2013. 7. 29. [제61호]

프로젝트 고신뢰성 확보를 위한 정형 기법 가이드라인

박사천(Sachoun Park), 김태호(Taeho Kim), 임채덕(Chadeok Lim)
Korea Conference on Software Engineering 2013

한국전자통신연구원 임베디드SW연구부

Contents

Ⅰ. 소개
Ⅱ. DO-178의 테스팅과 정형기법
Ⅲ. 정형 기법 고려사항
Ⅳ. 정형 검증 커버리지 분석
Ⅴ. 결론

정보통신산업진흥원 부설
nipa 소프트웨어공학센터

Ⅰ. 서론

자동차의 엔진 제어 시스템이나 군용 장비의 임베디드 시스템과 같이 고신뢰성을 갖추어야 하는 소프트웨어의 개발은 국제 규격을 준수하도록 요구 받고 있다. 2011년 11월에 자동차용 기능 안전 표준인 ISO 26262가 공식 발표되었다. 여기서 시스템에 대한 안전 요구를 네 개의 등급으로 제시하고 있다. ISO 26262 - Part 6는 소프트웨어 개발 프로세스를 정의하는데, 요구사항 작성 단계에서부터 모든 등급에 준정형(semi-formal) 기법의 적용을 권고하고 있다. 표준에서 언급된 준정형 모델이란 실행 가능한 모델을 의미한다[1]. 또한 같은 해 12월에는 항공용 소프트웨어 인증 표준인 DO-178B[2]의 다음 버전으로 DO-178C가 발표되었는데, 항공용 소프트웨어에 정형 기법 적용 표준으로 DO-333(Formal Methods Supplement to DO-178C and DO-278A)이 함께 발표되었다[3]. 이렇듯 최근에 발표되는 산업용 표준들이 정형 기법을 안전성 보장의 핵심 방법으로 공식화하는 것에는 지난 30여 년 간 축적된 정형 기법 기술이 산업에 적용되기 충분하다는 방증이라 할 수 있다.

그러나 프로젝트에 전격적으로 정형 기법을 적용하기에는 많은 어려움이 있다. 특히 정형 기법의 기술적 한계 즉, 다룰 수 있는 문제의 크기가 실제 시스템에 비해서 매우 작다는 것과 기술의 난이도가 일반 엔지니어들이 쓰기에는 결코 쉽지 않다는 것이 정형 기법의 산업 확산을 저해하는 두 가지 큰 걸림돌이다. 이러한 문제들은 요약해석에 의한 소스코드 검증[4], 정리 증명의 자동화[5] 개선 등 그 자체의 한계를 극복하려는 노력들에 의해서 상당부분 개선되고는 있으나 아직까지 흔쾌한 동의를 얻어내지는 못한 상황이다.

본 원고에서는 위와 같은 정형 기법의 기술적 한계와는 다른 측면에서 정형 기법의 한계를 극복해보고자 한다. 논의될 사항은 프로세스 내에서 정형 기법을 어떻게 적용할 것인가에 대한 물음에 대한 답이 될 것이다. 서두에 소개했던 두 표준과 같은 인증을 준비하는 프로젝트가 있다고 가정하면, 당장 어떻게 정형 기법을 적용할 것인가라는 난제에 부딪히게 된다. 물론 선진 외산 도구들을 중심으로 이러한 문제에 대한 솔루션들이 선전되고 있지만[6,7], 보다 학문적이고 도구에 독립된 접근법이 제시되어야 함은 분명하다. 우리는 이 문제를 풀어갈 시초로써 테스팅을 주목한다. 테스팅은 이미 산업에 적용되어 반드시 수행되어야 할 절차로 인식되고 있다. 특히 DO-178B에 특화된 테스팅

기법은 초기 항공용 소프트웨어의 안전성을 담보하는 중요한 수단이 되어왔는데, 우리는 여기에 주목할 것이다. 비록 DO-178B와 및 그 테스팅 기법들이 항공용 소프트웨어를 위한 표준이지만 고신뢰성을 요구하는 다른 산업 및 프로젝트에 대한 테스팅 적용도 크게 다르지 않을 것으로 확신한다.

II. DO-178의 테스팅과 정형 기법

우리 팀은 2010년 10월에 시작된 WBS 1차 사업을 통해서 무인항공기에 탑재되는 운영체제를 개발했다. 이 과제에서 세운 목표 중 하나가 OS 커널에 대한 DO-178B 레벨 A 인증이었다. DO-178B는 RTCA 사에 의해서 발표된 민간 항공기의 소프트웨어 인증 표준이다. 실제 인증은 미국 연방 항공국에서 지정한 DER(Designated Engineering Representative)에 의해서 진행된다. 그림 1은 인증의 전체 개관을 나타낸다. 한마디로 DO-178B 인증은 계획대로 개발이 진행되었는지 확인하는 작업이라고 할 수 있다. 따라서 상세한 계획과 엄격한 품질관리 절차를 중시한다. 5가지 계획(인증, 개발, 검증, 변경 관리, 품질 관리)과 3가지 표준(요구사항, 설계, 코딩)을 근간으로 개발이 진행되며, 모든 변경 절차는 이슈로부터 유도된다. 모든 활동의 결과물은 철저한 리뷰 및 감사를 통해서 그 정당성을 부여 받게 된다.

그림 1_DO-178B 인증 개관

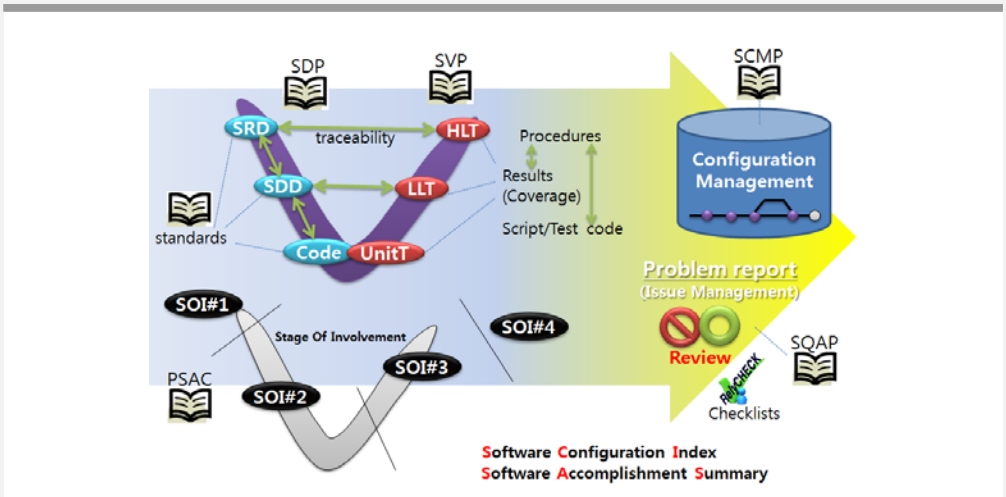


그림 1의 왼쪽 하단의 V-모델은 인증 단계를 나타낸 것이다. 인증은 보통 네 번의 단

계 중(계획, 설계, 검증, 배포) 혹은 각 단계 완료 후 이루어지며, 이때 품질 관리자의 품질 활동이 DER에 의해서 감사 된다. 프로젝트 진행에서 생산되는 수 많은 문서 중에 마지막 SOI(Stage Of Involvement)에서 중시되는 산출물은 배포 버전의 소프트웨어 형상에 대한 인덱스 문서(SCI)와 개발 과정 중에 수행된 인증 노력을 인증 계획에 대비해 요약한 문서(SAS)이다.

1. DO-178B 테스팅

철저한 절차 준수와 함께 중시되는 부분이 바로 테스팅 활동이다. 우리는 레벨 A를 목표로 했기 때문에 독립된 기관을 통한 테스팅을 수행해야 했고 MC/DC 커버리지 100%를 달성해야 했다. DO-178B의 검증은 테스팅 수행과 테스트 케이스 및 절차, 그리고 수행 결과에 대한 엄격한 분석 및 리뷰로 이루어진다. 테스팅 계획이라고 하지 않고 검증 계획이라고 한 것은 바로 분석과 리뷰를 중시한다는 의미를 강조하기 위함이다. 리뷰는 앞서 설명한 것과 같이 각 단계마다 특화된 체크리스트를 이용하게 되며, 분석은 커버리지의 다른 말이다.

그림 2_DO-178B 테스트/분석 프로세스[8]

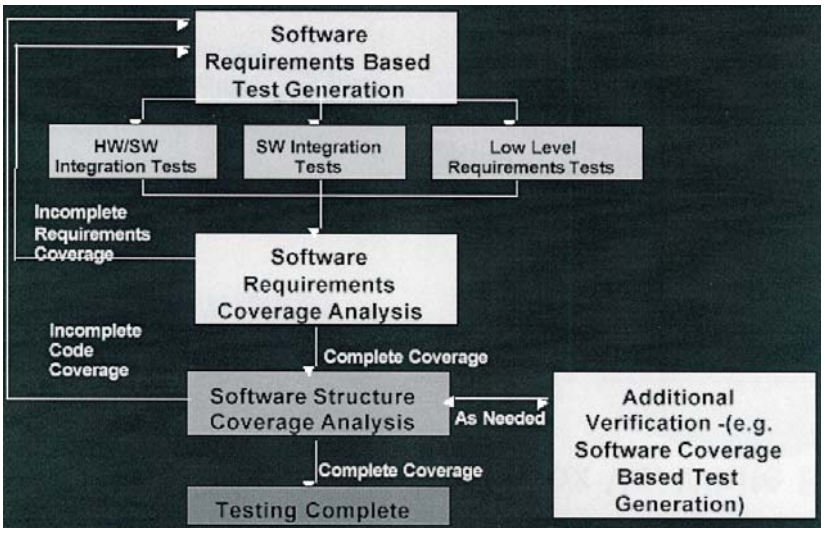


그림 2는 DO-178B 테스트/분석 프로세스를 보여준다. DO-178B 에서는 요구사항 단계의 결과물을 상위 요구사항(High Level Requirement), 설계 단계의 결과물을 하위 요구사항(Low Level Requirement)라고 하고 그림 1에서와 같이 각 단계에 대한 테스팅을 HLT와 LLT라고 명명한다. 그림 2의 과정은 위의 두 단계 테스팅을 함께 나타낸 것이다. 테스

트를 수행하고 수행된 테스트 활동을 검증하기 위한 수단으로 요구사항 커버리지를 분석한다. 모든 요구사항이 테스팅 되었는지 확인하는 것으로 각각의 요구사항들이 Normal Range와 Robustness를 테스팅 했는지 살펴보게 된다. 그런 후 구조적인 커버리지를 분석하는데, 여기에서 MC/DC 100% 커버리지가 요구된다. MC/DC커버리지는 HLT와 LLT에서 각각 측정된다. 만약 위의 두 테스트에서 측정된 커버리지가 부족하다면 유닛 테스트 케이스로써 100% 커버리지를 충족하도록 권고된다.

여기서 MC/DC커버리지는 구현된 소프트웨어가 요구사항들을 충분히 만족하는지 확인하기 위한 방법으로 사용된 것인데, 단순히 철저한 테스팅 만을 의미하는 것은 아니다. 이는 상위 요구사항과 하위 요구사항이 잘 작성되었는지, 추적성이 제대로 관리되는지 여부를 우회적으로 드러내는 수단이 된다. 즉, 상위 요구사항으로부터 작성된 HLT 테스트 케이스와 하위 요구사항으로부터 작성된 LLT 테스트 케이스가 각각 또는 연합하여 MC/DC를 얼마큼 커버하는지 확인함으로써 요구사항 작성 및 테스트 케이스 작성이 충분히 되었는지 정량적으로 분석할 수 있고, MC/DC 수행 후 100% 커버되지 않았을 때, 데드 코드를 없애기 위한 유닛 테스트 케이스의 생성 회수 및 규모 그리고 회기 테스트 절차를 분석함으로써 시스템이 얼마나 강건하게 개발되었는지 확인할 수 있게 된다.

2. DO-178C 정형 기법

DO-178C에서 정형 기법은 정형 모델로 수행되는 정형 분석이라고 정의된다. 즉, 개발 프로세스에서는 정형 모델이 적용되고 검증 프로세스에는 정형 분석이 적용된다. 이른바 모델이란 분석과 시뮬레이션 및 코드 생성에 사용하기 위해 특징되는 소프트웨어의 추상 표현이라고 할 수 있다. 모델이 정형적이기 위해서는 명백하고 수학적인 구문과 의미로써 정의되어야 한다. 인증 맥락에서는 정형적인 모델이어야 자동 분석의 가치가 부여된다. 정형 분석이 신뢰성을 얻기 위해서는 분석 방법의 안전함(soundness)이 확립되어야 한다. 안전한 분석이란 참이라는 결론에 대해서 의심의 여지가 없어야 함을 의미한다. 수 많은 형태의 정형 분석들이 존재하지만, DO-178C에서는 (1) 정리 증명(Theorem Proving)과 같은 연역적 기법, (2) 모델 체킹(Model Checking), 그리고 (3) 요약 해석(Abstract Interpretation)으로 분류한다.

그림 3은 DO-178C 레벨 A 검증 프로세스를 나타낸 것이다. 정형 분석은 굵은 점선으로 표시된 리뷰/분석 활동에 적용될 수 있다. 정형 분석이 타당하기 위해서는 정형 언어에 대한 수학적 구문과 의미가 정의되어야 하고, 분석 방법이 안전해야 하며, 분석에 사용된 가정들에 대한 정당성이 제시되어야 한다.

그림 3_DO-178C 레벨 A 검증 프로세스[3]

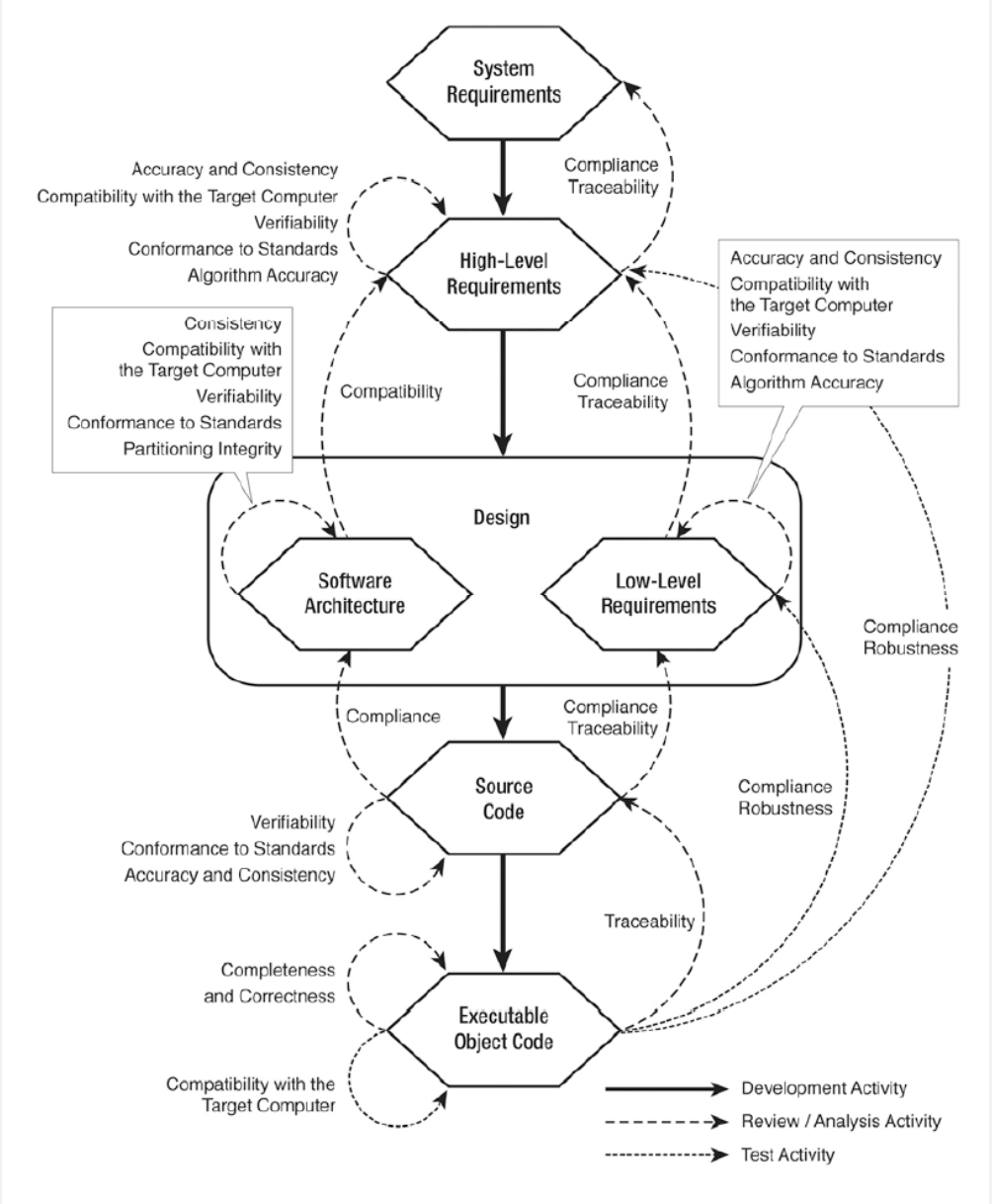


그림 3에서 언급된 분석 및 리뷰 활동을 정형 기법 관점에서 기술하면 다음과 같다:

1) 준수성(Compliance): 준수성은 개발 활동의 입출력에 대해서 출력이 입력을 만족하는가에 대한 질문이다. 따라서 설계가 요구사항을 만족하는지, 코드가 설계를 만족하는지 정형 검증하는 활동으로 기존의 리뷰를 대체할 수 있다.
2) 정확도(Accuracy): 정형 표기법은 정확하고 분명해야 한다. 이것은 앞서 언급한 정형적 구문과 의미의 정의, 분석 기법의 안전성 확립, 그리고 가정의 정당성이 전제되

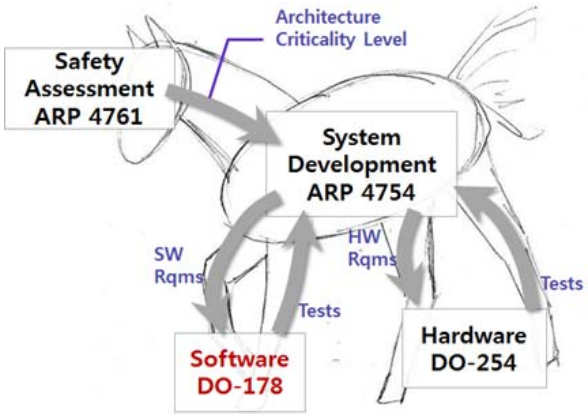
어야 한다. 분석 기법의 안전성은 학계에서 인정되는 논문에 대한 참조로써 대치될 수 있다. 분석에 사용된 가정의 정당성을 제시하는 문제는 매우 어렵고 경우에 따라 각기 다른 방법이 적용될 수 있다.

3) 일치성(Consistency): 요구사항이나 설계에서 상충되는 부분의 존재 유무는 다른 분석에 앞서 검증되어야 할 요소이다.
4) 타겟 컴퓨터 적합성(Compatibility with the target computer): 타겟 컴퓨터와의 개발 산출물 간의 적합성이 분석되기 위해서는 타겟 컴퓨터에 대한 정형 모델링이 전제되어야 한다.
5) 검증 가능성(Verifiability): 요구사항이 정형화 가능하다면 정형 분석을 통해서 검증될 수 있다. 또한 '항상' 또는 '결코'와 같은 의미가 첨가된 요구사항에 대해서는 테스팅 보다 정형 분석이 더 적합한 검증 방법이다.
6) 규격 적합성(Conformance to standards): 정형 표기법으로 기술된 정형 모델과 속성은 해당 구문에 맞게 작성되었는지 검사할 수 있는 구문 검사 도구가 제공되어야 한다. 또한 자동화된 검증 도구를 사용했을 경우 그 결과를 신뢰하기 위해서는 리뷰 등의 부가적인 방법이 사용되어야 한다.
7) 추적성(Traceability): 프로세스의 입력으로부터 출력으로의 추적성은 각 출력이 어떤 입력을 만족한다는 것을 정형 분석함으로써 입증할 수 있다. 그러나 시스템 요구사항과 정형 요구사항 및 모델 그리고 검증 결과 사이의 추적성 유지는 또 다른 문제이다.
8) 알고리즘 측면(Algorithm aspects): 알고리즘이 정형 모델 된다면 정형 분석 가능하다.
9) 요구사항 정형화 정확성(Requirement formalization correctness): 자연어로 된 요구사항에 대한 정형화의 타당성은 어쩔 수 없이 리뷰에 의존할 수밖에 없다. 요구사항을 애초에 정형적으로 작성하면 모든 문제가 해결될 것이지만, 현실 프로젝트에 적용하기에는 무리가 있으므로, 자연어 요구사항의 의미를 충분히 반영할 수 있는 정형 요구사항 작성 가이드 및 리뷰를 돕는 체크리스트의 개발이 현실적인 대안이라고 하겠다.

Ⅲ. 정형 기법 고려사항

항공 시스템 관련 표준들의 역할이 그림 4에 표현되었다. 비단 항공 시스템뿐 아니라 고신뢰를 요구하는 모든 시스템에서 안전성 분석은 머리에 해당하는 매우 중요한 기술이기 때문에 이 단계에서도 정형 기법의 사용이 강력히 요구된다. 시스템을 모두 정형 분석하는 것은 많은 비용이 발생하므로 정형 분석 대상을 식별하는 작업은 반드시 선행되어야 한다. 그러나 문제의 범위를 좁히기 위해, 본 논문에서는 안전성 및 위험 분석이 완료되고 각 시스템 및 소프트웨어 수준까지 안전성 레벨이 할당 되었다고 가정하고, 소프트웨어에 대한 정형 분석을 어떻게 할 것인가에 대한 가이드에 초점을 맞추도록 한다. 논의될 가이드라인은 절차와 방법 그리고 탬플릿 제공이라는 목표를 가지며, [9]를 참조하였다.

그림 4_항공 시스템 관련 표준



1. 전통적인 개발 과정에서 정형 검증 고려 사항

요구사항으로부터 순차적으로 시스템을 개발할 때, 정형기법의 적용 역시 순차적으로 요구사항 정형화, 정형 모델링, 정형 검증으로 진행될 수 있다. 요구사항을 정형화할 때는 다음과 같은 단계를 생각해 볼 수 있다:

- 명제 추출: 입출력 변수 및 이벤트 등과 같은 명확하게 식별되는 용어로 구성되고 참/거짓이 분명한 문장을 유도하여 명제를 추출한다.

- 명제의 순서 결정: 시스템의 행위는 사건의 연속이므로 추출된 명제들 간의 시간적 순서를 찾는다.
- 타이밍 요소 결정: 타이밍 요소는 명시적으로 주어지기도 하지만 도메인에 따라 생략되거나 암시적으로 표현될 수 있다. 명시적이지 않을 경우 시간에 대한 상한과 하한 값을 얻어내야 한다.
- 가정 설정: 속성이 적용될 환경과 시스템의 범위를 분석해서 이에 대한 가정을 수립한다.
- 두리뭉실한 제약사항: 비 정형 요구사항은 간혹 중의적인 단어를 사용할 경우가 있다. 이러한 단어로써 제약사항이 정의되었다면, 더욱 구체적인 수치로써 요구사항이 정의되도록 유도해야 한다.
- 최적의 정형 표기법: 정의된 명제나 사용된 변수의 타입을 통해서 적절한 표기법을 선택한다.
- 결과 문서화: 해당 속성이 어떤 특징(under- or over-approximation)을 갖는지 그리고 예상되는 검증 결과 및 검증 방법 등이 기술되어야 한다. 또한 정형화 과정의 특이사항(명제화, 용어의 구체화)도 기술해야 한다. 만일 비 정형 요구상이 의미하는 바를 일부만 표현한다면, 그에 대한 내용과 이유도 같이 기술해야 한다.

정형 모델은 하위 요구사항과 시스템 아키텍처로부터 추출될 수 있다. 정형 모델링 과정을 통해서 설계 과정의 오류 개입을 최소화할 수 있고, 애매한 부분을 정제함으로써 구현 시간을 단축할 수 있다. 최소한의 정형 모델은 동작하는 모델이어야 한다. 요구사항이 정형화 되고, 정형 모델이 구축되었다는 전제하에 정형 검증을 위해서 다음과 같은 사항들이 고려될 수 있다.

- 반례가 제공된 경우: 요구사항이 만족되지 않으면 모델 체커로부터 반례가 제공된다. 반례에 대한 시뮬레이션을 통해서 모델 혹은 요구사항을 수정해야 한다.
- 정형 요구사항이 만족될 경우: 요구사항이 만족되는 경우에는 모델 체커가 특별히 돌려주는 결과가 없으므로, 결과의 신뢰성 확보 차원에서 반례가 예상되도록 속성을 변환하여 검증하는 기법(mutation technique)이 적용될 수 있다.
- 상태폭발이 발생한 경우: 모델과 속성을 검증 가능하도록 분할하거나 추상화 기법들을 적용해 볼 수 있다. 그러나 이러한 속성에 대해서는 테스팅에 의지하는 것이 현실적인 해결 방법이다.
- 결과의 문서화: 검증 속성과 모델에 대해서 기술하고 반례 및 반례 처리 결과 그리

고 추가적인 검증 사항에 대해서도 기술한다. 상태폭발이 발생한 경우에는 해당 속성에 대한 대안적 검증이 이루어졌는지에 대해서도 기술한다.

요구사항에 기반 한 속성 검사 외에 모델 체킹 도구를 통해서 설계 단계에서 다음과 같은 속성들이 검증될 수 있다.

- Reachability: 모델 내의 어떤 상태에 도달 하는가
- No useless transition: 모델링 된 전이는 모두 발생 가능 한가
- Determinism: 동일한 조건과 동일한 상황에서 두 개의 전이가 동시에 발생할 가능성이 있나
- No deadlock: 어떤 전이도 발생하지 않는 경우가 존재하나
- Recurrence: 각 상태들이 다른 모든 상태로부터 도달 가능 한가

2. 모델 기반 개발에서 정형 검증 고려사항

앞서 언급된 사항들은 전통적인 개발 방식을 전제한 정형 기법 가이드라인이었다. 서두에 언급된 두 표준에서 공이 강조되는 또 다른 기술은 모델기반 개발 방법론이다. 즉 개발 과정에 대한 자동화로써 오류의 개입을 줄이고 소프트웨어 생산성과 유지보수 능력을 높이려는 것이다. 다분히 툴 벤더들에 의해 유도된 결과라고 할 수 있지만, 시스템의 복잡성을 극복할 유력한 대안임은 확실하다. 따라서 정형 분석 도구들에 대한 적격 여부가 선행 심사되어야 한다.

그림 5_도구 자격 검정 수준

Software Level	Criteria 1	Criteria 2	Criteria 3
A	TQL-1	TQL-4	TQL-5
B	TQL-2	TQL-4	TQL-5
C	TQL-3	TQL-5	TQL-5
D	TQL-4	TQL-5	TQL-5

특히 DO-178C에서는 도구 자격 검정(Tool Qualification)이 세분화 되었다. 기존에는 도구들을 개발 도구와 검증 도구로 이분하고 개발 도구는 엄격하게 심사한 반면, 검증 도구는 비교적 낮은 수준에서 심사되었다.

새로 발표된 도구 자격 검정 기준은 그림 5와 같이 사용 범위와 소프트웨어 생명 주기에 미칠 잠재적 영향력을 기반으로 5 단계의 TQL(Tool Qualification Level)을 정의하고 있다. 그림에서 Criteria 1은 도구의 결과가 소프트웨어를 구성하는 일부로써 오류 개입의 가능성이 있는 것, Criteria 2는 검증 프로세스를 자동화하는 도구, 오류 발견에 실패할 수 있거나 개발 및 검증 과정을 생략하게 하는 도구, Criteria 3은 오류 발견을 실패할 수 있는 도구를 의미한다. 그리고 각 단계에 대한 자격 검정은 다음과 같은 수준으로 진행된다.

- TQL-1: DO-178 level A
- TQL-2: DO-178 level B
- TQL-3: DO-178 level C
- TQL-4: Complete Requirements, describe architecture, More Verification
- TQL-5: Requirement Based Verification

정형 검증 도구는 그림 5에서 붉은 색으로 표시된 것과 같이 레벨 A와 B에서 사용되고 개발 도구와 검증(테스팅) 도구의 중간 정도의 엄격함으로 심사되어야 한다는 의견이 있다[10]. 그러나 예를 들어 정리 증명기의 자격 검정을 위해 도구의 정확성을 검사한다는 것은 학문적 결과물들과 산업에 적용된 사례들에 의존될 수밖에 없다.

Ⅳ. 정형 검증 커버리지 분석

커버리지라는 용어는 적용 기준에 따라서 의미가 달라질 수 있다. 우선 DO-178과 같은 인증 표준의 검증 활동에 대한 커버리지로 해석할 수 있다. 즉, 전체 프로젝트의 검증 활동 중에 테스팅을 적용할 부분과 리뷰 또는 정형 기법을 적용할 부분에 대한 범위를 규정 문제로 볼 수 있다.

두 번째로 정형 기법을 사용함에 있어서 얼마큼 정형적으로 적용했는가를 질문할 수 있다. 즉 정형 분석 케이스, 프로시져, 그리고 결과가 얼마나 정확한가? 사용한 정형 기법은 안전한가? 요구사항 정형화는 정확한가? 등의 질문이 여기에 속한다. 경우에 따라서 부분적인(partial) 방법도 사용될 수 있을 것이다. DO-178에서는 테스트 케이스, 절차, 결과에 해당하는 카운트파트로써 각각 정형 분석 케이스, 절차 결과를 정의한다. 이

러한 종류의 커버리지를 분석하기 위해서 적어도 정형적인 커버리지 분석 방법들이 정착되기 전까지 적절한 체크리스트를 통한 리뷰가 현실적인 대안이라고 할 수 있다.

마지막으로 고려할 커버리지는 앞서 언급했던 바와 같이 DO-178 표준에서 제시하는 검증 결과에 대한 검증이다. 이는 두 단계로 이루어진다. 첫 번째는 요구사항 기반 커버리지 분석이다. 요구사항 기반 커버리지 분석은 각 요구사항에 대해서 정형 분석 케이스가 정의되어 있다면 만족된다. 그러나 두 번째 구조적인 커버리지 분석은 조금 상황이 다르다. 검증 방법으로 오직 테스팅만 사용할 때에는 문장 커버리지, 결정 커버리지, MC/DC 커버리지 등이 측정되었는데, 이들 기준은 정형 검증 결과를 검증하는 데에는 전혀 효용가치가 없다. 결론적으로 정형 분석에서 커버리지를 대체할 방법으로 [3,11]에서 제시되고, [12]에서 적용된 네 가지 요건들은 다음과 같다.

- 각 요구사항에 대한 완전한 커버리지: 정형 분석을 위해서는 앞서 언급한 바와 같이 분석 대상을 분석 가능한 크기로 줄여야 하므로 시스템 자체와 환경에 대한 가정을 하게 된다. 이러한 가정들은 정형 검증 결과가 자칫 요구사항의 일부만을 증명하는 결과를 낳는다. 이러한 문제를 해결하기 위해서는 가정들도 검증되어야 한다. 체계적이고 논리적인 검증 방법과 함께 가정에 대한 리뷰도 진행되어야 한다. 이러한 리뷰를 돕는 체크리스트가 필요하다.
- 요구사항 집합의 완전성: 요구사항과 시스템 기능은 다대다로 매핑되는 경우가 일반적이다. 따라서 요구사항 집합의 완전성이 검사되어야 한다. 이를 위해서 (1)모든 입력 조건들에 대해서, 요구되는 출력이 명시되어야 하고, (2) 모든 출력들에 대해서 요구되는 입력 조건들이 명시되어야 한다.

  이 문제는 예를 들어 프로그램에 사용된 모든 입출력 변수가 정형 분석에 사용되었는지 구문 검사함으로써 분석될 수 도 있다. 그러나 보다 엄밀하게 검사하려 한다면, 가령 모든 기능에 대한 선조건(precondition)들이 해당 요구사항들의 입력 조건을 커버하는지 다시 말해 입력 조건들이 선조건의 논리합에 논리적 귀결관계가 성립하는지 검사하고, 모든 출력 변수들에 대해서 그들의 가능한 모든 값을 생성해 내는 입력 변수 값들의 조합이 존재하는지 검사함으로써 보일 수 있다. 그러나 이것은 입증하기 매우 어렵고 사용되는 정형 표기법에 의존적이다. 따라서 향후 세심한 논의가 필요할 것이다.
- 의도하지 않은 데이터 플로우 검출: 소스코드 상에서 입출력 간의 잘 못된 의존 관계가 없다는 것을 보이려면, 정보 흐름이 요구사항을 만족함을 증명해야 한다.
- 데드코드 및 비활성 코드 검출: 데드코드는 요구사항에 없는 불필요한 코드이고 비

활성 코드는 시스템 설정에 의해서 동작하지 않게 되는 코드를 의미한다. 이에 대한 검출은 기존의 테스팅과 리뷰를 통해서 가능하다.

어떤 테스트 케이스에 대해서 구조적 커버리지가 달성되지 않았다면, 해당 요구사항에 대한 테스트 케이스가 부족하거나, 요구사항이 누락됐거나, 데드 코드가 존재하는 경우이다. 정형 기법을 적용하면 요구사항에 대해서 모든 경로를 커버하게 된다. 따라서 요구사항이 정확하게 정형 분석되었다는 것을 나타내기 위해서는 위의 네 가지 요건들이 추가적으로 분석되어야 한다.

V. 결론

정형 기법 적용함으로써 얻을 수 있는 이점들은 요구사항을 개선할 수 있다는 것, 오류의 개입을 최소화 한다는 것, 오류 검출을 효과적으로 할 수 있다는 점, 그리고 개발 비용을 절약 할 수 있다는 점 등을 들 수 있다. 그러나 이러한 이점이 주장에 그치는 것이 아니라 테스팅과 같이 효용성이 확인되어 실무에 적용되기 위해서는 많이 난제를 해결해야 한다. 본 논문에서는 그러한 문제를 극복하기 위해서 정형 기법 가이드라인을 제시했다. 여기서 제시된 가이드라인은 실제 DO-178B 인증을 통한 경험에서 체득된 것이다.

우리는 이 원고에서 제시된 방법들을 다음 프로젝트서 개발될 마이크로 하이퍼바이저의 신뢰성을 확보하는 수단으로 적용할 계획이다. 프로젝트가 진행되면서 이 가이드라인은 보다 실무에 적합하도록 정제될 것이다. 이렇게 정제된 가이드라인은 고신뢰성이 요구되는 산업 분야에 정형 기법을 확산하는 중요한 고리가 될 것으로 기대한다.

* 본 원고는 해당 논문의 요약본임에 따라 자세한 내용은 논문원문을 참조하시기 바랍니다.

참고 자료

1. ISO 26262-6:2011. Road vehicles — Functional safety - Part 6: Product development at the software level. Nov. 14, 2011.
2.. RTCA/EUROCAE. DO-178B/ED-12B Software Considerations in Airborne Systems and Equipment Certification, December 1992.
3. RTCA/EUROCAE. RTCA DO-333 Formal Methods Supplement to DO-178C and DO-278A, December 13, 2011.
4. P. Cousot and R. Cousot, Refining Model Checking by Abstract Interpretation, Automated Software Engineering January 1999, Vol. 6, Issue 1, pp. 69-95, 1999.
5. B. Cook, D. Kroening, and N. Sharygina, "Accurate Theorem Proving for Program Verification," In Proceedings of the ISoLA 2004, LNCS 4313, pp. 96-114, 2006.
6. J. Gärtner, DO-178C and COST: Challenges and opportunities for avionic software, Esterel Technologies White Papers, April, 2010.
7. M. Conrad, P. Munier, and F. Rauch, "Qualifying Software Tools According to ISO 26262," In Proceedings of MBEES 2010, Dagstuhl, Germany, 2010.
8. V. Hilerman and T. Baghai, AVIONICS CERTIFICATION: A Complete Guide to DO-178/DO-254, Avionics Communications Inc., 2007.
9. B. Josko and H. Dierks, Guidelines for verification and validation of dependability requirements. Electronic Architecture and System Engineering for Integrated Safety Systems Deliverable D3.2 Part 2, Nov. 30, 2006.
10. B. Brosgol and C. Comar, "DO-178C: A New Standard for Software Safety Certification," Systems and Software Technology Conference 2010 Presentation, 26 April 2010.
11. D. Brown, H. Delseny, K. Hayhurst, and V. Wiels, "Guidance for using formal methods in a certification context," In Proceedings of the Embedded Real Time Software and Systems Conference, Toulouse, pp. 1-7, 2010.
12. H. Blasum, F. Dordowsky, B. Langenstein, and A. Donnengart, "DO-178C Complince of Verisoft Formal Methods," the Report of Verisoft XT Project funded by the German Federal Ministry of Education and Research, December 3, 2011.