FAST ABSTRCT: A Study on SFMEA Method for UML-Based Software

Zhang Hong
Dept. of System Engineering of Engineering Technology
Beihang University
Beijing, P. R. China, 100191
zh@buaa.edu.cn

Wang Wentao
Quality Department
Southwest Institute of Electronic Equipment
Chengdu, P. R. China, 610036
wwtjun@gmail.com

Abstract

As one of the most important methods to improve software reliability and safety, Software Failure Modes and Effects Analysis (SFMEA) has been increasingly applied to software products in many fields. Meanwhile, the Unified Modeling Language (UML) has become the main design language for Object-Oriented software, with Rational Unified Process(RUP) as the most often used development process model. While traditional methods of SFMEA are not suitable for software developed in UML. In this paper, we present a method of SFMEA applicable to UML-based software, covering the main core workflows in RUP, e.g. the requirement, analysis and design workflows.

1. Introduction

Software Failure Modes and Effects Analysis (SFMEA) is a bottom-up analysis method to identify the consequences of possible software failure modes on software systems. As a major method of software analysis, SFMEA has been playing an important role in improving software reliability and safety.

Meanwhile, the Unified Modeling Language (UML) has become the main design language in the software industry, and the Rational Unified Process (RUP), where UML is usually applied, has been adopted by many corporations. Traditional methods of SFMEA are not suitable for software developed in UML. Therefore, we see a great need for study on SFMEA for UML-based software.

However, research dedicated to this field still remains very few and incomprehensive. In 2004 Nathaniel Ozarin[1] discussed in his paper the relationships among software development stages, UML diagrams and FMEA levels (method, class, module and package), but didn't focus on the concrete process to perform SFMEA with UML diagrams. In the same year, Herbert Hecht[2] described in his paper the application of UML to SFMEA during the concept phase and the implementation phase respectively. But he used the use case diagrams and class diagrams only, without using dynamic diagrams such as activity diagrams and sequence diagrams, etc.

In this paper, a method to perform SFMEA for UML-based software is presented, covering the main core workflows in RUP.

2. Approach

In order to find software defects and weak links as early as possible, SFMEA should be performed primarily in the inception phase and elaboration phase of RUP, instead of the construction phase and transition phase. The basic process of SFMEA in RUP for UML-based software is not different from traditional method, but the characteristic of RUP determines that the analysis results of SFMEA will be refined and extended with the incremental and iterative development process.

In each RUP phase, there are five core workflows, which could be used alternately during each iterating development process. We focused our study on three core workflows, i.e. the requirement, analysis and design workflows. Therefore, the SFMEA method for UML-based software is discussed basing on the characteristics of UML models used in these three core workflows in RUP respectively.

We analyzed the links between SFMEA elements and UML models in each core workflow. Upon this, the analysis basis for SFMEA in each core workflow is identified, as shown in table 1.

Table 1 Links between SFMEA and UML

RUP workflow	UML models as basis for SFMEA
Requirement workflow	Use case diagram
Analysis workflow	Class diagram Collaboration diagram Sequence diagram Activity diagram
Design workflow	Class diagram Sequence diagram

Then, the analysis methods in each of these three core workflows in RUP are described respectively as follows.

In the requirement workflow of RUP, software requirements are described primarily with use case diagrams. Thus SFMEA in this workflow can be implemented according to the use case diagrams. The analysis level can be determined with use case diagrams, and use cases can be considered as analysis items. For each analysis item, failure modes can be identified basing on its function and performance requirements. Failure effects are traced basing on the interrelationship among use cases, and sometimes activity diagrams can also be used to trace the failure effects.

In the analysis workflow of RUP, activity diagrams, sequence diagrams and class diagrams are mostly often used to describe the structure and functions of a system. Therefore, there are two ways to perform SFMEA in this workflow.

One way is based on activity diagrams and use case diagrams, in which activity diagrams of a use case are used to identify analysis level, and activities are considered as analysis items. Failure effects on the system can be traced from activity level to use case level and use case diagram level successively.

The other way is based upon use case diagrams and sequence diagrams, where sequence diagrams can be used to identify analysis level, and messages can be considered as analysis items. Failure effects on the system can be traced from message level to use case level and use case diagram level successively.

In the design workflow of RUP, system model is extended and refined to describe class attributes, methods and relationship between classes on the basis of class diagrams. And sequence diagrams are often used to describe the collaboration among objects and message calling process. SFMEA in this workflow is based upon use case diagrams, sequence diagrams and class diagrams, where analysis level is identified with class diagrams, which are more refined than in analysis workflow, and messages are considered as analysis items. Failure effects can be traced from message level to use case level and use case diagram level successively.

Finally, we use a case study to illustrate the effectiveness of SFMEA method for UML-based software presented in this paper. The case study shows that several software defects have been found through SFMEA during the requirement, analysis and design workflows.

3. Conclusions and Future Work

We proposed a SFMEA method for UML-based software in the main core workflows of RUP, i.e. the requirement, analysis and design workflows. In each core workflow, the methods to identify the analysis levels and analysis items, and to trace the failure effects are discussed.

As future work, we shall continue our research to include other UML diagrams in the analysis to obtain more precise and complete analysis results. This should also lead to further improvements in the analysis method. Furthermore, detailed SFMEA for UML-based software should also be studied, which could be used to verify the effectiveness of system SFMEA and expose more defects in software products. Our ultimate goal is to establish a set of comprehensive SFMEA approaches applicable to the UML-based software in the whole development process of RUP.

4. References

[1] Nathaniel Ozarin. Failure Modes and Effects Analysis during Design of Computer Software [J]. *The annual Reliability and Maintainability symposium*. IEEE, 2004: 201-206

[2] Herbert Hecht, Xuegao An, Myron Hecht. Computer Aided Software FMEA for Unified Modeling Language Based Software [J]. *The annual Reliability and Maintainability symposium*. IEEE, 2004: 243-248

[3] Bowles, J.B., The new SAE FMECA standard. *The Annual Reliability and Maintainability Symposium* [C]. IEEE, 1998: 48–53.

[4] Haapanen Pentti, Helminen Atte, Failure modes and effects analysis of software based automation systems [C], *STUK-YTO-TR* 190, 2002.8

[5] Peter L.Goddard. Software FMEA Techniques [J]. *Reliability and Maintainability Symposium*. Jan.2000: 118-123

[6] Lutz, R.R., Woodhouse, R.M. Contributions of SFMEA to Requirements Analysis [J]. *Proceedings of the Second International Conference*. April 1996: 44 – 51