Proposal of "Ask Why" Framework to Analyze Defect Root Causes

Takeshi Mori
NEC Corporation
Tokyo, Japan
ta-mori@ab.jp.nec.com

Ryou Kurashita
NEC Corporation
Tokyo, Japan
r-kura@aj.jp.nec.com

Naomi Honda
NEC Corporation
Tokyo, Japan
n-honda@ay.jp.nec.com

1. Abstract

In the department which the authors belong to, when there's the defect occurred at the customer field, we apply the "ask why" method to analyze defect root causes and find a way to improve product quality. However, it's said only highly skilled engineers can analyze defects. In this paper, we would like to introduce "ask why" framework which anyone can apply to defect root cause analysis.

2. Analysis of Current Situation

The organization to which the authors belong has attained a CMMI level of 5. We defined the processes, and then specified the purposes of each process. Our products were developed based on a V-shaped model (Figure 1).

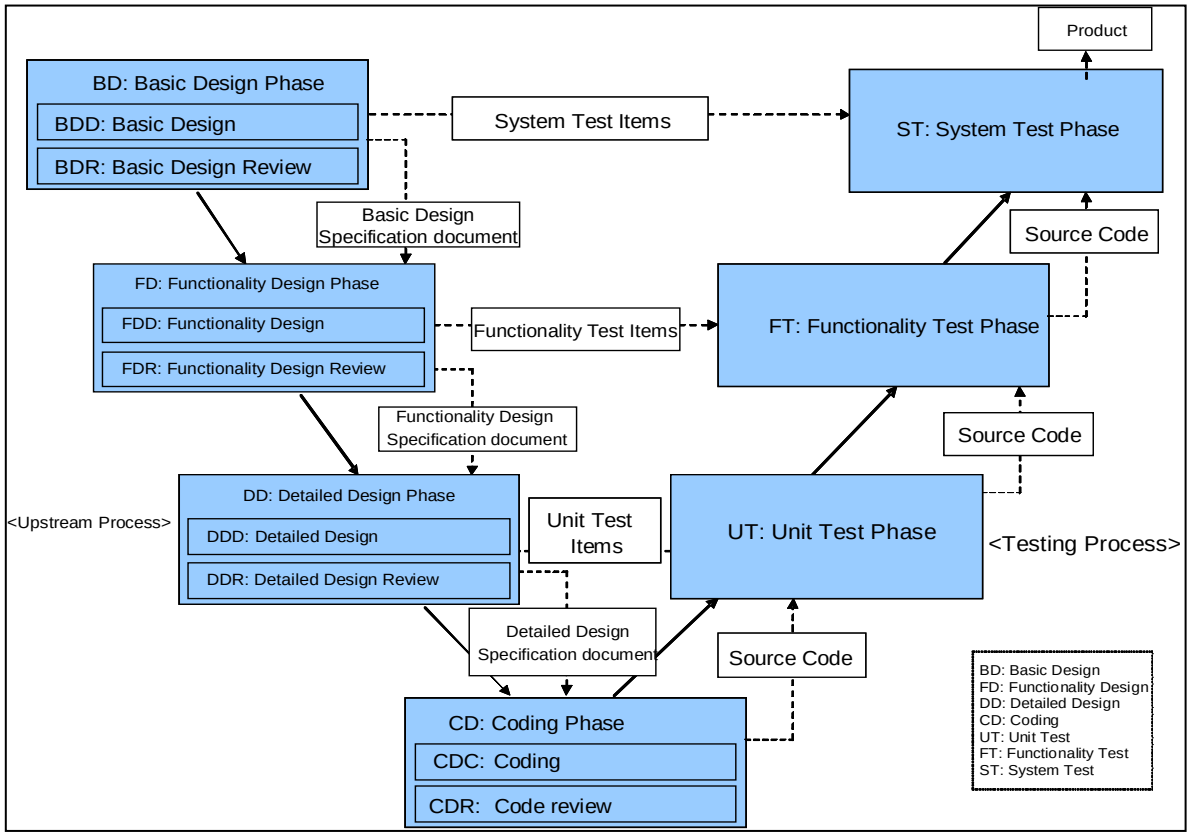

Figure 1. Development model [4]

On the other hand, we apply our "ask why" analysis method to analyze the root cause of defect generated in the field, with the objective of implementing preventive maintenance measures (hereafter, called [1+n procedure] [4]) to detect any remaining same-type defects in released products (Figure 2). In this paper, the method that makes it possible to detect the defect root cause by repeatedly asking why is called the "ask why" analysis method. To improve the quality of general-purpose products with 1+n procedure makes it possible to use the same solution to other products for several users. This greatly contributes to preventive maintenance.

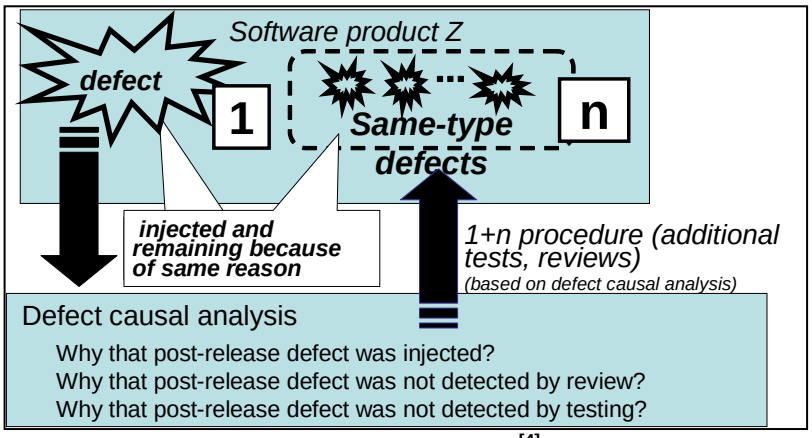

Figure 2. 1+n procedure [4]

To accomplish this goal, it is important to accurately identify the root cause. The "ask why" analysis, which assumes the development model shown in Figure 1, becomes fully effective in identifying the cause upon the occurrence of a defect. Due to the fact that the use of "ask why" method has not sunk into the department which authors belong to, however, the following problems were identified.

- The same concept may be expressed more than one way. This does not provide for further "ask why" analysis.
- The step causing the defects exists upstream, but the defect is treated as a problem of coding phase.
- The results of the analysis do not lead to appropriate countermeasures.
- The scope of the countermeasures can be narrowed down, but why the purpose of narrowing down is necessary is unclear.

In the case of insufficient analysis of the root cause, the success rate of the 1+n procedure (the percentage of remaining defects detected in released products) is low. In some typical cases, only a superficial phenomenon analysis of the causes is performed, and the causes are analyzed with practicable measures in mind. In these cases, viewpoints for the 1+n procedure or the scope of the implementation of the 1+n procedure are misleading, resulting in a reduced possibility of detecting defects that arise from any given cause.

Under these circumstances, it can be understood that the tips for "ask why" analysis, and the causes of the occurrence of defects in the field, can be classified into some representative defects. Assuming that the purpose of the "ask why" analysis is to implement 1+n procedure, it was found that a standard sequence of analyses could be defined as a flow, and be classified into nine representative patterns covering approximately 75% of the analyses. Through this experience, it will be possible to construct a "ask why" analysis framework that enables anyone to easily analyze the root causes.

3. Proposed "Ask Why" Analysis Framework

3.1 "Ask Why" Analysis Framework

A framework, in which the root causes of field defects are analyzed to derive the 1+n procedure, is called "ask why" analysis framework (Figure 3).

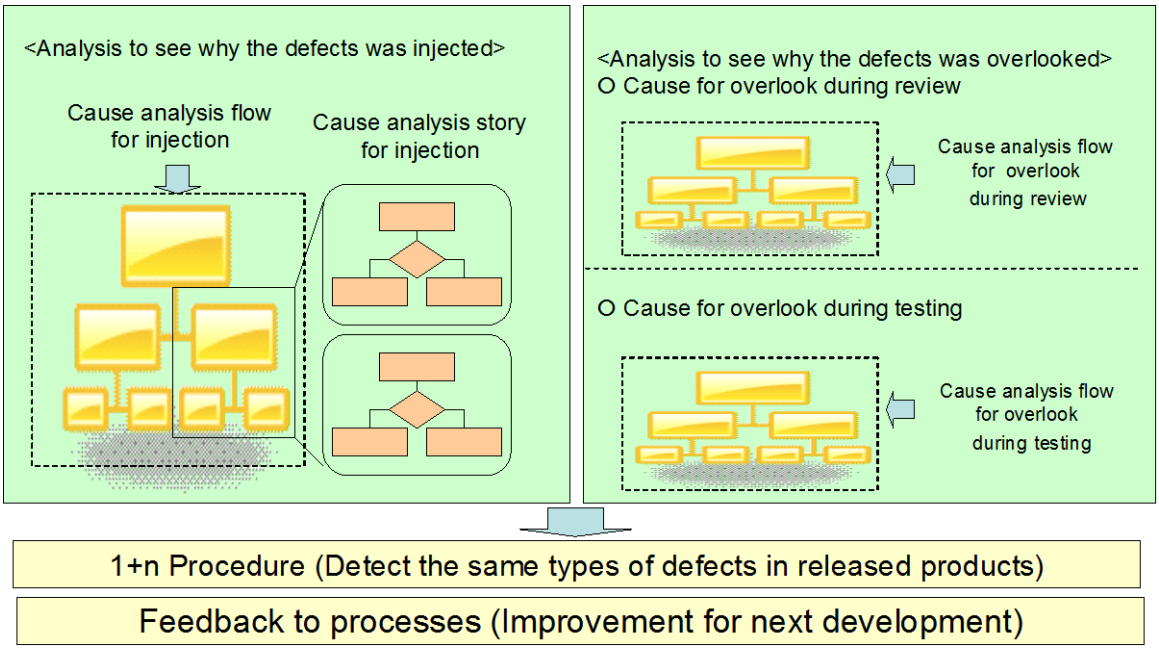

Figure 3. "Ask why" analysis framework

When the purpose is to implement 1+n procedure, the basic concept for the implementation of "ask why" analysis is that the following two viewpoints are independently treated for analysis of these causes.

1) Why a defect was injected? = <cause analysis for injection>

2) Why a defect was not detected by review or testing? = <cause analysis for overlook>

The proposed "ask why" analysis framework employs the concepts of " 'ask why' analysis flow" and "defect cause analysis story" which define a standard sequence of analyses.

"Ask why" analysis framework:

The "ask why" analysis framework shows the flows in analyzing the different causes for the injection of any defect, and for non-detection despite review and testing. These analysis flows are classified into three types by definition: analysis flow to see why the defect was injected, analysis flow to see why the defect was overlooked during review, and analysis flow to see why the defect was overlooked during testing.

Cause analysis story for injection :

Due to the diversification of the types of root causes of defects that are injected, it is difficult to easily find the root causes through the "ask why" analysis flow alone. Consequently, the processes subsequent to the cause analysis flow for injection processes being identified are categorized per installation cause type (e.g., out-of-bound access error, I/F error, etc.). Next, the different analysis flows are integrated into a single story. These integrated flows are called the "cause analysis story for injection."

The cause analysis story for injection classifies/organizes common root cause analysis flows based on actual analysis results, and also defines the 1+n procedure derived from the results of cause analysis.

For reference, we regard it as being unnecessary to form a story for cause analysis of overlook, due to the fact that there are few factors that depend on the types of the defects and the results of the cause analysis for injection can be utilized.

3.2 Cause analysis from the Standpoint of injection

This section describes the cause analysis flow and story for injection, as the cause analysis from the standpoint of defect injection.

(1) Cause analysis flow for injection

The outline of the cause analysis flow for injection is shown in Figure 4.

The most important point for the implementation of a cause analysis, from the standpoint of defect occurrence, is to define defect symptoms, the direct program operations that cause them (hereafter, called installation causes), so as not to misidentify the starting point of the analysis. Based on the results of the analysis, it is important to determine the process in which the defect is generated. Figure 4 is detailed as follows.

<1> Describe, in a short sentence, the direct program operation that causes the defect, in an affirmative way, such as “A did B”, in order to clarify the main points of the occurrence of the problem. Clarify the installation causes that trigger the above direct program operation, using terms such as “error” and “lacked consideration”. If the staff members are aware of these terms, they will consciously think about what should be done to the installation to prevent the generation of the defect.

<2> After defining the installation causes, determine the process that generates the defect.

<2>-1 First, determine whether a part to which the element that becomes the installation cause is to be designed is located inside or outside the module (e.g., when the defect results from a buffer overflow, determine whether the buffer should be designed inside or outside the module). If it is judged that the buffer should be outside, the process causing the defect can be determined to be an FD phase.

<2>-2 Assuming that it is judged that the causal element should be designed inside the module, if the cause corresponds to a violation of the coding standards or common coding conventions, like a collection of “don’ts”, the process causing the defect can be determined to be a CD phase.

<2>-3 In cases other than the above example, judge the defect-causing process based on whether the causal element is designed inside the module and whether this design method is clearly included in the design specification. If the design is fully described in the specification, the process shall be determined to be the CD phase, assuming that the defect is generated during coding in spite of the full implementation of the internal design. If the design is not conducted or poorly implemented, the defect-causing process shall be determined to a DD phase.

<2>-4 If the defect-causing process is determined to be the CD phase, the management or basic technical measures, such as the re-education of coding standards, the verification of coding under the coding standards, and re-education about coding input documents, will be effective. On the other hand, if the defect arises from the internal or external design, the types of the root causes will be diversified, depending on the details of the defect. For this reason, the “cause analysis story for injection” described in (2) shall be used.

However, this analysis framework excludes requirement issues. If the defect-causing process lies in any step prior to the basic design, another approach must be taken to track down the root cause.

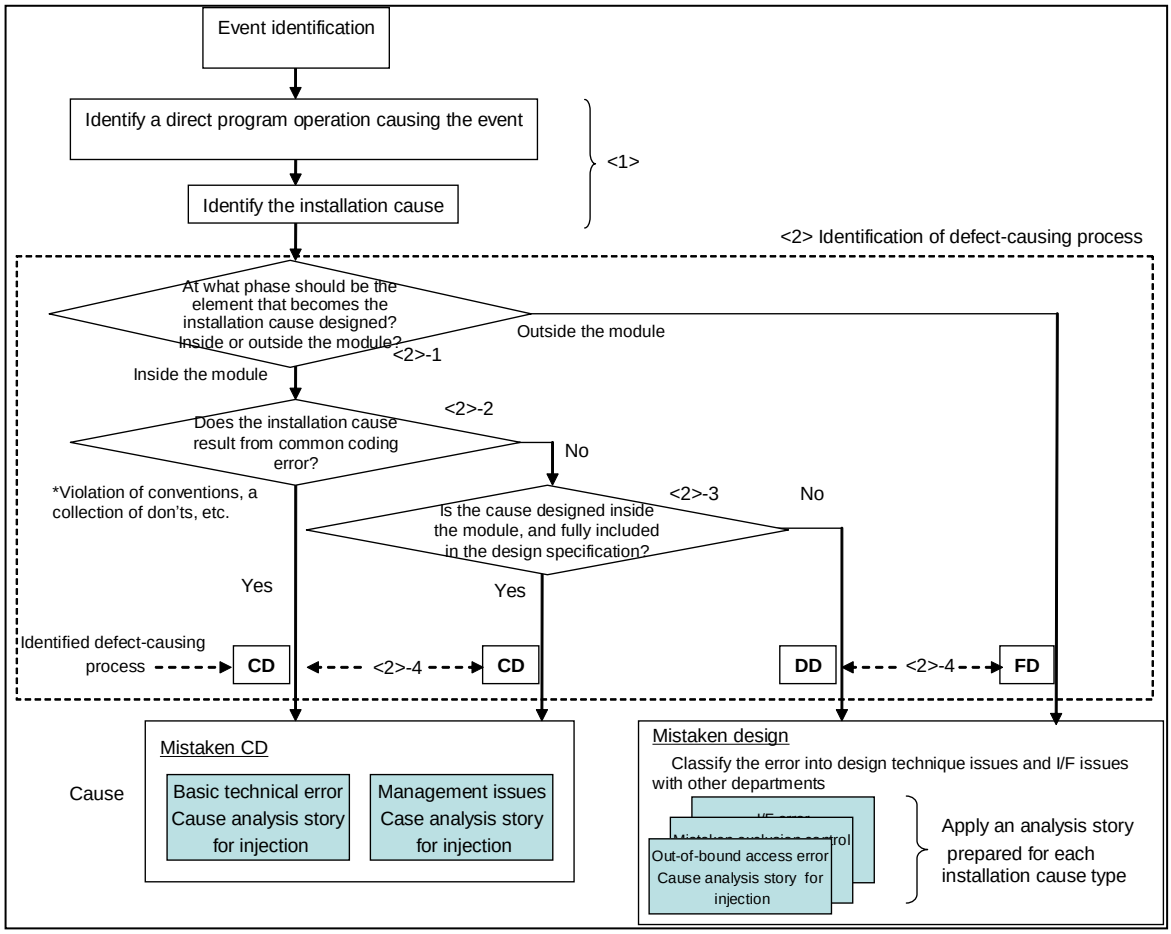

Figure 4. Cause analysis flow for injection

(2) Cause analysis story for injection

Based on the installation causes determined in <1> of Figure 3, the applicable story shall be selected from the prepared cause analysis stories for injection, according to which the analysis is performed. In the absence of a relevant story, a new analysis story is required.

The authors' organization has the following nine types of standard analysis stories. The use of these nine types of stories is known to cover about 75% of the analyses.

Table 1. Analysis stories of the relevant organization

Story name	Description (Installation cause)
Out-of-bound access error	Access to areas beyond buffer space or NULL access
I/F error	Recognition error of argument, return value and function specifications about function I/F
Exclusion control error	Resource exclusion control error in multi-process (thread) environment
Mistaken error processing	Mistaken error processing, omission of error processing
Threshold value/boundary value error	Mistaken processing of threshold value/boundary value
Startup/termination error	Mistaken processing of startup/termination at startup/termination of AP and server
String operation error	Mistaken processing of special character or Japanese character

	codes
Resource release error	Omission of resource release
Mistaken processing of unexpected cases	Mistaken processing of cases where unexpected message/object arrives

As an example of a cause analysis stories for injection, the story for an “out-of-bound access error” is described in Figure 5. After the identification of a defect-causing process, it is determined whether an installation cause is designed in the applicable process, as a starting point. The starting point causes a branch to assumed cases, thereby deriving a root cause and 1+n procedure for each case.

At the present moment, the use of this cause analysis story for injection can cover all field defects for more than 10 “out-of-bound access error” cases which occurred in the relevant organization. However, we think that “assumed cases” are likely to exist that cannot be envisaged now. For these cases, “other” is explicitly placed. This case requires a new root cause analysis and the story will be expanded based on the results of the analysis.

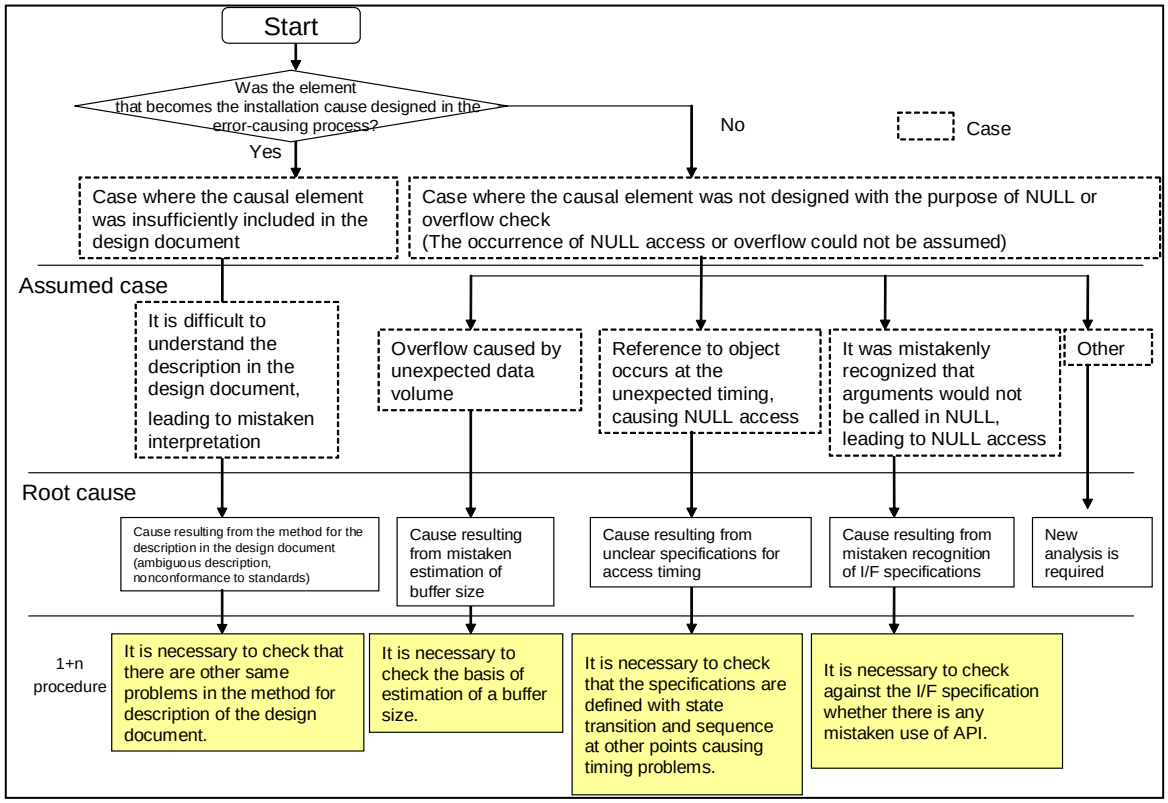

Figure 5. Out-of-bound access error cause analysis story

3.3 Cause Analysis from the Standpoint of overlook

Cause analysis flows from the standpoint of overlook during review and testing are shown in Figure 6 and Figure 7. These flows provide a direction leading to the root causes. The flows can be used to find almost all root causes. For the implementation of the measures, the measures must be detailed in consideration of the determined reason for injection.

A common case is that where a defect is missed because a reviewer is improper (a typical case is that in which the staff of Development group for closely linked components do not participate in the review). For the 1+n procedure, the sufficiency of the reviewer must be verified based on all the review records. The parts missed by the reviewer shall be re-reviewed after assignment of a new reviewer. For the re-review, it is important to conduct a review with a focus on the reason for injection.

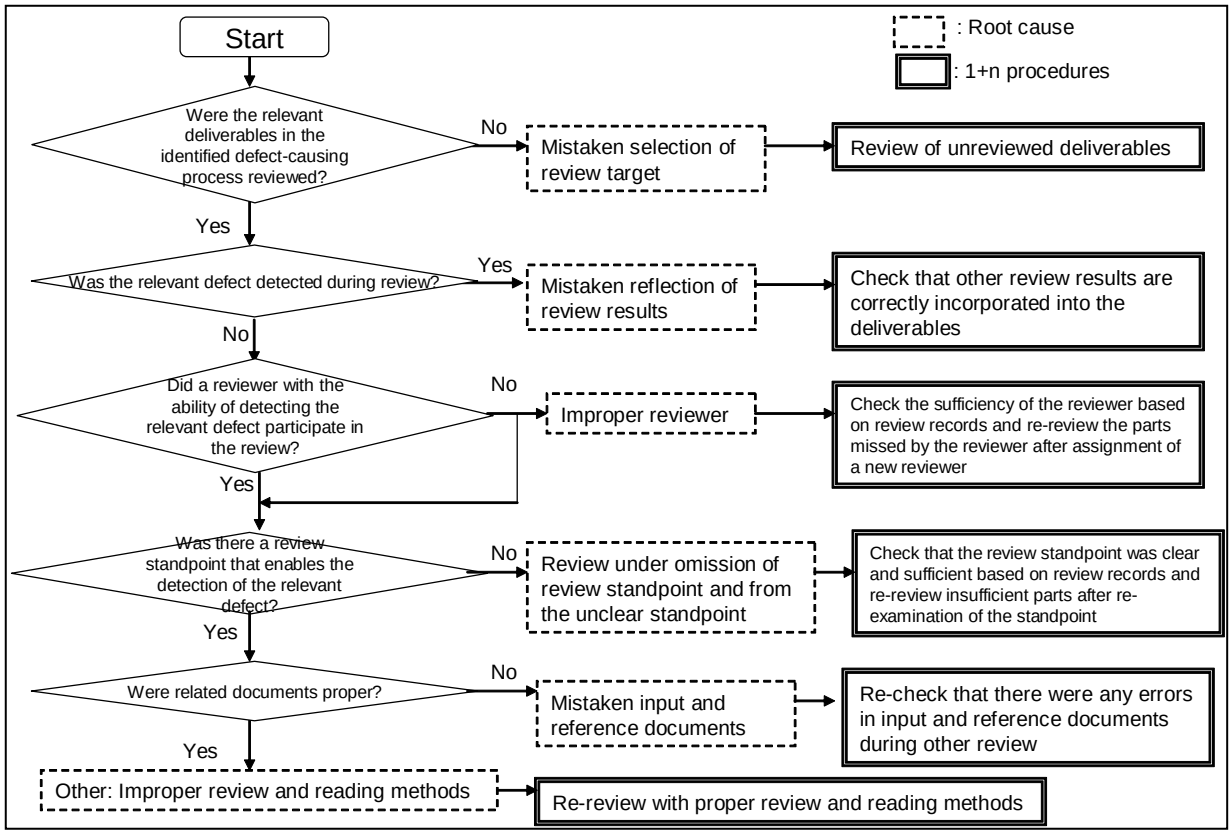

Figure 6. Cause analysis flow for outlook during review

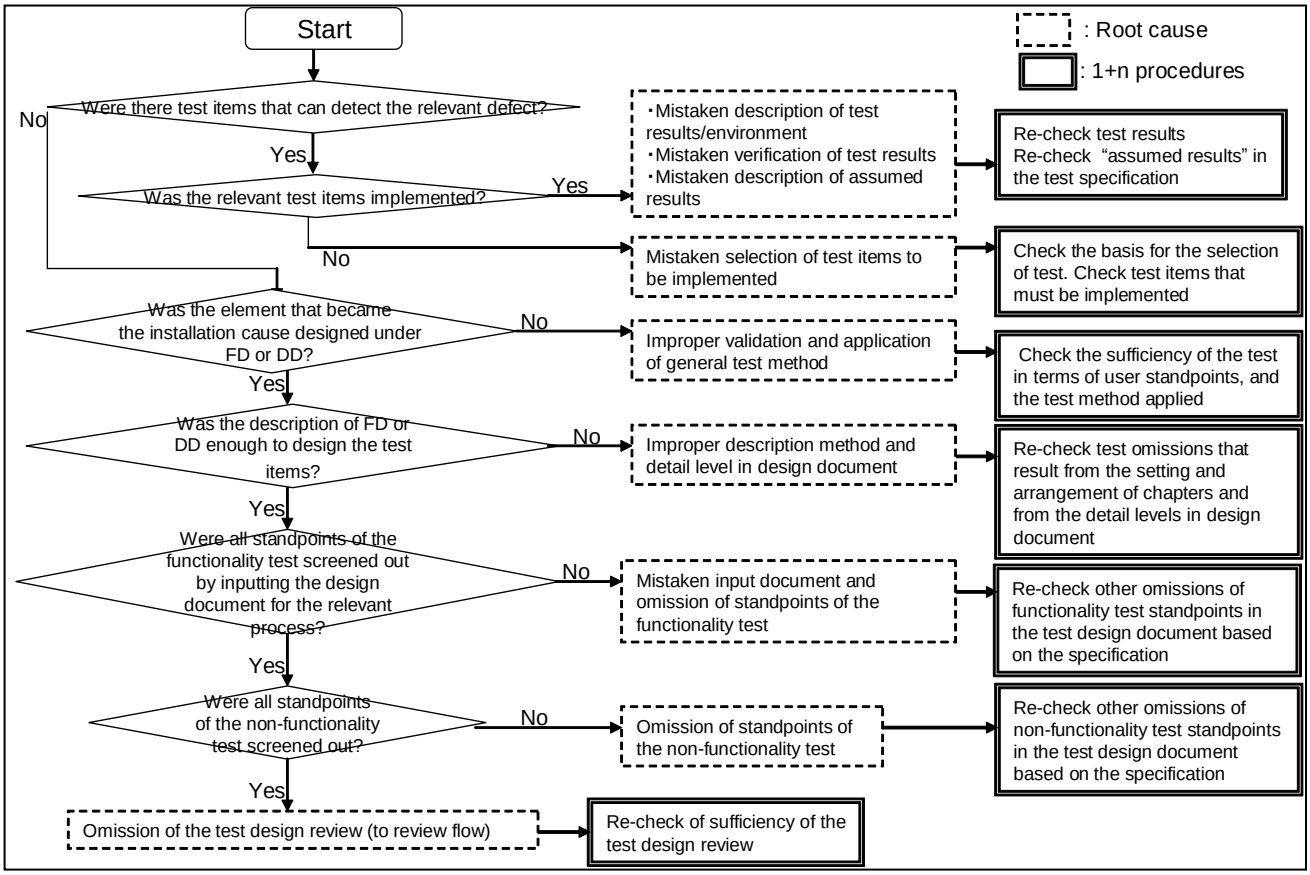

Figure 7. Cause analysis flow for outlook during testing

4. Effectiveness and Discussion

The “ask why” analysis framework presented in this paper was promoted throughout the organization by revising old forms and providing education about the analysis. As a result, a success rate of the 1+n procedure was increased from 17% to 38%. The cases in which only a superficial phenomenon analysis is performed or an analysis based on practicable measures is performed were reduced. On-site staff members’ inputs and opinions included that they deepened their understanding of the analysis standpoints and that in the analysis by an inexperienced person in charge, the initial quality of the analysis seems to be improved. They are gaining the intended effectiveness.

Furthermore, the 1+n procedure were successfully taken only by specific groups before the application of this method, but after the application of the framework, the rate of success of the measures was increased in almost all groups (of the eight groups, the number of groups for which the measures succeeded increased from 2 to 7). All personnel have started to feel the effect of the “ask why” analysis framework as a means of easily finding root causes.

5. Conclusion

It has been said that because of the difficulty to use the “ask why” analysis method only highly-skilled engineers can detect root causes with the method. But in the organization promoting the standardization/establishment of the SW process, it was found that it is possible to break the procedure of “ask why” analysis down into patterns for the purpose of implementation of 1+n procedure. The concepts presented in this paper were applied to other departments that develop the same general-purpose software packages. As a result, it was found that the nine types of cause analysis stories for injection covered in this paper could be used to find about 65% of all field defects. We expect that this framework can be successfully applied by other departments.

We will now accumulate case studies on the “ask why” analysis and review the sufficiency and applicability of the “ask why” analysis framework presented in this paper.

References

[1]Hitoshi Ogura: How to Make Full Use of "Ask Why" Analysis, JIPM Solutions, 1997

[2]Hayakawa, Ishii, et al.: Applying "ask why five times" method on software development, Collection of Papers Presented in Software Quality Symposium 2008, Union of Japanese Scientists and Engineers, pp.185, 2008

[3]Naomi Honda: Beyond CMMMI Level5 – Comparative Analysis of Two CMMI Level5 Organizations-, 4th World Congress for Software Quality-Bethesda, Maryland, USA, 2008

[4]Naomi Honda: Software Quality Accounting System - Quality assurance technology that supports high quality software development at NEC, JUSE Press, Tokyo, 2010.