# 정형기법을 이용한 Safety-Critical System 개발 방법론


ᵘ성창훈* 이나영** 오승록*** 최진영*

고려대학교 컴퓨터학과*

서울대학교 원자력공학과**

단국대학교 전자공학과***

{chsung, choi}@formal.korea.ac.kr*

grasia2@snu.ac.kr**

ohrk@dankook.ac.kr


# Development Methodology of Safety-Critical System Using Formal Method


ᵘSung, ChangHun* Lee, Na-Young** Oh, Seoungrohk*** Choi, Jin-Young*

Dept. of Computer Science and Engineering, Korea University*

Dept. of Nuclear Engineering, Seoul National University**

Dept. Electronics Engineering, Dankook University***



**요 약**

본 연구는 정형기법을 사용하여 Safety-Critical System의 개발 방법론을 제시한다. Safety-Critical System의 전체적인 개발 과정을 제시하고 Safety-Critical System 중의 하나인 원자력 발전소 시스템 중 Reactor Protection System(RPS)을 정형 명세(Formal Specification)하고 정형 검증(Formal Verification)하는 과정과 그에 따른 각 과정의 Compliance를 확인하는 예를 든다. 여기서 정형 명세에는 Software Cost Reduction(SCR)이라는 도구가 사용되었고, 정형 검증에는 SPIN이, Compliance를 확인하는 데에는 Prototype Verification System(PVS)를 사용하였다.


## 1. 서론

오늘날 하드웨어와 소프트웨어 시스템은 넓게 사용되고 있다. 특히 전자 상거래, 전화 교환 네트워크, 지하철과 항공기 제어 시스템, 의료 기계 등 그 실패를 용납하지 않는 곳에서도 많이 사용되고 있는 실정이다. 특히 하드웨어와 소프트웨어 시스템은 규모면 이나 기능 면에서 점점 커지고 복잡해지고 있으며, 이러한 복잡성의 증가로 인하여 시스템에서의 에러 발생 가능성은 더욱 커지고 있다. 실수 변환에러로 인한 Arian 5 rocket의 폭발, 덴버 공항의 지하 화물 처리 시스템의 실패, 인텔 펜티엄 프로세서의 FDIV 명령어 오류 등에서 알 수 있듯이 이러한 에러들은 비용, 시간, 인력 면에서 엄청난 대가를 치러야 했으며, 심지어 인간의 생명에도 위협을 주었다. 이러한 이유로 정형 기법(formal method)의 중요성이 강조되었다.

정형 기법은 수학과 논리학에 가반을 둔 방법으로 하드웨어나 소프트웨어 시스템을 명세하거나 검증하는 것으로, 이러한 정형기법에는 크게 정형 명세(formal specification)와 정형 검증(formal verification)이 있다. 정형 명세는 시스템을 요구 명세할 때 자연어로 기술하면서 발생하는 모호성과 불완전성으로 인한 오해나 허점을 정형 논리(formal logic) 또는 수리 논리(mathematical logic)에서 사용되는 기호 등을 이용하여 시스템이 동작할 환경에 대한 가정, 시스템이 만족해야 하는 요구 사항, 그리고 요구 사항을 수행할 시스템 설계 등을 기술하는 것이다. 또한, 정형 검증은 정형 논리 또는 수리 논리에서 제공하는 증명 방법 등을 이용하여 정형 명세를 분석하여 무모순성 및 완전성을 검증하거나, 설계가 주어진 가정에서 요구사항을 만족하는지를 검증하는 기법이다.

본 논문에서는 앞에서 말한 실패를 용납하지 않는 시스템, 즉 safety-critical system의 개발 방법에 있어서 어떻게 그 시스템을 명세하고 검증할 것인가에 대해 이야기한다. 2장에서는 safety-critical system의 개발 과정과 각 과정들의 검증방법에 대해 이야기하고 3장에서는 실례를 들어 어떻게 safety-critical system을 명세하고 검증하는지에 대해 알아보고 4장에서는 결론 및 향후 과제에 대해 제시한다.

## 2. Safety-Critical System의 개발 방법

Safety-critical system은 실패를 용납하지 않는 시스템이다. 즉, 의료 기계나 교통 통제 시스템, 스마트 자동차(smart vehicle), 원자력이나 군사 방어 어플리케이션 등이 이에 속한다. 이러한 시스템들에서 실패가 발생하였을 경우, 사람의 생명에 치명적인 결과를 가져 올

수 있기 때문에 그 개발에서부터 시스템이 운용되기까지 많은 인적, 물적 자원을 투자한다. 이 때문에 safety-critical 시스템의 개발 과정에 정형기법의 사용이 중요한 이슈로 등장하였다. 개발 시작부터 끝까지 정형 기법을 사용하여 단계별로 정형 명세하고 정형 검증을 하여, 시스템의 property를 확인하고 verification과 validation 과정을 거치는 것이다.

Safety-critical system의 개발 방법은 여러 가지 소프트웨어 개발 방법론에 접목시킬 수도 있으나 전체 순서는 그림 1과 같은 순서로 진행이 된다.

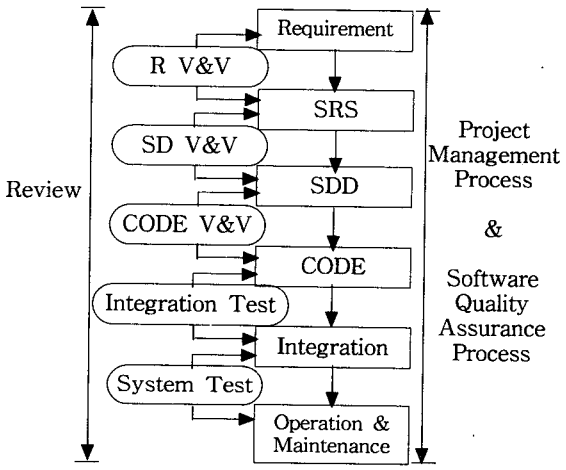


그림 1 Safety-Critical System의 개발 과정

그림에서 SRS는 Software Requirement Specification이고, SDD는 Software Design Description을 말한다. V&V는 Verification and Validation이고, R V&V와 SD V&V는 각각 Requirement V&V와 Software Design V&V이다[3],[4].

Safety-critical system의 개발은 각 단계별로 진행되지만 전체 개발 단계는 management 과정과 assurance과정이 동시에 수행된다. 그리고 각 단계로 진행되는 동안 그에 알맞은 검증과정이 필요하고 필요하다면 바로 전 단계로 돌아가 작업을 다시 하기도 한다.

## 3. Reactor Protection System(RPS)

### 3.1 Reactor Protection System

Safety-critical system에 하나로 원자력 발전소 시스템이 있다. 그 시스템 중에 Reactor Protection System (RPS)은 빠른 시간 안에 원자로 트립을 시켜야 하는 주요 기능을 담당하고 있다.

RPS는 7개의 신호를 받아 하나의 출력을 하는 시스템이다. 이는 원자로의 각 상태를 실시간으로 체크를 하며, 단 하나의 신호라도 지정된 값을 벗어나면 경보를 울리게 되는 시스템이다. 이 때문에 원자로의 상태를 안전하게 유지하기 위한 필수 장치로 정형 기법을 사용한 개발이 필수적이다.

### 3.2 RPS의 요구 명세

RPS의 이해를 돕기 위해 RPS의 요구 조건 사양을 자연어로 된 가상의 기능요구조건을 기술한 것은 다음과 같다.

> 계통은 7개의 센서로 그에 따르는 경보장치로 구성되며, 센서의 출력결과에 따라 경보장치가 작동한다. 센서_1, 센서_2, 센서_3, 센서_4, 센서_5, 센서_6, 센서_7 은 7개의 입력을 동시에 받아서 각각의 값이 각각의 기준을 만족하는지의 여부를 판단한다. 기준을 만족하는 경우는 경보장치에 영향을 주지 않은 상태로 작동하며, 7개의 입력중 하나라도 기준을 만족하지 않는 경우에는 경보장치를 작동시켜 경보를 울리게 된다.

정리하면,

- 7개의 입력을 받아서 입력 값을 확인한다.
- 입력은 전단의 신호처리를 거쳐 T, F의 boolean 값을 가지게 된다.
- 입력이 F의 값을 가지는 경우는 다음 입력을 확인하고, T의 값을 가지면 경보장치에 신호를 주어 경보를 울리게 한다.

위 명세를 그림으로 간단히 표현하면 다음과 같다.

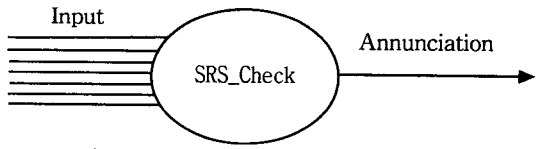


그림 2 Functional overview diagram

### 3.3 방법론 적용 사례

RPS는 원자력 발전소 시스템에서 아주 주요한 기능을 담당하고 있기 때문에 이는 safety-critical system으로 분류되고 엄정한 정형기법에 의한 V&V가 요구된다. 이 논문에서는 여기서 사용되는 RPS를 가지고 그에 대한 SRS, SDD에 대해 알아보고 정형 명세 및 정형 검증 방법과 SRS와 SDD 사이의 compliance check에 대해 알아본다.

정형 기법을 사용한 Requirement와 SRS, SDD 사이의 관계를 살펴보면 그림 3과 같이 나타낼 수 있다.

Requirement는 자연어로 많이 기술되어 있기 때문에 그 모호성과 불완전성이 많이 존재하게 된다. 그렇기 때문에 정형 명세를 통해 SRS로 표현되어 지고, 이때 requirement와 SRS 간의 completeness와 consistency 검사가 꼭 필요하다. 이러한 기능을 해주는 정형 명세 도구는 많이 있으나 RPS의 특성을 고려할 때 자연어 요구 사항을 table로 표현 할 수 있는 Software Cost

Reduction(SCR)[5]을 이용한다. 또한 그 명세를 검증하기 위해 model checking방법인 SPIN[6]을 이용하는데 그 이유는 SPIN이 자동으로 SCR 명세 언어를 SPIN 검증 언어로 바꾸어 주기 때문이다. 다시 말하면, 자연어로 기술되어 모호하고 불완전한 Requirement를 SCR로 completeness와 consistency를 검사하고, 또한 그 property를 검사한다. 이를 SPIN을 사용하여 정형 검증을 하는 것이다.

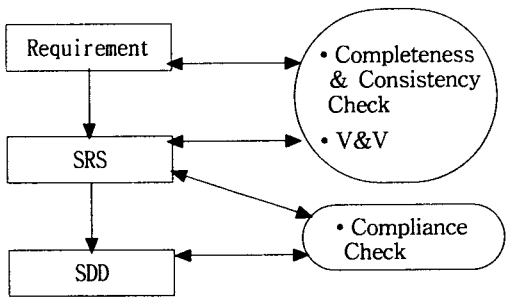


그림 3 Safety-Critical System 개발 방법론

자연어의 모호성과 불완전성을 해결하고 나면 system을 설계를 시작하게 된다. 물론 이때도 설계에 대한 completeness와 consistency 검사와 property검사가 필요하다. 때문에 이를 다시 SCR로 명세하고 SPIN으로 검증을 하게 된다. 하지만 이렇게 정형 명세와 검증을 하였다 하더라도 그것은 SDD에 관한 명세와 검증일 수밖에 없기 때문에 SRS와 SDD간의 compliance를 검사해야 한다. 즉, SRS에서 SDD로 변환되면서 SRS와 SDD가 서로 일치하는지에 대해 확인할 필요가 있다는 것이다. 이에 Prototype Verification System(PVS)[7]를 통해 PVS언어로 SRS와 SDD를 변환하고 theorem prover 등의 자동화된 도구를 사용하여 그 일치성을 확인할 수 있다.

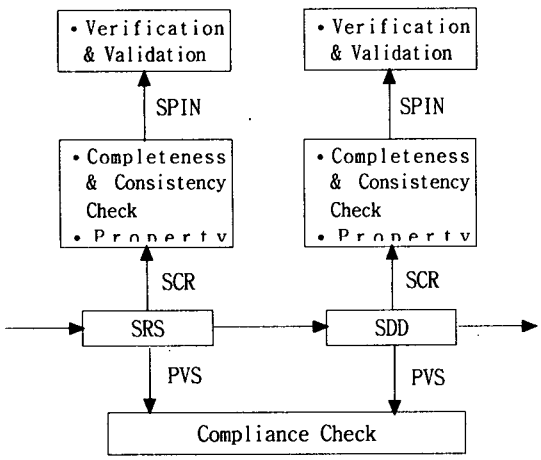


그림 4 Compliance Check

그림 4는 SRS와 SDD의 정형 명세 및 정형 검증과정과 SRS와 SDD사이의 compliance check과정을 도식화한 그림이다.

이러한 방법은 사용하면 자동화된 다양한 정형 기법을 적용하기 때문에 빠른 시간에 여러 특성들을 여러 방면으로 검토할 수 있고, SRS와 SDD와의 일관성까지 확인할 수 있기 때문에 각 단계에서 오류가 없음을 효율적으로 증명할 수 있다.

## 4. 결론

이렇듯 safety-critical system은 실패를 용납하지 않는다. 그렇기 때문에 그 개발 과정에서부터 시스템이 운용될 때까지 많은 인적 물적 자원을 투자한다. 이 때문에 safety-critical system의 개발 과정에 정형 기법의 사용이 중요한 이슈로 등장하였다. 개발 시작부터 끝까지 정형 기법을 사용하여 단계별로 정형 명세하고 정형 검증을 하여, 시스템의 property를 확인하고 V&V과정을 거치는 것이다.

본 연구는 safety-critical system의 개발에 있어서 정형 기법이 어떻게 적용될 수 있는지를 살펴보고, 정형 기법을 이용한 safety-critical system의 개발 방법론을 제시한다. 또한 safety-critical system중의 하나인 원자력 발전소 시스템 중 RPS를 정형 명세(SCR을 사용)하고 정형 검증(SPIN을 사용)하는 과정과 그에 따른 각 과정의 Compliance를 확인(PVS로 변화하여 theorem prover 사용)하는 예를 들었다.

향후 과제로서는 본 연구에서 언급하지 못한 다른 과정에도 적용시켜보는 것이다. 또, 본 연구에서 제시한 방법론을 다른 safety-critical system에도 적용시켜 보는 것이다.

## 5. 참고 문헌

[1] Edmund M. Clark and Jeannette M. Wing, "Formal methods : State of the Art and Future Directions", *ACM Computing Surveys*, Dec. 1996.

[2] Robyn R. Luts, "Software Engineering for Safety : A Roadmap", *ACM The Future of Software Engineering 2000*, p215~p224, Apr. 2000.

[3] Dolores R. Wallace and Laura M. Ippolito, "A Framework for the Development and Assurance of High Integrity Software", *NIST 500-223*, Dec. 1994.

[4] Ian Sommerville, "Software Engineering", *Addison-Wesley*, 1996.

[5] James Kirby, "SCR* Toolset : The User Guide", Jan. 1997.

[6] Gerald J. Holzmann, "The Model Checker SPIN", *IEEE Transactions on Software Engineering, VOL. 23., pp 279~295*, May. 1997.

[7] S. Owre and N. Shankar, "Abstract Datatypes in PVS", *Technical Report CSL-93-9R*, Dec. 1993.