한국산학기술학회논문지
Vol. 11, No. 11 pp. 4460-4467, 2010

철도시스템 소프트웨어 테스트 커버리지 자동화 도구 및 기준 분석

조현정[1*], 황종규[1], 신승권[2], 오석문[3]
[1]한국철도기술연구원 열차제어통신연구실, [2]초고속열차연구실, [3]정책전략연구실

Analysis of S/W Test Coverage Automated Tool & Standard in Railway System

Hyun-Jeong Jo[1*], Jong-Gyu Hwang[1], Seung-Kwon Shin and Suk-Mun Oh[3]
[1]**Division of Train Control & Communication Research, Korea Railroad Research Institute**
[2]**Division of Ultra High Speed Rail, Korea Railroad Research Institute**
[3]**Office of Policy & Strategy, Korea Railroad Research Institute**

요 약 최근 컴퓨터시스템으로 전환되고 있는 철도시스템에서 소프트웨어에의 의존성이 급격히 증가함에 따라 임베디드화된 철도시스템 소프트웨어 신뢰성과 안전성의 검증이 중요한 문제로 대두되기 시작했다. 이에 따라 철도 소프트웨어 관련 국제표준에서도 각종 소프트웨어 테스트 및 검증활동을 요구하고 있으며, 이에 대응하여 본 논문에서는 철도시스템 소프트웨어 테스트 커버리지 자동화 도구 및 기준 분석과 개발 결과에 대해 제시하고 있다. 본 논문에서는 철도시스템 소프트웨어 안전성 검증을 위한 정량적인 항목으로 매우 중요한 테스트 커버리지를 자동으로 측정할 수 있는 제어흐름 분석도구를 개발하였으며, 본 도구의 결과를 실제 철도 산업 현장에서 활용하기 위해 타분야 제시기준 등을 분석하여 철도 소프트웨어 안전무결성레벨(SWSIL)에 따른 판단 기준을 제시하였다. 개발한 도구는 기존 해외 도구에 비해서 여러 테스트 커버리지를 효과적으로 측정할 수 있는 강점이 있으며, 실제 철도 현장에서 활용성이 높아 철도 소프트웨어의 개발 및 테스트 기술 발전을 기대할 수 있다.

Abstract Recent advances in computer technology have brought more dependence on software to railway systems and changed to computer systems. Hence, the reliability and safety assurance of the vital software running on the embedded railway system is going to tend toward very critical task. Accordingly, various software test and validation activities are highly recommended in the international standards related railway software. In this paper, we presented an automated analysis tool and standard for software testing coverage in railway system, and presented its result of implementation. We developed the control flow analysis tool estimating test coverage as an important quantitative item for software safety verification in railway software. Also, we proposed judgement standards due to railway S/W Safety Integrity Level(SWSIL) based on analysis of standards in any other field for utilizing developed tool widely at real railway industrial sites. This tool has more advantage of effective measuring various test coverages than other countries, so we can expect railway S/W development and testing technology of real railway industrial sites in Korea.

Key Words : Railway systems, Software testing, Test coverage, Safety integrity level(SIL)

1. 서론

철도시스템은 최근 기존의 기계적 장치로부터 컴퓨터 시스템으로 전환되고 있으며, 소프트웨어에의 의존성이 급격하게 증가하고 있다. 컴퓨터 기술의 발달에 따라 지능화 및 자동화를 위해 소프트웨어가 더욱 복잡해지게

*교신저자 : 조현정(hjjo@krri.re.kr)
접수일 10년 08월 11일 수정일 (1차 10년 09월 27일, 2차 10년 10월 25일) 게재확정일 10년 11월 19일

되면서, 철도시스템에서 소프트웨어가 차지하는 비중이 더욱 증대되고 있다. 철도시스템 소프트웨어의 크기와 복잡도는 하드웨어의 발달 속도보다는 느리지만, 점차적으로 규모가 커지며 복잡도도 증가할 것으로 예상된다. 이에 따라 임베디드화된 철도시스템 소프트웨어의 신뢰성과 안전성을 검증하는 것이 중요한 문제로 대두되기 시작했다. 철도시스템 소프트웨어 안전성 요구사항들이 최근 들어 IEC 61508과 IEC 62279에 의해 국제표준화 되었고[1,2], 또한 국내에서도 철도안전법이 제정되어 이러한 철도시스템 관련 국제표준에서 요구하는 각종 소프트웨어 테스트 및 검증활동을 요구하는 분위기가 조성되고 있다[3]. 하지만 아직까지 소프트웨어 검증은 개발과정에 대한 문서에 주로 의존하고 있으며, 극히 일부만 테스트에 의한 정량적인 분석이 이루어지고 있다[4,5]. 또한, 국내에서의 국제표준에 따른 철도시스템 소프트웨어 테스트 및 검증을 위한 기준이나 이에 부합하는 기술에 대한 연구는 이제 막 시작하는 초기단계에 불과한 실정이다[6].

따라서 철도시스템 소프트웨어관련 국제표준에서 요구하고 있는 안전성 활동에 대한 문서 검증뿐만 아니라 소프트웨어 테스트를 통한 분석 및 검증에 대응하기 위한 구체적인 기술 개발과 확인을 위한 판단 기준이 매우 필요한 상황이다. 특히, 국제표준에서 요구하고 있는 철도 소프트웨어 검증 항목 중 정량적인 결과를 도출할 수 있는 제어흐름 분석은 대부분 철도 소프트웨어 안전무결성레벨(SIL: Safety Integrity Level) 등급이 3 또는 4로 분류되는 바이탈(Vital) 철도시스템 소프트웨어 검증의 경우, 관련 국제 표준에서 'HR : Highly Recommend' 조건으로 규정되어 있다[1,2]. 철도 소프트웨어의 제어흐름 분석(Control Flow Analysis)을 통한 결과들은 코드 기반의 테스트 커버리지(test coverage)로 활용이 가능하며, 항공, 원전 등의 바이탈한 타분야에서도 소프트웨어 안전성 등급에 따르는 소프트웨어 검증을 위해 적용하고 있는 코드 커버리지를 측정할 수 있게 해주는 매우 중요한 항목임을 알 수 있다[7-11]. 인명과 직결되는 바이탈한 철도시스템은 소프트웨어 규모가 큰 임베디드 시스템으로서 반드시 안전성 확보를 해야만 하며, 이를 위해 철도 소프트웨어 검증을 위한 이와 같은 코드 커버리지 적용이 불가피하다.

이와 더불어 소프트웨어의 복잡도 증가에 따라 점차 난해한 부분인 소프트웨어의 확인 및 검증에 대한 판단 기준 확립을 위해 코드 커버리지 결과를 소프트웨어 SIL 등급에 맞춰 측정할 수 있는 기준 수립도 요구된다. 따라서 본 논문의 2장에서는 이러한 철도시스템 소프트웨어의 검증을 위한 코드 기반 테스트 커버리지 측정을 자동적으로 수행할 수 있는 도구인 제어흐름 분석모듈의 개발결과를 보여주고 있으며, 3장에서는 실제 철도 산업현장에서 활용할 수 있는 커버리지 측정 결과에 대한 판단 기준을 제시하였다.

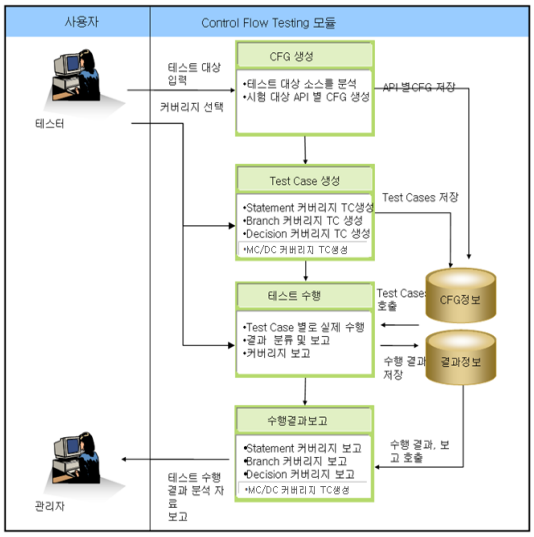
2. 철도시스템 소프트웨어 코드기반 테스트 커버리지 자동화 도구의 개발

2.1 제어흐름 분석모듈의 테스트 커버리지

제어흐름 분석모듈에서는 그림 1과 같이 소스 코드의 영역 및 분기, 조건문 등의 실행 여부에 따라 커버리지를 측정하여 보고한다. 커버리지는 타산업분야에서도 정량적으로 소프트웨어를 검증할 수 있는 대표적인 척도로써 사용되고 있으며, 나아가 소프트웨어 품질까지도 가늠하게 해준다. 일반적으로 소프트웨어 테스트 분야에서는 여러 종류의 커버리지를 제시하고 있는데, 본 논문에서 개발한 철도용 소프트웨어 제어흐름 분석모듈은 그 중에 문장 커버리지, 분기 커버리지, 변경조건/결정 커버리지를 제공한다. IEEE Std. 1008-1997 규격에 따르면 단위시험 단계에서 모든 소프트웨어는 테스트케이스(Test Case)에 의해 검증되어야 하며, 테스트케이스에 의해 모든 소프트웨어 코드의 문장 및 분기 커버리지를 만족해야 한다고 언급되어 있다[7]. 이러한 단위시험 수행의 테스트 커버리지를 측정하기 위해 본 규격에서는 자동화된 수단을 권고하고 있다. 이 외에도 IEC Std. 60880- 2006 규격에서도 소프트웨어 구현단계 검증을 위해서는 문장 및 분기 커버리지 등의 방법들을 사용하도록 권고하고 있고[10], 실제 산업분야에서도 이미 문장 및 분기 커버리지가 널리 활용되고 있다.

또한, 항공산업 표준규격인 RTCA/DO-178B에서는 시스템 안전성 평가를 통해 확인된 사고종류에 따라 소프트웨어 등급을 중요도에 따라 구분해 놓았으며, 코드 커버리지 요건은 그 등급에 따라 정의해 놓았다[11]. [11]에 따르면 안전성이 요구되는 중요도가 높은 등급일수록 변경조건/결정 커버리지를 반드시 만족하도록 규정하고 있고, 적용성을 입증하는 사례들 또한 제시되고 있다[12,13]. 따라서 SIL 등급이 3 또는 4로 분류되는 전자연동장치와 같은 바이탈한 철도시스템 소프트웨어에서도 변경조건/결정 커버리지를 만족해야 할 것이며, 이에 따라 본 논문에서 개발한 철도용 소프트웨어 테스트 커버리지 자동화 도구를 구성하고 있는 제어흐름 분석모듈에서는 문장 커버리지, 분기 커버리지, 변경조건/결정 커버리지를 모두 제공하도록 구현하였다. 각각의 커버리지에

대한 구체적인 내용은 다음과 같다.



CFG: Control Flow Graph, API: Application Programming Interface, TC: Test Case

[그림 1] 제어흐름 분석모듈의 기능 구성도

- 문장 커버리지

블록은 연속적인 문장의 집합으로 분기나 반복문이 없이 순차적으로 실행되는 영역을 의미한다. 분기가 발생하면 새로운 블록이 생성된다. 이러한 블록을 구성하는 문장 커버리지는 테스트 대상 소스 코드에 존재하는 블록 영역내 문장 대비 실제 수행된 문장을 비율로 구한 것이다.

- 분기 커버리지

분기 커버리지는 If 문과 같은 분기문에 의한 제어 흐름이 참인 경우, 거짓인 경우와 같이 다양하게 수행되었는지를 평가한다. 프로그램 내에 분기문에 의해 결정조건이 6개 있다고 가정할 경우, 그 중 4가지 조건이 수행되면 분기 커버리지는 4/6, 66%가 된다. 보통 분기커버리지는 프로그램의 모든 결정에서 가능한 모든 결과에 대해 적어도 한번 이상 수행되어야 한다.

- 변경조건/결정 커버리지

변경조건/결정 커버리지는 분기문의 결정 조건뿐만 아니라, 분기문을 구성하는 각 조건문에 대해서 다양하게 수행되었는지를 평가한다. 그림 2의 프로그램 코드를 보면 3번 줄에 분기문이 존재한다. 분기문의 결정 조건을 진리표로 나타내면 다음 표 1과 같다. 변경조건/결정 커버리지의 케이스는 각 조건이 결정 조건을 만족하는 조합이다. 조건1 결정을 만족하는 상태는 1, 3번이고, 조건2의 상태는 참으로 고정되어 있으며, 이는 조건1의 참/거짓에 의해 전체 조건문이 참/거짓으로 결정되기 때문이다. 조건2 결정을 만족하는 상태는 1, 2번이다. 따라서 조건1과 조건2의 결정 조건을 만족하는 상태는 1, 2, 3이다. 변경조건/결정 커버리지의 경우 1, 2, 3과 같은 각 조건의 상태를 수행하면, 커버리지는 100%가 된다.

```
1   int f (int a, int b) {
2     int c = a + b;
3     if (a > 10 && c > 50)
4     {
5       printf ("c = %d\n", c);
6     } else {
7       printf ("b = %d\n", b);
8     }
9   }
```

[그림 2] 변경조건/결정 커버리지의 예제 프로그램

[표 1] 연산자 진리표

상태	조건1 (a>10)	조건2 (c>50)	조건1&& 조건2
1	참	참	참
2	참	거짓	거짓
3	거짓	참	거짓
4	거짓	거짓	거짓

이와 같은 코드기반 테스트 커버리지를 자동으로 측정해주는 상용화된 해외 도구가 존재하며, 바이탈한 항공분야에서는 이 도구를 활용하고 있다[14]. 하지만, 다른 산업분야보다 소프트웨어의 개발이나 테스트하는 기술이 앞선 항공 분야임에도 불구하고 아직까지 국내 자체적인 기술로 항공 분야에 적용하기에 적합한 테스트 커버리지 자동화 도구를 개발하지 못했으며, 이에 따라 실제 적용하는데 있어서 커스터마이징 기술지원이 불가함은 물론이고 소프트웨어 검증 도구에 대한 전용 지침도 제정해야 하는 등 산업 현장에서 활용하는데 어려움이 많은 상황이다. 이 외에도 해외 테스트 커버리지 자동화 도구는 상기 언급한 커버리지들을 한꺼번에 효과적으로 측정해주지 못하고, 각각의 커버리지 측정에 적합한 도구를 따로 활용해야 하는 단점을 지내고 있다.

따라서 본 논문에서는 철도 산업분야에서 활용할 수 있도록 소프트웨어 분야에서 필요로 하는 검증 항목들(코딩 룰 검사, 메트릭 지원, 제어흐름 분석, 데이터흐름 분석, 경계값 분석 등) 중에 테스트 커버리지를 정량적으로 측정해 주는 주요 항목인 제어흐름 분석모듈을 자체

적으로 개발하여 커스터마이징 기술지원이 가능하도록 하였으며, 다른 검증 항목들을 측정하는 도구와도 호환성을 지닐 수 있도록 제작하였다. 또한 무엇보다 여러 커버리지를 본 도구만을 사용하여 한꺼번에 효과적으로 측정할 수 있다는 강점을 갖고 있으므로 높은 실용성을 기대할 수 있다.

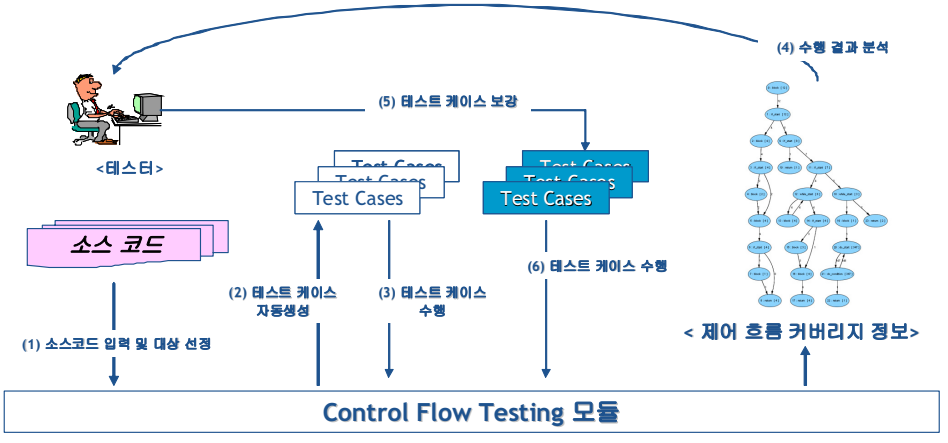
2.2 제어흐름 분석모듈의 설계 및 구현결과

제어흐름 분석모듈은 소스코드를 분석하여 제어 흐름 그래프를 생성, 문장 커버리지, 분기 커버리지, 변경조건/결정 커버리지를 만족할 수 있는 테스트 케이스를 생성, 수행하여 오류의 존재를 찾는 모듈이다. 제어흐름 분석모듈은 프로그램 분석, 테스트 케이스 생성, 테스트 케이스 수행, 테스트 수행 결과 보고와 테스트 시나리오를 가지며, 프로그램 분석 정보와 테스트 결과 정보를 나타내는 자료구조 및 저장소를 가진다. 제어흐름 분석모듈은 제어흐름 그래프 생성, 테스트 데이터 자동 생성, 문장 커버리지, 분기 커버리지, 변경조건/결정 커버리지 측정과 같은 주요 기능을 갖는다. 다음 표 2는 주요 기능에 대한 설명을 정리한 것이다. 이러한 기능을 지니는 제어흐름 분석모듈의 동작 시나리오는 크게 테스트 케이스 준비, 수행, 결과 보고로 진행된다. 제어흐름 분석모듈은 전 과정을 자동으로 수행하되, 자동으로 테스트가 되지 않는 부분에 대해서는 테스터가 개입하여 점진적으로 테스트를 보강하는 것이 주요 사용 시나리오이다. 다음 그림 3은 제어흐름 분석모듈의 동작 시나리오이다.

테스트 준비단계는 크게 테스트 대상 선정 및 테스트 케이스 작성으로 구성된다. 우선 테스터는 컴파일이 가능한 C/C++ 소스코드를 준비해야 한다. 제어흐름 분석모듈은 소스 파일을 입력으로 받아 어떤 함수와 타입들이 정의되어 있는지를 분석을 한 다음, 테스트 대상이 될 함수의 목록을 테스터에게 제공한다. 테스터는 테스트 대상으로 선정한 함수를 제어흐름 분석모듈의 테스트 데이터 자동생성 기능을 이용하여 해당 함수에 적합한 테스트 케이스를 자동으로 추출한다. 테스트 수행단계는 수행 가능한 테스트 프로그램을 만들기 위한 작업과 준비된 테스트 케이스를 이용하여 실제 수행하는 작업으로 구성된다. 이는 제어흐름 분석모듈에서 자동으로 생성해 준다.

[표 2] 제어흐름 분석모듈 기능 리스트

기능	설명
프로그램 분석 및 CFG생성	프로그램에 구현되어 있는 함수와 최상위 API 함수의 리스트를 추출한다. 함수별 제어흐름 그래프는 테스트의 기반 모델이 되며, 테스트 과정은 최대한 많은 제어흐름 그래프의 최대한 많은 부분을 수행하는 것을 목적으로 한다. 각 함수 간의 호출그래프(Call Graph)를 생성하여 테스트 스크립트, 테스트 데이터를 생성하는데 활용한다. 분석 된 내용 및 CFG를 저장소에 저장한다.
Test Cases 생성	Statement, Branch, MC/DC 커버리지를 위한 테스트 케이스 생성한다. 생성된 테스트 케이스를 저장소에 저장한다.
테스트 수행	실제 테스트 대상 소스와 테스트를 위하여 생성된 소스코드 등을 통합해서 컴파일 하여 생성된 프로그램을 테스트케이스별로 수행하여 그 수행 결과를 저장소에 저장한다.
제어흐름 분석수행 결과보고	Statement, Branch, MC/DC 커버리지를 보고한다.

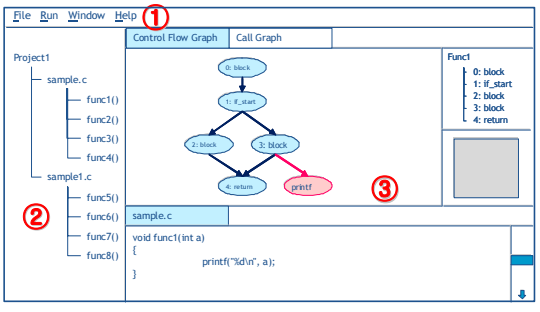


[그림 3] 제어흐름 분석모듈의 동작 시나리오

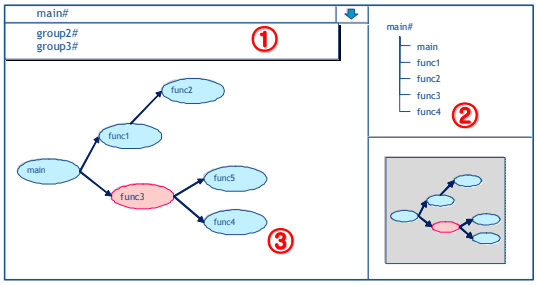
제어흐름 분석모듈은 테스트 준비 단계에서 준비한 테스트 케이스를 자동으로 수행하며, 수행 결과를 내부 저장소에 저장한다. 이 과정에서는 대부분의 작업이 제어흐름 분석모듈에 의해 자동화가 되며, 테스터가 직접적으로 개입해야 될 부분은 거의 없다.

마지막 단계는 현재까지 수행한 테스트 결과를 분석하여, 테스트가 충분한지를 평가하는 작업이다. 제어흐름 분석모듈에서는 문장 커버리지, 분기 커버리지, 변경조건/결정 커버리지 수치를 기반으로 테스트가 충분한지를 평가한다. 테스터는 테스트 수행 결과 목표 수치에 도달하지 않을 경우 테스트를 더 해야 하며, 목표 수치에 도달하도록 테스트 케이스를 보강해야 한다. 제어흐름 분석모듈은 수행 결과로 문장 커버리지, 분기 커버리지, 변경조건/결정 커버리지 수치와 입력 값에 따른 실제 수행된 코드 정보를 제공한다. 테스터는 제어흐름 분석모듈에서 제공되는 정보를 기반으로 실제 수행되지 않을 문장 및 분기를 확인한 다음, 그 부분을 테스트하기 위한 테스트 케이스를 보강할 수 있다.

개발한 테스트 커버리지 자동화 도구 프로그램의 전체 화면은 그림 4와 같이 메뉴①, 프로젝트 화면②, 기능별 화면③으로 구성된다. 기본메뉴는 프로젝트 생성, 열기, 작업 수행, 제품 정보와 같은 기능을 수행하기 위한 메뉴 아이템으로 구성되어 있다. 프로젝트 화면에는 현재 작업 중인 프로젝트와 프로젝트에 포함된 소스코드 함수목록을 보여주는 화면으로 구성된다. 본 도구에서는 기본 메뉴①를 이용하여 소스코드를 분석하게 되며, 이 과정에서 내부적으로 제어 흐름 정보를 추출할 수 있다. 또한 소스코드의 함수 간 호출그래프는 연관 관계가 있는 함수들을 그룹을 선정해 그룹 단위로 보여준다. 그림 5에서와 같이 먼저 Graph 목록① 을 선정하면, 해당 그룹에 속해 있는 모든 함수의 호출 관계 ③을 보여준다. 또한 현재 그룹에 포함된 함수를 ②와 같은 목록형태로 제공한다.

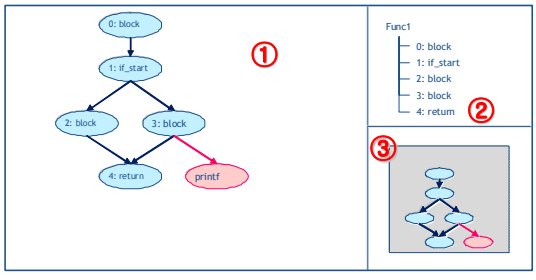


[그림 4] 프로그램 전체 화면

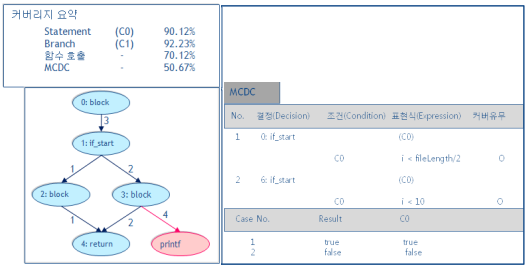


[그림 5] 소스 프로그램 분석: Call Graph

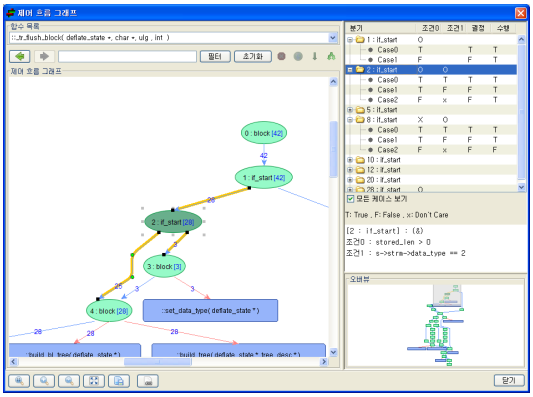
또한 소스코드 분석을 위한 CFG(제어흐름 그래프)는 각 함수별로 1개씩 존재하며, 노드와 수행 흐름을 표현하는 에지가 결합하여 그림 6의 ①과 같은 그래프 형태로 표현된다. CFG의 노드는 분기가 없는 문장(statement)의 집합을 나타내며 에지는 문장간에 수행 순서 관계를 나타낸다. CFG는 호출되는 외부 함수를 특별한 형태로 표현하며, 이를 이용하여 다른 함수의 CFG를 찾아갈 수 있도록 제공한다. 그래프가 큰 경우 전체를 개괄적으로 보여주고 내비게이션을 지원하는 개략화면③을 제공한다. 그림 6의 ②는 CFG내의 노드 목록을 보여준다. 소스코드가 분석 완료된 이후, 기본 메뉴를 이용하여 테스트 케이스를 생성하고 수행한다. 이와 같은 과정을 거쳐 테스트 수행이 이루어지고 나서 확인은 다음 그림 7과 같이 커버리지 결과로 가능하다.



[그림 6] 소스 프로그램 분석: Control Flow Graph



[그림 7] 테스트 수행 및 커버리지 결과확인



[그림 8] 실제 제작한 철도시스템 소프트웨어 커버리지 자동화 도구의 테스트 실행화면

테스트 커버리지 결과는 크게 수치 정보와 코드의 수행 여부를 제공한다. 코드의 수행 여부는 CFG를 기반으로 노드와 에지에 수행 횟수를 기록하여 제공한다. 0으로 기술된 부분은 수행되지 않은 부분이므로 테스터는 이 정보를 바탕으로 테스트 케이스를 보강할 수 있다. 또한 변경조건/결정 커버리지의 경우 수행되어야 하는 케이스와 현재 수행된 케이스에 대해서 표 형태로 제공한다. 이와 같은 설계 개발과정을 바탕으로 실제 제작 완료한 철도시스템 소프트웨어 테스트 커버리지 자동화 도구의 실행 결과 화면은 그림 8과 같다.

3. 철도시스템 소프트웨어 적용을 위한 테스트 커버리지 기준 제시

본 논문의 2장에서 언급한 바와 같이 항공, 원전 분야뿐만 아니라 철도 분야와 같이 바이탈한 산업제어시스템의 소프트웨어가 복잡해지면서 소프트웨어의 안전성 확보를 위한 확인 및 검증이 어려워지고 있다. 이러한 어려움을 해결하고자 타분야에서는 이미 소프트웨어 검증을 위해 점차 테스트 커버리지를 적용하기 시작했으며, 각종 관련 국제 기준에서는 정략적 기준을 제시할 수 있는 코드 기반의 테스트 커버리지를 자동으로 측정해주는 도구들을 사용할 것을 권고하고 있다. 본 논문에서는 철도시스템 소프트웨어 전용으로 활용이 가능한 테스트 커버리지 자동화 도구를 설계 및 구현하였으며, 구체적인 결과는 2장에서 확인하였다.

하지만 개발한 본 도구를 실제 철도 산업현장에서 바로 사용하기 위해서는 테스트 커버리지 수행 결과를 소프트웨어 안전성 중요도를 나타내는 소프트웨어 안전무결성레벨(SWSIL) 등급에 따라 할당하기 위한 정량적 평가기준을 제시해 주어야만 실제 활용도를 높일 수 있을 것이다. 이에 따라 정량적인 기준을 구체적으로 제시하고자 타분야의 테스트 커버리지 적용 현황 조사 및 관련 국제 표준들을 분석하였고, 이를 근거로 철도 분야의 소프트웨어 검증을 위해 산업현장에서 활용할 수 있는 판단기준을 제시하기 위해 기술지침(안) 형태를 작성 중에 있다. 이와 같이 개발한 도구의 실용화를 높이기 위해서 본 논문에서는 표 3과 같은 철도시스템 소프트웨어 테스트 커버리지의 정량적 기준을 제시하고자 하며, 기술지침(안)에 포함시켜 운영기관 및 관련 산업체에서도 유용하게 활용될 수 있도록 추진할 계획이다.

[표 3] 테스트 커버리지의 정량적 평가기준 제시

	SWSIL				
문장 커버리지 (Statement Coverage)	0	1	2	3	4
	NIL	분기 커버리지 100% 만족 권고	분기 커버리지 100% 만족 필수		
분기 커버리지 (Branch Coverage)	0	1	2	3	4
	NIL	분기 커버리지 100% 만족 권고	분기 커버리지 100% 만족 필수		
변경조건/ 결정 커버리지 (MC/DC: Modified Condition/ Decision Coverage)	0	1	2	3	4
	NIL	변경 조건/ 결정 커버리지 100% 만족 권고			변경 조건/결정 커버리지 100% 만족 필수

4. 결론

최근 들어 컴퓨터 기술의 발달에 따라 철도시스템들이 컴퓨터 소프트웨어에의 의존성이 급격하게 증가하고 있으며, 이러한 기술발전에 따라 바이탈한 철도 소프트웨어에 높은 신뢰성과 안전성이 요구되고 있다. 이에 따라 철도시스템 소프트웨어관련 국제표준에서 소프트웨어의 테스트 및 검증을 의무사항으로 요구하고 있고, 이러한 국

제표준에서 요구하고 있는 소프트웨어 검증 항목 중 정량적인 테스트 커버리지 결과를 도출할 수 있는 제어흐름 분석에 대한 정확한 분석을 위해 본 논문에서 개발한 자동화 도구를 제시하였다. 이러한 제어흐름 분석은 코드기반의 테스트 커버리지인 문장 커버리지, 분기 커버리지, 변경조건/결정 커버리지 등의 결과를 도출해주며, 바이탈한 항공 및 원전분야 등에서는 이미 이러한 테스트 커버리지를 소프트웨어 안전성 등급에 따르는 소프트웨어 검증을 위해 적용하고 있는 매우 중요한 검증 항목이다. 먼저 개발한 철도시스템 전용 소프트웨어 제어흐름 분석 도구의 기능 설계에 대해 설명하였고, 그 구현 결과를 구체적으로 보여주었다.

이와 같은 본 논문에서 개발한 철도용 소프트웨어 검증을 위한 테스트 커버리지 측정을 자동으로 수행해주는 도구인 제어흐름 분석모듈은 소스코드의 문장, 분기, 변경조건/결정 커버리지에 대한 분석 결과를 사용자 입장에서 파악하기 쉽도록 제어흐름 그래프와 함수호출 그래프로 표현하였다. 또한, 각각의 측정결과를 최종 수치상으로도 보여줌으로써, 비교 분석도 쉽게 파악 가능하도록 제작하였다. 이러한 철도 소프트웨어 테스트 커버리지 자동화 도구인 제어흐름 분석모듈은 기본적으로 철도 운영기관 등의 수요처에서 철도시스템의 소프트웨어 검증을 위해서 크게 활용될 도구이며, 동시에 철도 관련 산업체들의 소프트웨어 개발과정에서도 해당 개발품의 단위 혹은 통합 테스트 단계에서도 충분히 활용도가 높을 수 있다고 본다.

물론 최종적으로 코드기반 테스트 커버리지 측정 결과가 사용자가 원하는 소프트웨어 안전무결성레벨(SWSIL) 등급에 맞춰질 수 있는 판단 기준을 제시해야만 비로소 실제 철도분야 산업현장에서의 효율성을 최대화시킬 수 있을 것이다. 따라서 본 논문에서는 실제 철도 산업현장에서 활용 가능하도록 커버리지 측정 결과에 대한 각각의 평가기준을 제시하였다. 본 도구를 소프트웨어 검증 및 개발 단계에서 널리 이용한다면, 이를 통해 바이탈한 철도 소프트웨어의 오류를 미연에 방지하여 안전성과 신뢰성을 확보하는데 크게 기여할 수 있을 것이다.

참고문헌

[1] IEC 61508, "Railway Applications - The specification and demonstration of RAMS", 1998.

[2] IEC 62279, "Railway Applications - Software for railway control and protection systems", 2002.

[3] 철도안전법[법률 8852호], 일부개정 2008. 02.

[4] M. Fewstar, D. Graham, “Software Testing Automation: Effective use of test execution tools”, ACM Press, Addison Wesley, 1999.

[5] J.D. Lawrence, "Software qualification in safety applications", Reliability Engineering & System Safety, Vol. 70, No. 2., pp. 167-184, 2000.

[6] 황종규, 조현정, 김형신, “열차제어시스템 소프트웨어 안전성 평가도구의 설계”, 한국철도학회 논문집, 제11권 제2호, pp. 139-144, 2008. 4.

[7] IEEE Std. 1008-1997, "Software Unit Testing", 1997.

[8] IEEE Std. 829-1998, "Software Test Documentation", 1998.

[9] IEEE Std. 1012-2004, "Software Verification and Validation", 2004.

[10] IEC std. 60880-2006, "Software aspects for computer-based systems performing category A functions", 2006.

[11] RTCA/DO-178B, "Software considerations in airborne systems and equipment certification", 1992.

[12] Arnaud Dupuy and Nancy Leveson, "An Empirical Evaluation of the MC/DC Coverage Criterion on the HETE-2 Satellite Software", Proceedings of DASC (Digital Aviation Systems Conference), Phildelphia, 2000. 10.

[13] Peter G Bishop, "MC/DC based estimation and detection of residual faults in PLC logic networks", 14th IEEE International Symposium on Software Reliability Engineering(ISSRE), Denver, Colorado, 2003. 11.

[14] 박무혁. “항공용 S/W 개발 및 인증 기술동향”, 항공우주산업기술동향 5권1호, pp. 15-24, 2007.

조 현 정(Hyun-Jeong Jo) **[정회원]**

- 2003년 2월 : 한국항공대학교 항공전자공학과 (공학학사)
- 2005년 2월 : 광주과학기술원 정보통신공학과 (공학석사)
- 2005년 ~ 현재 : 한국철도기술연구원 선임연구원

<관심분야>
열차제어 및 정보통신 기술, 철도 S/W 테스팅 기술

황 종 규(Jong-Gyu Hwang) **[정회원]**

- 1996년 2월 : 건국대학교 일반대학원 전기공학과 (공학석사)
- 2005년 2월 : 한양대학교 일반대학원 전자통신공학과 (공학박사)
- 1995년12월 ~ 현재 : 한국철도기술연구원 책임연구원

<관심분야>
철도 신호제어 및 정보통신, 임베디드 SW 테스팅

신 승 권(Seung-Kwon Shin) **[정회원]**

- 1998년 2월 : 성균관대학교 공과대학원 전기공학과 (공학석사)
- 2001년 8월 : 성균관대학교 공과대학원 전기전자컴퓨터공학부 (공학박사)
- 2001년 8월 ~ 2003년 3월 : 한국원자력연구소 Post-Doc.
- 2003년 4월 ~ 현재 : 한국철도기술연구원 선임연구원

<관심분야>
제어공학, 열차제어, 신교통시스템, 자기부상열차

오 석 문(Suk-Mun Oh) **[정회원]**

- 1996년 2월 : 전북대학교 전기공학과 (공학석사)
- 2010년 2월 : 고려대학교 산업공학과 (공학박사)
- 1995년 11월 ~ 현재 : 한국철도기술연구원 선임연구원

<관심분야>
OR, 철도시스템 최적화