Software Interface Failure Modes and Effect Analysis Based on UML

Chenguang Hou, Qihua Wang
China Aero-Polytechnology Establishment
Beijing,China,100028
houcg1982@hotmail.com

***Abstract*—Software is widely used in equipment control, signal processing, communication and other fields. In today's world, we should pay more attention to its quality needs. Interface is one weak-link in large-scale software design. In order to improve software quality, we need to enhance interface quality work, including management, analysis, design, test methods etc. FMEA (Failure Modes and Effect Analysis) is a common quality analysis tool which has been used in software developing progress, but the progress is ineffective. Unified Modeling Language (UML) is a standardized modeling language. It provides a unified tool for software developers of different domains. Even if quality engineers do not involve in software design, they can understand software functions and design it by UML files. In this paper, we apply FMEA to software interface quality analysis, analyze the problems in the previous SFMEA application, investigate the possibility of SIFMEA based on UML（Unified Modeling Language ）, discuss the feasibility of using UML diagrams and introduce the approach and steps of SIFMEA utilizing different UML diagrams. At last, the method's effectiveness is proved by an flight control system example.**

Key Words—software interface; FMEA; UML

I. FOREWORD

In the last twenty years, on-board electronic equipments have experienced different stages from hardware-intensive to software-intensive. The avionics software of F-22, which was developed in 1980's, has more than 2000KLOC codes. And JSF, which was developed since 1990's, has more than 6000KLOC codes. It's obvious that the proportion of software in on-board equipments have risen quickly. Large-scale software development is a complex systematic engineering. It's necessary to decompose the top-level requirements to contractors, teams and engineers. After all the modules were developed, they were assembled together. By the connection of interfaces, software modules build up an integral system. However, different contractors may have distinctive develop modes, develop engineers and management ways, the quality of software modules is different with each other. So interfaces become the software's weak link [1] , we often found many bugs during integration testing.

The reasons of above phenomena include: 1) contractors were not aware of the significance of software interface quality, they were absorbed in schedule and cost, but didn't realize that quality also can help them save time and money; 2) lack of efficacious software quality analyse technology, now usable software quality analyse technology can't be combined with software design process.

It's well known that quality is the inherent attribute of product. In order to improve the quality of software, we need to do quality work as early as possible, utilize design & analysis technology to improve software quality.

In this paper, we use the Unified Modeling Language (UML) in Software Interface Failure Mode and Effect Analyse (SIFMEA). During every phase of software development, carry on SIFMEA utilizing UML design documents. This method helps quality engineers to do SIFMEA at the beginning of requirement analysis, and throughout the whole software develop process.

II. UML & SOFTWARE DEVELOPMENT

Unified Modeling Language (UML) is a standardized

978-0-7695-4522-6/11 $26.00 © 2011 IEEE
DOI 10.1109/ICM.2011.375

modeling language. It was created and managed by OMG (Object Management Group). UML can be used for modeling data, structure, behavior, architecture, etc. By using UML, developers can focus on the product model and structure, no matter what language and algorithm used. After the UML models were established, they can be transformed into codes automatically.

System models are represented by diagrams in UML. A system model has two different views. Static view emphasizes the static structure of the system using objects, attributes, operations and relationships. The structural view includes class diagrams and composite structure diagrams. Dynamic view emphasizes the dynamic behavior of the system by showing collaborations among objects and changes to the internal states of objects. This view includes sequence diagrams, activity diagrams and state machine diagrams. Different UML views establish models from different aspects(Table 1). UML provided a unified tool for software developers from different domains, design, test or integrate. We can use UML views in whole software development process.Even though quality engineers didn't involve in softwere design, they can understand software function and design by UML.

Table1 Point of Different UML Views

View	point of view
Use case diagram	function of system, inner relation, system & user relation, mission scene
Class diagram	relation between classes
State diagram	state of object, the affect of event to state
Sequence diagram	interaction between objects , real-time requirements
Activity diagram	actions and results

III. UML-BASED SIFMEA

FMEA is a quality and reliability analysis techniques widely used in equipment reliability analysis, its effectiveness has been verified in industry. Today, the importance of software quality has earned widespread respect. So FMEA has also been applied to software interface reliability analysis[4]. However, it's still difficult to use this technique to improve software quality. One important reason is that software quality engineers don't familiar with the software product usually, when they get software interface development documentations, they may find they are irregular, so the FMEA work can't be done effectively.

UML is a general modeling language, it builds a bridge between software developers and quality engineers. The main advantage of SIFMEA based on UML is that software quality engineers can quickly learn about the design of interface by UML views, so the FMEA work can be carried out effectively. Another advantage is that FMEA can combined with design through software development.

IV. STEPS OF SIFMEA BASED ON UML

A. Choice of software interface needs to analyse

There are many interfaces in software, which one should we choose to analyse? There are three aspects need to consider:

1) The type of interface

Interfaces are different from each other in complexity, so the difficult level of design and implementation are different. Generally speaking, complex interfaces usually have more bugs than others. Expecially some particular interfaces, often have some special requirements in design, need to pay more attention.

2) The important degree of interface

Different interfaces have different fuctions in system, so they have different important degrees. Some interface are used for transporting core data, some only transporting secondary data. The more important the interface is, the greater its impact is once it is failure. So we'd better choose important interface to analysis first.

3) Design experience

When we choose the interface, we can found whether our designers have the design experience, or whether there are similar interfaces showing good quality in last test or use.

B. Determine the analysis grade and assumptive conditions

SIFMEA can be divided into functional FMEA and detail FMEA. We can determine analysis grade by two

aspects. One is the importance of the interface, generic interface can usually do functional FMEA, important interface needs to do detail FMEA. Another is the schedule software develop. For example, during requirements analysis phase, we can develop functional FMEA by using case diagram and primary class diagram, when there are detailed class diagram, activity diagram, sequence diagram and code, we can do detail FMEA.

Input data of on-board software is closely related to its mission conditions. Before FMEA, we should assume the conditions, include mission phase, environmental conditions, etc. If the software has several mission requirements, we should develop FMEA separately.

C. *Find possible failure modes*

Utilize UML views and software development documents, find out possible failure modes of software interfaces. UML views are the descriptions of software interface from different points, so it's necessary to use all of them to find failure modes. During this work, we may found views are related to each other. For example, the failure mode of sequence views may be found in the interface's activity diagram.

D. *Analyse failure reason and effect*

After get the possible failure modes, we need to find out the failure reasons and effects. Failure reasons maybe data, component or environment. Failure effects can be divided into three grades: the output of interface, local, system. When we analyse local and system effect, the views of associated interfaces and system are useful.

E. *Determine the severity of failure*

See GJB/Z 1391-2006.

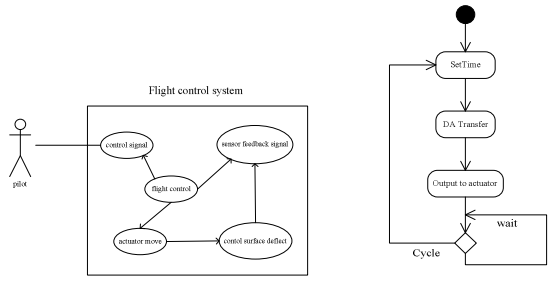
F. *Fill out the FMEA table*

Fill out the FMEA table with analysis results.

V. EXAMPLE

Following is an example of SIFMEA based on UML. Flight control system is a safety-critical system of aircraft. There are many interfaces in flight control system, for example man-machine interface, sensor interface, actuator control interface, etc.[6]

During requirements analysis phase, software developer established use case diagram of flight control system(Figure1(a)). We can found the interface between flight computer and actuator is a critical interface.



(a) (b)

Figure1 Use case and Activity diagram of flight control system

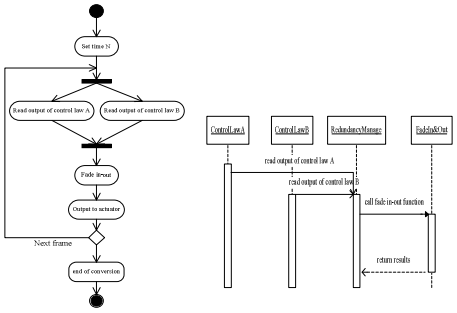
The activity diagram of interface between flight computer and actuator is shown in Figure 1 (b). We can develop function SIFMEA using that two views:

Table 3 Function SIFMEA

Failure modes	Failure reason	mission	Failure effects			Severity	Corrective measures
			Interface effects	Local effect	System effect		
DA no inputs	Sensor failure	flight	No outputs	Control system is open loop	Flying qualities decline	Ⅲ	redundant sensor
	AD module failure	flight	No outputs	Control system is open loop	Flying qualities decline	Ⅲ	DA module back-up
Interface output too high or too low	Parameter drifting	flight	Outputs Deviates the real value	Actuator is not in place	Flying qualities decline	Ⅱ	Calibrate ouputs
Outputs not smooth	Environment disturbance	flight	Outputs not continual have burrs	Actuator is unstable	Flying qualities decline	Ⅰ	Add filter and smoother

During different mission phases, aircraft needs different control laws. So flight control software should switch between different control modes. If the switch progress was performed instantaneous, that may cause control surface tremble. In order to improve flying qualities, it's necessary to increase a fade in-out step, and then the aircraft can switch smoothly from one mode to another.

During detailed design phase, developers established activity diagram and sequence diagram of fade in-out step(Chart 2).



(a) (b)

Figure 2 activity diagram and sequence diagram

We can develop detail SIFMEA and fill FMEA table (Table 4).

Table 4 Detail SIFMEA

Variable	Failure modes	mission	Failure effects			Severity	Corrective measures
			Interface effects	Local effect	System effect		
CA/CB	Too high or too low	flight	Output too high or too low	Deflection too big or too small	Flying qualities decline	Ⅲ	Add limiter
	No inputs	flight	No inputs	Actuator don't Action	Plane out of control	Ⅳ	Add Monitor
N	Too big	flight	Outputs change too fast	Actuator flutter	Flying qualities decline	Ⅱ	use reasonable N
	Too small	flight	Outputs change too slow	Can meet the control requirements	Flying qualities decline	Ⅱ	

VI. CONCLUSION

This paper introduced SIFMEA based on UML in software quality control. Our analysis shows that SIFMEA based on UML diagrams is feasible. The procedures of SIFMEA using UML was listed. Its effectiveness is proven by a flight control system example.

Most quality analysis tools are statistical, their main weak point is separation with software design. UML is a standardized software modeling language. The application of UML in software development process is more and more widely and modeling methods will become standardized. Quality engineers can join in software development from requirement analysis to test, so SIFMEA based on UML will combine with software design more easier than before and make FMEA more effecctive.

REFERENCES

[1] Nathaniel Ozarin. The role of Software Failure Modes and Effects Analysis for Interfaces in safety- and mission-critical systems. IEEE International Systems Conference. April.2008

[2] Herbert Hecht, Xuegao An, Myron Hecht Computer Aided Software FMEA for Unified Modeling Language Based Software RAMS 2004

[3] GJB/Z 1391-2006 Guide to Failure Mode, Effects and Criticality Analysis

[4] Chenguang Hou，Qihua Wang, Zhanyong Ren, One Test Case Generation Method for SW&HW Reliability Co-testing, ICRMS 2011