소프트웨어 요구명세와 설계 명세간의 부합성 검사

+이수영[O] 김진현 안영아 심재환 양진석 이나영* 손한성** 최진영
고려대학교 컴퓨터학과, 서울대학교 원자력공학과*, 한국원자력 연구소**
{sylee[O], jhkim, ellaahn, jhsim, jsyang, choi}@formal.korea.ac.kr
grasia2*@snu.ac.kr hsson**@kaeri.re.kr

Compliance Checking between Software Requirement and Design Specification

Su-Young lee[O] Jin-Hyun Kim, Youna-Ah Ahn, Jae-Hwan Sim, Jin-Seock Yang, Na-Young Lee*,
Han-Seong Son**, Jin-Young Choi
Dept. of Computer Science Korea University
*Dept. of Nuclear Engineering Seoul National University
**Korea Atomic Energy Research Institute

요 약

항공 및 국방 산업 등에서는 고안전성 임베디드 시스템의 신뢰성과 안전성을 보장하기 위해 임베디드 시스템의 설계 및 구현에 정형기법을 적용하고 있다. 본 논문은 그러한 정형기법을 적용하여 임베디드 시스템 소프트웨어의 요구사항 명세(Software Requirements Specification:SRS)와 요구명세를 바탕으로 실제 구현을 위한 설계명세(Software Design Specification:SDS)를 정형기법을 이용하여 명세하였다. 본 논문은 정형 명세 언어로 작성된 요구명세와 설계명세 간의 부합성 검사를 하고 더 나아가 임베디드 시스템 요구 분석에서 더 정확하게 소프트웨어를 구현할 수 있는 방법을 제시하고자 한다.

1. 서 론

근래에 임베디드 시스템들의 사용이 점차 늘어나고 있다. 특히 항공과 원자력 발전소와 같은 산업분야에서 사용되는 고안전성(Safety-Critical) 임베디드 시스템은 신뢰성과 안정성을 보장해야 한다. 따라서 추상화와 정형적 설계는 임베디드 시스템 개발 과정에서 없어서는 안될 부분이다. 그러한 임베디드 시스템 소프트웨어 설계를 정형 명세 기법을 이용하여 시스템의 주요 컴포넌트(component)들을 설계하고 시스템의 정형적 모델들을 모든 가능한 입력값들 아래 시스템의 행위를 엄격하게 검증함으로써 에러가 없는 소프트웨어를 개발할 수 있다. 게다가 대부분 그런 모델들은 결정적이고 검증과정이 자동적으로 이루어진다. 정형 모델의 형태는 "명확하고 수학적인 명세"[1]를 가져야 한다. 이것은 보증된 도구들을 이용하여 모델들을 검증하는 것을 가능하게 한다.

본 논문에서는 원자력 발전소의 제어 시스템으로 들어가는 실시간 임베디드 시스템 소프트웨어 요구명세를 작성하는데 자연어로 명세된 요구 명세의 모호함을 찾고 제거하는데 정형기법을 사용하였다. 즉, 요구명세(Software Requirement Specification:SRS)[2]와 설계명세(Software Design Specification:SDS)[3]를 명세하고 검증하는데 정형 명세 언어인 Statecharts[4, 5]와 SyncCharts[6]를 이용하였다. Statecharts는 자연어로 명세된 요구명세가 갖는 모호함을 찾고 제거하는데 사용되었으며, SyncCharts는 Statecharts로 명세된 요구명세를 기반으로 SyncCharts의 텍스트적(Textual)인 표현을 이용하여 실제 구현 가능한 설계명세를 작성하였다. 따라서 다른 정형 명세언어로 작성된 요구명세와 설계명세는 설계명세가 요구명세가 요구하는 사항들을 만족하는지를 검증하는 부합성 검사(compliance checking)가 필요하다.

2장에서는 요구명세와 설계명세에 대해 간단한 소개를 하고 Statecharts와 SyncCahrts로 작성한 실시간 운영체제 세마포(Real-time Operating System Semaphore)를 통해 요구명세와 설계명세의 부합성 검사를 보이고자 한다. 마지막으로 3장에서 결론 및 향후 과제를 제시한다.

2. 요구명세와 설계명세의 부합성 검사

2.1 요구명세와 설계명세 소개

원자력 발전소의 제어시스템이나 항공 시스템의 고안전성 시스템은 사고로 인해 큰 비용을 요구하는 데미지가 일어날 수 있다. 그러한 안전성과 신뢰성을 요구하는 시스템의 요구명세를 정형 명세 언어를 통해 일관성 있게 작성해야 한다. 요구명세는 소프트웨어 개발자가 설계하는데 필요한 모든 것들을 포함해야 한다. 설계명세는 소프트웨어 시스템이 요구명세에 정의된 요구들을 만족하는데 어떻게 구현해야 할지를 보여준다. 소프트웨어 설계 명세는 만들어진 소프트웨어 시스템의 모델이다. 모델은 소프트웨어 시스템을 계획하고 분석, 구현하는데 필요한 명확한 설계 정보를 제공해야 한다.

2.2 부합성 검사

+ 본 논문은 "원자력 연구개발 중장기사업인 원전계측제어 사업단"에 의해 지원되었음.

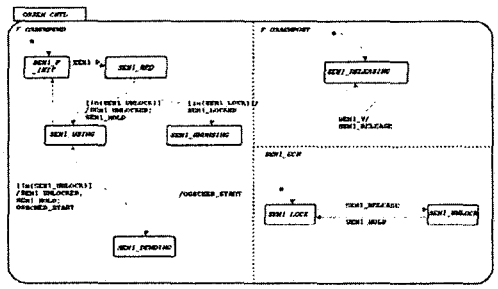
요구명세 모델은 Statecharts를 통해 시스템의 요구사항을 도식적으로 명세하였고, Statecharts와 같은 도식적 명세를 하는 SyncCharts는 텍스트적인 표현이 가능하여 요구명세를 기반으로 제약 조건을 두어 좀더 정확하고 자세한 설계명세를 구현하였다. 따라서 설계명세가 요구명세의 요구사항들을 만족하는 지를 보이는 부합성 검사가 필요하다. 즉, 설계명세 ⊢ 요구명세 을 보여야 한다.

부합성 검사는 여러 가지 방법으로 검증할 수 있는데 본 논문에서는 항상 어떤 입력 값들에 대해 요구명세 모델과 설계명세 모델은 언제가 그 입력 값에 반응하는 출력 값들을 출력해야 한다를 검증 특성으로 두어 요구명세와 설계명세간의 부합성 검사를 하고자 한다.

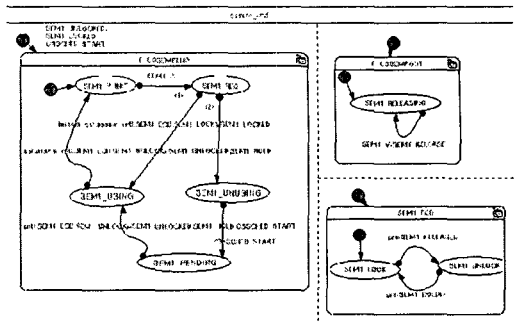
다음 절에서 실시간 운영체제 세마포어의 요구명세와 설계명세를 Statecharts와 SyncCharts로 구현하여 두 정형 명세 모델 간의 부합성 검사를 하고자 한다.

2.3 실시간 운영체제 세마포어(Real-Time Operating System Semaphore)

실시간 운영체제 세마포어는 Micro C/OS-II[8]에서 정의된 세마포어를 기반으로 Statecharts로 요구명세를 명세하고 SyncCahrts로 설계명세를 명세하였다. 요구명세와 설계명세의 두 정형 모델 사이의 부합성 검사를 위해 Statecharts로 명세된 요구명세를 SyncCharhts로 변환하였다. 이를 위해 SyncCahrts에 의해 정의된 Statecharts의 행위적 의미론[7]을 가지고 실시간 운영체제 세마포어를 명세하였다. Statecharts로 구현된 세마포어는 전체 하나의 모듈(OSSEM_CNTL)안에 세마포어를 대기하는 함수를 구현한 모듈(F_OSSEMPEND)과 세마포어를 반환하는 함수를 구현한 모듈(F_OSSEMPOST), 세마포어가 lock 상태인지 unlock 상태인지를 나타내는 모듈(SEM1_ECB)로 구성된다. 그림 1은 Statecharts로 요구명세를 명세하였다.



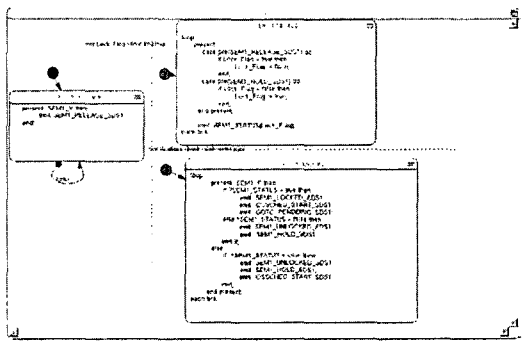
[그림 1] Statecharts에 의한 세마포어 요구명세



[그림 2] Syncchartsㅔ 의한 Statecharts의 세마포어 요구명세

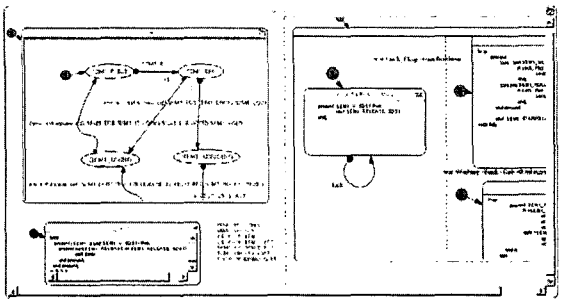
부합성 검사를 위해 Statecharts로 구현된 요구명세를 그림 2와 같이 SyncCharts로 변환하였다. Statecharts의 state들은 SyncCharts의 Graphical macro state를 사용하여 Statecharts와 같은 행위를 나타내었다. SEM1_ECB 모듈에서 상태간 전이는 pre() 연산자를 사용하여 Statecharts의 동기적인 의미를 표현하였다.

실시간 운영체제 세마포어의 요구명세 모델을 기반으로 SyncCharts로 실시간 운영체제 세마포어의 설계명세를 그림 3과 같이 명세하였다.



[그림 3] SyncCharts에 의한 세마포어 설계명세

SyncCharts에 의해 변환된 요구명세 Statecharts와 SyncCharts로 명세한 설계명세의 부합성 검사를 하기 위한 검증 특성 모듈을 SyncCharts로 명세하였다. 그림 4와 같이 SyncCharts로 요구명세 모듈, 설계명세 모듈, 검증 특성 모듈을 병렬적으로 구현하여, SyncCharts의 시뮬레이션 도구를 이용해 매 Tick을 발생시킴에 따라 요구명세 모듈과 설계명세 모듈이 같은 입력 이벤트가 들어왔을 때 동일한 출력 이벤트를 내보내는지를 비교해보았다.

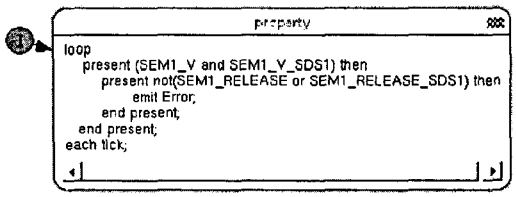


[그림 4] 요구명세와 설계명세 간의 부합성 검사

요구명세 모듈과 설계명세 모듈의 부합성 검사를 위한 검증 특성의 자연어 명세는 다음과 같고 Logic을 이용하여 표현하였다.

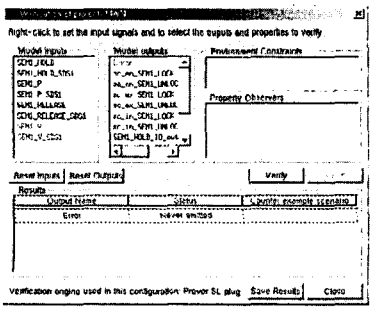
검증 특성: 매 tick 마다 요구명세 모듈과 설계명세 모듈의 모든 입력 이벤트에 대한 언젠가 출력되어야 할 모든 출력 이벤트들이 발생하지 않는 경우 Error가 발생해야 한다.

$$G\,(tick \wedge ((SRSinput\_event \wedge SDSinput\_event) \wedge \neg(SRSoutput\_event \vee SDSoutput\_event)) \rightarrow Error)$$



[그림 5] 검증 특성

그림 5는 SyncCharts로 명세된 검증특성 모듈이다. 본 논문에서는 내부 입,출력 이벤트들은 요구명세보다 더 자세하게 구현한 설계명세의 경우 제약 조건을 두어 내부 입, 출력 이벤트들이 요구명세보다 더 많기 때문에 고려하지 않았다. 따라서, 두 명세 모듈의 모든 외부 입력 이벤트들인 SEM1_V와 SEM1_V_SDS1에 대한 외부 출력 이벤트들인 SEM1_RELEASE와 SEM1_RELEASE_SDS1이 발생되는지를 검증 특성으로 명세하여 부합성 검사를 하였다.



[그림 6] XEVE에 의한 요구명세와 설계명세 간의 부합성 검사

그림 6은 SyncCharts의 검증 도구인 XEVE로서 요구명세 모듈과 설계명세 모듈이 검증 특성을 만족하는지를 실행한 결과이다. 그림 5와 같이 매 tick마다 입력 이벤트에 대한 출력 이벤트가 발생하지 않은 경우 Error가 발생하도록 명세한 검증 특성을 통해 XEVE 실행 결과, Error는 "Never emitted" 이라는 결과를 보았다. 즉, 설계명세는 요구명세의 요구사항을 만족하는지를 두 명세간의 부합성 검사를 통해 알 수 있었다.

5. 결론 및 향후 계획

임베디드 시스템의 개발을 위해 자연어를 기반으로 하는 요구명세는 개발자들이나 소비자들에게 다르게 해석될 수 있는 모호함을 가지고 있다. 본 논문은 요구명세를 Statecharts로 정확하게 명세하고 SyncCharts를 통해 실제 실행가능한 코드를 생성할 수 있는 설계명세를 구현하였다. 우리는 요구명세와 설계명세를 더 정형적으로 설계할 수 있는 접근 방법을 제시하였으며 부합성 검사를 통해 요구명세와 설계명세간의 일관성을 보장할 수 있었다. 본 논문은 실시간 운영체제 세마포어의 요구명세와 설계명세를 구현하여 요구명세와 설계명세 간의 부합성 검사를 수행해 볼 수 있었다.

결론적으로 우리의 접근 방법은 요구명세와 설계명세 간의 비일관성을 피할 수 있고 임베디드 시스템 요구 분석에서 더 정확하게 소프트웨어를 구현할 수 있도록 도울 수 있는 방법을 제시하였다. 향후 연구 과제로서는 소프트웨어 테스팅을 적용하여 설계명세와 실제 실행 코드간에 동일한 요구사항을 만족하는지를 검증해보고자 한다.

6. 참고 문헌

[1] S. Edwards, L. Lavagno, E.A. Lee, and A. Sangiovanni-Vincentelli. Design of Embedded Systems: Formal Models, Validation, And Synthesis. In Proceedings of the IEEE, 1997

[2] IEEE Std 830-1984,"IEEE Guide to Software Requirements Specifications".

[3] IEEE Std 1016-1998,"IEEE Recommended practice for Software Design Descriptions".

[4] D. Harel, "Statecharts: A visual Formalism for Complex Systems", Sci. Comput. Prog. 1987.

[5] D. Harel and A. Naamad, "The STATEMATE Semantics of Statechart", ACM Trans. Software Enginnering and Method, Vol. 5, No. 4, pp 293-333, Oct 1996.

[6] C. Andre, "SyncChart: A Visual Representation of Reactive Behavior", Technical report RR-95-52, 13S, 1995.

[7] 이수영, 김진현, 이장수, 최진영, "SyncCharts를 이용한 UML Statecharts의 의미론", 추계 정보과학 학술대회, 2003.

[8] J J. Labrosse, "Micro C/OS-II The Real-Time Kernel Second Edition", CMP Books, 2000.