# 정책규칙에서 Provision과 Obligation

김 수 희*


## 요 약

XML은 웹 응용을 위한 데이터의 처리와 전송을 위해 가장 일반적인 툴이 되고 있다. 모든 온라인 비즈니스 솔루션에서 정책들은 광범위하게 사용되고 있으며, 접근 요청에 대한 "yes/no"와 같은 이분법적인 결정은 충분하지 않다고 인식되고 있다. 이 논문에서는 provision과 obligation을 지닌 로직 형식으로 표현하는 정책규칙을 보안정책 프로그래머들이 융통성있게 명시할 수 있고 구현할 수 있도록 XML로 변환하여 표현하는 기법을 개발하였다. 사용자 정보, 객체 정보, 액션들을 XML로 표현하기 위한 일반적인 형식을 정의하였으며, 이 세 요소를 기본 요소로 하여 정책들을 명시하기 위한 XML DTD를 개발하였다. 향후에는 권한부여 정책에 기반한 접근 제어뿐만 아니라 데이터의 성격에 따라 각 필드에 있는 데이터를 변환하고 부인방지 기법을 도입하는 등 다양한 보안 기능을 지원하기 위해 이들을 XML 정책 규칙으로 명시하는 것에 대해 연구하고자 한다.


# Provisions and Obligations in Policy Rules

Suhee Kim*


## ABSTRACT

XML is the most common tool for data processing and data transmission in web applications. Policies are extensively used in all online business solutions and it is recognized that abinary decision such as "yes/no" for access requests is not enough. In this paper, a method is developed to convert policy rules with provisions and obligations in logic formula formats into XML formats. The primary purpose is to enable security policy programmers to write flexible authorization policies in XML and to implement them easily. General syntaxes are defined to specify information for users, objects and actions in XML formats and an XML DTD is developed to specify authorization rules with these three components. To support various security features such as data transcoding and non-repudiation depending on data in addition to access control based on authorization policies, studies for specifying them in XML policy rules will be performed in the future


Key words : Provisions, Obligations, Authorization Model, Policy Rules, XML Representation

* 호서대학교 컴퓨터공학부 컴퓨터공학전공

## 1. 서 론

XML은 인터넷상에서 정보의 구조와 내용을 기술하고 데이터를 전송하는 가장 일반적인 툴이 되고 있다[1]. 데이터의 컨테이너로서 XML을 사용하는 것의 장점들로서는 XML의 단순함, 데이터 구조의 풍부함을 들 수 있다. 그러나, 데이터가 비밀정보를 포함할 수 있기 때문에 혹은 부인을 방지하는 것이 필요하기 때문에, 데이터는 가능한 위협으로부터 보호되어야만 하는 경우가 종종 있다.

점점 대중화되고 있는 전자상거래를 원활히 수행하기 위해 모든 온라인 비즈니스 솔루션에서 기밀성, 무결성 그리고 비밀정보에 대한 보안 요구사항을 만족하는 것은 필수적인 요소이다. 웹 문서와 프로토콜에서 XML 기술이 점점 증대되어 사용되어짐에 따라, 권한부여와 접근제어 정책이 XML 솔루션들과 통합되어야 하는 것은 필연적이다.

많은 응용에서 접근 요청에 대한 "yes/no"와 같은 이분법적인 결정은 충분하지 않다고 인식되어져왔다. 어떤 정책들은 접근요청에 대한 권한부여 결정이 이루어지기 전이나 후에 만족되어야할 조건들과 수행되어야 할 액션들을 요구한다. 이러한 이슈들이 [2-4]에서 제안되었고 provision 기반 XML 문서의 접근제어에 대한 모델이 [2]에서 제안되었다. 여기서 provision은 어떤 권한부여 요청에 대한 어떤 결정이 이루어지기 전에 만족될 필요가 있는 조건이나 수행되어야만 하는 액션이고, 반면에 obligation은 그 결정이 이루진 후에 사용자나 시스템에 의해 충족되어야 조건이나 액션이라고 볼 수 있다. 예를 들면, 어떤 대출을 지원하는 지원자는 사전에 구매약관에 동의하여야 지원할 수 있다면, 구매약관에 동의하는 행위는 일종의 provision이라고 볼 수 있다. 또한 지원자가 지원하는 대출을 허락하기 위해서, 대출금을 일정기간 동안 다달이 일정액씩 갚아 나가겠다고 수락하면, 이 행위는 실제 대출을 받고난 후, 반드시 지켜야하는 obligation이다.

이 논문에서는 provision과 obligation을 가진 권한부여 아키텍처를 기술한다. [4]에서 제안한 로직에 기반한 권한부여 정책 모델을 XML로 표현하는 것에 대해 논의하며, 대출에 대한 정책규칙을 예제로 개발하여 이에 대한 접근제어와 권한부여에 대해서 서술한다. 그리고 온라인 대출을 수행하기 위해 필요한 사용자 데이터, 객체들, 액션들을 정의하고 이들을 XML로 표현한다[1].

제2장에서 관련 연구를 살펴보고, 3장에서 provision과 obligation을 지원하는 권한부여 시스템의 아키텍처를 간단히 기술한다. 4장에서 provision과 obligation을 가지는 정책들을 논리형식으로 표현하는 모델을 소개하고, 은행 대출을 위한 간단한 정책 규칙을 이 모델에 기반하여 표현하다. 5장에서 권한부여 요청에 필수적으로 명시되는 사용자, 객체, 액션의 삼요소를 XML로 표현하고 provision과 obligation을 지원하는 정책의 명세에 대한 DTD에 대해 논의한다. 마지막으로 6장에서 결론을 맺는다.

## 2. 관련 연구

초기의 권한부여 모델은 시스템이 접근 요청을 허락하거나 그렇지 않으면 거절하는 것으로 가정하였다. [5-7]에 의한 연구는 sub-subject이 정책을 뒤엎을 수 있고 융통성이 있고 여러 개의 접근 제어 정책을 지원할 수 있는 일반적인 프레임워크를 제공하는 데에 목표를 두고 있다. 이 모든 모델들은 시스템이 접근요청을 허락하거나 그렇지 않으면 거절하는 것으로 가정하였다.

Damiani와 그의 공동 연구자들은 XML 문서와 스키마 정의를 위한 접근 제어 모델을 제안

하였다[8]. 그들은 XML 문서에 접근제어 메커니즘을 제공하며 XML 문서를 읽는 의미론에 무게를 두고 있다.

PolicyMaker[9]와 KeyNote[10]는 보안 정책, 자격, 관계를 명시하고 해석하는 것의 접근법을 통일하였다. KeyNote에 있는 접근 제어 규칙은 발행인으로 구성되어 있는 단언, 신임이 부여되는 사람, 대표되는 조치들의 급들을 그 대표 되는 조치들이 적용되는 조건들과 함께 명시하는 술부로 칭해진다.

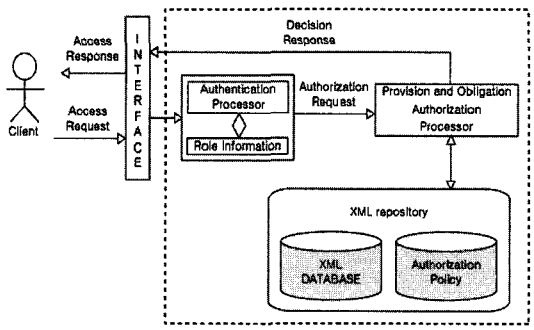
권한부여 모델에서 provision이라는 개념은 Kudo와 Hada[2]에 의해서 소개되었다. 그들은 XML 문서에 대한 권한부여 명세에서 provision이 고려되는 접근제어 시스템을 제안했다. 이 모델은 권한부여 시스템이 잠정적인 조치의 개념을 전통적인 권한부여 의미론과 결합함으로써 좀 더 융통성이 있는 권한부여 결정을 할 수 있게 한다. 이들은 XML 문서를 읽는 기능을 다룰 뿐만 아니라 쓰고, 생성하고 삭제하는 기능을 취급하고, XML로 정의된 접근 제어 명시언어(XACL)를 기술한다. 이 모델에서 provision과 관련한 액션들은 읽기, 쓰기, 생성, 삭제, 암호화, 검증 등의 함수로 표현된다. 이러한 함수들은 사용자에 의한 권한부여 요청에 대한 결정을 기초로 하여 시스템에 의해 실행된다. 이들은 형식화된 잠정적인 권한부여와 규칙들을 제안하였다[2]. 이 도델은 Jajodia 외 2명에 의하여 일반화되었다. [4]에서는 obligation에 중점을 두어 정책부여 모델이 논의되었다.

Provision과 obligation의 개념은 복잡한 응용에서 접근 요청에 대한 잠정적인 결정과 그 후 수행되어야 하는 책무를 모니터링하기 위해서는 반드시 고려되어야 하고 지원되어야 하는 개념들이다. Provision과 obligation의 개념을 도입한 권한부여 모델이 근래에 많은 관심을 받고 있으나, 이를 체계적이면서 포괄적으로 구현한 시스템은 거의 찾아보기 힘들다.

## 3. 시스템 아키텍처

(그림 1)은 provision과 obligation을 지원하는 권한부여 시스템의 아키텍처이다. 이 권한 부여 시스템은 사용자가 시스템을 연결할 수 있는 어떤 인터페이스가 있음을 전제한다. 그 인터페이스는 사용자의 신원(identity)과 역할(role)을 확인하는 모듈을 호출한다.

일단 사용자가 인증이 되고 현재의 역할이 확인되면, 사용자는 권한부여 요청을 할 수 있다. 권한부여 요청은 provision-obligation authorization processor(권한부여 처리기, POAP)로 전달된다. 각 권한부여 요청의 형식은 어떤 사용자 $s$(subject)가 어떤 객체 $o$를 대상으로 어떤 액션 $a$를 요청하는 형식이다. POAP는 권한부여 정책 규칙들, 객체들의 정보와 사용자들의 정보 등을 이용하여 권한부여 요청에 대한 권한부여에 대한 결정을 내리고 그 결과를 요청한 사용자에게 보낸다.




(그림 1) 시스템 아키텍처

## 4. Provision과 Obligation을 가진 정책 규칙

접근제어와 권한부여 시스템 분야에서 대부분의 연구는 사용자의 요청의 결과가 "yes/no" 만

으로 결정되는 정적인 정책들을 다룬다. 여기에서는 [4]에 제안된 정책규칙에 대한 형식적인 모델을 이용하여 provision과 obligation을 취급하는 정책 규칙의 형식을 간단히 소개하고 그 다음으로 은행 대출을 위한 정책의 예를 들어 본다.

## 4.1 정책규칙 모델

이 절에서 [4]에서 제안한 provision과 obligation을 지닌 융통성이 있는 권한부여 정책모델을 소개하고, provision과 obligation을 지닌 대출 응용에 대한 권한 부여 정책 규칙의 예를 든다.

직관적으로, 하나의 정책은 기본적인 규칙들의 집합으로 표현될 수 있다. 각 규칙은 그 자신의 provision과 obligation을 가질 수 있다. 기호 $V$, $C$와 $Q$는각각 변수, 상수, 술부들의 유한 집합이다. 권한 부여 정책에서 사용하는 원자들(atoms)와 규칙들(rules)은 다음과 같이 정의된다.

- Atom : 공식 $Q(t_1, \cdots, t_n)$, 여기서 $Q$는 술부이고 각 $t_i$는 상수이거나 변수이다.
- Rule : 공식 $A \leftarrow B_1, \cdots, B_m$,
  $A, B_1, \cdots, B_m$ : atoms,
  $A$ : head, $B_1, \cdots, B_m$ : body

심볼 $P$와 $O$는 각각 provision과 obligation들의 집합으로 중첩되지 않는다. 이들 $P$와 $O$는 역시 술부 $Q$의 집합과도 중첩되지 않는다. 이와 반대로 $P$와 $O$에 사용되는 변수 심볼 $V$와 상수 심볼 $C$의 집합은 정책 규칙에 사용될 수 있다.

- $P$ : sets of provisions
- $O$ : sets of obligations
- Provisions과 obligations에서 atom :
  술부 $Pi$ $(t_1, \cdots, t_k)$, $P_i \in P$ 혹은
  $O_i$ $(t_1, \cdots, t_k)$, : $O_i \in O$,
  각 $t_i$는 : 상수이거나 변수이다.
- $PO$-atoms : provisions과 obligations에서 atom들
- $PO$-formula : $PO$-atom, 혹은
  $PO$-formulas의 논리합
  혹은 논리곱

$PO$-formula에 대한 해석은 주어진 원자식들에 있는 변수들을 모두 상수 값으로 대입하여 참과 거짓으로 사상하는 것이다. <표 1>은 provision과 obligation이 있는 정책 규칙들의 예를 나타내고 있다.

하나의 정책에 대한 권한 부여 요청을 위한 결정은 바디에 있는 모든 atom들이 참으로 계산되고 provision에 대한 집합을 만족하고 요구하는 obligation을 수락할 때 참으로 계산이 된다. 즉, 요청에 대한 권한 부여를 허락한다는 것이다. 권한 요청을 한 사용자가 수락한 obligation은 모니터링 되어야 하고 그것의 이행과 불이행에 대해 적당한 액션을 취해야 한다. Obligation의 모니터링에 대한 정의 및 이의 실행은 향후에 논의하기로 한다.

〈표 1〉 Provision과 Obligation이 있는 정책

| 정 책 규 칙 | PO-formula |
|---|---|
| (R1) $Q_1(x) \leftarrow Q_2(x,y), Q_3(y)$ | $O_1(s,x,y)$ |
| (R2) $Q_2(a, b) \leftarrow$ | $P_1(b)$ |
| (R3) $Q_3(b) \leftarrow$ | |
| (R4) $Q_1(y) \leftarrow Q_4(z,y,c)$ | $P_2(y,a) \wedge P_3(a) \wedge O_2(y,c)$ |
| (R5) $Q_4(c,a,c) \leftarrow$ | |

## 4.2 대출응용의 정책 규칙

이 절에서는 provision과 obligation을 지닌 대출 응용에 대한 권한 부여 정책 규칙의 예를 들고, 이를 XML로 표현한다.

<표 1>에서 보는 바와 같이 각 정책은

head ← body, provision list, obligation list

형식으로 표현되는데, head는 논리의 atom 형

태이며, body, provision list, obligation list는 atom list의 형태를 취한다. 논리에서 atom은 $Q(t_1, \cdots, t_n)$의 형식인데, 여기서 $Q$는 술부이고 각 $t_i$는 상수 혹은 변수이다. 그리고 head는 항상 access(subject, object, action)의 형태이다. 그러므로 어떤 subject이 어떤 object를 대상으로 어떤 action을 수행하는 형식으로 권한 요청을 할 수 있다.

<표 2>에 있는 정책 규칙 R1, R2, R3는 다음과 같이 해석할 수 있다.

- 정책 규칙 R1
  Head : access(user, loan, read)
  Body : 없음
  Provision : registered(user)
  Obligation : 없음

이 정책은 body와 obligation이 없는 경우이다. 이 정책 규칙의 의미는 사용자가 대출에 대한 정보를 읽기 위해서는 먼저 등록을 해야 한다는 것이다. 즉 등록한 사용자만이 대출에 대한 정보를 읽을 수 있다는 정책이다.

〈표 2〉 대출 정책의 예

| 정책 규칙 | Provision과 obligation |
|---|---|
| R1 : access(user, loan, read) ← | registered(user) |
| R2 : access(user, loan, apply) ← read(user, loan) | signPurchaseAgreement (user) |
| R3 : access(user,easyLoan, selfAppove) ← application(easyLoan, user), (amount <= 1,000,000), computePay(user, monthlyPayment, 12), (credit >= 3.0) | payLoan(user, easyLoan, monthlyPayment, 12) |
| R4 : access(user,speedLoan, selfAppove) ← application(speedLoan, user), (amount <= 10,000,000), computePay(user, monthlyPayment, 36), (credit >= 6.0) | payLoan(user, speedLoan, monthlyPayment, 36) |

- 정책 규칙 R2
  Head : access(user, loan, apply)
  Body : read(user, loan)
  Provision : signPurchaseAgreement(user)
  Obligation : 없음

정책 규칙 R2는 어떤 대출을 지원하기 위해서는 다음과 같은 조건들을 만족해야 한다.

- 요청자가 미리 특정 대출에 대한 문서를 읽어야 한다. 즉 read(user, loan)을 요청자에게 적용하였을 때 'true'로 계산되어야 한다.
- 요청자는 구매 약관에 동의한다.

이 경우에 대출 문서를 읽지 않았거나 구매 약관에 사인을 하지 않으면 그 대출을 지원하는 행위가 허락되지 않는다.

- 정책 규칙 R3
  Head : access(user, easyLoan, selfAppove)
  Body : application(easyLoan, user),
  (amount <= 1,000,000),
  computePay(user,
  monthlyPayment, 12),
  (credit >= 3.0)
  Provision : 없음
  Obligation : payLoan(user, easyLoan,
  monthlyPayment, 12)

이 정책은 사용자기 요청한 easyLoan이라는 대출에 대해서 시스템이 자동으로 승인하기 위한 것이다. 사용자기 요청한 easyLoan이라는 대출을 승인하기 위해서는 다음 조건을 만족해야 한다.

- 먼저 그 사용자가 easyLoan 대출을 지원하였고, 그 지원서에 원하는 대출 액수가 명시되며, 그 액수는 100만원 이하이어야 한다.
- 이를 12개월로 나누어 매월 지불해야 하는 액수를 계산하고 신청자의 신용도가 3.0 이

상이어야 한다.

- 사용자는 바디에서 계산된 월지급금을 12개월동안 지불하겠다는 obligation을 수락하여야 한다.

즉 바디에 있는 모든 술부들이 'true'로 계산되고, 명시한 obligation을 수락하면 요청한 대출이 허락된다. 이 경우에 obligation을 특별히 모니터링하는 과정이 필요하다.

이러한 정책들을 구현하기 위해서 provision과 obligation, 그리고 각 정책 규칙의 바디에 있는 atom들을 함수 형태로 구현할 수 있다.

## 5. 정책규칙을 XML로 표현

Provision과 obligation을 가진 정책규칙을 XML로 표현하는 주된 목적은 보안 정책 프로그래머들이 XML로 융통성이 있는 권한 부여 정책을 명시할 수 있도록 하고, 또한 쉽게 이 규칙들을 구현할 수 있다는 점이다.

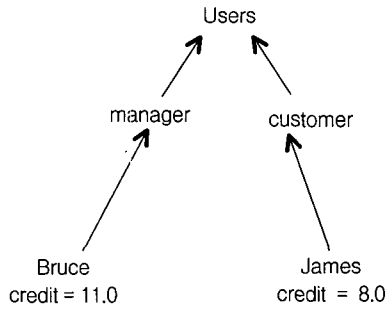
권한 부여 정책 명시는 두 가지 방법으로 접근할 수 있다. 한 방법은 스키마(DTD)를 정의하여 사용하는 것이고 또 다른 방법은 특정 응용에 따라 명시하는 것이다.

### 5.1 사용자 정보

사용자는 기본적으로 사용자 id, 역할 정보, 소속한 그룹 정보가 명시될 수 있고, 4장에서 소개한 대출 응용을 위해서는 신용정보가 필요하다. 역할과 그룹은 계층적 구조를 가지고 있다고 볼 수 있다. 한 사용자가 어떤 그룹에 속해 있다면, 그 사용자는 그 그룹의 모든 수퍼 그룹의 멤버이다. 그러므로 한 그룹과 어떤 역할의 모든 권한부여는 각각 그것의 서브 그룹과 서브 역할로 전파된다.

```
<!ELEMENT subject (uid?, role*, group*,
                   credit?)>
<!ELEMENT uid (#PCDATA)>
<!ELEMENT role (#PCDATA)>
<!ELEMENT group (#PCDATA)>
<!ELEMENT credit (#PCDATA)>
```

(그림 1) 사용자 정보 형식




(그림 2) 사용자 계층

(그림 3)에 있는 사용자 James의 정보를 XML로 표현하면 다음과 같다.

```
<subject>
    <uid>James </uid>
    <group> Customer</group>
    <credit> 8.0 </credit>
</subject>
```

### 5.2 객체 정보

대출 응용의 예에서 대상이 되는 객체들은 주로 대출 관련한 문서들이다. 대출과 관련한 문서, 특정 대출에 대한 지원 폼 등은 모두 XML 형식으로 구성된 문서로 가정하자. 이 문서들은 여러 엘리먼트들로 구성되어 있으며, 각 엘리먼트가 역시 하나의 객체가 될 수 있다.

이를 XPath 형식으로 표현하여 각 엘리먼트를 식별할 수 있다. 이를 'href' 속성을 사용하여 명시한다.

```
<!ELEMENT object EMPTY>
<!ATTLISTobject href CDATA #REQUIRED>
```

### 5.3 액션

액션은 <action> 엘리먼트로 명시된다. 'name' 속성은 액션의 이름을 명시하기 위해 사용된다. 액션에는 read, write, apply, selfApprove 등이 있다.

```
<!ELEMENT action EMPTY>
<!ATTLIST ACTION name
          (read|write|apply|selfApprove)
           #REQUIRED>
```

### 5.4 정책 규칙

제4장에서 기술한 접근 정책은 많은 규칙들로 이루어져있다. 각 규칙은 head, body, provision list, obligation list로 구성된다.

rule : (head, body, provision list, obligation list)

정책에 대한 명세를 위한 XML의 DTD는 (그림 4)와 같다. 이 DTD는 rule(head, body, provision list, obligation list) 형식으로 명시되는 모든 규칙들을 명시할 수 있다. 여기서 provision list와 obligation list 요소가 없으면 그 규칙은 전통적인 정책 규칙과 그 형식이 동일함을 알 수 있다.

```
<!ELEMENT <!ELEMENT POLICY (RULES+)>
<!ELEMEN <!ELEMENT RULE (HEAD, BODY?,
          PROVISIONS, OBLIGATIONS)>
<!ELEMENT HEAD (OBJECT, SUBJECT,
          ACTION)>
<!ELEMENT OBJECT EMPTY>
<!ELEMENT SUBJECT (UID?, GROUP*,
          ROLES*, CREDIT? )>
<!ELEMENT UID (#PCDATA)>
<!ELEMENT GROUP (#PCDATA)>
<!ELEMENT ROLE (#PCDATA)>
<!ELEMENT ACTION EMPTY>
<!ELEMENT BODY (ATOM)*>
<!ELEMENT PROVISIONS (PROVISION)*>
<!ELEMENT OBLIGATIONS (OBLIGATION)*>
<!ELEMENT ATOM (PARAMETER|ATOM)*>
<!ELEMENT PARAMETER (#PCDATA)>
<!ELEMENT PROVISION (ATOM)*>
<!ELEMENT OBLIGATION (ATOM*, FULFILL?,
          DEFAULT?)>
<!ELEMENT FULFILL (ATOM)*>
<!ELEMENT DEFAULT (ATOM)*>
<!ATTLIST OBJECT href CDATA #REQUIRED>
<!ATTLIST ACTION name
   (read|write|apply|selfApprove) #REQUIRED>
<!ATTLIST RULE name CDATA #REQUIRED>
<!ATTLIST ATOM name CDATA #REQUIRED>
<!ATTLIST ATOM value CDATA #IMPLIED>
<!ATTLIST PARAMETER name CDATA
          #REQUIRED>
```

(그림 4) 정책 명세를 위한 XML DTD

(그림 5)는 <표 2>에 있는 정책들 중에서 정책 R3를 XML로 표현한 것이다.

```
<RULE name="R3">
<HEAD>
<ATOM name="access">
<PARAMETER value="user"/>
<PARAMETER value="easyLoan"/>
<PARAMETER value="selfApprove"/>
</ATOM></HEAD>
<BODY>
<ATOM name="application">
<PARAMETER value="easyLoanr"/>
<PARAMETER value="user"/>
</ATOM>
<ATOM name="calculate">
<PARAMETER value="integer"/>
<PARAMETER value="amount"/>
<PARAMETER value="<="/>
<PARAMETER value="1000000"/>
</ATOM>
<ATOM name="computePay">
<PARAMETER value="user"/>
<PARAMETER value="monthlyPayment"/>
<PARAMETER value="12"/>
</ATOM>
```

```
<ATOM name="computePay">
<PARAMETER value="user"/>
<PARAMETER value="monthlyPayment"/>
<PARAMETER value="12"/>
</ATOM>
<ATOM name="calculate">
<PARAMETER value="real"/>
<PARAMETER value="credit"/>
<PARAMETER value=">="/>
<PARAMETER value="3.0"/>
</ATOM></BODY>
<PROVISIONS><PROVISION>
<ATOM name="log">
<PARAMETER value="user"/>
</ATOM>
</PROVISION></PROVISIONS>
<OBLIGATIONS><OBLIGATION>
<ATOM name="payLoan">
<PARAMETER value="user"/>
<PARAMETER value="easyLoan"/>
<PARAMETER value="monthlyPayment"/>
<PARAMETER value="12"/>
</ATOM>
</OBLIGATION></OBLIGATIONS>
</RULE>
```

(그림 5) 대출 정책 R3을 XML로 표현

## 6. 결 론

이 논문에서는 provision과 obligation을 취급하는 정책 규칙의 필요성을 논의하고 [3]에서 제안한provision과 obligation을 지원하는 로직의 형식으로 정책을 명시하는 모델을 소개하였고, 이러한 정책들을 XML로 변환하여 표현하기 위해 정책 명세를 위한 DTD를 개발하였다. Provision과 obligation을 가진 정책규칙을 XML로 표현하는 주된 목적은 보안 정책 프로그래머들이 XML로 융통성이 있는 권한 부여 정책을 명시할 수 있도록 하고, 또한 쉽게 이 규칙들을 구현할 수 있다는 점이다.

대출 응용 분야를 이용해 provision과 obligation을 가진 정책의 예를 들고 사용자들에 대한 표현 형식, 객체에 대한 표현 형식, 액션에 대한 표현 형식을 논의하였으며, 이들과 대출 정책에 대한 것을 XML로 표현하였다.

이 연구에서 제안한 XML을 이용한 정책 명세 모델을 이용하여 대출과 관련한 문서에 대한 처리, 대출 요청, 승인 여부에 대한 결정을 정의하고 현재 프로토타입 시스템을 구현 중에 있으며, 이를 통하여 경험하는 구현상의 문제점들을 바탕으로 하여 실제 정책을 정의하는 데 있어서 수정 및 보완을 하고자 한다. 향후에는 사용자 정보, 대출 정보에 대해 특정 필드에 대한 접근 제어, 암호화, 부인방지 등에 대한 기능을 추가하고자 한다. 다시 말하면, 정책에 기반한 접근제어 뿐만 아니라 데이터의 성격에 따라 기밀을 유지하기 위해 각 필드에 있는 데이터를 변환하여 통신하고 부인방지 기법을 도입하는 등 다양한 보안 기능을 가진 시스템으로 확장하고자 한다.

## 참 고 문 헌


[1] World Wide Web Consortium, XML http://w3c.org/XML.

[2] M. Kudo and S. Hada, "XML document security based on provisional authorization", In *Proc. of the 7th ACM conference on Computer and communications security*, pp. 87-96, 2000.

[3] Sushil Jajodia, MichiharuKudo and V. S. Subrahmanian, Provisional authorizations. : In Anup Gosh (ed.), *E-Commerce Security and Privacy, Kluwer Academic Press*, pp.133-159, 2001.

[4] Claudio Bettini, Sushil Jajodia, X. Sean Wang and Duminda Wijesekera, "Provisions and Obligations in Policy Rule Management", *Journal of Network and Systems Management*, Vol.11, No.3, pp.351-

372, 2003.

[5] T. Y. C. Woo and S. S. Lam, "A Framework for Distributed Authorization", 1st *ACM Conference on Computer and Communications Security*, November 1993.

[6] S. Jajodia, P. Samarati, and V. S. Subrahmanian, "A Logical Language for Expressing Authorizations", Proc. *1997 IEEE Symposium on Security and Privacy*, pp.31-42, May 1997.

[7] S. Jajodia, P. Samarati and V. S. Subrahmanian, and E. Bertino, "A Unified Framework for Enforcing Multiple Access Control Policies", Proc. *ACM SIGMOD International Conference on Management of Data*, pp.474-485, May 1997.

[8] E. Damiani, S. D. C Vimercati, S. Paraboschi and P. Samarati, "Securing XML Documents", Proc. *EDBT 2000, Lecture Notes in Computer Science*, Vol.1777, pp.121-135, 2000.

[9] M. Blaze, J. Feigenbaum and J. Lacy, "Decentralized Trust management", Proc. *1996 IEEE Symposium on Security and Privacy*, pp.164-173, May 1996.

[10] M. Blaze, J. Feigenbaum and A. Keromytis, "KeyNote : Trust Management for Public-Key Infrastructures", Proc. *1998 Cambridge Security Protocols International Workshop*, LNCS, Vol.1550, pp.59-63, 1999.

**김 수 희**

1986년 University of Georgia 전산학과(MS)

1988년 University of Georgia 수학과(MA)

1993년 University of South Carolina 전산학과(Ph.D.)

1993년~1994년 Benedict College 조교수

1994년~현재 호서대학교 컴퓨터공학부 컴퓨터공학전공 교수