KIPS Trans. Softw. and Data Eng.
Vol.9, No.7 pp.205~212 pISSN: 2287-5905

https://doi.org/10.3745/KTSDE.2020.9.7.205

A Study on the Methodology for Defect Management in the Requirements Stage

Eun-Ser Lee†

ABSTRACT

Defects are an important factor in the quality of software developments. In order to manage defects, we propose additional information of search and classification. Additional information suggests a systematic classification scheme and method of operation. In this study, we propose additional information at the requirements analysis stage for defect management.

Keywords : Defect Management, Software Quality, Requirement Management, Estimation of Software Quality

요구사항단계의 결함관리를 위한 방법론에 관한 연구

이 은 서†

요 약

결함은 소프트웨어의 품질에서 중요한 요소가 된다. 결함을 관리하기 위하여 검색과 분류의 추가적인 정보를 제안하고자 한다. 추가적인 정보는 체계적인 분류체계와 연산방법을 제시한다. 본 연구에서는 결함관리를 위하여 요구사항 분석 단계의 추가적인 정보를 제시하고자 한다.

키워드 : 결함관리, 소프트웨어 품질, 요구사항 관리, 소프트웨어 품질 추정

1. 서 론

소프트웨어 검토는 결함을 제거하기 위하여 많은 단계와 프로세스로 구성이 되어 있다. 소프트웨어 개발 시에 완성도를 높이는 노력은 간과할 수 없으며 결함을 제거하기 위한 활동이 전반적인 시스템의 효율성과 효과성을 높일 수 있다. 결함을 제거하기 위하여 검토는 다양한 관점으로 오류와 결함을 제거하여 문제가 없는 소프트웨어를 만들고자 한다[1-4].

결함은 소프트웨어의 개발 단계에서 요구사항분석, 설계, 구현, 유지보수 등 모든 단계에서 신뢰성을 떨어뜨리고 잠재적인 문제를 발생해서 커다란 파장을 발생시키기도 한다. 이러한 형태는 버그, 오류, 결함으로 나타나게 된다. 따라서 이와 같은 문제들을 관리하여 소프트웨어의 신뢰성을 향상하고자 한다. 신뢰성은 하부 속성등에 의하여 소프트웨어 가용할 수 있는 것을 나타내는 것으로 가용과 연관된 시간의 양을 의미한다[5-7].

본 연구에서는 결함을 관리하기 위하여 요구사항분석 시에 추가적인 정보를 제시하고자 한다. 추가적인 정보는 결함을 분류하고 이를 찾는데 활용되는 정보이다. 추가적인 정보를 활용하여 소프트웨어 개발 시에 입력을 통하여 결함의 발생 위치를 정확하고 신속하게 찾아서 제거하기 위함이다.

2. 기반연구

2.1 요구사항단계의 결함

소프트웨어 개발 생명주기에서 소프트웨어 제품을 평가하는 동안 사용되는 방법에 대하여 성능 요구사항의 정의가 필요하다. 검토 프로세스는 제품 개발 상태 평가를 기반으로 계획에 따른 진행 상황을 평가한다. 그리고 프로젝트 방향을 변경하거나 대체 계획의 필요성을 식별하기 위한 권장 사항이 제공되어야 한다. 적절한 자원 배분을 통해서 프로젝트의 전반적인 통제를 유지하기 위한 권장 사항이 제공된다. 표준, 지침, 사양 및 절차에 대한 제품 및 프로세스 준수의 증거가 제공된다. 특정 소프트웨어 요소가 해당 사양을 준수한다는 증거가 제공된다[8,9]. 소프트웨어 요소의 개발(또는 유지 보수)이 프로젝트에 적용 가능한 계획, 표준 및 지침에 따라 수행되고 있다는 증거가 제공된다. 소프트웨어 요소에 대한 변

※ 이 논문은 안동대학교 기본연구지원사업에 의하여 연구되었음.
† 종신회원 : 안동대학교 컴퓨터공학과 교수
Manuscript Received : February 12, 2020
Accepted : February 27, 2020
* Corresponding Author : Eun-Ser Lee(eslee@anu.ac.kr)

경이 올바르게 구현되고 변경 사양으로 식별 된 시스템 영역에만 영향을 준다는 증거가 제공된다. 검토 결과는 작업 내역을 제공하고 작업에 대한 정보를 제공하기 위해 개발자, 관리자 및 담당자에게 전달된다. 소프트웨어 검토 프로세스에 대해 범위, 목표, 프로세스 목표 달성 정도를 결정하기위한 조치, 프로세스 고유의 역할, 프로세스가 적용되는 제품, 조건(들)을 지정하게 된다[10,11].

소프트웨어는 대부분의 장비 및 비즈니스 시스템에 사용됩니다. 소프트웨어 사용자의 기대를 달성하려면 소프트웨어 개발 수명주기에 따라 소프트웨어 제품을 평가할 수있는 훈련 된 방법을 갖추어야 한다[12].

요구사항 명세의 결함 형태는 다음과 같다[13].

1. 요구사항이 문서에서 일부가 누락되거나 또는 의미가 완전한 형태로 명세가 되지 않는 경우
2. 요구사항에서 제공하는 정보가 잘못 기입된 경우
3. 요구사항의 내용이 일관성 없이 표현되는 경우
4. 요구사항의 정보가 객관적이지 않은 경우
5. 요구사항의 명세가 기능을 잘못 기입하거나 패키지와 모듈에 잘못 배치한 경우
6. 요구사항의 실현 가능성과 기능이 잘 구현되었는지 검증이 불가능한 경우
7. 요구사항의 기능이 중복되어서 명세가 되는 경우
8. 요구사항의 표현의 오류가 발생하는 경우
9. 요구사항이 개발하는 범위에 해당되지 않는 경우

Table 1. Comparison of Review Types

Characteristic	Management review	Technical review	Inspection	Walk-through	Audit
Objective	Ensure progress; recommend corrective action; ensure proper allocation of resources	Evaluate conformance to specifications and plans; ensure change integrity	Find anomalies; verify resolution; verify product quality	Find anomalies; examine alternatives; improve product; forum for learning	Independently evaluate compliance with objective standards and regulations
Decision-making	Management team charts course of action; decisions made at the meeting or as a result of recommendations	Review team requests management or technical leadership to act on recommendations	Review team chooses predefined product dispositions; defects must be removed	The team agrees on changes to be made by the author	Audited organization, initiator, acquirer, customer or user
Change verification	Leader verifies that action items are closed; change verification left to other project controls	Leader verifies that action items are closed; change verification left to other project controls	Leader verifies that action items are closed; change verification left to other project controls	Leader verifies that action items are closed; change verification left to other project controls	Responsibility of the audited organization
Recommended group size	Two or more people	Three or more people	Three to six people	Two to seven people	One to five people
Group attendance	Management, technical leadership and peer mix	Technical leadership and peer mix	Peers meet with documented attendance	Technical leadership and peer mix	Auditors, audited organization, management and technical personnel
Group leadership	Usually the responsible manager	Usually the lead engineer	Trained facilitator	Facilitator or author	Lead auditor
Volume of material	Moderate to high, depending on the specific meeting objectives	Moderate to high, depending on the specific meeting objectives	Relatively low	Relatively low	Moderate to high, depending on the specific audit objectives

2.2 소프트웨어 위험 관리

소프트웨어를 효율성 있게 개발하기 위하여 많은 방법론과 내용이 연구되었다. 개발의 효율성뿐만이 아니라 발생하는 결함을 예방하고 원인을 찾아서 제거하기 위한 연구도 활발이 진행되고 있다. 소프트웨어 결함을 그대로 방치하게 되면 위험요소로 발생하여 비용, 일정, 품질에 영향을 미치게 된다.

위험 분석과 관리는 여러 단계로 구성되어 있으며 소프트웨어 팀의 이해를 높이고 불확실성을 관리하는데 많은 도움을 준다. 많은 문제들은 소프트웨어 프로젝트에서 다루어진다[14,15]. 위험은 잠재적인 문제들이라고 정의할 수 있으며 소프트웨어 개발 시에 나타날 수 도 있고 나타나지 않을 수도 있다. 따라서 이와 같은 것을 인지하고 발생하는 확률을 예측하고 다른 곳에 미치는 영향을 파악하기 위해서 지속적으로 계획과 실제 발생하는 문제들을 확인하고 수정해야 한다[16,17]. 이와 같은 위험요소는 많은 프로세스와 연관되어 있으며 프로세스와 연관된 스테이크홀더가 위험을 제거하기 위하여 노력해야 한다.

위험 요소를 제거하기 위하여 위험요소를 인지해야 한다. 위험요소를 제거하고 관리하기 위해서는 대상이 무엇인지 먼저 정의하고 내용을 정확히 파악하고 있어야 한다. 그 다음에 위험 요소에 대한 분석을 수행해야 한다. 위험 요소가 어떻게 영향을 미치고 언제 발생하는지에 대하여 분석을 해야 한다. 마지막으로 위험요소의 높은 발생비율과 영향을 계획에 반영하여 제거하고 관리하도록 해야 한다[18,19].

3. 본 론

소프트웨어 개발 시에 많은 결함이 발생하고, 이를 해결하여 프로젝트를 종료하게 된다. 이와 같은 과정에서 결함 발생의 원인을 제거해야 한다. 그 이유는 개발된 소프트웨어가 결함의 원인이 해결되지 않은 상황에서 유사한 분야에 재사용하게 된다면 같은 결함이 발생하여 전체 시스템의 기능과 성능에 악영향을 미치게 된다.

본 연구에서는 2.1에서 제시된 결함유형의 분류를 기반으로 결함을 관리하기 위하여 요구사항분석, 설계, 구현 시에 추가적인 정보를 제시하고자 한다. 추가적인 정보는 결함을 분류하고 이를 찾는데 활용되는 정보이다. 추가적인 정보를 활용하여 소프트웨어 개발 시에 입력을 통하여 결함의 발생위치를 정확하고 신속하게 찾아서 제거하기 위함이다.

추가적인 정보는 요구사항단계에 의하여 제시하였으며, 개발단계별로 제시하였다. 본 연구에서는 이와 같은 추가적인 정보를 CI (Critical Items)라고 명명하였다. 각 단계별로 요구사항분석과 연관된 것은 RCI (Requirement of Critical Items) 라고 정의하였다. 본 연구에서는 RCI를 대상으로 하며 DCI와 CCI는 추가적인 연구를 통하여 제시하고자 한다.

개발단계별 결함이 발생되면 결함의 종류에서 해당되는 결함의 발생빈도를 저장한다. 저장된 결함은 누적된 수치를 기반으로 하여 결함의 원인으로 판단하게 된다. 소프트웨어 개발단계에서 발생되는 이슈들은 해당되는 결함을 찾게 되고, 그 발생빈도를 누적하게 된다. 그리고 지속적으로 발생되는 경우 결함으로 인식되어서 결함관리를 수행하게 된다. 이와 같은 과정을 다음과 같이 도식화할 수 있다.

결함의 원인으로 판단하기 위해서 기준을 제시할 필요가 있다. 따라서 이벤트 발생에 의한 결함 누적 수치를 기반으로 결함의 원인으로 판단할 기준을 제시하였다. 판단 기준은 다음과 같다.

Table 2. Criteria of Degree of Defect

Measurement	Points
H (High)	81 or more
N (Normal)	51-80
L (Low)	0-50

결함 판단 기준을 기반으로 하여 이벤트가 발생되면 단순 이슈인지 아니면 결함이 될 것인지를 판단하게 된다. 결함 기준에서 H는 결함이 될 가능성이 높으므로 결함으로 인식하여 관리를 수행하게 된다. L의 경우는 단순 오류나 이슈로 끝나는 경우가 대부분이어서 결함관리를 수행할 필요가 없다. 그리고 N의 경우는 단순오류인지 결함이 될 가능성이 있는지 단계로서 간과해서는 안되는 항목이 된다. N이 발생된 항목에 대해서는 주기적으로 모니터링하여 누적된 발생빈도를 확인해야 한다. 누적 점수는 3.1절에서 기능 정의, 수행, 확인의 세 단계를 기반으로 높음, 중간, 낮음으로 제시하여 점수를 누적하게 된다. 누적 점수를 합산하여 결함 판단 기준으로 활용하게 된다.

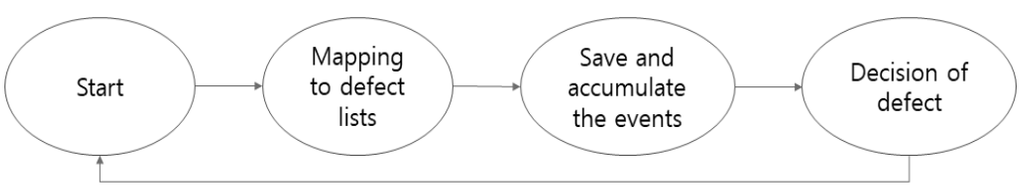

Fig. 1. Algorithm of Defect Accumulate

Table 3. Criteria of Phase

	Contents and Examples
Definition of Function	Functional and Non-functional, Procedure, Process, Related data
Performance	Using of correct data, Performed of correct procedure and process
Check	Result of using data, Result of procedure and process

Table 4. Degree of Definition of Function

Measurement	Contents	Points
DF-H (Definition of function-High)	Identify objects and relationships of functions	0-50
DF-N (Definition of function-Normal)	Identify simple functions	51-80
DF-L (Definition of function-Low)	Unidentified targets and relationships of functions	81 or more

Table 5. Degree of Performance

Measurement	Contents	Points
P-H (Performance-High)	Performance by objects and relationships of functions	0-50
P-N (Performance-Normal)	Performance by simple functions	51-80
P-L (Performance-Low)	Performance is inconsistent	81 or more

Table 6. Degree of Confirm

Measurement	Contents	Points
C-H (Confirm-High)	The result of analysis of the object and relationship of the function can be confirmed.	0-50
C-N (Confirm-Normal)	The result of analysis of the operations of the function can be confirmed.	51-80
C-L (Confirm-Low)	The result is not confirm.	81 or more

3.1 RCI (Requirement of Critical Items)

요구사항단계에서 발생되는 결함을 관리하기 위한 추가적인 정보는 결함의 정보가 기본적으로 제공되어야 한다. 따라서 요구사항과 연관된 결함은 2.1절에서 소개한 내용을 기반으로 한다. 요구사항단계에서 결함의 발생 위치와 이에 대한 대응을 위하여 다음과 같은 추가적인 정보가 필요하게 된다. 요구사항단계에서 발생하는 결함을 처리하기 위해서는 결함의 유형을 선행적으로 정의해야 한다. 제시된 결함 항목에서 요구사항 결함과 연관된 항목으로는 목표설정, 의사결정, 변화적용이 있다. RCI에서는 3가지를 대상으로 결함관리의 추가적인 정보와 누적을 위한 연구를 수행하고자 한다.

결함관리를 위하여 여러 관점의 정보가 필요하게 된다. 여러 관점은 목표설정(Goal, G), 의사결정(Decision, D), 변화적용(Change, C)을 대상으로 분석하였다. 각 관점에 의한 결함분석에 의한 추가적인 정보를 추출하기 위하여 다음과 같은 기준으로 수행하였다. 기준은 다음과 같다. 기능 정의, 수행, 확인에 의하여 각 관점을 분석하였다.

기능 정의는 목표설정에서 수행할 기능을 정의하고 있는지와 연관된 정보 및 결함을 의미한다. 예로는 기능적 · 비기능적 요구사항의 정의와 연관하여 핵심기능이 추출되어 있는가를 의미한다. 핵심기능은 없어서는 안될 기능으로서 이 기능이 누락된다면 기능 수행에 있어서 치명적인 문제가 발생되는 것을 의미한다. 핵심기능이 아닌 것은 해당 기능이 없어도 기능수행에 문제가 되지 않는 기능을 의미한다. 본 연구에서는 핵심기능을 대상으로 수행하게 된다.

수행은 정의된 기능을 기반으로 올바른 방법의 프로세스와 데이터를 활용하여 기능이 수행되는 것을 의미한다. 예로는 올바른 절차와 프로세스의 수행, 올바른 데이터의 사용 등이 있다.

확인은 정의된 기능이 명시된 절차와 프로세스를 기반으로 데이터를 활용하는가를 검토하는 단계이다. 확인이 완벽히 끝나면 기능 정의부터 수행까지 문제없이 수행되고 종료가 되는 상황을 의미한다.

기능 정의는 기능을 얼마나 정확하고 필요한 항목을 추출

하는가를 측정한다. 기능 정의의 성숙도를 측정하기 위하여 다음과 같은 기준을 제시하고자 한다.

기능 정의의 성숙도는 3단계로 제시하였다. 기능 정의의 성숙도를 분류하기 위하여 DF(Definition of function)으로 명명하였다. 이를 기반으로 하여 높음, 중간, 낮음의 3단계로 성숙도를 제시하였다. DF-H는 기능의 대상과 관계가 명확하고 완전하게 파악되는 경우를 의미한다. DF-N은 기능의 대상과 관계가 완전히 파악되지는 않지만, 단순 기능이 파악되는 경우를 의미한다. DF-L는 기능의 대상과 관계, 기능이 파악이 되지 않는 것을 의미한다.

수행의 성숙도는 3단계로 제시하였다. 기능 정의의 성숙도를 분류하기 위하여 P(Performance)으로 명명하였다. 이를 기반으로 하여 높음, 중간, 낮음의 3단계로 성숙도를 제시하였다. P-H는 기능의 대상과 관계에 의하여 수행되는 경우를 의미한다. P-N은 단순 기능에 의하여 수행되는 경우를 의미한다. P-L는 수행이 일관성 있게 되지 않는 것을 의미한다.

확인의 성숙도는 3단계로 제시하였다. 기능 정의의 성숙도를 분류하기 위하여 C(Confirm)으로 명명하였다. 이를 기반으로 하여 높음, 중간, 낮음의 3단계로 성숙도를 제시하였다. C-H는 기능의 대상과 관계의 분석결과물을 확인할 수 있는 경우를 의미한다. C-N은 단순 기능에 의한 오퍼레이션을 확인할 수 있는 경우를 의미한다. C-L는 결과물을 확인할 수 없는 경우를 의미한다.

기능 정의, 수행, 확인의 기준을 기반으로 하여 목표설정(G), 의사결정(D), 변화적용(C)을 정의하고자 한다. 결함 정도를 파악하기 위하여 목표설정, 의사결정, 변화적용의 분류를 기반으로 하여 기능 정의, 수행, 확인의 성숙도를 측정하게 된다. 산정의 결과는 분류에 대하여 전반적인 결함 정도를 예측하고 판단할 수 있다. 산정은 각 항목의 점수를 산정하여 가중치를 곱하여 최종 점수를 산정하여 합산하게 된다. 그리고 비율을 측정하기 위하여 백분율로 환산하게 된다. 수식은 다음과 같다. 여기서 가중치는 개발되는 소프트웨어의 요구사항에 따라서 다르게 산정이 된다. 그 범위는 0-1을 입력하게 된다. 산정을 한 후에 전체 요구사항의 개수(n)로 나누어 준다.

$$(G(\sum_{DF=1}^{i}(DFpoint*\ Weight)+\sum_{P=1}^{i}(Ppoint*\ Weight)+\sum_{C=1}^{i}(Cpoint*\ Weight))/3+$$
$$D(\sum_{DF=1}^{i}(DFpoint*\ Weight)+\sum_{P=1}^{i}(Ppoint*\ Weight)+\sum_{C=1}^{i}(Cpoint*\ Weight))/3+$$
$$C(\sum_{DF=1}^{i}(DFpoint*\ Weight)+\sum_{P=1}^{i}(Ppoint*\ Weight)+\sum_{C=1}^{i}(Cpoint*\ Weight))/3)/n$$

결과물은 70이상이 결함 발생 가능성이 높고, 41-69인 경우에는 중간, 40이하인 경우에는 발생가능성이 낮게 나오는 것으로 기준을 정하였다.

4. 적용 및 사례

4.1 미아방지 목걸이 설계

본 연구에서 3장의 이론을 적용하기 위하여 미아방지 목걸이를 설계하였다. 설계의 기반은 UML을 기반으로 수행하였다. 미아방지 목걸이는 실시간으로 아동의 위치를 파악하여 위험요소를 사전에 인지하게 된다. 그리고 위험요소를 네트워크를 통하여 전달하는 방식으로 실종되는 아동을 줄이고 예방하기 위함이다. 유스케이스와 클래스다이어그램은 다음과 같다.

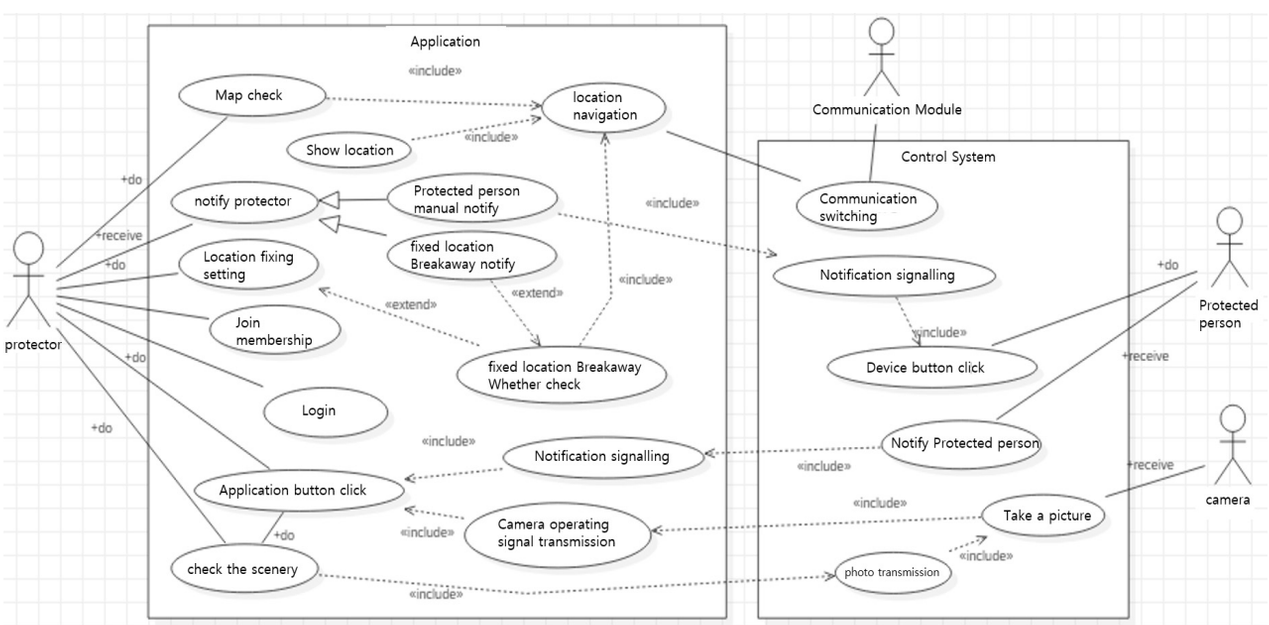

Fig. 2. UseCase Diagram

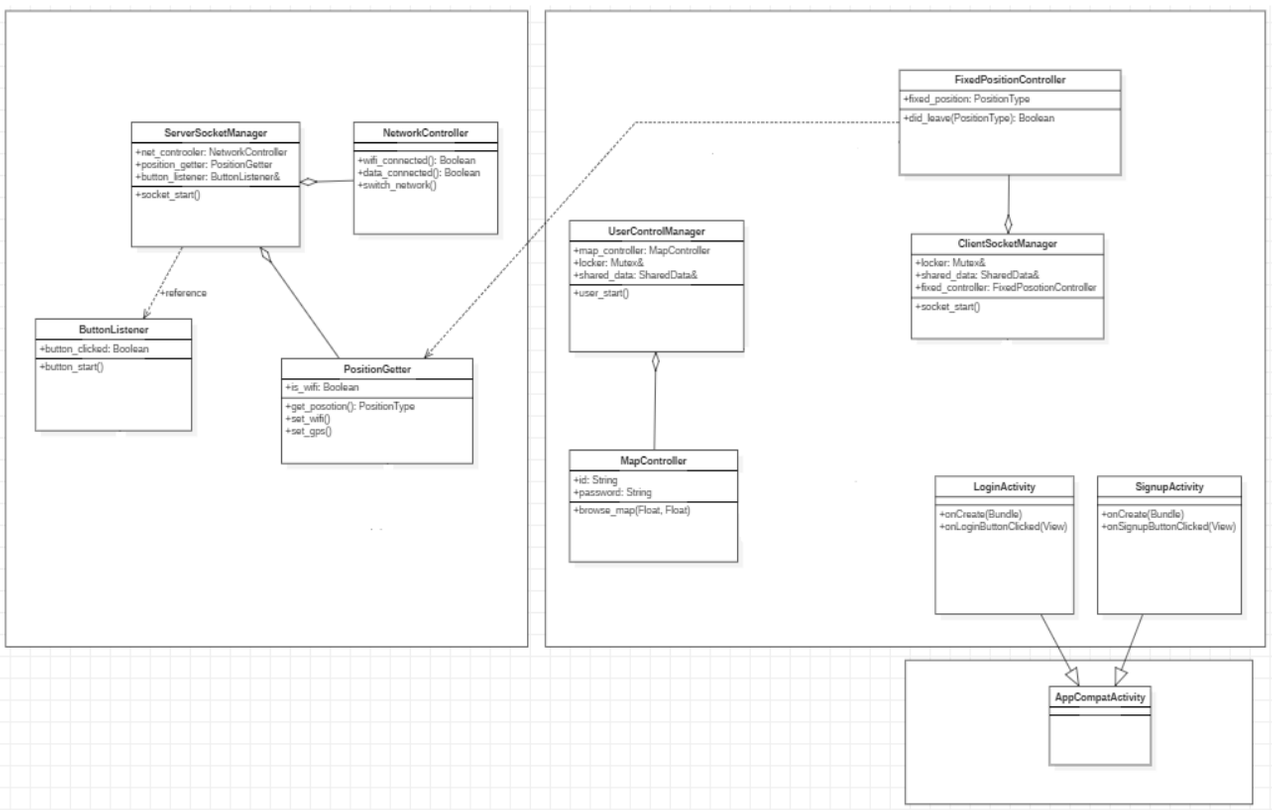

Fig. 3. Class Diagram

4.2 검증

미아방지 목걸이 설계를 기반으로 본 연구의 이론을 검증하고자 한다. 검증을 연구에서 제시한 이론을 적용하여 산정한 결과 결함제거효율의 산정 방식을 적용하였다.

제시한 이론에 의하여 요구사항단계에서 목표설정, 의사결정, 변화적용을 분석하였다.

요구사항에 대한 내용은 다음과 같다.

R-0001(지도 확인), R-0002(위치 표시), R-0003(위치탐색), R-0004(통신스위칭), R-0005(보호자에게 알림), R-0006(피보호자 수동 알림), R-0007(피보호자에게 알림), R-0008(알림 신호 전달), R-0009(위치 고정 설정), R-0010(고정 위치 이탈 여부 측정), R-0011(고정 위치 이탈 알림), R-0012(주변 풍광 확인), R-0013(카메라 작동 신호 전송), R-0014(사진 찍기), R-0015(회원 가입), R-0016(로그인), R-0017(어플 버튼 눌림), R-0018(기기 버튼 눌림), R-0019(사진 전송), R-0020(알림), R-0021(기능적 요구사항-기기 on, off), R-0020(비 기능적 요구사항-알림), R-0021(비 기능적 요구사항-기기 on, off), R-0022(비 기능적 요구사항-방수 기능), R-0023(비 기능적 요구사항-휴대 가능한 사이즈)

요구사항에 대한 추출은 기능적 요구사항과 비기능적 요구사항으로 분류하였다.

기능 정의, 수행, 확인을 3장에서 제시한 수식에 의하여 산정을 한 결과에 나머지 수식을 적용한 결과는 다음과 같다.

(88/3 + 55/3 + 96/3 + 12/3 + 24/3 + 24/3 + 24/3 + 12/3 + 12/3 + 168/3 + 30/3 + 50/3 + 12/3 + 12/3 + 6/3 + 6/3 + 15/3 + 15/3 + 12/3 + 6/3 + 15/3 + 6/3 + 6/3 + 6/3 + 6/3) /25 = 9.57

전반적인 산정 결과는 9.57로 결함이 낮게 분석되었다. 세부적인 결함 원인을 분석하기 위하여 목표설정, 의사결정, 변화적용 별로 산정한 결과는 다음과 같다.

목표설정(G) = (88/3 + 96/3 + 12/3 + 12/3 + 50/3 + 12/3 + 6/3 + 6/3 + 12/3 + 15/3 + 6/3 + 6/3 + 6/3 + 6/3) / 14 = 7.9

의사결정(D) = (55/3 + 12/3 + 30/3 + 12/3 + 15/3 + 15/3 + 6/3 + 6/3) / 8 = 50

변화적용(C) = (24/3 + 24/3 + 24/3 + 168/3) / 4 = 20

세부 산정 결과에 따르면 의사결정과 연관된 항목에서 결함 원인의 가능성이 있다는 것을 나타내고 있다.

본 사례를 통하여 이론을 적용한 결과 발생 가능성이 높은 요구사항이 추출되었다. 이와 같은 결함 요소는 의사결정에 있어서 위험 요소가 될 가능성이 높음을 의미한다. 따라서 의사결정의 명확성, 정확성, 타당성, 상호의존성에 대한 추가적인 분석과 대안이 필요하다. 그리고 추가적인 대안의 검증을 알고리즘과 정량적인 상태로 나타내어서 결함을 제거할 수 있는 노력이 요구된다.

Table 7. Requirements of Categories

Main Category	Category	Serial Number	DF points	P points	C points	Weight	Result
Function	G	R-0001	50	30	30	0.8	88
	D	R-0002	50	30	30	0.5	55
	G	R-0003	60	30	30	0.8	96
	G	R-0004	20	20	20	0.2	12
	C	R-0005	50	40	30	0.2	24
	C	R-0006	50	40	30	0.2	24
	C	R-0007	50	40	30	0.2	24
	D	R-0008	20	20	20	0.2	12
	G	R-0009	20	20	20	0.2	12
	C	R-0010	70	70	70	0.8	168
	D	R-0011	20	20	20	0.5	30
	G	R-0012	50	30	20	0.5	50
	D	R-0013	20	20	20	0.2	12
	G	R-0014	20	20	20	0.2	12
	G	R-0015	10	10	10	0.2	6
	G	R-0016	10	10	10	0.2	6
	D	R-0017	10	10	10	0.5	15
	D	R-0018	10	10	10	0.5	15
	G	R-0019	30	20	10	0.2	12
	D	R-0020	10	10	10	0.2	6
	G	R-0021	10	10	10	0.5	15
Non-function	D	R-0020	10	10	10	0.2	6
	G	R-0021	10	10	10	0.2	6
	G	R-0022	10	10	10	0.2	6
	G	R-0023	10	10	10	0.2	6

5. 결 론

본 연구에서는 결함을 관리하기 위하여 요구사항분석 시에 추가적인 정보를 제시하고자 한다. 추가적인 정보는 목표설정, 의사결정, 변화적용의 세 부분을 기반으로 기능 정의, 수행, 확인의 관점으로 분석하였다. 추가적인 정보는 결함을 분류하고 이를 찾는데 활용되는 정보이다. 추가적인 정보를 활용하여 소프트웨어 개발 시에 입력을 통하여 결함의 발생 위치를 정확하고 신속하게 찾아서 대안을 세우고 제거하기 위함이다.

향후 연구내용으로는 추가적인 정보를 개발 분야에 의하여 세분화하여 제시하고 통계를 활용하여 추가적인 정보에 대한 증명을 수행하고자 한다. 또한 설계와 구현단계의 추가적인 정보를 정의하고 구축하고자 한다.

References

[1] Roger S. Pressman, “Software Engineering,” McGraw-Hill Higher Education, 2005.

[2] Alvin, W. H., von (ed.), “Reliability Engineering,” Prentice-Hall, 1964.

[3] B. Boehm, “Software Engineering Economics,” Prentice-Hall, 1981.

[4] J. Rook, “Software Reliability Handbook,” Elsevier, 1990.

[5] A. J. Albrecht, “Measuring Application Development Productivity,” *Proc. IBM Application Devlopment Symposium, Monterey*, CA, 1997.

[6] V. R. Basilly and D. M. Weiss, “ A Methodology for Collect Valid Software Engineering Data,” *IEEE Trans. Software Engineering*, Vol. SE-10, 1984.

[7] D. N. Card and R. L Glass, “Measuring Software Design Quality,” Prentice-Hall, 1990.

[8] IEEE Standard for Software Reviews, IEEE Std 1028-2008, 1998.

[9] “Software Engineering Standards,” IEEE Computer Society, 1994.

[10] I. Somerville, Software Engineering,“ Addison-Wesley, 2001.

[11] R. Snee and R. Horel, “Leading Six Sigma,” Prentice-Hall, 2003.

[12] N. G. Leveson, “Safeware: System Safety and Computer,”

Addison-Wesley, 1995.
[13] Isabel Lopes Margarido, "Classification of Defect Types in Requirements Specifications: Literature Review, Proposal and Assessment," *6th Iberian Conference on Information Systems and Technologies (CISTI 2011)*, 2011.
[14] "Software Risk Abatement," AFCS/AFLC Pamphlet 800-45, U.S. Air Force, 1988.
[15] B, W. Boehm, "Software Risk Management," IEEE Computer Society Press, 1989.
[16] P. Drucker, "Management," W. H. Heinemann, 1975.
[17] T. Gilb, "Principles of Software Engineering Management," Addison-Wesley, 1988.
[18] E. M. Hall, "Managing Risk: Methods for Software Systems Development Risks," CMU/SEI-94-TR-14, Softwarre Engineering Institute, 1994.
[19] D. W. Karolak, "Software Engineering Risk Management," IEEE Computer Society Pess, 1996.

이 은 서
https://orcid.org/0000-0002-7637-3036
e-mail : eslee@anu.ac.kr
2001 ~ 현 재 ISO/IEC 15504 국제 선임 심사원
2004년 중앙대학교 컴퓨터공학과(박사)
2008년 ~ 현 재 안동대학교 컴퓨터공학과 교수
관심분야 : CBD, Formal Method, Quality Model, SPI(Defect Analysis)