객체지향 지리정보시스템에서의 역할 기반 접근 제어

김 미 연* · 이 철 민** · 문 창 주*** · 이 동 훈****

Role-Based Access Control in Object-Oriented GIS

Kim, Mi-Yeon* · Lee, Cheol-Min** · Moon, Chang-Joo*** · Lee, Dong Hoon****

Abstract

Role-based access control (RBAC) models are recently receiving considerable attention as a generalized approach to access control. In line with the increase in applications that deal with spatial data, an advanced RBAC model whose entities and constraints depend on the characteristics of spatial data is required. Even if some approaches have been proposed for geographic information systems, most studies focus on the location of users instead of the characteristics of spatial data.

In this paper, we extend the traditional RBAC model in order to deal with the characteristics of spatial data and propose new spatial constraints. We use the object-oriented modeling based on open GIS consortium geometric model to formalize spatial objects and spatial relations such as hierarchy relation and topology relation. As a result of the formalization for spatial relations, we present spatial constraints classified according to the characteristics of each relation. We demonstrate our extended-RBAC model called OOGIS-RBAC and spatial constraints through case studies. Finally, we compare our OOGIS-RBAC model and the DAC model in the management of access control to prove the efficiency of our model.

Keywords : RBAC, GIS, Object-Oriented Modeling

논문접수일 : 2007년 05월 20일 논문게재확정일 : 2007년 08월 31일
※ 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음(IITA-2006-(C1090-0603-0025).
* 주저자, KT 인프라연구소 책임연구원, 고려대학교 정보경영공학전문대학원 박사수료
** KT, 인프라연구소 책임연구원
*** 교신저자, 건국대학교 항공우주정보시스템공학과 조교수, (143-701) 서울시 광진구 화양동 1번지, e-mail : cjmoon@konkuk.ac.kr
**** 고려대학교 정보경영공학전문대학원 교수

1. 서 론

국토 관리, 자원 관리, 도로, 전력, 통신 분야에서는 시설물 관리를 위해서 지리정보시스템(Geographic Information System: GIS)이 필수적으로 요구된다. GIS는 다른 정보시스템과 달리 시각적 판단근거를 제공하는 공간데이터에 대한 처리 기능을 제공한다. 그러므로, GIS에서 처리하는 국가나 기업의 주요 시설물에 대한 정보보안을 위해서는 보다 정밀한 접근제어가 필요하다.

지금까지 공간데이터에 대한 접근제어는 임의적 접근제어(Discretionary Access Control: DAC) 모델에 기반하여 연구되어 왔다[Belussi et al., 2004; Atluri and Chun, 2004]. 그러나, DAC 모델을 GIS에 적용하기에는 여러 가지 문제점이 존재한다[Ferraiolo and Barkley, 1999; Ferraiolo et al., 2003]. DAC 모델은 일반적으로 접근제어리스트(Access Control List : ACL)에 의해 구현되며, ACL은 특정 객체에 대한 접근권한을 시스템 최종 사용자에게 직접 부여한다. 이와 같은 구현 방식으로 인해 다음과 같은 문제점이 존재한다. 첫째, 최종 사용자는 시스템 관리자의 중재 없이 임의의 사용자에게 자신의 권한을 양도할 수가 있다. GIS에서 관리되는 지리정보 또는 시설정보는 최종 사용자의 소유이기보다는 국가나 기업 조직의 소유로 관리되어야 한다. 둘째, 관리되어야 할 시스템 자원의 증가에 비례하여 ACL의 크기가 증가하게 된다. GIS에서는 관리하여야 할 정보의 양이 수 테라바이트(Terabyte) 혹은 수 페타 바이트(Petabyte)에 이르며 이를 ACL로 관리하기 위해서는 매우 많은 관리 시간이 소요된다. 셋째, 관리하는 모든 시스템 단위로 사용자 권한을 관리해야 한다. GIS는 방대한 양의 정보를 처리하기 위해 분산환경으로 구축되어야 한다. 그러므로, 모든 시스템 별로 접근제어를 관리하기 위한 관리 비용 또한 증가하게 된다. 결과적으로 DAC은 GIS의 접근제어 방식으로는 부적합하다.

이와 같은 DAC 모델의 문제점으로 인해 GIS의 접근제어 방식으로 역할기반 접근제어(Role Based Access Control : RBAC) 모델을 도입하는 것이 필요하다. RBAC은 접근제어 관리의 효율성을 위해 사용자와 자원의 접근 권한을 직접 연관시키지 않고, 사용자에게 부여된 역할을 통해 자원에 접근하도록 하는 방식이다. 현재 RABC은 DAC을 대체하기 위한 방식으로 주목받고 있다[Sylvia Osborn et al., 2000]. 그러나, GIS에 RBAC을 적용하는 연구는 대부분 사용자의 위치에 기반한 연구가 주로 이루어지고 있으며 GIS에서 관리되는 공간데이터의 특성에 기반한 연구는 거의 이루어지지 않고 있는 실정이다[Hansen and Oleshchuk, 2003; Elisa Bertino et al., 2005; Claudio A. Ardagna, 2006].

본 논문에서는 GIS에서 공간데이터의 특성에 기반한 RBAC을 적용하기 위해, NIST(National Institute of Standards and Technology)에서 제정한 RBAC의 표준[Ferraiolo et al., 2001]을 확장한 모델을 제시하고자 한다. 일반적으로 GIS의 공간데이터를 이용한 다양한 응용과 접근제어를 구현하기 위해서, GIS는 객체지향 데이터 모델링 기법을 적용하여 구현된다. 그러므로, 본 논문에서는 객체지향 GIS에 적용하기 위한 확장된 RBAC을 제시하며 이를 OOGIS-RBAC(Object-Oriented GIS-RBAC) 이라고 명명한다.

본 논문은 다음과 같이 구성된다. 제 2장에서는 본 논문에서 제시하는 OOGIS-RBAC 모델의 기본 기술인 GIS, 객체지향 모델링 및 RBAC에 대한 기본 개념을 설명한다. 제 3장에서는 OOGIS-RBAC을 모델링하기 위해 필수적으로 요구되는 객체지향 GIS의 구성요소를 정형화한다. 객

체지향 GIS는 제 2장에서 설명된 객체지향 모델링 기법의 기본 요소를 이용하여 정의되며 이는 현재 표준으로 제정되어 있는 OGC(Open GIS Consortium)의 기하학 모델 [OGC, 1999]과 호환된다. 제 4장과 제 5장은 본 논문의 핵심이 되는 부분으로 OOGIS-RBAC 모델의 기본요소와 OOGIS-RBAC 모델에서 요구되는 제한사항(Constraint)을 제시한다. 제 6장에서는 OOGIS-RBAC 모델의 관리의 효율성을 증명하기 위한 간단한 사례를 제시하고, OOGIS-RBAC 모델과 DAC 모델을 연관관계 관리 및 저장공간 사용의 효율성 측면에서 비교한다. 제 7장에서 결론을 맺는다.

2. 선행연구

본 논문에서 제시하는 OOGIS-RBAC 모델의 선행 기술의 이해를 위해 본 장에서는 GIS, 객체 지향 모델링 및 RBAC 모델에 대한 기본 개념을 간단히 설명한다.

2.1 지리정보시스템(Geographic Information System : GIS)

GIS는 지리적 위치에 존재하는 다양한 지리적 객체를 관리하는 시스템으로써 공간데이터와 비공간데이터로 구성된다[김계현, 1998]. 공간데이터는 지리적 객체의 모양과 위치를 표현하며 시각적인 판단근거를 제공한다. 비공간데이터는 시각적인 형태는 갖지 않으나 지리적 객체와 연관되는 다양한 관련 정보를 표현한다. 비공간데이터는 텍스트 형태로 간단히 표현되지만, 공간데이터는 지리정보시스템에 따라 표현방식이 달라진다. 공간데이터를 표현하는 방식은 크게 레스터 형식 또는 백터 형식으로 분류한다[F. PIRES et al., 1993]. 레스터 형식은 주로 인공위성에 의해 촬영된 영상의 이미지를 셀 단위로 분할하는 방식이다. 백터 방식은 지리적 객체를 점, 선, 면으로 구성된 집합으로 간주하고 2차원의 공간좌표 혹은 3차원의 공간좌표를 이용한다. 일반적으로 벡터 방식의 지리정보시스템은 객체지향 기법에 의해 모델링된다 [Egenhofer and Frank, 1992; OGC, 1999; Voisard and David, 2002].

레스터 방식은 구현하기가 용이하다. 그러나, 지리적 객체에 다양한 응용과 접근제어를 구현하기 위해서는 벡터 방식으로 지리정보시스템이 구현되어야 한다[Belussi et al., 2004]. 그러므로, 본 논문에서는 객체지향 모델링과 벡터 방식을 이용하여 구현된 GIS 환경에 적용하기 위한 접근제어 기법을 기술하고자 한다.

OGC 모델은 상용 GIS에서 가장 널리 적용된 GIS 모델링 표준이다[Bertino et al., 2005]. OGC 모델은 벡터 형식으로 표현된 공간데이터를 객체지향 모델링 기법으로 구현하였으며, 본 논문에서는 OGC 기본 모델과 호환이 되도록 객체지향 GIS를 정형화한다.

2.2 객체 지향 모델링(Object-Oriented Modeling : OOM)

객체지향 시스템에서 모든 데이터는 객체로써 모델링된다. 모든 객체는 상태 값을 표현하는 속성들과 속성들을 처리하기 위한 메소드들로 구성된다[Banerjee et al., 1987]. 객체는 네 가지 주요한 개념 - 클래스화(Classification), 일반화(Generalization), 연관화(Association), 집단화(Aggregation) - 에 의해 구현된다[Egenhofer and Frank, 1992]. 다음 <그림 1>은 객체지향 모델링을 위한 네 가지 핵심 개념에 대한 각각의 사례를 UML(Unified Modeling Language)을 이용하여 보여준다.

클래스화는 다양한 객체들을 공통적인 특성으로 분류하여 클래스로 모델링하는 개념이다. 클래스는 객체의 공통적인 속성과 메소드로 구성된다. 모든 객체는 최소한 하나의 대응되는 클래스가 존재한다. 다음 <그림 2>의 (a)는 빌딩 클래스 예를 보여준다. 가장 첫 번째 행은 클래스의 이름을 나타내며 다음 행은 속성 이름이며 마지막 행은 메소드 이름을 포함한다.

일반화는 공통적인 특성의 클래스를 그룹화하여 좀더 일반적인 개념의 상위 클래스를 모델링하는 것이다. 다음 <그림 1>의 (b)는 '아파트', '단독주택' 클래스는 '주택' 클래스로 일반화되고 '주택', '빌딩' 클래스는 '건물' 클래스로 일반화되는 예를 보여준다. 기호 ◁—는 일반화를 나타낸다. 클래스는 표기를 단순히 하기 위해서 속성과 메소드는 생략하고 클래스 이름만 나타내었다.

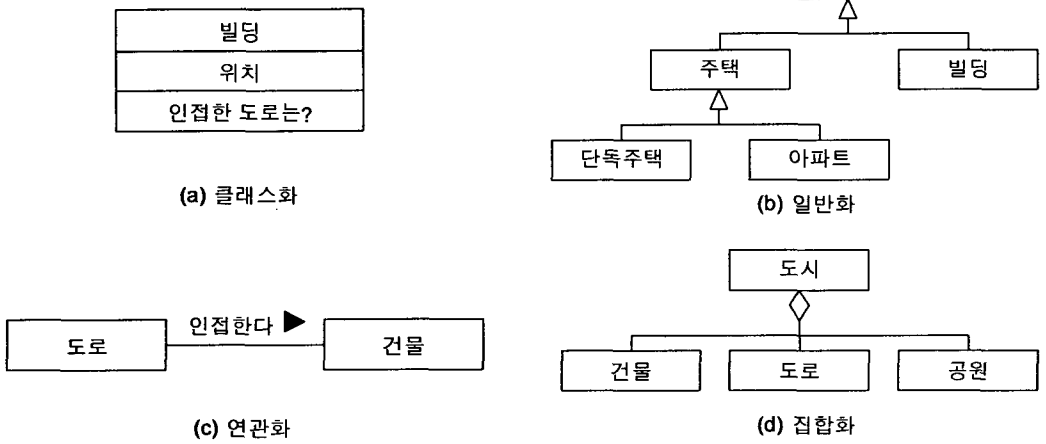
연관화는 독립적인 두 클래스 사이의 관계를 모델링한다. 다음 <그림 1>의 (c)는 '도로' 클래스가 '건물' 클래스에 인접한 관계를 갖는 연관화의 예를 보여준다. 기호 ▶는 연관화를 나타내며 관련된 연관관계를 표기한다.

집합화는 다른 클래스들로 구성되는 하나의 합성클래스를 모델링한다. 다음 <그림 1>의 (d)는 '도시' 클래스가 '건물', '도로', '공원' 클래스들로 구성되는 집합화의 예를 보여준다. 기호 ◇— 는 집합화를 나타낸다.

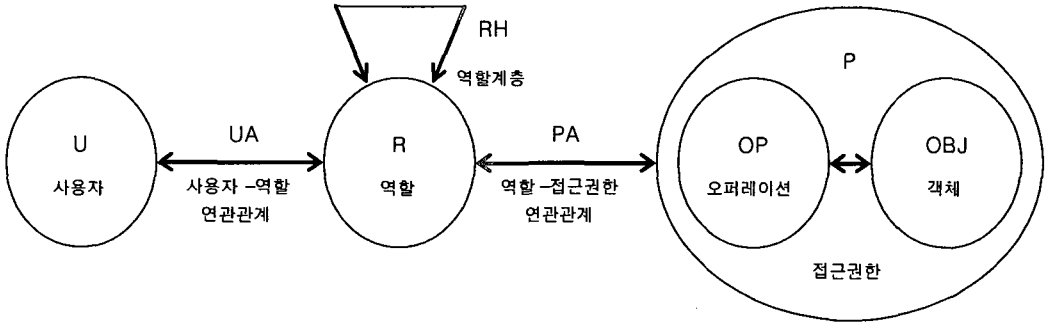
2.3 RBAC 모델

RBAC에 대한 기본 프레임워크는[Sandhu et al., 1996]에 의해 정의되었으며, 이후[Ferraiolo et al., 1999; Ahn and Sandhu, 2000]에 의해 확정되어왔다. 그리고 2001년 NIST에 의해 RBAC 모델이 표준화되었다.

NIST 표준의 핵심 RBAC은 RBAC을 구현하기 위한 최소한의 요소로 구성되며 계층 RBAC에는 핵심 RBAC의 기본요소 외에 역할의 계층을 지원하기 위한 요구사항이 추가되었다. 핵심 RBAC은 사용자(Users : U), 역할 (Roles : R), 객체(Objects : OBJ), 오퍼레이션(Operations : OP), 접근권한(Permissions : P)으로 구성된다. 핵심 RBAC의 요소간의 관계는 사용자와 역할간의 연관관계 (User Assignment : UA)와 역할과 접근권한



〈그림 1〉 객체지향 모델링에 대한 사례



〈그림 2〉 NIST 표준의 핵심 RBAC과 계층 RBAC

과의 관계(Permission Assignment : PA)가 존재한다. 다음 <그림 2>는 NIST 표준의 핵심 RBAC과 계층 RBAC을 나타낸다.

3. 객체지향 지리정보시스템의 정형화

OOGIS-RBAC 모델을 제시하기 위해서는 GIS에서 관리되는 공간데이터에 대한 정형화가 필요하다. 본 장에서는 OGC 표준과 호환이 되도록 클래스화와 일반화 개념을 적용하여 GIS의 공간데이터를 정형화한다. OGC 표준 모델에서 루트 클래스는 'GEOMETY'로 명명되며, 루트 클래스에는 루트 클래스의 식별자 속성과 공간데이터에 대한 공통적인 위상(Topology) 관계를 포함하는 공간 메소드가 포함된다. 이에 기반하여 정의 1에서는 'GEOMETRY'라는 공간 루트 클래스를 정의하였다.

정의 1 (공간 루트 클래스). 공간 루트 클래스는 rs_class로 표기하며, rs_class = (id, SMTHDS) ∈ RS_CLASS = {GEOMETRY}이다.

(i) 'id'는 공간 루트 클래스에 대한 식별자이다.

(ii) 'SMTHDS'는 공간 객체에 대한 공통적인 위상관계에 대한 공간 메소드의 집합으로 정의 7에서 정의된다.

공간 루트 클래스는 추상클래스로 정의된다. 그러므로, 공간 루트 클래스의 인스턴스에는 객체는 존재하지 않고 공간 루트 클래스에 대한 하위 클래스만이 존재한다. 루트 클래스의 직속 하위클래스에는 공간 데이터 타입 클래스가 존재한다. 공간 데이터 타입 클래스는 점, 선, 면, 공간클래스집합으로 구성되며 OGC에서는 각각 'POINT', 'LINESTRING', 'POLYGON' 그리고, 'GEOMCOLLECTION'으로 명명된다. 다음 정의 2에서는 OGC 표준의 타입 레벨 1에 기반하여 공간 데이터 타입 클래스를 정의한다.

정의 2 (공간 데이터타입 클래스). 공간 데이터 타입 클래스는 sdt_class로 표기하며, sdt_class = (id, supclass, attrs, mthds) ∈ SDT_CLASS = {POINT, LINESTRING, POLYGON, GEOMCOLLECTION}이다.

(i) 'id'는 공간 데이터 타입 클래스의 식별자이다.

(ii) 'supclass'는 공간 데이터타입 클래스의 상위 클래스로서 supclass ∈ {RS_CLASS}이다. 공간 데이터타입 클래스는 상위 클래스의 속성과 메소드를 계승한다.

(iii) 'attrs'는 클래스의 속성 집합이다.

(iv) 'mthds'는 클래스의 메소드 집합이다.

〈표 1〉 공간 데이터타입 클래스 리스트

클래스명	내 용	표 현
POINT	지리적 형태의 점; 지구의 경도와 위도 값을 나타내는 좌표	s_point = 각각의 지리적 형태의 점 = (x, y) ∈ POINT
LINESTRING	지리적 형태의 선; 시작점과 끝점이 만나지 않는 점들의 집합.	s_line = 각각의 지리적 형태의 선 = (s_point_1, s_point_2, ⋯ , s_point_n) ∈ LINESTRING , $s\_point_1 \neq s\_point_n$
POLYGON	지리적 형태의 면; 시작점과 끝점이 만나는 점들의 집합	s_polygon = 각각의 지리적 형태의 면 = (s_point_1, s_point_2, ⋯, s_point_n) ∈ POLYGON , $s\_point_1 = s\_point_n$
GEOMCOLLECTION	하나 이상의 공간 객체의 집합으로 구성된 복합 객체	s_geomcollection = 각각의 공간 객체 집합 = (sdt_class_1, sdt_class_2, ⋯ , sdt_class_n) ∈ GEOMCOLLECTION , sdt_class_i ∈ SDT_CLASS ($1 \leq i \leq n$)

다음 <표 1>은 정의 2에서 정의한 공간 데이터타입 클래스들의 리스트를 나타낸다.

정의 3 (일반 공간 클래스). 일반 공간 클래스는 gs_class라고 표기하며 gs_class = (id, supclass, attrs, mthds) ∈ {GS_CLASS}이다.

(i) 'id'는 일반 공간 클래스에 대한 식별자이다.

(ii) 'supclass'는 일반 공간 클래스의 상위 클래스로서 supclass ∈ {SDT_CLASS}이다. 일반 공간 클래스는 상위 클래스의 속성과 메소드를 계승한다.

(iii) 'attrs'는 클래스의 속성 집합이다.

(iv) 'mthds'는 클래스의 메소드 집합이다.

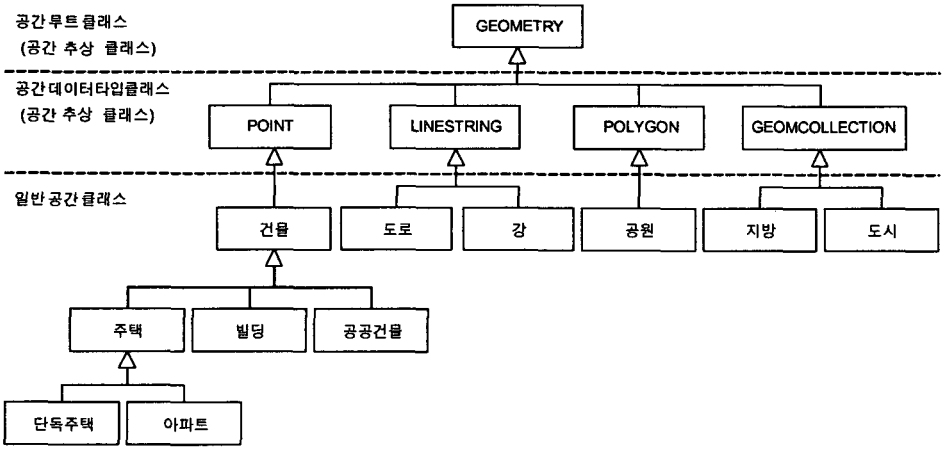
공간 클래스는 위에서 정의한 공간 루트 클래스, 공간 데이터타입 클래스, 일반 공간 클래스를 모두 포함한다. 공간 클래스는 정의 4에서 정의한다.

정의 4 (공간 클래스). 공간 클래스는 sclass로 표기하며 sclass ∈ SCLASS = {RS_CLASS, SDT_CLASS, GS_CLASS}이다.

공간 클래스 집합에는 클래스 간의 계층관계가 존재한다. 다음 정의 5는 공간 클래스 계층관계를 정의한다.

정의 5 (공간 클래스 계층). 공간 클래스 집합 SCLASS = {$sclass_1$, $sclass_2$, ⋯}는 공간 클래스 간에 부분 순서 (Partial Order)를 가지며 이러한 관계를 공간 클래스 계층이라고 한다. 만약($sclass_1 \geq sclass_2$)이 만족되면 $sclass_1$는 $sclass_2$의 상위클래스(Superclass)라고 하며 $sclass_2$는 $sclass_1$의 하위 클래스(Subclass)라고 한다. 하위 클래스 $sclass_2$는 상위클래스 $sclass_1$의 속성과 메소드를 계승한다.

위 정의 5에 따라 공간 루트 클래스, 공간 데이터타입 클래스, 일반 공간 클래스 간에는 공간 클래스 계층 관계가 존재한다. 즉, rs_class ≥ std_clas ≥ gs_class 관계가 성립된다. 또한, 일반 공간 클래스 간에도 공간 클래스 계층



〈그림 3〉 OGC 모델에 기반한 공간 클래스 계층

관계가 존재한다. 다음 <그림 3>은 공간 클래스 계층 관계의 사례를 보여준다.

공간 루트 클래스 'GEOMETRY'의 하위 클래스로는 공간 데이터타입 클래스 'POINT', 'LINESTRING', 'POLYGON', 'GEOMCOLLECTION'이 존재한다. 각 공간 데이터타입 클래스의 하위 클래스로는 일반 공간 클래스 '건물', '도로', '강', '공원', '지방' 및 '도시'가 존재한다. 일반 공간 클래스 간에도 공간 클래스 계층관계가 존재한다. <그림 3>에서는 일반 공간 클래스 '건물'에는 세 개의 일반 공간 클래스 - '주택', '빌딩', '공공건물' - 가 하위클래스로 존재한다. 또한, 일반 공간 클래스 '주택'에도 하위 클래스로서 두 개의 일반 공간 클래스 - '단독주택', '아파트' - 가 존재한다.

일반 공간 클래스의 인스턴스로는 실 세계의 개체를 표현하는 공간 객체가 존재한다. 다음 정의 6은 공간 객체를 정의한다.

정의 6 (공간 객체). 공간 객체는 sobj로 표기하며 sobj = (id, class, attrs, mthds) ∈ SOBJ 이다.

(i) 'id' 는 공간 객체 식별자이다.

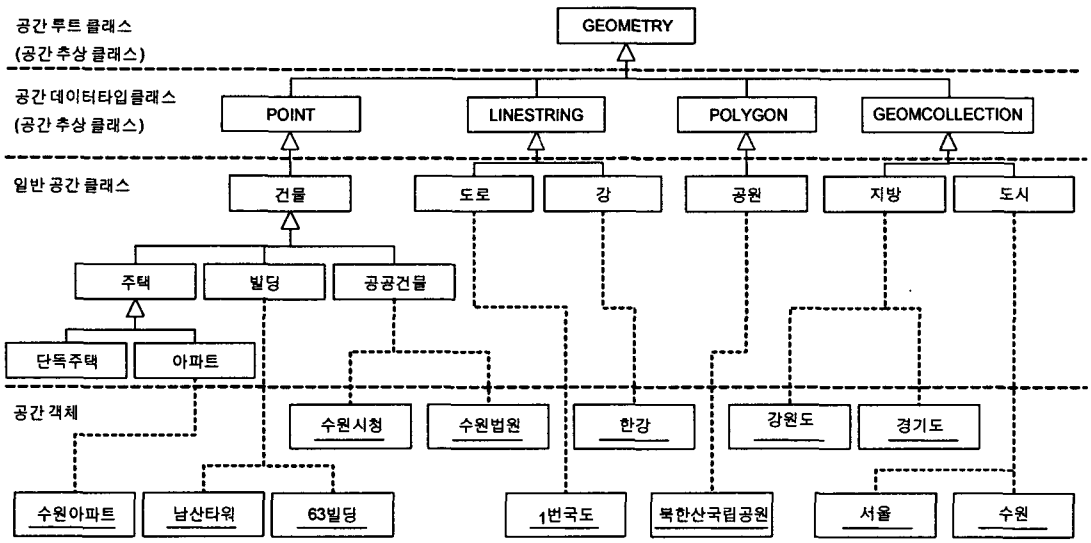
(ii) 'class'는 공간 객체가 속한 공간 클래스를 나타내며, class ∈ {GS_CLASS}이다.

(iii) 'attrs'는 클래스의 속성 집합이다.

(iv) 'mthds'는 클래스의 메소드 집합이다.

다음 <그림 4>는 <그림 3>의 공간 클래스가 인스턴스로 가지는 공간 객체의 사례를 보여준다. 예를 들면, 공간 객체 '수원아파트'는 공간 클래스 '아파트'의 인스턴스임을 나타낸다. <그림 4>에서 보는 것처럼 오직 일반 공간 클래스만이 공간 객체를 인스턴스로서 갖는다. 공간 루트 클래스와 공간 데이터타입 클래스는 추상 클래스이므로 공간 클래스만을 인스턴스로 갖는다. UML 표기 법에 따라 공간 객체는 공간 클래스와 동일한 형태로 표현하고, 공간 객체명에는 밑줄을 그어 공간 클래스와 구분한다. 공간 클래스에서 인스턴스된 공간 객체와의 연관관계는 점선으로 표시하였다.

공간 객체 상호간에는 지리적인 위치 관계가 존재하며, 이를 위상 관계라고 한다. 위상 관계는 연관화와 집합화 개념과 관련된다. 정의 7은 OGC 표준에 근거하여 모든 공간 객체가 공통적으로 가지는 위상 관계에 대한 메소드를 정의



〈그림 4〉 OGC 모델에 기반한 공간 객체

한다. 위상 관계 메소드는 정의 1에 의해 공간 루트 클래스에 직접적으로 포함되며, 모든 하위 공간 클래스와 공간 객체는 공통적으로 위상 관계 메소드를 계승한다.

정의 7 (공간 위상 관계 메소드). 공간 위상 관계 메소드는 smthd로 표기하며, smthd ∈ SMTHDS = {Equals, Disjoint, Intersects, Touches, Crosses, Within, Contains, Overlaps}이다. 각 메소드는 다형성 (Polymorphism)을 가지므로, 입력 값의 개수와 타입에 따라 서로 다른 결과 값을 반환한다.

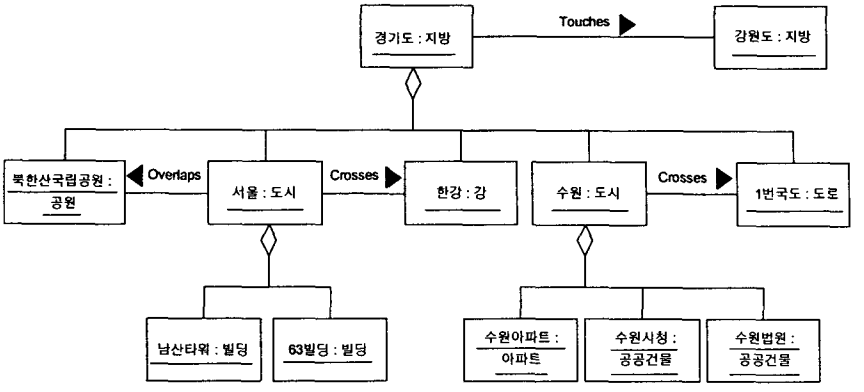
다음 <표 2>은 정의 7에서 정의한 공간 위상 관계 메소드들의 리스트를 나타낸다.

다음 <그림 5>은 <그림 4>에서 사례로 든 공간 객체 간의 공간 위상 관계를 UML 표기법으로 나타낸다. 공간 위상 관계에서 'Overlaps', 'Touches', 'Crosses'는 연관화로 표시하며, 'Contains', 'Within'은 집합화에 따른 합성 클래스로 표기한다. 예를 들면, 공간 객체 '경기도'와 '강원도'는 지리적으로 서로 만나는 관계이므로 '경기도. Touches(원도) = TRUE'이다. '경기도' 공간 객체는 공간 객체 '서울', '수원', '북한산국립공원', '한강' 및 '1번국도'로 구성되므로, '경기도. Contains(서울) = TRUE'와 '경기도. Contains(번국도) = TRUE'가 성립한다. 공간 객체 '서울'은 공간 객체 '북한산국립공원'과 서로 겹치는 관계이므로, '서울. Overlaps(북한산국립공원) = TRUE'이 성립한다. 공간 객체 '수원'은 공간 객체 '1번국도'와 교차하는 관계이므로 '수원. Crosses(1번국도) = TRUE'가 성립한다.

공간 위상 메소드에서 공간 객체를 반환하는 메소드는 반환하는 결과 값을 'GEOMETRY'클래스 타입에서 하위 클래스 타입으로 형 변환 (Type Casting)이 가능하다. 공간 위상 관계 메소드 '경기도. Contains() : GEOMETRY'는 공간 객체 '경기도'가 포함하는 모든 타입의 공간 객체 - '서울', '수원', '북한산국립공원', '1번국도', '남산타워', '63빌딩', '한강', '수원아파트', '수원시청', '수원법원' -를 결과값으로 반환한다. 그

〈표 2〉 공간 위상 관계 메소드 리스트

메소드	입력값	반환값	내용
Equals	anotherGeometry : GEOMETRY	Boolean	자신이 속한 공간 객체와 입력 값으로 받은 공간 객체가 지리적으로 동일한 위치에 존재하면 TRUE 값을 반환하고 그렇지 않으면 FALSE 값을 반환.
	-	GEOMTRY	자신이 속한 공간 객체와 지리적으로 동일한 위치에 존재하는 공간 객체들을 결과 값으로 반환.
Overlaps	anotherGeometry : GEOMETRY	Boolean	자신이 속한 공간 객체와 입력 값으로 받은 공간 객체가 지리적으로 겹치는 위치에 존재하면 TRUE 값을 반환하고 그렇지 않으면 FALSE 값을 반환.
	-	GEOMTRY	자신이 속한 공간 객체와 지리적으로 겹치는 위치에 존재하는 공간 객체들을 결과 값으로 반환.
Touches	anotherGeometry : GEOMETRY	Boolean	자신이 속한 공간 객체와 입력 값으로 받은 공간 객체가 지리적으로 만나는 위치에 존재하면 TRUE 값을 반환하고 그렇지 않으면 FALSE 값을 반환.
	-	GEOMTRY	자신이 속한 공간 객체와 지리적으로 만나는 위치에 존재하는 공간 객체들을 결과 값으로 반환.
Crosses	anotherGeometry : GEOMETRY	Boolean	자신이 속한 공간 객체와 입력 값으로 받은 공간 객체가 지리적으로 교차하는 위치에 존재하면 TRUE 값을 반환하고 그렇지 않으면 FALSE 값을 반환.
	-	GEOMTRY	자신이 속한 공간 객체와 지리적으로 교차하는 위치에 존재하는 공간 객체들을 결과 값으로 반환.
Contains	anotherGeometry : GEOMETRY	Boolean	자신이 속한 공간 객체가 입력 값으로 받은 공간 객체를 지리적으로 포함하는 위치에 존재하면 TRUE 값을 반환하고 그렇지 않으면 FALSE 값을 반환.
	-	GEOMTRY	자신이 속한 공간 객체가 지리적으로 포함하는 위치에 존재하는 공간 객체들을 결과 값으로 반환.
Within	anotherGeometry : GEOMETRY	Boolean	자신이 속한 공간 객체가 입력 값으로 받은 공간 객체에 지리적으로 포함되는 위치에 존재하면 TRUE 값을 반환하고 그렇지 않으면 FALSE 값을 반환. 그러므로, a, b ∈ sobj 일 때 a.Within(b) = b.Contains(a) 조건을 만족함.
	-	GEOMTRY	자신이 속한 공간 객체가 지리적으로 포함되는 위치에 존재하는 공간 객체들을 결과 값으로 반환.
Intersects	anotherGeometry : GEOMETRY	Boolean	자신이 속한 공간 객체와 입력 값으로 받은 공간 객체가 지리적으로 'Equals', 'Overlaps', 'Touchs', 'Crosses', 'Contains' 및 'Within' 중 하나의 위치 관계라도 만족하면 TRUE 값을 반환하고 그렇지 않으면 FALSE 값을 반환.
	-	GEOMTRY	자신이 속한 공간 객체와 지리적으로 'Equals', 'Overlaps', 'Touchs', 'Crosses', 'Contains' 및 'Within' 위치 관계를 만족하는 모든 공간 객체들을 결과 값으로 반환.
Disjoint	anotherGeometry : GEOMETRY	Boolean	자신이 속한 공간 객체와 입력 값으로 받은 공간 객체가 지리적으로 'Equals', 'Overlaps', 'Touchs', 'Crosses', 'Contains' 및 'Within' 중 하나의 위치 관계라도 만족하지 않으면 TRUE 값을 반환하고 그렇지 않으면 FALSE 값을 반환. 그러므로, a, b ∈ sobj 일 때 a.Disjoint(b) = ! a.Intersects(b) 조건을 만족함.
	-	GEOMTRY	자신이 속한 공간 객체와 지리적으로 'Equals', 'Overlaps', 'Touchs', 'Crosses', 'Contains' 및 'Within' 위치 관계를 만족하지 않는 모든 공간 객체들을 결과 값으로 반환.



〈그림 5〉 OGC 모델에 기반한 공간 위상 관계

러나, 공간 메소드 '경기.Contains() : 빌딩'은 공간 객체 '남산타워'와 '63빌딩'만을 반환한다.

일반적으로 객체지향 시스템에서 클래스와 클래스의 인스턴스는 객체라는 동일한 관점으로 바라본다[Jay Banerjee 외 6인, 1987]. 그러므로, 본 논문의 다음 장부터는 문장의 의미에 따라 클래스도 하나의 객체로서 다루도록 한다.

4. OOGIS-RBAC 모델

본 장에서는 제 3장에서 정의한 객체지향 GIS에서 필요한 개념들을 기반으로 OOGIS-RBAC 모델을 제시한다. OOGIS-RBAC 모델은 객체지향 GIS의 특성을 반영하기 위하여 NIST RBAC 표준 모델을 확장하여 제시한 버전이다. 다음의 정의 8에서 OOGIS-RBAC 모델의 기본 요소를 제시한다.

정의 8 (OOGIS-RBAC 모델의 기본 요소). OOGIS-RBAC 모델은 다음의 기본 요소를 포함한다.

(i) U(Users) = $\{u_1, \cdots, u_i\}$, 사용자들의 집합.

(ii) R(Roles) = $\{r_1, \cdots, r_j\}$, 역할들의 집합.

(iii) RH(Role Hierarchies) $\subseteq$ R X R, 역할 계층 구조 또는 역할의 지배관계에서 R에 대한 부분순서 : $r_1, r_2 \in R$ 이고 $r_1 \geq r_2$ 이면 r_1은 r_2 보다 상위 역할로서 r_2의 접근권한을 계승한다.

(iv) SE(Spatial Extents) = $\{se_1, \cdots, se_n \mid se_k \in sobj, 1 \leq k \leq n\}$, 공간 영역들의 집합.

(v) SR(Spatial Roles) = R X SE = $\{sr_1, \cdots, sr_m \mid sr_k = (r, se), r \in R, se \in SE, 1 \leq k \leq m\}$, 공간역할들의 집합: 공간역할은 역할의 직무 범위가 지리적 공간 영역의 내부 범위로 제한된다.

(vi) SRH(Spatial Role Hierarchies) $\subseteq$ SR X SR, 공간역할계층 구조 또는 공간역할의 지배관계에서 R에 대한 부분순서 및 SE에 대한 포함관계 : $sr_1 = (r_1, se_1)$, $sr_2 = (r_2, se_2)$, $(r_1, r_2 \in R)$, $(sr_1, sr_2 \in SR)$ 이고 $sr_1 \geq sr_2$ 이면 $(r_1 \geq r_2) \wedge (se_1.\text{Contains}(se_2) = \text{TRUE})$가 성립한다.

(vii) OP(Operations) = {Read, Write | Write $\geq$ Read}, 오퍼레이션들의 집합 : $op_1, op_2 \in OP$ 이고 $op_1 \geq op_2$ 이면 op_1은 op_2의 기능을 함축한다. Write 연산은 Read 연산에 대한 기능을 함축하고 있으므로 (Write $\geq$ Read)가 성립한다.

(viii) STOBJ(Spatial Target Objects) = $\{tobj_1,$

…, $tobj_t$ | $tobj_k$ ∈ {SOBJ, SCLASS, smthd() : Geometry }, 1 ≤ k ≤ t}, 공간 객체들의 집합: 1) $tobj_k$ ∈ SOBJ 이면, $tobj_k$는 하나의 공간객체를 대상으로 하며, 2) $tobj_k$ ∈ SCLASS 이면 클래스 계층관계에 기반하여 해당 클래스에 직접적으로 포함되는 공간객체들과 하위 클래스에 포함되는 공간객체들을 모두 대상으로 하며, 3) $tobj_k$ ∈ smthd() : Geometry 이면 공간객체의 위상관계에 기반하여 의해 해당 공간객체와 특정한 위치 관계를 갖는 공간객체의 집합을 대상으로 한다.

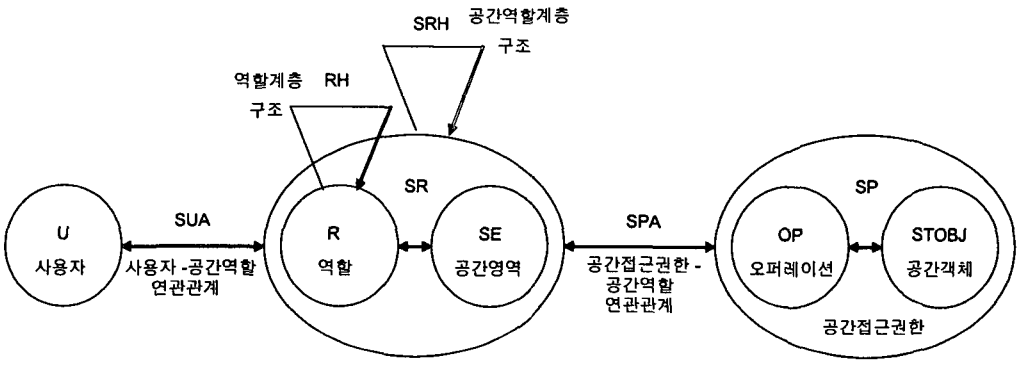
(ix) SP(Spatial Permissions) = OP × STOBJ = {sp_1, …, sp_q | sp_k = (op, tobj), op ∈ OP, tobj ∈ STOBJ, 1 ≤ k ≤ q }, 공간접근권한들의 집합.

(x) SUA(Spatial User Assignment) ⊆ U X SR = U X R X SE, 다대다 사용자와 공간역할의 연관관계.

(xi) SPA(Spatial Permission Assignment) ⊆ SR X SP = R X SE X OP X STOBJ = {sp_1, …, sp_w | sp_k = pa_type(sr, sp) = pa_type(r, se, op, tobj), pa_type ∈ PA_TYPE, sr ∈ SR, sp ∈ SP, r ∈ R, se ∈ SE, op ∈ OP, tobj ∈ STOBJ, 1 ≤ k ≤ w}, 다대다 공간접근권한과 공간역할과의 연관관계 : PA_TYPE = {Strong_PA, Weak_PA} [Rabitti et al., 1991], 1) 'Strong_PA'는 한번 접근권한이 설정되면 다른 접근권한으로 변경될 수 없는 접근권한의 유형이며, 2) 'Weak_PA'는 접근권한이 설정되었더라도 다른 접근권한으로 변경이 가능한 접근권한 유형이다.

다음 <그림 6>은 정의 8에서 정의된 OOGIS-RBAC의 기본 요소와 기본 요소들간의 연관관계를 도식적으로 나타낸다.

기존 표준 RBAC 모델과 OOGIS-RBAC 모델의 주요한 차이점은 공간역할과 공간접근권한이 정의되었다는 점이다. 공간역할은 기관 내에서 특정한 직무를 나타내는 역할뿐만이 아니라 직무를 수행하기 위한 지리적 업무 영역을 포함하여 정의하며, 공간역할이 포함하는 역할의 직무 계층과 공간 영역의 범위에 의해 공간역할계층을 구성한다. 공간접근권한은 시스템 내에서 수행하는 오퍼레이션의 대상이 제 3장에서 정의한 지리적 공간 클래스 또는 공간 객체임을 정의한다. 다음은 OOGIS-RBAC 모델



〈그림 6〉 OOGIS-RBAC 기본 모델

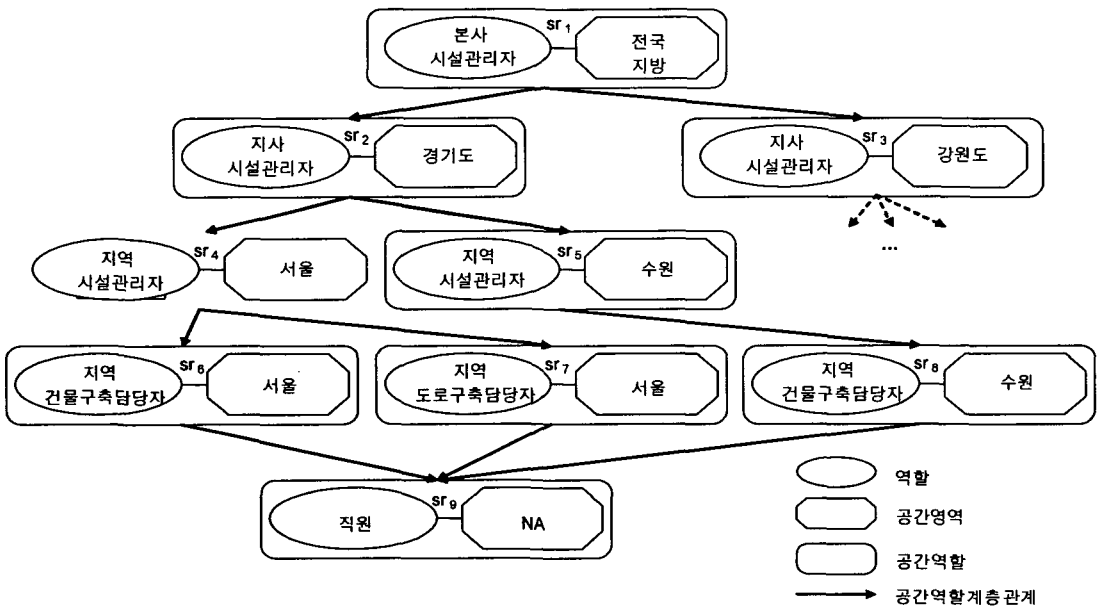
의 핵심적인 요소인 공간역할, 공간역할계층 및 공간접근권한에 대한 구체적인 사례를 제시한다.

4.1 공간역할과 공간역할계층

다음 <그림 7>은 공간역할 간의 계층 관계에 대한 사례를 보여준다. 타원은 역할을 표현하고, 팔각형은 공간 영역을 표현한다. 둥근 사각형은 공간역할을 표현한다. 화살표는 공간역할 간의 계층 관계를 표현한다. 공간역할은 정의 8에 의해 역할과 공간영역으로 구성된다. 공간역할은 직무에 의해 규정되는 업무 범위에 지리적인 업무 영역을 추가적으로 제한함으로써 최소 권한의 원칙을 제공한다. 다음 사례 1은 <그림 7>에서 표현된 공간역할에 대한 사례를 제시한다.

사례 1 (SR). <그림 7>에서 본사시설관리자에게는 공간 영역으로 '전국지방'을 할당하였다. 그러나, 경기도와 강원도 지사에서 근무하는 지사시설관리자는 각각 그들의 근무지에 따라 '경기도' 지방과 '강원도' 지방을 공간 영역으로 할당하였다. 그리고, 서울과 수원에 근무하는 지역시설관리자 또한 그들의 근무지에 따라 '서울'과 '수원'으로 공간 영역을 할당하였다. 또한, 공간 객체에 대한 접근 권한이 주어지지 않은 사원은 공간 영역에 대해 'NA(Not Available)'를 지정하였다. 이와 같이 공간역할은 역할에 따라 동일한 직무 권한을 부여하더라도, 공간 영역을 추가적으로 할당하여 직원의 근무지에 따라 접근 영역을 제한함으로써 최소 권한의 원칙을 제공한다.

역할 계층은 조직의 직무 권한과 책임을 수직적으로 나타내는 것으로써 다이어그램의 상위에 위치하는 역할은 하위에 위치하는 역할보다 보다 높은 직책과 보다 많은 권한 및 책임을 갖는다. 그러므로, 상위 역할은 하위 역할의 접근 권한을 계승한다. 공간역할의 구조는 직무의 계승뿐만이 아니라 공간 영역에 대한 포함 관계를 함축적으로 의미한다. 그러므로, 상위의 공간역할의 공간 영역은 하위의 공간역할의 공간 영역



〈그림 7〉 공간역할과 공간역할계층 관계의 사례

을 포함한다. 다음 사례 2는 공간역할계층에 대한 사례를 <그림 7>을 통해 보여준다.

사례 2 (SRH). <그림 7>에서 공간역할 sr_2 = (지사시설관리자, 경기도)는 sr_4 = (지역시설관리자, 서울)와 sr_5 = (지역시설관리자, 수원)의 상위 공간역할이다. 지사시설관리자는 지역시설관리자보다 상위 역할이고 경기도는 서울과 수원을 포함하므로 공간역할 sr_2는 공간역할 sr_4와 sr_5의 상위 공간역할이 된다. 그러나, 공간역할 sr_3 = (지사시설관리자, 강원도)는 sr_4와 sr_5의 상위 공간역할이 될 수 없다. 왜냐하면, 지사시설관리자가 지역시설관리자보다 상위의 역할이지만 강원도는 서울과 수원을 지리적으로 포함하지 않기 때문이다.

4.2 공간접근권한

공간접근권한은 특정한 모드의 오퍼레이션에 대해 하나 혹은 그 이상의 공간객체에 접근할 수 있는 권한을 부여하는 것이다. 오퍼레이션의 모드는 시스템의 구현 방식에 따라 달라진다. 예를 들면, 파일 시스템의 경우에는 'Read', 'Write' 만을 포함한다. 그러나, 데이터베이스관리시스템(Data Base Management System : DBMS)의 경우에는 오퍼레이션으로 'Select', 'Insert', 'Delete', 'Update' 등을 갖는다. 본 논문에서는 오퍼레이션의 표현을 단순화 하기 위해 'Read'와 'Write' 오퍼레이션 만을 포함한다. 대부분의 오퍼레이션은 'Read'와 'Write' 기능으로 분류가 가능하기 때문이다. DBMS의 'Select'는 'Read'로 분류가 가능하고 'Insert', 'Delete', 'Update'는 'Write'로 분류가 가능하다.

객체지향 GIS에서 오퍼레이션의 대상이 되는 공간객체는 하나의 지리적 객체뿐만 아니라 클래스화, 일반화, 연관화 및 집합화 개념을 통해 공간객체의 그룹이나 공간객체의 집합으로 확장될 수 있다. 하나의 오퍼레이션에 대해 대상이 되는 여러 개의 공간객체를 직접적으로 연관관계를 설정할 수 있지만, 이와 같은 방법은 다수의 연관관계를 설정함으로 인한 비효율성을 가져온다. 특히, GIS에서와 같이 접근해야 할 객체의 수가 증가하는 경우 연관관계를 관리하기 위한 비용은 더욱 증가하게 된다. 그러므로, 접근권한을 효율적으로 관리하기 위해 공간클래스의 계층관계와 공간객체의 위상관계와 효율적으로 이용하는 것이 중요하다. 다음 사례 3은 <그림 4>와 <그림 5>에서 예시된 공간객체와 공간객체의 위상관계를 이용하여 공간접근권한을 효율적으로 할당하는 사례를 제시한다.

사례 3 (SP). <그림 4>에서 공간 클래스 '건물'에 속한 모든 공간객체에 대해 접근권한을 설정을 할 때 다음과 같은 방법을 적용할 수 있다. 첫 번째 방법은 '수원아파트', '남산타워', '63빌딩', '수원시청' 및 '수원법원'에 대해 각각의 연관관계를 설정하는 것이다. 그러나, 이와 같은 방법은 공간객체에 대한 직접적인 연관관계의 수를 증가시키므로 접근제어 관리에 대한 비효율성을 가져온다. 두 번째 방법은 각각의 공간객체에 직접적인 연관관계를 설정하는 대신 공간클래스 '건물'에 대해 직접적인 연관관계를 설정하는 것이다. 이와 같은 방법은 단일한 직접연관관계 설정에 있어 공간클래스의 계층관계를 이용함으로써 공간클래스 '건물'의 모든 하위클래스 '단독주택', '아파트', '빌딩' 및 '공공건물'에 속한 모든 공간객체에 대한 다수의 연관관계를 함축적으로 설정할 수 있는 기능을 제공한다. 또한, <그림 5>에서 공간객체 '서울'에 속한 모든 공간객체에 대한 접근권한을 설정하려고 할 때도 공간객체 '서울'에 속한 '남산타워', '63빌딩' 및 '한강'에 대해 다수의 직접연관

관계를 설정하는 대신 공간 위상관계 메소드 'Seoul.Contains() : GEOMETRY'를 이용한 단일한 직접연관관계를 적용할 수 있다. 이는 공간객체의 위상관계를 이용한 단일한 직접연관관계를 적용함으로써 접근권한 설정이 필요한 공간객체들에 대해 다수의 함축적인 연관관계를 설정할 수 있는 효율적인 기능을 제공한다.

5. OOGIS-RBAC 모델의 연관관계 제한조건

본 장에서는 객체지향 GIS 환경에서 OOGIS-RBAC 모델을 적용하는 경우에 공간역할과 공간접근권한에 대한 연관관계 설정에 필요한 제한조건을 공간영역에 의한 제한, 공간클래스 계층관계에 의한 제한 및 공간객체 위상관계에 의한 제한으로 분류하여 제시한다. 또한 공간클래스 계층관계에 의한 제한 및 공간객체 위상관계에 의한 제한은 공간역할의 계층관계에 의한 접근권한 계승 여부에 따라 특성을 분류하여 제한조건을 제시한다.

5.1 공간영역에 의한 공간역할-공간접근권한 연관관계에 대한 제한조건

SPA에 의해 공간역할에 공간접근권한을 부여할 때 공간접근권한의 지리적 영역범위는 공간역할에 포함된 공간영역에 의해 제한된다. SPA에서 공간영역에 의한 제한조건은 다음 정의 9에서 정의한다.

정의 9 (공간영역에 의한 SPA 제한조건). SPA에 대해 spa = PA_TYPE(sr, sp) = PA_TYPE(r, se, op, tobj)가 실행될 때 다음의 제한조건이 적용되어야 한다. **1)** se.Intersects(tobj) = FALSE이면 SPA는 거부된다. **2)** (se.Intersects (tobj) = TRUE)이면 SPA는 허가된다. 그러나, 승인된 공간객체 authobj의 범위는 SPA에 의해 할당된 tobj와 se간의 위상관계에 의해 결정된다 : **2-1)** ((se. Contains(tobj) = TRUE) ∨ (se. Equals(tobj) = TRUE))이면, authobj = {tobj}이다. **2-2)** ((se.Overlaps(obj) = TRUE) ∨ (se.Touches(tobj) = TRUE) ∨ (se.Crosses(tobj) = TRUE) ∨ (se. Within(tobj) = TRUE))이면, authobj = {se.Contains(tobj):GEOMETRY ∧ tobj.Contains() : GEOMETRY}이다.

다음의 사례 4와 <그림 8>은 정의 9에서 정의한 공간영역에 의한 SPA 제한조건에 대한 사례를 기술한다.

사례 4 (공간영역에 의한 SPA 제한조건). <그림 7>에서 예시된 공간 역할 sr_4 = (지역시설관리자, 서울)의 직무범위는 <그림 8>에서 빗금 친 '서울' 공간영역 내부로 범위가 제한된다. 그러므로, spa_1 = Strong_PA(sr_4, sp_1) = Strong_PA(지역시설관리자, 서울, Read, 수원)이 실행될 때는 <그림 8>에서 보듯이 '서울. Intersects(수원) = FALSE'이므로 spa_1의 실행은

〈그림 8〉 공간영역에 의한 SPA 제한조건의 사례

거부된다. spa_2 = Strong_PA(sr_4, sp_2) = Strong_PA(지역시설관리자, 서울, Read, 경기도)가 실행될 때는 '서울.Within(경기도) = TRUE'이므로 sr_4는 공간객체 '서울'과 '경기도'가 겹치는 영역인 '서울'로 접근권한이 제한된다.

5.2 공간클래스 계층관계에 의한 공간역할-공간접근권한 연관관계에 대한 제한조건

SPA에 의해 공간역할에 공간접근권한을 부여할 때, 공간역할은 공간접근권한에 포함되어 있는 공간객체에 의해 직접적으로 접근권한을 할당받거나, 공간클래스의 계층관계에 의해 공간클래스에서 인스턴스된 하위 공간클래스 또는 공간객체에 대해 함축적으로 접근권한을 할당받는다. 그러므로, 이미 공간접근권한을 할당받은 공간객체에 대해 추가적으로 공간접근권한을 할당할 때는 공간클래스의 계층관계에 따른 제한조건을 가져야 한다. 추가적으로 공간접근권한을 할당할 때의 제한조건은 SPA의 모드에 따라 영향을 받는다. 즉, SPA의 모드가 Strong_PA인지 Weak_PA인지에 따라 제한조건이 달라지게 된다. 기존에 이미 Strong_PA로 공간접근권한이 할당되었다면 공간접근권한에 대한 변경은 허용되지 않는다. 그러나, Weak_ PA로 공간접근권한이 할당되었다면 공간접근권한을 변경하는 것은 허용된다. 단, Weak_PA로 공간접근권한이 할당되었더라고 이미 할당된 공간접근권한에 대해 중복적으로 공간접근권한을 할당하는 것은 허용되지 않는다. 다음 정의 10은 SPA에서 공간클래스의 계층관계에 의한 제한조건을 정의한다.

정의 10 (공간클래스 계층관계에 의한 SPA 제한조건). SPA에 대해 spa_1 = PA_TYPE(sr_1, sp_1) = PA_TYPE(r_1, se_1, op_1, $tobj_1$)이 실행된 후($tobj_1 \leq tobj_2$) $\vee$ ($tobj_1 \geq tobj_2$)인 조건을 만족하는 spa_2 = PA_TYPE(r_2, se_2, op_2, $tobj_2$)가 실행될 때 다음의 제한조건이 적용되어야 한다. **1)** spa_1 = Strong_PA(sr_1, sp_1) = Strong_PA(r_1, se_1, op_1, $tobj_1$)이 이미 실행된 후에 **1-1)** spa_2 = Strong_PA(sr_2, sp_2) = Strong_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때는(($tobj_1 \leq tobj_2$) $\wedge$ ($op_1 = op_2$))인 경우를 제외하면 spa_2의 실행은 허가되지 않는다. **1-2)** spa_2 = Weak_PA(sr_2, sp_2) = Weak_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때는 모든 경우에 대해 spa_2의 실행은 허용되지 않는다. **2)** spa_1 = Weak_PA(sr_1, sp_1) = Weak_PA(r_1, se_1, op_1, $tobj_1$)가 실행된 후에는 **2-1)** spa_2 = Strong_PA(sr_2, sp_2) = Strong_ PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때는 모든 경우에 대해 spa_2의 실행이 허용된다. **2-2)** spa_2 = Weak_ PA(sr_2, sp_2) = Weak_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때는(($tobj_1 \geq tobj_2$) $\wedge$ ($op_1 = op_2$))인 경우를 제외하고 spa_2의 실행이 허용된다.

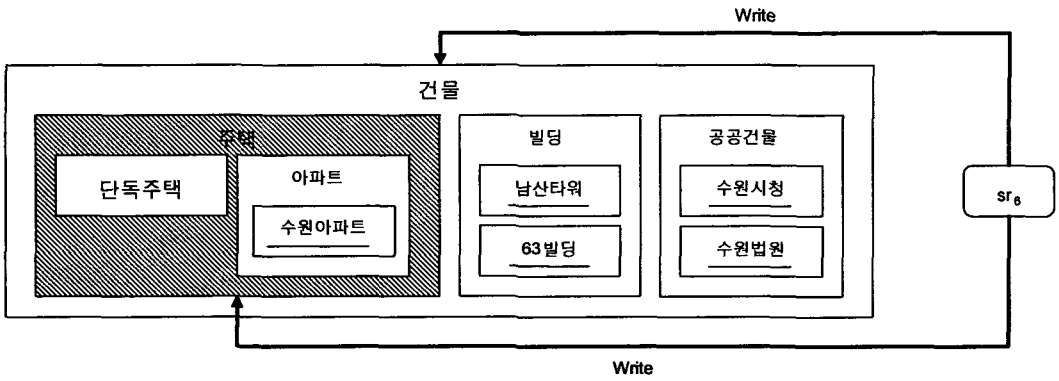
다음 사례 5와 <그림 9>는 정의 10에서 정의한 공간클래스 계층관계에 의한 SPA 제한조건에 대한 사례를 기술한다.

사례 5 (공간클래스 계층관계에 의한 SPA 제한조건). spa_1 = Strong_PA(sr_6, sp_1) Strong_PA(지역건물구축담당자, 서울, Write, 주택)이 실행된 경우, 공간 역할 sr_6는 <그림 9>에서 빗금 친 공간클래스 '주택' 뿐만이 아니라 '주택' 클래스의 하위 공간 클래스인 '단독주택'과 '아

파트'에 대해 Strong_PA에 의한 'Write' 접근권한을 갖는다. 그러므로, spa_2 = Strong_PA(sr_6, sp_2) = Strong_PA(지역건물구축담당자, 서울, Read, 건물)에 대한 실행은 거부된다. 왜냐하면, ((주택 ≤ 건물)∧(Write ≠ Read))조건이 성립되므로 기존에 이미 Strong_PA에 의해 접근권한이 할당된 공간클래스 '주택'에 대한 'Write' 권한을 'Read' 권한으로 변경하려고 시도하기 때문이다. 그러나, spa_1를 실행한 이후에 spa_3 = Strong_PA(sr_6, sp_3) = Strong_PA(지역건물구축담당자, 서울, Write, 건물)에 대한 실행은 허용된다. 이는 (주택 ≤ 건물) 조건이 성립되고 설정하고자 하는 오퍼레이션이 'Write'로 동일하므로 기존에 할당된 공간클래스 '주택'에 대한 'Write' 권한을 변경하지 않으면서 접근권한을 상위 공간클래스인 '건물'에 대한 'Write' 권한으로 확장하는 것이기 때문이다.

SPA에 대한 공간클래스 계층관계에 의한 제한조건은 공간역할에 대한 계층관계에 따라 상위 공간역할로 공간접근권한이 계승되는 경우도 고려하여 적용되어야 한다. 즉, 하위 공간역할에 할당된 공간접근권한은 함축적으로 상위 공간역할에 계승되기 때문에 공간클래스 계층관계와 공간역할 계층관계가 동시에 고려되어 제한조건이 정의되어야 한다. 상위 공간역할에 대한 접근권한은 하위 공간역할에 대한 접근권한 보다 권한이 낮게 설정되어서는 안 된다. 그러므로, 상위 공간역할에 공간접근권한을 설정하는 경우는 하위 공간역할로부터 함축적으로 계승한 공간접근권한을 기존보다 낮게 변경하는 공간접근권한은 제한되어야 한다. 또한, 상위 공간역할에 대한 접근권한은 이미 하위 공간역할로부터 함축적으로 계승 받은 접근권한에 대해 중복적으로 접근권한을 설정해서는 안 된다. 다음 정의 11은 SPA를 실행할 때 공간클래스 계층관계와 공간역할 계층관계에 의해 이미 함축적으로 할당된 접근권한인해 추가적으로 요구되는 제한조건을 정의한다. 정의 11의 제한조건도 SPA의 모드에 따라 달라진다.

정의 11 (공간역할 계층관계와 공간클래스 계층관계 의한 SPA 제한조건). SPA에 대해 spa_1 = PA_TYPE(sr_1, sp_1) = PA_TYPE(r_1, se_1, op_1, $tobj_1$)이 실행된 후 $sr_2 \geq sr_1 = (r_2, se_2) \geq (r_1, se_1) = ((r_2 \geq r_1) \vee (se_2 \supseteq se_1))$이고 $(tobj_1 \leq tobj_2) \vee (tobj_1 \geq tobj_2)$인 조건을 만족하는 spa_2 = PA_TYPE(sr_2, sp_2) = PA_

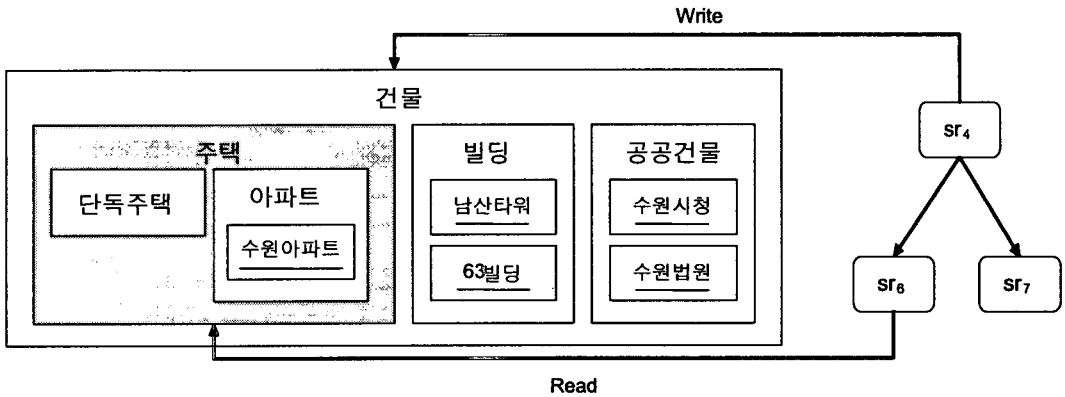


〈그림 9〉 공간클래스 계층관계에 의한 SPA 제한조건의 사례

TYPE(r_2, se_2, op_2, $tobj_2$)가 실행될 때 다음의 제한조건이 적용되어야 한다. **1)** spa_1 = Strong_PA(sr_1, sp_1) = Strong_PA(r_1, se_1, op_1, $tobj_1$)이 실행된 후에, **1-1)** spa_2 = Strong_PA(sr_2, sp_2) = Strong_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때는 (($tobj_1 \leq tobj_2$)∧($op_1 = op_2$))인 경우를 제외하면 spa_2의 실행은 거부된다. **1-2)** 또한, spa_2 = Weak_PA(sr_2, sp_2) = Weak_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때는 모든 경우에 대해 spa_2의 실행이 거부된다. **2)** spa_1 = Weak_PA(r_1, se_1, op_1, $tobj_1$)가 실행된 후에, **2-1)** spa_2 = Strong_PA(sr_2, sp_2) = Strong_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때(($tobj_1 \leq tobj_2$)∨($tobj_1 \geq tobj_2$))∧($op_1 \geq op_2$) 인 경우를 제외하고 spa_2의 실행이 허용된다. **2-2)** spa_2 = Weak_PA(sr_2, sp_2) = Weak_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때, (($tobj_1 \leq tobj_2$)∨($tobj_1 \geq tobj_2$))∧($op_1 \geq op_2$)이거나 (($tobj_1 \geq tobj_2$)∧($op_1 = op_2$))인 경우를 제외하고 spa_2의 실행이 허용된다.

다음 사례 6과 <그림 10>은 정의 11에서 정의한 공간클래스 계층관계와 공간역할 계층관계 의한 SPA 제한조건에 대한 사례를 보여준다.

사례 6 (공간클래스 계층관계와 공간역할 계층관계 의한 SPA 제한조건). spa_1 = Weak_PA(sr_6, sp_1)=Weak_PA(지역건물구축담당자, 서울, Read, 주택)이 실행된 경우, 공간 역할 sr_4는 <그림 10>에서 빗금 친 공간클래스 '주택'에 대해 Weak_PA에 의한 'Read' 권한을 하위 공간역할인 sr_6로부터 계승받는다. 이 경우 spa_2 = Strong_PA(sr_4, sp_2) = Strong_PA(지역시설관리자, 서울, Write, 건물)에 대한 실행은 허용된다. 왜냐하면, spa_2는 공간클래스 '주택'의 상위 공간클래스인 '건물'에 대한 'Write' 권한 설정을 시도함으로써 기존의 공간클래스 '주택'에 대한 'Read' 권한을 'Write' 권한으로 향상시키기 때문이다. 그러나, spa_3 = Weak_PA(sr_4, sp_3) = Weak_PA(지역시설관리자, 서울, Read, 아파트)에 대한 실행은 거부된다. 왜냐하면, ($sr_6 \leq sr_4$)이고 sr_4는 하위 공간역할 sr_6으로부터 이미 공간클래스 '주택'의 하위 공간클래스 '아파트'에



〈그림 10〉 공간클래스 계층관계와 공간역할 계층관계에 의한 SPA 제한조건의 사례

대한 'Read' 권한을 함축적으로 계승 받았으므로 spa_3에 대한 실행은 공간클래스 '단독주택'에 대한 'Read' 권한을 중복시키기 때문이다.

5.3 공간객체 위상관계에 의한 공간역할-공간접근권한 연관관계에 대한 제한조건

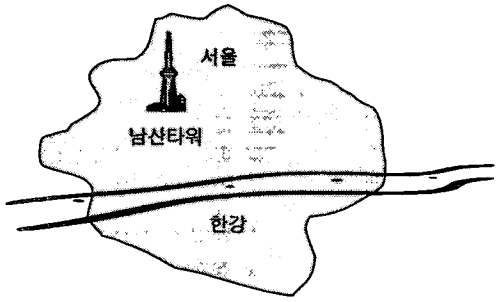
SPA에 의해 공간역할에 공간접근권한을 부여할 때, 공간객체의 위상관계를 이용한 공간객체 집합을 대상으로 접근권한을 부여할 수 있다. 그러므로, 이미 공간접근권한을 할당 받은 공간객체에 대해 추가적으로 공간접근권한을 할당할 때는 공간객체의 위상관계에 따른 제한조건이 추가적으로 필요하다. 공간객체 위상관계를 이용한 SPA 역시 SPA의 모드에 따라 제한조건이 달라진다. 다음 정의 12는 SPA를 실행할 때 공간객체 위상관계에 의한 제한조건을 정의한다.

정의 12 (공간객체 위상관계 의한 SPA 제한조건). SPA에 대해 spa_1 = PA_TYPE (sr_1, sp_1) = PA_TYPE(r_1, se_1, op_1, $tobj_1$)이 실행된 후 $tobj_2$. Intersects ($tobj_1$) = TRUE 인 조건을 만족하는 spa_2 = PA_TYPE(sr_2, sp_2) = PA_TYPE(r_2, se_2, op_2, $tobj_2$)가 실행될 때 다음의 제한조건이 적용되어야 한다. **1)** spa_1 = Strong_PA(sr_1, sp_1) = Strong_PA(r_1, se_1, op_1, $tobj_1$)가 실행이 된 후에 **1-1)** spa_2 = Strong_PA(sr_2, sp_2) = Strong_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때는 (($tobj_2$. Contains ($tobj_1$) = TRUE) ∧ (op_1 = op_2))인 경우를 제외하면 spa_2의 실행은 허가되지 않는다. **1-2)** spa_2 = Weak_PA(sr_2, sp_2) = Weak_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때는 모든 경우에 대해 spa_2의 실행은 허용되지 않는다. **2)** spa_1 = Weak_PA(sr_1, sp_1) = Weak_PA(r_1, se_1, op_1, $tobj_1$)이 실행된 후에, **2-1)** spa_2 = Strong_PA(sr_2, sp_2) = Strong_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때는 모든 경우에 대해 spa_2의 실행이 허용된다. **2-2)** spa_2 = Weak_PA(sr_2, sp_2) = Weak_PA(r_2, se_2, op_2, $tobj_2$)를 실행하고자 할 때는 (($tobj_2$. Within($tobj_1$) = TRUE) ∧ (op_1 = op_2))인 경우를 제외하고 spa_2의 실행이 허용된다.

다음 사례 7과 <그림 11>은 정의 12에서 정의한 공간객체 위상관계 의한 SPA 제한조건에 대한 사례를 보여준다.

사례 7 (공간객체 위상관계에 의한 SPA 제한조건). spa_1 = Strong_PA(sr_4, sp_1) Strong_PA(지역시설관리자, 서울, Read, 서울)이 실행된 경우, 공간 역할 sr_4는 <그림 11>에서 빗금친 공간객체 '서울' 뿐만이 아니라 '서울' 내에 포함된 모든 공간 객체에 대해 Strong_PA에 의한 'Read' 접근권한을 갖는다. 이 경우 spa_2 = Strong_PA(sr_4, sp_1) = Strong_PA(지역시설관리자, 서울, Write, 남산타워)에 대한 실행은 거부된다. 그 이유는 공간객체 '남산타워'는 이미 합성 공간객체 '서울'에 포함되어 있기 때문이다. 즉, '남산타워.Within(서울) = TRUE'이다. 그러므로, 공간영역 '서울'의 '지역시설관리자'는 이미 Strong_PA에 의해 공간객체 '남산타워'에 대한 'Read' 접근권한을 가지고 있음에도 불구하고 spa_2는 공간객체 '남산타워'의 접근권한을 'Write'로 변경하려고 시도하기 때문이다. spa_3 = Weak_PA(지역시설관리자, 서울, Write, 한강)에 대한 실행 역시 거부된다. 공간객체 '한

강' 은 공간객체 '서울'을 가로질러 흐르기 때문에 '한강.Crosses(서울) = TRUE'이다. 그러므로, 공간영역 '서울'의 '지역시설관리자'는 이미 Strong_PA에 의해 공간객체 '서울'에 포함되어 있는 '한강'의 일부분에 대한 'Read' 권한을 가지고 있지만, spa_3는 공간객체 '한강'을 모두 'Write' 권한으로 변경하려고 시도하기 때문이다.

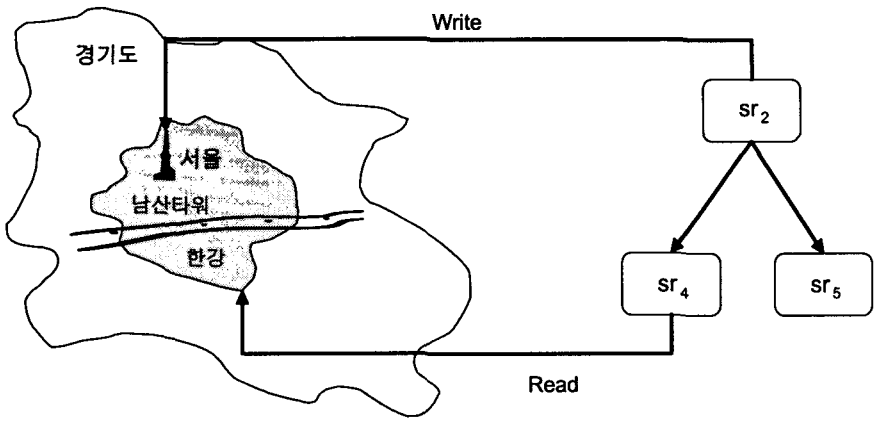


〈그림 11〉 공간객체 위상관계에 의한 SPA 제한조건의 사례

공간객체의 위상 관계를 이용한 SPA에 대해서도 공간역할의 계층관계에 의해 상위 공간역할로 공간접근권한이 계승되는 경우에 대한 제한조건이 추가되어야 한다. 상위 공간역할에 공간접근권한을 설정하는 경우는 하위 공간역할로부터 함축적으로 계승한 공간접근권한을 기존보다 낮게 변경하는 공간접근권한은 제한되어야 한다. 또한, 상위 공간역할은 하위 공간역할에 의해 함축적으로 계승된 공간객체에 대해 중복적으로 접근권한을 할당하거나 이미 할당된 접근권한에 대한 변경이 제한되어야 한다. 다음 정의 13는 SPA를 실행할 때 공간객체의 위상 관계와 공간역할의 계층관계에 의해 함축적으로 할당되는 접근권한으로 필요한 제한조건을 정의한다. 정의 13의 제한조건도 SPA의 모드에 따라 정의한다.

정의 13 (공간객체 위상관계와 공간역할 계층관계에 의한 SPA 제한조건). SPA에 대해 $spa_1 = PA_TYPE(sr_1, sp_1) = PA_TYPE(r_1, se_1, op_1, tobj_1)$이 실행된 후 $sr_2 \geq sr_1 = (r_2, se_2) \geq (r_1, se_1) = ((r_2 \geq r_1) \vee (se_2 \supseteq se_1))$이고 $tobj_2$. Intersects($tobj_1$) = TRUE 인 조건을 만족하는 $spa_2 = PA_TYPE(sr_2, sp_2) = PA_TYPE(r_2, se_2, op_2, tobj_2)$가 실행될 때 다음의 제한조건이 적용되어야 한다. **1)** $spa_1 = Strong_PA(sr_1, sp_1) = Strong_PA(r_1, se_1, op_1, tobj_1)$가 실행되었을 때, **1-1)** $spa_2 = Strong_PA(sr_2, sp_2) = Strong_PA(r_2, se_2, op_2, tobj_2)$에 대한 실행은 ($tobj_2$.Contains($tobj_1$) = TRUE) $\wedge (op_1 = op_2)$인 경우를 제외하면 spa_2의 실행은 거부된다. **1-2)** $spa_2 = Weak_PA(sr_2, sp_2) = Weak_PA(r_2, se_2, op_2, tobj_2)$를 실행하고자 할 때는 모든 경우에 대해 spa_2의 실행이 거부된다. **2)** $spa_1 = Weak\_PA(sr_1, sp_1) = Weak\_PA(r_2, se, op_1, tobj_1)$ 가 실행되었을 때, **2-1)** $spa_2 = Strong_PA(sr_2, sp_2) = Strong_PA(r_2, se_2, op_2, tobj_2)$를 실행하고자 할 때 ($tobj_2$. Intersects ($tobj_1$) $\wedge (op_1 \geq op_2)$)인 경우를 제외하고 spa_2의 실행이 허용된다. **2-2)** $spa_2 = Weak_PA(sr_2, sp_2) = Weak_PA(r_2, se_2, op_2, tobj_2)$를 실행하고자 할 때 ($tobj_2$.Intersects($tobj_1$) $\wedge (op_1 \geq op_2)$)이거나 ($tobj_2$.Within($tobj_1$) $\wedge (op_1 = op_2)$))인 경우를 제외하고 spa_2의 실행이 허용된다.

다음 사례 8과 <그림 12>는 정의 13에서 정의한 공간객체 위상관계와 공간역할 계층관계에 의한 SPA 제한조건에 대한 사례를 보여준다.



〈그림 12〉 공간객체 위상관계와 공간 역할 계층관계에 의한 SPA 제한조건의 사례

사례 8 (공간객체 위상관계와 공간역할 계층관계에 의한 SPA 제한조건). spa_1 = Weak_PA(sr_4, sp_1) = Weak_PA(지역시설관리자, 서울, Read, 서울)이 실행된 경우, 공간 역할 sr_2는 <그림 12>에서 빗금 친 공간객체 '서울'뿐만이 아니라 '서울' 내에 포함된 모든 공간 객체에 대해 Weak_PA에 의한 'Read' 접근권한을 하위 공간 역할인 sr_4로부터 계승을 받는다. 이 경우 spa_2 = Strong_PA(sr_2, sp_2) = Strong_PA(지사시설관리자, 경기도, Write, 남산타워)에 대한 실행은 허용된다. 왜냐하면, $sr_1 \leq sr_2$이고 '남산타워.Within(서울) = TRUE'이므로 spa_2의 실행은 '남산타워'에 대한 'Read' 권한을 'Write' 권한으로 향상시키기 때문이다. 그러나, spa_3 = Strong_PA(sr_2, sp_3) = Strong_PA(지사시설관리자, 경기도, NA, 남산타워)에 대한 실행은 거부된다. 상위 공간역할은 하위 공간역할로부터 함축적으로 계승한 공간접근권한을 기존보다 낮게 변경하는 공간접근이 제한되기 때문이다. 즉, spa_3의 실행은 상위 공간역할 sr_2가 함축적으로 계승 받은 공간객체 '남산타워'에 대한 'Read' 권한을 제거하기 때문이다.

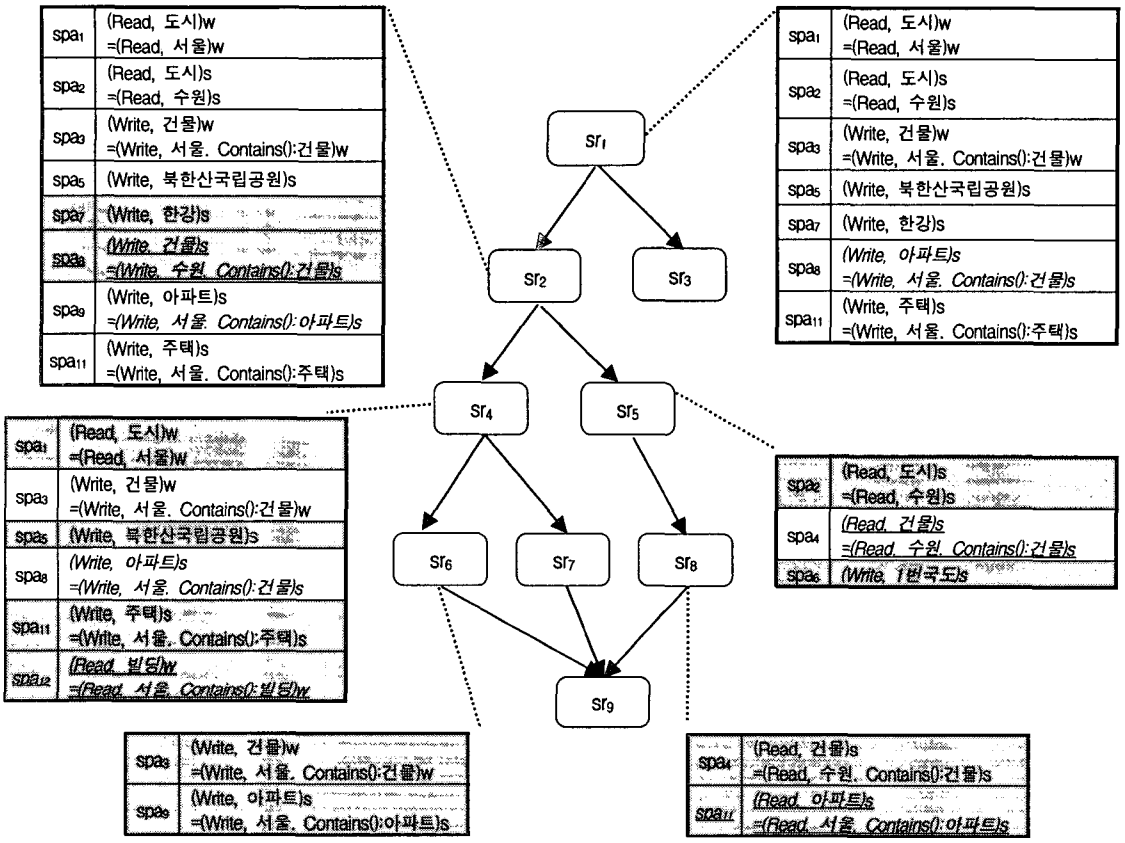
6. OOGIS-RBAC 모델의 적용 사례와 효율성 비교

본 장에서는 사례 연구를 통해 OOGIS-RBAC 모델의 정확성과 관리의 효율성을 증명하고자 한다. OOGIS-RBAC 모델의 정확성을 증명하기 위해서 OOGIS-RBAC의 제한조건이 적용되는 SPA 과정을 단계별로 제시한다. OOGIS-RBAC 모델의 관리의 효율성을 증명하기 위해서 OOGIS-RBAC 모델의 사례를 DAC 모델로 동일하게 표현하고 각 모델에서 연관관계 관리 측면 및 저장공간 사용 측면에서 효율성을 비교하고자 한다.

6.1 OOGIS-RBAC 모델에 대한 사례

<그림 13>은 <그림 4>에서 제시된 공간객체의 사례와 <그림 7>에서 제시한 공간역할의 사례를 이용하여 OOGIS-RBAC 모델에서 제한조건이 적용되는 SPA의 사례를 단계별로 제시한다.

<그림 13>에서는 공간역할은 둥근 사각형으로 표현되고, 공간역할과 연관관계를 맺은 공간접근권한은 표를 이용하여 보여준다. 각 공간접



〈그림 13〉 OOGIS-RBAC 모델의 단계별 SPA와 제한조건 사례

근권한 표에서는 접근권한을 할당 받은 순서대로 위에서부터 아래로 순서적으로 표시한다. 직접연관관계가 아니더라도 함축적으로 할당 받은 접근권한도 순서적으로 표시한다. 공간역할과 접근권한과의 연관관계는 점선으로 보여준다. 각 표에서 음영이 표시된 것은 직접연관관계를 나타내며 음영이 없는 것은 공간역할의 계층관계에 의해 함축적으로 접근권한이 설정된 것을 보여준다. SPA의 실행이 거절된 경우는 밑줄 친 이태릭체로 표시하였다. 접근권한에서 소문자 표기된 's'는 Strong_PA를 나타내고 'w'는 Weak_PA를 나타낸다.

다음은 <그림 13>에서 나타내는 SPA 적용 사례와 각 단계에서 공간객체의 특성에 의해 적용된 제한조건을 기술한다.

첫째, spa_1 = Weak_PA(sr_4, (Read, 도시)) = Weak_PA(지역시설관리자, 서울, Read, 도시)와 spa_2 = Strong_PA(sr_5, (Read, 도시)) = Strong_PA(지역시설관리자, 수원, Read, 도시)가 실행되었을 때, 정의 9에서 정의된 공간영역에 의한 SPA 제한조건에 의해 접근권한이 할당되는 것을 보여준다. 공간역할 sr_4와 sr_5에는 spa_1와 spa_2의 실행으로 공간클래스 '도시'가 할당되었지만, 각 공간역할에 할당된 공간영역의 범위에 의해 공간역할 sr_4는 공간객체 '서울'로 접근권한이 제한되고, 공간 역할 sr_5는 공간객체 '수원'으로 접근권한이 제한된다. 공간역할 sr_1과 sr_2는 sr_4와 sr_5의 상위 공간역할이므로 공간객체 '서울'

및 '수원'의 접근권한을 계승한다.

둘째, spa_3 = Weak_PA(sr_6, (Write, 건물))와 spa_4 = Strong_PA(sr_8, (Read, 건물))가 실행되었을 때는 공간역할 sr_6는 공간클래스 '건물'에 대한 'Write' 접근권한을 할당받고 정의 8의 공간영역에 의한 SPA 제한조건에 의해 '서울.Contains() : 건물'에 대한 접근권한을 받는다. 그리고, 이 접근권한은 상위 공간역할인 sr_4로 계승된다. 공간역할 sr_4가 이미 할당받은 공간객체 '서울'과 '서울.Contains() : 건물'은 위상관계를 가지므로 정의 12에서 정의된 공간역할 계층관계와 공간객체 위상관계 의한 SPA 제한조건이 적용되는지 판단한다. 이 경우 '서울'에 대한 'Read' 접근권한이 'Weak_PA'에 의해 설정되었으므로 spa_3에 의해 접근권한이 'Write'로 변경되는 것이 허용된다. 이와 달리 spa_4를 실행하였을 경우에는 공간역할 sr_8에 할당된 '서울 .Contains() : 건물'에 대한 접근권한이 상위 공간역할 sr_5에 계승되지만 이미 sr_5는 '수원'에 포함된 모든 객체에 대해 'Read' 접근권한을 가지고 있기 때문에 spa_4의 실행에 의해 접근권한이 중복되는 것이 거절된다.

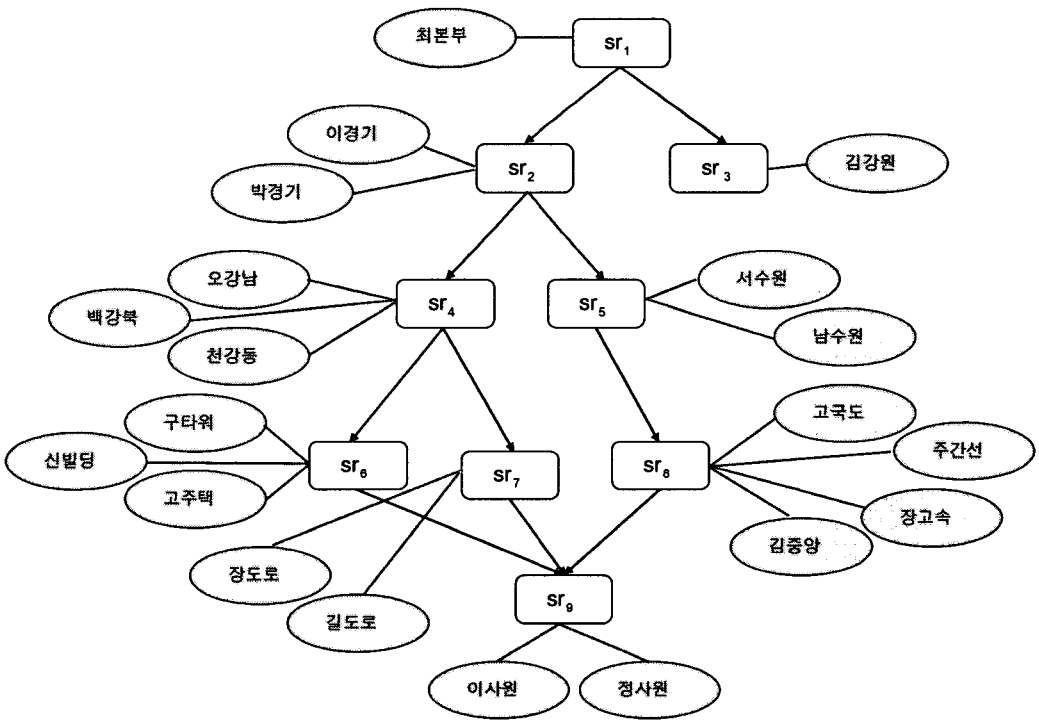
셋째, spa_5 = Strong_PA(sr_4, (Write, 북한산국립공원))과 spa_6 = Strong_PA(sr_5, (Write, 1번국도))가 실행되었을 때는 정의 11에 의한 공간객체 위상관계에 의한 SPA 제한조건이 적용되는 지 여부를 판단하게 된다. 공간역할 sr_4는 이미 공간객체 '서울'에 대한 접근권한을 가지고 있고 spa_5의 실행에 따라 새로 설정하려는 공간객체 '북한산국립공원'은 '서울'과 'Overlaps' 관계를 가지기 때문이다. 그러나, 이 경우는 공간객체 '서울'이 'Weak_PA'에 의한 'Read' 접근권한을 가지므로 '북한산국립공원'에 대해 'Write' 접근권한으로 변경하는 것이 허가된다. 이와 반면에 spa_6의 실행은 거절된다. 공간역할 sr_5는 공간객체 '수원'에 대해 'Strong_PA'에 의한 'Read' 접근권한을 가지고 공간객체 '수원'은 공간객체 '1번국도'와 'Crosses' 관계를 가진다. 그러므로, spa_6은 '수원' 내에 포함되어 있는 '1번국도'에 대한 'Read' 접근권한을 'Write' 접근권한으로 변경하는 것을 시도하는 것이기 때문에 공간객체 위상관계에 의한 SPA 제한조건이 적용되기 때문이다.

넷째, spa_7 = Strong_PA(sr_2, (Write, 한강))와 spa_8 = Strong_PA(sr_2, (Write, 건물))의 실행도 정의 13에 의해 공간역할 계층관계와 공간객체 위상관계 의한 SPA 제한조건이 적용되는 사례를 보여준다. 공간역할 sr_2는 접근권한에 대한 직접적인 연관관계를 갖지 않았으나 공간역할 계층관계에 의해 하위 공간역할인 sr_4와 sr_5에 할당된 접근권한을 계승을 받는다. 그리고, 함축적으로 계승된 접근권한이 이후에 실행되는 SPA의 접근권한과 충돌이 발생하는 지 여부를 판단하게 된다. 이 경우 spa_7의 실행은 공간객체 '한강'이 공간객체 '서울'과 'Crosses' 관계를 갖지만 공간객체 '서울'에 대한 접근권한이 'Weak_PA'에 의해 계승되었으므로 이미 '서울' 내에 포함되어 있는 '한강'의 일부분을 포함하여 '한강' 전체에 대해 'Write' 접근권한으로 설정하는 것이 가능하다. 그러나, spa_8에 대한 실행은 이미 함축적으로 접근권한을 가지고 있는 공간객체 '수원'에 포함된 '건물'에 대해 접근권한을 변경하려고 시도하지만 공간객체 '수원'에 대한 접근권한이 'Strong_PA'에 의한 접근권한이므로 'Read' 권한을 'Write' 권한으로 변경하려고 시도하는 것이 거절된다.

다섯째, spa_9 = Strong_PA(sr_6, Write, 아파트))와 spa_{10} = Strong_PA(sr_8, (Read, 아파트))의 실행은 정의 10에서 정의된 공간클래스 계층관계에 의한 SPA 제한조건의 사례를 보여준다. 공간역할 sr_6는 기존의 spa_3에 의해 '서울' 내에 포함된 '건물'에 대한 'Write' 접근권한을 'Weak_

PA'에 의해 획득하였고, 이를 spa_9의 실행에 의해 공간클래스 '건물'의 하위 클래스인 '아파트'에 대한 접근권한을 'Weak_PA'에서 'Strong_PA'로 변경하는 것이 허가된다. 그러나, 공간역할 sr_8은 spa_4에 의해 이미 공간클래스 '건물'에 대한 'Read' 접근권한을 'Strong_PA'에 의해 획득하였고 spa_{10}은 공간클래스 '건물'의 하위 클래스인 '아파트'에 대해 동일한 접근권한으로 중복 적용하려고 시도하는 것이기 때문에 공간클래스 계층관계에 의한 SPA 제한조건에 따라 실행이 거부된다.

마지막으로, spa_{11} = Strong_PA(sr_4, (Write, 주택))과 spa_{12} = Weak_PA(sr_4, (Read, 빌딩))의 실행은 정의 11에서 정의된 공간역할 계층관계와 공간클래스 계층관계 의한 SPA 제한조건의 사례를 보여준다. 공간역할 sr_4는 하위 공간역할인 sr_6로부터 공간클래스 '건물'에 대한 접근권한을 이미 계승받았다. 그러므로, spa_{11}의 실행 시에 공간클래스 '건물'의 하위 공간클래스인 '주택'에 대한 접근권한이 기존의 접근권한과 제한조건을 갖는 지 판단하게 된다. 이 경우 spa_{11}은 공간클래스 '주택'의 접근권한을 기존의 'Weak_PA'에서 'Strong_PA'로 변경하려는 것이므로 실행이 허용된다. 또한, 공간역할 sr_4는 '주택'의 하위 클래스인 '아파트'에 대한 접근권한을 가지고 있지만 spa_{11}의 실행은 아파트에 대한 접근권한을 변경시키지 않으므로 하위 클래스 '아파트'에 대한 접근권한은 상위 클래스 '주택'에 대한 접근권한으로 대체되게 된다. 공간클래스 '주택'에 대한 접근권한은 상위 공간역할인 sr_2로 계승되고 sr_2의 '아파트'에 대한 접근권한 역시 상위 클래스 '주택'에 대한 접근권한으로 대체된다. 이와 반면에 spa_{12}는 공간역할 계층관계와 공간클래스 계층관계 의한 SPA 제



〈그림 14〉 OOGIS-RBAC에서 사용된 사용자와 공간역할과의 관계

한조건에 따라 실행이 거절된다. 그 이유는 spa_{12}는 하위 공간역할로부터 함축적으로 계승받은 '건물'에 대한 하위 공간클래스 '빌딩'에 대해 'Write' 권한을 가지고 있고 이미 계승받은 '빌딩'에 대한 'Write'권한을 'Read' 권한으로 하향시키려고 하기 때문이다. 공간역할의 계승관계에 따라 상위 공간역할은 하위 공간역할의 접근권한을 포함해야 하기 때문이다.

6.2 DAC 모델의 사례

다음은 <그림 13>에서 제시된 OOGIS-RBAC 모델의 공간역할과 공간접근권한과의 관계에 대한 사례를 DAC 모델을 이용하여 표현하고자 한다. OOGIS-RBAC 모델에서 사용된 사용자와 공간역할과의 관계는 <그림 14>와 같다고 가정한다. <그림 14>에서는 공간역할은 둥근 사각형으로 표현되고, 공간역할과 연관관계를 맺은 사용자는 타원을 이용하여 보여준다. 공간역할과 사용자와의 연관관계는 가는 점선으로 보여준다.

DAC 모델은 사용자의 식별자에 따라 객체에 대한 접근 권한을 제한하는 방식으로 일반적으로 시스템 내에서는 ACL로 구현된다. ACL은 특정 객체에 대해 접근권한을 가진 최종 사용자에 대한 리스트를 표현하는 방식이므로, <그림 13>에서 할당된 공간객체와 <그림 14>에서 예시된 최종사용자를 이용하여 표현된다. 다음 <표 3>은 <그림 13>과 <그림 14>에서 표현된 OOGIS-RBAC 모델의 SUA와 SPA에 대한 사례를 ACL을 이용하여 나타낸다. 다만, ACL은 공간영역제한에 의한 공간객체의 일부에 대한 접근권한을 보여주지 못하므로 <그림 13>에서 공간객체의 일부에 대한 접근권한은 ACL에서는 공간객체 전체에 대한 접근권한으로 할당한다. 또한, ACL에서는 'Strong_PA'와 'Weak_PA'를 구별하지 않았으며, Write 연산을 Read 연산의 함축적인 의미로 사용하는 대신 독립적인 연산으로 구별하였다.

〈표 3〉 ACL의 사례

SOBJ	U : OPs
남산타워	최본부 : Read, Write, 이경기 : Read, Write, 박경기 : Read, Write, 오강남 : Read, Write, 백강북 : Read, Write, 천강동 : Read, Write, 구타워 : Read, Write, 신빌딩 : Read, Write, 고주택 : Read, Write
63빌딩	최본부 : Read, Write, 이경기 : Read, Write, 박경기 : Read, Write, 오강남 : Read, Write, 백강북 : Read, Write, 천강동 : Read, Write, 구타워 : Read, Write, 신빌딩 : Read, Write, 고주택 : Read, Write
수원아파트	최본부 : Read, 이경기 : Read, 박경기 : Read, 서수원 : Read, 남수원 : Read, 고국도 : Read, 장고속 : Read, 주간선 : Read, 김중앙 : Read
수원시청	최본부 : Read, 이경기 : Read, 박경기 : Read, 서수원 : Read, 남수원 : Read, 고국도 : Read, 장고속 : Read, 주간선 : Read, 김중앙 : Read
수원법원	최본부 : Read, 이경기 : Read, 박경기 : Read, 서수원 : Read, 남수원 : Read, 고국도 : Read, 장고속 : Read, 주간선 : Read, 김중앙 : Read
북한산국립공원	최본부 : Read, Write, 이경기 : Read, Write, 박경기 : Read, Write, 오강남 : Read, Write, 백강북 : Read, Write, 천강동 : Read, Write
1번국도	최본부 : Read, 이경기 : Read, 박경기 : Read
한강	최본부 : Read, 이경기 : Read, 박경기 : Read

6.3 OOGIS-RBAC 모델의 사례와 DAC 모델의 사례에 대한 비교

다음은 <그림 13>과 <그림 14>에서 제시된 OOGIS-RBAC 모델의 사례와 <표 3>에서 제시된 DAC 모델의 사례를 통해 접근제어 관리를 위해 필요한 연관관계의 수와 연관관계 기능의 효율성 및 저장공간 사용의 효율성 측면에서 두 모델을 비교 설명하고자 한다.

첫째, 접근제어 관리를 위해 필요한 연관관계의 수를 비교한다. OOGIS-RBAC 모델에서의 연관관계는 사용자와 공간역할과의 관계, 공간역할과 공간접근권한과의 관계 및 역할과 공간영역과의 관계가 존재한다. 공간객체의 계층관계 및 위상관계는 객체지향 GIS에서 데이터를 관리하기 위해 기본적으로 필요한 기능이므로 OOGIS-RBAC 모델을 관리하기 위한 추가 비용에서는 제외한다. OOGIS-RBAC 모델에서 사용자와 공간역할의 연관관계의 수는 <그림 14>에서 보듯이 전체 사용자 수 |U|와 동일하며, 공간역할과 접근권한과의 연관관계의 수는 <그림 13>에서 음영으로 표현된 직접 연관된 접근권한의 수 |P|와 동일하다. <그림 14>에서 음영이 표시가 안 된 연관관계는 공간역할의 계층관계에 의해 함축적으로 의미되는 것이므로 직접적인 연관관계의 수에서는 제외된다. 또한, 역할과 공간영역과의 연관관계는 <그림 7>에서 보듯이 전체 공간역할의 수 |R|과 동일하다. 즉, OOGIS-RBAC 모델에서 필요한 연관관계의 수 $N_{OOGIS\text{-}RBAC}$ = |U| + |P| + |R|이 성립하며 <그림 13>과 <그림 14>의 사례에서 $N_{OOGIS\text{-}RBAC}$ = 20 + 12 + 9 = 41개가 된다. 반면, DAC 모델에서는 공간객체와 최종사용자와의 연관관계가 존재하며 전체 연관관계의 수는 전체 공간객체의 수 |SOBJ|와 각 공간객체에 할당된 평균 사용자 수 $|U|_{AVG}$의 곱과 동일하다. 즉, DAC 모델에서 필요한 연관관계의 수 N_{DAC} = |SOBJ| x $|U|_{AVG}$가 성립하며 <표 3> 사례에서는 N_{DAC} = 57개의 연관관계가 필요하다. 본 논문에서 제시한 사례연구에서는 OOGIS-RBAC 모델의 특성을 설명하기 위한 최소한의 사용자와 공간객체만을 포함하였으므로 OOGIS-RBAC 모델의 사례와 DAC 모델의 사례에서 연관관계의 수가 많은 차이를 보이지는 않았다. 그러나, 실 환경에 있어서는 GIS의 최종 사용자 |U|는 수백에서 수천만에 이를 것이며, GIS를 구성하는 공간객체의 수 |SOBJ|는 수천만 혹은 수백만 이상이 된다. 그러므로, 본 논문에서 제시한 공간클래스의 계층관계와 공간객체의 위상관계를 이용한 공간객체의 접근권한 관리 방법을 적용하는 경우 |P| << |SOBJ|의 조건이 만족함을 고려할 때, |U| + |P| + |R| << $|U|_{AVG}$ x |SOBJ|가 성립된다. 즉, GIS에서 OOGIS-RBAC 모델로 공간객체에 대한 접근 권한을 관리하는 경우 DAC 모델에 비해 관리해야 할 연관관계의 수가 급격히 감소함으로 인해 접근제어 관리를 위한 전체적인 비용이 감소된다.

둘째, 연관관계 관리에 대한 기능적 효율성을 비교한다. 기업에서는 직원의 채용, 퇴직 및 부서이동 등으로 인해 최종사용자의 직무역할이 수시로 변화하게 된다. 이와 같은 경우 OOGIS-RBAC 모델과 DAC 모델에서의 관리의 효율성을 비교한다. 예를 들면, 사용자 '신규원'이 경력직원으로 채용되어 서울지역에 지역시설관리자가 되는 경우 OOGIS-RBAC 모델에서는 <그림 14>에서 공간역할 'sr_4'와 사용자 '신규원'에 대해 단 1번의 연관관계를 생성함으로써 공간역할 'sr_4'에 할당된 모든 접근권한을 할당할 수가 있다. 그러나, DAC 모델에서는 '신규원'이 지역시설관리자 임무를 수행하기 위해서 필요한 모든 공간객체에 대해 접근권한을 부여해야 하므로 <표 3>의 사례에서 공간객체 '남산타워', '63빌

딩' 및 '북한산국립공원' 각각의 행에 사용자 '신규원'에 대한 접근권한 리스트를 설정해야 하므로 모두 3번의 연관관계가 생성되어야 한다. 또한, '신규원'이 수원지역의 지역시설관리자로 부서가 변경되는 경우 OOGIS-RBAC 모델에서는 <그림 14>에서 공간역할 'sr_4'에서 '신규원'에 대한 연관관계를 삭제하고 공간역할 'sr_5'와 '신규원'에 대한 연관관계를 생성함으로써 부서변동에 따라 '신규원'이 필요로 하는 모든 접근권한을 변경할 수 있다. 즉, '신규원'의 부서변동에 따라 한번의 연관관계 삭제와 한번의 연관관계 생성으로 모두 2번의 연관관계 설정이 필요하다. 그러나, DAC 모델에서는 <표 3>에서 공간객체 '남산타워', '63빌딩' 및 '북한산국립공원' 각각의 행에서 사용자 '신규원'에 대한 접근권한 리스트를 삭제하고 공간객체 '수원아파트', '수원시청' 및 '수원법원'의 행에 사용자 '신규원'을 새로이 추가해야 하므로 모두 6번의 연관관계 설정을 처리해야 하는 문제가 발생한다. GIS 환경에서와 같이 하나의 사용자가 다수의 공간객체를 관리하는 환경에서는 사용자의 추가, 변경 및 삭제에 따라 ACL에 대한 관리 비용이 급격하게 증가하므로 GIS 환경에서 ACL로 접근권한을 관리하는 것은 매우 비효율적인 방법이다.

셋째, 저장공간 사용의 효율성을 비교한다. OOGIS-RBAC 모델에서 필요한 저장 공간은 <그림 14>에서 보듯이 사용자의 식별자를 저장하기 위한 문자열과 각 사용자에 대해 연관된 공간역할의 주소를 나타내는 정수값이 필요하다. 또한, <그림 13>에서 보듯이 음영으로 표시된 접근권한을 표현하기 위한 오퍼레이션의 종류를 나타내는 정수값, 공간객체 또는 공간 메소드를 표현하는 문자열, 접근권한의 타입을 나타내는 정수값 및 연관된 공간역할의 주소를 나타내는 정수값이 필요하다. 그 외에 <그림 7>에서 보듯이 공간역할을 나타내는 문자열, 공간영역을 나타내는 문자열 및 역할과 연관된 공간영역의 주소를 나타내는 정수값이 필요하다. 문자열에 대한 크기를 최대 128바이트라고 하고 정수값에 대한 크기를 8바이트라고 할 때 OOGIS-RBAC 모델을 표현하기 위한 저장공간 $S_{OOGIS-RBAC}$ = |U| x (128 + 8) + |P|x (8 + 128 + 8 + 8) + |R|x (128 + 128 + 8) 바이트이다. 본 사례에서는 |U| = 20, |P|= 12, |R|= 9이므로 $S_{OOGIS-RBAC}$ = 7,016 바이트이다. DAC 모델에서 필요한 저장공간은 <표 3>에서 보듯이 공간객체를 나타내는 문자열과 각 공간객체에 대해 접근권한을 가지는 사용자에 대해 사용자의 식별자에 대한 문자열과 오퍼레이션의 종류를 나타내기 위한 정수값이 필요하다. 그러므로, DAC 모델을 표현하기 위한 저장공간 S_{DAC} = |SOBJ| x 128 + |SOBJ| x |U| x 8 바이트이다. 본 사례에서는 |U| = 20, |SOBJ| = 8이므로 S_{DAC} = 2,304 바이트이다. 본 사례에서는 OOGIS - RBAC 모델을 표현하기 위해 DAC 모델에 비해 3배 정도의 저장공간을 필요로 하였다. 실제 기업 환경에서 공간역할의 수 |R|은 전체 사용자의 수 |U|나 전체 접근권한의 수 |P| 혹은 전체 공간객체의 수 |SOBJ|에 비해 매우 적다. 그러므로, $c_i(1 \le I \le 5)$를 상수라고 할 때 OOGIS- RBAC 모델에서 필요한 저장공간은 $S_{OOGIS-RBAC} = c_1|U| + c_2|P| + c_3$로 표현이 가능하고 DAC 모델에서 필요한 저장공간은 $S_{DAC} = c_4|SOBJ| + c_5|U||SOBJ|$로 표현된다. 본 논문에서 제시한 공간객체의 계층관계와 위상관계에 따른 공간접근권한 설정에 따라 |P| << |SOBJ| 조건을 만족하고, |SOBJ|는 실제 GIS 환경에서는 수백 혹은 수천만 개에 이르게 되므로 $c_1|U| + c_2|P| + c_3 << c_4|SOBJ| + c_4|U||SOBJ|$의 조건이 성립하게 된다. 즉, 실제 GIS 환경에서 공간객체에 대한 접근권한을 설정함에 있어 OOGIS-RBAC 모델에 의한 표현이 DAC 모델로 표현하는 것보다 적은 수의 저장공간을 필요로 하게

된다.

본 장에서는 OOGIS-RBAC 모델과 DAC 모델의 비교에 있어 연관관계 관리 및 저장공간 사용의 효율성 측면에서 OOGIS-RBAC 모델이 DAC 모델에 비해 우수한 특성을 보임을 제시하였다. 다만, OOGIS-RBAC 모델은 DAC 모델에 비해 접근권한을 가진 공간객체를 판단하기 위해 보다 많은 수의 연산이 요구된다는 문제점이 제기될 수 있다. OOGIS-RBAC 모델에서는 접근권한을 가진 공간객체를 판단하기 위해 공간역할 계층 관계에 따른 접근권한의 계승에 대한 연산 및 공간 클래스에 대한 계층관계와 공간 객체에 대한 위상관계를 이용한 연산이 추가적으로 요구되기 때문이다. 그러나, GIS는 서비스의 특성상 수 테라 혹은 수 페타에 이르는 공간 객체를 신속히 처리할 수 있도록 시스템 성능 및 용량을 산정하여 구축되므로 공간객체의 접근제어 관리를 위해 필요한 성능이 보장될 수 있는 특성이 있다.

7. 결 론

본 논문에는 NIST RBAC 표준을 확장하여 GIS 환경에서 효율적인 접근제어 방법을 제공하기 위한 OOGIS-RBAC 모델을 제시하였다. OOGIS-RBAC 모델의 기반이 되는 공간객체의 특성을 정형화하기 위해 객체지향 모델의 OGC 표준을 이용하였다. 객체지향 기법으로 실 세계의 공간 데이터를 정형화함으로써 접근제어의 기본이 되는 접근권한의 단위를 공간객체 자체뿐만 아니라 공간클래스의 계층관계 및 공간객체의 위상관계를 이용한 공간객체의 그룹 또는 집합으로 확장하여 효율적으로 공간역할에 접근권한을 할당할 수 있도록 하였다. 또한, OOGIS-RBAC 모델에서 필요한 공간제한조건을 공간객체의 특성 별로 제시하고, 다양한 사례를 통해 OOGIS-RBAC 모델의 정확성을 증명하였다. 또한, OOGIS-RBAC 모델의 사례와 동일한 결과를 갖도록 ACL을 이용하여 DAC 모델을 적용한 결과를 제시하고, 연관관계 관리 및 저장공간 사용의 효율성 측면에서 OOGIS-RBAC 모델이 DAC 모델에 비해 우수함을 증명하였다. 본 논문의 사례연구에서는 GIS 환경의 접근제어 관리에 있어 OOGIS-RBAC 모델을 DAC 모델과 비교하여 효율성을 제시하였지만, 전통적인 RBAC 모델과 비교하는 경우에도 OOGIS-RBAC 모델이 공간객체의 계층관계와 위상관계를 이용하여 전통적인 RBAC 모델에 비해 관리하여야 할 공간 접근권한의 수를 감소시키고 이에 따른 연관관계 수의 감소로 인해 접근제어 관리 비용 및 기능적 효율성과 저장관리의 효율성을 제공할 수 있음을 쉽게 예측할 수 있다.

향후에는 본 연구를 기반으로 공간객체에 대한 접근정책을 GML(Geographic Markup Language) 및 XACML(XML-based Access Control Language) 표준에 기반하여 구현할 수 있는 방법을 연구하고, 분산 시스템 환경하에서 SAML (Security Assertion Markup Language)을 이용하여 접근정책에 대한 정보를 제공할 수 있는 방법을 연구하고자 한다.

참 고 문 헌

[1] 김계현, *GIS 개론*, 대영사, 1998.

[2] Ahn, G. J. and Sandhu, R., "Role-based authorization constraints specification, "ACM Transactions on Information and System Security (TISSEC), Vol. 3, No. 4, Nov. 2000, pp. 207-226.

[3] Ardagna, C. A. et al., "Supporting location-based conditions in access control policies, "ACM Conference on Computer

and Communications Security, 2006, pp. 212-222.

[4] Atluri, V. and Chun, S. A., "An authorization model for geospatial data", IEEE Transactions on Dependable and Secure Computing, Vol. 1, No. 4, Oct-Dec 2004, pp. 238-254.

[5] Banerjee, J. Chou, H. T. Garza, J. F., Kim, W., Woelk, D., Ballou, N., and Kim, H. J., "Data model issues for object-oriented applications", ACM Transactions on Information Systems (TOIS), Vol. 5, No. 1, Jan. 1987.

[6] Belussi, A., Bertino, E., Catania, B., Damiani, and Nucita, M. L. A., "An authorization model for geographic maps", *In Proc. of the 12th annual ACM international workshop on* Geographic information systems, Washington DC, USA, No. 12-13, 2004, pp. 82-91.

[7] Bertino, Elisa et al., "GEO-RBAC : a spatially aware RBAC", *In Proc. of the tenth ACM symposium on Access control models and technologies*, Stockholm, Sweden, 2005, pp. 29-37.

[8] Egenhofer, M. and Frank, A., "Object-Oriented Modeling for GIS", Journal of the Urban and Regional Information Systems Association, Vol. 4, No. 2, 1992, pp. 3-19.

[9] Ferraiolo, D. F., Barkley, and J. F., Kuhn, D. R., "A role-based access control model and reference implementation within a corporate intranet", ACM Transactions on Information and System Security (TISSEC), Vol. 2, No. 1, Feb. 1999.

[10] Ferraiolo, D. F. et al., "Role-Based Access Control", Artech House, 2003.

[11] Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn D. R., and Chandramouli, R., "Proposed NIST Standard for Role-Based Access Control", ACM Transactions on Information and Systems Security (TISSEC), Vol. 4, No. 3, 2001.

[12] Hansen, Frode. and Oleshchuk, Vladimir., "SRBAC : A Spatial Role-Based Access Control Model for Mobile Systems", Nordsec 2003, No. 15-17 October 2003, Norways, pp. 129-141,

[13] Open GIS Consortium, "OpenGIS Simple Features Specification for SQL Revision 1.1", OpenGIS Project Document 99-049, May 5, 1999.

[14] Osborn, S., Sandhu, R., and Munawer, Q., "Configuring role-based access control to enforce mandatory and discretionary access control policies", ACM Transactions on Information and System Security (TISSEC), Vol. 3 No. 2, May 2000.

[15] Pires, F., Medeiros, C. B., and Silva, A. B., "Modeling Geographic Information Systems using an Object-Oriented Framework", *In Proceedings of the 13th International Conference of* the Chilean Computer Society, Chile, 1993, pp. 217-232.

[16] Rabitti, F. Bertino, E., Kim, W., and Woelk, D., "A model of authorization for next-generation database systems", ACM Transactions on Database Systems (TODS), Vol. 16, No. 1, March 1991, pp. 88-131.

[17] Sandhu, R. S., Coyne, E. J., Feinstein, H. L., and Youman, C. E., "Role-Based Acc-

ess Control Models", IEEE Computer, Vol. 29, No. 2, Feb. 1996, pp. 38-47.

[18] Voisard, A. and David, B., "A database perspective on geospatial data modeling", *IEEE Transactions on* Knowledge and Data Engineering, Vol. 14, No. 2, March-April 2002, pp. 226-243.

□ 저자소개

김 미 연

덕성여자대학교 전산학과 졸업하였으며, 고려대학교 컴퓨터과학기술대학원 소프트웨어공학과 석사, 고려대학교 정보경영공학전문대학원 정보보호전공 박사수료, 현재 KT 인프라연구소 책임연구원이며, 주요 관심분야는 데이터베이스, 네트워크보안, 정보보호 등이다.

이 철 민

경북대학교 컴퓨터공학과 졸업, KAIST 전산학과 석사, 현재는 KT 인프라연구소 책임연구원이며, 주요 관심분야는 컴퓨터 그래픽스, 데이터베이스, XML based 프로토콜 등이다.

문 창 주

고려대학교 컴퓨터과학과 학사, 석사, 박사를 받았으며, 고려대학교 정보보호대학원 연구교수, 건국대학교 컴퓨터응용과학부 조교수등을 역임했으며, 현재는 건국대학교 항공우주정보시스템공학과 조교수이다. 주요 관심분야는 실시간 임베디드 소프트웨어, 시스템통합, 정보보호 등이다.

이 동 훈

고려대학교 경제학사, Okahoma University 전산학 석사, Oklahoma University 전산학 박사, 고려대학교 전산학과 조교수, 고려대학교 전산학과 부교수, 현재는 고려대학교 정보경영공학전문대학원 교수, 주요 관심분야는 암호프로토콜, 암호이론, USN 이론, 키 교환, 익명성 연구, PET 기술 등이다.

◈ 이 논문은 2007년 05월 20일 접수하여 1차 수정을 거쳐 2007년 08월 31일 게재 확정되었습니다.