구성가능한 소프트웨어 시스템의 시험 커버리지 측정 연구

한수빈*, 고서연*, 김태영*, 이지현*
*전북대학교 소프트웨어공학과
hanidiot@gmail.com, tiduscn@jbnu.ac.kr, wareengineer@gmail.com, jihyun30@jbnu.ac.kr

A Study on Test Coverage Measurement for Configurable Software System

Soobin Han*, Seoyeon Go*, Taeyoung Kim*, Jihyun Lee*
*Dept. of Software Engineering, Jeonbuk National University

요 약

SPL 방법론을 적용하여 개발한 제품군 시험은 모든 제품에서 사용되는 공통 부분과 일부 또는 단일 제품에서만 사용되는 가변 부분을 종합적으로 고려해야 하기 때문에 단일 소프트웨어 시험과는 상당히 다르다. 시험 커버리지는 작성된 시험에 대한 적절성을 측정하는 데에 사용되는 동시에 적절한 시험을 작성하기 위한 가이드로 사용되기 때문에 중요하다. SPL 시험에서 시험 커버리지 측정은 제품군을 구성하는 멤버제품 별로 측정될 수도 있지만, 이는 재사용을 기반으로 중복된 개발 관련 활동의 최소화를 지향하는 SPL 의 원칙에 맞지 않다. 따라서 개별 제품이 아닌 SPL 수준에서 시험 커버리지 기준을 측정하고 시험의 적절성을 평가하기 위해서는 다른 방법이 필요하다. 이 논문에서는 구성가능한 소프트웨어 시스템(highly configurable software system)에 SPL 시험 방법을 적용하여 SPL 기반 제품군을 위한 시험 커버리지의 측정 방법을 제안하고 실험의 수행 결과를 기술하여 제안한 방법의 적절성을 검증한다.

1. 서론

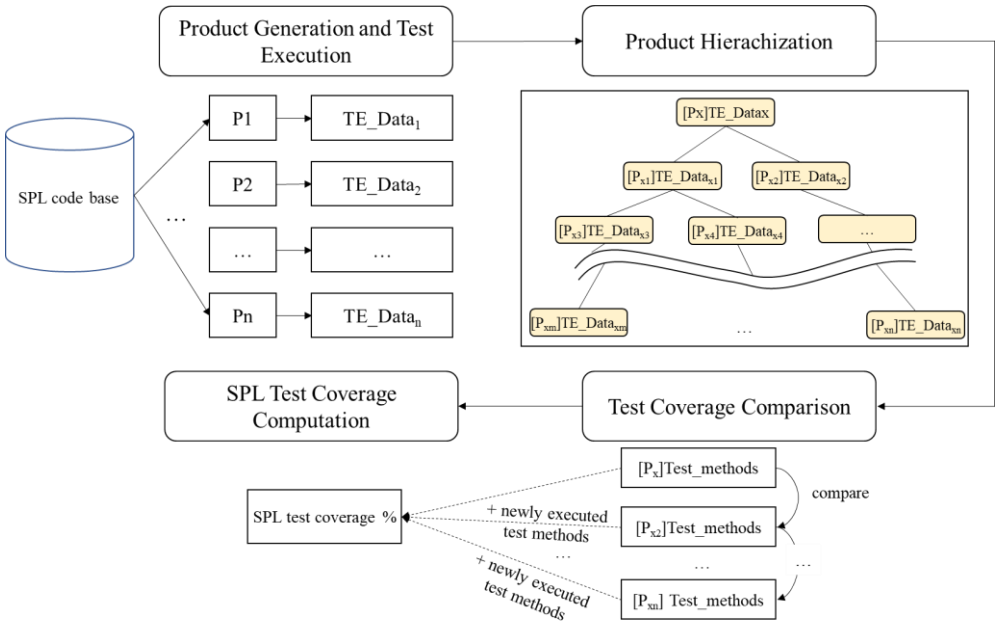
최근 기업들은 단일 제품 보다는 유사한 제품들을 일련의 제품군으로 생산하고 관리함으로써 고객의 다양한 니즈를 충족시키기 위해서 노력한다. SPL(Software Product Line)은 유사한 제품군을 개발하고, 유지할 수 있도록 지원하는 개발 방법론이다[1]. SPL 방법론을 적용하여 개발한 제품군에 대한 시험은 제품군 전체에 공통인 부분과 일부 또는 단일 제품에서만 사용하는 부분 등을 고려해야 하기 때문에 단일 소프트웨어 시험과는 상당히 다르다. 시험 커버리지 또한 마찬가지이다. SPL 시험 커버리지 측정은 제품군의 멤버제품 별로 측정될 수도 있지만, 이는 재사용과 중복된 개발 관련 활동의 최소화를 지향하는 SPL 의 원칙에 맞지 않다. 그러나 SPL 수준에서의 시험 커버리지 측정 방법은 아직 연구되지 않았다[2].

따라서 개별 제품이 아닌 SPL 수준에서 시험 커버리지 기준을 측정하고 시험의 적절성을 평가하기 위한 방법이 필요하다. 이 논문에서는 SPL 시험 방법을 구성가능한 소프트웨어 시스템(highly configurable software system)에 적용하고 커버리지를 측정하는 방법을 제안한다. 방법의 적절성을 검증하기 위해 실험을 수행한 결과를 기술한다.

2. SPL 커버리지 측정 방법

단일 제품 시험의 시험 커버리지 측정 도구인 JaCoCo (https://www.jacoco.org)는 명령(instruction), 라인(line), 브랜치(branch), 메소드(method) 단위로 커버리지를 추적한다. JaCoCo 로 시험 커버리지를 측정한 결과 데이터는 특정 시험 단위에 대한 시험 여부만을 보여주기 때문에 이를 SPL 시험에 그대로 적용하여 SPL 코드베이스(code base)로부터 도출한 제품들을 시험하면, 특정 시험 단위가 어떠한 피처 집합으로 구성된 제품에 의해서 커버되었는지를 파악할 수 없게 된다. 이는 제품을 구성하는 피처 집합이 서로 다를 때, 시험을 수행한 결과로부터 서로 다른 피처 집합에 따라 각 시험이 새로이 커버한 부분을 추적하기 어렵게 만들고 SPL 수준의 커버리지를 측정하는 데에 있어서 큰 제약이 된다.

따라서 본 연구에서는, SPL 수준의 커버리지 측정을 위해 멤버제품의 커버리지가 아닌 멤버제품에 실행된 시험의 최소 단위인 원자단위 시험(atomic test)



(그림 1) SPL 시험 커버리지 측정 개요

들의 커버리지를 추적한다. JUnit 프레임워크의 경우 시험 메소드가 원자단위 시험이다. SPL 커버리지 측정은 정해진 수의 제품들을 생성하고 각 제품에 시험을 실행한다.

SPL 시험 커버리지 측정 방법은 다음과 같다:

- 제품 생성 및 시험 실행: (그림 1)의 왼편과 같이 먼저 코드베이스(SPL code base)로부터 제품들을 생성(instantiation)하고 제품의 시험케이스를 실행한다.
- 제품 계층화: (그림 1)의 오른편과 같이 각 제품을 구성하는 피처들의 수와 피처 집합 간의 포함관계를 고려하여 제품 계층 그래프(product hierarchy graph)를 구성한다.
- 시험 커버리지 비교: 루트 제품을 시작으로 시험 실행 데이터로부터 원자단위 시험들의 커버리지를 추적한다. 부모와 자식 제품들에서 실행된 원자단위 시험들의 커버리지 데이터를 비교하여 일련의 제품 시험이 커버한 시험 커버리지를 누적한다.
- 시험 커버리지 계산: 총 SPL 코드베이스의 라인 수 대비 SPL 시험이 실행한 전체 라인수로 최종 SPL 시험 커버리지 값을 계산한다.

3. 실험 및 결과

실험 대상으로 사용된 제품군의 목록과 세부 정보는 <표 1>과 같다[3]. 이 제품군은 구성가능 소프트웨어 시스템으로부터 생성되며, 실험에서 사용된 구성가능 소프트웨어 시스템은 어노테이션 기반 방법(annotation-based approach)으로 개발되었다. 구성가능 소프트웨어 시스템에 특정 구성을 제공하면 구체적인 개별 소프트웨어 시스템이 생성되고 그와 관련된 시험스위트도 함께 구성된다.

SPL 시험 커버리지 측정 도구는 이클립스 플러그인으로 개발되었으며 시험 커버리지 측정도구로는 JaCoCo 를 사용하였고 시험케이스 구현 및 실행은 JUnit 프레임워크를 활용하였다.

<표 1> 실험 대상 제품군 메트릭

Subject	Size metrics			
	#LOC	# of packages	# of classes	# of methods
ATM	1,160	2	27	100
BankAccount	189	3	9	22
Chess	2,149	7	22	162
Elevator	405	3	9	60
FeatureAMP1	1,350	4	15	93
FeatureAMP8	2,376	2	6	106

<표 2>는 SPL 코드베이스로부터 제품들을 생성하여 제안한 방법을 토대로 시험을 수행하고 SPL 수준의 시험 커버리지를 추적한 결과를 보여준다. 여기서 사용된 시험 케이스는 각 제품에 대하여 최소 70%의 라인 커버리지를 달성하도록 정의하고 있다[3]. <표 2>의 SPL 커버리지 열은 제안한 방법으로 시험을 진행하여 누적된 SPL 커버리지를 나타낸다.

부가적으로 제안한 방법인 원자단위 시험의 커버리지 추적은 시험이 커버하지 못한 피처들을 더 정확하게 추적하였다. 기존의 시험 커버리지 추적은 특정 시험 단위에 대한 시험의 수행 유무만을 고려할 뿐 횟수를 고려하지 않기 때문에 피처(f1)가 다른 피처

<표 2> 실험 결과

Subject	Variability metrices			Test suite metrics				
	# of features	Total # of valid products	# of generated products	# of test cases	SPL coverage	Mutants killed	#1)	#2)
ATM	7	80	79	76	91%	91%	**0**	**0%**
BankAccount	10	144	143	42	92%	92%	**2**	**43%**
Chess	3	8	6	77	72%	72%	**0**	**0%**
Elevator	6	19	18	59	92%	10%	**1**	**17%**
FeatureAMP1	28	6732	250	18	85%	46%	**11**	**54%**
FeatureAMP8	27	15708	250	78	82%	42%	**0**	**0%**

#1) 커버리지에 변화를 주지 않는 피처 수, #2) 부모 제품과 비교해서 커버리지가 변하지 않는 제품들의 비율

(f2)를 포함해야 하는 'requires' 제약조건에서 문제가 발생할 수 있다. 이때 f1 은 f2 를 'requires'하기 때문에 f1 의 시험 커버리지는 f2 의 시험 커버리지에 포함될 가능성이 존재하며 이 경우에 수행되는 전체 시험의 양이 증가하여 더욱 정교한 시험이 수행됐음에도 불구하고 시험 커버리지는 증가하지 않은 것으로 판단될 수 있다. 그러나 원자단위 시험을 토대로 커버리지를 추적하는 경우, 같은 부분일지라도 서로 다른 시험에 의해서 얼마나 정밀하게 커버되는지 파악할 수 있도록 관련 정보를 제공하기 때문에 시험 커버리지의 증감을 더 정확하게 판단할 수 있다.

특정 피처가 존재하건 존재하지 않건 시험 커버리지가 동일하다면 이는 해당 피처들에 대한 시험이 제대로 이루어지지 않는다는 것을 의미한다. 제안한 방법은 시험이 커버하지 못한 피처들에 대한 정보를 제공하기 때문에 시험 엔지니어가 가능한 모든 피처를 커버하는 방향으로 시험케이스를 설계할 수 있도록 도움을 준다.

본 연구에서 제시하는 방법을 통하여 커버리지를 측정한 결과, 커버리지에 변화를 주지 않는 피처들을 추출할 수 있었다. 또한, 부모 제품과 비교해서 커버리지가 변하지 않는 자식 제품들의 비율도 측정할 수 있다.

4. 결론

본 연구에서 제안한 방법은 임의의 제품들을 생성하고, 피처 집합의 포함 관계를 기준으로 시험으로부터 얻어진 제품의 시험 커버리지들을 계층화함으로써 피처의 증가에 따른 커버리지의 누적을 확인하는 방법이다. SPL 시험 커버리지를 피처 관점에서 정량적으로 평가할 수 있으며, 얻어진 결과로 특정 피처에 대한 시험 케이스의 부재, 즉 피처가 증가하였으나 커버리지가 증가하지 않은 경우를 식별할 수 있었다. 이는 기존의 피처 커버리지 기준, 피처 상호작용 기준, 라인 커버리지 등의 기준에 충분히 부합하는 SPL 시험에서 추가적인 문제들을 발견할 수 있었다는 점에서 의의가 있다.

그렇지만, 본 논문이 제안한 시험 커버리지 측정 방법은 구성가능한 소프트웨어 시스템에만 그 적합성을 실험하였다. 실험과 프로토타이핑 도구 또한 구성가능한 소프트웨어 시스템의 코드의 부분들이 선택한 피처에 따라 선택되는 어노테이션 기반 방법 이므로, SPL 코드베이스와 제품들의 라인 수가 변하지 않는 특수한 경우이다. 따라서, SPL 코드베이스가 변하는 조합 기반 방법(composition-based approach)에서도 적용할 수 있는 추가적인 연구가 필요하다.

* 이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2020R1F1A1071650).

참고문헌

[1] Klaus Pohl, Günter Böckle and Frank van der Linden "Software Product Line Engineering: Foundations, Principles, and Techniques" Springer

[2] Jihyun Lee, Sungwon Kang, Pilsu Jung, "Test Coverage Criteria for Software Product Line Testing: Systematic Literature Review", Information and Software Technology, Vol.122, pp.1-22, 2020

[3] Fischer Ferreira, Markos Viggiato, Maurício Souza, Eduardo Figueiredo, "Testing Configurable Software Systems: The Failure Observation Challenge", Software Product Line Conference (SPLC'20), October 19–23, pp. 1-6, 2020,