이산사건 시스템의 계층적 검증방법론

송해상* · 이완복**

Hierarchical Verification Methodology of Discrete Event Systems

Hae-Sang Song* · Wan-Bok Lee**

요 약

상태폭발문제는 이산사건 시스템 검증과정에서 피하기 어려운 문제로 알려져 있으며, 큰 시스템 해석을 어렵게 만드는 본질적인 요인이다. 본 논문에서는 이산사건 시스템의 무시간 DEVS/DEVS 명세에 대한 계층적인 설계/검증 방법을 제안하여 상태폭발문제를 회피할 수 있는 방법을 보인다. 제안한 방법은 DEVS 상위모델과 DEVS 하위모델간의 설계/검증 과정을 계층적이면서도 반복적인 방식을 채택하여 검증에 필요한 요소 이외에는 정제 과정을 통하여 없애는 방안이다. 간단한 예제를 통하여 제안된 방법론을 소개하고 있다.

ABSTRACT

State explosion is a well-known problem that impedes analysis and testing of discrete event systems , thus making the verification of large systems intrinsically difficult job. This paper suggests a hierarchical verification methodology of untimed DEVS model which can alleviate the state explosion problem. The method is a repetitive procedure of designing and verifying between the upper level and the lower level models abstracting away the unnecessary information with respect to a given verification task. A small example was employed to show our suggested method in detail.

키워드

DEVS, 검증 방법, 이산사건 시스템, 계층적인 설계

Ⅰ. 서 론

이산사건 시스템을 표현하고 해석하기 위하여 개발된 DEVS 형식론[1]은 이에 기반한 DEVS 정형기법에 이용되기에 이르렀다[2]. 정형기법은 통신망, 교통, 제조, 컴퓨터, 이산사건 제어 시스템 등 이산사건시스템 수준의 복잡한 소프트웨어 시스템을 구조적으로 오류 발생 여지를 줄이면서 개발하는 방법론을 말한다. 정형기법은 시스템 개발에 필요한 여러 부분에 대한 방법론을 제공한다. 일반적으로, 시스템 명세를 통한 설계과정, 설계된 명세와 사용자 요구사항의 검증 또는 설계중인 시스템 명세간의 검증을 다루는 논리 분석 과정, 설계된 시스템 명세의 시뮬레이션 과정, 마지막으로 검증된 설계 명세를 바탕으로 프로토타입 구현과정에 대한 방법론이 이에 해당한다. 그림 1은 DEVS 정형론을 기반으로 한 시스템 개발 방법론 (DEVS 정형기법)을 표현한 것이다.

DEVS 정형기법의 핵심인 DEVS 정형론은 1970년대

* 서원대학교 컴퓨터정보통신학부
** 공주대학교 게임디자인학과

접수일자 : 2006. 10. 26

고안된 이후로 이산사건 시스템의 표현력과 시뮬레이션을 통한 해석에 주력하였기 때문에 표현력이 아주 큰 반면 논리적 해석에는 약한 것으로 알려져 왔다. 그러나 최근에 이르러 DEVS로 명세된 시스템에 대한 논리적 해석에 대한 연구가 활발해지고 있다[3][4]. 또한 모델 교환을 용이하게 하기 위해 모델명세에 대한 표준화 움직임이 일고 있으며[6] 그 일환으로 수학적인 DEVS 정형론을 언어적으로 표현하는 방법을 제공하기 위한 DEVS 명세언어가 제안되었다[3][5].

DEVS 정형론은 일반적인 이산사건 시스템의 특성을 완전하게 표현할 수 있으며 그 표현력은 입출력 시스템 수준으로 CFSM, Automata 등의 표현력과 동등한 수준이다. 하지만 DEVS 정형론은 다른 정형론과 달리 명백히 입력과 출력이 구분되고 시간이 지원되며 내, 외부상태천이함수가 구분되어 자발적 천이를 표현하기 쉽다는 특징을 가지고 있다. 또한 모듈러하고 계층적인 표현방법을 가지고 있으므로 다른 어떤 정형론보다 실제 시스템을 직관적으로 표현할 수 있는 표현력을 제공한다. 이는 계층적인 시스템 설계 시 매우 유용한 특성이다.

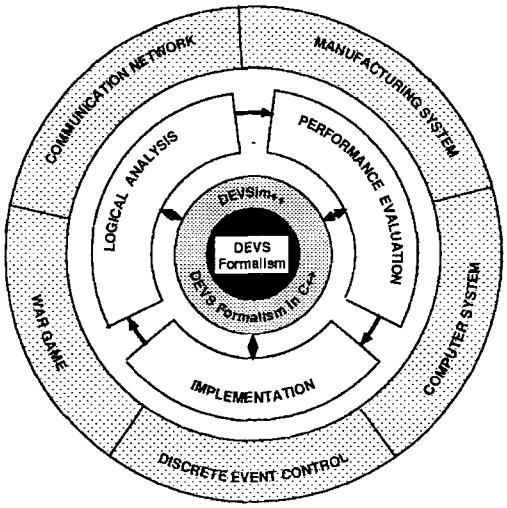


그림 1. DEVS 정형 기법
Fig. 1. DEVS based Formal Method

본 논문에서는 DEVS 정형론에 바탕을 둔 DEVS 정형기법 중 시스템의 설계과정과 검증과정을 다룬다. 시스템의 설계는 사용자 요구명세로부터 구현명세에 이르기까지 분할과 정제를 반복하면서 구체화시키는 계층적 설계 방식을 제안하고자 한다. 설계 과정에서 반드시 필요한 것은 설계된 구현명세가 주어진 요구명세를 만족하는지를 검증하는 것이다. 설계가 진행됨에 따라 구현명세가 복잡해지면서 시스템의 복합 상태 수가 급격히 증가한 결과 검증불능의 상태에 도달하는 상태폭발문제가 야기될 수 있다. 따라서 본 논문에서는 이 문제를 완화시키기 위하여 계층적 설계 방법에 대응하는 계층적 검증방법을 제시하고자 한다.

II. DEVS 정형기법에서의 시스템 개발 단계

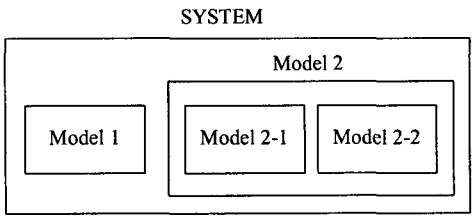
2.1. DEVS 정형론에 의한 명세

DEVS 정형론은 원자 모델(atomic model)과 결합 모델(coupled model) 표현 형식론으로 나뉜다. 원자모델은 더 이상 나누어질 수 없는 모델로서 시스템의 동적 특성을 표현한다. 결합모델은 이 원자모델들과 내포된 같은 구조의 결합모델을 구성요소로 가지고 있으면서 모델간의 연결관계를 표현하고 있다. 이 결합모델은 다른 결합모델들을 내포하는 것을 허용함으로써 계층적인 구조를 가진다. 또한 모델간의 결합은 순수하게 사건에 의해서만 이루어지고 공유하는 변수는 존재하지 않기 때문에 모듈화된 특성을 가진다. 이 두가지 모델에 관한 자세한 수학적인 표현은 [1]을 참조할 수 있다.

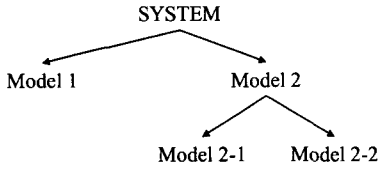
DEVS 모델의 예를 들면 그림 2와 같다. 왼쪽 그림에서 SYSTEM, Model 2는 결합모델이며, 맨 안쪽의 Model 1, Model 2-1, Model 2-2는 원자모델로서 각각의 동적인 움직임을 나타낸다. 결합모델 SYSTEM은 결합모델 Model 2와 원자모델 Model 1을 내포하고 다시 결합모델 Model 2는 그 내부에 원자모델 Model 2-1, Model 2-2를 내포한다. 이를 나무 구조로 나타내면 그림 2b)와 같다.

2.2. 계층적 개발 단계

DEVS 정형 기법에서의 시스템 설계는 분할(decomposition)과 정제(refinement) 과정을 거치는 계층적 설계과정을 따른다. 분할이란 그림 3처럼 상위 명세에서의 모델을 하위 명세에서 여러 모델들로 나누어 구체화하는 것을 뜻한다. 한편, 정제란 그림 4에서처럼 상위 명세에서의 원자모델을 하위 명세의 원자모델에서 분할 없이 동작만을 더욱 구체화하는 방법이다.



a) 블록 구조



b) 나무 구조

그림 2. DEVS 모델의 전형적인 모습
Fig. 2. Traditional Appearance of DEVS Model

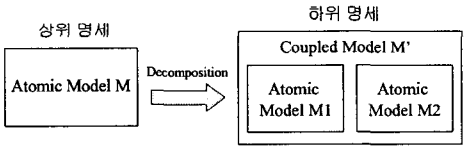


그림 3. 분할
Fig. 3 Decomposition

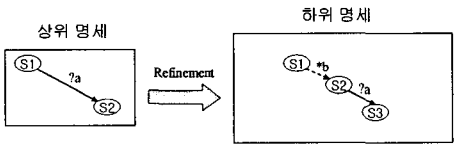


그림 4. 정제
Fig. 4. Refinement

전체적으로 DEVS 정형론을 바탕으로 한 시스템 개발은 그림 5와 같은 분할과 정제가 반복적으로 이루어지는 계층적 단계를 따른다. 제일 첫 번째 명세는 사용자의 요구명세르 반영하고 있으며 맨 마지막 단계에서 설계되어져 나온 명세는 구현 직전의 단계로서 프로토타입 구현에 이용된다.

2.3. 올바른 계층적 설계

정의 1) 구조적 등가

깊이 l인 두 모델 M_1과 M_2의 외부입출력 사건집합이 같고, 이들을 내포하는 깊이 $k=l, l-1, ..., 1$에 대한 결합모델들 간의 입출력 사건 집합이 같고 이들이 내포하는 깊이 1인 대응하는 모델들간의 입출력사건 집합 및 연결관계가 같을 때, 우리는 두 모델을 구조적 등가(structural equivalence)라고 한다.

정의 2) 행위적 등가

두 모델 M_1과 M_2의 주어진 초기상태들로부터 입출력 순서가 똑 같을 때 이것을 행위적 등가(behavioral equivalence)라고 부른다.

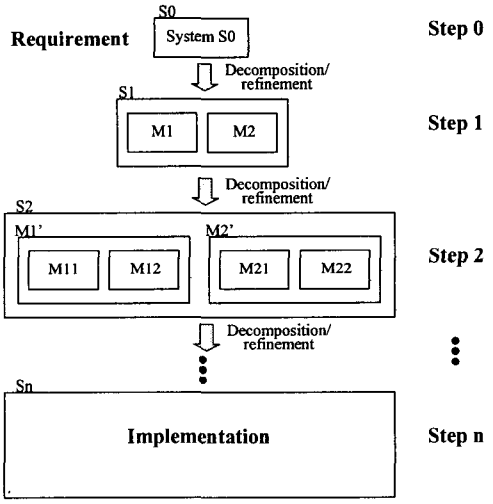


그림 5. DEVS 정형기법의 계층적 설계 방법
Fig. 5. Hierarchical Design Strategy of DEVS Formal Method

정의 3) 올바른 계층적 설계 방식

임의의 상위명세 S_1이 분할과 정제를 통해 하위 명세 S_2를 얻었을 때 만약 S_1과 S_2의 구조적 등가인 내부 모델들이 동시에 행위적 등가이면 우리는 구현명세 S_2는 상위명세 S_1의 올바른 계층적 설계 (proper hierarchical deployment)에 따른 구현이라고 하고 모든 설계단계에서 이 관계가 성립할 때 이를 올바른 계층적 설계방식이라고 말한다.

예를 들어, 그림 5에서 원자모델 M_1과 구체적으로 설계된 결합모델 M_1', 마찬가지로 M_2와 M_2'가 각각 구조적 등가이면서 행위적 등가이면 명세 S_2는 S_1의 올바른 계층적 설계에 따른 구현이라고 말한다.

Ⅲ. DEVS bisimulation

구현된 시스템이 시스템 요구명세를 만족하는 가를 측정하는 한 척도는 입출력 관점에서의 행위적 등가인지를 검증하는 것이다. 이를 행위검증이라 하며, 행위검증 방법은 지금까지 여러 방법들이 제안되었다. 대표적인 것으로 homomorphism [1], language acceptance checking, weak bisimulation에 의한 검증[7] 등을 들수 있다. 본 논문에서는 세 가지 방법 가운데 시간을 고려하지 않은 무시간 DEVS/DEVS 모델에 대한 weak bisimulation에 의한 검증을 적용한다.

두 DEVS 원자모델 P와 Q에 있어서 각각의 초기값 p와 q가 주어질 때 이를 프로세서 P(p)와 프로세스Q(q)라고 부르자. 두 프로세스 P(p)와 Q(q)에 대한 weak bisimulation 관계를 P(p) ≈ Q(q) 라고 할 때 재귀적으로 weak bisimulation 은 다음과 같이 정의된다.

정의 4) Weak bisimulation 관계

$P(p) \approx Q(q)$ iff for each $\alpha \in EF(p) \cup EF(q)$

(1) whenever $p \xrightarrow{\alpha} p'$ then, for some q', $q \xrightarrow{\alpha} q'$

and $P(p') \approx Q(q')$.

(2) whenever $q \xrightarrow{\alpha} q'$ then, for some p', $p \xrightarrow{\alpha} p'$ and $P(p') \approx Q(q')$.

여기에서 $p \xrightarrow{\alpha} p'$ 는 state p에서 이벤트 α가 발생되어 state p'로 도달함을 의미하며, $p \Rightarrow p'$는 $p \xrightarrow{(\tau)^*\alpha(\tau)^*} p'$를 뜻한다. $EF(p)$는 state p에서 실행될 수 있는 이벤트들의 집합을 의미하고, τ는 숨겨진 이벤트(hidden event)를 의미한다.

DEVS/DEVS(요구명세/구현명세) weak bi- simulation 검증 방법은 그림 6과 같은 구조를 갖는다.

첫 단계에서는 검증 대상이 되는 두 모델이 원자모델 혹은 결합모델로서 DEVS 명세언어로 주어진다.

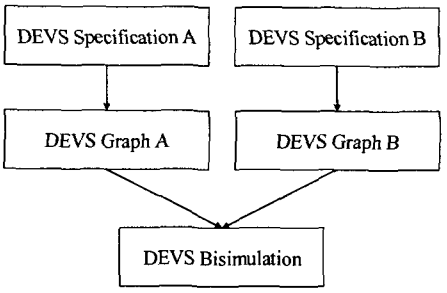


그림 6. 검증과정
Fig. 6. Verification Process

단 논리분석을 위하여 기존의 DEVS 원자모델의 몇 가지 특성이 변경되었다. 먼저 한 상태에서 다수의 내부천이가 정의되는 병렬성(non-sequential)이 추가되었다. 다음으로 DEVS모델의 시간 전진 함수를 무시한 무시간(untimed) 모델을 다룬다.

둘째 단계에서는 두 모델들을 논리분석에 유리한 형태인 DEVS 그래프로 변환한다. DEVS 그래프는 일종의 reachability 그래프로 제 6절 예제의 그림과 같은 모양으로 기술된다. 결합모델은 등가의 원자모델로 합성(composition)[8] 시킨 후 다음 단계를 행한다.

마지막 단계에서는 변환된 DEVS 그래프에 대하여 weak bisimulation 검증을 수행한다. DEVS 그래프는 사실상 labeled Transition System (LTS)[9]과 같은 구조이다. 따라서 DEVS 그래프를 이용하면 기존의 LTS를 위한 bisimulation에 관한 효율적인 알고리즘들이 있지만 여기서는 정의 4)를 그대로 적용한 알고리즘을 사용하였다.

Ⅳ. DEVS 정형기법의 계층적 검증 방법

그림 5에서 요구명세 S_0이 계속 분할과 정제를 반복하여 시스템 구현명세 S_n이 얻어질 때, 기존의 검증방법은 S_n을 합성하여 얻어진 행위모델이 S_i의 행위와 등가인지를 검증하는 것이다. 하지만 구체화 단계 i가 증가할수록 원자모델 수와 합성상태수는 지수함수적으로 증가하므로 기존의 방법으로는 상태폭발을 피할 수 없다.

제안된 계층적 검증방법은 복잡한 시스템의 행위 검증에 있어서 구체화 단계별로 여러 부분으로 나누어서 검증하는 분할정복(divide-concur) 방법의 일종으로서 계층적 설계 방식으로 개발되는 시스템에만 적용할 수

있는 검증방법이다. 정의 1)에 의해 계층적 설계 방식으로 분할된 모델사이에는 하위명세로 가더라도 더 이상 상호작용이 새로이 정의되지 않는다. 따라서, 일단 상위 명세의 검증이 끝난 후에는 이후로 분할된 모델을 더욱 구체화시키더라도 전체적으로 합성하여 검증할 필요가 없고 오로지 구조적 등가인 모델끼리 검증하면 된다.

예를 들면 그림 5에서 기존의 검증방법으로 시스템명세 S_2가 S_0을 만족하는지를 검증하기 위해서 M11, M12, M21, M22를 모두 합성하여 등가인 원자모델을 얻은 후 S_0와 행위적 등가임을 검증해야 한다. 그러나 계층적 검증에서는 S_1의 M1과 M2를 합성하여 S_0과 이미 행위검증을 통과하였고, S_1의 M1과 S_2의 M1', M2와 M2'가 각각 구조적 등가라면, M1과 M1', M2와 M2'가 각각 행위적 등가인지만을 검증하면 S_2와 S_0의 검증이 끝나게 된다.여기서 계층적 검증방법으로 검증해야 할 구조적 등가인 모델들은 S_0과 S_1, S_1의 M1과 S_2의 M1', S1의 M2와 S_2의 M2'의 세 개다.

다음 이론에서는 계층적 검증 방법에 의한 검증이 기존의 검증 방법의 결과와 같다는 것을 보여준다.

이론 1) $S_0, S_1, S_2, ..., S_n$이 올바른 계층적 개발 방법으로 구체 설계되었다고 가정하자. 만약 모든 $i = 1,2,...,n$에 대해 S_{i-1}와 S_i간의 모든 구조적 등가모델들 간에 weak bisimulation 관계가 성립된다면 S_0과 S_n은 weak bisimulation 관계가 성립한다.

증명) (by induction) 먼저 단계 $i = 1$일 때. S_1이 S_0에 대해서 만족함을 보인다. 정의 3)에 의해 S_1과 S_0는 구조적 등가이면서 행위적 등가이다. 따라서 S_1은 S_0를 만족한다. 임의의 단계 $i > 1$에서 S_i가 S_0와 행위적 등가라고 가정하자. 단계 $i+1$에서 S_{i+1}과 S_i의 구조적 등가모델들을 $M_{i+1,k}$와 $M_{i,k}$, $k = 1,...,m$이라고 하면 가정에 의해 $M_{i+1,k}$와 $M_{i,k}$와는 행위적 등가이다. S_{i+1}의 $M_{i+1,k}$ 대신에 등가원자모델 $M'_{i+1,k}$를 넣어 S'_{i+1}이라고 할 때 S'_{i+1} 는 S_i와 구조적으로 완전히 같고 동작만이 $M_{i,k}$ 대신 $M'_{i+1,k}$로 바뀌었음을 알 수 있다. 어떤 내부 모델 대신 행위적 등가인 다른 모델을 서로 교체한 후 합성한 전체 행위는, 바꾸지 않았을 때 합성한 전체 행위와 같다는 정리[8]에 의해 결국 S'_{i+1}의 합성은 S_i의 합성과 같은 결과를 얻는다. 그런데, S_i는 S_0와 행위적 등가이므로 당연히 S_{i+1}도 S_0와 등가임이 증명된다.

V. 계층적 검증 방법의 효율성

기존의 검증방법보다 계층적 검증방법이 한번에 검증해야할 최대 상태 수 (혹은 복잡도) 면에서 낫다는 것을 보이기 위해 다음과 같은 시스템을 가정하자. 사용자의 요구 명세가 원자모델 S_0로 주어지고 이로부터 그림 5와 같은 시스템 개발 과정을 통해 최종 구현 모델S_n를 얻었다고 가정하자. 임의의 단계 i에서 Step i을 설계할 때 S_{i-1}의 각 원자모델이 S_i에서 각각 평균 m개의 원자모델을 가진 결합모델로 분할되고 모든 원자모델은 평균 k개의 일정한 상태 수를 가진다고 가정하자.

이 때 기존의 방법으로 적합성 검증을 수행할 때 요구명세와 S_0와 구현명세 S_n에 대하여 검증 대상이 되는 최대 상태 수는 다음과 같다.

$$Maximum\ \#\ of\ states = k + k^{m^n} \quad (1)$$

위 식에서 최종 구현 모델을 구성하는 원자모델을 수는 m^n개이다. 즉 적합성 검사의 복잡도가 최종 구현모델을 구성하는 원자모델의 수와 지수 관계에 있다. 이는 최종 모델의 구성이 약간만 복잡해지면 결국 상태 폭발이 발생함을 의미한다.

계층적 방법을 사용할 때 임의의 요구모델과 그 모델의 구조적 등가인 구현 모델의 검증시 대상이 되는 최대 상태 수는 식 (2)과 같다. 여기서 k는 요구모델의 상태 수이고 k^m은 구현모델의 합성된 최대 복합상태수이다.

$$\#\ of\ states\ for\ a\ bisimulation = k + k^m \quad (2)$$

$$\#\ of\ bisimulations = \sum_{i=1}^{n} m^{i-1} = \frac{m^n - 1}{m - 1} \quad (3)$$

$$total\ \#\ of\ states = (k + k^m) \sum_{i=1}^{n} m^{i-1}$$

$$= (k + k^m) \frac{m^n - 1}{m - 1} \quad (4)$$

위 식에서 $m \gg 1, k^m \gg k$로 가정하고 근사값을 구하면 (1)식은 $k^m \cdot k^{m^{n-1}}$ (4)식은 $k^m \cdot m^{n-1}$가 얻어진다. 이들을 각각 (1'),(4')이라 하자. (4')식이 의미하는 바는 원자모델이 늘어나더라도 한 번 계산할 때 k^m만큼의 일정한 비용으로 m^{n-1}만큼 반복하여 계산함을 의미한다. 반면, (1')식은 한 번에 $k^m \cdot k^{m^{n-1}}$ 만큼의 상태 수에 대해 검증해야 함을 의미하고 n이 증가함에 따라 지수함수적으로 늘어나서 상태폭발 문제가 쉽게 야기됨을 볼 수 있다. 반면 계층적 검증에서는 상태수가 k^m으로 일정하다. 또한 전체비용면에서도 (4')식이 (1')식에 비교해 볼 때 훨씬 적음을 알 수 있다.

따라서 계층적 검증 방법은 적합성 검증에 있어 상태폭발 문제를 완화시킬뿐더러, n이 커질수록 전체 계산량 또한 크게 낮다. 특히 시스템이 복잡하고 개발 단계가 많아질수록 큰 효과를 볼 수 있다. 하지만 실제 효율 면에서는 위에서 비교된 것만큼 향상이 있지는 않다. 이는 여러 모델들이 합성되어 최종 구현 모델을 구성할 때 전체 상태공간은 위의 계산보다 훨씬 작은 영역을 갖기 때문이다.

VI. 예제 : PingPong Double-Player Model

DEVS 정형론을 바탕으로 시스템을 개발하고, 요구명세와 구현 모델사이의 적합성 검증을 수행하는 과정을 복식 탁구경기 모델을 통하여 보여 주고자 한다.

계층적 설계: 그림 7은 두 단계를 거쳐 개발된 탁구경기의 모델들이다. PP0은 어느 한 쪽 팀이 서비스를 하도록 하면 이어서 어느 한 쪽이 OUT되고 다시 이를 반복하도록 한 주어진 요구명세이다. PP0 구현은 일 단계에서 PP1로 구체화되고 PP1은 다시 이 단계에서 PP2로 계층적인 설계방식을 따라 구체화되었다. PP1은 원자모델 PP0의 구조적 등가모델로 내부에 원자모델 Team_A와 Team_B를 구조적 등가모델인 결합모델 Team_A'과 Team_B'로 구체화 시켰다. 각각의 결합모델 Team_A'와 Team_B'는 각각 선수 1과 2 그리고 TC로 분할하여 구체화하였다. 그림 8은 PP2의 TEAM_A'를 원자모델까지 자세히 나타낸 것이다. TEAM_B'는 TEAM_A'과 같다. 올바른 계층적 설계임을 보려면 구조적 등가뿐만 아니라 구조적 등가인 모델간의 행위도 등가임을 증명하여야 한다. 그러면 정리 3)과 이론 1)에 의해 행위검증이 끝나게 될 것이다.

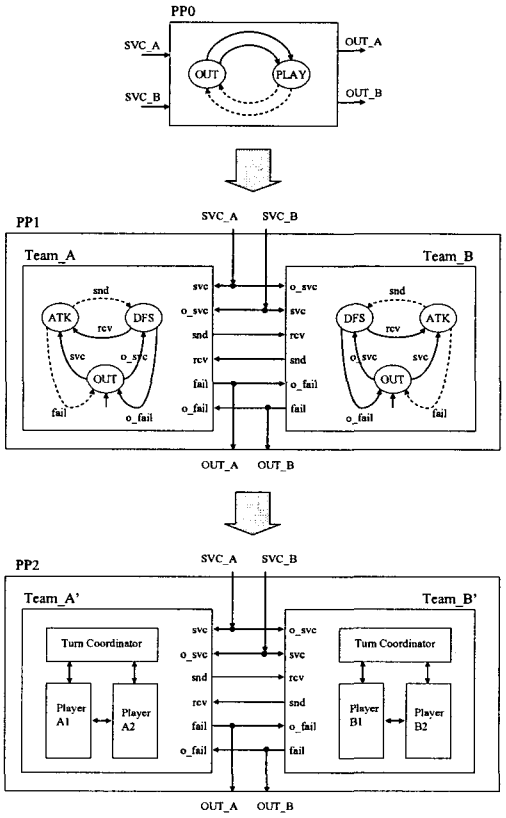


그림 7. 탁구복식경기의 계층적 모델링
Fig. 7. Hierarchical Modeling of PingPong Doubles Game

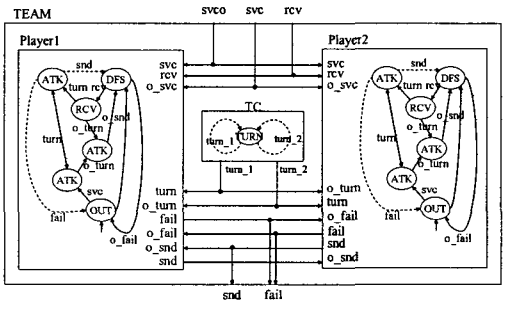


그림 8. 탁구팀 모델
Fig. 8. Model of Player Team

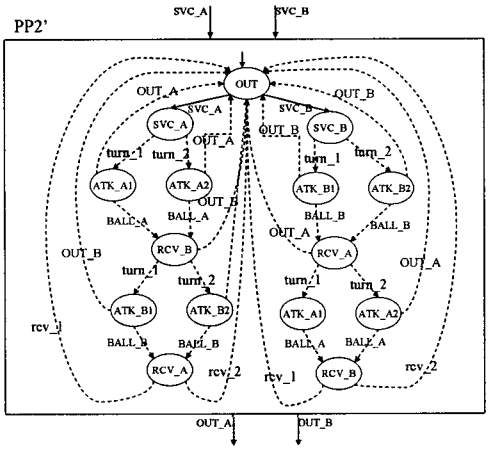


그림 9. 합성된 PP2의 등가 원자모델
Fig. 9. Equivalent Atomic Model of Synthesized PP2

계층적 검증: 계층적 검증 단계에서는 각 개발 단계에서 구조적 등가 모델끼리 계속 행위적 등가임을 증명해 나가야 한다. PP0은 P1과 구조적 등가이므로 그에 대한 검증을 수행한다.

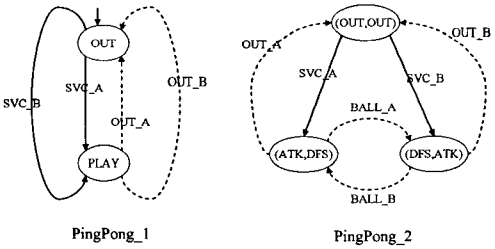


그림 10. PP0과 PP1의 등가원자모델
Fig. 10. Equivalent Models of PP0 and PP1, Respectively

그림 10은 PP0과 PP1의 등가 원자모델로서 정의 4)에 의해 쉽게 행위적 등가임을 알 수 있다. 두 번째 개발 단계에서 PP1의 Team_A 및 Team_B가 PP2의 Team_A' 및 Team_B'로 구조적 등가이므로 그에 대한 검증을 수행한다. 이때 Team_A와 Team_B는 같은 모델이므로 한번의 검증만을 수행하면 된다. 같은 방법으로 PP1과 PP2의 모든 구조적 등가 모델들에 대해 검증한 결과 역시 모두 만족함을 알 수 있었다. 따라서 PP1과 PP2는 올바른 계층적 설계방식에 따른 PP0의 구현이며 (정리 3)이는 이론 1)에 의해 PP1과 PP2는 PP0을 만족함이 증명된다. 이는 기존의 검증방법으로 재검증 해 보더라도 만족함을 볼 수 있었다. 실제로 검증은 그림 6을 구현한 자동화된 DEVS/DEVS bisimulation 증명 도구를 사용하였다.

복잡도: 적합성 검증 과정의 복잡도를 비교하면 다음과 같다. PP0과 PP2에 대하여 직접 bisimulation 검증을 수행할 때 검증대상의 상태 수는 PP0 2개, PP2 15개 합하여 17개이다. 반면 계층적 검증에 대한 상태수는 총 23개이다. 즉, PP0은 2개, PP2는 3개, Team_A는 3개, Team_A'는 6개, Team_B는 3개, Team_B'는 역시 6개의 상태를 갖는다. 전체 상태 수는 이 경우 시스템이 매우 간단하고 합성 시 동기 되는 사건이 많았기 때문에 그 차이가 미미하였다. 그러나 한번에 계산해야할 상태 수는 최대 17개에서 9개로 줄었음을 알 수 있다.

Ⅶ. 결 론

본 논문에서는 DEVS 정형론을 바탕으로 한 계층적 시스템 설계 방법과 이에 따른 새로운 계층적 검증방법을 소개하였다. 계층적 시스템 설계 방법에서는 구조적 등가와 행위적 등가에 대해 정의하였고 두가지 등가를 만족하면서 분할과 정제를 통한 구체화하는 방법을 올바른 계층적 개발 방법이라고 정의하였다. 올바른 계층적 개발 방법을 사용한 시스템의 검증은 계층적 검증방법을 사용하였으며 이 때 초기명세와 최종명세 간의 검증이 성립함을 보였다. 이 검증 방법에서는 한번에 검증해야 하는 상태의 수가 구체화 수준이 깊어지더라도 일정한 수를 가지므로 상태폭발 문제가 해결된다는 것이 가장 큰 효과이다. 다만 올바른 계층적 개발 방법을 사용하지 않고 개발된 시스템에 대해서는 이러한 방법을 적용할 수 없다는 단점이 있다. 향후 시간을 포함한 strong bisimulation을 정의하고 여기에 근거한 실시간 모델의 검증에 관한 연구가 이루어져야 할 필요가 있다.

참고문헌

[1] B. P. Zeigler, Multifacetted Modelling and Discrete Event Simulation, Academic Press, 1984.

[2] 김탁곤 외, 프로세스 명세에 관한 동적 행위 분석기법에 관한 연구, 시스템공학연구소 연구보고서, 1998.

[3] G. P. Hong and T. G. Kim, "A Framework for Verifying Discrete Event Models within a DEVS-based System Development Methodology," Trans. Society for Computer Simulation International, vol. 13, pp. 19-34, 1996.

[4] M. H. Hwang and F. Lin, "State Minimization of SP-DEVS,"" Lecture Notes in Computer Science, Vol.3397, pp.243-252, Springer 2005.

[5] K.J. Hong and T.G. Kim, "DEVSpecL-DEVS Specification Language for Modeling, Simulation and Analysis of Discrete Event Systems," Information and Software Technology pp.221-234, Vol.48, No.4, 2006.

[6] C. Thomas, H. Luckhoff, and T.G. Kim, "OpenDEVS: A Proposal for a Standardized DEVS Model Exchange Format," Proc. 6th Conf. A1, Simul, Planning in High Autonomous Systems, pp. 371-377, 1996.

[7] R. Milner, Communication and Concurrency, Prentice Hall, 1989.

[8] H.S. Song and T.G. Kim, "Application of Real-Time DEVS to Analysis of Safety-Critical Embedded Control Systems: Railroad Crossing Control Example," Simulation, Vol.81, No.2, pp. 119-136, 2005.

[9] R.Cleavaland, J. Parrow, B. Steffen, "Testing Equivalence as a Bisimulation Equivalence," Automatic Verification Methods for Finite State Systems, LNCS 407, pp.11-23, Springer, 1989.

저자소개

송 혜 상(Hae-Sang Song)

2000년 8월 KAIST 전자전산학과 박사
2002년 9월 ~ 현재 서원대학교 컴퓨터정보통신공학부 교수

※관심 분야 : 이산사건시스템 모델링 해석, 임베디드 시스템, 소프트웨어공학, 시스템공학

이 완 복(Wan-Bok Lee)

2004년 2월 KAIST 전자전산학과 박사
2003년 3월 ~ 2007년 2월 중부대학교 컴퓨터공학부 교수
2007년 3월 ~ 현재 공주대학교 게임디자인학과 교수

※관심 분야 : 시뮬레이션, 컴퓨터 게임, 이산 사건 시스템, 정보보호