XML 명세 기반 Built-In-Test 코드 자동 생성 체계

박두호*, 신원*, 장천현*
노영남**, 유석진**, 하동현**

*건국대학교 컴퓨터공학과
e-mail : {pakddo, wonjjang, chchang}@konkuk.ac.kr
** 현대로템㈜
{roh, ysjok, haelec}@hyundai-rotem.co.kr

Design for Automatic code generation of Built-In-Test based on XML Description

Doo-Ho Park* , Won Shin*, Chun-Hyon Chang*
Young-Nam Roh**, Suk-Jin Yu**, Dong-Hyun Ha**

*Dept. of Computer Science and Engineering, Konkuk University
**Hyundai Rotem Company

요 약

BIT(Built-In Test)란 S/W 또는 H/W 의 기능 및 상태를 진단하고 오류에 대응하기 위한 방법론으로 기능에 대한 신뢰성 및 빠른 오류 복구를 보장하기 때문에 다양한 분야에서 BIT 처리를 통해 시스템의 안정성을 높이고 있다. 현업에서의 BIT 는 도메인 특성에 따라 처리해야 하는 작업의 변화가 크기 때문에 구조화 되지 않은 형태로 각각 개발되고 있다. 따라서 BIT 개발 시 반복적인 작업이 수반되며 처리 과정의 수정 또는 처리 범위의 확장을 위해서는 많은 시간 및 인력이 요구된다. 이에 본 논문에서는 BIT 처리를 구조화하기 위하여 처리과정에 필요한 정보들을 일반화된 형태로 기록할 수 있도록 하는 BIT 처리 명세 방안과 BIT 처리 명세를 기반으로 한 자동 코드 생성 체계를 제안한다. 이를 통해 개발 과정의 편의성과 생산성을 향상하고 BIT 처리의 유연성과 확장성을 높일 수 있다.

1. 서론

BIT 기법은 시스템 자체에 소프트웨어 또는 하드웨어의 기능 및 상태를 진단하고 오류를 제어할 수 있는 추가 요소를 포함하는 테스트 방법론이다. 군용 장비, 항공기, 선박 등 기능 동작에 대한 높은 신뢰성과 빠른 오류 복구가 요구되는 다양한 분야에서 활용되고 있다. 현업에서의 BIT 는 적용하고자 하는 도메인에 포함된 여러 장치와 프로그램의 특성을 고려한 형태로 개발되고 있다. 구조화 되지 않은 형태로 개발된 BIT 는 처리 구조를 재사용 할 수 없고 처리 과정을 수정하거나 적용 범위를 확장하기 위해서 많은 시간과 인력이 요구된다는 문제점을 지닌다. 이에 본 논문에서는 BIT 처리 과정을 분석하여 처리 과정 및 범위 명세에 필요 정보를 도출하고 이를 일반화하여 구조화된 XML 형태로 기술할 수 있는 명세 방안을 제시한다. 또한 명세 된 정보를 기반으로 문서 템플릿 구조를 적용한 코드 자동 생성 과정 체계를 제안한다. 이를 통해 BIT 처리 구조의 확장성을 높이고 BIT 처리 구조를 다양한 시스템에 유연성 있게 적용할 수 있도록 한다.

본 논문은 2 장에서 기존 BIT 처리 과정의 문제점 및 명세를 위한 관련 연구를 다루고 3 장에서는 XML 기반의 BIT 명세 방안 및 코드 자동 생성체계에 대한 설계 내용을 기술한다. 마지막 4 장에서는 본 논문의 성과 및 향후 연구 방향에 대해서 기술한다.

2. 관련연구

2.1. BIT

BIT 란 시스템이 주 동작 구조의 적절한 기능 수행을 시험하기 위해 설계된 테스트 구조를 함께 포함하는 기술을 가리킨다. 동작 과정 중 시스템 구성요소 간의 상관 관계로 인하여 발생 가능한 오류를 진단할 수 있는 장점을 가진다[1]. 따라서 우주선 응용 시스템 등에서 지속적으로 BIT 기법을 수행하여 빠른 오류 진단 및 제어 가능하도록 하는 연구가 진행되고 있다[2]. 하드웨어뿐만 아니라 소프트웨어 테스트를

* 본 논문은 2011 년도 지식경제부 기술혁신사업의 지원을 받아 연구되었음 (기동무기 내장형 실시간 제어 시스템용 마이크로초급 정밀도와 99.99% 신뢰성의 RTOS 기술 개발, 10040090)

위한 활용도 가능하다. 객체지향 프레임웍의 가변 부위에 테스트 컴포넌트를 내장하는 형태의 적용 방안이 연구되고 있으며 이는 프레임웍 자체의 시험성을 높여주는 역할을 한다[3].

하지만, 다양한 테스트 대상을 지원하기 위해서는 각 대상 마다 필요한 정보를 도출하고 이에 적합한 BIT 처리 구조를 구성해야 한다. 이 과정에서 많은 시간과 인력을 필요로 한다는 문제점을 가진다. 한번 만들어진 BIT 구조는 테스트 대상에 특화된 형태로 개발되기 때문에 다른 시스템에서 재사용하기 힘든 문제를 포함한다. 따라서 BIT 구조 설계 시 테스트 대상 및 범위를 명확히 규정해야 할 필요가 있고[4], 확장 및 수정을 고려한 형태의 적용이 필요하다.

2.2. 메타 모델링

메타 모델링은 메타 정보를 활용하여 특정 도메인 내에서 통용되는 모델들 간의 관계 및 속성을 언어로 표현하는 과정을 뜻한다[5]. 속성, 연산, 제약 조건의 요소를 통해 개체 정보를 구성하고 값 전달의 주체와 객체를 명시하는 방식으로 개체간의 관계 정보를 구성한다. 각 정보는 메타모델링 언어 및 문법을 통해 표현된다[6][7].

BIT 처리 구조를 일반화하여 명세하기 위해서 메타 모델링 기법이 활용 가능하다. 일반화 하는 목적은 개발 과정의 편의성 증대에 있으며 메타 모델링을 통해 생성된 모델을 기반으로 코드 생성과정을 자동화 시키는 방안이 적용 가능하다.

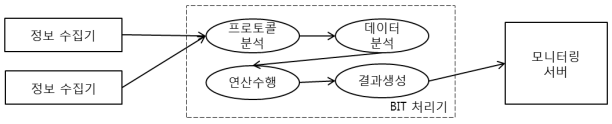
하지만, 명세 과정에서 관계상에서 전달되는 값에 대해 세부적으로 명세 하는 방안이 고려되지 않아 다양한 관계를 효율적으로 표현하기 어렵다. 또한 속성 및 명령과 제약 조건 간에 중복적으로 기술 되는 개념이 존재하고 문법이 어려워 사용상의 불편함이 존재한다. 이를 해결하기 위하여 XML 형태의 명세를 기반으로 한 코드 생성 체계가 연구되고 있다[8].

3. XML 명세 기반 BIT 코드 자동 생성 체계

본 체계는 메타 모델링을 활용하여 BIT 처리 구조를 XML 형태로 명세하고 이를 템플릿 정보와 결합하여 자동으로 BIT 코드를 생성한다.

3.1. 구조화를 위한 BIT 처리 구조 분석

BIT 처리 구조는 일반적으로 정보를 수집하는 부분과 정보를 처리하는 부분 그리고 결과 정보를 취합하는 부분으로 구성된다.

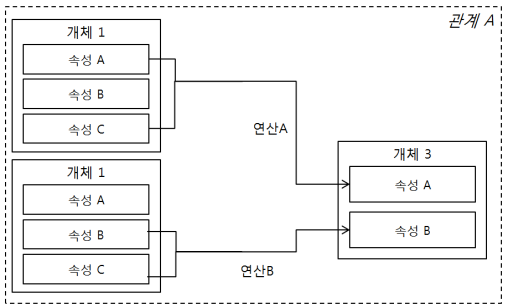


(그림 1) BIT 처리구조 개념도

테스트를 구성하는 각 요소를 메타 모델링에서 사용되는 개체를 활용해 나타낼 수 있다. 개체는 속성을 통해 수집된 정보를 표현하고 나타내는 정보에 대한 제약조건을 함께 포함한다. 개체에 표현된 각 속성은 서로의 정보에 영향을 미치는 관계를 형성하며 이를 연산 정보를 통해 나타낸다. 여러 개체 간에 서로 속성 값을 주고 받는 내용은 관계 정보로 표현한다. 이와 같이 개체와 관계 정보에 따라 속성간의 연산을 수행하여 결과를 도출하는 형태로 BIT 구조를 일반화하여 표현할 수 있다.

3.2. BIT 처리 구조 표현을 위한 XML 명세

XML 로 명세하는 BIT 처리 정보는 크게 개체 정보와 관계 정보로 나뉜다. 우선, 개체 정보는 메타 모델링 과정에서 활용하는 개체 정보와 동일한 내용(속성, 연산, 제약조건)을 가지고 그 중 중복적으로 기술되는 부분인 연산과 제약조건을 통합한 형태의 연산 정보로 구성한다. 연산 정보는 실제 처리 가능한 코드 내용을 그대로 표현할 수 있는 형태이다. 추가로 값 전달 시 필요한 정보와 내부적 처리를 위한 정보를 구분하여 기술한다. 전달 정보 상에 각기 다른 형태로 구성되는 여러 개의 전송 정보를 표현할 수 있도록 하며 이는 관계 설정 시 전송되는 값을 명확히 나타내기 위함이다.



(그림 2) BIT 처리 구조 정보 구성 개념도

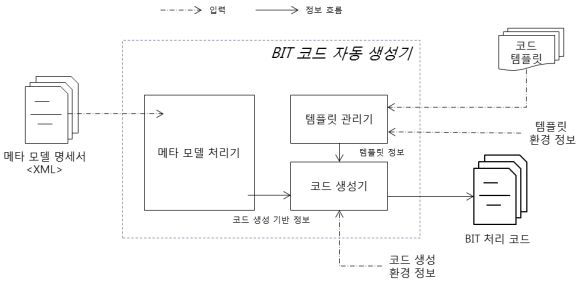
관계 정보는 관계를 구성하는 참여 개체들을 기술하고 각 개체들의 속성 전달 내용을 원본 및 대상을 지정하는 형태로 값 전달 내용을 기록한다. 이를 통해 개체간의 속성에 존재하는 연관 관계를 상세히 표현 할 수 있다.

3.3. 명세 기반 자동 코드 생성 체계

BIT 코드 자동 생성기는 XML 명세를 입력 받아 코드 생성 기반 정보를 생성하고 템플릿 정보를 결합하여 BIT 코드를 생성한다.

메타 모델 처리기는 명세서를 입력 받아 각 파일을 분석한다. 개체 및 관계 정보에서 명세한 속성이 정확히 입력 정보 상에 표현 되었는지 검증하고 입력 정보를 재구성하여 코드 생성 기반 정보를 생성한다. 코드 생성 기반 정보는 입력 정보를 다른 정보와 손쉽게 결합하기 위해 생성하며 개체명, 속성, 값 전달 내용 등으로 구성된다. 템플릿 관리기는 코드 템플릿을 입력 받아 템플릿환경 정보를 적용한 템플릿 정보

를 생성한다. 코드 템플릿은 BIT 처리 과정을 단계별로 구현한 프로토타입 코드 중 가변적인 부분을 표현한 형태를 가진다. 코드 생성기는 템플릿 정보와 코드 생성 기반 정보를 결합하여 코드 정보를 생성하고 파일 형태, 저장 위치 등을 포함하는 코드 생성 환경 정보에 따라 BIT 처리 코드를 파일로 출력한다.



(그림 3) BIT 코드 자동 생성기 구조

BIT 처리과정을 나타내는 템플릿과 정보를 수집할 대상 및 정보간의 관계에 대한 명세 정보를 분리함으로써 한 정보가 수정되어도 다른 정보에 영향을 끼치지 않고 코드를 생성할 수 있도록 한다. 따라서 수정 및 확장이 용이하고 다양한 구조에 유연하게 적용할 수 있는 기반이 된다.

4. 결론

BIT 처리 구조를 수정하거나 확장하려 할 경우 많은 시간과 인력이 필요하다. 또한 한 번 사용된 BIT 처리 구조는 재사용이 어렵다는 단점을 가진다. 본 논문에서는 이 문제점을 해결하기 위해 메타 모델링 기법을 활용한 BIT 처리 구조 명세 방안을 제시하였다. 이를 통해 확장된 메타모델링 개념을 포함하는 XML 형태로 BIT 처리 구조에 대한 명세 정보를 생성할 수 있다. 또한 명세 정보를 기반으로 하는 코드 자동 생성체계를 통해 BIT 처리 대상을 기록한 명세 정보와 실제 처리 과정을 기록한 템플릿 정보를 결합하는 방법으로 코드 생성 과정을 자동화 하였다. 이를 통해 다양한 분야에 대한 BIT 처리 구조 적용이 쉬워지고, 단순한 수정으로 BIT 처리 구조를 변형할 수 있도록 하여 BIT 처리 구조 개발 시 사용성 및 생산성의 증대를 꾀할 수 있다.

향후에는 제안한 코드 자동 생성체계를 구현하고 이를 통해 생성된 BIT 처리코드가 정상적으로 오류를 검출 하는지 시험하고 검증하는 연구를 수행할 예정이다.

참고문헌

[1] V.D. Agrawal, C.R. Kime, and K.K. Saluja, "A Tutorial on Built-In Self-Test, Part 1: Principles", IEEE Design and Test of Computers, Vol. 10, Issue 1, pp. 73-82, 1993

[2] A. Agarwal, G. Bhatia, S. Chakraverty, "State Model for Scheduling Built-in Self-Test and Scrubbing in FPGA to maximize the system availability in space applications", Proc. India International Conference on Power Electronics 2010, pp. 1-6, 2010

[3] T.W. Jeon, H.W. Seung, S.Y. Lee, "Embedding Built-in Tests in Hot Spots of an Object-Oriented Framework", ACM SIGPLAN Notices, Vol. 37, Issue 8, pp. 25-34, 2002

[4] J. Smith, D. Lowenstein, "Built in Test Coverage and Diagnostics", Proc. IEEE AUTOTESTCON 2009, pp. 169-172 , 2009

[5] Y. Liu, Y. Wang, "A study of metamodeling based on MDA", Proc. IEEE international Conference on Computer Research and Development 2011, pp. 171-173, 2011

[6] S. Hong, F. Maryanski, "Using a metamodel to represent object-oriented data models", Proc. International Co-nference on Data Engineering 1990, pp.11-19, 1990

[7] S. Hong, F. Maryanski, "Representation of object-oriented data models", Information Sciences, Vol. 52, Issue 3, pp. 247–284, 1990

[8] D.H. Park, S.D. Kim, "XML rule based source code generator for UML CASE tool", Proc. Asia-Pacific Software Engineering Conference 2001, pp. 53-60, 2001