

논문 2015-52-5-25

사물인터넷 관련 버그 정정을 위한 버그리포트 속성 분석

(Bug Reports Attribute Analysis for Fixing The Bug on The Internet of Things)

권기문*, 정성순**

(Ki Mun Knon and Seong Soon Jeong[©])

요약

최근 사물인터넷과 관련된 연구와 산업이 급속히 발전하고 있다. 사물인터넷과 관련된 소프트웨어 개발 및 유지보수 활동에서 버그 정정은 큰 비중을 차지하는 활동이다. 본 논문에서는 사물인터넷과 관련된 버그를 정정하는데 소요되는 시간을 분석함으로써 버그 정정 시간에 영향을 미칠 수 있는 속성이 무엇인지 분석한다. 버그 리포트가 제공하는 속성 정보에 따라 k-NN 분류 방법을 사용하여 버그 리포트를 분류하고 유사한 속성을 가진 버그 리포트를 선별한다. 유사한 버그 리포트의 버그 정정 시간을 계산하여 새로운 버그의 정정 시간을 예측한다. 예측 정확도에 따라 버그 정정 시간에 영향을 미치는 속성 중 운영체제(os), 컴포넌트, 리포터, 할당자(assignee) 속성을 사용했을 때 버그 정정 시간 예측에 가장 좋은 정확도를 나타냈다.

Abstract

Nowadays, research and industry on the internet of things is rapidly developing. Bug fixed field of the Software development related internet of things is a very important things. In this study, we analyze the properties that can affect what the bug fix-time by analyzing the time required to fix a bug associated with the Internet of Things. Using the k-NN classification method based on the attribute information to be classified as bug reports. Extracts a bug report based on the results of a similar property. Bug fixed by calculating the time of a similar bug report predicts the fix-time for new bugs. Depending on the prediction of the properties that affect the bug correction time, the properties of os, component, reporter, and assignee showed the best prediction accuracy.

Keywords : 사물인터넷, 버그 리포트, 속성 분석, 버그 정정 시간, 소프트웨어 유지보수

I. 서론

최근 통신기술이 발달함에 따라 사물인터넷(IoT)이 큰 이슈가 되고 있다. 사물인터넷이라 함은 인터넷과 같은 구조상에서 고유하게 식별 가능한 객체(사물)들 간에 가상으로 연결된 서비스를 의미한다^[11~12]. 사물인터넷과 관련하여 여러 단체와 기업에서 연구가 진행되고 있으며, 오픈 소스 프로젝트인 이클립스(Eclipse)에서도 사물인터넷과 관련된 프로젝트가 진행되고 있다. 사물인터넷이 연구되고 실생활에 사용 가능한 제품이

* 정회원, 서울시립대학교 컴퓨터과학과
(Department of Computer Science and Engineering,
University of Seoul)

** 정회원, 동양미래대학교 전기전자통신공학부 정보통신과
(Department of Information & Communications,
Dongyang Mirae University)

© Corresponding Author(E-mail: ssjeong@dongyang.ac.kr)

※ 본 연구는 동양미래대학교에서 2014년 학술연구비 지원을 받아 수행되었습니다.

Received ; March 1, 2014 Revised ; March 25, 2015

Accepted ; April 20, 2015

생산되기 위해서는 관련 소프트웨어의 개발과 유지보수는 필수적 요소이다. 소프트웨어 개발과 유지보수 단계에서 버그는 항상 발견되며, 발견된 버그에 대해 수정하는 작업은 지속적으로 이루어져야 한다. 그러므로 사물인터넷과 관련된 소프트웨어의 개발시간 및 인력, 비용관리 측면에서 버그 정정은 매우 중요한 부분이다. 기존의 소프트웨어 버그 정정 시간 관련 연구들은 주로 정정의 긴급성(빠른 정정, 느린 정정)을 추정^[1~5]하고 있으며 사물인터넷(IoT)만을 대상으로 한 연구는 아직 초기 단계이다. 그리고 현재까지 버그 정정 시간을 예측하는 연구 결과들은 일관성 있는 결과를 보여주지 못했다. 또한 과거 연구에서는 버그 정정의 긴급성만을 예측하거나 실제 버그정정 시간 예측할 경우 신뢰성 있는 결과를 보여주지 못했다. 실험에 사용된 속성이 일관성이 없고 실험에 사용된 프로젝트마다 상이하게 나타나고 있다. Weiss는 버그리포트의 설명과 요약 2가지만을 사용하였고^[6], Ansan은 프로그램 파일과 개발자의 노력(effort) 정보 2가지 속성을 사용하였다^[7]. Zhang은 5가지 속성(reporter, 심각도, 우선순위, ESC, 요약)을 사용하였다^[8]. 사용된 속성의 개수도 2개부터 5개까지 기존 연구에서 다르게 사용되고 있다. 따라서 본 논문에서는 이러한 문제를 해결하기 위하여 사물인터넷(IoT)과 관련된 버그 리포트를 이용하여 버그정정 시간 예측에 사용가능한 주요한 속성을 선별하는 방법을 제안하고 모의실험을 통해 성능을 평가한다.

II. 배경 지식

1. 사물인터넷(IoT)

지금까지 정보통신에서 의사소통은 사람 대 사람으로 이루어졌다. 사람들은 직접 컴퓨터나 휴대용 단말기를 사용하여 인터넷에 접속하여 의사소통을 하였다. 그러나 사물인터넷의 발달로 사람뿐 아니라 사물들도 인터넷에 연결될 것이고, 사람의 직접적인 개입 없이 사물들간의 의사소통도 이루어질 것이다. 이는 사물인터넷이 서로 다른 객체를 제어하는 프로세스와 연결된 객체의 통합된 정보 구조를 가지기 때문이다. 여러 산업분야와 오픈 소스 분야에서 사물인터넷과 관련된 연구가 진행되고 있다. 현재 오픈 소스 프로젝트 이클립스에서는 사물인터넷과 관련하여 14개의 프로젝트(<https://projects.eclipse.org/technology-type/internet-t>

표 1. Eclipse IoT Project

Table 1. Eclipse IoT Project.

| Eclipse Project (Internet of Things) | |
|--------------------------------------|---------------------------------|
| 1 | wakaama |
| 2 | californium (Cf) CoAP Framework |
| 3 | OM2M |
| 4 | Krikkit |
| 5 | Kura |
| 6 | Eclipse SmartHome |
| 7 | Mosquitto |
| 8 | Concierge |
| 9 | Ponte |
| 10 | Eclipse SCADA |
| 11 | koneki |
| 12 | mihini |
| 13 | paho |
| 14 | eclipse communication framework |

hings)를 진행하고 있다(표 1).

2. 버그리포트

버그리포트는 컴퓨터 소프트웨어 프로그램이 개발 또는 운영되는 과정에서 발견되는 오류 및 에러 사항에 대해서 다양한 정보를 기록한 내용이다. 그림 1은 실제 오픈 소스 이클립스 프로젝트 ECF의 버그리포트이다.

버그리포트에는 Status(현재 상태), Product(제품), 컴포넌트(Component), 버전(Version), 하드웨어(Hardware), 중요도(Priority, Severity), 할당자(Assignee) 등 해당 버그와 관련된 여러 정보들이 포함되어 있다. 또한, 해당 버그의 문제점이나 추가 정보를 기술한 설명(Description)과 코멘트(Comments)에 대한

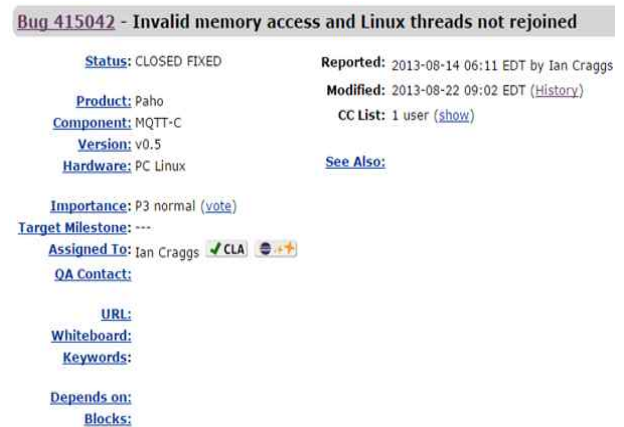


그림 1. 버그 리포트(Eclipse #415042)

Fig. 1. Bug Report(Eclipse #415042).

| Who | When | What | Removed | Added |
|---------|-------------------------|------------|----------|----------|
| icraggs | 2013-08-14 06:11:46 EDT | Status | NEW | RESOLVED |
| | | Resolution | --- | FIXED |
| icraggs | 2013-08-22 09:02:00 EDT | Status | RESOLVED | CLOSED |

그림 2. 버그 이력(Eclipse #415042)
Fig. 2. Bug History(Eclipse #415042).

정보도 포함된다. 그림 2는 버그가 발견되어 버그리포트가 생성될 시점부터 버그정정이 완료되어 종료될 때까지 이력정보가 기록된 내용이다. 여기에는 할당자(assignee)의 변경 정보, 해결되는 과정을 볼 수 있는 status(상태)의 변경 이력을 알 수 있다.

III. 주요 속성 분석 방법(버그 정정 시간 예측에 사용될 수 있는 주요 속성 분석 방법)

본 논문에서는 버그리포트에서 제공되는 속성들을 대상으로 k-NN(k-Nearest Neighbor) 분류 방법을 적용하여 버그 정정 시간 예측에 사용될 수 있는 주요 속성을 선별한다.

1. 개요

그림 3은 본 논문에서 제안하는 방법론의 전체적인 흐름을 나타낸다. 버그리포트 데이터의 속성정보에 k-NN 분류방법을 활용하여 각각의 특징으로 분류하고 유사한 특징을 지닌 버그리포트들을 대상으로 버그 정정시간을 예측한다. 그리고 k-NN 분류에 사용된 속성들 중 예측 정확도에 따라 버그 정정시간 예측에 영향을 미치는 주요 속성을 선별한다.

새로운 버그가 식별되면 버그리포트에 포함된 컴포

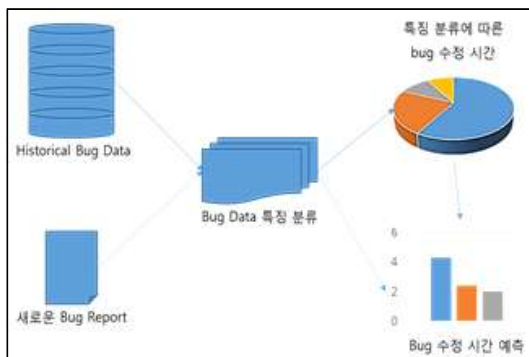


그림 3. 전반적인 도식도
Fig. 3. Overview of our approach.

넌트, 리포터, 할당자, 우선순위, 심각도 등의 속성에 대해 k-NN 분류 방법을 적용하여 동일한 특징을 가진 기존 버그 리포트들의 그룹을 식별한다. 분류된 동일한 특징에 따라 기존 버그들의 정정 시간을 바탕으로 새로운 버그의 정정 시간을 예측한다.

2. k-NN(k-Nearest Neighbor)분류

k-NN 분류방법을 활용하여 동일한 속성을 가진 버그리포트들을 분류한다. k-NN 방법을 적용하기 위해, 우선 6가지 속성(컴포넌트, 할당자, 리포터, 운영체제, 우선순위, 심각도)을 사용하여 버그 리포트 사이의 거리(유사도의 반대)를 계산한다. 컴포넌트, 할당자, 리포터, 운영체제는 명목형 속성으로서 단일 속성 값을 가지고 있다. 그러나 다중 속성 값을 가지는 속성도 있을 수 있기 때문에 버그 리포트 사이의 거리는 다음 식(1)과 같이 정의한다^[9]. 이 거리는 버그 리포트 속성들의 거리에 가중치 합계로 정의한다.

$$D(a,b) = \sum w_i d(a_i, b_i) / \sum w_i \quad (1)$$

- a 와 b 는 버그 리포트이다.
- a_i 와 b_i 는 버그 리포트의 속성이다.
- w_i 는 가중치다.

버그 리포트 속성들 간의 거리는 공통적인 값과 다른 속성과 구분될 수 있는 값을 측정하여 서로 다른 가중치를 적용하여 계산한다. 본 연구의 실험에서는 모두 동일한 값의 가중치를 주고 계산하도록 한다. 속성이 버그 정정 시간 예측에 미치는 상관관계에 따라 가중치를 조정할 수 있다. 거리의 값은 0과 1사이를 가진다.

$$d(a_i, b_i) = 1 - f(A \cap B) / [f(A \cap B) + f(A - B) + f(B - A)] \quad (2)$$

- A 와 B 는 각 속성 a_i 와 b_i 와 관련된 값의 집합이다.
- f 는 집합 원소의 개수이다.

표 2. 우선순위 거리
Table 2. Distance of priority.

| | P1 | P2 | P3 | P4 | P5 |
|----|------|------|------|------|------|
| P1 | 0 | 0.25 | 0.5 | 0.75 | 1 |
| P2 | 0.25 | 0 | 0.25 | 0.5 | 0.75 |
| P3 | 0.5 | 0.25 | 0 | 0.25 | 0.5 |
| P4 | 0.75 | 0.5 | 0.25 | 0 | 0.25 |
| P5 | 1 | 0.75 | 0.5 | 0.25 | 0 |

우선순위와 심각도는 각 속성 값이 일정한 거리만큼 떨어져있는 것을 가정한다. 예를 들어, 5개 단계로 구성된 우선순위는 각 속성들 간의 거리를 0.25로 계산한다.

IV. 실험

1. 실험 Data

실험 대상 자료는 이클립스 사물인터넷과 관련된 프로젝트 중 가장 많은 버그리포트를 가지고 있는 3가지 제품(ECF, Koneki, Paho)에서 수집된 버그 리포트를 실험을 위한 데이터로 사용한다. 상태가 종료(Closed)되었거나 검증확인(Verified) 단계인 것들과 resolution이 FIXED인 것을 대상으로 Eclipse bugzilla (<https://bugs.eclipse.org/bugs/>)로부터 데이터를 수집하였으며, 총 215개의 버그 리포트 데이터를 사용하였다. 다른 제품의 경우 사용될 수 있는 버그리포트의 개수가 상당히 적어 실험에서 제외하였다.

2. 평가 방법

실험 평가 방법으로 실제 버그 정정 시간과 버그 정정 예측 시간과의 차이를 계산한 잔차(Residual)를 사용한다. 버그 정정 시간은 버그가 open(제기)된 시점으로부터 최종 정정된 시점까지의 시간으로 계산한다.

$$Residual = |Actual Time - Predicted Time| \quad (3)$$

잔차는 r_i 로 표기하며 실제 버그 정정 시간(Actual Time)과 예측 버그 정정 시간(Predicted Time)의 차이를 절대값으로 계산한다. 계산된 잔차를 사용하여 오차 범위를 측정할 수 있는 MRE(Magnitude Relative Error)를 측정한다. 잔차를 실제 버그 정정 시간으로 나누어줌으로써 개별 MRE를 구할 수 있다. MMRE (Mean Magnitude Relative Error)는 MRE의 평균이다. 이 측정 방법은 잔차의 합과 관련성이 크고 공정한 통계량으로 사용된다^[10].

$$MMRE = \sum \frac{r_i}{A_i} / n = \sum \frac{|A_i - P_i|}{A_i} / n \quad (4)$$

Pred(Percentage of predictions)은 허용 오차 범위를 x 로 설정함으로써 추정 값에 대한 정확성을 비율로 제공한다. 시간 예측과 관련된 기존 연구에서 일반적으로 Pred(25)와 Pred(50)을 사용하고 있다^[6,7]. 본 연구에서

는 차후 다른 연구와의 비교를 위해 Pred(50)을 사용하기로 한다.

$$Pred(x) = \frac{\{i|MRE < x/100\}}{n} = \frac{\{i|r_i/A_i < x/100\}}{n} \quad (5)$$

3. 실험 결과(성능 평가)

버그리포트의 속성을 대상으로 k-NN 방법을 사용하여 특징을 분류하고 버그 정정 시간을 예측하는 데 사용될 수 있는 속성을 선별하는 방법에 대해 실험을 실시하였다. 한 가지 속성을 사용하여 버그 정정 시간을 예측한 실험부터 6가지 속성의 조합으로 버그 정정 시간을 예측한 실험을 진행하였다.

그림 4는 한 가지 속성만을 사용하여 버그 정정 시간을 예측한 Pre(50) 결과로 상위 4개만을 보여준다. ECF에서는 리포터가 가장 좋은 예측 결과를 보여주고 있으며, Koneki는 운영체제, Paho는 컴포넌트가 가장 좋은 예측 결과를 보여준다. 각 제품마다 가장 좋은 결과를 보여주는 속성이 모두 다르다.

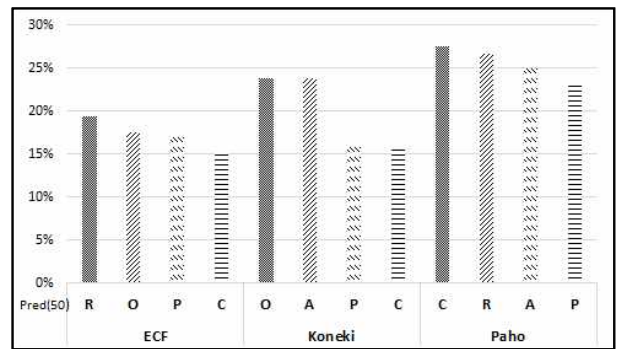


그림 4. 1가지 속성 Pred(50) 결과
Fig. 4. Pred(50) result of 1 Attribute.

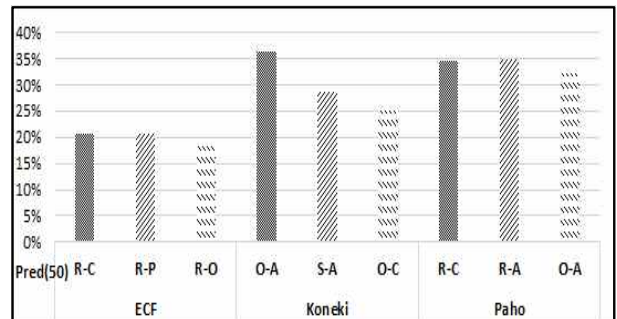


그림 5. 2가지 속성 조합 Pred(50) 결과
Fig. 5. Pred(50) result of 2 Attribute.

그림 5는 2가지 속성의 조합으로 버그 정정시간을 예측한 Pred(50) 결과이며, 그림 6은 3가지 속성을 조합하여 버그 정정 시간을 예측한 Pred(50) 결과다.

2가지 속성을 조합하였을 때 ECF는 리포터와 컴포넌트의 조합이 약 21로 가장 좋은 결과를 보여주고 있으며, Koneki는 운영체제와 할당자의 조합이 약 36으로 가장 좋은 결과를 보여주고 있다. Paho는 리포터와 컴포넌트 조합과 리포터, 할당자 조합이 약 35로 가장 좋은 결과를 보여준다. 3가지 속성을 조합하였을 때는 2가지 속성에서 가장 좋은 결과를 가진 속성에 프로젝트에 따라 운영체제 또는 컴포넌트를 추가하여 버그 정정 시간을 예측하였을 때 가장 좋은 결과를 보여준다. 예측 성능 역시 2가지를 조합했을 때보다 3가지를 조합하여 사용했을 때 조금 더 좋아지는 것을 알 수 있다.

그림 7은 4가지 속성의 조합으로 버그 정정시간을 예측한 Pred(50) 결과이며, 그림 8은 5가지 속성을 조합하여 버그 정정 시간을 예측한 Pred(50) 결과를 보여준다. 4가지 속성을 조합하였을 때 ECF와 Koneki는 5가지 속성을 조합하였을 때보다 좋은 예측 결과를 보여주고 있으나 paho는 4가지와 5가지 속성을 조합하였을 때 약

47로 동일한 예측 결과를 보여준다.

그림 9는 6가지 속성 모두를 조합하여 버그 정정 시간을 예측한 결과를 보여준다. 모든 속성을 조합하여 버그 정정 시간을 예측한 결과가 5가지 속성을 조합하여 예측한 결과보다 다소 성능이 떨어지는 것을 알 수 있다.

그림 10은 한 가지 속성부터 모든 속성의 조합을 사용하였을 때 가장 좋은 결과를 나타낸 것을 비교한 그래프다. 한 가지 속성을 사용했을 때보다 2가지 3가지

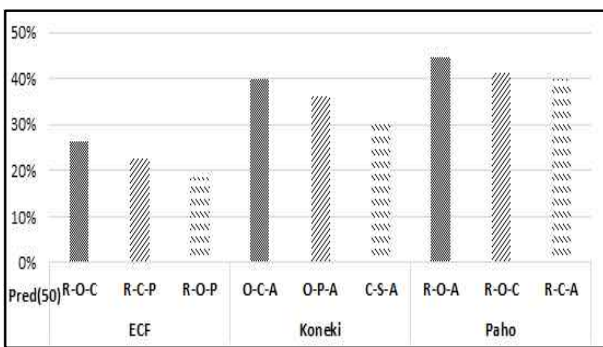


그림 6. 3가지 속성 조합 Pred(50) 결과
Fig. 6. Pred(50) result of 3 Attribute.

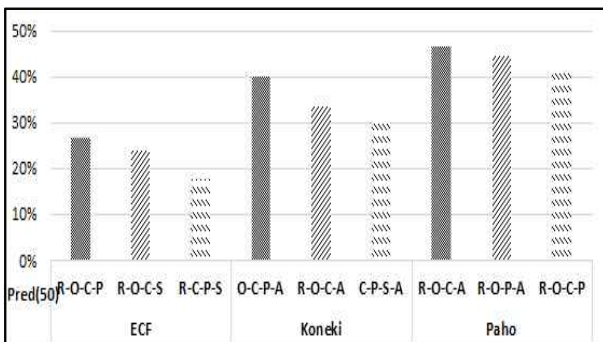


그림 7. 4가지 속성 조합 Pred(50) 결과
Fig. 7. Pred(50) result of 4 Attribute.

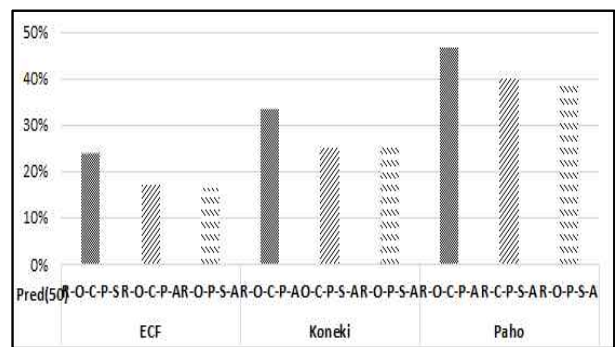


그림 8. 5가지 속성 조합 Pred(50) 결과
Fig. 8. Pred(50) result of 5 Attribute.

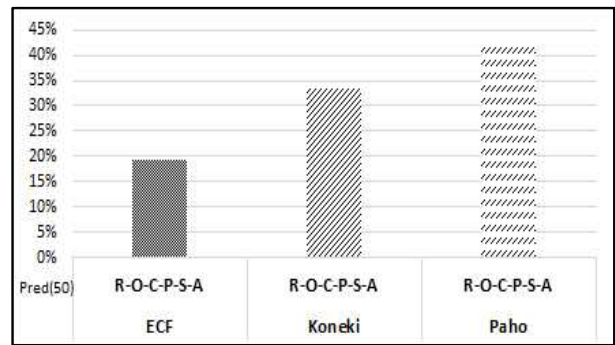


그림 9. 6가지 속성 조합 Pred(50) 결과
Fig. 9. Pred(50) result of 6 Attribute.

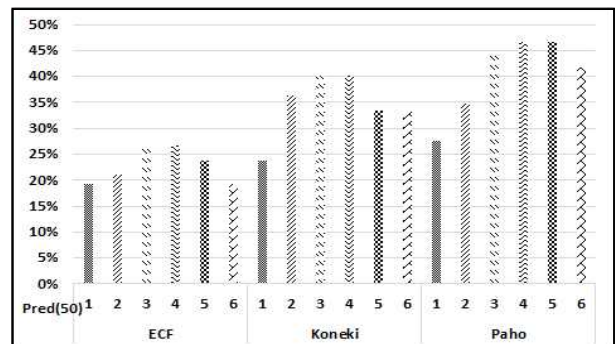


그림 10. 사용된 속성 개수에 따른 Pred(50) 비교 결과
Fig. 10. Pred(50) compare result(number of attribute).

속성의 조합을 사용하였을 때 예측 결과에 대한 성능이 향상되고 있다. 그러나 3~5가지 속성의 조합을 사용하였을 때를 기점으로 모든 속성의 조합을 사용하였을 때 버그 정정 시간 예측 결과 성능은 다시 저하된다.

따라서 모든 프로젝트(ECF, Koneki, Paho)에서 4가지 속성을 조합하여 버그 정정 시간을 예측했을 때 가장 좋은 Pred(50) 결과를 보여주는 것을 알 수 있다.

또한, ECF는 리포터, 운영체제, 컴포넌트 속성을 사용하였을 때 가장 좋은 결과를 보여주고 koneki는 운영체제, 컴포넌트, 할당자 속성을 사용하였을 때 가장 좋은 결과를 보여준다. Paho는 리포터, 운영체제, 컴포넌트, 할당자 속성을 사용하였을 때 가장 좋은 결과를 보여주고 있다. 가장 좋은 결과를 보여주는 속성을 보면 모든 프로젝트에서 공통적으로 운영체제와 컴포넌트 속성을 사용하고 있으며, 프로젝트에 따라 리포터와 할당자가 포함되었을 때 성능을 향상시키는 것을 알 수 있다.

V. 결 론

본 논문에서는 사물인터넷과 관련된 소프트웨어의 개발과 유지보수에서 버그 리포트를 k-NN 분류 방법을 활용하여 특징을 분류하고 버그 정정 시간에 영향을 미치는 속성을 선별하는 방법을 제안하였다. 제안한 방법을 평가하기 위해 이클립스 버그 리포트를 대상으로 실험과 분석을 실시하였다. 실험에서 본 방법이 버그 정정 시간을 예측하는데 영향을 미칠 수 있는 속성으로 운영체제와 컴포넌트가 가장 유효하다는 것을 알 수 있었다. 다음으로 리포터와 할당자 속성이 성능 향상에 도움이 될 수 있는 것을 알 수 있었다. 또한, 실험에 사용된 3가지 프로젝트 모두 4가지 속성을 조합하여 버그 정정 시간을 예측하는 것이 가장 좋은 결과를 보여줄 수 있다는 것을 발견하였다. 본 연구의 실험에서는 운영체제, 컴포넌트, 리포터, 할당자 4가지 속성을 활용한 버그 정정 시간 예측이 가장 효과적으로 나타났다.

본 논문에서는 사물인터넷과 관련된 오픈 소스(Open Source) 이클립스에서 3가지(ECF, Koneki, Paho) 프로젝트의 버그 리포트를 대상으로 실험하였다. 앞으로 사물인터넷과 관련된 오픈 소스 프로젝트가 아닌 비즈니스 소프트웨어 버그 리포트에 대한 분석과 실험에 대한 연구도 진행되어야 할 필요성이 있다.

REFERENCES

- [1] M. Sharma, M. Kumari, and V.B. Singh, Understanding the meaning of bug attributes and prediction models, In Proc. of 5th Workshop on IBM Collaborative Academia Research Exchange (I-CARE), 2013.
- [2] L. D. Panjer, Predicting Eclipse bug Lifetimes, In Proc. of the Fourth International Workshop on Mining Software Repositories (MSR), pp. 29, 2007.
- [3] S. Kim and J. E. James Whitehead, How long did it take to fix bugs?, In Proc. of the 2006 International Workshop on Mining Software Repositories (MSR), pp. 173-174, 2006.
- [4] E. Giger, M. Pinzger, and H. Gall, Predicting the Fix Time of bugs, In Proc. of International Workshop on Recommendation System for Software Engineering (RSSE), pp. 52-56, 2010.
- [5] A. Mockus, R. T. Fielding, and J. D. Herbsleb, Two case studies of open source software development: Apache and Mozilla, ACM Transaction on Software Engineering and Methodology (TOSEM), 11(3), pp. 309 - 346, 2002
- [6] C. Weiss, R. Premraj, T. Zimmermann, and A. Zeller. How long will it take to fix this bug?, In Proc. of the Fourth International Workshop on Mining Software Repositories (MSR), 2007.
- [7] S.N.Ahsan, J.Ferzund, and F.Wotawa, Program file bug fix effort estimation using machine learning methods for oss, In proceedings of the 21th International Conference on Software Engineering and Knowledge Engineering(SEKE), pp. 129-134, 2009.
- [8] H. Zhang, L. Gong, and S. Versteeg, Predicting bug-Fixing Time: An Empirical Study of Commercial Software Projects, In Proc. of International Conference on Software Engineering (ICSE), 2013, pp. 1042-1051.
- [9] T. W. Ryu, Christoph and C. F. Eick, Similarity measures for multi-valued attributes for database clustering, In Proc. of the Conference on Smart Engineering System Design: Neural Networks, Fuzzy Logic, Evolutionary Programming, Data Mining and Rough Sets (ANNIE'98), 1998.
- [10] B. A. Kitchenham, L. M. Pickard, S. G. MacDonell and M. J. Shepperd, What accuracy statistics really measure, IEE Proceeding - Software, 148(3), pp. 81-85, 2001.

- [11] Seo young seok, Jo seong gyu, "Expectations for the new IOT(the Bluetooth v4.1)", IEIE Journal of The Institute of Electronics and Information Engineers Vol.41, NO.8, pp26-31, Aug 2015.
- [12] Kwon jong won, Kim hee sik, "Domestic Conformity Assessment Trends of the IOT device", IEIE, journal of The Institute of Electronics and Information Engineers Vol.41, NO.8, pp16-25, Aug 2015.

— 저 자 소 개 —



권 기 문(정회원)
2008년 용인대학교 컴퓨터정보처
리/경영학과 학사 졸업
2015년 서울시립대학교 컴퓨터과
학과 석사 졸업
2015년 현재 이베스트투자증권 정
보서비스팀 과장

<주관심분야 : 사물인터넷, 데이터마이닝, 소프트
웨어진화>



정 성 순(정회원)-교신저자
1985년 연세대학교 전자공학과
학사 졸업.
1987년 연세대학교 전자공학과
석사 졸업.
2004년 연세대학교 전기전자공학
박사 졸업.

1987년~1995년 삼성전자 정보통신연구소
선임연구원

1995년~현재 동양미래대학교 전기전자통신공학
부 정보통신공학과 교수

<주관심분야 : 정보통신, 통신공학, 네트워크>