Risk-based inspection

Shuji Morisaki

May 28, 2008

Graduate School of Information Science Nara Institute of Science and Technology

1. Introduction

The role of software is growing important. Failures in software potentially cause severe result. Testing methods and inspection methods are proposed in order to detect defects.

As software becomes bigger, the numbers of failures grow larger. Efficient methods to detect failures are required. Bach proposed risk-based testing in order to detect more severe failures first in the article [3]. FMEA (Failure Mode Effect Analysis) [2] is one of the methods for detecting severe failure in designing hardware.

This paper proposes risk-based inspection that is a method of inspection that enables to detect more severe failures earlier. Risk-based inspection can be used in all inspection including requirement, architectural design, system design, coding design, testing document, and test cases. Risk-based inspection consists of identification of risk and inspection according to identified risks.

Investigations on cost of removal failures indicate that the earlier failures found and fixed, the less cost of fix[1][4][5]. Risk-based inspection is expected more efficient than risk-based testing or other reading techniques in inspection.

In this article, an example scenario of risk-based inspection is described.

2. Risk-based inspection

Fig. 1 shows process of risk-based inspection. First, possible risks are defined by stakeholders. Second, features, functions, subsystem or other decomposition of software are defined according to risk definition. Third, risks of decomposed parts are identified according to risk definitions and decomposition. At last, inspection is conducted based on risks and decomposed parts.

Risk identification / definition

This process is done in project planning phase. Stakeholders of the project identify and define what is the risk of the software or the development project.

Decomposition of software inspected

When the software or system is large, decomposition of the software or system is required. Appropriate unit of decomposition depends on the software inspected. Unit of decomposition includes function definition, subsystem definition or program events such as batch event and user event.

Identification of risks

To identify risks according to the definition determined in the “Risk definition” process. Identified risks could differ among decomposed units, if necessary. Identified risks depend on the software or project. The risk may include critical subsystem, required quality, complexity and fully new function.

Inspection based on the risks identified

Inspection to reduce the risks identified. The larger risk is expected to be pointed earlier.

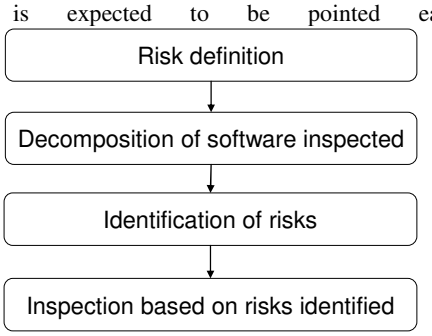

Fig. 1 process of risk-based inspection

† Graduate School of Information. Science, Nara Institute of Science and Technology, Japan

Table 1. Calculation of risk exposure in the example scenario

Subsystem	Probability	Cost (person hour)	Risk exposure
Database	0.02	80	1.6
Registration	0.20	40	8
Listing	0.12	20	2.4

3. An Example Scenario

In this section, an example scenario is described in order to clarify risk-based inspection. A web registration system for event management is described in this section as an example scenario. The system collect attendee information including name, email address and phone number. The system also provide list of registered attendees.

Risk definition

In discussion among system development project members, biggest risks are lost of attendee list in operation phase and development cost grows high.

Decomposition of software inspected

The example system consists of registration subsystem, database subsystem and listing subsystem.

Identification of risk indicator

In order to prioritize inspection at the point view, risk indicators are identified for each part. Risk exposure calculation is one of the methods for risk indicator identification. Table 1 shows an example of risk matrix of subsystem, probability, cost, and risk exposure. Each subsystem in Table 1 is identified in the procedure "Decomposition of software inspected". Probability and cost is of example of risk exposure calculation.

Inspection based on the risks identified

Software is inspected according to risk indicator. Subsystem having higher risk indicator is assigned more inspection resource. In this example, Registration subsystem is prior to the other subsystems.

4. Conclusion

This paper proposes risk-based inspection. Risk -based inspection enables inspection resource to be assigned efficiently. Since risks or risk indicators differ among projects, first stakeholders of project identify and define risks and risk indicators, then inspection is conducted based on the identified risks and risk indicators. Because risks and risk indicators are defined and agreed before inspection, transparent decision making is achieved at inspection.

References

[1] Haapanen Pentti, Helminen Atte, "Failure mode and effects analysis of software-based automation systems", STUK-YTO-TR 190 (2002)

[2] IEC, "IEC 60812: E-Standard - Analysis Techniques for System Reliability", IEC(2006)

[3] James Bach, "A Framework for Good Enough Testing," IEEE Computer, vol. 31, no. 10, pp. 124-126 (1998)

[4] Jianyun Zhou, Tor Stalhane, "Using FMEA for early robustness analysis of Web-based systems", Proceedings of COMPSAC 2004. pp. 28 – 29 (2004)

[5] Qian Feng, Robyn R. Lutz, "Bi-directional safety analysis of product lines", Journal of Systems and Software vol. 78, No. 2, pp. 111-127(2005)