Web과 CORBA를 이용한 OpenGIS 정보 저장소 접근

서울대학교 김기홍*·송창빈·유승원*

1. 서 론

GIS(Geographic Information System)는 수치 지도 정보를 근간으로 토지, 자원, 시설물, 교통, 환경, 통계 등 관련 정보를 체계적으로 관리하여 각종 의사 결정에 활용하는 시스템으로, 사회 경제적 관점에서도 GIS 기술이 매우 중요한 것으로 인식되고 있다. 특히 우리나라에서는 국토 개발이 서구에 비해 매우 빠른 속도로 진행되었음에도 불구하고, 이에 대한 체계적 정보 관리 부재로 1995년 대구 가스 폭발 사고와 같이 직접적인 사회 경제적 손실마저 빈번하게 겪게 되었다. 우리 나라의 현실은 미국이나 유럽과 같이 국토 개발이 서서히 진행된 국가보다도 지리 정보의 갱신이 훨씬 빈번하기 때문에, 이들 국가보다 더 선진화된 지리 정보 관리를 필요로 한다.

한편, 인터넷 기술 등 관련 기술의 변화에 영향을 받아, GIS 분야의 패러다임 변화가 최근 시작되었다. 우선 사용자 측면에서 보면, 지리 정보의 활용이 토지, 시설물 등을 관리하는 전문가 그룹에 한정되지 않고 ubiquitous GIS의 개념에 따라 다수의 일반 사용자에게로 빠르게 확산되고 있으며, 인터넷을 통해 공개되는 GIS 서비스의 수도 매우 빠른 속도로 증가하고 있다[1, 2]. 또 기술적으로는, 이러한 사용자 그룹과 접속 가능한 시스템의 다양화로 인해, 지리 정보 데이터 모델 및 프로그래밍 인터페이스의 표준화를 통한 GIS의 컴포넌트(component)화와 상호 연동(interoperability)이 주요 관심사가 되었다.

소프트웨어 시스템의 상호 연동을 위한 산업 표준으로는 CORBA[3]와 OLE/COM[4]이 대표적이며, GIS 분야로 특화된 표준으로는 OGC(OpenGIS Consortium, Inc.)에서 제정중인 OpenGIS(Open Geodata Interoperability Specification)가 있다[5, 6]. OLE/COM이 아직 윈도우즈 운영체제를 중심으로 지원되는데 비해, CORBA는 PC, 유닉스 서버, 메인프레임 등의 다양한 환경에 대해 지원되므로 거시적 통합에 적합하다. CORBA 기술은 크게 1) 소프트웨어 버스에 해당하는 ORB(Object Request Broker), 2) 이벤트 서비스 같은 기본 기능을 제공하는 Object Services, 3) 문서 인쇄와 같이 여러 응용 분야에 공통적인 기능을 정의하는 Common Facilities, 4) 통신, 금융 등의 응용 분야별 공통 기능을 제공하는 Domain Facilities로 나누어진다. 이 분류에 비추어 보면, OpenGIS 표준은 GIS 분야에 대한 Domain Facility로 볼 수 있다.

OpenGIS 표준은 CORBA뿐만 아니라 OLE/COM과 SQL[7] 환경을 대상으로, GIS 분야에 공통적으로 사용되는 형상 정보(geometry), 시공간 좌표계 등에 대한 표준 모델을 정의한다. 최근 OpenGIS 표준을 지원하는 제품들이 속속 발표되고 있는데, 예로 SDE[8], Oracle[9] 등과 같은 SQL 기반의 정보 저장소 제품들이 있다. CORBA나 COM 기반의 구현은 아직 프로토타입 수준이지만, 가까운 장래에는 상용품도 발표될 것으로 기대된다.

본 고에서는 CORBA 기반의 OpenGIS 표

* 학생회원

준 정보 저장소를 웹을 통해 접근하는 시스템의 구조와 구현 사례를 기술한다. 기술된 시스템은 웹 브라우저 상에서 실행되는 자바 애플릿, OpenGIS 표준을 따르는 정보 저장소, 그리고 소프트웨어 버스로서의 ORB(Object Request Broker)로 구성되며, 웹 브라우저만 있으면 어디서나 서비스 이용이 가능하다.

2. OpenGIS 표준

OpenGIS 표준은 지리 정보 소프트웨어를 개발할 때 준수해야 하는 명세서들로 구성되며, 형상 정보, 지리 정보, OpenGIS 서비스, 정보 공동체 등의 14개 주제로 나누어진 추상 명세서와 CORBA, OLE/COM, SQL 세 가지 환경에 대한 구현 명세서가 공개되었다. 현재의 추상 명세서 버전은 3.0이며, 구현 명세서 버전은 1.0이다.

2.1 상호 연동(interoperability)

GIS 구조는 단일 거대 시스템 구조에서 상호 연동이 용이한 분산 컴퓨팅 구조로 발전하고 있다. 이러한 발전에 따라 상호 연동 기술도 1) 한 시스템에서 다른 시스템으로 데이터를 파일 형태로 옮기는(transfer) 방법에서, 2) CORBA나 OLE/COM과 같은 범용 미들웨어를 사용하여 미리 정의된 인터페이스를 통해 데이터를 교환하는 방법으로 발전하는 추세이다. OpenGIS는 CORBA와 OLE/COM을 바탕으로 GIS 분야 고유의 공통 데이터 모델 및 프로그래밍 인터페이스를 정의한다.

1980년대에 개발된 초기 GIS들은 대부분 독자적인 방식으로 공간 데이터를 저장 관리하기 때문에, 다른 GIS가 관리하는 데이터를 이용하기 위해 데이터를 파일 형태로 옮기는 방법을 사용하였다. 1990년대 들어 전문 DBMS 업체의 제품 위에 소규모 공간 데이터 관리 계층을 얹는 방식이 도입되었으나, 시스템간의 데이터 공유를 위해서는 역시 파일을 매개체로 사용할 수밖에 없었다. 데이터 파일을 매개체로 사용하기 위해서는 수많은 데이터 포맷을 서로 변환하기 위한 필터가 필요하였으므로, 데이터 포맷 변환을 용이하게 하기 위한 SDTS(Spatial Data Transfer Standard) 등의 표준이 마련되었다. 데이터를 파일 형태로 옮기는 방식의 문제점 중 하나는 변경된 데이터만 옮기기가 어려워, 전체를 다시 옮기기 전까지 그 동안 변경된 데이터를 이용할 수 없다는 것이다.

최근에는 GIS를 구성하는 데이터 저장 관리 모듈, 지리 정보 분석 모듈, 사용자 인터페이스 모듈 등의 인터페이스를 표준화함으로써, 이 인터페이스를 따르는 컴포넌트(component) 프로그램들을 조합하여 응용 프로그램을 개발하는 구조로 바뀌고 있다. 이런 표준화 작업은 현재 OGC를 통해 추진되고 있으며, 그 결과물이 OpenGIS 산업 표준이다. OpenGIS 표준 인터페이스를 따르는 시스템들끼리는 서로 다른 업체에 의해 개발된 시스템이라도 상호 연동시키기가 용이하다. 이 방식은 다른 시스템이 관리하고 있는 현 시점의 데이터를 이용할 수 있도록 한다.

2.2 추상 명세서

OpenGIS 추상 명세서는 코딩 가능한 실세계 객체의 모델을 제시하며 14개 주제로 나누어진다. 그림 1은 추상 명세서의 14개 주제와 그들 간의 관계를 나타낸다. 그림에서 아래쪽에 위치한 주제는 선으로 연결된 위쪽 주제를 전제한다. 예로 주제 5에서 논의하는 Feature 모델은 주제 1, 2, 3. 8, 11에 정의된 형상, 공간 좌표계, 메타데이터 등의 내용을 활용하여 정의된다. 이 절에서는 앞으로의 논의에 필요한 Feature와 Geometry를 중심으로 기술한다.

실세계의 지리적 사실 또는 현상은 일반적으

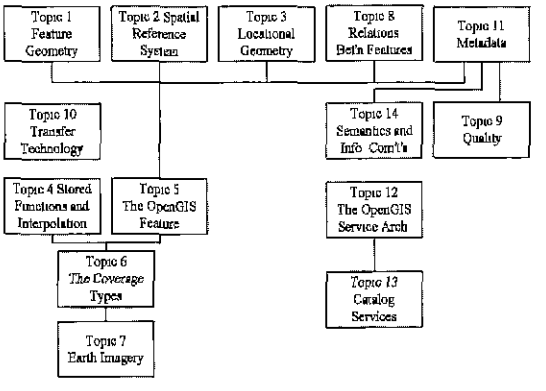

그림 1 추상 명세서 14개 주제 간의 관계

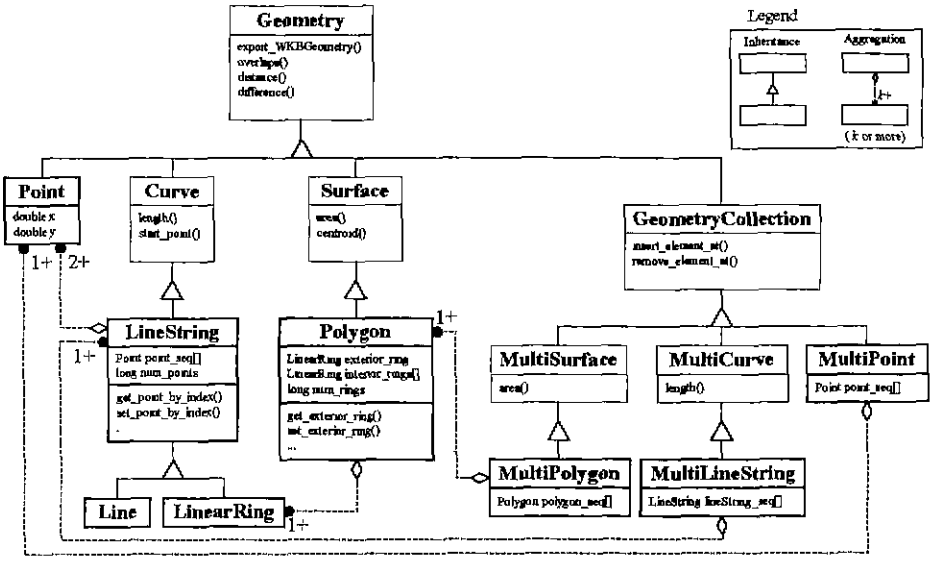

그림 2 Geometry 클래스의 계층 구조

로 Feature, Coverage, 이미지 중 하나로 모델된다. Feature는 공간 및 비공간 속성을 가지는 객체를 지칭하며, 넓은 의미에서는 Coverage와 이미지를 포함한다. Coverage는 GIS 분야에서 널리 사용되는 개념으로, 하위 타입에는 격자, TIN(Triangulated Irregular Network), DEM(Digital Evaluation Model) 등이 있다. 이미지는 래스터(raster)라는 이름으로도 알려져 있으며, Coverage의 특수한 경우이다. 예로, 도로망의 경우 각 도로 구간을 Feature로 모델하는 것이 자연스러우며, 고도 정보는 TIN, DEM, 이미지 등의 형태로 모델할 수 있다. 이들 각각은 주제 5, 6, 7에서 기술한다.

상호 연동을 위해서는 한 시스템에서 정의한 Feature 타입을 다른 시스템에서도 이해할 수 있어야 한다. 이를 위해 속성 타입의 표준화가 필요한데, 주제 1에서 공간 속성으로 사용되는 Point, LineString, Polygon과 같은 표준 Geometry 타입을 정의하고, 비공간 속성은 CORBA, OLE/COM, SQL 등에 정의된 바를 따른다. 한편, 공간 속성의 좌표계나 거리의 단위가 다를 수 있기 때문에, 주제 2에서 시공간 좌표계에 대한 모델을, 주제 3에서 서로 다른 두 좌표계를 매핑하는 구조를 정의한다. 예로, 직교 좌표계, 지도 투영에 사용되는 UTM(Universal Transverse Mercator) 좌표계, GPS에서 사용되는 WGS84 좌표계 등이 있으며, 같은 직교 좌표계라도 축의 방향이나 원점의 지도상 위치가 다를 수 있다.

2.3 Geometry와 Feature

구현 명세서는 추상 명세서를 CORBA, OLE/COM, SQL 환경으로 매핑한 것이며, OpenGIS Simple Feature Spec. for CORBA와 같은 이름을 가지는 세 가지 명세서가 공개되었다. 이들을 줄여서 SFCORBA, SFCOM 등과 같이 부르기도 한다. 구현 명세서 1.0 버전은 Feature, Geometry, Spatial Reference System의 세 부분으로 이루어지며, 이 절에서는 SFCORBA에 정의된 Geometry와 Feature를 중심으로 기술한다.

그림 2는 Geometry 클래스의 계층 구조를 나타낸 것이다. 얇은 선으로 표시된 Geometry, Curve 등은 추상 클래스이다. 베이스 클래스인 Geometry를 상속하여 Point, Curve, Surface, GeometryCollection이 정의된다. Curve를 상속하여 LineString, Line, LinearRing이 정의되며, 같은 방식으로 Surface와 GeometryCollection의 자손 타입도 정의된다. LineString은 polyline으로도 알려져 있는 여러 개의 연결된 선분의 모임을 나타내며, Line은 선분을,

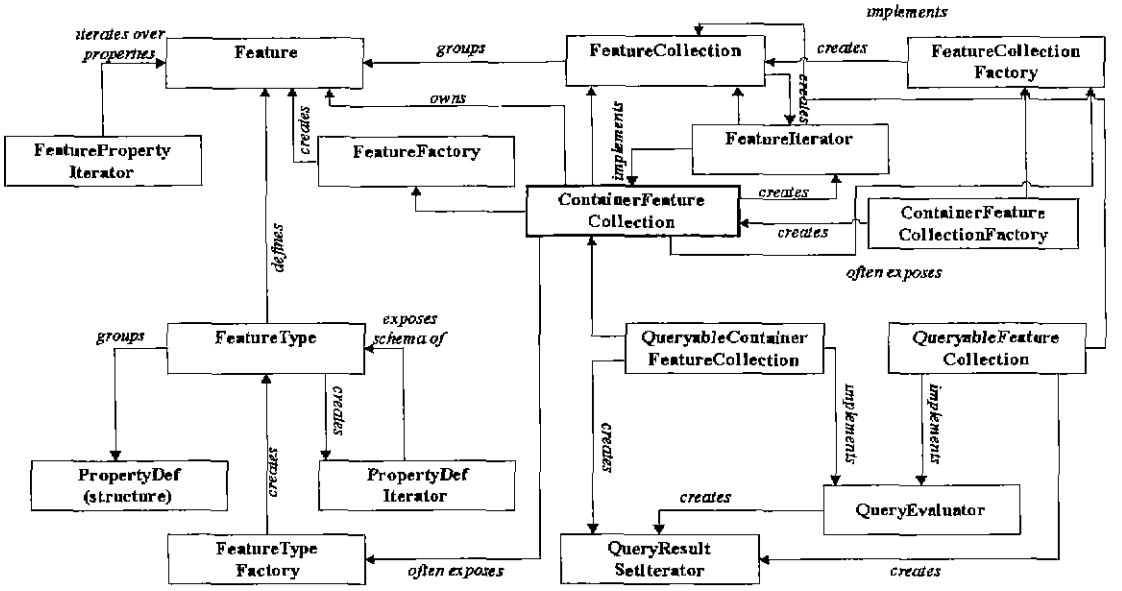

그림 3 OpenGIS Feature 모델

LinearRing은 양 끝점이 동일한 LineString을 나타낸다. 자식 타입이 하나밖에 없는 Curve, Surface 등이 존재하는 이유는 추후 BSplineCurve 등을 추가할 수 있도록 하기 위함이다.

그림 3은 SFCORBA에 정의된 Feature 모델을 타입간의 관계와 함께 나타낸 것이다. Feature, FeatureType, FeatureCollection을 중심으로 Iterator와 Factory 클래스들이 정의된다. 예로, FeatureFactory 타입은 Feature 객체를 생성하기 위한 인터페이스를 제공하며, FeaturePropertyIterator 타입은 Feature의 속성을 스캔하는 인터페이스를 제공한다.

FeatureFactory 인터페이스의 구현은 ContainerFeatureCollection에서 이루어지며, 이 타입에 대한 Factory 타입도 정의된다. 그밖에 질의 인터페이스를 제공하는 QueryEvaluator가 있으며, 이를 구현하는 두 가지 타입과 질의 결과에 대한 Iterator 타입도 정의된다.

Feature는 식별자를 가지는 지속 객체로써, 그림 4와 같이 자신의 타입을 얻거나 속성 정보를 읽고 쓸 수 있는 인터페이스를 가진다. FeatureType은 Feature의 타입 이름, 각 속성의 타입과 이름, 조상 타입 및 후손 타입 정보 등을 제공하는 메타 객체이다. 참고로, 사용자 정의 타입은 Feature를 상속하여 정의할 수 있다. FeatureCollection은 Feature 객체에 대한 참조를 순차적으로 저장 관리하며, Feature를 상속하여 정의된다.

```
interface Feature {
  readonly attribute FeatureType
      feature_type;
  Geometry get_geometry( ... );
  any get_property(in Istring name);
  void set_property(in Istring name,
      in any value);
  void destroy();
  ......;
};
```

그림 4 IDL로 정의된 Feature 인터페이스

ContainerFeatureCollection 객체는 Feature 객체를 생성하거나 이미 만들어진 객체를 바인딩하기 위한 인터페이스를 제공한다. 즉, 클라이언트에서 최초로 바인딩하는 객체로써, 그 이름이 미리 알려져 있어야 한다. 나머지 Feature 객체들은 질의를 통하거나 컬렉션 객체를 스캔하여 바인딩할 수 있다. ContainerFeatureCollection 객체는 시스템에 따라서 하나만 존재할 수도 있고, 여러 개 존재할 수도

할 수 있는 Factory를 지원할 수도 있다.

3. 클라이언트 프로그램 예

그림 5는 웹 브라우저 상에서 실행되는 자바 애플릿에서 SFCORBA 인터페이스를 지원하는 정보 저장소에 접근하는 구조이다. OpenGIS CORBA 서버는 SFCORBA를 구현한 정보 저장소이다. 클라이언트는 서버 데이터를 OpenGIS Feature 타입이나 사용자 정의 타입으로 바인딩할 수 있다. 사용자 타입은 Feature 타입을 상속하는 것이 보통이다. Feature 타입으로 바인딩할 경우는 get_/set_property()와 같은 범용 인터페이스를 이용하여 속성을 읽고 쓸 수 있다. 사용자 정의 타입으로 바인딩할 경우는 사용자 타입에 정의된 전용 인터페이스를 통해 속성을 읽고 쓰거나 메소드를 호출한다. 전자는 사용자 타입과 무관하게 사용할 수 있는 Feature 브라우저와 같은 클라이언트에 적합하다.

그림 6은 Feature 타입으로 바인딩된 서버측 객체의 속성을 디스플레이하는 자바 코드이다.

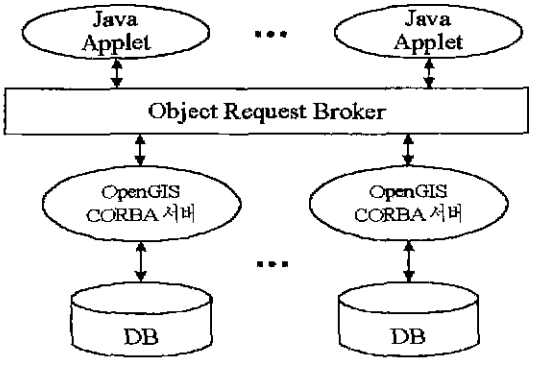

그림 5 웹과 CORBA를 이용한 OpenGIS 정보 저장소 접근 구조

다음은 각 행에 대한 설명이다. 행 1에서 4는 변수를 정의한다. 행 5는 서버측 Feature 객체에 바인딩된 featureRef로부터 해당 FeatureType 객체를 featureTypeRef에 바인딩한다. 참고로 Feature 객체의 바인딩은 QueryableFeatureCollection 객체에 대한 질의 등을 통한다. 행 6과 7은 바인딩된 FeatureType 객체로부터 속성의 이름과 타입 코드 정보를 가지는 PropertyDef 객체의 집합을 얻는다. 행 9와 10은 바인딩된 Feature 객체에 대해 속성의 이름을 주고 속성값을 얻는다. 참고로, 얻어진 속성값은 실제 값과 타입 코드를 함께 가지는 CORBA의 Any 타입이다. 행 11에서 14는 얻어진 속성값이 Geometry 타입이면 그래픽 윈도우에, 아니면 텍스트 윈도우에 디스플레이한다.

```
Feature featureRef;
FeatureType featureTypeRef;
PropertyDef[] propertyArray;
org.omg.CORBA.Any propertyValue;

featureTypeRef=featureRef.feature_type();
propertyArray=
  featureTypeRef.get_property_def_
    sequence();
for(i=0; i<propertyArray.length; i++) {
  propertyValue =
    featureRef.get_property(propertyArray
      [i].name);
  if(isGeometry(propertyValue))
    ...//display in a graphic window
  else
    ...//display in a text window
}
```

그림 6 범용 클라이언트 프로그램의 예

그림 7은 서버측 데이터의 타입을 미리 알고 있는 경우로, Feature의 타입이 RoadSegment인지 확인한 후 road_name과 center_line이라는 두 가지 속성을 읽는 코드이다. 그림 7(a)는 Feature 타입으로 바인딩한 경우이다. Feature의 get_property() 메소드의 리턴 타입은 CORBA의 Any이므로, 이를 실제 데이터 타입으로 변환해야 한다. String과 같은 시스템 정의 타입에 대해서는 Any의 메소드로 extract_string() 같은 것이 제공된다. LineString과 같은 사용자 정의 타입의 경우, IDL 컴파일러에 의해 자동 생성되는 LineStringHelper 클래스를 통해 타입을 변환할 수 있다. 그림 7(b)는 서버측 객체를 RoadSegment 타입으로 바

는 서버측 객체를 RoadSegment 타입으로 바인딩한 경우이다. IDL로 정의된 인터페이스를 컴파일하면, 속성을 읽고 쓸 수 있는 한 쌍의 함수가 속성마다 생성된다. RoadSegment의 road_name() 메소드는 이렇게 생성된 것이다.

```
Any a;
if (featureRef.feature_type().name()==
  "RoadSegment") {
  a=featureRef.get_property("road_name").
    value();
  roadName=new String(a.extract_string());
  a=featureRef.get_property("center_line").
    value();
  roadCenterLine=LineStringHelper.extract
    (a);
}
```

(a) Feature로 바인딩하는 경우

```
RoadSegment segmentRef;
if(featureRef.feature_type().name()==
  "RoadSegment")
  segmentRef=(RoadSegment) featureRef;

roadName = roadRef.road_name();
roadCenterLine = roadRef.center_line();
```

(b) 사용자 정의 타입으로 바인딩하는 경우

그림 7 전용 클라이언트 프로그램 예

4. SFCORBA 구현 사례

이 절에서는 국가 GIS 기술 개발 사업의 일환으로 서울대학교에서 SFCORBA를 구현한 사례를 간략하게 기술한다. 특정 스키마에 대해 SFCORBA 인터페이스를 하드 코딩한 예는 있지만[10], 임의의 사용자 스키마에 대해 SFCORBA 인터페이스를 지원할 만큼 범용으로 구현한 예는 거의 없다.

서울대학교의 구현은 ORB를 가운데 두고, SFCORBA 인터페이스를 제공하는 OpenGIS CORBA 서버와 캐쉬를 관리하는 클라이언트 모듈로 나누어진다. 이 절에서는 OpenGIS CORBA 서버에 대해서만 기술한다. 구현된 OpenGIS CORBA 서버는 상용 ODBMS를 사용하여 데이터를 저장하며, 데이터베이스에 저장된 사용자 정의 타입에 해당하는 wrapper 클래스를 전처리 방식으로 자동 생성한다. Feature 인터페이스는 이들 wrapper 클래스에서 구현된다.

그림 8은 서울대학교에서 구현한 SFCORBA 클라이언트의 실행 화면이다. 이 클라이언트는

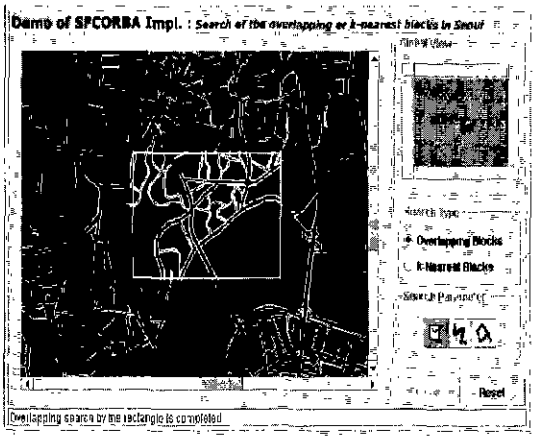

그림 8 SFCORBA 클라이언트 애플릿 예

자바 애플릿 형태로 웹 브라우저 내에서 실행된다. CORBA 서버와의 통신을 위해 OrbixWeb 3.0을 사용하였다. 그림은 가운데 질의 사각형과 겹치는 데이터를 찾은 결과이다. 그림의 오른쪽 위는 전체 영역에서 왼쪽 주화면에 표시된 부분을 나타낸다. 이 경우는 서울시 동호대교 북단 지역을 나타낸다. 그 아래, 공간 중첩 질의와 공간 근접 질의를 선택하는 버튼이 있으며, 중첩 질의의 경우 질의 인자를 사각형, LineString, Polygon 중에서 선택할 수 있다.

그림 9는 데이터베이스에 저장된 db_RoadSegment란 타입을 OpenGIS Feature로 매핑하는 예이다. 이 그림은 클라이언트가 Java 언어를 사용하고 서버가 C++를 사용하는 경우이다. 최종 매핑이 이루어지는 서버측의 RoadSegmentImpl 클래스를 생성하는 과정은 다음과 같다. 먼저, ODBMS에서 제공하는 ODL로 정의된 db_RoadSegment의 스키마를 전처리하여, IDL로 정의된 RoadSegment 인터페이스를 생성한다. 이때 RoadSegment는 미리 정의된 Feature를 상속한다. 참고로, ODL은

IDL 정의는 대부분 1대 1로 쉽게 매핑된다. 이 IDL 정의를 서버와 클라이언트 측에서 각각 컴파일하면 그림 9의 점선 사각형 안에 있는 나머지 클래스들이 자동 생성된다. 마지막으로 wrapper 클래스인 RoadSegmentImpl을 서버 측에서 생성하는데, 이 클래스는 앞에서 db_RoadSegment를 전처리할 때 함께 생성된다. Wrapper 클래스는 Feature 인터페이스에 정의된 get_/set_property() 등의 메소드, 사용자 타입에 정의된 속성을 읽고 쓰기 위한 메소드 등을 구현한다.

5. 결 론

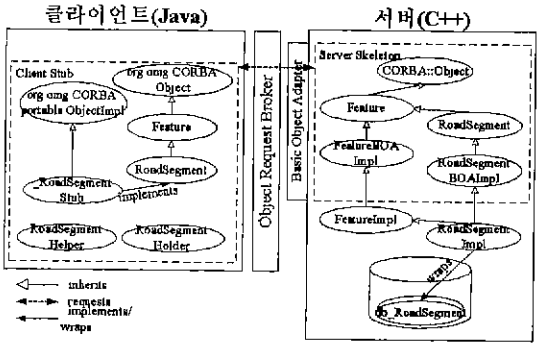

그림 9 OpenGIS 서버 구현 예

본 고에서는 웹과 CORBA를 통해 OpenGIS 정보 저장소에 접근하기 위한 시스템 구조를 설명하고 구현 사례를 살펴보았다. 웹, CORBA, OpenGIS의 세 가지 요소 기술 모두 상호 연동을 위한 국제 표준 또는 산업 표준으로, 정보 시스템 구조에 상당한 영향을 주고 있다. 이중 OpenGIS 표준의 상당 부분은 아직 초안 상태로, 현재 Simple Feature에 관한 구현 명세서만 발표되었고, Simple Coverage 등에 대한 논의가 진행되고 있다. Simple Feature에 대한 구현 명세서는 CORBA, OLE/COM, SQL 등의 세 가지 환경에 대해 따로 존재하는데, SQL 환경에 대한 구현 사례가 가장 많으며, 이미 OGC Conformance 테스트를 통과한 제품도 있다. 반면 CORBA나 COM 환경에서의 범용 구현 사례에 대한 보고는 매우 적다. 그러나, 소프트웨어 시스템의 구조가 CORBA나 COM 환경을 바탕으로 한 분산 컴포넌트 구조로 발전하는 추세이므로, CORBA나 COM 환경에서의 구현이 증가하고 SQL 환경에서 구현된 시스템을 CORBA 환경에 연결하는 예도 증가할 것이다.

참고문헌

[1] Yahoo! Inc., Yahoo! Maps, http://maps.yahoo.com/py/maps.py.

[2] 한국통신, KT114 지도 정보, http://map.kt114.com/.

[3] OMG, *The Common Object Request Broker: Architecture and Specification*, Object Management Group, Framingham, Massachusetts, 1998(http://www.omg.org).

[4] D. Box, *Essential COM*, Addison Wesley, 1998.

[5] K. Buehler and L. McKee, editors, *The OpenGIS Guide: Introduction to Interoperable Geoprocessing*. Technical Report, Open GIS Consortium, Inc. 1996.

[6] OGC Inc., http://www.opengis.org.

[7] J. Melton and A. R. Simon, *Understanding the New SQL: A Complete Guide*, Morgan Kaufmann, 1993.

[8] ESRI Inc., SDE for Oracle Version 3.0.2, 1998.

[9] Oracle Corp., Oracle8 Spatial Cartridge, 1998.

[10] S. Shimada and H. Fukui, "Designing a Mediator for Managing Relationships between Distributed Objects", in *Proceedings of Interoperating GIS*, 1999.

[11] R. G. G. Cattell and D. K. Barry, Ed., *The Object Database Standard: ODMG 2.0*, Morgan Kaufmann, 1997.

김 기 홍

1993 서울대학교 제어계측공학과 학사
1995 서울대학교 제어계측공학과 석사
1995~현재 서울대학교 전기공학부 박사과정
관심분야 : 분산 공간 DB, DB 인덱싱 및 클러스터링, ITS
E-mail : next@kdb.snu.ac.kr

송 창 빈

1996 서울대학교 전기공학부 학사
1998 서울대학교 전기공학부 석사
1999~현재 서울대학교 전기공학부 박사과정
관심분야 : 분산 DB, 공간 DB, 객체지향 DB
E-mail : tsangbi@kdb.snu.ac.kr

유 승 원

1998 서울대학교 전기공학부 학사
1998~현재 서울대학교 전기공학부 석사과정
관심분야 : Interoperable 공간 DB, ITS, 웹 DB
E-mail : mokmon@kdb.snu.ac.kr

'99 주요행사 연간일정표

월 별	주 요 행 사		
	행 사 명	일 시	장 소
3월	Y2K 해결을 위한 호남지역 워크숍	4일(목) 14:00	광주대학교
4월	Y2K 해결을 위한 충청지부 워크숍	16일(금) 14:00	한남대학교
	제26회 임시총회 및 춘계학술발표회	23일(금)~24일(토)	목포대학교
5월	Y2K 해결을 위한 영남지역 워크숍	28일(금) 10:00~12:30	부산대학교
6월	제12회 정보문화의 달 기념 행사	3일(목) 14:00	전북대학교
	Y2K 문제 해결을 위한 심포지움	29일(화) 14:00	과학기술회관
7월	제1차 리눅스 포럼	27일(화)~28일(수)	서울 롯데호텔
8월			
9월			
10월	제26회 정기총회 추계학술발표회	22일(금)~23일(토)	광운대학교
11월			
12월	제17회 정보산업리뷰 심포지움	10일(금) 13:00	코 엑 스
	1999년도 학회 송년회	10일(금) 18:30	무역크럽