객체지향 메트릭을 이용한 결함 예측 모형의 임계치 설정에 관한 실험

(An Experiment for Determining Threshold of Defect Prediction Models using Object Oriented Metrics)

김 윤 규 † 채 흥 석 ††
(Yun Kyu Kim) (Heung Seok Chae)

요 약 소프트웨어의 결함을 예측하고 검증과 확인 활동을 통하여 효율적인 자원을 관리하기 위하여 많은 연구에서 결함 예측 모형을 제안하고 있다. 하지만 기존의 연구는 예측율이 최대 효과를 보이는 임계치에 결함 예측 모형의 예측율을 평가하고 있다. 이는 측정 시스템의 결함 정보를 알고 있는 가정하에서 평가가 이루어지는 것이기 때문에 실제 결함 정보를 알 수 없는 시스템에서는 최적의 임계치를 결정할 수 없다. 그러므로 임계치 선정의 중요성을 확인하기 위하여 본 연구에서는 결함 예측 모형으로 타 시스템의 결함을 예측하는 비교 실험을 하였다. 실험은 기존에 제안된 3개의 결함 예측 모형과 4개의 시스템을 대상으로 하였고 결함 예측 모형의 임계치별 예측의 정확성을 비교하였다. 실험 결과에서 임계치는 모형의 예측율과 높은 관련이 있었지만 실제 결함 정보가 확인 안 되는 시스템에 대하여 결함을 예측하는 경우에는 임계치를 선정할 수 없음을 확인하였다. 따라서 결함 예측 모형을 타 시스템에 적용하기 위하여 임계치 선정에 관한 추후 연구가 필요함을 확인하였다.

· 본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원 사업의 연구결과로 수행되었음(IITA-2009-(C1090-0902-0032))
· 이 논문은 2009 한국컴퓨터종합학술대회에서 '객체지향 메트릭을 이용한 결함 예측 모형의 임계치 설정에 관한 비교 실험'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 부산대학교 컴퓨터공학과
kim_yunkyu@pusan.ac.kr
†† 정 회 원 : 부산대학교 컴퓨터공학과 교수
hschae@pusan.ac.kr
(Corresponding author임)

논문접수 : 2009년 8월 18일
심사완료 : 2009년 10월 9일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제15권 제12호(2009.12)

키워드 : 객체지향 소프트웨어 메트릭, 결함 예측 모형, 임계치

Abstract To support an efficient management of software verification and validation activities, many defect prediction models have been proposed based on object oriented metrics. In order to apply defect prediction models, we need to determine a threshold value. Because we cannot know actually where defects are, it is difficult to determine threshold. Therefore, we performed a series of experiments to explore the issue of determining a threshold. In the experiments, we applied defect prediction models to other systems different from the system used in building the prediction model. Specifically, we have applied three models - Olague model, Zhou model, and Gyimothy model - to four different systems. As a result, we found that the prediction capabilities varied considerably with a chosen threshold value. Therefore, we need to perform a study on the determination of an appropriate threshold value to improve the applicably of defect prediction models.

Key words : Object-Oriented metrics, Defect prediction models, Threshold

1. 서 론

소프트웨어의 유지 보수 비용을 줄이기 위하여 객체지향 메트릭을 이용한 결함 예측 모형들이 제안되고 있다[1-4]. 결함 예측 모형은 소프트웨어의 결함을 개발 초기 혹은 이후 단계에서 예측하여 한정된 검증 및 확인 활동의 효율적인 관리 위한 목적으로 사용할 수 있다. 하지만 기존의 연구에서는 결함 예측 모형의 임계치 선정에 대한 연구가 부족하다[1-4].

본 논문에서는 기존에 제안된 3개의 결함 예측 모형과 4개의 대규모 시스템을 대상으로 모형의 임계치와 예측율간의 관계를 비교 실험하였다. 결함 예측 모형으로 결함 발생 가능성이 높은 클래스를 예측하려면 우선적으로 모형의 임계치가 선정되어야 한다. 하지만 예측 대상 시스템의 결함 정보를 알 수 없기 때문에 해당 시스템에서 모형의 임계치를 선정하는 것은 어렵다. 따라서 임계치 선정 방법에 의해 모형이 최대 효과를 보이는 예측력과는 차이가 남을 확인하였다.

2. 연구 배경

2.1 로지스틱 회귀 기법

로지스틱 회귀 기법[5]은 특정 사건이 일어날 확률을 이분법적인 방식으로 평가하는 통계적 모델링 기법이다. 소프트웨어의 결함 발생 여부를 분석하기 위하여 기존의 많은 연구에서는 소프트웨어 메트릭을 바탕으로 로지스틱 회귀 기법을 사용하고 있다[1-4,6-8].

식 (1) 로지스틱 회귀식

$$\pi(M_1, M_2, ..., M_n) = \frac{e^{C_0+C_1M_1+...+C_nM_n}}{1+e^{C_0+C_1M_1+...+C_nM_n}}$$

로지스틱 회귀 기법은 일반적으로 식 (1)과 같은 형태로 표현한다. 식 (1)에서 M_i은 독립변수인 객체지향 메트릭 측정값이며 C_i는 M_i의 회귀계수를 의미한다. 종속변수인 π는 해당 클래스에 결함이 발생할 확률을 의미한다. π는 0부터 1까지 범위를 가지며 0에 근접할수록 해당 클래스에 결함이 발생할 가능성은 낮은 것으로 판단하고 1에 근접할수록 높은 것으로 판단한다.

결함 발생 확률을 분석할 경우 주의사항은 절단 값(cutoff value)을 선정하는 문제이다. 절단 값이란 클래스의 결함 발생 여부를 결정하는 기준이 된다. 절단 값 선정에 의한 모형의 예측력과 문제점은 2.2절에 자세히 설명하였다.

2.2 예측율 평가 방법

결함 예측 모형의 예측율을 평가하기 위하여 기존의 연구에서 정밀성, 정확성 그리고 완전성을 사용하고 있다[2,3]. 정밀성은 전체 클래스 중에서 예측 모형에 의하여 정확히 분류된 클래스가 차지하는 비율이다. 정확성은 예측 모형에 의하여 결함으로 분류된 전체 클래스 중에서 실제로도 결함인 클래스가 차지하는 비율이다. 완전성은 실제로 존재하는 총 결함 중에서 예측 모형에서 결함으로 분류한 클래스에 존재하는 결함의 비율이다.

표 1은 결함 예측 모형의 예측율을 계산하기 위한 분류표이다. 표 1의 열은 모형의 절단 값에 의해서 개별 클래스의 결함 발생 여부를 분류한다. 그리고 최적의 예측율을 보이는 절단 값이 결함 예측 모형에서의 임계치가 된다. 표 1의 행은 실제로 클래스의 결함이 발생한 기준으로 모형이 예측한 개별 클래스를 분류한다.

표 1 적합도 분류표

		모형에 의한 예측 결과	
		결함 무 (π < 절단 값)	결함 유 (π ≥ 절단 값)
실제 결함	결함 무	C1	C2
	결함 유	C3(B3)	C4(B4)

표 1의 분류된 영역은 C1부터 C4까지 네 개이다. C는 해당 영역으로 분류된 클래스의 총 개수를 의미하고 B는 해당 영역으로 분류된 클래스가 가지는 결함의 총 개수를 의미한다. 결함 예측 모형의 예측율은 표 1에 의해서 다음과 같이 계산된다.

- 정밀성(Precision): (C1+C4) / (C1+C2+C3+C4)
- 정확성(Correctness): C4 / (C2+C4)
- 완전성(Completness): B4 / (B3+B4)

정확성과 완전성은 절충 관계(trade-off)가 있기 때문에 일반적으로 두 개의 예측율이 교차되는 지점을 결함 예측 모형의 예측율로 평가한다. 그리고 교차 지점인 결함 발생 확률 값이 결함 예측 모형의 임계치로 선정된다.

그림 1에서는 결함 발생 확률의 절단 값 선정에 의해서 변화되는 결함 예측 모형의 예측율을 확인할 수 있다. 그림 1의 세로축은 모형의 예측율을 나타내며 퍼센트 단위이다. 그리고 가로축은 모형의 절단 값인 결함 발생 확률이다. 예를 들어 모형의 절단 값을 0.5로 선정할 경우 해당 모형의 정확성과 완전성은 약 70%가 된다.

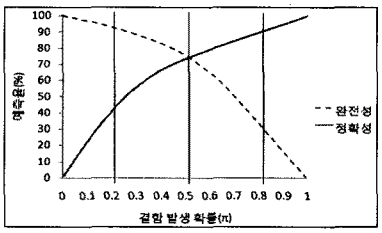


그림 1 정확성과 완전성의 절충 관계

그림 2는 그림 1과 연관하여 낮은 정확성과 낮은 완전성에 대한 문제점을 표현한 것이다. 실선인 원은 예측 대상 시스템 전체를 의미하고 회색영역의 원은 결함 예측 모형에 의해서 결함이 있을 것으로 분류된 클래스들을 의미한다. 그리고 작은 점들은 실제 시스템에 존재하는 결함 클래스를 의미한다. 그림 2의 좌측은 낮은 정확성을 나타내고 우측은 낮은 완전성을 나타낸다.

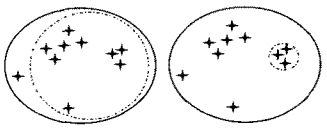


그림 2 낮은 정확성(좌)과 낮은 완전성(우)

- 낮은 정확성

절단 값을 임계치보다 좌측으로 떨어진 값으로 선택했다면 이는 낮은 정확성의 문제를 유발한다. 예를 들어 그림 1에서 결함 발생 확률의 절단 값을 0.2로 선택했다면 완전성은 90%로 높지만 40%의 낮은 정확성을 가진다. 이는 결함 예측 모형이 대부분의 클래스를 결함이 발생할 클래스로 분류하였고 정확성은 40%정도의 매우 낮은 확률을 가지는 것을 의미한다. 그러므로 해당 사항은 결함 예측 모형의 목적인 특정 모듈에 한정된 자원을 투입하여 효율적으로 결함을 예측하는 것과 위배된다.

- 낮은 완전성

절단 값을 임계치보다 우측으로 떨어진 값으로 선택

했다면 이는 낮은 완전성의 문제를 유발한다. 예를 들어 그림 1에서 결함 발생 확률의 절단값을 0.8로 선택했다면 정확성은 90%로 높지만 30%의 낮은 완전성을 가진다. 이는 예측 대상 시스템에 숨겨진 결함을 대부분 찾지 못함을 의미한다. 그러므로 숨겨진 결함을 찾기 위한 별도의 다른 검증 및 확인 작업이 필요하기 때문에 개발 비용의 증가를 초래할 수 있다.

요약하면 기존의 결함 예측 모형을 사용하여 타 시스템의 결함을 예측하기 위해서는 임계치를 선정해야 한다. 하지만 최적의 임계치는 예측 대상 시스템의 결함 정보가 없이는 선정할 수 없다. 따라서 잘못된 임계치 선정은 낮은 정확성 혹은 낮은 완전성 문제를 초래할 수 있다.

3. 실험 설계

3.1 결함 예측 모형

표 2는 대표적인 객체지향 메트릭과 로지스틱 회귀 기법을 사용하여 개발된 결함 예측 모형이다. 표 2에 모형들은 본 비교 실험을 위하여 선정한 모형이다. 결함 예측 모형의 선정 기준은 다음과 같다.

- 결함 예측 모형의 정확성이 60%이상인 모형
- 대규모 시스템을 대상으로 최근에 개발된 모형
- 많은 연구에서 사용하고 있는 CK 메트릭[10]과 로지스틱 회귀 기법으로 개발된 모형[1-4,6,7]

표 2 객체지향 메트릭을 이용한 결함 예측 모형

개발자	Olague[1]	Zhou[3]	Gyimothy[2]
모형이 개발된 환경	Mozilla Rhino 14R3	NASA data set KC1	Mozilla v1.6
개발 언어	Java	C++	C++
규모	58.7 K라인	40 K라인	1 M라인
정확성	85.1%	60.34%	72.6%
완전성	-	79.23%	65.2%
임계치	-	0.865	0.5

단 Olague의 연구에서는 완전성과 임계치가 명시되지 않았다.

3.2 실험 대상 시스템

결함 예측 모형으로 타 시스템에 결함을 예측할 경우 임계치 결정의 어려움을 분석하기 위하여 실험 대상 시스템은 표 3에 정의된 대규모 시스템으로 선정하였다.

표 3의 실험 대상으로는 총 4개의 시스템이 있으며 대규모 공개 소프트웨어이다. 실험 대상의 소스 라인은 약 4만 라인에서 107만 라인까지 다양하다.

실험 대상 시스템의 메트릭 측정 값은 Eclipse 시스템인 경우 Together 상용 도구로 직접 측정하였고 KC1 시스템은 NASA의 MDP 저장소에서 제공 받았으며 jEdit v4.0 시스템은 PROMISE Data 저장소에서 제공 받았다.

표 3 측정 대상 시스템의 요약 정보

실험 대상	Eclipse 2.0[11]	Eclipse 3.0[11]	NASA KC1	jEdit 4.0
전체 클래스 수	6,163	10,929	145	369
결함 클래스 수	2,348	3,019	58	204
총 결함 수	7,205	8,814	542	-
소스라인	590 K	1,077 K	40 K	83 K
개발언어	Java	Java	C++	Java
메트릭 값 수집	Together [12]	Together [12]	NASA [13]	PROMISE [14]
결함 정보 수집	PROMISE [14]	PROMISE [14]	NASA [13]	PROMISE [14]

그리고 결함 정보는 모두 NASA와 PROMISE 저장소에서 제공 받았다.

현재 많은 연구자들이 소프트웨어 메트릭과 결함 예측을 분석하기 위하여 NASA와 PROMISE에서 제공하는 시스템의 메트릭과 결함 정보를 사용하고 있다[3,9,14].

3.3 결함 예측 모형의 임계치 선정에 대한 방법

실제 결함 예측 대상 시스템의 결함 정보는 확인할 수 없기 때문에 모형별 최적의 임계치는 선정할 수 없다. 따라서 현실적으로 다음과 같은 두 가지 방법으로 임계치를 선정할 수 있다.

- 묵시적인 임계치를 선정하는 방법

결함 예측 모형 연구 중에는 임계치를 묵시적인 0.5로 선택한 연구가 있다[2,15]. 특히 로지스틱 회귀 분석[5]은 결함 발생 확률의 절단값을 0.5를 사용하여 결함 예측 모형의 유효성을 확인한다.

- 모형에서 명시된 임계치를 선정하는 방법

결함 예측 모형을 새롭게 제안한 연구에는 모형의 개발 당시 사용된 임계치를 명시하기도 한다. 따라서 모형을 적용하기 위하여 해당 모형 연구에서 명시한 임계치를 동일하게 선정하는 방법을 선택할 수 있다.

4. 실험 결과

4.1 Zhou 모형의 임계치 선정 결과

Zhou 모형인 경우 그림 3에서 묵시적인 0.5 임계치와 모형 개발시에 사용된 명시적 임계를 함께 표시하였다.

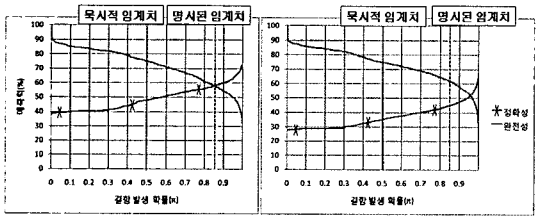
- 묵시적인 0.5를 임계치로 선정한 경우

Eclipse 2.0, Eclipse 3.0 그리고 KC1 시스템에서 낮은 정확성 문제를 보였다. 그리고 jEdit인 경우 최적의 임계치와 예측율이 유사하였다.

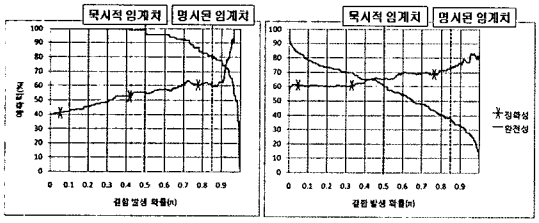
- 명시된 0.865를 임계치로 선정한 경우

Eclipse 3.0과 KC1 시스템에서는 낮은 정확성을 보였고 jEdit4.0에서는 낮은 완전성을 보였다. 그리고 Eclipse 2.0 시스템에서는 최적의 임계치와 명시된 임계치가 거의 일치하였다.

Zhou 모형에서는 4개의 시스템을 대상으로 총 8가지 다른 경우의 임계치 선정 방법이 가능하다. 6가지 경우에서는 기존의 임계치 선정 방법으로는 적절한 영역을 구할 수 없음을 확인하였다. 나머지 2가지 경우에서는 최적의 임계치와 유사한 결과를 보였다.



(a) Eclipse 2.0에 적용 (b) Eclipse 3.0에 적용

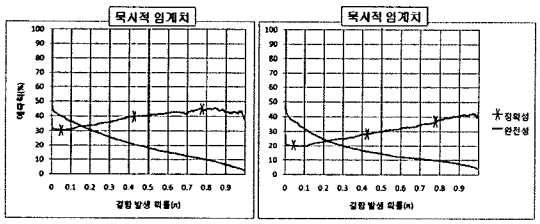


(c) KC1에 적용 (d) jEdit 4.0에 적용

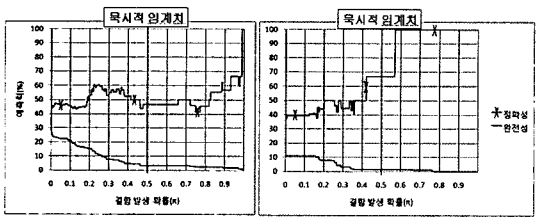
그림 3 Zhou 모형에서 임계치별 예측율 실험 결과

4.2 Olague 모형의 임계치 선정 결과

Olague 모형인 경우 해당 연구에서 임계치를 명시하지 않았기 때문에 그림 4에서 묵시적인 0.5 임계치만을 표시하였다.



(a) Eclipse 2.0에 적용 (b) Eclipse 3.0에 적용



(c) jEdit 4.0에 적용 (d) KC1에 적용

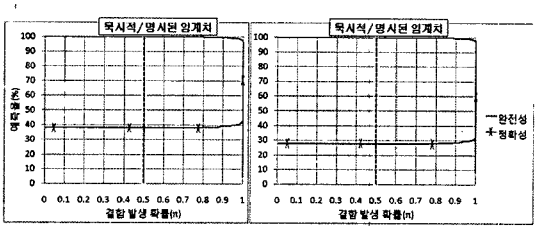
그림 4 Olague 모형에서 임계치별 예측율 실험 결과

• 묵시적인 0.5를 임계치로 선정한 경우

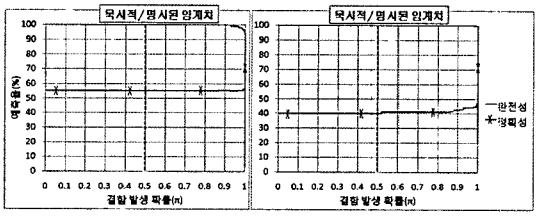
묵시적 임계치는 모든 시스템에서 최적의 임계치와 제법 떨어져 있으며 모형은 낮은 완전성을 보였다. 특히 그림 4의 (c)와 (d)에서는 묵시적인 임계치를 사용할 경우 완전성이 0에 근사하게 되며 이는 예측율의 해석 자체를 불가능하게 만드는 요인이 된다.

4.3 Gyimothy 모형의 임계치 선정 결과

Gyimothy 모형인 경우 명시적 임계치는 0.5이기 때문에 묵시적 임계치와 함께 표시하였다.



(a) Eclipse 2.0에 적용 (b) Eclipse 3.0에 적용



(c) KC1에 적용 (d) jEdit 4.0에 적용

그림 5 Gyimothy 모형에서 임계치별 예측율 실험 결과

• 묵시적, 명시적인 0.5 임계치로 선정한 경우

임계치를 0.5로 선정한 경우 모든 시스템에서 낮은 정확성을 보였다. 그림 5에서 최적의 임계치는 결함 발생 확률 값이 1에 매우 근사한 값이다.

4.4 실험 결과 요약

4개의 대규모 시스템을 대상으로 Olague, Zhou, 그리고 Gyimothy 모형의 임계치 선정에 대한 실험을 총 12번 수행하였다. 실험 결과에서 절단 값의 선택에 의해서 결함 예측 모형의 예측율에 차이가 있음을 확인하였다.

- 최적의 임계치를 선정하는 방법은 예측 대상 시스템의 결함 정보를 사전에 알고 있어야 하기 때문에 이는 현실적으로 임계치 선정이 불가능하다. 또한 12번의 실험에서 임계치는 0에 근사한 값부터 1에 근사한 값까지 다양하게 분포하였다.
- 묵시적인 임계치를 선정하는 것은 12번의 실험 중에서 그림 3(d)의 경우를 제외하고는 모두 낮은 정확성 또는 낮은 완전성 문제가 발생함을 확인하였다.
- 명시적인 임계치를 선정하는 것도 12번의 실험 중에서 그림 3(a)의 경우를 제외하고는 모든 경우에서 낮은 정

확성 또는 낮은 완전성 문제가 발생함을 확인하였다.

5. 관련연구

Watanabe[16]의 연구에서는 개발 언어가 C++과 JAVA인 시스템에서 결함 예측 모형을 개발하고 모형이 두 시스템간에 재사용이 가능한지 비교 실험하였다. 하지만 해당 연구에서는 결함 정보를 알고 있는 가정하에서 예측율을 계산하기 때문에 여전히 임계치를 선정해야 하는 현실적인 문제가 배제되어 있다.

Lessmann[9]의 연구에서는 결함 예측 모형의 개발 기법간의 예측율을 비교 실험 하였다. 비교 실험시에 모형 기법들의 임계치를 결정하는 것이 현실적으로 어렵다고 지적하고 있다. Lessmann은 앞서 제시된 문제점을 해소하기 위하여 모형간의 예측율 평가에서 ROC 기법을 사용하였다. ROC 기법[17]은 모형의 평균적인 예측율을 평가하는 기법이다. 하지만 ROC 기법으로 평가된 모형에서는 결함을 예측 하기 위하여 절단 값을 결정해야 하는 현실적인 문제가 여전히 존재한다.

6. 결론 및 향후 연구 과제

객체지향 메트릭을 이용하여 소프트웨어의 결함을 예측하는 연구들이 많이 진행되고 있다. 하지만 기존의 연구에서는 결함 예측 모형으로 다른 시스템에 존재하는 결함을 예측할 경우 임계치 결정에 대한 논의가 미흡하다.

본 논문에서는 대규모 시스템을 대상으로 기존의 결함 예측 모형을 타 시스템에 적용하여 예측율을 비교 실험하였다. 실험 결과에서는 결함 예측 모형의 임계치의 결정에 의해 모형의 예측율에 차이가 있음을 확인하였다.

향후 연구에서는 측정 대상 시스템의 메트릭 분포를 분석하여 임계치를 선정하는 방법을 연구할 것이다.

참 고 문 헌

[1] H.M. Olague, L.H. Etzkorn, S. Gholston and S. Quattlebaum, "Empirical validation of three software metrics suites to predict fault-proneness of object-oriented classes developed using highly iterative or agile software development processes," *IEEE Transactions on Software Engineering*, vol. 33, no.6, pp.402-419, 2007.

[2] T. Gyimothy, R. Ferenc and I. Siket, "Empirical validation of object-oriented metrics on open source software for fault prediction," *IEEE Transactions on Software Engineering*, vol.31, no.10, pp.897-910, 2005.

[3] Y. Zhou and H. Leung, "Empirical analysis of object-oriented design metrics for predicting high and low severity faults," *IEEE Transactions on Software Engineering*, vol.32, no.10, pp.771-789, 2006.

[4] A. Marcus, D. Poshyvanyk and R. Ferenc, "Using the conceptual cohesion of classes for fault prediction in object-oriented systems," *IEEE Transactions on Software Engineering*, vol.34, no.2, pp. 287-300, 2008.

[5] SW. Menard, Applied logistic regression analysis, Sage Publications Inc, pp.128, 2002.

[6] LC. Briand, WL. Melo and J. Wust, "Assessing the applicability of fault-proneness models across object-oriented software projects," *IEEE Transactions on Software Engineering*, vol.28, no.7, pp. 706-720, 2002.

[7] K.El. Emam, S. Benlarbi, N. Goel and SN. Rai, "The confounding effect of class size on the validity ofobject-oriented metrics," *IEEE Transactions on Software Engineering*, vol.27, no.7, pp. 630-650, 2001.

[8] VR.Basili, LC. Briand and WL.Melo, "A validation of object-oriented design metrics as qualityindicators," *IEEE Transactions on Software Engineering*, vol.22, no.10, pp.751-761, 1996.

[9] S. Lessmann, B. Baesens, C. Mues and S. Pietsch, "Benchmarking classification models for software defect prediction: A proposed framework and novel findings," *IEEE Transactions on Software Engineering*, vol.34, no.4, pp.485-496, 2008.

[10] SR. Chidamber, CF. Kemerer and C. MIT, "A metrics suite for object oriented design," *IEEE Transactions on Software Engineering*, vol.20, no. 6, pp.476-493, 1994.

[11] Eclipse., eclipse project archived downloads, http://archive.eclipse.org/eclipse/downloads.

[12] Borland., Together 2007, http://www.borland.com/us/products/together/index.html.

[13] NASA IV&V FACILITY., Metrics Data Program, http://mdp.ivv.nasa.gov/repository.html.

[14] PROMISE., PROMISE data sets, http://promisedata.org/.

[15] T. Zimmermann, R. Premraj and A. Zeller, "Predicting defects for eclipse," in *Predictor Models in Software Engineering(PROMISE'07)*: ICSE Workshops 2007, International Workshop on, pp. 9-15, 2007.

[16] S. Watanabe, H. Kaiya and K. Kaijiri, "Adapting a fault prediction model to allow inter languagereuse," in *Proceedings of the 4th international workshop on Predictor models in software engineering*, pp.19-24, 2008.

[17] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol.27, no.8, pp. 861-874, 2006.