소프트웨어 관점에서 본 내장형 시스템의 테스트 프로세스

성아영[0], 최병주[0], 최진영[1], 이나영[2], 이장수[3]
[0]이화여자대학교, [1]고려대학교, [2]서울대학교, [3]한국 원자력 연구소
{aysung, bjchoi}@ewha.ac.kr, choi@formal.korea.ac.kr, jslee@kaeri.re.kr

Test Process of the Safety-Critical Embedded System in Software Perspective

Ahyoung Sung[0], Byoungju Choi[0], Jinyoung Choi[1], Nayoung Lee[2], Jangsoo Lee[3]
[0]Ewha Womans Univ., [1]Korea Univ., [2]Seoul National Univ., [3]Korea Atomic Energy Research Institute

요 약

내장형 시스템의 기능이 복잡해지면서, 내장형 소프트웨어에 대한 테스팅이 중요하게 인식되고 있다. 특히 원자력 발전소 계통 기기와 같이 안전에 대한 치명도가 높은 Safety-Critical Embedded System일수록 탑재되는 내장형 소프트웨어에 대한 철저한 테스팅이 요구되기 때문에 본 논문에서는 대상 시스템에 탑재되는 소프트웨어 테스트를 중심으로 하여 Safety-Critical Embedded System을 위한 테스트 프로세스 및 이에 필요한 테스팅 기법을 제안한다.

1. 서론

내장형 시스템(Embedded System)은 하드웨어, 소프트웨어, 운영체제가 결합된 시스템으로, 최근 내장형 시스템 및 내장형 소프트웨어의 시장 규모 및 수요가 국내외적으로 증가하고 있다. 하드웨어 기술 개발의 속도가 매우 빠르게 증가하는 현실을 고려할 때, 내장형 시스템의 가치는 하드웨어보다 소프트웨어가 좌우하는 기술 집약적 산업이라고 할 수 있기 때문에, 고기능의 소프트웨어가 시스템 안에서 올바르게 작동할 수 있는지를 보장하기 위한 테스트 프로세스가 필요하다.

내장형 시스템 특성상 한번 개발된 시스템에서 오류가 나면 오류를 수정하는 것이 쉽지 않으며, 시스템을 구성하는 하드웨어 및 소프트웨어가 독립적으로 잘 돌아가더라도, 이들을 통합하였을 때 발생하는 오류의 원인 파악이 어렵다. 특히 발전소 계통 기기 및 의료기기와 같이 안전에 대한 치명도가 높은 내장형 시스템(Safety-Critical Embedded System)일수록, 내장형 시스템에 들어가는 소프트웨어에 대한 철저한 테스팅이 요구된다.

현재 내장형 소프트웨어 및 내장형 시스템 개발에 대한 연구는 활발히 진행되고 있는 반면에, 내장형 시스템의 테스트에 관한 연구는 미흡한 실정이다. 따라서 본 논문에서는 내장형 시스템의 가치를 좌우하는 내장형 소프트웨어에 중심을 두고, 소프트웨어 관점에서 본 내장형 시스템의 테스트 프로세스(Process)를 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 소프트웨어 관점에서 본 내장형 시스템의 테스트 프로세스에 대해 기술하며, 3장에서는 2장에서 기술된 각 테스트 활동에 대한 기법을 기술하며, 4장에서는 결론을 맺는다.

2. 내장형 시스템의 테스트 프로세스

테스트 프로세스란 테스트를 수행하기 위한 일련의 절차 및 산출물을 나타내는 것으로, 본 절에서는 내장형 시스템 테스트를 위하여 소프트웨어 관점에서 본 내장형 시스템 테스트 프로세스를 제시함으로써, 내장형 시스템 테스트를 구성하는 활동 및 활동에 따른 산출물을 제시한다.

2.1 Process Flow Chart

내장형 시스템 테스트 프로세스를 나타내기 위하여, 노드(Node)와 에지(Edge)로 이루어진 Process Flow Chart(PFC)를 제안한다. PFC는 대상 시스템의 활동 및 활동에 따른 산출물들을 함께 표현하기 위한 다이어그램으로, 노드와 에지를 이용하여 대상 시스템에 대한 프로세스를 나타낸다.

PFC를 구성하는 노드는 활동과 산출물을 의미한다. PFC를 구성하는 노드 및 산출물, 에지에 대한 설명 및 기호는 아래와 같다.

- 노드
  - 둥근 사각형: 둥근 사각형은 프로세스를 구성하는 활동(Activity)을 의미하며, 내부에는 활동 명을 기술한다.
  - 사각형: 프로세스로부터 산출되는 산출물을 의미한다.
- 에지
  - 화살표: 다음 단계의 활동을 나타낸다.
  - 점선 화살표: 해당 단계의 산출물을 활용한다.

2.2 내장형 시스템의 테스트 프로세스

2.1절에서 기술한 PFC의 기호를 이용하여 소프트웨어 관점에서 본 내장형 시스템의 테스트 프로세스를 나타내면 아래의

그림 1과 같다.

내장형 시스템은 운영체제, 어플리케이션 소프트웨어, 하드웨어로 조합된 시스템으로 각 요소에 대한 테스팅이 수반되어야 한다. 일반적으로 내장형 시스템은 운영체제와 응용 소프트웨어가 모두 탑재된 경우와, 응용 소프트웨어만 탑재된 경우가 있다[1]. 운영체제가 있는 내장형 시스템의 경우, 운영체제 테스팅, 응용 소프트웨어 테스팅, 응용 소프트웨어와 보드와의 상호작용 테스팅을 수행하며, 운영체제가 탑재되지 않은 경우, 응용 소프트웨어 테스팅, 응용 소프트웨어와 보드와의 상호작용 테스팅을 수행한다.

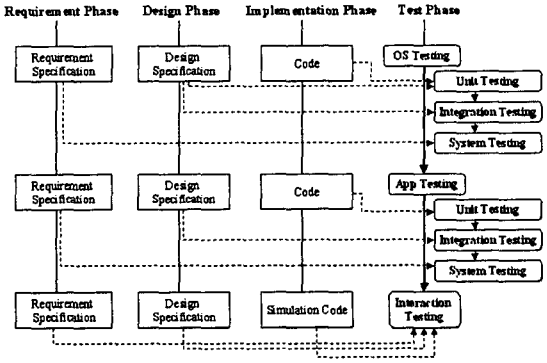

그림 1. 소프트웨어 관점에서 본 내장형 시스템의 테스트 프로세스

3. 테스트 기법

Safety-Critical Embedded System과 같이 운영체제가 있는 내장형 시스템에 대하여, 2장에서 기술된 내장형 시스템의 테스트 프로세스에 있는 활동에 해당하는 각 테스팅 기법을 기술한다.

3.1 운영체제 테스팅

Safety-Critical Embedded System에 들어가는 운영체제는 V&V(Verification and Validation) 프로세스에 의해 개발되며, V&V 프로세스로부터 산출되는 산출물을 활용하여 테스팅 한다[2].

3.1.1 단위 테스팅

단위 테스팅은 코드 기반 테스팅과 대상 시스템을 명세한 Statechart들을 대상으로 한다.

(1) 코드기반 테스팅

IEEE Std. 1008-1987에서는 Safety-Critical Embedded System에 탑재되는 코드 기반 테스팅을 위한 테스트 기준에 대한 지침을 제공하고 있다[3]. IEEE Std. 1008-1987에서는 Safety-Critical Embedded System에 들어가는 모든 코드는 Statement 커버리지(Coverage) 기준을 적용해야 한다고 명시되어있으나, Statement 커버리지 기준만으로는 프로그램 내의 모든 분기(Branch)를 커버할 수 없기 때문에, 분기 커버리지 기준도 적용한다.

(2) Statechart 테스팅

Statechart로 이루어진 운영체제의 설계 단계 명세(Design Specification)로부터 다음과 같이 두 가지 방법으로 단위 테스팅을 위한 테스트 데이터를 선정한다. 첫째는, Statechart의 모든 상태를 커버하는 상태 테스팅을 수행하며, 둘째는 명세 된 Statechart의 행위(Behavior)를 나타내는 시뮬레이션 시나리오에 따른 경로(Path) 테스팅을 수행한다[4,5].

3.1.2 통합 테스팅

통합 테스팅은 단위 테스팅의 대상이 되었던 유닛(Unit)들을 통합한 모듈(Module)을 테스팅 하는 것으로, 설계 명세를 활용한다. 단위 테스팅의 대상이 되었던 Statechart를 모듈로 통합함으로써, 통합된 모듈들간의 인터페이스에 초점을 두는 것이 통합 테스팅의 목적이다.

통합 테스팅의 테스트 데이터 선정을 위하여, Statechart들로 이루어진 유닛들을 재구성한 상태 구성도를 제안한다. 상태 구성도란 통합 테스팅을 위하여, 단위 테스팅의 대상이 되었던 유닛들을 의미 있는 단위로 재구성 하여, 모듈간의 통신이 제대로 이루어졌는지를 확인한다.

통합 테스팅을 위한 테스트 데이터 선정 과정은 아래의 두 단계를 거치며, 통합 테스팅을 위한 상태 구성도의 기호 및 작성 방법은 아래와 같으며, 그림 2에서는 통합 테스팅의 대상이 되는 커널과 태스크 모듈에 대한 상태 구성도의 예를 표현한다.

단계 1. 노드와 에지로 구성된 상태 구성도를 작성한다.
단계 2. 시나리오에 따라 상태 구성도의 대상 및 전이를 커버 할 수 있는 Branch Coverage 기준을 적용한다.

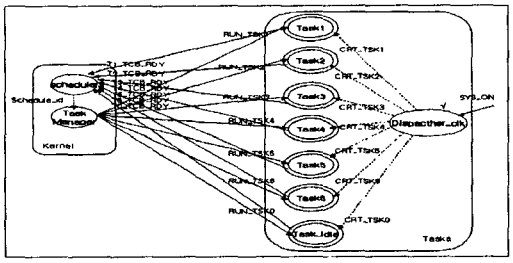

그림 2. 커널과 태스크 모듈에 대한 상태 구성도

통합 테스팅을 위한 상태 구성도의 기호 설명 및 상태 구성도의 작성 방법은 다음과 같다.

[상태 구성도 기호]

상태 구성도는 노드와 에지로 구성하며, 각각에 대한 설명은 아래와 같다.

- 노드
  - 유닛: 타원으로 표시한다.
  - 모듈: 유닛들의 집합이며, 둥근 사각형으로 표시한다.
  - 초기 유닛: 유닛에 체크를 표시한다.
  - 최종 유닛: 이중 타원으로 표시한다.
- 에지
  - 점선: 모듈 내 유닛간의 관계를 표현한다.
  - 실선: 모듈들간의 관계를 표현한다.

[상태 구성도 작성 방법]

통합 테스트 데이터 선정을 위한 상태 구성도 작성 방법은 아래와 같다.

① 모듈로 통합할 유닛들을 재구성한다.
② ①에서 식별된 유닛들을 재구성하여 모듈로 통합하고, 모듈 내에 있는 유닛들간의 관계를 표현한다.
③ ②에서 식별된 모듈들간의 관계를 표현한다.

3.1.3 시스템 테스팅

요구사항 명세(Requirement Specification)에 기술된 요구대로 기능이 구현되었는지 확인하기 위하여 시스템 테스팅을 수행하며, Activity chart를 활용한다. 시스템 테스팅을 위한 테스트 데이터 선정 방법은 아래와 같다.

단계 1. Activity chart로부터 시나리오를 추출한다.
단계 2. 시나리오에 따른 순서도를 작성한다.
단계 3. 순서도의 각 흐름(Flow)에 따른 테스트 데이터를 선정한다.

[Activity chart를 순서도로 바꾸는 방법]

① Activity chart의 각 모듈이 순서도의 클래스가 된다.
② Activity chart의 이벤트(Event)와 데이터는 순서도의 메시지가 된다.

3.2 어플리케이션 소프트웨어 테스팅

운영체제 테스팅에서 수행한 코드 기반 테스팅과 같이, 대상 시스템에 탑재할 어플리케이션 소프트웨어에 대한 코드 기반 테스팅을 수행한다.

3.3 인터렉션 테스팅

3.1절과 3.2절의 테스팅을 수행한 후, 내장형 시스템에 탑재되는 어플리케이션 소프트웨어와 하드웨어 사이의 인터렉션 사이에서 발생 할 수 있는 오류들을 발견할 수 있는 테스트 데이터 선정 기법이 필요하다[6].

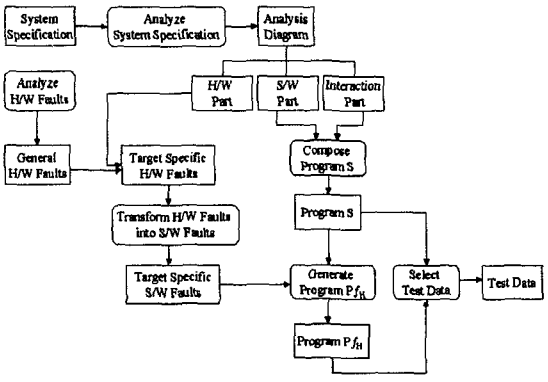

그림 3. 인터렉션 테스팅

그림 3에서는 인터렉션 테스팅을 활동과 활동별 산출물을 각각 둥근 사각형과 사각형으로 나타내었다. 그림 3에서 보는 바와 같이 인터렉션 테스팅은 대상 시스템의 동작을 시뮬레이트 한 프로그램 S를 작성한다. 그리고 하드웨어 오류를 소프트웨어 오류로 전환하여 프로그램 S에 삽입함으로써 프로그램 Pf_H를 생성한 후, 프로그램 S와 프로그램 Pf_H의 결과값을 차별화 하는 입력 데이터를 테스트 데이터로 선정 한다.

4. 결론

내장형 시스템의 기능이 복잡해지면서, 내장형 소프트웨어에 대한 테스팅이 중요하게 인식되고 있다. 특히 원자력 발전소 계통 기기와 같이 안전에 대한 치명도가 높은 Safety-Critical Embedded System일수록 탑재되는 내장형 소프트웨어에 대한 철저한 테스팅이 요구되기 때문에 체계적인 테스트 프로세스를 제안하였다.

내장형 시스템은 운영체제, 어플리케이션 소프트웨어, 하드웨어로 조합된 시스템으로 각 요소에 대한 테스팅이 요구된다. 특히, Safety-Critical Embedded System에 들어가는 운영체제, 소프트웨어 및 하드웨어는 V&V 프로세스에 의해 개발되며, V&V 프로세스로부터 산출되는 산출물을 활용하여, 테스트 데이터를 선정하였다. 시스템에 탑재되는 운영체제의 경우 운영체제의 요구사항 명세를 활용한 테스팅 기법과 설계 명세를 활용한 테스팅 기법을 제안하였으며, 어플리케이션 소프트웨어를 포함한 코드 기반 시험의 경우 원전 규제지침에서 명시한 바와 같이 화이트박스 테스트 기준을 적용하였으며, 마지막으로 하드웨어와 어플리케이션 소프트웨어 사이의 인터렉션으로 인한 오류는 쉽게 감지하기 어렵기 때문에, 이들 사이의 오류를 발견할 수 있는 테스트 데이터 선정 기법을 제안하였다.

향후에는, 일반적인 내장형 시스템으로의 확장을 위하여, 각 기법에 대한 실험 및 사례 연구를 통하여 기법의 효율성을 나타내고, 이를 자동화 할 수 있는 도구를 구현할 예정이다.

감사의 글

본 연구는 "원자력 연구개발 중장기사업인 원전계측제어 사업단"에 의해 지원되었음.

5. 참고 문헌

[1] Jean J. Labrosse, *MicroC/OS-II*, R & D Books, 2001.
[2] IEEE Std. 1012-1998 IEEE Standard for Software Verification and Validation, IEEE Computer Society, 1998.
[3] IEEE Std. 1008-1987, IEEE Standard for Software Unit Testing, IEEE Computer society, 1987.
[4] Beizer, B., Black Box Testing: techniques for functional testing of software and systems, John Wiley & Sons Inc.,1995.
[5] S.Burton, J.Clarck, J.Mcdermid, " Automatic Test Generation From Statechart Specifications" , *Proc. The Formal Approaches to Testing of Software (FATES' 2001)*, Denmark, pp 31-46, 2001.
[6] Ahyoung Sung, Byoungju Choi, "Interaction Testing in an Embedded System using Hardware Fault Injection and Program Mutation," *Formal Approaches to Testing of Software(FATES' 2003)* accepted, Canada, Oct. 2003.