韓國 컴퓨터情報學會 論文誌
第12卷 第3號, 2007. 7.

2007-12-3-4-1

프로세스 대수를 이용한 XML 문서의 접근권한 표현법

이 지 연*, 김 일 곤**

A Method for Specifying the Access Control of XML Document using Process Algebra

Ji-Yeon Lee *, Il-Gon Kim **

요 약

웹서비스 기술의 활성화와 더불어, XML 문서에 대한 접근통제기술이 새롭게 개발되었다. 그 결과, 기존의 접근통제시스템에서와 마찬가지로, 중요정보 자원에 대한 접근통제문제에 대한 연구가 큰 각광을 받고 있다. 본 논문에서는 XML 문서에 대한 접근통제정책을 정형명세언어인 CSP로 변환하는 방법을 제시한다. 이를 위해, 첫째, XPath 경로지정 언어로 표현된 XML 문서의 계층적 접근을 CSP 프로세스 알제브라로 변환하는 방법을 소개한다. 둘째, XML 스키마 문서를 오토마타 형태의 정형모델로 표현하는 방법을 설명한다. 셋째, XML 접근통제정책에 사용되는 적용 규칙 및 충돌규칙의 의미론을 프로세스 알제브라 언어로 표현하는 방법을 제시한다. 마지막으로, 본 논문에서 제시한 방법론의 타당성을 보이기 위해, XML 스키마 문서 및 경로지정 표현에 대한 CSP 명세 예제를 보여준다.

Abstract

With the increase of a web service technology, a new access control mechanism has developed for XML documents. As a result, as legacy access control systems, access control systems has become an active research topic. In this paper, we propose a methodology to translate access control policies for XML documents into formal specification language CSP. To do this, first, we introduce a method to translate a hierarchical access to XML documents using XPath language into CSP process algebra. Second, we explain a method to represent a XML schema as a formal model like automata. Third, we present a method for representing the semantics of access control policies such as the scope of rules and confliction resolution into a process algebra language. Finally, a CSP specification example of an XML schema and path expressions are shown to illustrate the validity of our approach.

▸ Keyword : XML, XPath, Formal Specification, Process Algebra, CSP

• 제1저자 : 이지연
• 접수일 : 2007.5.29, 심사일 : 2007.6.20, 심사완료일 : 2007. 7.20.
* 동남보건대학 e-비즈니스과 조교수 ** 한국정보보호진흥원 보안성평가단 평가1팀 선임연구원
※ 이 논문은 2006년도 동남보건대학 연구비 지원에 의하여 수행된 것임

Ⅰ. 서 론

XML기반 웹서비스 분야의 활성화와 더불어, 정보를 요청한 사용자가 XML 트리로 구성된 정보에서 허가받은 데이터 영역에만 접근하도록 하기 위한 보안 문제점이 대두되었다. 즉, XML 트리 정보는 일반 데이터 정보와 달리 트리 형태로 구성되어 있기 때문에, 데이터의 접근을 정의하기 위한 보안 정책이 트리의 데이터 노드들을 표현할 수 있어야 한다.

이러한 트리의 데이터 노드들의 보안성(예, 읽기 허가/거부 등)을 표현하기 위해 데이터베이스 영역에서는 경로지정 언어(예, XPath[1])가 새롭게 개발되었다. 예를 들어, 비밀정보 XML 문서 D에 대한 쿼리 q가 주어졌을 때, 정보 요청자가 접근 할 수 있는 계층적인 XML 트리 경로를 표현할 수 있다. 이는 비 계층적인 데이터 구조와 달리 XML 데이터의 계층적 구조로 인해 비밀성 및 권한허가 문제점을 분석하기 매우 어려운 특성을 갖고 있다. 예를 들어, XML 데이터 보안은 다음과 같은 이유로 인해, 일반 데이터 보안 문제점 분석에 비해 어려운 특징을 갖고 있다.

- 접근통제 규칙은 트리 경로를 표현하기 위해 '경로 표현식'을 이용하여 표현된다.
- 비 계층적 데이터와 달리, XML 노드는 부모노드와 자식노드간의 구조적인 특성을 갖기 때문에, 정보 요청자가 허가되어 접근할 수 있는 데이터 영영 또한 구조적인 특성을 갖게 된다.
- 유닉스와 같은 일반 파일 시스템에서는 모든 데이터에 대한 접근통제 규칙이 정의되어져야 한다. 그러나, XML 문서의 경우에는 모든 노드에 대한 접근 통제 규칙을 명시할 필요는 없다.

XML 문서의 접근통제규칙을 명시하기 위한 지금까지의 연구방향은 자연어 형태에 의존하여 왔기 때문에, 명세언어의 모호성으로 인해 접근통제 정책의 명확성 및 데이터 비밀성 문제점을 분석하는데 비효율적인 문제점을 갖고 있다. 또한, XML 문서에 대한 접근통제규칙의 복잡성과 오류로 인해, 허가받지 않은 사용자가 특정 중요 XML 데이터 노드영역에 접근할 수 보안 문제점을 야기 시킬 수 있다. 본 논문에서는 CSP (Communicating Sequential Processes)[2]와 같은 프로세스 알제브라 언어를 이용하여, XML 문서의 계층적인 접근통제영역을 정형적으로 표현할 수 있는 방법을 제시하여, XML 문서의 접근통제정책 문제점을 검증할 수 있는 기반을 마련하고자 한다.

본 논문의 구성은 다음과 같다. 제Ⅱ장에서는 관련연구를 소개한다. 제Ⅲ장에서는 XPath 경로지정 언어에 대해 간략히 언급한다. 제Ⅳ장에서는 CSP 프로세스 알제브라 언어를 이용하여, XML 문서에 대한 접근통제규칙을 명시하는 방법을 설명한다. 마지막으로 제Ⅴ장에서는 결론 및 향후 연구방향을 제시한다.

Ⅱ. 관련연구

보안 시스템의 보안성을 정형적으로 명세하기 위해 다양한 연구가 진행되어 왔으며, 그중에서도 CSP 정형명세 언어를 이용하여 보안 프로토콜의 보안성을 검증하기 위한 연구가 큰 성과를 거두어 왔다[3]. 하지만, 정형기법을 보안 분야에 적용하는 연구는 일반적으로 보안시스템 영역 분야보다는 보안프로토콜에 집중되어 왔다. 특히, 정형명세 언어를 이용하여 XML 문서기반 접근통제 시스템 정형적으로 명세한 연구는 거의 진행되어 오지 않았기 때문에 본 연구는 기존 연구와 차별화 된다.

XML 문서에 대한 대부분의 접근통제 표현방법은 XPath와 같은 쿼리 언어를 이용하여 XML의 보호영역을 지정할 수 있도록 하는 연구가 진행되어 왔다[4,5,6]. Irini[7]는 XPath 언어를 이용하여 XML 문서에 대한 접근통제 정책을 표현하는 방법을 제시하였다.

Ⅲ. XPath 경로지정 언어

XML 문서에 대한 접근을 규정하는 대부분의 방법들은 XPath 언어를 이용하여 문서 혹은 문서의 노드에 대한 보안정책을 명시한다. 이에 본 논문에서는 XPath 언어를 이용하여 기술된 보안정책을 CSP 프로세스 알제브라 언어로 변환하는 방법을 제시하고자 한다.

XPath는 W3C에 의해 제안되었으며, 매우 잘 정의된 구문과 의미론을 제공한다. 현재 개발된 XML 구문 분석도구와 높은 호환성을 갖고 있으며, 이 도구를 통해 XML 문서의 관련 엘리먼트 및 속성을 검색할 수 있도록 지원해 준다. 본 논문에서는 XML 문서의 노드들을 지정하도록 하는 XPath 언어에서 사용되는 'node', '/', '//', '*' 및 '@' 이와 같은 주된 몇 가지 표현방식을 사용한다.

- node : 노드의 모든 자식노드들을 선택하기 위해 사용
- / : 루트 노드부터 시작하여 엘리먼트 이름을 구분하기 위해 사용되는 표현식

- // : 검색 노드와 일치하는 노드 선택을 위한 표현식
- * : 모든 적용 가능한 노드를 나타내기 위해 사용
- @ : 노드 속성 값을 나타내기 위해 사용

```
〈Records〉
  〈Patient @Noe = David〉
    〈Medical〉
      〈Doctor〉 김의사〈/Doctor〉
      〈Diagnosis〉 암 〈/Diagnosis〉
      〈Prescription〉
    〈/Medical〉
  〈/Patient〉
〈/Records〉
```

그림 1. XML 문서 예제
Fig 1. Optical Flow Constraint Line

그림 1에서 보이는 XML 문서 예제를 통해, 앞에서 간단한 언급한 XPath 언어 표현식의 이해를 돕고자 한다.

- /Records : 이 표현식은 Records 노드를 가리킴
- /Records/* : 이 표현식은 Records 노드의 모든 자식 노드를 가리킴
- /Records//Medical : Medical 노드의 자식 노드들을 가리킴
- /Records/Patient[Name=David] : 'Name' 속성 값이 'David'인 Patient 노드를 가리킴

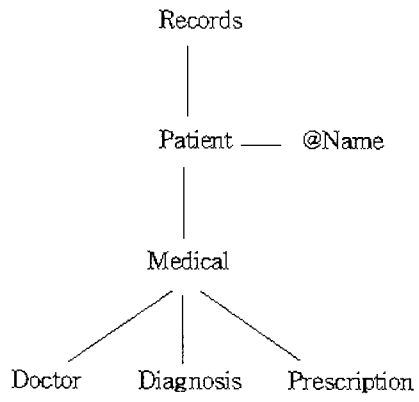

그림 2. 전자의료기록카드의 스키마 구조
Fig 2. Schema Structure of Electronic Medical Records

IV. XML 문서에 대한 접근통제정책

4.1 스키마 레벨의 접근통제

XML 문서에 대한 접근통제정책은 일반적으로 4개의 튜플로 구성되어 있다: 〈주체, 객체, 행동, 효과, 범위〉

- 주체: 객체에 대한 접근을 요청한 사용자, 그룹 또는 프로세스
- 객체: 주체가 접근하고자 하는 대상으로 XPath 언어로 표현됨
- 행동: 주체가 객체에 대해 수행하는 읽기, 쓰기 및 삭제와 같은 행동들
- 효과: '+'와 '-' 기호로 구성되며, 전자는 객체에 대한 접근 허용을 의미하고, 후자는 객체에 대한 접근 거부를 의미함
- 범위: 규칙을 명시하는 'PL', 'PR', 'NL' 및 'NR' 요소로 구성

접근통제정책은 단일 XML 문서상에서 혹은 스키마 레벨에서 명시할 수 있다. 스키마 레벨의 접근통제정책의 경우에는 스키마에 일맥상통하는 모든 XML 문서에 접근통제정책을 적용할 수 있기 때문에, 객체에 대한 접근권한을 명시하는데 매우 효율적인 방법이다. 따라서, 본 논문에서는 스키마 레벨에서 접근통제정책을 규정하는 방법을 선택하였다.

4.2 접근통제 규칙의 의미론

XML 기반 웹서비스 제품에서 사용되는 접근통제모델은 일반적으로 다음과 같은 특징을 기준으로 접근통제규칙의 의미론을 규정한다.

4.2.1 규칙의 적용 범위

XML 문서의 계층적 특징으로 인해, 규칙의 적용 범위는 웹서비스 기반 접근통제시스템을 구축하는데 매우 중요한 요인이 된다. 아래 4개의 집합요소들(PL, PR, NL, NR)을 이용하여 규칙의 적용 범위를 규정하도록 하겠다.

- PL(Positive Local) : 허용(grant) 규칙의 적용범위는 단지 특정 로컬 노드로만 한정
- PR(Positive Recursive) : 허용 규칙의 적용 범위는 특정 노드와 그 자식 노드까지 적용가능
- NL(Negative Local) : 거부(deny) 규칙의 적용범위는 단지 특정 노드로만 한정

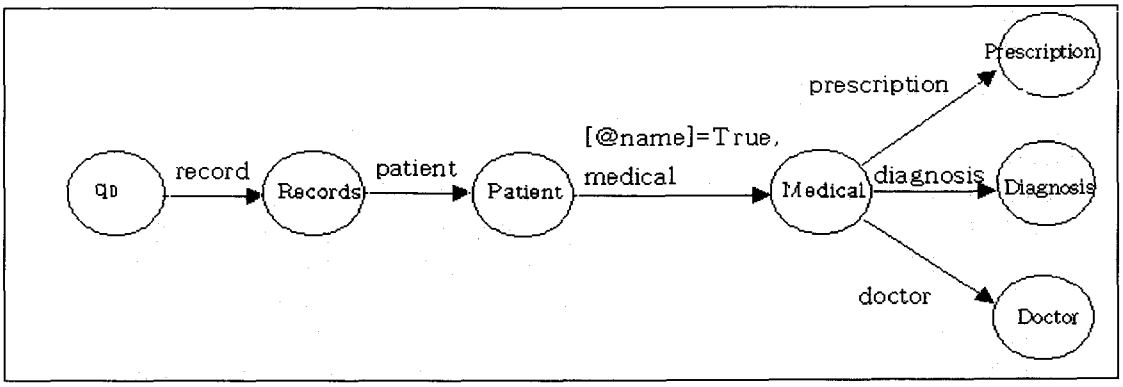

그림 3. 스키마 정형모델 M_S 예제

Fig 3. Example of Schema Formal Model M_S

· NR(Negative Recursive) : 거부 규칙의 적용범위는 특정 노드와 그 자식 노드까지 적용가능

예를 들어, "Bob 이라는 이름을 가진 주체는 객체 Diagnosis 노드에 대해 읽기 권한을 갖고 있다"는 보안규칙을 명시하고 싶다면, 다음과 같은 보안정책을 정의할 수 있다.

〈Bob, /Records/Patient[@Name=홍길동]/Medical, read, +, PR〉

규칙 적용범위가 PR로 명시되어 있기 때문에, Bob은 Medical 노드의 자식노드들에 까지 읽기 권한을 갖게 된다.

만일 다음과 같은 보안정책이 정의되어 있다면,

〈Bob, /Records/Patient/[@Name=홍길동/Medical, read, -, NL〉

이 보안정책은 "사용자 Bob은 단지 Medical 노드영역(자식 노드 제외)에 대해서만 읽기 권한을 갖는다"는 의미이다.

4.2.2 규칙 충돌

앞에서 언급한 바와 같이, 보안정책은 허용('+'로 표기) 규칙 또는 거부('-'로 표기) 규칙을 갖을 수 있다. 따라서, 만일 어떤 한 노드에 대한 접근규칙이 허용 또는 거부로 명시될 수 있다. 이런 경우, 두 가지 규칙의 충돌의 어떻게 해석 하는지가 매우 중요한 문제로, 본 논문에서는 '규칙충돌'이란 용어를 사용하고 있다.

이런 규칙충돌을 해결하기 위해 일반적으로 두 가지 알고리즘이 사용된다.

첫째는 'grant overwrites' 알고리즘으로, 동일한 노드에 대해 허용규칙이 거부규칙에 우선하는 방식이다. 둘째는 'deny overwrites' 알고리즘으로, 이 경우는 동일한 노드에 대해 거부규칙이 허용규칙에 우선하는 방식이다.

예를 들어, 다음과 같이 주체 Bob에 두 가지 규칙들이 동시에 적용되어 있고,

〈Bob, /Records/Patient/[@Name=David/Medical, read, +, PL〉
〈Bob, /Records/Patient/[@Name=David/Medical, read, +, PL〉

deny overwrites 알고리즘을 선택하였다면, 거부규칙이 허용규칙에 우선하기 때문에, 사용자 Bob은 Medical 노드에 대한 읽기 권한을 갖지 못하게 된다. 반대로, 만일 grant overwrites 알고리즘을 선택하였다면, 사용자 Bob은 Medical 노드에 대한 읽기 권한을 갖게 된다.

V. 프로세스 알제브라를 이용한 접근통제정책 정형명세

본 장에서는 앞에서 언급한 XML 문서에 대한 접근통제 의미론을 CSP 언어로 정형명세 하는 방법을 제시하고자 한다. 이를 위해, 본 논문에서는 우선적으로 XML 문서에 대

한 접근통제규칙을 명시한 XPath 표현식을 오토마타 형태의 스키마 모델로 변환하는 방법을 사용하였다. 그런 다음, 오토마타형 스키마 모델을 CSP 프로세스 알제브라 언어로 정형명세 하는 방법을 이용하였다. CSP 표현에 대해 보다 상세한 내용은 [2]를 참고하기 바란다.

5.1 스키마 정형모델

오토마타 이론[8]에 근거하여, 스키마에 대한 정형 모델을 생성하였다. 스키마 s가 주어졌을 때, XML 문서에 대한 수학적인 정형모델 M_s는 6개의 튜플로 구성되어 있다:

$M_s = (T, \Sigma_e, \Sigma_a, \delta, N, q_0)$

- T : 초기상태 q_0와 XML 트리의 비말단(non-terminal) 노드 집합 N으로 구성된 상태집합
- Σ_e : XML 트리의 엘리먼트 이름을 나타내는 입력 알파벳
- Σ_a : 속성 이름 집합
- δ : $T\times(\{\varepsilon\}\cup G)\times\Sigma_e \rightarrow 2^T$ 전이함수 (단, G는 Σ_a 속성 값에 의해 결정되는 대수 표현)
- N : 최종 상태들의 집합
- q_0 : XML 트리의 루트노드로 전이하기 위한 입력 알파벳

예를 들어, 그림1에서 보여진 XML 문서예제에 대한 스키마 문서(그림2)의 경우, 앞에서 언급한 수식을 적용하면, 다음과 같은 스키마 정형모델 M_s을 만들 수 있다.

T= {q_0 , Records, Patient, Medical,
Doctor, Diagnosis, Prescription}
Σ_e = {record, patient, medical, doctor,
diagnosis, prescription}
Σ_a = {@name}
$\delta(q_0, \varepsilon, record)$ = {Records}
δ(Records, [@name]=True, medical) = {Patient}
δ(Patient, [@name]=False, medical) = {q_0}
δ(Medical, ε, doctor) = {Doctor}
δ(Medical, ε, diagnosis) = {Diagnosis}
δ(Medical, ε, prescription) = {Prescription}
N = {Records, Patient, Medical,
Doctor, Diagnosis, Prescription}

그림 3에서 보여진 바와 같이, 그림2의 전자의료카드 XML 문서를 스키마 정형모델로 변환하면, 오토마타와 유사한 정형모델을 구할 수 있다. 단지, 기존의 오토마타와 차이점은 XML 문서의 속성이름을 대수표현으로 전환하여, 속성 값이 참일 경우에만 다음 상태로 전이하도록 구성 하였다는 점이다. 예를 들어, 그림 3의 Patient 상태에서는 name 속성이 True일 경우 medical 입력 알파벳 입력을 통해 다음 상태Medical로 전이할 수 있다.

5.2 프로세스 알제브라 표현

5.2.1 스키마 명세

앞에서 언급한 스키마 정형모델 M_s를 CSP 프로세스 알제브라 문법에 맞게 전환하면, 다음과 같이 표현할 수 있다.

```
datatype NAME = David | ... | N
channel records, patient, medical,
        doctor, diagnosis, prescription

S = record → patient →
    if (NAME ==  or ... or N) then
      (doctor → STOP []
      diagnosis → STOP []
      prescription → STOP)
    else S
```

위의 CSP 코드에서 NAME은 스키마 정형모델에서 기술한 Σ_a 집합에 대한 데이터 타입을 정의하고 있다. channel은 입력 알파벳 집합 Σ_e 을 표현하며, 프로세스간의 통신 채널로 사용되게 된다. 앞으로 모델 M에 대한 CSP 정형모델은 CSP_M 로 기술되며, IF-THEN-ELSE 구문은 일반 프로그래밍 언어와 같이 조건문 표현을 위해 사용되어 진다. 따라서, 앞의 예제에서 S 프로세스에서 record와 patient 이벤트가 차례로 발생한 다음, if 구문에서 NAME 값이 참이 되면 doctor 혹은 diagnosis 혹은 prescription 이벤트가 발생한 후, STOP 상태(즉, 종료상태)로 전이하게 되며, 그렇지 않고 else구문을 만나면 S 프로세스로 머물게 된다.

CSP 코드를 간략히 표현하기 위해, 스키마 정형모델에서 상태집합 T에 대응하는 명시적인 프로세스 표현은 생략하기로 하겠다. 또한 CSP 코드에서 대수표현은 항상 참이라고 가정 하고자 한다.

5.2.2 접근통제정책 명세

본 장에서는 XPath 경로지정 언어로 명세된 접근통제정책을 프로세스 알제브라로 변환하는 방법을 설명하도록 하겠다.

XML 문서 D와 XPath 경로지정 표현 p가 존재할 때,

$p(D)_{node}$ 표현은 문서 D에 있는 XML 트리 노드에 대한 접근통제정책 표현이라고 말할 수 있다. 예를 들어, 그림 1에서 보는 XML 문서에 대해 다음과 같은 $p(D)_{node}$ 표현이 있다고 가정하자:

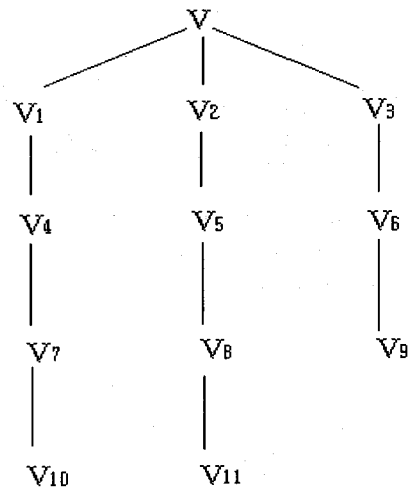

그림 4. V 프로세스 전이 다이어그램
Fig 4. Transition Diagram of V Process

/Records/Patient/{@Name=David}/Medical/Doctor

이런 경우, $p(D)_{node}$ 의 반환값은 Doctor 노드가 된다. 지금부터는 3.2 장에서 언급한 '규칙의 적용 범위'를 CSP 표현으로 바꾸는 방법에 대해 소개하고자 한다. 우선, 간단한 예제를 통해, PL 및 PR 규칙을 CSP 표현으로 바꾸는 과정을 보이도록 하겠다.

Rule 1:
⟨Bob,
/Records/Patient[Name=David]/Medical/doctor,
read, +, PL⟩

Rule 2:
⟨Bob, /Records/Patient[Name=David]/Medical,
read, +, PR⟩

Rule 1과 Rule 2는 각각 PL 과 PR 적용규칙의 예를 보여주고 있고, 이 두 가지 규칙들은 앞장에서 언급한 방법과 유사하게 다음과 같이 쉽게 CSP 코드로 변환할 수 있다.

Rule 1 ⇒ p(D)node ⇒ R1
∴R1 = records → patient → medical
→ doctor → STOP

Rule 2 ⇒ p(D)node ⇒ R1
∴R2 = records → patient → medical
→ (doctor → STOP □
diagnosis → STOP □
doctor → STOP)

NL 및 LR 적용규칙 또한 위와 유사하나 좀더 복잡한 방법을 이용하여, CSP 표현으로 변환 할 수 있다.

Rule 3:
⟨Bob, /Records/Patient[Name=David]/Medical
/Diagnosis, read, -, NL⟩

Rule 4:
⟨Bob, /Records/Patient[Name=David]/Medical,
read, -, NR⟩

Rule 3과 Rule 4는 각각 NL 과 NR 적용규칙의 예를 보여주고 있다.

Rule 3 ⇒ s - p(D)node ⇒ R3
∴R3 = records → patient → medical
→ (doctor → STOP □
prescription → STOP)

Rule 4 ⇒ s - p(D)node ⇒ R4
∴R4 = records → patient → STOP

위의 CSP 코드에서 s는 '스키마'를 나타내며, '-' 기호는 차집합을 표현하기 위해 사용되었다. 따라서, 스키마 s에서 p(D)node에 의해 지정 노드까지 표현된 허용 가능한 경로를 제외한 경로들만이 접근가능하다는 의미를 갖게 된다. 예를 들어, Rule 3에서는 Diagnosis 노드를 제외한 doctor 노드 또는 prescription 노드에 접근가능한 경로를 보여주고 있다.

5.2.3 예제

본 장에서는 본 논문에서 제시한 방법론의 타당성을 증명하기 위해, 그림 1의 XML 문서에 대해 다음과 같은 접근통제정책이 존재할 때, CSP 코드로 명세하는 예제를 보여주고자 한다.

Rule 1:
〈Bob, /Records/Patient[Name=David]/Medical /Diagnosis, read, +, PL〉

Rule 2:
〈Bob, /Records/Patient[Name=David]/Medical /Prescription, read, +, PL〉

Rule 3:
〈Bob, /Records/Patient[Name=David]/Medical /*, read, -, NR〉

규칙 충돌 현상을 해결하기 위해 deny-overwrites 알고리즘이 사용되었다고 가정하며, 아래 예제에서는 ACP_{deny} 프로세스로 표현하였다. 다음은 예제에 대한 CSP 코드를 보여주고 있다.

```
AV = S |[A]| ACPdeny
ACPdeny = (R1 |[A]| R2) |[A]| R3
R1 = records → patient → medical → diagnosis → STOP
R2 = records → patient → medical → prescription → STOP
R3 = records → patient → medical → STOP
```

위 코드에서, S는 그림1의 XML 문서에 대한 스키마 s에 대한 프로세스 표현이며, $A=\alpha(S)\cup\alpha(ACP_{deny})$는 S 프로세스와 ACP_{deny} 프로세스의 이벤트 합집합을 나타내고 있다. 따라서, V는 스키마 s에 대해 deny-overwrites 충돌규칙이 적용되었을 때 프로세스를 가리키며, 그림 4와 같은 프로세스 전이 다이어그램으로 표현된다.

VI. 결론

본 논문에서는 XML 문서에 대한 접근통제정책을 CSP 프로세스 알제브라 변환하는 방법론을 제시하였다. 이를 위해, 첫째, XPath 경로지정 언어로 표현된 XML 문서의 계층적 접근을 CSP 프로세스 알제브라로 변환하는 방법을 소개하였다. 둘째, XML 스키마 문서를 오토마타 형태의 정형모델로 표현하는 방법을 설명하였다. 셋째, XML 접근통제정책에 사용되는 적용 규칙 및 충돌규칙의 의미론을 프로세스 알제브라 언어로 표현하는 방법을 제시하였다. 마지막으로, 본 논문에서 제시한 방법론의 타당성을 보이기 위해, XML 스키마 문서 및 경로지정 표현에 대한 CSP 코드 명세 예제를 보여주었다. 향후 연구방향으로는 CSP 정형명세 예제에 FDR(Failure Divergence Refinement)[9] 모델 체킹 도구를 적용하여, 접근통제정책의 정확성을 증명하는 연구를 진행하고자 한다.

참고문헌

[1] J. Clark and S. DeRose, XML Path language (XPath) Version 1.0, W3C Recommendation, http://www.w3c.org/TR/xpath, 1999.

[2] C. A. R. Hoare, Communicating Sequential Processes, Prentice-Hall, 1985.

[3] P. Ryan, S. Schneider, "Modelling and Analysis of Security Protocols", Addison- Wesley, 2001.

[4] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P.Samarati, "Securing XML Documents", Proceedings of EDBT Conference, pp.121-135, 2000.

[5] A. Gabillion and E. Bruno, "Regulating Access Documents", Proceedings of Working Conference on Database and Application Security, 2001.

[6] M.Murata, A.Tozawa, and M.Kudo, "XML Access Control Using Static Analysis", Proceedings of CCS Conference , pp.73-84, 2002.

[7] I. Fundulaki, M. Marx, "Specifying Access Control Policies for XML Documents with XPath", Proceedings of SACMAT Conference, pp.61-69, 2004.

[8] J. E. Hopcroft, R. Motwani, and J. D. Ullman, Introduction to automata theory, languages, and computation, 2nd ed., Addison-Wesley, 2001

[9] Formal Systems(Europe) Ltd. Failure Divergence Refinement-FDR2 User Manual, 1999.

저 자 소 개

이지연

현재 동남보건대학 e-비즈니스과
조교수

김일곤

한국정보보호진흥원
보안성평가단 평가1팀 선임연구원