한국시뮬레이션학회 논문지
Vol. 27, No. 2, pp. 61-69 (2018. 6)
http://doi.org/10.9709/JKSS.2018.27.2.061
ISSN 1225-5904

내장형 시스템의 동시적 개발을 위한 이산 사건 시스템 형식론 기반 요구사항 평가 방법

최재웅 · 최창범 †

Requirements Evaluation Method for Concurrent Development of Embedded System based on Discrete Event System Formalism

Jae-ung Choi · Chang-beom Choi†

ABSTRACT

As the technology of information and communication has developed over recent years, an embedded system is applied in multiple industries and becomes more complicated. For this reason, embedded system development cost and time are also increased. For decreasing the cost and time, this paper suggests requirement evaluation method for concurrent development of an embedded system based on Discrete Event System(DEVS) Formalism. This paper proposes a method of describing the requirements specification in the form of DEVS atomic model. Also, the paper proposes the evaluator model that compares evaluation target system and the requirements model that is an implementation of requirement specification and proposes the evaluation method using them. In addition, we propose a method to utilize the requirement model created for requirements evaluation in the concurrent development process of the embedded system. As the case study, this paper proceeds requirement evaluation of Kinect depth data processing system.

Key words : Embedded System, Discrete Event System Formalism, Requirement, Requirement Evaluation

요 약

최근 정보 통신 기술이 발전함에 따라, 내장형 시스템은 여러 산업 분야에 적용되고 있으며 더욱 복잡해지고 있다. 이로 인해, 내장형 시스템의 개발 비용과 시간도 덩달아 증가하고 있다. 본 논문에서는 내장형 시스템의 개발 비용 및 시간을 절감하기 위해서 내장형 시스템의 동시적 개발을 위한 이산 사건 시스템(DEVS) 형식론에 기반 요구사항 평가 방법을 제안한다. 본 논문에서는 요구사항 명세를 DEVS 원자 모델의 형태로 기술하는 방법을 제안하며, 이를 개발하여 만든 요구사항 모델과 요구사항 평가 대상 시스템을 비교하는 평가자 모델 및 이들을 활용하는 평가 방법을 제안한다. 또한, 요구사항 평가를 위해 만든 요구사항 모델을 내장형 시스템의 동시적 개발 과정에 활용하는 방법을 제안한다. 제안하는 요구사항 평가 방법에 대한 사례 연구로 키넥트 깊이 정보 처리 시스템에 대한 요구사항 평가를 진행하였다.

주요어 : 내장형 시스템, 이산 사건 시스템 형식론, 요구사항, 요구사항 평가

1. 서론

오늘날 일상생활에서 사용하는 기기들 상당수는 하드웨어 위에 소프트웨어가 탑재되어 유기적으로 동작하는 내장형 시스템(Embedded System)의 일종이다. 내장형 시스템이란 기계나 기타 제어가 필요한 시스템에 대해, 장치 내에 존재하는 전자 시스템 또는 전자 기기와 기계 부분을 포함하는 전체 장치의 일부로 내장된 시스템을

* 이 연구는 국방과학연구소 'BSM기반 검증 및 사용자 코드 검증 모듈(UD160075BD)'의 지원으로 수행되었습니다.

Received: 23 March 2018, **Revised**: 30 April 2018, **Accepted**: 15 May 2018

† **Corresponding Author**: Chang-beom Choi
E-mail: cbchoi@handong.edu
Dept. of Global Entrepreneurship and ICT, Handong Global University, Pohang, Korea

말하며 하드웨어의 성능과 소프트웨어의 지원을 통하여 서비스를 제공하여 일상생활의 많은 장치 제어에 사용되고 있다(Wolf, 2002; Baron et al., 1997).

일반적으로 내장형 시스템은 오랜 기간 동안 오류 없이 안정적으로 동작해야 하는 요구사항을 지니며 개인용 컴퓨터에서 쓰이는 범용 소프트웨어보다 엄격한 개발과정을 가진다(Tsai et al., 2007). 특별히 내장형 시스템은 하드웨어와 소프트웨어가 유기적으로 연계되어 동작한다는 특징으로 인하여 하드웨어 전문가, 소프트웨어 전문가 및 클라이언트의 상호 협력이 필요하다. 따라서 내장형 시스템 개발과정에서 발생할 수 있는 문제를 최소화하기 위해서는 내장형 시스템의 개발 과정 전에 하드웨어 전문가와 소프트웨어 전문가, 클라이언트가 도출한 요구사항을 정립하고 개발과정에서 요구사항을 평가 및 분석이 중요하다.

내장형 시스템 개발 방법 참조 가이드에 따르면 일반적으로 내장형 시스템의 개발과정은 하드웨어/소프트웨어 각각의 개발 및 요구사항 시험평가과정과 하드웨어와 소프트웨어의 통합과정 및 통합된 내장형 시스템에 대한 요구사항의 시험평가과정으로 정의된다(Software Engineering Center, 2012). Figure 1은 내장형 시스템 개발 방법 참조 가이드에 첨부된 내장형 시스템의 일반적인 개발 과정을 나타낸 그림이다.

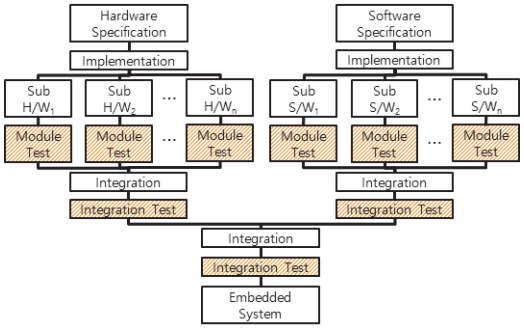

Fig. 1. Development process of Embedded System (Software Engineering Center, 2012)

Figure 1과 같은 구성요소에 대한 요구사항 평가 및 통합 시스템의 요구사항 시험평가는 내장형 시스템의 품질 보증에 대한 근거로 활용할 수 있다. 하지만, 개발과정에서는 발생할 수 있는 요구사항 변경은 일부 시스템 혹은 전체 시스템을 변경시킬 수 있으며 내장형 시스템의 동시적 개발적인 측면에서 부담을 유발시키며 변화된 요구사항을 평가하는 부분에서도 어려움이 발생한다. 따라서 내장형 시스템의 요구사항 변화에 유연하게 대처할 수 있으며 시간적 요소를 기술하고 평가하기에 적합한 요구사항 시험평가방법이 필요하다.

본 논문에서는 내장형 시스템의 동시적 개발을 위한 DEVS(Discrete Event System) 형식론 기반 요구사항 평가 방법을 제안한다. 제안하는 요구사항 평가 방법은 내장형 시스템의 요구사항에 대하여 DEVS 형식론에 기반을 두어 기술하고 이에 대응되는 시뮬레이션 모델로 구현하여, 구현된 시뮬레이션 모델을 활용하여 요구사항을 명확하게 정립하는 과정과 DEVS 모델로 정립된 요구사항을 내장형 시스템이 구현되고 난 뒤에 시험 평가하는 과정으로 정의할 수 있다. 또한, 이렇게 구현된 요구사항 모델은 내장형 시스템의 동시적 개발과정에서도 사용될 수 있다.

DEVS 형식론은 이산 사건적 특징을 지니는 다수의 세부 모듈을 바탕으로 하나의 시스템을 계층적으로 구성하는 모델링 및 시뮬레이션의 대표적 이론 중 하나이다(Zeigler et al., 2000). DEVS 형식론에서 계층적 구조를 구성하는 모듈들은 원자모델과 결합모델로 구분된다. 원자모델로 대상 체계의 구성 요소인 부 시스템을 구성하고, 결합모델로 이들 각 부 시스템들의 구성 및 연결 관계를 사용하여 구조적인 특징을 모델링한다. 따라서 본 연구에서는 요구사항 평가를 위하여 DEVS 형식론으로 표현된 요구사항 모델을 활용하여 내장형 시스템의 각 구성요소에 대한 요구사항 평가를 지원한다. 이는 대상 체계의 구성 요소인 부 시스템들을 원자모델로 개별적으로 구성하고, 대상 체계에 변경점이 생기는 경우, 변경점이 생긴 부 시스템에 대응되는 원자모델만을 수정함으로써 대상 체계의 변경에 유연하게 대처할 수 있도록 돕는다. 또한 내장형 시스템은 대개 제한된 하드웨어 자원 위에서 실시간(real-time) 제약조건을 가지고 동작한다고 앞서 설명한 바 있는데, DEVS 형식론은 시간적 요소를 원자모델의 구성요소로 포함하고 있으며, 이로 인해 내장형 시스템의 실시간 제약조건과 같은 시간적 요소에 대해 고려하고 표현하기에 적합하다.

본 논문은 구성은 다음과 같다. 2장에서는 본 논문과 관련된 연구들을 소개하고, 그 배경지식을 설명한다. 3장에서는 본 논문에서 제안하는 내장형 시스템의 동시적 개발을 위한 이산사건시스템형식론 기반 요구사항 평가 방법에 대해 설명한다. 4장에서는 제안하는 요구사항 시험평가 방법을 활용하여 Kinect를 이용한 고도 측정 시스템을 개발하고 평가한 사례연구에 대해 소개한다. 5장에서는 내용을 정리하고 향후 연구 방향을 제시한다.

2. 관련 연구 및 배경지식

본 장에서는 내장형 시스템의 동시적 개발을 위한 DEVS 형식론 기반 요구사항 평가 방법과 관련된 연구에 대하여 소개하고 본 논문에서 제안하는 요구사항 평가방법의 기반이 되는 DEVS 형식론에 대해 간단히 설명한다.

2.1 관련연구

내장형 시스템의 요구사항 평가 방법에 관련된 연구로는 내장형 시스템을 위한 상태 전이 모델 기반 테스트 케이스 생성 기법에 관한 연구가 있다(Jeong et al., 2011). 이 연구는 상태 전이 기반 테스트 모델에 대해 제시하고, 이 모델로부터 상태 전이 표를 작성한 뒤 이를 통해 테스트 케이스를 구체화한다. 제안하는 상태 전이 모델은 DEVS 모델과 유사한 측면이 있으나, 시간적 요소에 대해 전혀 고려하지 않았다.

다음으로 검증 패턴을 이용한 내장형 시스템 시험에 대한 연구가 진행된 바 있다(Tsai et al., 2005). 이 연구는 내장형 시스템의 시험에 사용되는 많은 시험 스크립트들을 시험 시나리오로 보고, 이 시나리오들의 패턴을 분석하여 그룹화하고, 그룹화된 시나리오들이 사용할 수 있는 시험 스크립트 틀을 제작하여 이용함으로써, 내장형 시스템의 시험에 드는 시간과 비용을 낮추는 방법을 제시했다. 하지만 이 연구는 대상이 되는 내장형 시스템에 변경점이 생길 때마다 시나리오 작성, 시나리오 분류, 시험 스크립트 틀 제작의 과정을 다시 거쳐야 한다는 문제점이 있다. 특히, 시험과 변경이 수시로 이루어지는 현대의 개발 프로세스에 적합하지 않다.

다음으로 서론에서 간단히 설명했던 내장형 시스템의 일반적인 개발과정에 대해 서술한 내장형 시스템 개발 방법 참조 가이드가 있다(Software Engineering Center, 2012). 가이드에서 제안하는 내장형 시스템의 일반적인 개발 과정을 자세히 설명하면 다음과 같다. 하드웨어와 소프트웨어 각각은 작성된 명세서에 따라서 개발되며 다양한 기능을 수행하는 독립적인 부체계들로 나뉘어 개발된다. 부체계들에 대한 개발들이 모두 끝나면 각각의 부체계들에 대해 테스트가 시행되고 결과에 이상이 없을 시 부체계들을 하나의 소프트웨어와 하드웨어로 각각 통합한다. 그리고 통합된 하드웨어와 소프트웨어에 대해 각각 테스트를 시행하고, 여기에서도 이상이 없으면 하나의 내장형 시스템으로 통합한다. 통합한 내장형 시스템에 대해서 마지막으로 테스트를 한 후 정상적인 결과를 얻으면 내장형 시스템의 개발 과정이 종료된다.

본 논문에서는 상술한 내장형 시스템의 일반적인 개발과정에 따라 내장형 시스템의 개발이 이루어지는 것으로 가정하며, 이에 따라 본 논문에서 제안하는 요구사항 시험 평가는 하드웨어, 소프트웨어 각각에 대한 요구사항 시험 평가뿐만 아니라 통합된 내장형 시스템에 대한 요구사항 평가도 포함한다.

본 논문에서 제안하는 요구사항 평가 방법에서 사용되는 요구사항 모델은 요구사항 정립과정과 요구사항 평가 과정 뿐만 아니라 내장형 시스템의 동시적 개발과정에서도 사용될 수 있다. 또한 DEVS 형식론의 모듈러한 특성상, 표현하고자하는 내장형 시스템의 부체계를 구성하는 원자모델들을 유사한 타 내장형 시스템의 표현에도 재사용할 수 있으므로 재사용성이 높다. 그리고 DEVS 형식론은 시간적 요소를 표현하기에도 적합하며, 앞서 설명한 바와 같이 모듈러한 특징을 지니므로 대상 내장형 시스템의 변경에 대처하기가 적합하다. 다음 절에서는 본 논문에서 제안하는 요구사항 평가방법의 기반이 되는 DEVS 형식론에 대해 간단히 소개한다.

2.2 DEVS 형식론

DEVS 형식론은 이산 사건 시스템을 표현하기 위한 집합론에 근거한 수학적인 틀이다. DEVS 형식론에는 원자 모델(Atomic Model)과 결합 모델(Coupled Model)이 있다. 원자 모델은 시스템의 기본적인 구성 요소가 되는 부시스템들을 객체 단위로 모듈화하여 표현하며, 결합모델을 이들 원자 모델을 계층적으로 결합하여 나타낸다(Kim and Park, 1992).

원자 모델은 이산 사건 시스템의 계층적인 구조에서 가장 기본적인 구성 요소로서 시스템의 행동을 기술한다. 원자 모델은 세 개의 집합과 네 개의 함수로 이루어져 있으며 세 개의 집합 X, Y, S는 각각 입력, 출력, 상태의 집합이고 네 개의 함수 δint, δext, λ, ta는 각각 내부 상태 천이, 외부 상태 천이, 출력, 시간 전진 함수이다.

결합 모델은 여러 원자 모델 혹은 하위 결합 모델을 연결하여 만든 모델이다. 결합 모델이 원자 모델 혹은 하위 결합 모델을 포함함으로써 거대한 시스템을 계층적이고 체계적으로 표현할 수 있게 된다. 결합모델은 세 개의 집합과 세 개의 관계 기술 정보, 그리고 하나의 함수로 이루어져 있다. 결합 모델의 세 개의 집합 X, Y, {Mi}는 각각 입력, 출력, 이산 사건 컴포넌트 모델의 집합을 나타낸다. 결합 모델을 구성하는 세 개의 관계 기술 정보는 EIC, EOC, IC이며, 각각 외부 입력 관계, 외부 출력 관계, 내부 연결 관계를 나타낸다. 그리고 SELECT 함수는 같은

시간에 존재하는 사건을 발생하는 모델들에 대한 선택 함수이다.

DEVS 형식론에 기반한 모델은 DEVS 형식론에 따라 수학적 명세로 표현될 수도 있지만, DEVS 다이어그램으로도 표현될 수 있다(Song and Kim, 2010). 본 논문에서는 DEVS 다이어그램의 형태로 DEVS 형식론에 기반한 모델들을 표현한다.

3. 내장형 시스템의 동시적 개발을 위한 DEVS 형식론 기반 요구사항 평가 방법

3.1 요구사항 명세 기술 방법 및 요구사항 모델의 개발

기존의 요구사항 명세는 주로 자연어를 사용하여 기술되어 왔으며, UML 다이어그램(RumBaugh et al., 2004), 각종 표, 또는 그림 등을 통해 자연어로 이루어진 요구사항 명세를 보충하는 방식으로 기술되었다. 자연어는 유연하고 보편적이며 광범위한 장점을 가지고 있지만, 자연어로 기술된 요구사항 명세는 불완전하고 일관성이 없으며 모호함을 잠재한다는 단점을 가진다(Kamsties and Paech, 2000). 본 논문에서 위와 같은 문제점을 해결하고 DEVS 형식론이 가지는 모듈러하고 계층적이며 재사용성이 높은 장점을 이용하기 위해, DEVS 형식론으로 개발하고자 하는 내장형 시스템의 요구사항을 기술한다.

요구사항 명세는 기본적으로 모든 입력에 대응하도록 기술되는 것이 이상적이나, 비용이나 시간과 같은 다양한 한계점들로 인해 이상적인 요구사항 명세를 기술하는 것이 불가능한 상황이 존재한다. 그러므로 본 논문에서는 제한된 입력에 대응하는 요구사항 명세를 기술하고 이에 대응하는 시뮬레이션 모델을 구현하는 방법을 제안한다.

요구사항은 일반적인 분류법에 의해 기능 요구사항, 비기능 요구사항으로 분류될 수 있다. 본 논문에서는 기능 요구사항을 대상으로 요구사항 명세를 기술한다. 기능 요구사항이란, 반드시 구현되어야 할 필수적인 작업과 동작 등을 정의함으로써 어떤 기능이 구현되어야 하는지를 설명하는 요구사항이다(Lightsey, 2001).

기능 요구사항 명세의 기술은 대상 내장형 시스템의 필수 기능이나 동작을 DEVS 형식론으로 정의하는 것이며, DEVS 형식론은 모델의 형태로 대상 시스템을 표현하기 위한 수학적 틀이다. 그러므로 이는 곧 대상 내장형 시스템을 DEVS 모델로 모델링하는 것과 동일하다. Figure 2은 요구사항 명세의 기술과 요구사항 모델 개발 절차에 대한 그림이다.

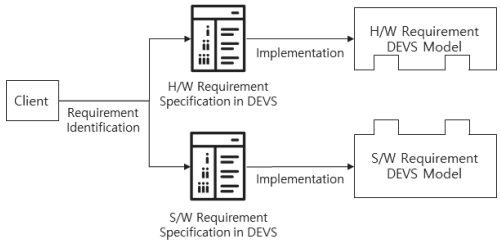

Fig. 2. Requirement Specification Description and Requirement Model Development Process

Figure 2에서 보는 것과 같이 클라이언트로부터 요구사항을 식별하고, 이를 DEVS 형식론으로 기술한 요구사항 명세는 곧 요구사항대로 동작하는 하드웨어 또는 소프트웨어 모델에 대한 명세이며, 이를 구현하면 하드웨어 또는 소프트웨어 요구사항 모델이 된다.

본 논문에서 제안하는 DEVS 형식론을 활용한 요구사항의 명세 방법은 기능적 요구사항을 입력과 출력 집합으로 표현하고 기술한다. 즉, 대상 내장형 시스템이 가지는 기능 자체에 대해 설명하는 방식으로 요구사항 명세를 기술하지 않고 기능을 수행하기 위한 특정한 입력 값 x를 시스템에 주었을 때 해당 기능에 의해 발생하는 출력 값 y를 (X, Y)의 형태로 기록함으로써 기능에 대해 기술한다. 그리고 해당 기능이 시간적 제약사항을 포함할 수 있기 때문에 시간 제약사항 t를 입력, 출력과 함께 (X, Y, T)의 형태로 나타낸다. 이를 DEVS 형식론을 사용하여 기술하면 Figure 3와 같이 원자모델의 형태로 기술할 수 있다.

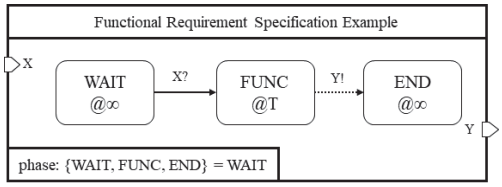

Fig. 3. Functional Requirement Specification Example

Figure 3와 같이 개별적으로 기술된 기능적 요구사항 명세를 부 체계 단위로 분류하여 결합하면 Figure 4와 같이 전체 시스템에 대한 요구사항 명세를 작성할 수 있다.

Figure 3와 같이, 기능 요구사항 원자모델들을 시스템의 계층적 분류에 따라 분류하고, 같은 분류에 속한 원자모델들을 결합함으로써 Figure 4과 같은 전체 시스템에 대한 요구사항 명세를 기술할 수 있다.

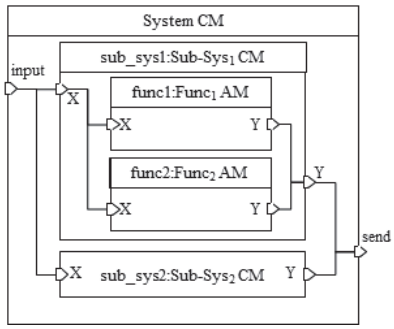

Fig. 4. Embedded System Requirement Specification Example

3.2 내장형 시스템의 요구사항 평가 방법

본 절에서는 내장형 시스템의 요구사항 평가 방법에 대해 설명한다. 다음 Figure 5은 요구사항 평가 방법의 핵심 아이디어를 개략적으로 표현한 그림이다.

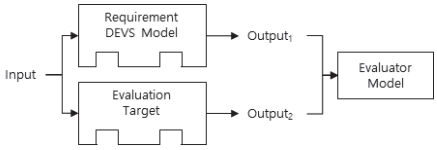

Fig. 5. Requirement Evaluation Method

본 논문에서는 Figure 5과 같이 요구사항 모델과 평가 대상에 동일한 값을 입력으로 주고, 출력되는 두 개의 값을 평가자 모델에 입력으로 주고 출력되는 값을 바탕으로 내장형 시스템의 요구사항을 평가하는 방법을 제안한다. 제안하는 평가 방법을 DEVS 형식론으로 표현하면 Figure 6과 같은 모델로 나타낼 수 있다.

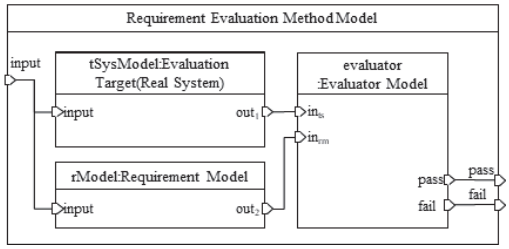

Fig. 6. Requirement Evaluation Simulation Model

Figure 6은 실제 시스템을 실행하는 모델과 요구사항 모델을 시뮬레이션하면서 평가하는 요구사항 평가 시뮬레이션 모델이다.

요구사항 모델과 실제 시스템이 1:1로 대응된다면 내장형 시스템을 구성하는 하드웨어, 소프트웨어뿐만 아니라 각각의 부 시스템들까지도 Figure 6의 방법을 사용하여 검증할 수 있다. 또한, 기능적 요구사항 모델은 부 시스템의 요구사항 모델의 계층적인 결합으로 구성된다. 그러므로 요구사항 명세 상세도에 따라 어느 계층부터 1:1 대응이 이루어지는 지에 대한 차이는 발생할 수 있지만, 일반적으로 기능적 요구사항 명세는 실제 시스템과 1:1로 대응된다고 할 수 있다.

Figure 6 내부의 평가자 모델은 Figure 7와 같이 표현할 수 있다.

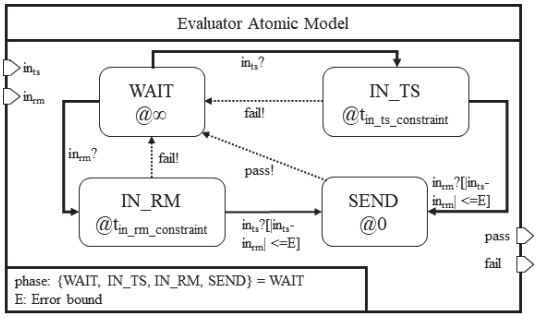

Fig. 7. Evaluator Atomic Model

Figure 7는 평가자 원자모델로써, 입력되는 두 개의 값이 오차범위 내에서 일치하는지를 판단하여 pass/fail로 평가 결과를 도출한다. 요구사항 평가를 위한 오차범위에 대한 지정은 성능 요구사항을 기술함으로써 별도로 지정할 수 있으며, 이는 구체적인 수치로 표기한다. 오차범위에 대한 성능 요구사항을 통한 별도의 지정이 없다면 오차율 10%로 요구사항을 평가한다. 평가 결과가 fail로 출력되면 대상 모델을 수정/보완하여 해당 요구사항에 대한 평가를 다시 시행하는 방식으로 전체 프로세스를 진행한다.

3.3 요구사항 모델의 활용

내장형 시스템의 개발은 하드웨어와 소프트웨어의 개발로 나뉘는데, 개발 순서에 따라 다양한 문제가 발생할 수 있다. 만약 소프트웨어를 먼저 개발하고 하드웨어를 개발할 경우, 소프트웨어가 요구하는 성능이 과도하게 높아서 성능과 비용의 트레이드오프 관계로 인해 발생하는 하드웨어의 비용 문제에 의해 전체 시스템의 비용이 너무 높아질 수 있다. 반면에, 하드웨어를 먼저 개발하고 소프트웨어를 개발하는 경우에는 소프트웨어가 요구하는 최소 성능을 하드웨어가 보장하지 못하여 소프트웨어가 제공하는 결과물의 질이 떨어지게 되거나 제공하는 속도

가 느려질 수 있으며, 심각하면 아예 결과물을 생산하지 못할 수도 있다. 이외에도 개발 순서에 따라 발생 가능한 잠재적인 문제들이 많이 있다. 그러나 요구사항 모델을 활용하면 하드웨어와 소프트웨어를 동시에 개발하여 이런 문제점을 다소 해결할 수 있다.

본 절에서는 하드웨어 개발 과정을 예로 들어 요구사항 모델 활용 방법을 설명한다. 그러나 요구사항 모델은 하드웨어뿐만 아니라 소프트웨어의 개발에서도 동일한 방법으로 활용될 수 있다. Figure 8은 요구사항 모델의 첫 번째 활용 방법인 통합 테스팅에서의 활용 방법에 대해 설명한다.

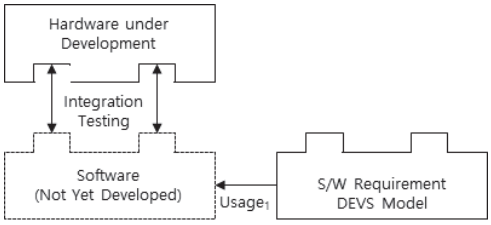

Fig. 8. Usage of Requirement Model in Integration Testing

Figure 8에서 볼 수 있듯이, 아직 개발되지 않은 소프트웨어를 대체하여 개발 중인 하드웨어와의 통합을 시험하는데 소프트웨어 요구사항 모델을 활용할 수 있으며 이를 통해 소프트웨어와의 인터페이스 문제와 같이 통합 과정에서 발생할 수 있는 다양한 문제들을 진단하고 수정할 수 있다. Figure 9은 하드웨어 개발에서의 요구사항 모델의 활용 방법에 대해 설명한다.

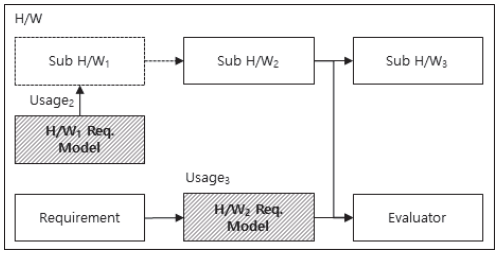

Fig. 9. Usages of Requirement Model in Hardware Development

Figure 9의 활용법 2는 다른 부 체계의 의존성(dependency)을 가지고 있으나 아직 개발되지 않은 부 체계를 대체함으로써 의존성으로 인해 발생할 수 있는 개발에서의 지연을 방지하는데 요구사항 모델을 활용할 수 있음을 설명한다. 그리고 Figure 9의 활용법 3은 부 체계들의 요구사항을 평가하는데 요구사항 모델을 활용할 수 있음을 설명한다. 본 논문에서 제안하는 요구사항 모델은 DEVS 형식론의 특성에 따라 계층적으로 구성되어 있으므로, 하위 요구사항 모델들을 이용하면 작은 단위의 부 체계들의 요구사항 평가도 진행할 수 있다. 이를 통해 부 체계들을 전체 시스템으로 합치기 이전에 유닛 단위로 평가를 하고 보완할 수 있다.

4. 사례 연구: 키넥트를 이용한 깊이 정보 처리 시스템의 요구사항 평가

4장은 시스템에 대한 전반적인 설명, 요구사항 명세의 기술 및 요구사항 모델의 개발, 요구사항 평가 및 결과로 구성된다.

4.1 시스템 개요

깊이 정보 처리 시스템은 모래놀이 상자(Sandbox)와 깊이 측정 센서(Kinect)를 활용하여 아이들의 심리치료 등에 활용할 수 있는 HCI 어플리케이션 중 하나이다. 깊이 정보 처리 시스템은 하드웨어인 키넥트 깊이 센서를 통해 모래놀이 상자에 있는 모래들의 깊이 정보를 읽어들여서 소프트웨어의 연산을 통해 색상 데이터(Red, Green, Blue: RGB) 값으로 변환하여 출력한다. Figure 10는 개발된 깊이 정보 처리 시스템이다.

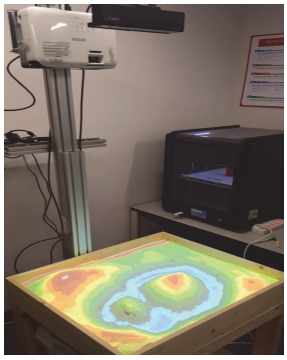

Fig. 10. Sandbox-Kinect depth data processing system

4.2 요구사항 명세 기술 및 요구사항 모델 개발

깊이 정보 시스템의 하드웨어와 소프트웨어는 Table 1과 같이 자연어로 표기된 요구사항을 가진다. 하드웨어와 소프트웨어는 각각 두 가지의 기능 요구사항을 가지며, 요구사항 평가에서의 오차율 지정을 위해 별도의 시스템 성능 요구사항을 식별하였다.

Table 1. Requirement Specification of Kinect depth data processing system in natural language

Classifications	Requirements
Hardware Function	HWF-01: Generates depth data by recognizing the surface of the sandbox at regular time intervals.
	HWF-02: The generated depth data should contain a certain level of noise.
Software Function	SWF-01: Removes noise of depth data through calibration process.
	SWF-02: Generates color data using noise-removed depth information.
System Performance	SP-01: The error rate between the color data generated by the real system and the color data generated by the requirement model should be within 3%.

위와 같이 자연어로 표기된 기능 요구사항은 제안하는 요구사항 명세 기법을 사용하여 Figure 11, 12와 같이 하드웨어 요구사항 모델과 소프트웨어 요구사항 모델로 기술할 수 있다.

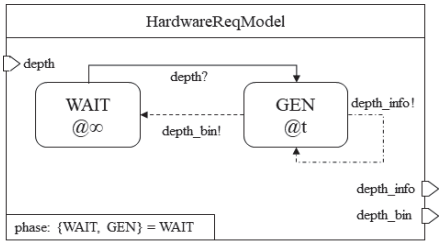

Fig. 11. H/W requirement model of Kinect depth data processing system

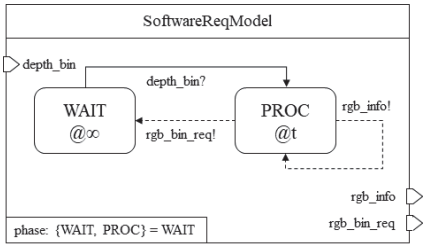

Fig. 12. S/W requirement model of Kinect depth data processing system

Figure 11의 하드웨어 요구사항 모델은 WAIT 상태에서 키넥트 깊이 센서가 일정 시간마다 512*424 크기의 깊이 정보를 실측하여 깊이 정보를 프레임 별로 생성하고 프레임 별 깊이 정보를 하나로 합친 깊이 정보 바이너리 파일을 depth 입력 포트를 통해 받는다. 그 다음, 전달받은 깊이 정보 파일을 프레임 별로 분석하여 각 데이터마다 10%의 확률로 -5~+5의 노이즈를 포함시켜서 다시 깊이 정보를 생성한다. 그리고 GEN 상태에서 일정 시간마다 프레임별 깊이 정보를 depth_info 포트로 출력하고 모든 프레임의 깊이 정보를 출력한 뒤 이들을 하나의 바이너리 파일로 저장하여 depth_bin 포트로 출력하면서 STOP 상태로 상태천이한 후 모델의 동작이 중지된다.

Figure 12의 소프트웨어 요구사항 모델은 생성된 깊이 정보 바이너리 파일을 읽어서 RGB 데이터로 변환하여 파일로 저장하고 이미지 프로세싱 라이브러리를 사용하여 시각화하여 보여준다. WAIT 상태로 초기화되어 기다리다가 하드웨어의 깊이 정보 생성이 완료되면 생성된 깊이 정보 파일을 depth_bin으로 입력받아서 PROC로 상태천이한 후 프레임별로 rgb 데이터로 변환하여 rgb_info를 출력한다. 그리고 모든 프레임을 변환하면 출력된 모든 rgb_info를 하나의 파일로 저장하여 rgb_bin_req로 출력하면서 WAIT 상태로 상태천이한 후 모델의 동작이 중지된다.

4.3 요구사항 평가 및 결과

상술한 시스템 성능 요구사항 SP-01을 평가하기 위해 3.3 절에서 설명한 요구사항 평가 모델의 포맷과 유사한 시뮬레이션 모델을 Figure 13와 같이 개발했다.

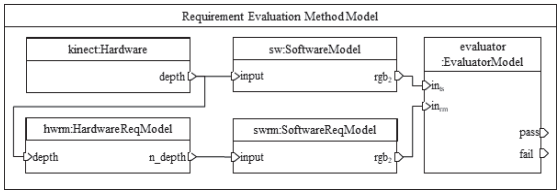

Fig. 13. Coupled Model(CM) for requirement evaluation

실험은 10회 진행하였으며, 각 회는 불규칙하게 변화하는 모래놀이 상자 표면의 깊이 정보를 5 프레임으로 촬영하여 발생한 depth 바이너리 정보를 바탕으로 진행하였다. 위 Figure 13와 같이 하드웨어와 소프트웨어는 측정한 depth 정보를 바탕으로 색상 정보를 생성하며, 하드웨어 모델과 소프트웨어 모델은 측정한 depth 정보에 노이즈를 섞은 n_depth 정보를 바탕으로 색상 정보를 생성한다. 실 시스템에 의해 생성된 색상정보 데이터 rgb1과 요구사항 모델에 의해 생성된 rgb2는 평가자 모델의 입

력으로 전달되며, 평가자 모델은 이 둘의 오차율을 계산하고 오차율이 0이면 pass로, 오차율이 0을 초과하면 fail로 출력한다. 두 데이터의 오차율은 다음과 같이 정의되는 개별 데이터 오차율 계산식 및 개별 데이터 오차율 평균 계산식을 통해 산출하였다.

개별 데이터 오차율(%): (|이론값 – 측정값|)/이론값 *100

개별 데이터 오차율 평균(%): 개별 데이터 오차율의 합 / 개별 데이터 오차율의 개수(오차가 발생한 데이터의 수)

위 식에서 개별 데이터 오차율은 개별 데이터에서 발생하는 오차의 정도를 나타낸다. 또한, 개별 데이터 오차율 평균은 개별 데이터에서 발생하는 오차의 정도의 평균을 의미한다. 편의상 요구사항 모델에 의해 생성된 값을 이론값, 실제 시스템에 의해 생성된 값을 측정값으로 명명하였다. 위 식을 통해 계산하여 도출된 개별 데이터 오차율 평균은 다음 Figure 14의 그래프와 같다.

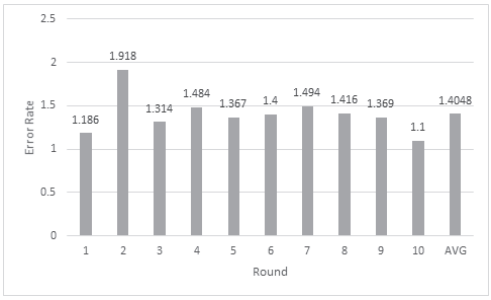

Fig. 14. Error rate graph of Kinect depth data processing system

10회의 실험에서 각 회마다 편차는 조금씩 존재했으나, 평균적으로 1.4048%의 개별 데이터 오차율 평균이 각 실험에서 발생함을 알 수 있었다. 또한, 개별 오차율 평균의 최소값은 1.1%이며, 최대값은 1.918%로 편차도 크지 않았다. 그러므로 개발된 깊이 정보 처리 시스템은 시스템의 성능 요구사항 SP-01에 대한 요구사항 평가에 통과하였다.

5. 결론

본 논문에서는 내장형 시스템의 특징에 맞도록 하드웨어와 소프트웨어, 그리고 하드웨어와 소프트웨어의 상호작용에 대한 요구사항 평가를 각각 진행하는 모델 기반 내장형 시스템 요구사항 평가 방법을 제안하였다. 제안하는 평가 방법은 이산 사건 형식론의 시간을 기술할 수 있는 특징과 모듈러한 특징을 지니고 있으며, 이에 따라 대상 내장형 시스템의 변경에 따른 평가 모델 변경이 용이하다.

앞서 내장형 시스템의 개발 부분에서 잠시 언급했던 바와 같이 시뮬레이션 모델을 이용한 내장형 시스템 개발은 하드웨어와 소프트웨어의 동시적 개발을 가능하게 하며 본 논문에서 제안하는 평가 모델과 결합하면 내장형 시스템의 시험 주도적 개발(test – driven development)이 가능하게 할 수 있다. 그러므로 향후 연구 과제로 이산 사건 모델 기반 내장형 시스템 개발 프로세스를 들 수 있다.

References

Wolf, W. (2002). What is embedded computing?. Computer, 35(1), 136-137.

Baron, C., Geffroy, J. C., & Motet, G. (Eds.). (1997). Embedded System Applications. Kluwer Academic Publishers.

Tsai, W. T., Wei, X., Paul, R., Chung, J. Y., Huang, Q., & Chen, Y. (2007). Service-oriented system engineering (SOSE) and its applications to embedded system development. Service oriented computing and applications, 1(1), 3-17.

Software Engineering Center, Technology Headquarters, Information-technology Promotion Agency, Japan. (2012) “Embedded System Development Process Reference Guide”.

Zeigler, B. P., Praehofer, H., & Kim, T. G. (2000). Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems. Academic press.

So-Young Jeong, Young-Won Chang, Cheol-Jung Yoo. (2011). Test Case Generation Technique Based on State Transition Model for Embedded System. The Journal of Korean Institute of Information Technology, 9(4), 11-21.
(정소영, 장영원, 유철중. (2011). 임베디드 시스템을 위한 상태 전이 모델 기반 테스트 케이스 생성 기법. 한국정보기술학회논문지, 9(4), 11-21.)

Tsai, W. T., Yu, L., Zhu, F., & Paul, R. (2005). Rapid

embedded system testing using verification patterns. IEEE software, 22(4), 68-75.

Kim, T. G., & Park, S. B. (1992, June). The DEVS Formalism: Hierarchical Modular Systems Specification in C++. In Proc of 1992 European Simulation Multiconference (pp. 152-156).

Rumbaugh, J., Jacobson, I., & Booch, G. (2004). Unified modeling language reference manual, the. Pearson Higher Education.

Kamsties, E., & Paech, B. (2000). Surfacing Ambiguity in Natural Language Requirement.

Lightsey, B. (2001). Systems engineering fundamentals. DEFENSE ACQUISITION UNIV FT BELVOIR VA.

Song, H. S., & Kim, T. G. (2010, October). DEVS diagram revised: a structured approach for DEVS modeling. In Proc. European Simulation Conference (Eurosis, Belgium, 2010) (pp. 94-101).

최 재 웅 (21631005@handong.edu)

2016 한동대학교 컴퓨터공학, 전자공학 학사
2016~ 현재 한동대학교 정보통신공학과 석사과정

관심분야 : 모델링 형식론, 모델 검증, 시뮬레이터 검증

최 창 범 (cbchoi@handong.edu)

2005 경희대학교 컴퓨터공학 학사
2007 KAIST 전산학 석사
2014 KAIST 전자공학 박사
2014~ 현재 한동대학교 ICT창업학부 조교수

관심분야 : DEVS 형식론, 소프트웨어 품질 보증, VV/A