SW-FMEA 기반의 결함 예방 모델
(A Defect Prevention Model based on SW-FMEA)

김 효 영 † 한 혁 수 ††
(Hyo Young Kim) (Hyuk Soo Han)

요 약 성공적인 소프트웨어 개발은 QCD에 의해 결정되며, 그 중 Quality는 Cost와 Delivery를 결정하는 핵심요소이기도 하다. 그리고 소프트웨어의 규모와 복잡도가 증가함에 따라 quality의 조기 확보의 중요성이 점차 커지고 있다. 이러한 관점에서 개발 후 결함을 찾아내고 수정하는 것보다 결함예방을 위해 더 많은 노력을 기울여야 할 것이다. 결함 예방을 위해서는 peer review, testing과 같은 결함 식별활동과 함께 기존에 발생된 defect에 대한 분석을 통해 발생 가능한 결함의 주입을 차단하는 활동이 필요하며, 이를 위해 기존의 품질 데이타의 조직화 및 활용이 필요하다.

소프트웨어의 품질 예방을 위한 방법으로 system safety 확보를 위해 사용되고 있는 FMEA를 활용할 수 있다. SW-FMEA(Software Fault Mode Effect Analysis)는 예측을 통해 결함을 예방하는 방법으로, 기존에는 요구사항 분석 및 설계 시 많이 활용되어 왔다. 이러한 SW-FMEA는 개발 활동을 통해 측정되는 정보를 활용하여, 분석, 설계, 나아가 peer review나 testing 등 개발 및 관리 활동에 적용하여 결함예방(defect prevention)의 수단으로 활용 할 수 있다. 본 논문에서는 기존에 시스템 분석, 설계에 focusing된 SW-FMEA를 변형하여 product 결함뿐 아니라, 개발과정 중 발생할 수 있는 fault를 줄일 수 있는 결함 예방 model을 제안한다.

키워드 : 결함예방, SW-FMEA, 결함검출

Abstract The success of a software development project can be determined by the use of QCD. And as a software's size and complexity increase, the importance of early quality assurance rises. Therefore, more effort should be given to prevention, as opposed to correction. In order to provide a framework for the prevention of defects, defect detection activities such as peer review and testing, along with analysis of previous defects, is required. This entails a systematization and use of quality data from previous development efforts.

FMEA, which is utilized for system safety assurance, can be applied as a means of software defect prevention. SW-FMEA (Software Failure Mode Effect Analysis) attempts to prevent defects by predicting likely defects. Presently, it has been applied to requirement analysis and design. SW-FMEA utilizes measured data from development activities, and can be used for defect prevention on both the development and management sides, for example, in planning, analysis, design, peer reviews, testing, risk management, and so forth. This research discusses about related methodology and proposes defect prevention model based on SW-FMEA. Proposed model is extended SW-FMEA that focuses on system analysis and design. The model not only supports verification and validation effectively, but is useful for reducing defect detection.

Key words : Defect Prevention, SW-FMEA, Defect Detection

1. 서 론

Defect Prevention refers to the activities involved in identifying defects or potential defects and preventing them from being introduced to a product[1].

The purpose of defect prevention is to identify root causes which cause defects injected into work products like specifications, source codes, etc., and to take action to eliminate these causes so that defects are prevented from being injected in future. Defect prevention activities involve identification of

† 정 회 원 : 상명대학교 컴퓨터과학과
goma1@smu.ac.kr
†† 종신회원 : 상명대학교 소프트웨어대학 교수
hshan@smu.ac.kr
논문접수 : 2005년 12월 12일
심사완료 : 2006년 5월 9일

defects that were occurred in the past, prioritization of defects, performing analysis to find why these defects were injected in the first place, identifying solutions, piloting solutions to verify the effectiveness and implementing the identified solutions [2]. Defect prevention must be active throughout the development cycle, and the key to preventing the re-occurrence of past defects is to collect and organize data from every development phase.

Software FMEA (Software Failure Mode Effect Analysis) is a method of defect prediction and prevention by analyzing past defects. Basically, FMEA is a system safety analysis technique which is currently widely used in the automotive, aerospace, and other safety critical industries[3]. The use of FMEA for software in not as common as in the hardware and systems fields, but software FMEA can be useful in improving design, and in analyzing potential design weaknesses. In that point, SW-FMEA is has been applied to the assessment of safety critical real-time control systems embedded in military and automotive products over fifteen years[4]. As a method of system analysis and design, SW-FMEA can be applied systematization of software defect.

An important factor in the final quality of a software product is the identification of defects in the software development cycle through methods such as peer review, test logs, problems reports, and field claim. Another important factor is a sound organizational quality system. The cause of defects occurred in the past must be carefully analyzed at both the project and organizational levels for prevention from re-occurring. And that must be systematically managed and applied for defect prevention. SW-FMEA attempts to prevent defects. In other words, SW-FMEA utilizes measured data from development activities, and can be used for defect prevention on both development and management sides, for example in planning, analysis, design, peer reviews, testing, risk management, and so forth. Information gathered from the development cycle facilitates this process. However, organizing data for SW-FMEA purposes is a time consuming task, and as a result, SW-FMEA is not widely practiced. This research explores a model of SW-FMEA that can be used at the organizational level for defect prevention.

The goal of this paper is to develop useful mechanism for defect prevention. SW-FMEA technique is from the basis of proposed defect prevention model. In this research, SW-FMEA is based on the organization of historical data collected from development activities. Proposed defect prevention model will act as a foundation for risk management, and will assist in the achievement of quality goals specified in project planning phases. Ultimately it can be applied for establishing foundation of effective quality system and improving of software quality.

The following is organization of this paper. Chapter 2 reviews defect prevention approaches and research such as peer reviews and SW-FMEA, and reviews challenge of that approaches. Chapter 3 provides a description of the proposed defect prevention model.

2. Background

2.1 Method of Defect detection and prevention

2.1.1 Defect detection

Defect detection and prevention are practiced through both constructive and analytical activities. Constructive activities prevent defects from being introduced by applying appropriate processes, methods, guidelines, and so on. Analytical activities prevent defects from risk and re-occurrence through the analysis of root causes and by means of removal. Static analysis like reviews, testing, field claims, etc. is representative of defect detection.

In a strict sense, defect prevention differs from defect detection. Defect detection places emphasis on product development and information as a single project, with a focus on discovery of injected defects at specific development phases. For example, code review checklists act as an early phase detection method. Defect prevention like coding standard, focuses on preventing injection of defects in the first place. Thus, defect detection should start in early stage for defect prevention [5,6].

Peer reviews are a typical detection activity as

well as prevention activity. In this paper, a peer review is considered as inspection and informal review for the purpose of defect detection. Peer reviews can be a mean for defect prevention, and can prevent injection of defects from the current phase to the following phases. Accordingly, reviews should be conducted with all important activities like requirement analysis, design, and coding on development cycle. Data should be collected from development activities such as requirements analysis, design, coding, etc. Measured data must be analyzed and be provided as feedback to related development activities. Generally, review checklists can effectively be used in defect detection. If analyzed data from similar projects or from prior phases is reflected in the review checklists, then the reviews can be even more effective. However, there are not many systematic methods and guide for the incorporation of related data into review checklists.

2.1.2 Software defect prevention models

Defect prevention includes defect prediction, and methods of defect prediction vary. Some utilize size and system complexity metrics, while others rely on process quality data and different metrics.

Most defect prediction studies have been based on size and complexity metrics. As an example, we can take Akiyama's study, which was based on a system developed at Fujitsu, Japan. In this study, he proposed a formula that reasonably estimates the total number of defects by taking the sum of defects found during testing and the defects found within two months after the release.

Prediction models that use testing metrics are widely used in Japan. This methodology is to predict residual defects involved in the collection of data about defects during the testing phases. Accuracy of predictions in this method is proven due to stability of the development and testing environment, and the extent of data collection. It appears that the IBM NASA Space shuttle team is achieving similarly accurate predictions based on the same kind of approach as testing metrics. Data for prediction can be referred through a local DB and benchmarking.

There have been many attempts to develop multilinear regression models based on multiple metrics. The basic premise of these attempts is that many metrics are colinear; that is, they capture the same underlying attributes[5].

In the case of predictions based on size and complexity metrics, despite the generally observed correlation between complexity and defect levels, this relationship is clearly not straightforward. For example, data about size and complexity cannot explain the presence of defects introduced when the requirements are defined. Predictions based on the multivariate approaches have there limits. One of potential problems is the lack of attention to the necessary assumptions for the successful use of a particular statistical technique. Other problems are the lack of distinction between model fitting and model prediction, and unjustified removal of data points or misuse of averaged data[5].

2.1.3 SW-FMEA

FMEA is a method for the estimation and cause analysis of faults. In other words, it is a reliability and safety analysis technique, and has enjoyed extensive application in diverse products for several decades. In the software field, FMEA has been used more at the module level than at the system level. SW FMEA was introduced in 1983. Musa has defined FMEA as the process of examining possible component failures and determining the types of system failures that would subsequently result[7].

Godderd has described that SW-FMEA can be applied to diverse system designs, allowing the analysis to identify potential design weaknesses and allowing design improvements to be recommended. System level SW-FMEAs can be safety assessment of the chosen software architecture at a time when changes to the software architecture can be made cost effectively. Detailed SW-FMEA is used to verify that the protection which was intended in the top level design and assessed using system level SW- FMEA has been achieved. Both system and detailed SW-FMEAs evaluate the effectiveness of the designed in software protections in preventing hazardous system behavior under conditions of failure[4].

In this way, SW-FMEA is useful in improving

designs, and in the identification of potential design weaknesses. SW-FMEA is also use for quality risk analysis. That is, SW-FMEA is applied to hazardous system behavior under failure conditions. Software failures can result from errors in design, from defects being exposed due to specific application or environment, or from hardware failures[4].

However, SW-FMEA cannot easily be used to identify system level hazards as is done in hardware and system FMEA. In Godderd's opinion, SW-FMEA has been effective in embedded software, but extra systematized activity is required for FMEA[6]. The first step in developing a SW-FMEA is to translate system hazards that have potential software error into an equivalent set of system and software states through the process of software hazard analysis. Therefore, prior to beginning the development of a software FMEA, hazard analysis such as PHA(Preliminary Hazard Analysis) and FTA(Fault Tree Analysis) for the system should be done[4].

Since many software failures are induced by failures in underlying hardware, the application of SW-FMEA requires a wide range of data, including number of defects, cause of defects, and related conditions. Generally, software organizations have accumulated data which gives priority to test results, but defect data during all development phases is required for more thorough defect prevention.

2.2 A challenge of existing approach

Defect injection is tightly related to development life cycle. Defects are injected and detected at each phase. Thus, effective defect prevention would best be achieved through good use of measured data from development cycle and quality control activities.

However, many methods of defect prevention focus on specific sections of development, such as analysis or design. As noted, SW-FMEA is useful for cause analysis and could make action for prevention, but SW-FMEA is not widely used in software development due to the time and effort required. Consequently, a framework for analysis of previous data and simple application of SW-FMEA is not sufficient. It is another reason. Several methods for defect prediction would ignore a variety of factors relating to prediction, including change activities, requirement errors, and so on[8,9].

If peer review checklists merely make reference to general guidelines, then such checklists will not have specific information about possible defects. In this case, we miss the opportunity for defect prevention in the review process. Finally, we should be taking much time for fix because we would find many defects at the end of development. On the other hand, if checklists include procedures based on data analysis of failure modes from related software projects, then we gain the chance to predict defects during the review cycle.

This is the point that this research proposes-a defect prevention mechanism based on SW-FMEA. The proposed method makes practical application of related defect data from the full software life cycle and can apply SW-FMEA more easily.

3. Defect prevention model based on SW-FMEA

3.1 Model overview

The proposed model was constructed with reference to data from completed real projects. These projects were related to embedded software as found in HD DVD, BD players, and wireless monitors. The concerned data repeatedly showed problems among similar products. The data are the results of analysis of inspection data and testing. In this research, data was organized using a SW-FEMA format that has been previously applied to other projects. Fig. 1 is the proposed defect prevention model based on SW-FMEA. Fig. 2 is the framework for the actual practice on real projects.

This mode and framework are based on three activities. First is the identification and measurement of defected data during the development cycle. There are defect detection by testing, inspection, field claim.

Second is the organization and analysis of this data by postmortem SW-FMEA. The result of postmortem SW-FMEA is integrated Historical DB for other projects and activities.

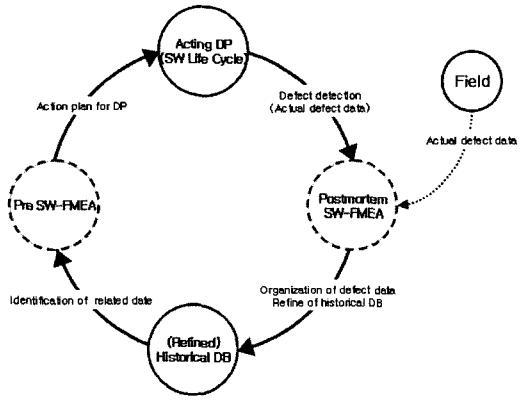

Fig. 1 Defect prevention model based on SW-FMEA

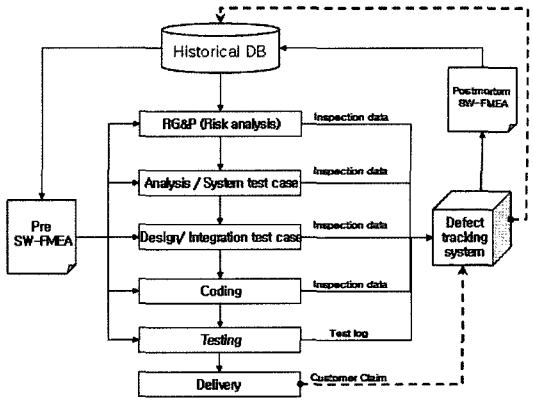

Fig. 2 defect prevention framework based on SW-FMEA

Third is the feedback of results into the development cycle through pre-SW-FMEA. Historical data in DB can used other project and activity. Under the necessity a data of historical DB is reorganized form of Pre SW-FMEA. Generally Pre SW-FMEA is maked when project start or at each development phase. Development team can refer to the data of Pre SW-FMEA and make action plan to defect prevention. They can used the data of Pre-FMEA for inspection and test activities.

3.2 Constituent and main activity of model

The following are three constituents in this model.

3.2.1 Identification of defect and measure in development process

Detection of defects is generally accomplished through verification and validation. In this research, these activities are actualized as inspection and testing, while customer claims after release serve

Table 1 Example of inspection data

Page/ Line	Defect type (Severity A,B,C)	Defect type Fault Type	Occurrence stage	Detection stage	Description	Author correction	Moderator review
R1	C	IC	A	A	Module name "Memory Card File (FAT)		
T1/T1.1	C	DF	A	A	Including electronics such as Decar, Camcorder, Mp3 player		
R1/R1.2	B	TY	A	A	Terminology error "Size.."		
R1/R1.3	B	TY	A	A	Terminology error "provided characteristic.."		
R1/R1.5	C	TY	A	A	Terminology error "In the PC.."		
T1/T1.5	C	DL	A	A	Name, property, size, date data check		
T1.8/T2.7	A	CS	A	A	keep consistency countermeasure of application under size shortage of Size		
T1.10	C	OM	R	A	Define the limit of long name file about FAR and UI		
R1.11/T1.11	C	DL	A	A	Indicate deleted file in more details		
T2.1	C	OM	A	A	Addition of "file"		
T2.2	C	OM	A	A	Specify media type / size		
R3.2	B	IC	R	A	media -> memory card		
R3.6	C	DL	A	A	Clarify update "object"		
R4.3/T4.3	B	DL	A	A	Specify "normal" operation		
T5.2	A	OM	A	A	definition of depth		
R5.2	A	DL	R	A	clarify specific test method		
T5.3	B	DL	R	A	clarity mode name and number		
T6.2	C	DL	R	A	clarify specific test method		

Table 2 Example of test log

Test item	NO	Problem	Severity	Injection phase	cause	Action plan
VPS/PDC /recording	152	HOME,Showview,Timer rec picture on the DV mode -> When out same display, Don't display key operation message	B	I	Missing about DV GUI when move to HOME, Showview, Timer rec	must delete DV picture when move to HOME, Showview, refresh DV picture when out from HOME, Showview
	162	Change to reservation recording mode -> don't display receive CH "01" CH on the PROG CH seeting mode ※ Can't advance record 01 CH	A	I	bug about reservation recording UI program	fix to reservation recording UI program
	163	Change to reservation recording mode-> must display memoryed CH on the PROG CH seeting mode, but display with previous CH	C	I	the problem is maked from don't check that done setting CH valid or unvalid in the program seeting function	first judge whether setting CH is Valid, must fix to be only selected Valide CH
	164	Change to reservation recording mode->when operate CURSOR DOWN on the PROG CH setting mode, only can UP operation, can't down operation	B	I	Itemtask() function Bug on the reservation recording UI program	reservation recording UI program Itemtask() function fix
	165	Change to reservation recording mode->permissible maximum CH 90 on the PROG CH seeting mode==> must check maximum CH (88 or 90)	B	I	Itemtask() function Bug on the reservation recording UI program	reservation recording UI program Itemtask() function fix
	166	doesn't move SHOWVIEW MODE by SHOWVIEW KEY	B	A	problem from missing key service	fix that must can Outed when Showview key is over again inputed,
	167	doesn't move from HDD MODE to reservation recording mode	A	A	problem from missing key service for reservation recording in the HDD UI function	fix that reservation recording can be serviced in HDD Ui function

as an additional method of defect identification.

The following Table 1 and Table 2 are examples of inspection and test logs.

3.2.2 Organize of defect data by Postmortem SW-FMEA

Careful collection of data throughout the defect detection cycle is very important. However, most important is the critical analysis of data, application of the analyzed results to the development cycle for prevention of future defects and fix of defect. Much time is required for data analysis with practical meaning. For this reason, many organizations rely simply on analyzing detected errors as the simple means of keeping defects out of the final product. This method provides only a little part in providing prevention. Measured data for defect prevention must be analyzed and organized as a reference for prevention of defects in further development efforts.

As mentioned above, this research has used Postmortem SW-FMEA for analysis and organization of defected data. The contents of SW-FMEA can, however, be a little different from applying field.

Fig. 3 is a case of suggested Postmortem SW-FMEA used in this paper, and Table 3 is a description of the data fields of this template.

Postmortem SW-FMEA is conducted based on results of defect detection activities such as inspection sheets, test logs, and field claims. This data is compiled at the point of postmortem review for each development phase, at the end of the

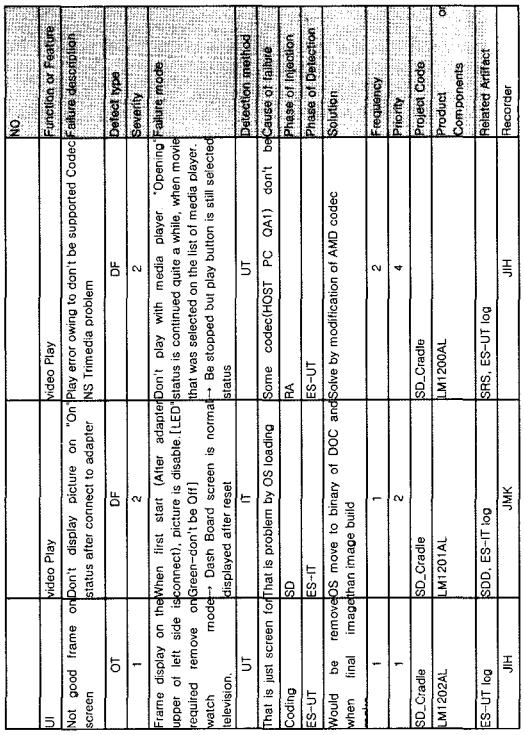

Fig. 3 Postmortem SW-FMEA Sheet

project, or as field claims are received.

Some fields of the postmortem SW-FMEA template may overlap inspection sheets or test logs. In these cases, we can use the related data-databases or appropriate spreadsheet macros to help eliminate overlaps. Databases and macros are also useful for the automatic insertion of data. However, in this research, such data is recorded manually.

Review of data compiling procedures by a project member and SQA should be conducted to ensure data reliability and to remove unnecessary data. The data processed through postmortem SW-FMEA should then be integrated with an historical DB for reference by other projects. When new data from SW-FMEA is integrated into an historical DB, some data reorganization may be required. The following relates to this reorganization.

- Add new defect information
- Group related defects, and add information about failure modes and defect causes for related groups.
- Correct invalid information
- Revise defect probabilities and priorities of defect prevention tasks
- Make possible countermeasures for high-inci-

Table 3 Postmortem SW-FMEA field

Field	Description
Function or Feature	Function or feature of measured and analyze object
Failure description	Description of failure
Defect type	Defect type is defined by organization or project
Severity	Quantify the impact of a failure on the system (1: Comment, 2: Minor, 3: Major,)
Failure mode	Description about that the process does failure (It is not applicable in the inspection)
Detection method	The method that is used to detection of the defect. For example inspection, Unit test, Integration test, system test, and field claim (It is not applicable in the inspection)
Cause of failure	Description of cause of the failure
Phase of Injection	The phase which defect is injected
Phase of Detection	The phase which defect is detection
Solution	Description of Solution how defect is solved
Frequency of occurrence	How many times same defect was occurred
Priority	It is calculated field, the result of multiplying severity and frequency. The more figure is high, the more risky (Severity X Frequency)
Project Code	Related Project code with this defect
Product or Components	The product model or component that this function is included
Related Artifact	Related development product (configuration item) like SRS, HLD, LLD, Source code, test case, test log etc.
Recorder	Enter the name of the person completing the Postmortem SW- FMEA

dence defects

Finally, the historical DB must be optimized into a most suitable structure to extract and insert data in question. This is conducted by the developer and SQA as a joint effort.

3.2.3 Pre SW-FMEA for defect prevention

While detection and analysis of defects are important, the most important task is the application of analyzed data to defect prevention in future projects or following activities. Generally, a risk management plan is devised through risk analysis in the project planning phase. Likewise, when testing is planned, priority and focus should be on identifying risks vis-à-vis the results of the risk analysis phase. The proposed Pre SW-FMEA is conducted at the beginning of project by refined historical DB. The Pre SW-FMEA can be conducted in each development phase.

In this research, we consider that Pre SW-FMEA is conducted at project's inception. Relevant data for risk management, analysis, and design are extracted from the historical DB and used as major checklist items for each phase. Pre SW-FMEA is also used for inspection checklists and design of test scenarios. Furthermore, the Pre SW-FMEA is used for reorganizing checklist items. This becomes an import solution when similar defects are detected in a new project.

Fig. 4 is a case of proposed Pre SW-FMEA used

NO.	Function or Feature	Failure description	Potential Failure mode	Potential Effects of Failure	Severity	likelihood	Priority	Recommended solution/action	Current Phase	Related Phase	Reference
FAT	Install/Uninstall	When system restarts after install media library, error with program add, remove and uninstall	Install/Uninstall be stopped	> Bring problem about usability, reliability, and portability > Bring of rework	3	1	3	> Code inspection > Functions about program add/remove can be available with supported whole OS	RA	SDD, Coding	SD_Cradle : ES-IT log
VDP	Error about play due to codec don't support	"Opening" status is continued quite a while and be stopped under play button is selected	> Reduce of usability and reliability > Need to modify of source code	2	3	6	> Inspection about Requirement anc code > Clarify support codec in RA phase and then implementation of codec connect function	RA	SDD, Coding	SD_Cradle : ES-UT log
Omission of required file information	When contents is played, required information omit	> Rework (GUI and code) > Reduce usability because of required information is omitted		2	1	2	> Check with requirement inspection > Clarify of requirement - detail information define on the GUI	RA	RA, Coding	RH2003_inspection_1

Fig. 4 Pre SW-FMEA sheet

Table 4 Pre SW-FMEA field

Field	Description
Function or Feature	Fuction or feature of measured and analyze object.
Failure description	Description of failure
Potential Failure mode	Identified potential failure mode by prediction. Describe ways in which the process might failure
Potential Effects of Failure	Describe the impact that the potential failure would have on system.
Severity	Quantify the impact of a potential failure on the system. (1: Comment, 2: Minor, 3: Major,)
likelihood	The scale of probability of something happening (1: low, 2: middle, 3: high)
Priority	It is calculated field, the result of multiplying severity and likelihood. The more figure is high, the more risky (Severity X likelihood)
Recommanded solution/action	Describe the solution that will be taken to reduce the occurrence or prevent the failure
Current Phase	Current phase that this SW-FMEA is conductting
Related Phase	Relatied phase with failure or phase is in need of some action for prevention
Reference (Project Code-NO)	Referenced documentation for SW-FMEA like number in the historical DB, name of relatied documentation or project code and so on.

in this paper. It uses data of Postmortem SW-FMEA and related data for analysis. Pre SW-FMEA somewhat varies from Postmortem SW-FMEA in that priorities are determined by likelihood as opposed to frequency. Table 4 describes the Pre SW-FMEA template fields.

4. Conclusions

Software will continue to grow in size and complexity, and the importance of software quality assurance will therefore increase. To improve the final quality of software, defect prevention should begin at an early stage of development. Toward the conclusion, our research has considered application of SW-FMEA to ensure software quality from an early stage of development. Many similar approaches have been attempted for defect prevention, and this research has attempted to analyze both the benefits and limitations of these approaches. We then proposed a defect prevention model based on a SW-FMEA framework which appears to be more effective than current methodology employed in the industry.

Several activities are required for the proposed defect prevention model which has been based on advanced SW-FMEA. This is a disciplined process, requiring proper collection of relevant data. The proposed model will become more effective in development organization with stable process. Thus, systematic activities such as data gathering, management, and organization of defects are important.

In the foreseeable future, I believe that continuous research will reduce the weak points seen presently in the proposed defect prevention model. Improvements in the use of available data sources could allow needed data to be more easily extracted from existing data, perhaps by better interconnection between Postmortem SW-FMEA, historical data, and Pre SW-FMEA.

References

[1] Software Engineering Institute, Key Practice of the Capability Maturity Model, version 1.1, CMU/SEI-93-TR-25, Canagie Mellon University, Pittsburgh, PA, 1993.

[2] Sanjay Mohapatra, B. Mohanty, "Defect Prevention through Defect Prediction: A Case Study at Infosys," *Pro. 17th IEEE International Conference on Software Maintenance (ICSM'01)*, pp.260-272, 2001.

[3] Yiannis Papadopoulos, David Parker, Christian Grante, "Automating the Failure Modes and Effects Analysis of Safety Critical Systems," *Pro. 8th IEEE International Symposium on High Assurance Systems Engineering (HASE'04)*, pp. 310-311, 2004.

[4] Peter L. Goddard, Raytheon,Troy, "Software FMEA techniques," *Pro. annual Reliability and Maintainability symposium*, IEEE, pp.119-123, 2000.

[5] Norman E. Fenton, et al., "A Critique of Software Defect Prediction Models," IEEE Computer Society, IEEE Computer Society, *IEEE Transactions on Software Engineering*, vol. 25, no. 5, Septemper/ October, pp.675-689, 1999.

[6] J.H. van Moll, J. C. Jacobs, B. Freimut, J.J.M. Trienekens, "The Importance of Life Cycle Modeling to Defect Detection and Prevention," *Pro. 10th International Workshop on Software Technology and Engineering Practice (STEP'02)*, IEEE, 2002.

[7] Zenzen, Frances Elisabeth, "Software Reliability planning prediction models," PhD thesis, Aeisona State University, 2000.

[8] Nancy S. Eickelmann and Debra J. Richardson, "A Defect Prevention Approach to Architecture-Based Testing," *Pro. COMPSAC '97 - 21st International Computer Software and Applications Conference*, IEEE, 1997.

[9] Forrest Shull et al., "What We have Learned About Fighting Defects," *Pro. 8th IEEE Symposium on Software metrics(METRICS'02)*, 2002.

김 효 영

1999 Dept. of Multimedia, SangMyung Univ. Graduate School of Information and Telecommunications(Master). 2000~2002 Dept. of Computer Science, SangMyung Univ. Graduate School (Completion of Ph.D Courses). 2002~2004 Software Strategy Gr. Digital Media Laboratory of LG Electronics (Manager). 2005~Present Software Engineering Gr. Software & Solution Center of LG Electronics (Senior Research Engineer). Research Interests: Software Quality, Software Process, Software Usability Evaluation, Software Testing, Software Metric

한 혁 수

1985 Seoul National Univ. Dept of computer science (Bachelor). 1987 Seoul National Univ. Dept of computer science (Master). 1992 University of South Florida. Dept of computer engineering (Ph.D). 2001~Present Chairman of SITRI (System Integration Technology Research Institute). 2003 Executive Director of KIPA (Korea IT Inductry promotion Agency) in KSI(Korea Software Institute). 2004~2005 Dean of SangMyung University School of Software. 1993~Present Sang Myung University School of Software(Professor). Research Interests: Software Process, Software Quality, Software Usability Evaluation etc.