비즈니스 프로세스 타당성 검증 기법*

설주영, 박정업, 김학수, 신영재, 장진근, 박찬희, 김태인, 손진현
한양대학교 컴퓨터공학과
{jysul, jupark, hagsoo, yjshin, jgjang, parkch, tikim, jhson}@cse.hanyang.ac.kr

Validation Checking Mechanisms of Business Processes

Joo Young Sul, Park Jung Up, Kim Hak Soo, Youngjae Shin,
Jang Jin Gun, Park Chan Hee, Kim Te In, Jin Hyun Son
Department of Computer Science and Engineering, Hanyang University, Ansan

요 약

과거에 비해 최근에 비즈니스 프로세스가 다양해지고 복잡해짐에 따라 비즈니스 프로세스를 디자인할 때 발생할 수 있는 문제점이 점차 증가하고 있다. 그로인해 비즈니스 프로세스 검증의 중요성이 높아지고 있지만 많은 검증 방법이 제안되고 있지 않은 실정이다. 한편, 최근 BPMI에서 주도하는 BPMN(Business Process Modeling Notation)은 비즈니스 프로세스를 위한 표준화된 그래픽 표기법으로써 BPMN을 지원하는 디자인 툴을 이용하면 다양하고 복잡한 프로세스 환경을 일반화된 형태로 디자인하고 분석할 수 있다. 이 논문에서는 이러한 BPMN으로 디자인할 때 발생할 수 있는 문제점들을 효율적으로 검증할 수 있는 몇 가지 검증기법을 제시한다. 이로 인해 비즈니스 프로세스 실행 시 발생할 수 있는 문제점들을 효율적으로 검증할 수 있는 몇 가지 검증기법을 제시한다. 이로 인해 비즈니스 프로세스 실행 시 발생할 수 있는 에러들을 사전에 검증하여 예기치 못한 큰 비용을 줄일 수 있다.

키워드: BPMN, 비즈니스 프로세스, 검증, π-calculus

1. 서 론

현재 비즈니스 환경이 과거에 비해 더욱 복잡해짐에 따라 비즈니스 프로세스 디자이너가 효율적으로 디자인할 수 있도록 하는 검증 툴의 필요성이 증가되고 있는 실정이다. 비즈니스 프로세스의 검증 목적은 비즈니스 프로세스 디자인할 때 발생할 수 있는 다양한 문제점들을 비즈니스 프로세스 실행 전에 검증함으로써 예기치 못한 큰 비용을 줄이는 것이다[3,5]. 따라서 비즈니스 프로세스 디자인에 대한 타당성 검증은 무엇보다 중요하다. 그러나 현재 다양하고 복잡해진 비즈니스 프로세스가 제대로 디자인되었는지 검증할 수 있는 응용프로그램이 많지 않은 실정이다. 비즈니스 프로세스 타당성 검증 툴은 비즈니스 프로세스 관리 시스템에서 상당히 중요한 부분에 속한다. 그 이유는 제대로 설계되지 않은 프로세스 디자인은 프로세스가 원하는 과정으로 진행하지 않을 수 있으며, 실행 도중에 예기치 않은 결과 초래 혹은 고객에게 불편함을 줄 수 있다.

비즈니스 프로세스를 제대로 디자인하기 위해서는 표준화된 그래픽 표기법을 사용하는 것이 바람직하다. 최근에 부각을 나타내고 있는 표준화된 그래픽 표기법으로는 BPMN(Business Process Modeling Notation)[7], UML 엑티비티 다이어그램[8], UML EDOC 비즈니스 프로세스, IDEF, ebXML BPSS, ADF(Activity-Decision Flow) 다이어그램, RosettaNet, LOVeM, EPCs(Event Process Chains) 등을 예로 들 수 있다. 그 중 BPMN은 다른 어떤 비즈니스 프로세스 모델링 표기법보다 많은 기능과 확장성을 갖고 있기 때문에, 본 논문에서는 BPMN을 사용해서 디자인할 경우 발생할 수 있는 에러를 검증하고자 한다.

비즈니스 프로세스 검증 단계를 크게 데이터 흐름에 대한 측면[6]과 컨트롤 흐름에 대한 측면[2] 두 가지로 나눌 수 있다. 데이터 흐름에 대한 측면은 비즈니스 프로세스 내의 태스크(Task) 사이에서 전달되는 데이터가 정상적인 순서로 이동되어 사용되는지 혹은 중복, 손실되지 않는 지 등을 검증한다. [6]에서는 데이터가 이동될 때 발생할 수 있는 문제점을 6가지로 구분하였다. 컨트롤 흐름에 대한 측면은 수행될 태스크가 결정된 후 올바른 진행 순서대로 태스크가 디자인되어 있는 지를 검증한다. 대표적인 컨트롤 플로우 4가지 이상현상으로는 특정 단계에서 다음 단계로 진행할 수 없을 경우(데드락), 의도치 않게 특정 태스크가 여러 번 수행될 경우(동기화의 부족), 시작과 종료가 없는 경우(시작과 종료가 없는 노드) 그리고 무한하게 특정 태스크들을 수행할 경우(무한루프)가 있다. 이 논문에서는 이 두 가지 측면 중 컨트롤 흐름에서 발생할 수 있는 문제점에 대한 검증으로 제한한다.

문제점 검증의 절차는 다음과 같다. 먼저 BPMN1.0 스펙에서 제시한 디자인 기본 사항을 준수한 프로세스를 플로우 오브젝트 변환 과정을 통해 정형화된 컨트롤 플로우 프로세스로 변환한다. 그 후에 프로세스 수학적 모델인 π-calculus로 비즈니스 프로세스 내의 모든 노드들을 변환 후 이 논문에서 제시한 검증 기법을 통해 다양한 이상현상을 검증하였다.

논문의 구성은 다음과 같다. 2장에서는 관련 연구로써 기존의 검증 기법들과 비교 및 분석을 기술한다. 3장에서는 본 논문의 이해를 위해 BPMN과 강력한 비즈니스 프로세스 수학적 모델인 π-calculus를 요약 기술한다. 4장에서는 BPMN에서 π-calculus로의 변환 방법을 기술한다. 5장에서는 프로세스 검증을 단순화하기 위해 BPMN의 흐름주체인 플로우 오브젝트를 다양한 형태로 변환하는 기법을 제시한다. 6장에서는 왜 프로세스 컨트롤 흐름 상에서 문제가 발생할 수 있는 지를 분석하고 그에 대한 검증 기법을 제시한다. 7장에서는 결론을 맺는다.

2. 관련 연구

Petri nets는 비즈니스 프로세스를 디자인할 수 있고 검증 또한 할 수 있기 때문에 널리 사용되어 왔다[13]. Petri net 기반의 WF-net의 이점은 정형화된 이론에 기반을 두고 있으며 토큰 기반의 워크플로우 상태 표현, 상세한 분석 및 검증 툴을 들 수 있다. 그러나 이런 장점에도 불구하고 현재 대부분의 WfMS에서 WF-net을 채택해서 사용하고 있지 않다. 그 이유는 다양한 종류의 표기법을 단순히 세 개의 컴포넌트로 표현에 제약을 두고 있기 때문에 이해하고 운영하기 어렵다는 단점이 있다. 또한 [2]에서는 과거에 비해 다양하고 복잡해진 비즈니스 프로세스의 기능들을 상당 부분에서 표현하지 못하고 있기 때문에 그에 대한 타당성 검증이 제약적임을 볼 수 있다. 예를 들어, 프로세스 간의 통신, 이벤트 호출, OR 라우팅 등에 대한 표현 방식이 언급되어 있지 않다. 또한, 타당성 검증 측면에선 데드락, 동기화의 부족, 시작과 종료가 없는 노드 등은 검증이 가능하나 무한루프에 대한 검증방법은 언급되어 있지 않다.

그래프 축소(Graph Reduction)는 데드락과 동기화의 부족 이상현상에 대해서만 검증 가능하다[3]. 프로세스 내에 남아있는 노드

* 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 육성·지원 사업(IITA-2005-C1090-0502-0016)의 연구결과로 수행되었음.

들에게 다섯 개의 축소 규칙을 반복적으로 적용하여 줄여 나가는 방식으로써 데드락과 동기화의 이상현상을 프로세스가 포함하고 있으면 프로세스 내의 전 노드가 삭제가 되지 않는다. 이 방식의 장점으로는 그래프 축소 알고리즘의 복잡도가 최악의 상황인 경우 O(n2)으로 상당히 성능이 좋다는 것이다. 하지만 루프를 포함하고 있으면 이 방식을 사용할 수 없다는 것과 이 기법 역시 단순한 기능을 표현할 수 있는 표기법에 대해서만 검증하고 있다.

논리 기반 접근 방식(Logic-based approach)은 명제 논리학(propositional logic)을 사용해서 프로세스의 이상현상을 검증하는 방법으로써 프로세스 내의 노드들을 논리적 표현으로 변환 후 이 논문에서 제한하는 프로세스 추론에 의해 간단한 논리적 표현으로 줄여 나가는 방식이다[4]. 이 방식은 알고리즘의 복잡도가 O(n2)로 성능이 좋고 대표적인 4가지 이상현상 전부를 검증을 할 수 있다는 장점이 있지만 중첩된 구조를 검증할 수 없고 단순한 구조만 검증 가능하다는 단점을 포함하고 있다.

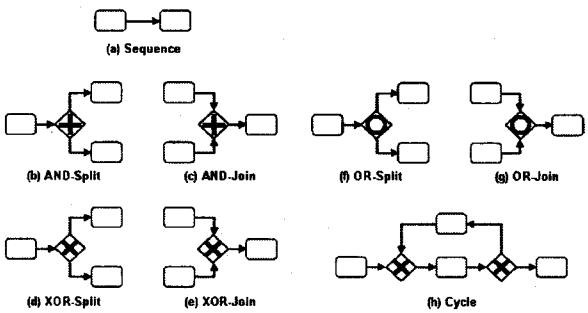
현재의 비즈니스 프로세스는 과거에 비해 다양하고 복잡해졌다. 따라서 프로세스를 디자인할 때 사용되는 모델들도 많은 부분 추가되었다(예를 들면, 프로세스 간의 통신, 이벤트 호출, OR 라우팅 등). 그러나 위의 3가지 검증 기법을 보면 상당히 단순한 형식의 프로세스를 가정에 두고 있다. 또한, 대표적인 4가지 이상현상을 모두를 처리할 수 있는 기법이 없다. 가장 대표적인 검증 기법인 Petri nets 기반의 방식 역시 무한 루프가 있는 경우를 처리할 수 있다는 언급이 없다. 이 논문에서는 복잡한 프로세스를 디자인 할 수 있는 강력한 표준화 모델인 BPMN을 기반으로 4가지 이상현상을 검증하는 기법을 제공한다. 그러나 프로세스 간의 통신은 제외하였다.

3. 배경 지식

3.1 BPMN의 기본 구성 요소

프로세스 내의 기본 구성 요소로는 그림 1에서와 같이 8가지로 이루어져 있다.

(a) Sequence : 특정 플로우 오브젝트에서 이와 연결된 다음 플로우 오브젝트로의 이동
(b) AND-Split : AND 게이트웨이에서 나가는 방향에 있는 모든 플로우 오브젝트를 동시에 실행시킴.
(c) AND-Join : AND 게이트웨이로 들어오는 방향에 있는 모든 플로우 오브젝트의 객체를 동기화 시킴.
(d) XOR-Split : XOR 게이트웨이에서 나가는 방향에 있는 모든 플로우 오브젝트 중 단지 하나의 플로우 오브젝트만을 실행시킴.
(e) XOR-Join : XOR 게이트웨이로 들어오는 흐름을 동기화하지 않고 바로 다음 단계로 진행시킴.
(f) OR-Split : OR 게이트웨이에서 나가는 방향에 있는 모든 플로우 오브젝트 중 특정 조건을 만족시키는 플로우 오브젝트만을 실행시킴.
(g) OR-Join : OR 게이트웨이의 이전 Split 조건에서 만족되어 들어온 모든 흐름을 동기화시킴.
(h) Cycle : 특정 조건을 만족할 때까지 특정 범위를 반복 수행함.



(그림 1) 프로세스 기본 구성 요소

위의 8가지 구성요소를 복합적으로 사용하여 다양한 비즈니스 프로세스를 디자인할 수 있다. WfMC(Workflow Management Coalition)에 의해서 (a)-(e) 그리고 (h)만이 워크플로우 기본 구성 요소로 정의되어 있다[14]. 그리고 다른 여러 논문에서도 OR 게이트웨이를 언급하지 않는 경우가 많다. 그 이유는 구현하기 어렵고 AND와 XOR 게이트웨이를 혼합해서 사용하면 가능하다는 것이다. 하지만 여러 상업화된 응용프로그램에서 OR 게이트웨이를 지원하고 있고 BPMN 엘리먼트로도 사용되기 때문에 이 논문에서는 OR 게이트웨이가 포함되었을 경우도 검증할 수 있는 방법을 제시하고 있다. 그러나 더욱 복잡한 라우팅 구조를 표현할 수 있는 콤플렉스 게이트웨이(Complex gateway)는 생략하였다.

3.2 대표적인 비즈니스 프로세스 이상현상

비즈니스 프로세스 디자인할 때 발생할 수 있는 대표적인 이상현상은 다음과 같이 4가지로 분류된다[3,4].

(a) 데드락(Deadlock) : 특정 단계에서 다음 단계로 진행할 수 없을 경우를 말한다.
(b) 동기화의 부족(Lack of Synchronization) : 의도치 않게 특정 태스크가 여러 번 수행될 경우를 말한다.
(c) 시작과 종료가 없는 엑티비티(Activities without termination 수 없는 엑티비티를 시작이 없는 엑티비티라고 말한다.or without activation) : 특정 엑티비티가 종료로 도달할 수 없을 경우 종료가 없는 엑티비티라고 하고 시작 이벤트에 의해서 실행됨
(d) 무한루프(Infinite Cyles) : 특정 범위 내의 태스크들을 무한하게 수행할 경우를 말한다.

(a), (b) 그리고 (c)에 대한 검증기법을 이 논문에서 제시한다. 그러나 '시작과 종료가 없는 엑티비티'에 대한 검증은 이 논문에서 제외하였다. 그 이유는 이 사항에 대해서는 BPMN 1.0 스펙에서 기본 준수 사항으로 제시하였기 때문이다.

3.3 가정

1. BPMN 1.0 Specification에 제시된 디자인 기본 사항을 모두 준수한 비즈니스 프로세스를 검증함.
2. 디자인된 비즈니스 프로세스는 반드시 시작이벤트와 종료이벤트로 시작하고 종료된다.
3. 데이터 플로우 측면에 대한 검증 방법은 제외하고 컨트롤 플로우 측면 대해서만 검증한다.
4. 비즈니스 간의 통신인 협력 프로세스는 검증에서 제외한다.

4. BPMN을 π-calculus로 매핑

π-calculus는 모든 워크플로우 패턴을 손쉽게 표현할 수 있을 뿐만 아니라 프로세스 간의 통신 또한 표현 가능하다[11,12]. 이 논문에서는 각각의 플로우 오브젝트를 프로세스 수학적 모델인 π-calculus로 변환 후 각 플로우 오브젝트의 포트와 채널정보를 이용하여 이상현상을 검증하려 한다. 그러므로 BPMN 내의 플로우 오브젝트를 π-calculus로 변경하는 방법을 알아야 한다. [11]에서는 워크플로우 패턴을 π-calculus로 정형화하는 방법에 대해 자세히 표현하고 있다.

5. BPMN의 변환

이 장에서는 BPMN1.0 Specification에서 제시한 디자인 기본 사항을 준수한 프로세스를 5가지 플로우 오브젝트 변환 과정을 통해 정형화된 컨트롤 플로우 프로세스로 변환하는 방법을 제시한다.

5.1 정형화된 컨트롤 플로우 프로세스

정의 1.

1. 모든 엑티비티는 이전 플로우 오브젝트로부터 들어오는 순차 플로우와 다음 플로우 오브젝트로 향하는 순차 플로우를 각각 최대 한 개씩으로 제한한다.
2. 모든 게이트웨이는 Split과 Join 중 하나의 용도로만 사용한다.
3. 프로세스는 엑티비티, 게이트웨이 그리고 순차 플로우로만 구성된다.

정의 1을 만족시키는 정형화된 컨트롤 플로우 프로세스는 아래에서 제시하는 5가지 변환과정을 거친다.

1. 한 개 이상의 플로우 오브젝트와 연결된 태스크의 경우 태스크와 게이트웨이로 분리시킴.

2. Join과 Split 두 가지 역할을 동시에 수행하는 게이트웨이의 경우 한 가지 역할만을 수행할 수 있도록 두 개의 게이트웨이로 분리한다.
3. 링크 이벤트의 경우, 순차 플로우와 직접 연결한다.
4. 중간 이벤트가 엑티비티에 부착되어 있는 경우, 엑티비티와 XOR 게이트웨이로 변환하여 수행한다.
5. 4와 5의 변환 과정 후 남아있는 모든 이벤트들을 엑티비티로 변환한다.

위의 다섯 가지 변환 과정을 통해 정형화된 컨트롤 플로우 비즈니스 프로세스는 다소 복잡한 비즈니스 프로세스가 단순화된다. 이 단순화된 비즈니스 프로세스에 이 논문에서 제시한 검증 기법을 적용함으로써 보다 쉽게 타당성 검증을 할 수 있게 된다.

6. 비즈니스 프로세스 타당성 검증

이 장에서는 비즈니스 프로세스 디자인할 때 문제점이 발생할 수 있는 상황에 대해서 집중 분석하고 그 이상현상들을 검증할 수 있는 방법을 제시한다. 검증 접근 방식은 단일 스트림 상에서의 검증 방식과 복합 스트림 상에서 검증하는 방식으로 구분하여 접근하였다.

단일 스트림은 비즈니스 프로세스의 흐름이 한 방향으로만 진행되는 상황을 말한다. 즉, 루프가 없는 프로세스를 말한다. 복합 스트림 상에서는 프로세의 흐름이 한 방향으로 흐르다가 역류해서 뒤에서 흐르던 흐름이 앞에서부터 다시 흘러 특정 구역에서는 두 흐름이 지날 수 있다. 즉, 프로세스 내에 루프가 존재는 경우이다.

6.1 문제 분석

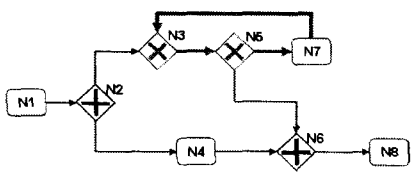
6.1.1 단일 스트림 상에서의 문제 분석

특정 Split 게이트웨이로부터 나온 두 개 이상의 경로가 서로 합쳐질 때 반드시 같은 타입의 Join 게이트웨이에서 만나야 한다. 분리된 모든 경로가 최종적으로 한 곳으로 모이기 이전에 그 중 최소 두 개의 경로라도 합쳐질 경우 또한 해당된다.

6.1.2 복합 스트림 상에서의 문제 분석

올바른 루프는 루프 내에 반드시 두 종류의 게이트웨이를 포함한다. 한 게이트웨이는 루프를 형성하는데 사용되어지고 다른 하나는 루프를 벗어나는 용도로 사용된다. 그림 2을 보면, N3은 루프 형성에 N5는 루프를 벗어날 때 사용되어졌음을 볼 수 있다. 만약에 N5 역할을 하는 게이트웨이가 존재하지 않으면 루프를 빠져 나갈 수 있는 길이 없으므로 무한루프가 발생하게 되고 N3이 없으면 루프를 형성할 수 없게 된다. 또한 이 두 가지의 게이트웨이는 Join과 Split의 한 쌍으로 존재하여 사용되어지는 것이 아니라 독립적으로 사용되어진다. 그러므로 복합 스트림이 존재하는 프로세스의 경우 이 게이트웨이의 존재 여부를 반드시 파악해야 한다.

이 두 가지 게이트웨이가 잘못 사용될 경우도 존재할 수 있다. 만약 N3가 AND Join 게이트웨이로 사용되어진다면 데드락 현상을 발생하고 N5가 AND나 OR로 사용되어진다면 N6이 여러 번 발생할 수 있는 상황인 동기화의 부족 현상을 초래할 수 있다.



(그림 2) 복합 스트림 상의 비즈니스 프로세스

6.2 단일 스트림 상에서의 타당성 검증

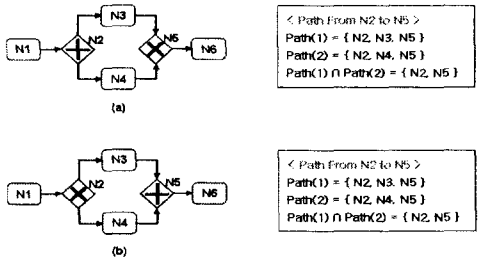
6.2.1 게이트웨이 불일치 (Gateway mismatch)

정의 2. 한 쌍의 게이트웨이 x와 y(한쪽은 XOR이고 다른 한쪽은 AND인 경우)에 대해서 x에서 y로 이르는 하나 이상의 경로가 존재할 경우, x에서 y로 이르는 특정 한 쌍의 경로 path(1)과 path(2)에 대해, path(1)∩path(2) = {x,y}이면 하나 이상의 이상현상이 발생한다.

이 검증기법은 데드락과 동기화의 부족 이상현상 문제를 검증할 수 있다.

6.2.1.1 기본 구조에서 이상현상 검증

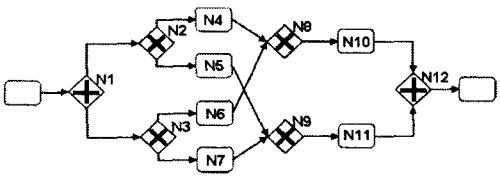
그림 3을 보면, 서로 다른 게이트웨이 N2와 N5에 대해 N2에서 N5로 이르는 경로가 두 개 이상 존재하고 그들의 교집합 결과가 N2와 N5임을 볼 수 있다. 이때, (a)와 같이 XOR Join 게이트웨이로 합쳐질 경우는 동기화의 부족 이상현상이 발생하고 (b)와 같이 AND Join 게이트웨이로 합쳐질 경우는 데드락 현상이 발생한다.



(그림 3) 기본 구조로 구성된 프로세스 내의 이상현상 검증 (1)

6.2.1.2 중첩구조에서 이상현상 검증

게이트웨이 불일치 검증기법을 이용하면 중첩구조에 대해서도 검증할 수 있다. 중첩구조란 두 개 이상의 게이트웨이가 선행하는 두 개 이상의 게이트웨이의 흐름을 공유하는 패턴이 존재하는 경우를 말한다. 예를 들어, 그림 4를 보면 게이트웨이 N8과 N9는 선행하는 게이트웨이 N2와 N3의 흐름을 공유하고 있다. 그러나 그림 4의 중첩된 구조는 여러 가지 문제를 내포하고 있다. N8과 N9에서 동기화의 부족 이상현상이 발생할 수 있으며 N12에서는 데드락이 발생할 가능성이 있다. 이렇게 중첩된 구조에서도 게이트웨이 불일치 검증기법의 서로 다른 게이트웨이 사이의 경로를 이용하면 쉽게 검증이 가능하다.



(그림 4) 중첩된 구조를 포함한 프로세스 내의 이상현상 검증

6.3 게이트웨이 불일치 검증기법의 타당성 증명

게이트웨이 불일치 검증기법의 타당성을 정형화하기 위해 보조정리와 정리를 제시하였다.

보조정리 1 (Lemma 1) 루프가 없는 구조에서 데드락과 동기화의 부족 이상현상을 최소한 하나라도 포함하고 있는 프로세스가 있다면 한 쌍 이상의 게이트웨이가 반드시 불일치한다.

토의 (Discussion) 이전 장에서 논하였듯이 비즈니스 프로세스 이상현상들은 서로 다른 두 게이트웨이 사이의 경로를 통해 검증될 수 있었다. 이때, XOR Split과AND Join의 불일치는 데드락 이상현상을 AND Split과 XOR Join의 불일치는 동기화의 부족 이상현상을 초래하였다.

정리 1 (Theorem 1) 프로세스 내의 모든 게이트웨이에 대해서, 서로 다른 한 쌍의 게이트웨이들이 알맞게 배치돼 사용되었다면 그 컨트롤 플로우 프로세스 내엔 이상현상이 발생하지 않는다.

반증에 의한 증명. 정리 1을 거짓이라고 하자. 다시 말하면, 프로세스 내에 하나 이상의 이상현상을 포함하고 있는 경우를 말한다. 이 경우 보조정리 1에 의해 프로세스 내엔 최소한 한 쌍의 게이트웨이 불일치가 존재한다. 이것은 서로 다른 한 쌍의 게이트웨이가 잘 배치돼 사용되어야 한다는 것에 대한 모순이다. 그러므로 정리 1은 참이어야 한다. □

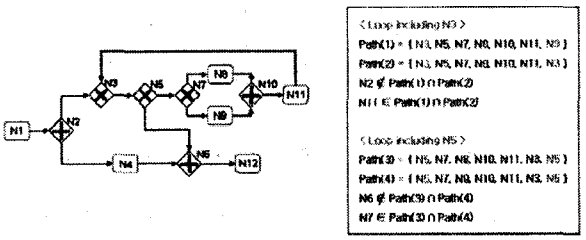
6.4 복합 스트림 상에서의 타당성 검증

6.4.1 단일 XOR 캐치 I (Single XOR catch I)

정의 3. 루프가 있을 때 (즉, XOR Join 게이트웨이가 두 번 발생

가능), 루프 내에 최소한 한 개의 XOR Split 게이트웨이가 반드시 존재해야 한다. 또한, XOR Split 게이트웨이로부터 나가는 방향에 연결되어 있는 플로우 오브젝트들 중 하나라도 루프 밖에 존재해야 한다. 그렇지 않으면, 무한 루프가 발생한다. 이 Split 게이트웨이는 게이트웨이 불일치 검증기법에 의해 체크가 될지라도 올바른 패턴으로 인정한다.

단일 XOR 캐치 검증기법 I 을 통해 루프 존재 시 루프 벗어나는 용도로 사용되어지는 게이트웨이 존재 여부를 체크한다. 그리고 이 게이트웨이는 서로 Split과 Join 쌍으로 존재하지 않고 독립적으로 사용되기 때문에 게이트웨이 불일치 검증기법에 의해 체크가 될지라도 올바른 패턴으로 인정해야 한다.



(그림 5) 루프를 포함하고 있는 프로세스 내의 이상현상 검증 (1)

그림 5을 보면, 서로 다른 게이트웨이 N5와 N6에 대해 N5에서 N6로 이르는 경로가 두 개 이상 존재하고 그들의 교집합 결과가 N5와 N6이므로 게이트웨이 불일치 검증기법에 의해 데드락 이상현상이 반환되어야 하지만 N5는 단일 XOR 캐치 I 검증기법에 의해 올바른 패턴으로 인식된다.

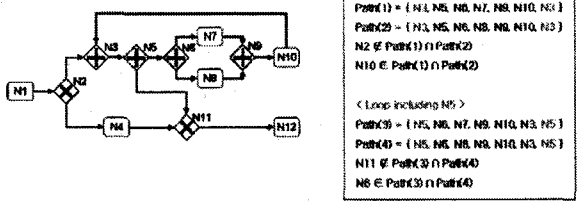
6.4.2 단일 XOR 캐치 II (Single XOR catch II)

정의 4. 같은 XOR Join 게이트웨이가 두 번 이상 발생할 수 있고 (즉, XOR 게이트웨이를 포함하는 루프가 존재 가능) XOR Join 게이트웨이의 바로 앞에 선행해 있는 플로우 오브젝트 중 최소한 한 개라도 루프 내에 존재하지 않을 경우, 게이트웨이 불일치 검증기법에 의해 오류 이상현상이 발견될지라도 올바른 패턴으로 인정한다.

단일 XOR 캐치 검증기법 II 을 통해 루프를 형성하는데 사용되어지는 XOR Join 게이트웨이를 올바른 패턴으로 인식한다. 이 게이트웨이 또한 서로 Split과 Join 쌍으로 존재하지 않고 독립적으로 사용되기 때문에 게이트웨이 불일치 검증기법에 의해 체크가 될지라도 올바른 패턴으로 인정해야 한다. 그림 5을 보면, 서로 다른 게이트웨이 N2와 N3에 대해 N2에서 N3로 이르는 경로가 두 개 이상 존재하고 그들의 교집합 결과가 N2와 N3이므로 게이트웨이 불일치 검증기법에 의해서 동기화의 부족 이상현상이 반환되어야 하지만 N3은 이상 AND/OR 캐치 II에 의해 올바른 패턴으로 인식된다.

6.4.3 이상 AND/OR 캐치 I (Wrong AND/OR catch I)

정의 5. 같은 AND/OR Split 게이트웨이가 두 번 이상 발생할 수 있고(즉, AND/OR 게이트웨이를 포함하는 루프가 존재 가능) AND/OR Split 게이트웨이부터 나가는 방향에 연결되어 있는 플로우 오브젝트 중에 루프 내에 존재하지 않는 오브젝트가 존재하면 동기화의 부족 이상현상이 발생할 수 있다.



(그림 6) 루프를 포함하고 있는 프로세스 내의 이상현상 검증 (2)

이상 AND/OR 캐치 검증기법 I 을 통해 루프 존재 시 루프 벗어나는 용도로 사용되어지는 XOR 게이트웨가 AND나 OR로 오용될 경우를 검증할 수 있다. 그림 6을 보면, AND Split 게이트웨이인 N5로 인해 N11이 여러 번 수행됨을 볼 수 있다. 이와 같은 이상현상 발생은 상 AND/OR 캐치 I 검증기법에 의해 체크된다. 그리고 AND Split 게이트웨인 N6 역시 두 번 이상 수행될 수 있지만 N7과 N8이 루프 내에 전부 존재하기 때문에 이 패턴은 올바른 패턴으로 인정된다.

6.4.4 이상 AND/OR 캐치 II (Wrong AND/OR catch II)

정의 6. 같은 AND Join 게이트웨이가 두 번 이상 발생할 수 있고 (즉, AND 게이트웨이를 포함하는 루프가 존재 가능) AND Join 게이트웨이의 바로 앞에 선행해 있는 플로우 오브젝트 중 루프 내에 존재하지 않는 오브젝트가 존재하면, 데드락 이상현상이 발생하게 된다.

이상 AND/OR 캐치 II 검증기법을 통해 루프를 형성하는데 사용되어지는 XOR Join 게이트웨이가 AND게이트웨이로 오용될 경우를 검증할 수 있다. 그림 6을 보면, AND Join 게이트웨이인 N3으로 인해 다음 단계로 진행할 수가 없다. 이렇게 AND Join 게이트웨이로 인해 데드락이 발생할 수 있는 경우는 이상 AND/OR 캐치 II 검증 기법에 의해 쉽게 해결된다. 그리고 N9 역시 두 번 이상 발생할 지라도 N7과 N8이 루프 내에 존재하기 때문에 올바른 패턴으로 인정된다.

6.5 루프 구조에서 검증기법의 타당성 증명

루프가 존재하는 구조에서 여기서 제시한 검증기법들의 타당성을 정형화하기 위해 보조정리와 정리를 제시하였다.

보조정리 2 (Lemma 2) 루프가 존재하는 구조에서 데드락, 동기화의 부족 이상현상 그리고 무한루프를 최소한 하나라도 포함하고 있는 프로세스가 있다면 한 쌍 이상의 게이트웨이 불일치가 존재하거나 루프에 필요한 두 종류의 게이트웨이가 잘못 사용되어진 경우이다.

토의 (Discussion) 이전 장에서 논하였듯이 비즈니스 프로세스 이상현상들은 서로 다른 두 게이트웨이 사이의 경로를 통한 검증, 루프에 반드시 필요한 두 종류의 게이트웨이의 존재 그리고/혹은 오용 여부 검증을 통해 해결되었다.

정리 2 (Theorem 2) 프로세스 내의 모든 게이트웨이에 대해서, 서로 다른 한 쌍의 게이트웨이들이 알맞게 배치돼 사용되고 루프에 반드시 필요한 두 종류의 게이트웨이가 자신의 역할에 맞게 사용되어진다면 그 컨트롤 플로우 프로세스 내엔 이상현상이 발생하지 않는다.

반증에 의한 증명. 정리 2를 거짓이라고 하자, 다시 말하면, 프로세스 내에 하나 이상의 이상현상을 포함하고 있는 경우를 말한다. 이 경우 보조정리 2에 의해 프로세스 내엔 최소한 한 쌍의 게이트웨이 불일치가 존재하거나 루프에 존재해야 하는 두 종류의 게이트웨이가 잘못 사용된 경우가 존재한다. 이것은 서로 다른 한 쌍의 게이트웨이가 잘 배치돼 사용되고 루프에 반드시 필요한 두 종류의 게이트웨이가 자신의 역할에 맞게 사용되어져야한다는 것에 대한 모순이다. 그러므로 정리 2는 참이어야 한다. □

6.6 OR 게이트웨이 검증 방법

OR 게이트웨이가 Join으로 사용되어지면 어떠한 Split 게이트웨이와 사용되어져도 문제가 발생되지 않는다. 그 이유는 OR Join 게이트웨이가 앞 조건에서 참으로 들어오는 모든 흐름을 기다리기 때문에 어디에 놓여도 문제가 되지 않는다. 하지만 추가 계산을 해줘야 한다는 단점이 있다. OR Split 게이트웨이로 사용되어지면 어떤 Join 게이트웨이와 사용되느냐에 따라 발생될 수 있는 문제가 다르다. AND Join 게이트웨이와 사용되면 데드락을 유발할 수 있고 XOR Join 게이트웨이와 사용되면 동기화의 부족 현상을 초래할 수 있다. 그 이유는 OR Split 게이트웨이는 조건에 따라 하나의 흐름에서 연결된 전체의 흐름까지 다음 단계로 진행시킬 수 있는 역할을 담당하기 때문이다. 그렇기 때문에 이상현상 검증 시에는 문제점이 발생할 수 있는 최악의 경우 게이트웨이로 대체해서 검증해야 한다. 즉, AND Join 게이트웨이와의 연산 시에는 XOR Join 게이트웨이로 설정한 후 검증 시작하고 XOR Join 게이트웨이와는 AND Join 게이트웨이로 여기고 검증을 시작해야 한다는 것이다. 이 논문에서 제시한 검증 방법이 서로 다른 두 종류의 경로

를 가지고 이상현상을 결정한다는 점에서 OR 게이트웨이는 이렇게 두 종류의 게이트웨이로 변경 후 체크가 가능하다.

7. 결론 및 향후 과제

본 논문에서는 BPMN으로 비즈니스 프로세스를 디자인할 때 발생할 수 있는 대표적인 세 가지 이상현상들을 효율적으로 검증할 수 있는 여러 기법들을 제시했다. 검증방법은 프로세스를 플로우 오브젝트 변환 과정을 통해 정형화된 컨트롤 플로우 프로세스로 변환하고 프로세스 수학적 모델인 π-calculus로 비즈니스 프로세스 내의 모든 노드들을 변환한 후 세 가지 검증 기법을 통해 다양한 이상현상을 검증하였다. 특히, OR 게이트웨이와 이벤트를 포함하고 있는 경우도 처리할 수 있으며 중첩 구조와 루프가 존재하는 구조에 대해서도 쉽게 검증할 수 있다. 향후 과제로는 비즈니스 프로세스 간의 통신인 협력 프로세스를 검증할 수 있도록 검증기법을 확장할 계획이다. 또한 이 논문에서 제시한 검증 기법을 적용한 툴을 구현할 예정이다.

8. 참고 문헌

[1] W.M.P. van der Aalst, Verification of workflow nets, in: The 18th International Conference on Application and Theory of Petri Nets, Lecture Notes in Computer Science, Vol. 1248 (1997) pp. 407-426.

[2] W.M.P van der Aalst. Workflow Verification: Finding Control-Flow Errors Using Petri-Net-Based Techniques. Business Process Management, Models, Techniques, and Emmrical Studies, 2000.

[3] W.Sadiq and M.E. Orlowska, Analyzing process models using graph reduction techniques. Information System 25(2), 2000.

[4] HENRY H. BI. Applying Propositional Logic to Workflow Verification. Information Technology and Management, Volume 5, 2004.

[5] W.M.P. van der Aalst and Arthur H.M. ter Hofstede, Verification of workflow task structures: A Petrinet-based approach, Information Systems 25(1) (2000) 43-69.

[6] Shazia Sadiq. Data Flow and Validation in Workflow Modeling. Proceedings of the fifteenth conference on Australasian database - Volume 27 (ACM), 2004.

[7] Businss Process Modeling Notation(BPMN) (version 1.0 - May 3, 2004).

[8] OMG Unified Modeling Language Specification(version 1.4, 2001,9, http://www.omg.org.

[9] Robin Milner, Communicating and Mobile System: the π -Calculus, Cambridge University Press, May 1999.

[10] Joachim Parrow. An Introduction to the π-Calculus. Chapter to appear in Handbook of Process algebra, ed. Bergstra, Ponse and Smolka Elsevier, 1999.

[11] Frank Puhlmann, Mathias Weske: Using the π-Calculus for Formalizing Workflow Patterns. In: W.M.P. van der Aalst et al. (Eds.): BPM 2005, volume 3649 of LNCS, Berlin, Springer-Verlag (2005) 153-168.

[12] Y. Dong and Z. ShenSheng, Using π-calculus to Formalize UML Activity Diagrams, Proceedings of the 10th International Conference and Workshop on the Engineering of Computer-based Systems, IEEE Press, NY(2003), pp. 47-54.

[13] W. Reisig and G. Rozenberg, editors. Lectures on Petri Nets I: Basic Models, volume 1491 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1998.

[14] WfMC, Workflow management coalition terminology & glossary(WFMC-TC-1011, Issue 3.0), Workflow Management Coalition (1999).