operation

A plan to perform the operation process based on the operation strategy shall be planned and executed. Additionally, a standard related to the operation shall be set. In principle, in planning, the following shall be considered.

- (1) Constraints which exist within the transformation from software or computer system requirements to architecture, design, implementation and transition shall be identified.
- (2) Systems or services needed to support the operation shall be identified.
- (3) The access to the enabling systems or services to be used shall be obtained or acquired.

5.4.1.3 Establishment of problem management for the operation

In accordance with the problem resolution process (refer to 6.8), a problem report handling procedure shall be developed e.g. for receiving problem reports, recording, resolving, tracking problems, and notice of the status.

5.4.1.4 Establishment of operational procedures for the computer system including software operation and user support

The following procedures shall be established.

- (1) The procedure for testing under the operational environment for the computer system including software
- (2) The procedure for modification request to the maintenance process (refer to 5.5)
- (3) The procedure for releasing the software or computer system for operational use.

5.4.2 Operational testing

An environment to perform the operational testing shall be established. For each release of the software or the computer system, operational testing shall be performed. The fact that the operational testing has been performed as planned and finished shall be confirmed, and the review for them shall be performed. After the software or the computer system satisfies the specified criteria, it shall be released for operational use.

5.4.3 Operation of computer system including software

The tasks that fill the following requirements shall be performed.

- (1) Operate in the designated operational environment followed the software user's manual
- (2) Apply facilities and resources as needed to maintain operation and service
- (3) Consider the followings and monitor the operation
 - (a) Conformity to the operation strategy(such as procedures related the operation)
 - (b) Records and reports of the possible invasions to the software and the serious matters such as

data confidentiality and completeness

- (c) Record that the software, computer system or service capacity is out of the tolerable parameter
- (4) According to the operation strategy, develop operation procedures to minimize operation trouble risks and automate them as much as you can
- (5) According to the operation strategy, analyze the measurement results to confirm the followings
 - (a) The service capacity is in the tolerable parameter or in the agreed service level for agreed work volume
 - (b) Availability and response for the software or computer system, and service are in the tolerable range
 - (c) Possible improvement items are identified and prioritized
- (6) Perform the disaster and trouble operations as necessary

5.4.4 Operation results management

This activity consists of the following tasks

- (1) The results of operation and anomalies encountered shall be recorded.
- (2) Operational incidents and problems shall be recorded and their resolution shall be tracked
- (3) The traceability of the operational services and configuration items shall be maintained.
- (4) Key artifacts and information items that have been selected for baselines shall be provided.

5.4.5 Customer and user support

The following tasks shall be performed for the customer and user support:

- (1) Support service shall be provided to the customers and users to resolve incidents, problems and service requests. These requests and subsequent actions taken for support shall be recorded and managed to provide support adequately.
- (2) If a problem is identified in the user support, it shall be resolved in accordance with the procedure established in 5.4.1.3.
- (3) If there is a temporary work-around for identified problem, it shall be provided to customers and users.
- (4) Determine the degree of the satisfaction to which the delivered software/computer system or services satisfy the needs of the customers and users.

5.5 Maintenance process

The purpose of the Maintenance process is to sustain the capability of the software or the computer system to provide service and the maintenance process is defined when the modification and maintenance of the software products, including design documents, requires a change of development environments.

This process consists of the following activities:

- (1) Process implementation
- (2) Problem identification and modification analysis
- (3) Modification implementation
- (4) Software reprogramming
- (5) Logistics support implementation
- (6) Management of the maintenance and logistics result
- (7) Transition
- (8) Software disposal

5.5.1 Process implementation

5.5.1.1 Define a maintenance strategy

- (1) A maintenance strategy shall be defined. The followings shall be considered in the strategy.
 - (a) Establishing priorities, typical schedules, and procedures for performing, verifying, distributing, and installing software maintenance changes in conformance with operational availability requirements
 - (b) Establishing techniques and methods for becoming aware of the need for correction
 - (c) Establishing priorities and resources to obtain access to the correct versions of the product and product information needed for performing maintenance (e.g., scheduled or phased installation, maintenance patches or software upgrades)
 - (d) Agreed rights to data and the impact on data in the software or computer system during problem resolution and maintenance activity
 - (e) Approach to assure that counterfeit or unauthorized software elements are not introduced into the software or computer system
 - (f) Impact of the maintenance change on other software systems elements versus the risk of leaving a reported software anomaly in place
- (2) For non - software elements, a logistics strategy (number of the store target elements, variation, location, condition, expected replace ratio, duration, and update frequency) shall be defined. Logistics helps to ensure that the necessary material and resources, in the right quantity and quality, are available at the right place and time.
- (3) Constraints from maintenance to be incorporated in the software or computer system requirements, architecture, or design shall be defined.
- (4) Trades such that the maintenance and logistics actions result in a solution that is affordable, operable, and sustainable shall be coordinated.
- (5) The necessary enabling systems or services needed to support maintenance shall be identified and planned.
- (6) The access to the enabling systems or services to be used shall be identified or acquired.

5.5.1.2 Document a maintenance plan

The maintenance plan for implementation of the maintenance process shall be documented in accordance with the maintenance strategy and implemented. The following shall be included in the plan:

- (1) Maintenance management method for developed software products
- (2) The structure and related documentation
- (3) Data, including the management information necessary for maintenance
- (4) Record
- (5) Maintenance environment (environment of development phase, and so on)
- (6) Actions for maintaining other related processes appropriately (problem resolution process (refer to 6.8), configuration management process (refer to 6.2), and so on)

5.5.2 Problem identification and modification analysis

- (1) In accordance with the plans, the analysis of the requested contents regarding maintenance or modification of the software product, and correspondence shall be evaluated.
- (2) If the software modification and reprogramming are implemented, agreement for the

modification plan shall be obtained.

5.5.3 Modification implementation

If a modification is needed, it shall be implemented in accordance with the modification implementation plan and procedures. The corresponding process of this standard shall be implemented again, as necessary.

5.5.4 Software reprogramming

This standard shall also be applied to the work such as the partial modification of the software which is called patches, and to the addition and extension of software functions during operational periods. In principle, upon encountering unexpected faults that cause a system failure while reprogramming, the software or the computer system shall be restored to the operational status.

*This does not imply that their development processes shall be exactly the same as the development-to-retirement processes of the software that is being used for operations. For projects assuming reprogramming, the development-to-retirement processes regarding reprogramming shall be concretized and tailored in advance.

In principle, this standard shall be applied and the compatibility shall be evaluated.

5.5.5 Perform logistics support

This activity consists of the following tasks.

- (1) Obtain resources to support the software product through project lifecycle
- (2) Monitor the quality and availability of the reprogrammed software elements and enabling systems, their distributing mechanisms and their continued integrity during storage.
- (3) Implement mechanisms for software product distribution
- (4) Confirm that logistics action to fulfill software product supportability requirements are planned and implemented.

5.5.6 Manage results of maintenance and logistics

This activity consists of the following tasks.

- (1) Record incidents and problems, including their resolutions, and significant maintenance and logistics results.
- (2) Identify and record trends of incidents, problems, and maintenance and logistics actions.
- (3) Maintain traceability of the software or computer system elements being maintained.
- (4) Provide key artifacts and information items that have been selected for baselines.
- (5) Monitor and measure customer and user satisfaction with software product or computer system and maintenance support.

5.5.7 Transition

This activity is a process for transitioning software to a new environment.

5.5.7.1 Define a transition strategy

A transition strategy as a preparation for software product transition shall be defined. In principle, the following shall be considered.

- (1) Establishing the type of transition and transition success criteria

- (2) Determining the frequency of recurring transitions, such as updates and upgrades to development, test, and operational software or computer systems
- (3) Minimizing security risks, disruption, and downtime during transition
- (4) Archiving, destroying, or converting and validating data from previous systems to the new system; including data received through external interfaces
- (5) Contingency planning(problem resolution, backup and return to the last working system version)
- (6) Scheduling transitions consistent with ongoing business processing, with phased or synchronized transition of systems
- (7) Change management for stakeholders, including interface partners, human operators, system administrators, and software system or service users
- (8) Associated strategies for validation of the transitioning system or element
- (9) Initiating user support and maintenance activities with the transfer and update of system design documentation, user documentation, and test specifications
- (10) Concurrent execution of the Transition, Operational, and Disposal processes, when a new system is commissioned and an old system is decommissioned

5.5. 7.2 Development and execution of transition plan

Based on the transition strategy, a transition plan shall be developed and executed.

- (1) Integrated software requirements according to the transition, and requirements analysis and definition including constraints of architecture/design
- (2) Frequency of transitions and their schedules
- (3) Facilities, sites, communication network and hardware environment that are needed to be changed because of the transition (or new release)
- (4) Documents for related people such as operators, users and system support members, and trainings
- (5) Tools for the transition(including enabling systems/services)
- (6) Conversion of software product and data
- (7) Transition rehearsal
- (8) Transition execution
- (9) Transition confirmation
- (10) Future support for old environments

5.5. 7.3 Notification to users

The migration plan and contents of implementation shall be notified to users. Notifications shall include the following viewpoints:

- (1) Statement of why the old environment is no longer to be supported
- (2) Description of the new environment, with its date of availability
- (3) Description of other support options available, if any, once support for the old environment has been removed

5.5.7.4 Manage transition results

The following records accompanied by the software transitions shall be managed.

- (1) Incidents and problems happened during transitions
- (2) Maintenance of the traceability of the transitioned software elements
- (3) Key artifacts and information items that have been selected for baselines

5.5. 7. 5 Storage of old environments

The records, such as documentation, logs, and other items that relate to old environments shall ideally be retained.

5.5. 8 Software disposal

The following viewpoints shall be considered for the software disposal:

- (1) A disposal strategy to remove active support by an institution engaged in operation and maintenance shall be defined. In principle, the following shall be considered in the strategy.
 - (a) Identification of permanent termination of the system's functions and delivery of services
 - (b) Identification of ownership and responsibility for retention or destruction of data and intellectual property in the software system
 - (c) Transformation of the product into, or retention in a socially and physically acceptable state, thereby avoiding subsequent adverse effects on stakeholders, society and the environment
 - (d) The health, safety, security and privacy concerns applicable to disposal actions and to the long-term condition of resulting physical material and information.
 - (e) Notification to relevant stakeholders of significant disposal activities.
 - (f) Identification of schedules, actions, responsibilities, and resources for disposal activities.
 - (g) Identification of the constraints on disposal for system/software requirements, architecture and design characteristics, or implementation techniques.
 - (h) Identification and planning for the necessary enabling systems or services needed to support disposal.
 - (i) Obtainment or acquisition of the access to the enabling systems or services to be used.
 - (j) Containment facilities, storage locations, inspection criteria and storage periods, if the software system or data is to be stored.
 - (k) Preventive methods to preclude disposed software.
- (2) A disposal plan shall be developed based on the disposal strategy. Users shall be included in the development of the plan.
- (3) Users shall be given notification of the disposal plans and activities. Notifications shall include the following:
 - (a) Description of the replacement or upgrade of to the software or computer system with its date of availability
 - (b) Statement of why the software or computer system is no longer to be supported
 - (c) Description of other support options available, once support has been removed
- (4) According to the disposal plan, the following activities shall be performed.
 - (a) Parallel operation of the retiring and the new software or computer system shall be performed, for smooth transition to the new computer system. It is recommended that user training shall be provided during this period.
 - (b) Remove the target software, computer system and data.
 - (c) Set the new software or computer system to the last status of the old system or the status already decided.
 - (d) Reuse, recycle, recondition, overhaul, archive, or destroy the removed software/computer system/data.
- (5) When the scheduled disposal arrives, notification shall be sent to all concerned parties. It is recommended that all associated development documentation, logs, and code shall be archived.

- (6) For the disposed software or computer system, data shall be collected to permit audits and reviews in the event of long-term hazards to health, safety, security, privacy, and the environment.

6 Supporting life cycle process

This clause defines the following supporting life cycle processes:

- (1) Documentation process
- (2) Configuration management process
- (3) Quality assurance process
- (4) Verification process
- (5) Validation process
- (6) Joint review process
- (7) Assessment process
- (8) Problem resolution process

The activities and tasks in a supporting process are the responsibility of the organization performing that process. This organization ensures that the process is in existence and functional.

6.1 Documentation process

This process consists of the following activities:

- (1) Process implementation
- (2) Development
- (3) Production
- (4) Maintenance/Revision/Disposal

6.1.1 Process implementation

A plan including the following shall be established as a documentation plan:

- (1) Documents developed through the software and computer system life cycle, and the documents schedule for development shall be clarified.
- (2) Procedures for the development of documents (development, inspection, and approval), issue (issue, distribution, and storing), and revision (revision and disposal) shall be decided.
- (3) The form (content, format, and so on) appropriate for such documents shall be decided.

6.1.2 Development

Documents shall be developed in accordance with the decided procedure. The following shall be considered:

- (1) The appropriateness of sources of information for the documents shall be confirmed.
- (2) The documents shall be checked and approved regarding format, technical content, against their documentation standards.

6.1.3 Production

Documents production shall be implemented in accordance with the defined procedures. The following shall be considered:

- (1) Production and distribution of documents shall be at the appropriate version.
- (2) The documents shall be managed in accordance with requirements (security management, backup, and so on) including parties for distribution.

6.1.4 Maintenance/Revision/Disposal

Documents shall be maintained, revised or disposed in accordance with the decided procedure. In addition, the following shall be considered.

- (1) In documentation, the integrity and anonymity requirement of the information included in that shall be considered.
- (2) The revised documents shall be checked and approved regarding the form and technical content.
- (3) Needless, ineffective or valid-less documents shall be disposed.

6.2 Configuration management process

Configuration management process is a process of applying the following administrative and technical procedures through the software life cycle:

- (1) Process implementation
- (2) Configuration identification
- (3) Configuration change control
- (4) Record of configuration change status
- (5) Evaluation of configuration change status
- (6) Release management and delivery

6.2.1 Process implementation

6.2.1.1 Configuration management strategy

A configuration management strategy shall be defined. In principle, the following shall be considered in the strategy.

- (1) Control of software licenses, data authorities, and other intellectual property assets
- (2) Frequency of releases, priorities, and content of software versions and software
- (3) The audit strategy and the responsibilities for validating continuous integrity and security of the configuration definition information
- (4) Change management, including users in operational software/services

6.2.1.2 Configuration management plan

A configuration management plan shall be developed according to the configuration management strategy. In principle, the following shall be included in the plan.

- (1) Configuration management activities
 - (a) Consideration of the level of risk and degree of influence in approval of configuration baselines, and regular and emergency change requests
- (2) Procedures and schedule for performing the activities
 - (a) The necessary baseline to be established, including criteria for accessing and changing of configuration items, controlling of actions, commencing configuration control and maintaining baselines of evolving configurations.
- (3) The organization responsible for performing the activities.
 - (a) Roles and responsibilities of the configuration management activities including the operation of the configuration control boards
- (4) The relationship with other organizations
 - (a) Coordination of the configuration management across the set of acquirer, supplier, and supply

chain organizations for the lifecycle of the software, or the extent of the agreement or project, as appropriate.

- (b) Configuration management and change management plan, including users in operated software/services
- (5) Store, archive and access procedure for configuration items, configuration management work products and records

6.2.2 Configuration identification

- (1) Configuration items shall be identified.
- (2) For configuration items, the following shall be identified:
 - (a) Version references
 - (b) Other identification details
- (3) For managed items, baselines shall be established according to their objectives.
- (4) Obtain stakeholders agreement to the established configuration baselines.

6.2.3 Configuration change control

The following shall be identified and defined for the configuration changes:

- (1) Identification and recording of change requests(including waiver requirements)
- (2) Analysis and evaluation of changes
- (3) Approval or disapproval of change requests
- (4) Implementation, verification, and release of modified configuration items
- (5) Traceable review records against the reason for the modification, and authorization of the modification
- (6) Not used

6.2.4 Record of configuration change status

Management records and status reports that show the status and history of controlled configuration items including baselines shall be prepared and the status shall be reported.

6.2.5 Evaluation of configuration change status

According to the planned schedule in advance, the following confirmations shall be performed.

- (1) The functional completeness of the configuration items against their requirements and the physical completeness of the configuration items
 - (2) The validities of the approved configuration changes
- Confirmed results and required actions shall be recorded.

6.2.6 Release management and delivery

The release and delivery of software product shall be formally controlled in accordance with the procedure. The source code and documentation that contain safety or security critical functions shall be handled, stored, packaged, and delivered in accordance with the policies of organizations involved.

The distribution of the assigned software product releases shall be managed.

6.3 Quality assurance process

The quality assurance process is a process for providing adequate assurance that the process activities and outputs, of the software or computer system life cycle and the project life cycle, adhere to their

established plans which are based on this standard or by tailoring its results.

This process consists of the following:

- (1) Process implementation
- (2) Products and services quality assurance
- (3) Process assurance
- (4) Assurance of quality system
- (5) Management of the quality assurance record

6.3.1 Process implementation

This activity consists of the following tasks:

6.3.1.1 Confirmation of the independence of the Organization and establishment of the adequate structure

The management organization of the quality assurance process shall be clarified, and the implementation status and validity of quality assurance process shall be reported.

The management organization of the quality assurance process shall be authorized to maintain organizational freedom and authority to assess the problem and offer a resolution.

In addition, the person who is invested with all the responsibilities and authority for quality assurance process shall be independent of the development organization.

6.3.1.2 Quality assurance activity strategy

- (1) A quality assurance strategy shall be defined. In principle, the followings shall be considered in the strategy.
 - (a) With priorities, the Quality Assurance resources to processes and tasks that have the most significant impact on the quality of the delivered products and services shall be applied.
 - (b) Defined responsibilities and authorities
 - (c) Evaluation criteria and methods for processes, products, and services, including criteria for product or service acceptance
 - (d) Quality Assurance activities to each supplier (including subcontractors)
 - (e) Verification, validation, monitoring, measurement, review, inspection, audit, and test activities specific and required to the products or services
 - (f) Problem resolution and process and product improvement activities

6.3.1.3 Quality assurance activity plan

Establish a quality assurance plan based on the quality assurance strategy.

The quality assurance activity plan shall include the following:

- (1) Identification of the system to be applied
- (2) Resources, quality standards, methodologies, procedures, and tools needed for performing the quality assurance process (including identification of all documents)
- (3) Selected activities and tasks from the processes, such as the verification process (refer to 6.4), the validation process (refer to 6.5), the joint review process (refer to 6.6), the assessment process (refer to 6.7), and the problem resolution process (refer to 6.8)
- (4) Schedules
- (5) Procedures for review
- (6) Procedures for the works, such as identification, collection, filing, maintenance, disposition and

disposal of quality assurance activity records

- (7) Requirements and process regarding quality assurance for purchase management and supplier
- (8) Management for existing software items (COTS items or knowledge assets)
- (9) Procedures for the delivery
- (10) Procedures for process and product improvement activities in problem identifications and their resolutions.

6.3.2 Product and service quality assurance

- (1) Products and services shall be assured to fulfil their plans.
- (2) Through the verification and validation of the software product or computer system development process for each product, the product shall be assured to fulfil its approved requirement.
- (3) Products quality shall be assured by IV&V as necessary.

6.3.3 Process assurance

- (1) It shall be assured that the software development plans, operational plans, and processes defined by maintenance plans, comply with this standard.
- (2) It shall be assured that the software developments perform in accordance with the processes defined in a development plan, an operational plan, or a maintenance plan.
- (3) It shall be evaluated that tools and environments that support or automate the processes are usable in the processes of this standard.

6.3.4 Assurance of quality system

It shall be assured that the quality system contains the quality management tasks listed below:

6.3.4.1 Education and training

All techniques, abilities, and qualifications needed for personnel engaged in development, maintenance and operation work with the software or computer system shall be identified, and education and training shall be conducted.

6.3.4.2 Purchase management and supplier management

- (1) Purchase management
The reliability and quality of purchased items (COTS included) shall meet the software quality assurance requirements of the organization for the developing software product.
- (2) Suppliers and purchase vendor selection
Based on the capability evaluation and selection record regarding suppliers and purchase vendor maintained in an organization, suppliers and purchase partners shall be selected.

6.3.4.3 Management of items supplied by acquirer

Procedures for inspection at the time of accepting items supplied or lent from the acquirer, and procedures for their storage and maintenance management, shall be established and followed.

6.3.4.4 Management of existing software items (COTS or knowledge assets)

With regard to existing software items (COTS or reused software items), the following shall be included in the management items. The items below are to be managed for applied projects and future reuse:

- (1) Any objectives, reasons, and benefits in using existing software items

- (2) Evaluation items and levels that judge the availability of existing software items
 - (a) Applicability of existing software with regard to the developing software
 - (b) Traceability relative to requirements applied to development software items
 - (c) Risk obtained from the information such as the past performance of product, and so on, of software items to be used
 - (d) Acceptance and assurance conditions
- (3) Consideration points and items in managing the use of the existing software items
 - (a) Associated documents, which are obtainable and usable
 - (b) Introduction, preparation, training, and constraints
 - (c) Identification of versions and other details, and the configuration management method
 - (d) Maintenance and future support
 - (e) The rights such as intellectual property rights
 - (f) Identification of newly available knowledge assets

6.3.4.5 Handling, storing, and labeling

To ensure appropriate handling, storing, and labeling of software or computer system, requirements which include the following items, shall be documented, and the products shall be released in accordance with that document:

- (1) Media shall have labels (names, identifiers, and so on) so that stored software can be identified.
- (2) Identification of software shall be checked when reading software from media.

6.3.5 Manage quality assurance records

- (1) Create records related to quality assurance activities
- (2) Maintain and store the records.

6.4 Verification process

The purpose of the verification process is to provide objective evidence that a computer system fulfils its specified requirements and characteristics. The verification process consists of the following activities. This process may include the checking works such as test, review, analysis, and so on:

- (1) Process Implementation
- (2) Verification
- (3) Management of the verification results

6.4.1 Process implementation

6.4.1.1 Verification strategy

- (1) A verification strategy shall be defined. In principle, the followings shall be considered in the verification strategy.
 - (a) Identify the verification scope(including the target software, items and products), the characteristics to be verified, and the expected verification results. In principle, the characteristics include system/software requirements, architecture, design characteristics such as security and important quality characteristics, integration, and the correctness of the documents.
 - (b) Identify the constraints that potentially limit the feasibility of verifications.
 - (c) Identify the priorities of the verification scenarios.
- (2) Identify the integration constraints included in the system/software requirements, the architecture,

and designs.

6.4.1.2 Establish the verification process

- (1) Based on the verification strategy, the verification plan including the followings shall be established.
 - (a) The target for verification shall be determined, and appropriate tasks defined in 6.4.2 below shall be selected according to degree of importance.
 - (b) Appropriate methods or technologies of the verification (inspection/analysis/demonstration/test), and the standard about the verification shall be selected.
 - (c) The verification procedures and a series of activities shall be defined. The procedures include how to record, analyze, store, and report the verification results.
 - (d) Identify and acquire access to the enabling systems or services needed to support verification.
 - (e) The verification (test, review, analysis and so on) shall be performed according to the verification plan. Problems detected through verification shall be resolved in accordance with the problem resolution process (refer to 6.8).

6.4.2 Verification

This activity consists of the following tasks:

6.4.2.1 Process verification

The following shall be considered:

- (1) Adequacy of the processes selected for the project
- (2) Planning of the processes is adequate
- (3) Applicable standards and environment for the project's processes in place
- (4) Adequate levels of competent staff allocated to the processes
- (5) Proper execution of the processes

6.4.2.2 Requirements verification

The following shall be considered:

- (1) The requirements are consistent, feasible, and verifiable.
- (2) Requirements for software items are appropriately allocated (not including requirements for hardware items and operation).
- (3) The higher level requirements and the standards applied to items shall be satisfied.
- (4) Concerning to the requirement to be especially taken care such as safety and security and so on, it is able to show the proper method that it satisfies the higher level requirements and the standards applied to items.

6.4.2.3 Design verification

The following shall be considered:

- (1) The design shall meet requirements, and shall be traceable to requirements.
- (2) Designed properly with respect to data interface, timing, computer resource (memory capacity, processing speed, and so on), logic design, processing sequence and processing contents (especially initialization, termination, exception handling and so on).
- (3) The characteristics such as portability, modifiability and ease of problem resolution have been

covered.

6.4.2.4 Source code verification

The following viewpoint shall be considered:

- (1) Source code shall meet design, and shall be traceable to the design.
- (2) Implemented properly with respect to data interface, timing, computer resources (memory capacity, processing speed, and so on.), logic design, processing sequence and processing contents (especially initialization, termination, exception handling and so on).
- (3) The characteristics such as portability, modifiability and ease of problem resolution have been covered.
- (4) Concerning to the source code to be especially taken care such as safety and security and so on, it is able to show the proper method that it satisfies the requirements and the standards applied to items.
- (5) Source code shall conform to e.g. coding standards.

6.4.2.5 Integration verification

The following viewpoint shall be considered:

- (1) The configuration of the software modules and data is in proper and correct versions. The software components and units of each software item have been completely and correctly integrated into the software item.
- (2) The modules and data configuring software have been completely and correctly integrated into the system and the traceability of the integration has been maintained.
- (3) The integration tasks have been performed in accordance with the plan.

6.4.2.6 Documentation verification

The following shall be considered:

- (1) The documentation is adequate and consistent.
- (2) The documentation has been organized in accordance with the plan.
- (3) Configuration management of the documentation is performed in accordance with the proper procedures.

6.4.3 Manage the verification results

Performed verification results shall be managed. This activity consists of the following tasks.

- (1) Maintain traceability of the verified software and items.
- (2) Provide key artifacts and information items (for example, verification strategy, verification procedure) that have been selected for baselines.

6.5 Validation process

The purpose of the validation process is to provide objective evidence that the computer system, when in use in its intended operational environment and ways, fulfils its expected mission objectives and stakeholder requirements.

This process consists of the following activities:

- (1) Process implementation
- (2) Validation
- (3) Management of the validation result

6.5.1 Process implementation

6.5.1.1 Validation strategy

- (1) A validation strategy shall be defined. In principle, the followings shall be considered in the validation strategy.
 - (a) Identification of the validation scope(including the target software, items and products) and the expected validation results.
 - (b) Identification of the constraints that potentially limit the feasibility of validations.
 - (c) Identification of the priorities of the validation scenarios.
- (2) Identify the integration constraints included in the system/software requirements, the architecture, and designs.

6.5.1.2 Validation plan

- (1) Based on the validation strategy, the validation plan shall be established and include the following:
 - (a) Not used.
 - (b) A validation plan shall be developed. The plan shall include the following:
 - (i) Items subject to validation
 - (ii) Validation tasks to be performed
 - (iii) Resources, responsibilities, and schedule for validation
 - (iv) Procedures for distributing validation reports
 - (c) Appropriate methods or technologies of the validation and the validation criteria shall be selected.
 - (d) The validation procedures and a series of activities shall be defined. The procedures include how to record, analyze, store, and report the validation results.
 - (e) Identify and acquire access to the enabling systems or services needed to support validation.
 - (f) Validation shall be performed according to the validation plan. Incidents and problems detected by validation shall be resolved in accordance with the problem resolution process (refer to 6.8).

6.5.2 Validation

- (1) Test requirements and test cases shall be selected for validation, and test specifications shall be prepared.
- (2) Test cases shall be checked to ensure they reflect the method and use of the software or computer system.
- (3) Tests in (1) and (2) above shall be performed to include the following view points, as necessary:
 - (a) Not used
 - (b) Fault testing
 - (c) Testing that representative users can successfully achieve their intended tasks
- (4) Validation shall be performed so that the software or computer system satisfies its intended use.
- (5) Tests for the software or computer system shall be performed as appropriate in the target environment. When a simulated instead of a real environment is used, the difference between them shall be evaluated.

*As a validation method, methods other than testing (analysis, modeling, simulation, and so on) may be appropriate.

6.5.3 Manage the validation results

Performed validation results shall be managed. This activity consists of the following tasks.

- (1) Review validation results and anomalies encountered and identify follow - up actions.
- (2) Record incidents and problems during validation and track their resolution.
- (3) Obtain stakeholder agreement that the software system or element meets the stakeholder needs.
- (4) Maintain traceability of the validated software and items.
- (5) Provide key artifacts and information items that have been selected for baselines.

6.6 Joint review process

The joint review process consists of the following activities:

- (1) Process implementation
- (2) Project management reviews
- (3) Technical reviews

6.6.1 Process implementation

- (1) Periodical reviews shall be held at predetermined milestones, as specified in the project plans.
It is recommended that ad hoc reviews shall be called when deemed necessary by either reviewing party or reviewed party.
- (2) All resources required to perform the reviews shall be agreed on between all the parties. These resources include personnel, location, facilities, hardware, software, and tools.
- (3) It is recommended that all the parties concerned shall agree on the following at each review:
 - (a) Matters to be reviewed
 - (b) Review scope and viewpoint
 - (c) Review method
 - (d) Entry and exit criteria for the review
- (4) Incidents and problems detected during the reviews shall be recorded, and appropriate action taken through the problem resolution process (refer to 6.8), as necessary.
- (5) The review results shall be documented and distributed.
- (6) All the parties concerned shall agree on the outcome of the review and any action item responsibilities and closure criteria.

6.6.2 Project management reviews

The software project status shall be evaluated against the applicable project plans, and risks to the project's complete development in accordance with the plan shall be managed. If a project delay is detected and it is difficult to complete the development according to the plan, a change of plan including a revision to the schedule and software requirements specifications shall be considered.

6.6.3 Technical reviews

Technical reviews shall be undertaken for software items from a technical viewpoint, and these reviews shall clarify the risks involved with the implementation of software and computer system that meet requirements specifications and standards.

This activity consists of the following tasks:

6.6.3.1 Review

- (1) The following shall be implemented for review preparation:

- (a) Reviewers shall be selected for review implementation. Reviewers shall include appropriate personnel, consisting not only of parties to the work but also personnel who have enough sufficient knowledge of the area under review, as well as project parties which include interface designers and the originator of the requirements.
- (b) The review subjects and purposes shall be clarified, and materials for a review shall be completed in advance.
- (c) The review purposes, subjects and reviewers shall be clarified, and documented.
- (2) Software items shall be evaluated in a step-by-step approach as developments progress.
- (3) Technical actions regarding software shall be evaluated.
- (4) Review reports shall be developed after the review is completed. Also, quantitative data, such as the evaluation time, and the number of questions and comments at the review, shall be recorded, and a quality evaluation for the review shall be implemented.

6.6.3.2 Walk-through

A walk-through, including a peer review and so on, is an action to improve quality by detecting and removing errors early in the design and development. A walk-through shall be implemented, and performed mainly by the persons in charge, as necessary:

- (1) As preparation for the walk-through, the reviewed party shall provide the relevant documents and codes, including those still in development.
- (2) Parties necessary for walk-through (such as persons in charge of higher level process, persons in charge of lower process, and so on) shall check the documents and codes.
- (3) Questions and problems found through walk-through shall be recorded and followed up until the resolution is completed, and mutually agreed.

6.7 Assessment process

The assessment process is a process for checking the process implementation status and identifying the items to be improved.

This process consists of the following activities:

- (1) Process implementation
- (2) Assessment implementation

6.7.1 Process implementation

The following directions for project designated personnel who are responsible for performing an assessment (hereafter, referred to as the "sponsor"), shall meet the following tasks and shall be planned and agreed to with sponsors:

- (1) It shall be evaluated whether the software development process to be implemented meets the requirements of this standard. The strengths and the weaknesses of the process shall be identified.
- (2) Personnel who are well informed of assessment methods and relevant standards shall be selected as assessors.
- (3) Assessment results, including improvement offers on items needing improvement, are reported in documents to sponsors.

6.7.2 Assessment implementation

Based on the assessment plan, the assessment shall be implemented.

6.8 Problem resolution process

The problem resolution process consists of the following activities:

- (1) Process implementation
- (2) Problem resolution
- (3) Prevention
- (4) Problem trend analysis

6.8.1 Process implementation

The following shall be determined in advance for preparation of incidents and problems (notifications) occurrence regarding software products and services:

- (1) Software products for management
- (2) Management period
- (3) Set of procedures for resolution (action)

If an incident or a problem has occurred, it shall be recorded and managed in accordance with the list above.

6.8.2 Problem resolution

This activity consists of the following tasks.

6.8.2.1 Incident and problem identification

If an incident has occurred, it shall be identified in order to be managed and linked to the known incidents or problems. When a corrective action is required, it shall be managed as a problem. It is recommended that it shall be prioritized for resolution.

Relevant parties shall be informed about the occurrence and status for both incidents and problems.

6.8.2.2 Incident and problem investigation and analysis

By doing the following activities, the incident phenomenon and occurrence condition shall be investigated. The root cause of a problem shall be cleared by investigation and analysis. Problem resolution shall be requested to the other organizations, as necessary.

- (1) Record, analyze and classify incidents. Any incident that needs to take corrective action shall be identified as a problem.
- (2) Record, analyze and classify the identified problems. Root causes shall be cleared by investigations.
- (3) Analyze trends in incidents and problems.
- (4) Inform of the status of incidents and problems.

6.8.2.3 Consideration for problem resolution

The following methods of problem resolution shall be considered:

- (1) Ways of avoiding modification to, or of decreasing the impact on, the software product shall be considered.
- (2) Determination of whether the software product requires a change shall be considered.

In addition, it is desirable that multiple options for resolution be prepared, including solutions which do not modify the software. If software modifications are required, the options must be evaluated by taking into consideration the points of view, such as the cost, the schedule, the overall risk, and the extent of the impact.

6.8.2.4 Determination of problem resolution plan

Plans shall be documented and agreed with the parties:

- (1) As a temporary measure, if there is a way of mitigating the problem, plans shall be documented and agreed with parties.
- (2) With respect to a permanent solution, the problem resolution methods shall be documented and agreed with parties.

6.8.2.5 Problem resolution implementation

In accordance with the agreed plans, problem resolution shall be implemented, and users shall be notified. Track problems to closure.

6.8.2.6 Recording and monitoring

A set of incident or problem resolution actions and the status shall be recorded, monitored, and managed. In principle, after the step has been implemented, monitoring shall continue until it is determined that the problems are resolved, and no new problem has occurred.

6.8.3 Prevention

Improvements for processes and software products shall be identified to prevent the occurrence of known incidents and problems (relapse prevention).

Preventive measures shall be implemented if the risks that need to be prevented can be assessed.

6.8.4 Problem trend analysis

Incident and problem trend analysis shall ideally be performed.

Appendix I
Appendix I Example of the overall configuration of the software development process (Water fall type)

Below is an example of the overall configuration of the software development process

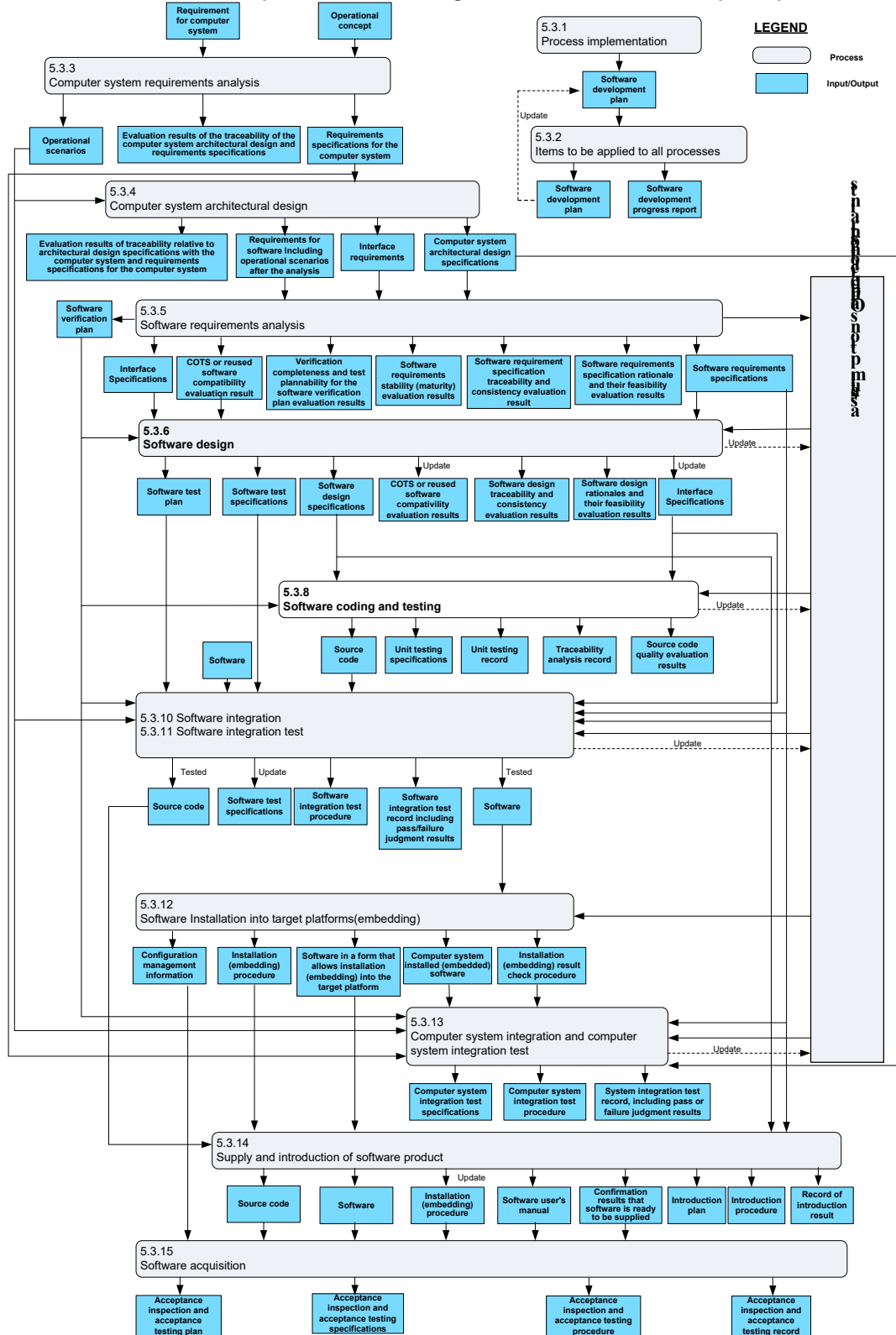


Figure I-1 Example of the overall configuration of the software development process

Appendix II Matrix of input, output and processes

Below is the correspondence between inputs and outputs for each process in the software development process
 Table II-1 Matrix of input, output and processes

	Process name														
	5.3.1	5.3.2	5.3.3	5.3.4	5.3.5	5.3.6	5.3.7	5.3.8	5.3.9	5.3.10	5.3.11	5.3.12	5.3.13	5.3.14	5.3.15
Input / Output															
Software development plan	O	M													
Software development progress report	O														
Requirements for computer system			I												
Operational concept			I												
Operational scenarios		O	I								I		I		
Requirements specifications for computer system		O	I										I		
Evaluation results of the traceability of the computer system architectural design and requirements specifications			O												
Computer system architectural design specifications				O	I										
Requirements for the software including operational scenarios after the analysis				O	I										
Interface requirements				O	I										
Evaluation results of traceability relative to architectural design specifications with the computer system and requirements specifications for the computer system				O											
Software requirement specifications					O	I					I		I	I	
Interface specifications					O	M		I			I				
Operational assumptions and constraints					O	M		M			M	I	M		
COTS or reused software applicability evaluation result					O	M									
Software verification plan including validation plan					O	I		I			I		I		
Verification completeness and test plannability for the software verification plan evaluation results					O										
Software requirements stability (maturity) evaluation results					O										
Software requirements specification traceability and consistency evaluation results					O										
Software requirements specification rationale and feasibility evaluation results					O										
Software architectural or detailed design specifications						O		I			I			I	
Software design traceability/consistency evaluation results						O									
Software design rationales and their feasibility evaluation results						O									
Software test plan						O					I				
Software test specifications						O					M				
Unit testing specifications								O							
Unit testing record								O							
Recording unit testing								O							
Traceability analysis record								O							
Source code quality evaluation results								O							
Source code								O			M	M		M	I
Software											O	M	I		
Software in a form that allows installation (embedding) into the target platform													O	M	I
Configuration management information													O		I
Installation (embedding) procedure													O		I
Software integration test procedure												O		M	I
Software integration test record including pass/failure judgment result												O			
Computer system installed (embedding) software													O	I	
Installation (embedding) result check procedure													O		
Computer system integration test specifications														O	
Computer system integration test procedure														O	
Computer system integration test record including pass or failure judgment results														O	
Software user's manual															O
Confirmation results that software is ready to be supplied															O
Record of confirmation that software is ready to be supplied															I
Introduction plan															O
Installation (embedding) procedure															O
Record of introduction results															O
Acceptance inspection and testing plan															O
Acceptance inspection and testing specifications															O
Acceptance inspection and testing procedure															O
Acceptance inspection and testing record															O

Legend: I->Input, O->Output, M->Input and Output(updated) <Typical>

Appendix III "Verification" and "Validation"

As a supplementary, the difference in concept between "Verification" and "Validation" is shown in the below example.

The purpose of "Verification" is the confirmation that the specified requirements, determined in the previous process, have been fulfilled.

The purpose of "Validation" is the confirmation that the requirements for a specific intended use or application have been fulfilled, and that it makes the accumulated difference from the requirements, occurred with the progress of development, smaller.

The explanation of verification and validation:

"Verification" refers to the provision of objective evidence, that specified requirements have been fulfilled.

"Validation" refers to the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled.

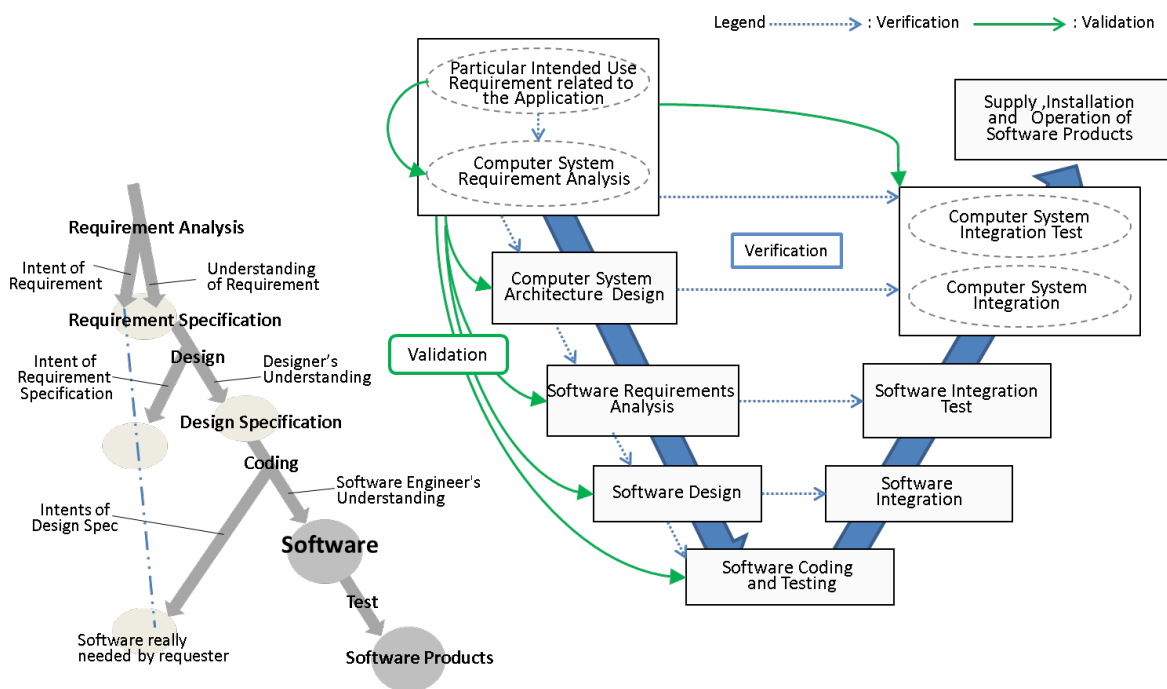


Figure III-1 Concept of verification and validation

Appendix IV Relationship between the problem resolution process and another process

As a supplementary between the relation of the problem resolution process and other processes, in case of incident occurrences during an operation, the relationship between the problem resolution process and the operation process are shown in below.

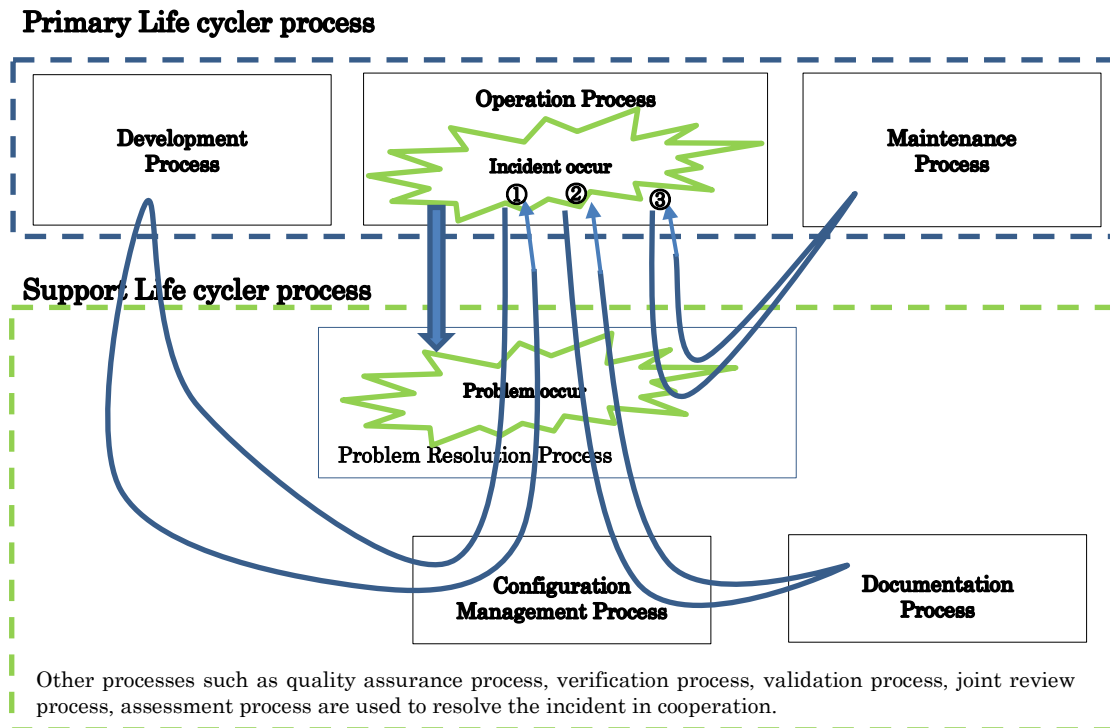


Figure IV-1 Relationship between the problem resolution process and another process

[(1)In case of reworked software]

Incidents occur during operation. → Sent to the problem resolution process. → As a result of the examination, a problem is identified, and the root cause and the countermeasure (modification of the software) is determined. → Take out software from the configuration management process. → Rework software in the development process. → Return software to the configuration management process. → Verify the results in the problem resolution process. → Return to the operation process.

[(2)In case of a document revision]

Incidents occur during operation. → Sent to the problem resolution process. → As a result of the examination, a problem is identified → As a result of the examination, the root cause and the countermeasure is determined. → Manage in the configuration process (judge whether under configuration) → Fix in the documentation process. → manage (update) in the configuration process (in case under configuration) → Verify the results in the problem resolution process. → Return to the operation process.

[(3)In case of countermeasure by operation]

Incidents occur during operation → Sent to the problem resolution process. → As a result of the examination, a problem is determined, and the root cause and a countermeasure is determined without modifying the software in order to support the operation. → Implement countermeasures operation in the maintenance process. → Check the results in the problem resolution process. → Return to the operation process.

Additionally, below is a conceptual example between the difference of incident and problem in the software development standard.

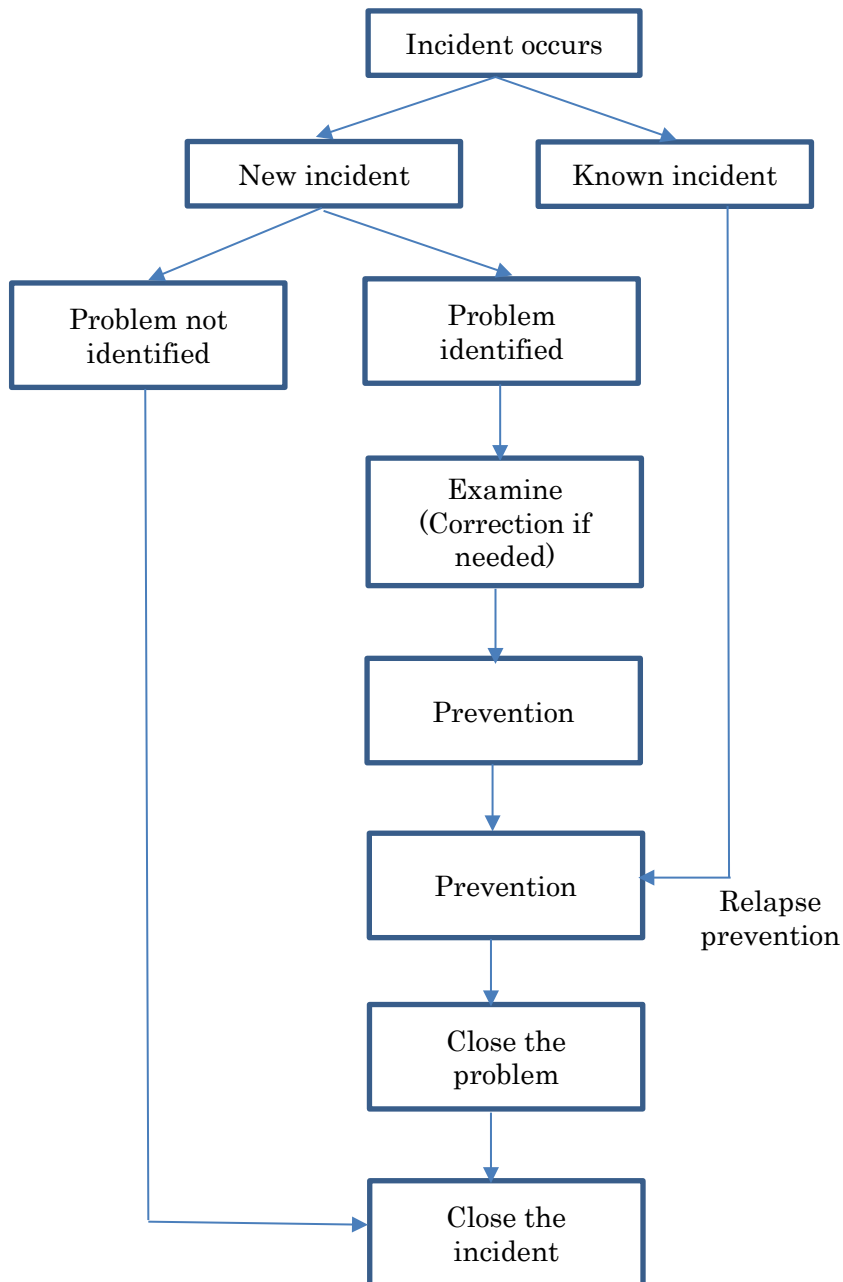


Figure IV-2 Procedure overview of incident and problem in the software development standard

Appendix V Four maintenance types

As a supplementary of four maintenance types, the following is a four maintenance types relation example using a ground system.

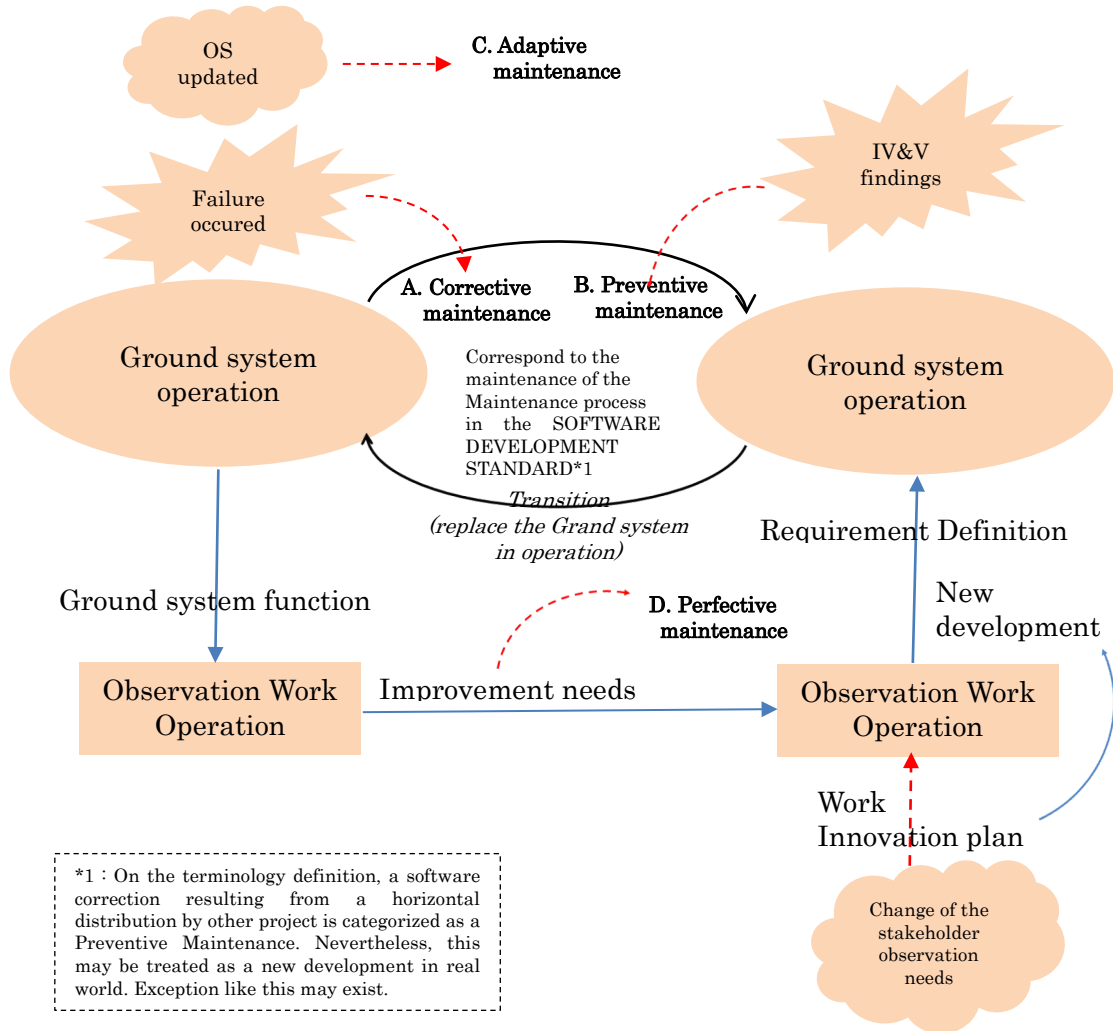


Figure V A relationship of four maintenance types, Ground system and Observation work(Example)

Correction

[A. Corrective maintenance]

During operation, unintended event (failure) happens in the ground system. -> The ground system developer corrects the software and installs it to the operation environment.

[B. Preventive maintenance]

After the transition to the operation, potential bugs are found by performing IV&Vs for the ground system software. (Any failure caused by those bugs has not been happened.) -> The ground system developer corrects the software for the correction and installs them to the operation environment.

Improvement

[C. Adaptive maintenance]

The ground system developer regularly monitors the upgrade status of the operating system. -> The

operating system is upgraded. -> The ground system developer corrects the software for the correction and installs them to the operation environment because skipping of the corresponding activities to any operating system update leads out of date and increase of security risks.

[D. Perfective maintenance]

The design of the Observation work is modified based on the improvement needs of its operation shown by the tracking Control Team. To implement the before mentioned design result of the Observation work, the ground system developer corrects the software for the processing time improvement, additional functionalities and so on, and installs them to the operation environment.