Journal of Korea Multimedia Society Vol. 7, No. 1, January 2004(pp. 12-26)

이질적인 공간정보시스템의 상호 운용성을 위한 효과적인 지리데이터의 GML 사상

정원일†, 배해영††

요 약

과거 지리정보시스템은 고유의 독립적인 형식으로 지리 정보를 구축하여 서비스를 제공하여 왔으며, 최근에는 기존의 이질적인 지리정보시스템간의 다양한 지리 정보를 효율적으로 활용하기 위해 지리정보시스템에서의 상호 운용성 제공이 요구되고 있다. 이에 OGC(Open GIS Consortium)에서는 분산 환경에서 이질적인 지리정보시스템들간의 상호 운용성을 제공하기 위해 GML(Geography Markup Language)을 제안하였다. GML은 XML(eXtensible Markup Language)을 기반으로 공간 정보와 비공간 정보를 포함하는 지리 정보를 저장하고 전송하기 위한 인코딩 방법에 대한 명세를 제공하고 있다. 또한, GML은 웹 환경에서 맵 서버를 통해 지리 정보를 서비스하기 위한 인터페이스의 구현에 관한 명세를 포함하고 있다. 이에 GML 문서에서의 지리 정보와 기존의 지리정보시스템들이 가지는 지리정보간의 호환을 위한 연구가 활발히 진행되고 있다. 본 논문에서는 이질적인 지리 정보들의 상호 운용성을 제공하기 위해 GML 문서와 지리정보시스템간의 지리 정보를 사상시키는 기법을 제안한다. 이를 위해 기존의 지리정보시스템의 지리 정보를 GML 문서로 표현하는 방법과 GML 문서에 나타나는 지리 정보를 공간데이터베이스로 구축하는 방법에 대해 기술한다. 제안된 지리 정보 사상 기법을 통해 기 구축된 지리 정보간의 상호 운용성을 제공하여 웹 기반의 통합된 지리 정보 서비스를 위한 프레임워크로 제공될 수 있다.

Efficient Publishing Spatial Information as GML for Interoperability of Heterogeneous Spatial Database Systems

Warnill Chung†, Hae-Young Bae††

ABSTRACT

In the past, geographic data is constructed and serviced through independent formats of its own according to each GIS(Geographic Information System). Recently, the provision of interoperability in GIS is important to efficiently apply the various geographic data between conventional GIS's. Whereupon OGC(Open GIS Consortium) proposed GML(Geography Markup Language) to offer the interoperability between heterogeneous GISs in distributed environments. The GML is an XML encoding for the transport and storage of geographic information, including both the spatial and non-spatial properties of geographic features. Also, the GML includes Web Map Server Implementation Specification to service the GML documents. Accordingly the prototype to provide the reciprocal interchange of geographic information between conventional GIS's and GML documents is widely studied. In this paper, we propose a mapping method of geographic information between spatial database and GML for the prototype to support the interoperability between heterogeneous geographic information. For this method, firstly the scheme of converting geographic information of the conventional spatial database into the GML document according to the GML specification is explained, and secondly the scheme to transform geographic information of GML documents to geographic data of spatial database is showed. Consequently, the proposed method is applicable to the framework for integrated geographic information services based on Web by making an offer the interoperability between already built geographic information of conventional GIS's using a mapping method of geographic information between spatial database and GML.

Key words: GML, GIS, Spatial DBMS(공간 DBMS), Interoperability(상호 운용성)

※ 교신저자(Corresponding Author) : 정원일, 주소 : 인천시 남구 용현3동 253번지(402-751), 전화 : 032)860-8712, FAX : 032)862-9845, E-mail : wnchung@dblab.inha.ac.kr
접수일 : 2003년 3월 6일, 완료일 : 2003년 7월 4일
† 준회원, 인하대학교 대학원 전자계산공학과 박사과정
†† 인하대학교 전자계산공학과 교수
(E-mail : hybae@inha.ac.kr)
※ 본 연구는 정보통신부의 대학 S/W 연구센터 지원사업의 연구 결과임.

1. 서 론

과거의 지리 정보 데이터는 각각의 지리정보시스템의 저장 형식에 따라 각각 독립적으로 구축되어 지리 정보 데이터의 사용이 특정 사용자에게만 국한되어 있었다. 그러나 오늘날에는 인터넷뿐만 아니라 무선 PDA등 다양한 분야에도 지리 정보의 응용이

필요하게 되었다. 한편, XML(eXtensible Makeup Language)은 1996년 W3C(World Wide Web Consortium)에서 제안한 것으로, 웹 상에서 구조화된 문서를 전송 가능하도록 설계된 표준화된 텍스트형식이다. 이는 인터넷에서 기존에 사용하던 HTML (HyperText Markup Language)의 한계를 극복하고 SGML(Standard Generalized Markup Language)의 복잡함을 해결하는 방안으로써 HTML에 사용자가 새로운 태그를 정의할 수 있는 기능이 추가되었다 [1-4]. 이러한 XML의 장점을 이용하여 지리 정보를 표현하는 방법이 연구되고 있다[5-7]. 이에 OGC에서는 공간 정보와 비공간 정보를 포함한 지리 정보 데이터를 저장하고 전송하기 위한 XML 기반의 인코딩 표준으로 GML을 발표하였다[8]. GML은 2000년 5월 버전 1.0이 발표되었고, 2001년 2월 버전 2.0이 발표되었다. OGC는 개방형 지리정보시스템 환경을 위해 지리 정보 데이터와 어플리케이션간 표준 인터페이스를 제시하는 것을 추구하였다. 이를 통해 이질적으로 표현된 다양한 형태의 지리 정보를 표준화된 형식으로 표현하여 각기 독립적으로 구축된 지리정보시스템간의 상호 운용성을 제공을 목표로 하고있다[6,7]. 또한 OGC에서는 분산 환경에서 이질적인 지리 정보들간의 상호 운용성을 제공에 대한 검증을 위해 웹 환경에 기반한 웹 매핑 테스트 베드를 제시하였고, 이를 이용한 다양한 연구가 광범위하게 수행되고 있다[9].

본 논문에서는 이질적인 지리 정보들간의 상호 운용성을 제공하기 위하여 GML 문서와 지리정보시스템간의 지리정보를 사상시키는 기법을 제안한다. 이를 위해서는 기존의 지리정보시스템의 지리 정보를 GML 문서로 표현하는 방법에 대한 연구와 GML 문서에 나타나는 지리 정보를 공간데이터베이스시스템으로 저장시키는 과정을 나뉜다. 기존의 지리정보시스템의 지리 정보를 GML 명세[10-12]에 따라 GML 문서로 변환하는 과정은 먼저 지리정보시스템내의 지리 정보 스키마를 기반으로 GML 문서로 변환하기 위한 응용 구조를 나타내는 응용 스키마를 생성한다. 사용자는 질의를 통해 지리정보시스템에 원하는 결과를 요구하게 되고 이에 지리정보시스템은 공간 정보와 비공간 정보를 질의에 대한 결과로 반환하게 된다. 생성된 응용 스키마는 반환된 정보들을 바탕으로 GML 문서를 생성할 때 GML 문서의 구조 정보로써 이용되어진다. 공간 정보에 대한 데이터 타입은 GML의 기하 모델에 대응시키고, 비공간 데이터 타입은 XML에서 제공하는 데이터 타입으로 처리된다. 공간 데이터 타입 중 공간 객체에 대한 주석을 위해 새로운 데이터 타입을 정의하고, SRS(Spatial Reference System)를 통해 공간 데이터들의 좌표계를 일치시킨다. GML 문서의 지리 정보를 공간데이터베이스시스템에 저장하기 위해서는 GML 문서의 규격화가 선행되어야 한다. 이는 DTD(Document Type Definition) 또는 스키마에 유효한 GML 문서임이 보장되어야 함을 말하는 데, GML 문서의 공간데이터를 공간데이터베이스에 저장하기 위해 DTD나 스키마 파일은 GML 문서와 함께 입력되어져야 한다. DTD 또는 스키마를 이용하여 GML 문서의 구조에 대한 분석 작업을 거치면 DOM(Document Object Model) 인터페이스를 통해 해당 GML 문서에 대한 계층적 트리를 생성할 수 있다. 이렇게 파싱된 DOM 트리를 통해 GML 문서내의 공간 데이터에 대한 접근이 가능해지고, 각 공간 데이터를 해당 공간데이터베이스시스템에 저장하기 위한 질의를 생성한다. 이 질의를 통해 GML 문서내의 공간 정보들을 공간데이터베이스로 구축할 수 있다. 따라서 본 논문에서 제안하는 GML 문서와 지리정보시스템간의 지리 정보 사상 기법을 이용하여 기 구축된 지리정보간의 상호 운용성을 제공할 수 있는 프레임워크를 생성할 수 있으며, 이를 통해 웹을 기반으로 하는 통합된 지리정보 서비스를 효율적으로 수행할 수 있다.

본 논문의 구성은 다음과 같다. 제 2 장에서는 관련 연구로 GML과 XML 스키마, 그리고 DOM에 대해 설명한다. 제 3 장에서는 이질적인 공간정보의 상호 운용성을 제공하기 위한 공간데이터베이스와 GML 문서간의 지리데이터 사상 기법에 대해 기술하고, 제 4 장에서는 본 논문에서 제안한 기법을 이용한 시스템 구현에 대해 설명한다. 마지막으로 제 5 장에서는 결론을 내리고 앞으로의 연구 방향을 제시한다.

2. 관련 연구

본 장에서는 이질적인 공간 정보의 상호 운용성을 제공하기 위한 지리 데이터 표현의 핵심인 GML 명

세[8], GML 문서의 접근을 위한 DOM[13], 그리고 GML 문서의 지리 정보 사상을 위한 공간데이터베이스시스템으로 GMS[14]를 소개한다.

2.1 GML

OGC는 개방형 지리정보시스템 환경을 위해 지리 데이터와 지리 정보 처리의 표준 인터페이스에 요구되는 추상 사양과 구현 사양을 포함하는 GML을 제시하였다. GML은 상호 이질적인 지리 정보 데이터에 대한 공간 정보와 비공간 정보를 저장하고 전송하기 위해 XML 기반의 인코딩 표준 명세이다[8,15,16]. 2000년 5월 발표된 GML 1.0은 지리 정보를 XML로 인코딩하기 위한 메커니즘으로 XML DTD에 기반하여 3가지 형태의 프로파일로 표현되었고, 2001년 2월 20일 발표된 GML 2.0에서는 XML 스키마를 기반으로 복잡한 속성과 속성 상호간의 상관 관계를 인코딩할 수 있도록 3가지 기본 스키마를 제공함으로써 보다 유연한 기능을 가지게 되었다. 2002년 4월과 9월에 스키마를 중심으로 수정된 GML 2.1.1, GML 2.1.2가 발표되었으며, 현재 GML 3.0은 드래프트 단계에 있으며. 이러한 GML은 아직까지 공식적인 표준으로 확정되기 이전 단계인 권고안 상태에 있다. GML 명세는 OGC에서 제시한 추상 사양의 단순 속성을 기반으로 객체 모델을 정의하고 있으며, 지리 정보 표현을 위한 세 가지 기하, 속성, XLink (XML Linking Language) 스키마를 제공하고 있다. 기하 스키마는 점, 선, 면과 같은 기하학적인 요소를 나타내고, OGC 단순 속성 모델에 따라 Point, LineString, LinearRing, Polygon, MultiPoint, MultiLineString, MultiPolygon, MultiGeometry 클래스와 관계된 기하 엘리먼트를 제공하며 Curve, Surface, MultiSurface, MultiCurve와 같은 타입은 생략된다. 속성 스키마는 속성 집합 모델을 정의하고 있으며, 속성 타입 정의를 위해 기하 스키마를 포함 엘리먼트를 사용하여 기하 스키마가 포함하고 있는 정의와 선언을 이용가능하게 한다. Xlink 스키마는 Link 속성을 통해 자원 사이에 링크를 생성하여 엘리먼트가 XML 문서에 삽입될 수 있도록 한다. 이러한 세 가지의 기본 스키마를 바탕으로 사용자의 다양한 지리 정보 표현에 대한 요구를 충족시킬 수 있도록 응용 스키마의 작성을 가능하게 한다.

2.2 DOM

웹에서 문서 구조를 액세스하고 조작하는 표준적인 방법으로 W3C에서는 W3C DOM이라는 명세를 제시하였다[13]. DOM은 XML과 HTML을 위한 프로그래밍 API로 언어와 플랫폼에 중립적인 정의이다. 즉 DOM을 구성하는 다른 객체에 대해 인터페이스는 정의하지만, 구현에 대해서는 어떤 명세도 제공하지 않는다. DOM 객체는 개발자가 XML 문서로부터 데이터에 대한 읽기, 변경, 추가, 삭제할 수 있는 인터페이스를 제공한다. DOM은 XML 문서의 문법이 적절하며 제대로 구성되었음을 보장하고, 내부 문서 조작을 단순화한다. 데이터 요소사이의 관계를 표현하는 방법에 있어서 DOM은 객체 지향형 데이터베이스와 관계형 데이터베이스에서 정보를 표현하는 방법과 유사하며, 이는 데이터베이스와 DOM을 사용하는 XML 파일 사이의 정보 이동을 용이하게 한다. 따라서, XML 문서를 읽고 조작할 경우 DOM을 사용하면 다양한 플랫폼 간에 높은 수준의 정보처리 상호 운용을 보장할 수 있다.

그림 1은 XML 문서를 처리하기 위해 DOM 트리 구조로 변환한 모습이다. 이와 같이 트리 형태로 문서가 변환되면 이 문서에 대한 접근 API는 트리에 대한 API가 되므로 프로그래밍을 통해 쉽게 제어할 수 있다.

DOM 명세는 3가지 레벨을 제공하고 있다. DOM 레벨 1은 임의의 구조적 문서를 액세스 할 수 있는 인터페이스에 대한 명세로 DOM 라이브러리는 적어도 이 부분에 대한 기능을 제공하도록 하고 있다. DOM 레벨 2는 DOM 레벨 1의 모든 객체를 포함하고, 스타일 시트를 적용한 객체 모델을 추가 지원하며, 문서에 대한 질의 기능과 이벤트 모델에 대한 정의 기능도 포함한다. DOM 레벨 3은 문서를 읽고 저장하는 방법, 내용 모델을 조작하고 유효성을 검증하는 방법, 새로운 이벤트 형과 인터페이스 추가 등에 대한 명세로 아직 작업 초안 상태에 있다. 특히, DOM 레벨 1에서의 DOM 코어(Core)는 최소한의 객체와 문서를 조작하고 접근하여 사용할 수 있는 인터페이스를 정의한 것으로 파싱(parsing)된 HTML이나 XML문서의 내용이 손실없이 표현할 수 있고, 문서의 계층적 구조를 생성할 수 있으며, 상위 레벨의 API가 쉽게 사용될 수 있도록 확장이 가능하도록 설

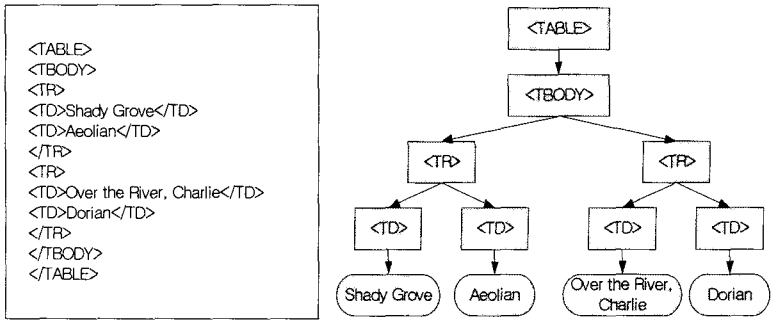

그림 1. XML 문서의 DOM 트리 변환

계되었다. DOM 코어는 API 타입에 따라 상속 관계를 가질 수 있으며, 그림 2는 DOM의 가장 기본적인 타입인 노드타입에 대한 상속 관계이다.

그림 2에서 노드 타입은 기본 타입으로 부모 노드와 정해지지 않은 순서의 자식 노드를 가질 수 있으며, 엘리먼트(element) 타입은 HTML이나 XML 문서의 엘리먼트 객체에 대한 내용을 저장할 수 있고 다수의 속성을 가질 수 있다. DocumentFragment 타입은 문서의 일부분을 가지며, Document 타입은 HTML이나 XML 문서의 루트 노드를 나타낸다. Attr은 엘리먼트의 속성을 의미하며, Text는 태그가 아닌 부분을 저장한다. 이러한 각각의 타입에 대해 getNodeType, getParentNode, getChildNodes 등과 같은 노드 인터페이스와 함께 다른 타입들에 대한 다양한 인터페이스가 존재한다.

2.3 GMS(GEOMania Millennium Server)

GMS는 다수의 사용자가 방대한 공간 데이터 및 비공간 데이터를 효율적으로 저장 및 관리할 수 있는 공간데이터베이스관리시스템으로 사용자가 통합된 데이터를 공유하여 운용 및 갱신할 수 있도록 데이터의 일치성 및 무결성을 유지하고 동일한 작업의 중복을 방지하며 데이터의 보안을 관리한다[14]. GMS는 OGC의 OpenGIS 단순 속성 명세에서 정의하고 있는 점, 선, 면, 다중점, 다중선, 다중면, 기하집합 등의 공간 데이터를 기존의 비공간 데이터와 통합 관리함으로써 사용자 질의를 공간 및 비공간 질의로 구분하여 처리하지 않고 통합된 처리가 가능하다. 또한 대용량의 공간 데이터를 관리하기 위하여 저장관리자에서 별도의 공간 데이터 관리 기능을 지원하여 빠른 탐색 및 질의 처리가 가능하다. 본 논문에서는 공간데이터베이스관리시스템인 GMS를 GML 문서와의 지리 정보 사상에 활용한다. 제안하는 사상 기법에서는 GMS의 공간 정보를 포함하는 레이어에 대한 스키마 정보를 입력으로 GML 문서를 위한 응용 스키마를 생성하고, 생성된 응용 스키마를 기반으로

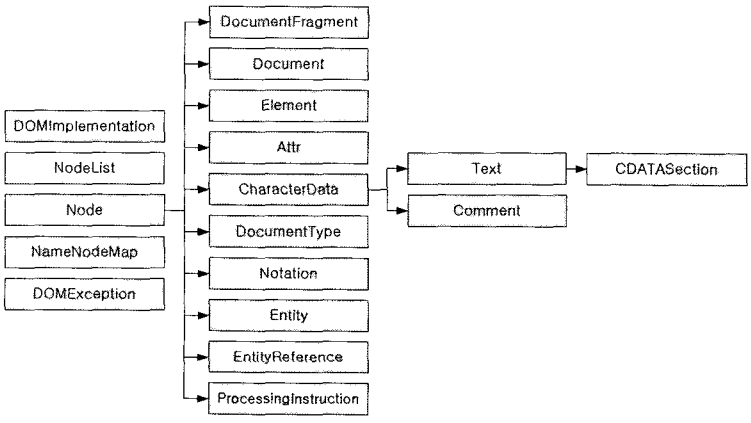

그림 2. DOM을 구성하는 객체에 대한 클래스 계층도

GMS의 공간 레이어를 GML 문서로 변환하는 방법에 대해 설명한다. 또한, GML 문서를 GMS의 공간 데이터베이스로 구축하기 위해 GML 문서에 대한 DTD와 스키마 구조를 제안하고 이를 따르는 GML 문서에 대해 공간 레이어 스키마를 생성하고 데이터베이스를 구축하는 방법에 대해 기술한다.

3. 공간 데이터베이스와 GML 문서간의 지리정보 사상 기법

3.1 공간 데이터베이스의 GML 문서 사상

3.1.1 공간 객체의 GML 표현

공간 데이터베이스의 데이터는 지리 정보를 포함하는 공간 데이터의 속성 정보를 포함하는 비공간 정보로 분류되며, 각각의 데이터들은 GML 문서에서 공간 데이터와 비공간 데이터로 표현된다. 공간 데이터베이스의 비공간 데이터 타입은 XML에서 제공하는 기본적인 타입으로 GML 문서에 사상되고, 공간 데이터 타입은 OGC에서 제시한 단순 속성 명세의 기하 모델로 GML 문서에 사상된다. 그림 3은 OGC에서 제시한 단순 속성 명세의 기하 모델이다.

그림 3에서 명시된 기본 데이터 타입들은 점(Point), 선(LineString), 면(Polygon), 다중점(MultiPoint), 다중선(MultiLineString), 다중면(MultiPolygon) 그리고 기하집합(GeometryCollection) 등이다. 본 논문의 사상기법에서 이용되는 GMS[14]에서는 공간 데이터 타입을 정의할 때 OGC의 기하 모델을 수용하여 구현되었으므로 그림 3의 공간 데이터 타입과 직접 대응된다. 그러나 그림 3에서는 지도의 핵심 정보에 대한 요약이나 주요 관심사에 대한 표시 등을 통해 의사 결정, 영상 분석 그리고 효과적인 지도의

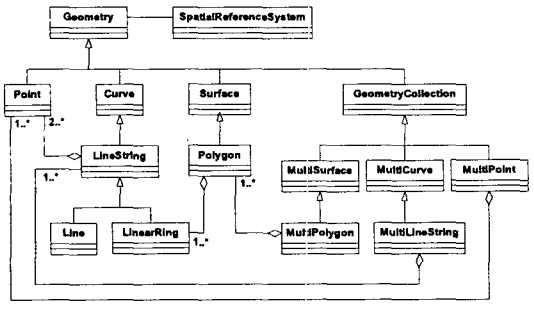

그림 3. OGC 단순 속성 명세의 기하 모델

이해를 돕기 위한 주석에 대한 처리가 제시되지 않고 있다. 따라서 공간 데이터에 대한 주석 처리를 위해 본 논문에서는 AnnotextType을 제시한다.

그림 4는 사용자 정의로 AnnotextType을 표현한 것으로, 주석을 표현할 위치 정보를 가지는 객체의 PointType, 객체에 대한 식별자를 정의하기 위한 정수형의 ID, 객체의 주석 내용을 표시하기 위한 StringType의 LABEL, 그리고 LABEL에 대한 출력 각도를 나타내는 단정도 실수형의 ANGLE로 구성된 Complex Type을 정의하고 있다.

그림 5는 GML 문서상에서 건물명에 대한 AnnotextType이 사용된 예제로 그림 4의 명세에 따라(199057,436819) 좌표를 Point형으로 갖는 ANNOTATES 엘리먼트, 아이디값으로 1을 갖는 ID 엘리먼트, 주석 내용인 '서울역'을 갖는 LABEL 엘리먼트

```
<xsd:complexType name="Anno:AnnotextType">
  <xsd:complexContent>
    <xsd:sequence>
      <xsd:element name="ANNOTATES"
                   type="gml:PointProperty"/>
      <xsd:element name="ID"
                   type="integer" />
      <xsd:element name="LABEL"
                   type="string" />
      <xsd:element name="ANGLE"
                   type="float" />
    </xsd:sequence>
  </xsd:complexContent>
</xsd:complexType>
```

그림 4. AnnotextType 정의

```
<건물ANNOTATION>
  <ANNOTATES>
    <gml:Point>
      <gml:coord>
        <gml:X>199057</gml:X>
        <gml:Y>436819</gml:Y>
      </gml:coord>
    </gml:Point>
  </ANNOTATES>
  <ID>1</ID>
  <LABEL>서울역</LABEL>
  <ANGLE>0.0</ANGLE>
</건물ANNOTATION>
```

그림 5. AnnotextType 사용

그리고 LABEL값의 출력 각도인 0.0을 갖는 ANGLE 엘리먼트에 대한 값을 명시하고 있다.

다음은 OGC의 단순 속성 명세에서 명세한 기하 모델의 각 공간 데이터 타입에 대한 SQL 구문 표현과 GML 인스턴스들을 보여주고 있다. 각 기하 클래스에서는 좌표값을 표현함에 있어 <coord> 엘리먼트와 <coordinate> 엘리먼트 모두 사용 가능하다.

(1) 점 기하 클래스

SQL 구문 표현

```
POINT(10.2 67.2)
```

Point 인스턴스

```
<PointsrsName="http://www.opengis.net/
                    gml/srs/epsg.xml#4326">
  <coordinates>10.2,67.2</coordinates>
</Point>
```

SQL 구문에서 점 기하 클래스는 POINT(…)로 표현되며, 이를 GML 문서에서 표현한 인스턴스는 위에처럼 <Point>…</Point>로 표현된다. srsName은 Point 인스턴스에 대한 좌표계 정보를 나타내고 있으며, <cooridnates>…</cooridnates>에서 좌표값을 나타내고 있다.

(2) 선 기하 클래스

SQL 구문 표현

```
LINESTRING(10 20, 30.56 55, 60 44.57)
```

LineString 인스턴스

```
<LineString srsName="http://www.opengis.net
                    /gml/srs/epsg.xml#4326">
    <coordinates>
      10,20 30.56,55 60,44.57
    </coordinates>
</LineString>
```

선 기하 클래스는 SQL 구문에서 LINESTRING (…)으로 표현되며, 이는 GML 문서에서 <LineString> …</LineString>으로 표현된다. 또한, <cooridnates> …</cooridnates>에서 LineString을 구성하는 좌표값들을 나타내고 있다.

(3) 면 기하 클래스

면 기하 클래스의 SQL 구문표현은 POLYGON (…)이며, 이는 GML 문서에서 <Polygon>… </Polygon>으로 표현된다. <outerBoundaryIs>…

SQL 구문 표현

```
POLYGON((0 0, 90 5, 85 70.5, 10 80),
                (10 20, 30.56 55, 60 44.57))
```

Polygon 인스턴스

```
<Polygon srsName="http://www.opengis.net
                    /gml/srs/epsg.xml#4326">
 <outerBoundaryIs>
   <LinearRing>
     <coordinates>
       0,0 90,5 85,70.5 10,80
     </coordinates>
   </LinearRing>
 </outerBoundaryIs>
 <innerBoundaryIs>
   <LinearRing>
    <coordinates>
      10,20 30.56,55 60,44.57
    </coordinates>
   </LinearRing>
 </innerBoundaryIs>
</Polygon>
```

</outerBoundaryIs>는 Polygon 인스턴스의 외곽 LinearRing을 정의하며, <innerBoundaryIs>… </innerBoundaryIs>는 Polygon 인스턴스의 내곽 LinearRing을 정의한다. 여기서 LinearRing은 좌표값들 중에서 시작과 끝점이 만나서 연결되는 데이터 타입이다.

(4) 다중점 기하 클래스

SQL 구문 표현

```
MULTIPOINT((0 0), (90 5), (85 70.5))
```

MultiPoint 인스턴스

```
<MultiPoint srsName="http://www.opengis.net/
                    gml/srs/epsg.xml#4326">
   <pointMember>
     <Point>
       <coordinates>0,0</coordinates>
     </Point>
   </pointMember>
   <pointMember>
     <Point>
       <coordinates>90,5</coordinates>
     </Point>
   </pointMember>
   <pointMember>
     <Point>
       <coordinates>85,70.5</coordinates>
     </Point>
   </pointMember>
</MultiPoint>
```

다중점 기하 클래스는 SQL 구문에서 MULTIPOINT(…)로 표현되며, 이는 GML 문서에서 <MultiPoint><PointMember>…</PointMember>…</MultiPoint>로 표현된다. pointMember 엘리먼트는 다중점을 구성하는 각 점들에 대한 좌표값들이 정의되어 있다.

(5) 다중선 기하 클래스

SQL 구문 표현

```
MULTILINESTRING((0 0, 90 5, 85 70.5, 10 80),
                (10 20, 30.56 55, 60 44.57))
```

MultiLineString 인스턴스

```
<MultiLineString srsName="http://www.opengis.net
                          /gml/srs/epsg.xml#4326">
  <lineStringMember>
    <LineString>
     <coordinates>
       0,0 90,5 85,70.5 10,80
     </coordinates>
    </LineString>
  </lineStringMember>
  <lineStringMember>
    <LineString>
      <coordinates>
        10,20 30.56,55 60,44.57
      </coordinates>
    </LineString>
  </lineStringMember>
</MultiLineString>
```

다중선 기하 클래스의 SQL 구문 표현은 MULTILINESTRING(…)이고, 이는 GML 문서에서 <MultiLineString>…</MultiLineString>으로 표현되며, LineStringMember 엘리먼트에서는 다중선을 구성하는 각각의 LineString을 정의하고 있다.

(6) 다중면 기하 클래스

다중면 기하 클래스의 SQL 구문 표현은 MULTIPOLYGON(…)으로, GML 문서에서 <MultiPolygon>…</MultiPolygon>으로 표기된다. 각 polygonMember 엘리먼트에서는 면 기하 클래스에 대한 인스턴스를 정의하고 있다.

SQL 구문 표현

```
MULTIPOLYGON(((0 0, 90 5, 85 70.5, 10 80)),
                        ((10 20, 30.56 55, 60 44.57)))
```

MultiPolygon 인스턴스

```
<MultiPolygon srsName="http://www.opengis.net
                          /gml/srs/epsg.xml#4326">
 <polygonMember>
   <Polygon>
     <coordinates>
       0,0 90,5 85,70.5 10,80
     </coordinates>
   </Polygon>
 </polygonMember>
 <polygonMember>
   <Polygon>
       <coordinates>
         10,20 30.56,55 60,44.57
       </coordinates>
   </Polygon>
 </polygonMember>
</MultiPolygon>
```

(7) 복합객체 기하 클래스

SQL 구문 표현

```
GEOMETRYCOLLECTION( LINESTRING(10 20,
30.56 55, 60 44.57), POLYGON((0 0, 90 5, 85 70.5,
10 80),(10 20, 30.56 55, 60 44.57)))
```

MultiGeometry 인스턴스

```
<MultiGeometry srsName="http://www.opengis.net
                          /gml/srs/epsg.xml#4326">
  <geometryMember>
    <LineString>
      <coordinates>
        10,20 30.56,55 60,44.57
      </coordinates>
    </LineString>
  </geometryMember>
  <geometryMember>
    <Polygon>
      <outerBoundaryIs>
        <LinearRing>
          <coordinates>
            0,0 90,5 85,70.5 10,80
          </coordinates>
        </LinearRing>
      </outerBoundaryIs>
      <innerBoundaryIs>
        <LinearRing>
         <coordinates>
           10,20 30.56,55 60,44.57
         </coordinates>
        </LinearRing>
      </innerBoundaryIs>
    </Polygon>
  </geometryMember>
</MultiGeometry>
```

복합객체 기하 클래스는 SQL 구문에서 GEOMETRYCOLLECTION(…)으로 표현되고, GML 문서에서는 <MultiGeometry>…</MultiGeometry>으로 표기된다. MultiGeometry 인스턴스는 앞에서 언급한 다양한 기하 클래스들로 구성될 수 있고, 위의 예에서는 멤버로 선과 면 기하 클래스를 인스턴스로 포함하고 있다.

3.1.2 응용 스키마 생성

GML 문서를 구성하기에 앞서 GML 명세에서 기본으로 제공되는 스키마를 이용하여 서비스의 형태에 따라 사용자가 정의하는 응용 스키마를 생성하여야 하며, 이는 공간 정보를 기반으로 GML 문서를 생성하는 구조적인 정보를 내포하게 된다.

응용 스키마의 생성은 GML 문서를 구성할 공간정보 및 비공간 정보를 포함하는 공간데이터베이스에서 레이어에 대한 메타 정보를 추출하는 단계와 추출된 메타 정보를 바탕으로 기하, 속성 그리고 Xlink 스키마를 이용하여 응용 스키마를 작성하는 단계로 나뉜다. 공간데이터베이스의 스키마로부터 메타 정보를 추출하는 단계는 공간데이터베이스에서 제공하는 해당 레이어에 대한 정보를 먼저 참조하여 그 정보를 얻어낸다. 레이어에 대한 메타 정보는 레이어 이름, 필드 정보, 레이어 권한 정보, 레이어 타입 정보 그리고 공간 메타 정보 등이다. 레이어 권한 정보는 테이블의 접근 권한 값과 사용자 아이디, 그리고 그룹 아이디로 분류된다. 공간 메타 정보는 레이어에 대한 최소경계사각형 정보, 좌표계 정보, 압축 정보, 줌 팩터 정보 등을 포함하고 있다. 이러한 레이어에 대한 메타 정보를 바탕으로 공간데이터베이스의 레이어 구조를 GML의 스키마 구조로 변환한다.

그림 6은 빌딩 스키마에 대한 UML 다이어그램으로 공간데이터베이스의 빌딩 스키마를 표현하기 위한 방법이다. 여기서 빌딩 스키마는 빌딩 스키마 타입으로 하나의 속성 집합을 가지며, 빌딩 멤버 릴레이션을 사용하여 하나의 속성을 포함하고 있다. 속성 집합은 GML 2.0의 Feature.xsd에서부터 제공되며, 속성 집합에 포함된 하나 이상의 속성들의 정의에 대해 멤버 속성을 사용한다. 속성 멤버 엘리먼트는 하나의 속성 또는 XLink 어트리뷰트를 사용하여 다른 문서에 저장된 속성을 지정할 수 있다. GML 명세는 OGC에서 제공하는 단순 속성을 기반으로 하여 공간 및 비공간 Property를 Feature 단위로 저장하고 있다.

3.1.3 GML 문서 생성

본 절에서는 앞 절에서 언급한 공간 데이터에 대한 기하 모델 및 주석 처리 기법을 이용한 GML 문서 작성에 대해 설명한다. GML의 구조는 최상위 노드인 루트 엘리먼트를 시작으로 GML에서 사용될 XML 네임스페이스[5]와 참조되는 스키마 위치에 대한 속성을 가진다.

그림 7은 건물 레이어에 대한 루트 엘리먼트 <건물Schema>에 대한 정의로 GML 문서에서 기본적으로 사용하는 엘리먼트에 대한 구분자로 'gml'이라는 XML 네임스페이스를 사용하도록 속성을 정의하고 있으며, schemaLocation 속성 값으로 건물 스키마에 대한 응용 스키마 정보를 포함하는 건물.xsd의 위치를 나타낸다. 그림 7에서의 루트 엘리먼트는 하위 엘리먼트로 최소경계사각형 영역을 갖는다. 최소경계사각형은 GML 문서에 나타나는 모든 공간 객체

```
<건물Schema xmlns:gml=http://www.opengis.net/gml
    xmlns:xlink="http://www.w3.org/1999/xlink/"
    xmlns:xsi="http://wwww.w3.org/2001
                /XMLSchema-instance"
    xsi:schemaLocation="file:/// c:\xml\\건물.xsd">
```

그림 7. 루트 엘리먼트에 대한 GML 문서

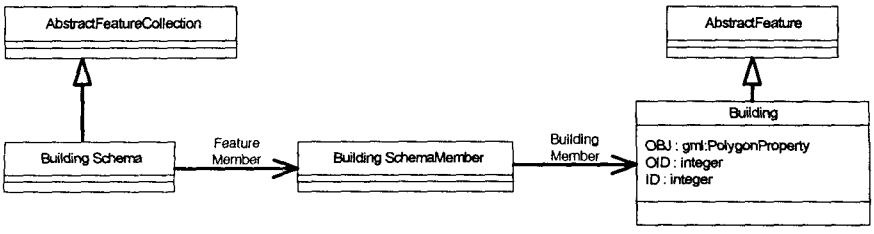

그림 6. 빌딩 스키마의 UML 다이어그램

들을 포함하는 영역으로, 최소경계사각형을 인코딩하기 위해 Box 엘리먼트를 사용한다.

그림 8은 건물 레이어에 대한 최소경계사각형에 대한 GML 표현으로 Box 엘리먼트를 표현한 것이다. 그림 8에서 나타난 SRS(Spatial Reference System)는 지도에 그려질 기하 객체들의 좌표들을 실제 지구의 위치와 일치시키는 좌표 변환 시스템이며, 기하 객체들은 자신이 기준으로 하고 있는 SRS가 어떤 것인가에 대한 정보를 가지고 있어야 한다. SRS는 이러한 좌표 변환 시스템에 대한 메소드를 가지고 있다. 이렇게 참조되는 SRS정보는 srsName 속성으로 정의한다. 위 그림에서는 EPSG:4326(European Petroleum Survey Group)을 참조함을 정의한다. 루트 엘리먼트의 또 다른 하위 엘리먼트로 레이어 엘리먼트를 정의한다. 레이어 엘리먼트는 하위 엘리먼트로 해당 레이어에 대한 필드 엘리먼트와 해당 데이터 값을 정의한다. 레이어는 공간 타입과 비 공간 타입으로 구성된 complexType으로 정의한다. 아래의 그림 9는 기하 타입이 Polygon인 건물 레이어에 대한 GML 표현이다.

그림 9에서 건물 레이어는 공간 객체로 면(Polygon) 객체를 가지며, 속성인 OID와 ID는 1이라는 값을 가진다.

```
<gml:boundedBy>
  <gml:Box srsName="http://www.opengis.net
                        /gml/srs/epsg.xml#4326>
    <gml:coord>
      <gml:X>196575</gml:X>
      <gml:Y>435627</gml:Y>
    </gml:coord>
    <gml:coord>
      <gml:X>208821</gml:X>
      <gml:Y>448112</gml:Y>
    </gml:coord>
  </gml:Box>
</gml:boundedBy>
```

그림 8. 지도 영역에 대한 GML 표현

```
<건물>
  <OBJ>
  <gml:Polygon>
    <gml:outerBoundaryIs>
      <gml:LinearRing>
        <gml:coord>
          <gml:X>199047</gml:X>
          <gml:Y>438327</gml:Y>
        </gml:coord>
        <gml:coord>
          <gml:X>199052</gml:X>
          <gml:Y>438322</gml:Y>
        </gml:coord>
        ……
      </gml:LinearRing>
    </gml:outerBoundaryIs>
  </gml:Polygon>
  </OBJ>
  <OID>1</OID>
  <ID>1</ID>
</건물>
```

그림 9. 건물에 대한 GML 표현

3.2 GML 문서의 공간 데이터베이스 사상

GML 문서의 공간 정보를 공간 데이터베이스에 저장하기 위해서는 입력되는 GML 문서가 규격화되어 있어야 한다. 이 규격화의 의미는 기 정의된 DTD 또는 스키마에 유효한 GML 문서임을 보장해야 한다는 것이다. 세부적인 처리 과정을 살펴보면, 이러한 GML 문서를 분석하고 그 결과를 기반으로 데이터 저장을 위한 질의문을 생성하는 과정과 공간 데이터베이스 인터페이스를 통해 실제 데이터베이스에 데이터를 저장하는 과정으로 구성된다.

3.2.1 GML 문서 구조의 규격화

OGC에서 제안하고 있는 GML을 이용한 웹 맵 테스트 베드에서는 기존의 공간 정보를 웹으로 표현하기 위한 구조를 제안하고 있으나, 공간 정보를 유통하기 위한 교환 포맷으로서의 GML을 응용하는 연구에 대한 내용은 기술되고 있지 않다. 이에 본 절에서는 기존의 공간 데이터베이스에 GML 문서의 공간 정보를 저장하기 위한 방법에서도 선행되어야 할 단계에 대해 설명한다.

GML 문서를 공간 데이터베이스로 저장하기 위해서는 먼저 GML 문서의 구조를 파악해야 한다. 그러나 GML 문서를 표현하기 위해 사용되는 구조 정보는 GML 문서를 생성하는 사용자만큼이나 다양하게 나타날 수 있다. 이 경우 수없이 많은 GML 문서의 구조 정보를 모두 포용하여 GML 문서의 공간 정보를 추출하기란 거의 불가능하다. 따라서 GML 문서를 이용한 공간 정보의 교환을 통해 공간 데이터베이스를 구축하기 위해서는 교환하고자 하는 GML 문서

들에 대한 공통된 구조를 제공하여야 하며, 이 규격에 따라 GML 문서를 생성해야 한다. GML 문서의 구조는 레이어에 대한 정의와 각 필드 정의 그리고 해당 데이터 정보를 갖는 구조로 정의하고 있으며, 하나의 GML 문서에는 하나의 레이어에 대한 정보를 갖도록 정의한다.

(1) DTD 형식을 이용한 규격화

GML 문서의 공간 정보를 공간 데이터베이스로 저장하기 위해 그림 10에서는 DTD 형식을 이용하여 GML 문서의 구조 정보를 정의하고 있다. 즉, GML 문서를 공간 데이터베이스로 구축하고자 할 경우 DTD 형식을 이용하여 GML 문서의 구조를 표현할 경우에는 아래 그림 10의 정의를 이용하여 GML 문서를 생성해야 한다.

그림 10은 GML 문서를 공간 데이터베이스에 구축하기 위한 선행작업으로 DTD 형식을 이용하여 GML 문서를 규격화하는 예를 설명하고 있다. GML 문서의 루트 엘리먼트는 테이블 정의를 위한 엘리먼트인 table-definition가 정의될 수 있으며 이는 오직 한번만 정의될 수 있다. table-definition 엘리먼트는 다시 name 엘리먼트와 column-definition 엘리먼트로 구분되고, name은 문자열의 형태로 테이블의 이름이 되며 column-definitioin은 다시 문자열 타입의 필드 이름인 name 엘리먼트, 필드의 데이터 타입을 문자열의 형태로 나타내는 type 엘리먼트, 그리고 필드에 대한 제약 조건을 문자열로 명시하기 위한 constraint 엘리먼트로 정의된다. 또 row-value 엘리먼트도 루트 엘리먼트로 정의될 수 있으며, 이는 하나 이상의 column-value 엘리먼트들로 정의된다. 이는 하나의 테이블이 다수의 레코드로 구성됨을 의미한다.

```
<!ELEMENT root (table-definition, row-value+)>
<!ELEMENT table-definition
                (name, column-definition+)>
<!ELEMENT column-definition
                (name, type, constraint*)>
<!ELEMENT row-value (column-value+)>
<!ELEMENT column-value (name, value)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT value (#PCDATA)>
<!ELEMENT type (#PCDATA)>
<!ELEMENT constraint (#PCDATA)>
```

그림 10. DTD 형식을 이용한 GML 문서 구조 정의

(2) 스키마 형식을 이용한 규격화

공간 데이터베이스를 구축하기 위한 지리 정보를 포함하고 있는 GML 문서의 구조 정보를 표현하기 위해 스키마 형식을 사용할 경우에도 통일된 구조 정보 표현 형식이 필요하며, 그림 11에서 스키마 형식을 이용한 GML 문서 구조 정보를 정의하는 형식에 대해 설명하고 있다.

```
<xsd:schema
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://165.246.31.93"
  xmlns="http://165.246.31.93"
  elementFormDefault="qualified">
<xsd:element name="ogc-sfsql-table">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="table-definition"
          type="tableDefinitionType"/>
      <xsd:element name="row-value"
          type="rowValueType" minOccurs="1"
          maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:complexType name="col_definitionType">
  <xsd:sequence>
    <xsd:element name="name"
        type="xsd:string"
        minOccurs="1" maxOccurs="1"/>
    <xsd:element name="type"
        type="xsd:string"
        minOccurs="1" maxOccurs="1"/>
    <xsd:element name="constraint"
        type="xsd:string"
        minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="col_valueType">
  <xsd:sequence>
    <xsd:element name="name" type="xsd:string"/>
    <xsd:element name="value" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="tableDefinitionType">
  <xsd:sequence>
    <xsd:element name="name" type="xsd:string"
      minOccurs="1" maxOccurs="1"/>
    <xsd:element name="column-definition"
      type="col_definitionType"
      minOccurs="1" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="rowValueType">
  <xsd:sequence>
    <xsd:element name="column-value"
      type="col_valueType" minOccurs="1"
      maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

그림 11. 스키마 형식을 이용한 GML 문서 구조 정의

그림 11은 그림 10의 DTD 구조를 스키마 형식으로 변환한 예로, 루트 엘리먼트로 ogc-sfsql-table가 정의되고, 루트를 정점으로 tableDefinitionType의 table-definition 엘리먼트와 rowValueType의 row-value 엘리먼트가 정의된다. table-definition 엘리먼트는 문자열형의 name 엘리먼트와 column-definition 엘리먼트를 하위 엘리먼트로 갖는다. column-definition 엘리먼트는 문자열형의 name 엘리먼트, 문자열형의 type 엘리먼트, 그리고 문자열형의 constraint 엘리먼트를 가진다. row-value 엘리먼트는 하위 엘리먼트로 col_valueType 타입의 column-value 엘리먼트를 가지며, column-value 엘리먼트는 하위 엘리먼트로 문자열형의 name 엘리먼트와 value 엘리먼트를 갖는다.

그림 10과 그림 11에서 정의한 GML 문서 구조 정보를 바탕으로 생성된 GML 문서의 예는 그림 12와 같다. 그림 12는 계층적 구조를 바탕으로 생성된 강 레이어를 위한 GML 문서를 보여준다.

그림 12에서 ogc-sfsql-table은 GML 문서의 루트 엘리먼트이며, table-definition 엘리먼트를 통해 정의될 테이블의 구조를 나타내고 있다. 정의될 테이블은 lakes 테이블이며 이 테이블은 shore라는 하나의 필드를 가지면 이 필드의 타입은 POLYGON으로 설정되어 있다. 또한 row-value 엘리먼트는 table-definition 엘리먼트를 통해 정의될 스키마에 삽입될 객체의 정보를 나타내며, shore 필드에 대한 공간 데이터로 POLYGON(…)을 정의하고 있다.

3.2.2 공간데이터베이스 변환

공간데이터베이스 변환은 3.1.1절에서의 규격화된 구조를 통해 생성된 GML 문서의 공간 정보를 분석하여 공간 데이터베이스를 구축하는 과정을 설명한다.

(1) 공간 데이터베이스 스키마 생성

GML 문서는 그림 10에서 설명하고 있는 DTD 또는 그림 11에서 설명하고 있는 스키마를 기반으로 규격화된 형태로 제공되어야 하며, 이러한 규격은 공간 데이터베이스의 스키마 생성을 위해 사용된다. GML 문서의 규격에 대한 분석 과정은 XML 파서를 이용하고, 분석된 GML 문서에서 레이어 정의와 필드 정의로부터 데이터 정의어를 이용하여 공간 데이터베이스의 스키마를 생성한다. 아래 그림 13은 강 스키마를 생성하는 질의를 표현하고 있다.

```
<ogc-sfsql-table>
 <table-definition>
   <name>Rivers</name>
   <column-definition>
     <name>OBJ</name>
     <type>LINESTRING</type>
   </column-definition>
   <column-definition>
     <name>River-Name</name>
     <type>string</type>
   </column-definition>
 </table-definition>
 <row-value>
   <column-value>
     <name>OBJ</name>
     <value>LINESTRING(10 10, 20 10, 30 20,
                       40 20, 50 10, 60 5)
     </value>
   </column-value>
   <column-value>
     <name>River-Name</name>
     <value>Han-River</value>
   </column-value>
 </row-value>
 …
 <row-value>
   <column-value>
     <name>OBJ</name>
     <value>LINESTRING(50 50, 60 50, 70 60,
                       80 60, 90 50, 100 50)
     </value>
   </column-value>
   <column-value>
     <name>River-Name</name>
     <value>NakDong-River</value>
   </column-value>
 </row-value>
</ogc-sfsql-table>
```

그림 12. GML 문서의 예

그림 13에서 2번째 줄의 OBJ는 그림 12의 Rivers에 대한 OBJ 엘리먼트에 대응하는 강 레이어의 공간 데이터 필드로 SPATIAL 타입으로 정의되고, 3~12번째 줄까지는 강 스키마에 대한 메타 정보로 강 레이어에서 사용하게 될 좌표계를 설정하기 위한 COORDINATENAME, 강 레이어의 공간 데이터들의 최소경계사각형(Minimum Boundary Rectangle)을 모두 포함하는 전체 최소 경계 사각형인 MBR, 배정도 실수(DOUBLE)을 갖는 공간 좌표 값에 대해 공간 데이터를 전송할 때 사용되는 압축 형태인 COMPRESSTYPE, COMPRESSTYPE에서 정의된

```
1. CREATE TABLE 강 (
2.    OBJ SPATIAL,
3.    COORDINATENAME='KOREA(MIDDLE)',
4.    MBR[196569.413600,435600.000000,
            208827.957700,448110.299600],
5.    COMPRESS TYPE = LONG,
6.    SCALEX = 0.010000,
7.    SCALEY = 0.010000,
8.    BASEX = 202698.690000,
9.    BASEY = 441855.150000,
10.          COMPRESS OFFSET BITS = 16,
11.          MIN ZOOM = -1.000000,
12.          MAX ZOOM = -1.000000,
13.          River-Name VARCHAR(20)
14. );
```

그림 13. 강 스키마 생성

압축 형태에서 X축과 Y축에 적용할 스케일과 베이스에 대한 값들인 SCALEX, SCALEY, BASEX, 그리고 BASEY, 압축을 수행할 때 사용할 오프셋 비트인 COMPRESS OFFSET BITS, 강 레이어가 특정 확대 배율에서 화면상에 출력되어야 하는 최소와 최대 배율 값을 나타내는 MIN ZOOM과 MAX ZOOM에 대한 정보를 정의하고 있다. 13번째 줄은 그림 12의 River-Name 엘리먼트에 대응하는 표현으로 강 객체에 대한 가변 문자열 타입의 River-Name 필드를 정의하고 있다. 그림 13의 SQL은 GMS의 입력으로 사용되어 강 레이어에 대한 스키마를 생성하게 된다.

(2) 공간 객체 정보 변환

그림 13을 통해 강 레이어에 대한 스키마를 생성한 후, GML 문서내의 공간 객체들을 공간 데이터베이스로 저장하기 위해 SQL의 데이터 조작어를 사용한다. 그림 14에서는 강 객체를 데이터베이스에 삽입하는 구문을 보여주고 있다.

그림 13을 통해 생성된 강 레이어 스키마는 데이터 필드로 OBJ와 River-Name가 정의되었으며, 이에 따라 그림 14의 삽입 구문에서는 OBJ에 대한 필드 값으로 LINESTRING(…)과 River-Name 필드에 대한 문자열 이름 필드 값을 갖는 레코드를 구성할 수 있다. 이를 GMS의 입력으로 삽입 연산을 수행하면 강 레이어에 레코드들이 생성된다.

```
INSERT INTO 강 VALUES ( LINESTRING(10 10,
20 10, 30 20, 40 20, 50 10, 60 5), 'Han-River');

INSERT INTO 강 VALUES ( LINESTRING(50 50,
60 50, 70 60, 80 60, 90 50, 100 50), 'NakDong-River');

INSERT INTO 강 VALUES ( LINESTRING(15 30,
25 30, 35 50, 45 50, 55 30, 65 40), 'YoungSan-River');
```

그림 14. 공간 객체 삽입

4. 구현 및 평가

이 장에서는 본 논문에서 제안한 공간 데이터베이스와 GML 문서간의 지리정보를 사상하는 기법을 적용한 GML 문서관리시스템을 구현한다. 구현 시스템에서는 공간 데이터베이스의 공간 데이터를 GML 문서로 변환하고, GML 문서의 지리정보를 공간 데이터베이스로 구축하는 기능을 포함한다. 또한 GML 문서관리시스템에 대한 평가를 수행한다.

4.1 공간 데이터베이스와 GML의 지리 정보 사상 기법 구현

제안 시스템은 GMS[14]를 기반으로 GML 문서관리시스템의 개발을 통해 공간데이터와 GML 문서간의 변환 방법을 구현하였다. GMS는 공간 및 비공간 데이터에 대한 저장 및 연산을 처리하는 공간데이터베이스관리시스템이다. XML 파서로는 DOM 레벨 1을 지원하는 MSXML을 사용하여 GML 문서의 유효성을 검증한다. GMS기반 GML 문서관리시스템의 구조는 그림 15와 같다.

그림 15에서 GML 문서관리시스템은 공간 데이터베이스 서버 인터페이스, 응용 스키마 생성기, GML

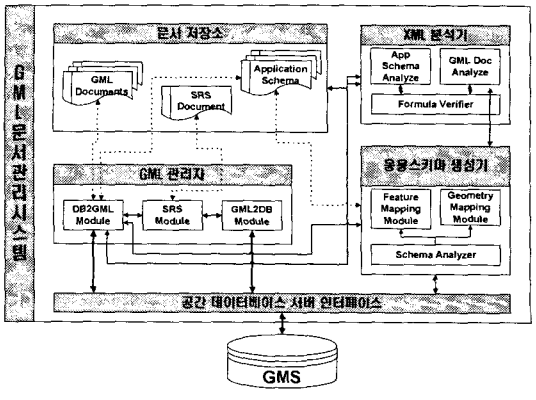

그림 15. GML 문서관리시스템 구조

관리자, 문서 저장소, XML 분석기로 구성된다. 실선은 프로세서의 흐름을 나타내고, 점선은 문서의 참조를 표현하고 있다.

공간 데이터베이스 서버 인터페이스는 공간데이터베이스관리시스템인 GMS와의 연결 및 질의 전달 및 결과 반환을 담당하는 인터페이스이다. 응용 스키마 생성기는 GML 문서를 생성할 때 참조하게 되는 GML 문서의 구조 정보를 담게 될 응용 스키마를 생성하는 역할을 수행한다. Schema Analyzer는 공간 데이터베이스의 레이어 테이블에 대한 스키마 정보를 분석하고, 이렇게 분석된 정보를 이용하여 속성 정보에 대해 Feature Mapping Module, 기하 정보에 대해서는 Geometry Mapping Module을 통해 응용 스키마가 생성된다. GML 관리기는 사용자의 질의에 대해 검색의 결과를 GML 문서로 출력하기 위한 DB2GML Module, 규격화된 GML 문서의 지리정보를 공간 데이터베이스에 저장하기 위한 GML2DB Module, 그리고 좌표 체계 확인 및 수정, 변환을 제공하는 SRS Module로 구성된다. 문서 저장소는 입력 및 출력되는 GML 문서와 GML 문서의 구조 정보를 담고 있는 응용 스키마 문서, 좌표계 정보를 담고 있는 SRS 문서들을 저장한다. XML 분석기는 XML 파서를 이용하여 입출력되는 GML 문서 및 응용 스키마 문서에 대해 정형 검증 및 유효성 검사를 수행한다.

본 논문에서 제안한 공간 데이터베이스의 GML 문서 사상은 다음과 같은 절차를 통해 수행된다. 먼저 사용자 질의를 처리하여 GML 문서를 생성하기 위해서는 생성될 GML 문서에 대한 스키마 정보를 알고 있어야 하므로, 공간 데이터베이스의 레이어에 대한 응용 스키마가 응용 스키마 생성기를 통해 사전에 생성되어 있어야 한다. 이어 OGC의 웹 맵 서버 인터페이스 구현 명세에서 규정하고 있는 URL의 형태 또는 SQL 형태의 사용자 질의는 DB2GML 모듈에 입력되어 공간 데이터베이스에 전달된다. 이에 공간 데이터베이스로부터의 질의 결과는 DB2GML 모듈에서 해당 응용 스키마를 바탕으로 GML 문서를 생성하게 되며, 생성된 GML 문서는 XML 분석기를 통해 정형 및 유효성에 대한 검증 과정을 거쳐 문서 저장소에서 관리되며 최종적으로 사용자에게 전달된다.

또한 GML 문서의 공간 데이터베이스 사상은 3.2.1절에서 설명한 GML 문서 구조의 규격화를 통해 생성된 GML 문서만을 대상으로 한다. 입력되는 GML 문서의 구조 정보는 DTD 또는 스키마의 형태를 취할 수 있으며, 이러한 구조 정보는 공간 데이터베이스의 스키마 생성을 위한 정의어로 GML2DB 모듈에서 변환되어 공간 데이터베이스로 전달되어 공간 레이어를 생성하게 된다. 그리고 GML 문서의 지리 정보들은 데이터 조작어로 변환되어 생성된 공간 레이어의 튜플을 구성하게 된다.

그림 16은 응용스키마 생성기에서 GML 문서 변환시 참조되는 응용 스키마를 생성하는 예를 보여주고 있다.

그림 16에서 응용 스키마 생성기는 공간 데이터베이스와 연결하여 현재 공간 데이터베이스에 존재하는 공간 레이어에 대한 정보를 GMS Table 목록에서 확인할 수 있다. 위 화면은 강 레이어에 대해 생성된 응용 스키마에 대한 내용을 보여주고 있다.

그림 17은 사용자 질의에 대한 GML 문서 생성 화면으로 웹 환경에서 생성된 GML 문서를 출력하고 있다.

그림 17에서는 서울 지역에 대해 구성된 공간 데

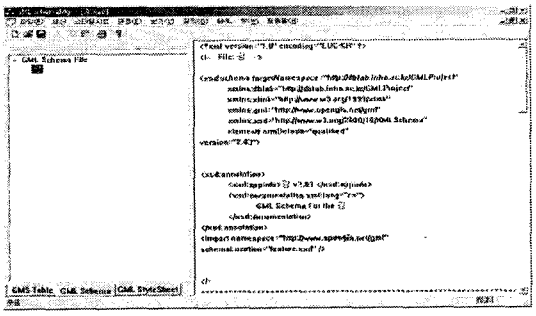

그림 16. 응용 스키마 생성

그림 17. GML 문서 출력

이터베이스에 대해 검색 질의를 수행한 결과 화면으로 서울 지역을 구성하는 레이어들을 GML 문서로 생성하여, 통합된 형태로 웹 브라우저에서 출력하고 있다.

그림 18은 규격화된 GML 문서의 지리 정보를 공간 데이터베이스에 저장하는 과정을 설명하는 화면이다.

그림 18에서는 서울 지역의 강, 건물, 녹지, 도로, 산, 행정경계 등의 지리정보를 담고 있는 GML 문서를 공간 데이터베이스에 저장하고 있다. 이를 위해 각 레이어에 대한 규격화된 구조정보로부터 공간 데이터베이스에 스키마를 정의하고, 해당 GML 문서의 지리정보들을 튜플로 구성하여 GMS에 저장한다.

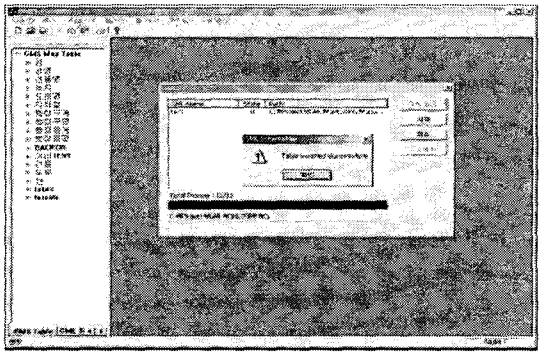

그림 18. GML의 공간데이터베이스 변환

4.2 평가

OGC에서는 지리정보 데이터의 웹 환경에서 상호 운용성을 제공하고 서로 다른 타입의 지리정보의 접근성을 제공하기 위해 웹 맵 서버 인터페이스 구현 명세를 제안하였고, 웹 기반 공간 데이터의 접근 표준을 정의하기 위해 웹 매핑을 제시하고 활용되고 있다. 즉, HTTP상에서 URL 형태의 질의를 입력받아 그 결과로 GIF, JPEG, GML 등의 웹 표준 데이터를 얻도록 한다[9]. 이러한 웹 맵 서버 인터페이스 구현 명세는 웹상에서 지리정보의 서비스하기 위한 처리 단계를 명시하고 있으나, 실제 공간 데이터를 어떻게 GML 문서로 변환할 것인가에 대한 명세와 효과적인 지도의 이해를 돕기 위한 주석(Annotation)에 대한 명세는 제시되지 않고 있다. 또한, GML에서는 상호 운용성을 제공하기 위해 생성된 GML 문서의 지리 정보를 기존의 지리정보시스템에 저장하는 방법에 대한 연구가 제시되지 않고 있다.

이 논문에서 제시하고 있는 공간 데이터베이스와 GML 문서간의 지리 정보 사상 기법은 OpenGIS의 단순 속성 명세의 공간 데이터 타입을 처리하는 공간 데이터베이스관리시스템인 GMS의 공간 데이터를 GML 문서로의 사상 및 주석을 처리하기 위한 명세를 추가하였다. 아울러 GML 문서의 공간데이터베이스 사상을 위해 GML 문서의 규격으로 DTD와 스키마를 정의하였고, 이를 이용하여 공간 데이터베이스 스키마를 구성하였으며, 정의된 스키마 정보를 바탕으로 GML 문서의 지리 정보를 공간 데이터베이스에 저장하였다. 제안 기법은 이질적인 환경의 공간 데이터베이스의 데이터간의 상호 운용성을 제공한다.

5. 결론 및 향후 과제

인터넷의 보급과 정보화 사회의 발전은 지리정보시스템에도 적용되어 이질적인 환경에서 지리 정보를 상호 운용할 수 있는 표준이 요구되었고, 이에 OGC에서는 XML을 기반으로 공간 정보와 비공간 정보를 포함하는 지리 정보를 저장하고 전송하기 위한 인코딩 방법에 대한 명세로 GML을 제시하였다. 또한 GML은 GML 문서를 처리하기 위한 웹 맵 서버 인터페이스 구현 명세를 포함하고 있다. 이에 GML 문서에서의 지리 정보와 기존의 지리정보시스템들이 가지는 지리 정보간의 호환을 위한 연구가 활발히 진행되고 있다.

본 논문에서는 이질적인 지리 정보들의 상호 운용성을 제공하기 위해 GML 문서와 지리정보시스템간의 지리 정보를 사상시키는 기법을 제안하였다. 제안된 지리 정보 사상 기법을 통해 기 구축된 지리 정보간의 상호 운용성을 제공함으로써 분산된 이질 지리 정보들을 통합한 지리 정보 서비스를 위한 프레임워크로 제공될 수 있다.

향후 연구 과제로는 본 연구에서 GML 문서를 파싱하기 위해 XML 파서를 사용하고 있는 데, 이는 공간 데이터의 다차원적인 특성과 공간 데이터간의 상관관계 등의 특성을 효과적으로 반영하지 못함으로 인해 GML 문서를 분석함에 있어 구조적 한계를 드러내고 있으므로 GML 문서를 위한 전용 파서에 대한 연구가 필요하다.

참 고 문 헌

[1] T. Bray and et. Al, "Extensible Markup

Language (XML) 1.0 (Second Edition)", W3C Recommendation, http://www.w3.org/TR/REC-xml, 2000.

[2] David C. Fallside, "XML Schema Part 0: Primer", W3C Recommendation, http://www.w3.org/TR/xmlschema-0/, 2001.

[3] H. S. Thompson, and et. Al, "XML Schema Part 1: Structures", W3C Recommendation, http://www.w3.org/TR/xmlschema-1/, 2001.

[4] P. V. Biron, and A. Malhotra, "XML Schema Part 2: Datatypes", W3C Recommendation, http://www.w3.org/TR/xmlschema-2/, 2001.

[5] T. Bray, and et. Al, "Namespaces in XML", http://www.w3.org/TR/REC-xml-names/, 1999.

[6] Ron Lake, "GML 2.0 Enabling the Geospatial Web", Geospatial Solutions, http://www.opengis.org/pressrm/published/OGCjuly.pdf, 2001.

[7] Galdos Systems Inc., GML Technology, http://www.galdosinc.com/technology-index.html, 2003.

[8] OpenGIS Consortium, Inc., "Geography Markup Language (GML) Implementation Specification v2.1.1", http://www.opengis.net/gml/02-009/GML2-11.html, 2002.

[9] OpenGIS Consortium, Inc., "A Web Mapping Scenario", http://www.opengis.org/wwwmap/98-068.pdf, 1998.

[10] OpenGIS Consortium, Inc., "OpenGIS Simple Features Specification For SQL Revision 1.1", http://www.opengis.org/techno/specs/99-049.pdf, 1999.

[11] OpenGIS Consortium, Inc. "The OpenGIS Abstract Specifiaction Topic 1: Feature Geometry", OpenGIS Project Document Number 99-101.doc, 1999.

[12] OpenGIS Consortium, Inc., "Conformance Test Guidelines for OpenGIS Simple Features Specification for SQL, Revision 1.0", OpenGIS Project Document 98-046r1, 1998.

[13] V. Apparao, and et. Al, "Document Object Model (DOM) Level 1 Specification Version 1.0", W3C Recommendation, http://www.w3.org/TR/REC-DOM-Level-1/, 1998.

[14] S. K. Park and et. Al, "GMS: Spatial Database Management System", Proceedings of the 2003 Joint Sprint Meeting, pp217-224, 2003.

[15] OpenGIS Consortium, Inc. "Geography Markup Language (GML) v1.0", OGC Document Number: 00-029, 2000.

[16] OpenGIS Consortium, Inc. "Geography Markup Language (GML) v2.0", OGC Document Number: 01-029, 2001.

정 원 일

1998년 인하대학교 전자계산공학과 졸업(공학사)
1998년~현재 인하대학교 대학원 전자계산공학과 박사과정

관심분야 : 공간 데이터베이스 클러스터, 이동객체 데이터베이스, XML, GML, LBS 등

배 해 영

1974년 인하대학교 응용물리학과 졸업(공학사)
1978년 연세대학교 대학원 전자계산학과 졸업(공학석사)
1989년 숭실대학교 대학원 전자계산학과 졸업(공학박사)
1985년 Univ. of Houston 객원교수
1992년~1994년 인하대학교 전자계산소 소장
1982년~현재 인하대학교 전자계산공학과 교수
1999년~현재 지능형 GIS 연구센터 소장
2000년~현재 중국 중경우전대학교 대학원 명예교수
관심분야 : 분산 데이터베이스, 공간 데이터베이스, 지리정보시스템, 멀티미디어 데이터베이스 등