개방형 GIS 표준에 따른 오브젝트 웹 시스템 설계

A Design of Open GIS Compliant Object Web

박기호(朴基豪)*, 정재곤(鄭載坤)**
Ki-Ho Park, Jae-Gon Jeong

요 약 지금까지 지리공간 데이터에 대한 접근과 처리에 있어서 상호운용성 보장을 위해 OpenGIS™ 명세에 준하는 다양한 프로토타입 시스템들이 설계되어 왔다. 그러나, 오브젝트 웹 GIS의 관점에서 보았을 때, 실질적인 상호운용성을 보장할 수 있는 운영 시스템이 되기까지는 많은 걸림돌들이 존재한다. 특히, 새로운 아키텍쳐 설계에 있어 핵심 기반 기술 중의 하나라고 할 수 있는 매핑 라이브러리의 상호운용성 측면에서 진행되는 연구는 거의 없는 실정이다. 본 논문은 타 OpenGIS™ 기반 데이터 제공자들과 상호운용될 수 있는 매핑 커널인 OpenViews를 개발하는 데 초점을 두고, 이를 기반으로 한 새로운 차원의 개방형 GIS 시스템 설계를 제시한다. 이를 위해 향후 등장하게 될 여러 시스템들을 ORB를 통해 통합하기 위한 기본적 시스템 요구 사항들을 추출하였으며, 특히 심볼 객체를 분산시킬 수 있는 표준 인터페이스를 제시하였다. 개발된 매핑 커널은 확장성과 유연성을 보장할 수 있도록 100% 순수 자바를 기반으로 하였으며, 잘 알려진 디자인 패턴과 UML을 설계 과정에 도입하였다. OpenViews를 내장한 GIS 애플리케이션 서버는 OpenBroker를 이용하는 데이터 제공자와 함께 서블릿과 EJB기반의 웹 전자지도 출판을 쉽고 빠르게 처리할 수 있도록 하는 개방형 GIS 시스템 아키텍쳐의 핵심 구성 요소이다.

ABSTRACT To meet the interoperability requirements associated with geospatial data access and geoprocessing, much efforts to design prototypical systems conforming to OpenGIS™ specification has been made. With respect to Object Web GIS, however, current internet mapping technology mainly focuses on either developing mapping libraries or client applications regardless of the future needs for interoperability such as an integration of OpenGIS™ standard for CORBA. In this paper, we propose an OpenGIS™ compliant mapping kernel, OpenViews, which is designed to meet those requirements. The kernel of OpenViews encapsulates the process of acquiring geospatial data in the format of OpenGIS™ Geometry through ORB(Object Request Broker). OpenViews, being designed based on well-known design patterns, is a highly extensible in that programmers can easily customize it on the object oriented architecture. The components implemented in OpenViews are CORBA/Java objects, and as such are portable and scalable in a networked environment. A companion package, OpenBroker, is also developed as a portable geoprocessing application server to facilitate the implementation and configuration of server side CORBA objects. It can be used for implementing objects for spatial analysis service which would be independent of legacy spatial database systems in many cases. OpenViews, together with OpenBroker, has been successfully prototyped using the technologies including EJB and servlet as the core components of an Open GIS compliant internet mapping.

키워드 : 오브젝트 웹 GIS, 매핑 커널, GIS 애플리케이션 서버, EJB, 개방형 시스템, 웹 전자지도 출판

* 서울대학교 지리학과 조교수
** 서울대학교 지리학과 {khp, jaegon}@snu.ac.kr

1. 서 론

애플리케이션 중심의 시스템은 다양한 데이터 제공자(data provider)와 필요한 경우, 서비스 제공자(service provider)를 통합할 수 있어야 한다[1]. 공간 데이터 처리의 관점에서 보았을 때, 이러한 요구사항은 지리공간 데이터와 지리정보 처리에 관한 상당한 수준의 상호 운용성을 요구한다. 현재 OpenGIS 콘소시엄(OGC[2])은 표준화(standardization)가 이러한 요구사항들을 만족시킬 수 있는 가장 좋은 방식 중의 하나라는 믿음아래 상호운용성(interoperability)을 보장할 수 있도록 명세서(specification)를 개발하고 있다. 기본적으로 표준 명세서의 개발과 배포는 실제 시스템 구현을 위한 기반 라이브러리의 설계 작업이 충분히 뒤따라 주어야 하지만 현실적으로는 벤더에 종속적인 시스템과 이로 인한 기술적 진보의 제약을 포함하고 있다. 다행히 컴포넌트 기술(component technology)은 재사용 가능한 객체들을 이용하여 공통의 기반 라이브러리들을 설계, 구현할 수 있게 함으로써 지금까지의 소프트웨어 개발 전략을 바꾸어 놓고 있으며, 이러한 접근 방식을 견지하는 경우 개방형 GIS 시스템을 설계, 구현하는 데 효율적으로 이용될 수 있는 새로운 핵심 컴포넌트(essential component)들을 구성하고 개방화할 수 있다.

많은 컴퓨터, 시스템 전문가들은 컴퓨팅 환경의 미래가 이러한 컴포넌트 기술에 의해 생성된 분산 객체와 월드 와이드 웹(World Wide Web)이 결합한 오브젝트 웹(Object Web)이 될 것이라고 주장하고 있다[3]. 좀더 나아가 CORBA/Java 오브젝트 웹 환경에서는 오브젝트들을 네트워크 상에 성공적으로 분산시킨 후 다양한 클라이언트 오브젝트와 서버 오브젝트를 결합시키는 유연한 배치(flexible configuration)가 가능하다. 현재 각 기관별, 부서별 응용시스템 개발 위주의 GIS 시스템 구축으로부터 발생하는 이질적 데이터베이스 간의 공유 문제, 사용자 중심의 데이터 통합 문제 등을 해결할 수 있는 대표적 대안으로 제시되고 있는 엔터프라이즈 GIS[4]도 최종적으로는 시스템 또는 데이터 객체들의 분산을 기반으로 하는 오브젝트 웹 GIS를 목표로 한다고 할 수 있다.

따라서, 오브젝트 웹 GIS의 관점에서 보았을 때, 새로이 만들어지는 개방형 GIS 시스템 컴포넌트들이 다양한 배치 하에서 데이터와 서비스에 대한 접근을 보장할 수 있어야 한다는 것은 분명하다. 예를 들어, OpenGIS™ Simple Feature Specification for SQL 명세를 준수하는 공간 데이터베이스에 채널을 열어두는 방식은 단지 데이터에 대한 접근을 보장하는 것으로, 그 시스템이 비록 OpenGIS™ 명세를 준수한다고 할지라도 데이터와 서비스 모두를 요구하는 오브젝트 웹 GIS 관점에서는 충분한 정도의 상호운용성을 지원한다고 할 수 없는 것이다. 특히, 윈도우 기반의 GIS 컴포넌트들은 네트워크를 통해 유닉스 기반의 노드(a node on a network)로 동적 이동이 불가능하며, 이러한 사실은 현재 인터넷 상에 존재하는 리눅스를 포함한 다양한 노드들에 접근 가능할 수 있어야 한다는 이동성(mobility)에 대한 제약을 의미한다. 이것은 차세대 오브젝트 기술이라고 할 수 있는 이동성 오브젝트(mobile object)[5][6] 설계에 대한 제약이기도 하다.

다양한 GIS 시스템 설계 중에서도 웹 기반 GIS 또는 인터넷 GIS는 기본적으로 분산 환경을 이용한다는 점에서 오브젝트 웹을 지향하는 개방형 GIS 시스템의 설계에 있어 가장 중요한 분야로 생각될 수 있다. 특히 자바 기반의 매핑 클라이언트들은 웹 기반 GIS의 주축을 이루고 있다. 그러나, 현재의 웹 매핑 기술(web-mapping technology)은 앞으로의 상호운용성을 적극적으로 고려하여 설계, 구현하고 있지 않으며, 따라서 현재와 같은 시스템 개발 방향을 견지할 경우 차후에 상당한 정도의 컴포넌트 재설계가 필요한 실정이다.

본 논문은 오브젝트 웹 GIS를 고려한 개방형 GIS 시스템을 설계함으로써 차후의 컴포넌트 재설계 부담을 줄일 수 있는 아키텍쳐를 제시하는 데 그 목적이 있다. 이를 위해 먼저 GIS 시스템 관련 객체들을 분산시키는 데 필요한 핵심 컴포넌트라 할 수 있는 매핑 커널(mapping kernel)을 설계, 구현하고 이를 내장한 GIS 애플리케이션 서버를 설계함으로써 유연하고 확장 가능한 시스템을 제시한다. 설계된 시스템은 지리공간 데이터와 서비스를 다루는 GIS 애플리케이션 서버 시장에 있어 어떤 위치(front-office, middle-office, back-office)에서도 고유 역할을 수행할 수 있어야 한다는 점을 고려하였다.

다음 장에서는 빠르게 변화하고 있는 분산 지리정보 처리 환경에 적응하기 위한 몇 가지 요구 사항들을 최신 GIS 기술과 컴퓨팅 환경의 측면에서 살펴본다. 또한 이러한 요구 사항들을 기본 방향으로 하여 오브젝트 웹 GIS 환경에 적응할 수 있는 개방형 시스템을 설계하는 데 필요한 기능적 요구 사항(functional requirement)들을 정의한다. 3장에서는 UML을 이용하여 개방형 GIS 시스템 설계의 기본 축이 되는 매핑 커널의 설계과정을 기술하고 OpenViews의 핵심

구조와 특징을 설명한다. 4장에서는 개발된 매핑 커널을 내장한 GIS 애플리케이션 서버(PASS)의 아키텍처와 주요 컴포넌트들을 기술하며, 개발된 프로토타입 응용 시스템의 몇 가지 실례를 중심으로 그 특징을 설명한다. 마지막 장에서는 설계된 시스템의 한계점과 앞으로의 개발 방향에 대해 설명한다.

2. 오브젝트 웹 GIS와 기능적 요구사항

2.1 분산 객체를 지원하는 매핑 커널

오브젝트 웹 환경은 애플리케이션 구성 객체를 네트워크 상에 적절히 분산시켜 처리할 수 있는 기술을 포함한다. 오브젝트 웹의 기반이 되는 분산 지리정보 처리의 관점에서 보았을 때 GIS 시스템을 운용하는 데 필요한 몇 가지 요구사항이 정의될 수 있다[7]. 그 중에서 세 가지 핵심 사항들은 다음과 같다.

2.1.1 분산(distributed)

네트워크 상의 여러 노드에 분포되어 있을 수 있는 데이터 저장소, 데이터 처리, 사용자 인터액션 등을 포함한다. 예를 들어, 노드 A에 있는 사용자는 노드 B에 있는 데이터를 노드 C로 보내어 처리하고 그 결과를 자신이 있는 노드 A에서 디스플레이할 수 있다.

2.1.2 분할 또는 분리 (disaggregated or decoupled)

특정 벤더(vendor)에 의해 만들어진 단일 시스템(monolithic system)이 여러 벤더들에 의해 만들어진 플러그-앤-플레이(plug-and-play) 컴포넌트로 대체된다. 이 컴포넌트들은 산업표준에 준하는 것으로 서로 상호운용될 수 있도록 설계된다.

2.1.3 상호운용(interoperable)

위의 두 사항들을 만족시키기 위한 전제조건이다.

그림 1은 CORBA, 자바 기반의 오브젝트 웹을 표현한 것이다. 위의 3-tier 구상은 곧 모든 tier가 분산된 상태로 확장 가능하므로 multi-tier 배치에서의 유연성을 보장하는 측면이 강조된다. 특히 GIS 매핑의 관점에서 보면, 이 그림을 통해 어떤 tier에서도 운용될 수 있는 커널이 필요하다는 것을 알 수 있다. 분산 객체를 지원하는 매핑 커널은 시스템을 구성하는 다른 객체들과 표준 인터페이스를 통하여 통신하므로 Open GIS와 같은 관련 표준을 준수하는 것이어야 한다.

표 1은 분산된 지리정보처리 환경에서 매핑이 가능하도록 시스템을 설계하는 경우, 위와 같은 구상을 만족시키는 가능한 대안들 몇 가지를 보여주고 있다. 이 표는 빠르게 변화하는 소프트웨어 기술을 포함하고 있다.

특히, 단일한 지리정보 데이터를 가지고 다양하게 볼 수 있는 멀티 뷰(multiple views)기술은 공간 뷰(geographic views)를 효과적으로 관리하기 위해 필수적이다. 예를 들어, 네트워크 관리 시스템(NMS: Network Management System)을 웹으로 서비스하는 비즈니스 객체의 경우, 지도 서비스를 하는 객체가 대용량 지리정보 데이터를 다수의 클라이언트 객체의 요구에 맞게 처리하여 결과를 보내주어야 하므로 데이터의 중복 사용이 전자지도 출판의 걸림돌이 될 수 있다. 이러한 경우에 애플리케이션 서버에서 멀티 뷰를 사용하면, 동일한 데이터를 재사용함으로써 부하를 줄일 수 있다.

분산이라는 측면은 기존의 단일 GIS 시스템의 각 기능을 분할함으로써 좀더 견고하고 재사용 가능한 컴포넌트들을 만들도록 요구한다. "매핑 커널" 이라는 용어는 따라서 분산된 지리정보 처리 환경에 있어서 어

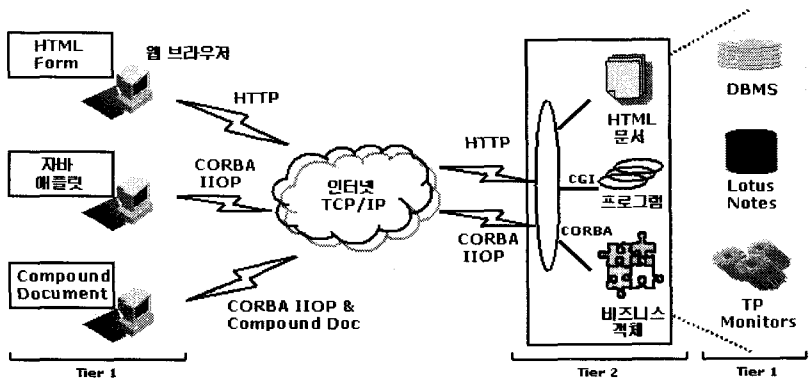

그림 1. CORBA/자바 오브젝트 웹(출처:[2])

표 1. 분산된 지리정보 처리 환경에서 매핑을 위한 가능한 대안들

특 징	가능한 대안	관련 기술
분산 (Distributed)	다중 표현 (Multiple representation) 원격지 매핑 (Mapping at remote site)	멀티 뷰 기술 (Multiple view technology) 미들웨어 기술 (Middleware technology)
분할 (Disaggregated)	매핑 컴포넌트 (Mapping component) 매핑 커널 설계 (Design of Mapping Kernel)	그래픽 프로그래밍 (Graphic Programming) 컴포넌트 아키텍쳐 (Component Architecture) 객체지향 방법론 (Object-Oriented Programming)
분리 (Decoupled)	매핑 커널의 탐색 (Search for a Mapping Kernel) 매핑 컴포넌트로의 이벤트 전달 (Event dispatching to a Mapping Component)	로케이션 서비스 (Location Service) 이벤트 서비스 (Event Service)

느 tier, 어느 플랫폼에서도 운용될 수 있는 이동 가능한 매핑 컴포넌트(portable mapping component)를 의미한다고 할 수 있다. 오브젝트들을 분산된 노드들에 분리시킨다는 것은 각 오브젝트들을 탐색할 수 있는 방법을 요구한다. 특히, 이벤트 서비스나 메시지 서비스와 같이 분리된 오브젝트들을 하나로 볼 수 있도록 만드는 메커니즘이 필요하다.

2.2 관련 표준 검토

공간 객체를 분산시키고 분산된 객체들을 이용하여 지리 정보 처리가 가능하도록 하기 위해서는 표준(standards), 참조 기술(reference technology), 공통 인터페이스(common interface) 등을 정의, 적용해야 한다. 네트워크를 통해 공간적으로 분산되어 있는 사용자들 간의 협력이 가능하도록 연계시키는 방안으로는 현재까지 RPC, DCE, CORBA, DCOM 등의 기술이 제시되어 왔다. 이 중에서 CORBA/자바 기반의 오브젝트 웹 환경에 적용 가능한 IT 기술 표준으로는 OMG의 CORBA와 OpenGIS 컨소시엄의 OGIS가 있다. OMG에서 제시하고 있는 CORBA 미들웨어는 OMA(Object Management Architecture)라는 표준 기술에 근거하고 있으며 핵심이 되는 ORB를 통해 공간 객체들이 상호운용될 수 있도록 하는 표준 명세 및 구현 사양은 OpenGIS 컨소시엄에 의해 이미 개발되어 배포되고 있다.

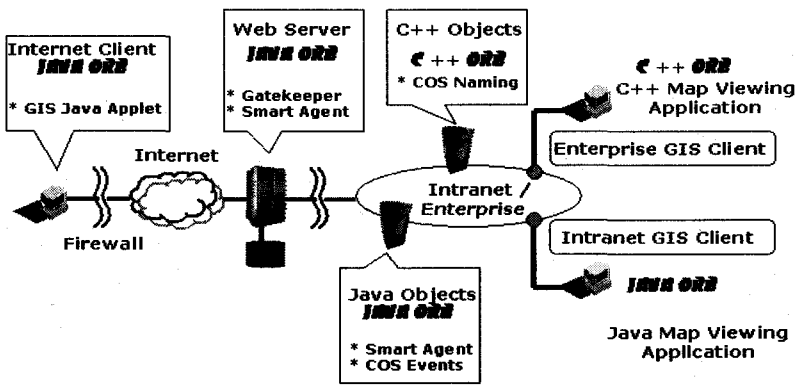

그림 2. CORBA 기반 전자지도 출판 및 관리 전략
(출처: http://www.inprise.com)

그림 2은 ORB 제품 중의 하나인 Visibroker™의 전략을 원용한 것으로 그림의 내용은 효율적인 오브젝트 배치가 가능하다는 점을 강조한다. 이와 같은 전략을 공간 객체에 활용하기 위해서는 기존 시스템의 확장도 필요하지만 오브젝트 웹이라는 환경에 적합한 새로운 차원의 GIS 컴포넌트 개발이 더욱 필요하다고 할 수 있다.

CORBA/자바 오브젝트 웹 운영의 기반이 되는 참조 기술로는 선 마이크로 시스템즈의 자바 개발 도구(Java SDK)와 EJB(Enterprise Java Beans)를 들 수 있다. 참조 기술은 하나의 공통 기술이 주도적으로 사용될 경우 개발자들이 그 기술에 대해 일관된 시각을 가지고 개발할 수 있다는 점에서 실질 표준(de facto standard)으로 작용할 수 있다. 특히 EJB는 서버상에서 자바 플랫폼으로 운영되는 컴포넌트 기술로 객체 컴포넌트 개발에 필요한 스펙들로 구성되어 있으며, 현재 실질적인 객체 표준으로 인식되고 있다. 따라서, 웹 서비스를 위한 애플리케이션 서버와 함께 EJB 컴포넌트를 ORB와 함께 사용하는 전략을 채택하는 것은 다양한 커스터마이징이 필요한 GIS 비즈니스 객체에 적합하다고 할 수 있다.

2.3 기존 시스템의 한계점 및 컴포넌트의 필수 기능

컴포넌트 아키텍쳐를 기반으로 표준 인터페이스를 정의하는 방식은 컴포넌트가 가지는 시스템 확장성, 재사용성 등의 장점이 부각됨으로써 소프트웨어의 상호운용성을 보장하기 위한 가장 일반적인 방법으로 자리잡고 있다. 이러한 컴포넌트 기술 중 현재 많은 GIS 시스템 컴포넌트들이 OLE/COM 모델[8]에 기반하고 있는데 이와 같은 접근 방식은 기존 시스템의 핵심 기능들을 그대로 이식할 수 있게 함으로써 빠른 시스템 설계 및 구현을 보장한다. 그러나, 윈도우(Windows™)가 아닌 이질적 시스템(heterogeneous system)에서도 문제없이 운용될 수 있어야 하는 오브젝트 웹 GIS 환경하의 이동 가능한 매핑 시스템을 설계하는 데에는 부적합하다.

이와는 반대로 GIS 시스템을 구성하는 기본 컴포넌트들이 CORBA와 자바를 기반으로 하는 경우에는 자바 가상 기계가 작동하는 모든 시스템에서 오브젝트들이 운용가능하며, IIOP 채널을 통해 의사 소통할 수 있으므로 이러한 문제가 해결될 수 있다.

새로운 컴포넌트를 설계하는 경우에 필요한 기능 중의 하나로 OpenGIS Geometry 구조체나 Feature 인터페이스를 통해 지리공간 데이터에 접근하는 방법을 캡슐화하는 것을 들 수 있다. 공통된 공간 데이터 접근 방식을 정의하고 캡슐화함으로써 대부분의 비즈니스 객체가 표준 공간 포맷의 데이터를 읽어 들이는 방식을 제공하는 것은 매핑 커널을 설계하는 데 있어 기본적인 고려 사항이기 때문이다.

점차적으로 웹 상에서 지리공간 정보를 서비스해야 하는 경우가 많아지면서 GIS 기술을 사용하는 애플리케이션의 요구사항이 변화하면서 동시에 다양해지고 있다[9]. 따라서, 이렇게 다양한 시스템 개발 요구사항들을 만족시키기 위해서는 매핑 커널과 기타 다른 컴포넌트들과의 분리가 가능해야 하며, 특히 이식성(portability)과 확장성(scalability)의 문제가 해결되어야 한다.

2.4 기능적 요구사항

앞에서 언급한 여러 측면들을 고려하여 다음과 같은 기능적 요구사항들을 도출할 수 있다. 이는 시스템 설계에 있어서의 가이드라인 역할을 하는 것으로 주로 외부 요인들이나 상위 레벨에서의 기능적 요구사항을 의미한다.

2.4.1 OpenGIS 호환성(OpenGIS compliant)

• 데이터 전송 및 서비스 제공을 위한 표준으로서 OpenGIS 명세를 채택하는 경우 표준 인터페이스와의 호환성을 확보하여야 한다.

• ORB를 통한 지리공간 데이터의 획득 과정을 추상화할 필요가 있으며, 데이터 제공자의 데이터베이스 유형(FileBase, RDBMS, OODBMS)에 관계없이 동일한 방식으로 데이터 접근이 가능해야 한다.

• 데이터의 의미 무결성(semantics)을 준수해야 한다. 예를 들어, 여러 데이터 제공자에 채널을 열어두고 데이터를 획득하는 경우 객체의 아이덴티티(Object Identity)를 유지할 수 있는 방안 등을 고려해야 한다.

2.4.2 확장가능성(Extensible)

• 객체지향 설계 방법론(OOAD) 및 잘 알려진 디자인 패턴(well-known design pattern) 또는 UML등을 이용한 투명한 시스템 설계 과정을 거침으로써 설계된 시스템의 유지, 보수가 수월해야 하며, 객체 또는 컴포넌트의 확장이 용이해야 한다.

2.4.3 유연성(Flexible)

• 다양한 애플리케이션이 요구하는 기능에 부합할 수 있도록 비즈니스 객체의 분리가 가능해야 하며 각 컴포넌트의 네트워크 상 배치가 자유로워야 한다.

• 이동가능하고(portable) 내장가능한(embeddable) 컴포넌트로 설계함으로써 네트워크 상의 어떤

노드에서도 동일하게 운용될 수 있다.

• 애플리케이션마다 매핑 오브젝트에 대해 요구하는 다양한 측면들을 수용할 수 있어야 한다. 예를 들어, 협력지원시스템(CSCW)에서 요구하는 브로드케스팅(broadcasting)이나 GIS 애플리케이션 서버에서 발생 가능한 성능 문제를 개선할 수 있는 커스터마이징이 가능해야 한다.

• 다중-tier 방식의 배치에서는 매핑 오브젝트들이 어느 tier에 존재해야 하는가가 미리 정해져 있는 경우가 별로 없으므로 어떠한 유형의 배치도 수용할 수 있어야 한다.

3. 매핑 커널의 설계

3.1 매핑 컴포넌트의 필요성

공간 객체의 디스플레이와 조작은 공간 데이터의 저장 및 관리 기술과 함께 가장 중요한 GIS 시스템 구성 기능들이다. 공간 데이터 제공자는 주로 공간 데이터베이스를 이용하는데 국내에서는 이미 OLE/COM에 기반하여 상당한 작업이 진행되었으므로[10] 구체적인 설명은 생략하였다.

그림 3은 멀티-tier 분산 환경에서의 오브젝트 배치 시나리오를 표현한 것으로 웹 서버를 통한 지도 이미지 서비스 및 ORB를 통한 벡터 스트림 서비스가 가능한 다이어그램이다. A, B, C, D는 분산 객체들 중에서 매핑 커널의 기능을 필요로 하는 것들을 표현한 것으로 모든 tier에서 필수적/선택적으로 사용한다는 것을 알 수 있다. 그림에서 보면 공간 데이터베이스는 데이터의 저장과 관리를 담당하고 있으며, SDE, Oracle Spatial 등과 같은 대용량 데이터베이스가 될 수 있다. Spatial Data Loader는 데이터 로딩을, Accessor는 저장된 데이터에 대한 접근을 담당하는데 이 과정에서 관리를 위한 공간 데이터 디스플레이가 필요하다. Map Viewer는 애플리케이션 기능을, Map Server는 엔터프라이즈 환경에서 필요한 서버 기능을, CORBA Server는 ORB를 통한 통신에 있어 서버 객체가 해야 하는 기능을 담당한다.

이러한 매핑 라이브러리를 설계하기 위해 일반적인 매핑 라이브러리를 고려해 보면 크게 공간 데이터의 통신과 심볼의 통신이 문제가 된다. 공간 데이터의 경우에는 Simple Feature에 관한 OpenGIS 표준 인터페이스를 따르면 되지만 심볼의 경우에는 관련 표준이 아직 제정되지 않아 호환성에 문제가 있다.

본 연구에서는 심볼의 통신을 구현하기 위해 새로운 인터페이스를 제시하는 방법을 취하였다. 이 과정에서 일반적인 매핑 라이브러리가 심볼에 대한 고유 포맷을 가지고 있으며 원격지의 심볼을 접근하기 어렵다는 점을 감안하여 심볼을 분산 객체로 만드는 방식을 고려하였다.

3.2 설계 가이드라인

운영 시스템의 설계에 있어 많은 수의 비즈니스 객체를 다루는 시스템은 설계 가이드라인을 정하는 것이 중요하다. 앞에서 언급한 여러 가지 요구사항을 만족하는 시스템을 설계하기 위한 기본 가이드라인은 다음과 같다.

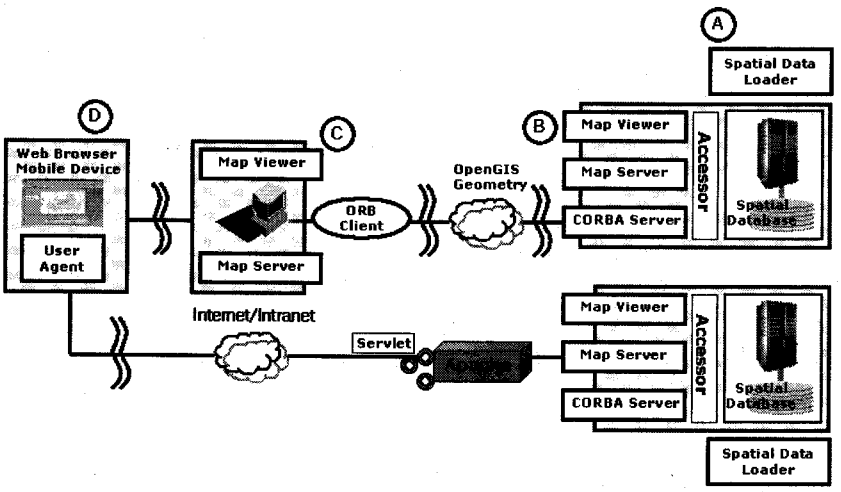

그림 3. 멀티-tier 분산 환경에서의 공간 정보 서비스 시나리오

3.2.1 UML 모델링

UML 방법론은 OMT, OOSE, Booch 방법론[11]을 통합하여 개발된 것으로 시스템의 유지, 관리 측면뿐만 아니라 명확한 시스템 설계를 위해서도 UML을 이용한 시스템 설계가 바람직하다. 본 시스템 설계 및 관리는 RUP(Rational Unified Process) 방법론을 다수 수용하여 이를 통해 차후의 시스템 유지, 관리를 용이하게 하였다.

3.2.2 디자인 패턴

잘 알려진 디자인 패턴을 사용하는 경우 시스템 아키텍처가 좀더 단순하고 간결해지는 장점이 있다[12]. 또한 코드 분석 및 시스템 리엔지니어링(reengineering)의 관점에서도 유용하다. 본 시스템 설계를 위해 20여 가지의 디자인 패턴이 사용되었다.

3.2.3 100% 순수 자바

오브젝트 웹 GIS를 표방하는 경우 100% 순수 자바 기반의 시스템이 큰 장점을 지닐 수 있다. 데이터베이스에서부터 매핑 라이브러리에 이르기까지 100% 순수 자바로 개발되는 경우 모든 시스템 관련 오브젝트들이 네트워크 상에서 이동 가능하며, 이를 통해 다양한 애플리케이션의 요구사항을 만족시킬 수 있다.

3.2.4 CORBA

오브젝트 웹 GIS의 관점에서는 CORBA 기반의 구현이 다른 구현 명세에 비해 차별적인 특징을 가지고 있다. ORB 상의 공간 데이터 통신을 위한 표준으로는 OpenGIS 명세를 사용한다.

3.2.5 OpenGIS 명세의 확장

OpenGIS 명세는 계속 변화하고 있다. 이는 GIS 시스템이 가지는 여러 측면들을 동시에 모두 만족시킬 수 없으므로 당연한 결과이기도 하다. 따라서, 명세에서 충분히 정의하고 있지 못하거나 기존 시스템(legacy system)의 특징을 그대로 적용하는 부분들은 새로 정의하거나 확장하는 방법을 취하였다.

3.3 핵심 아키텍처

개발된 매핑 커널(OpenViews)의 핵심 아키텍처는 ESRI 사의 MapObjects의 것과 유사하다. 이는 매핑을 담당하던 기존의 프로그래머들이 MapObjects의 API에 익숙해 있다는 사실과 안정적으로 운영되는 기존의 시스템과 기능적인 면에서 벤치마킹이 가능하다는 장점을 지니고 있다. 그림 4은 OpenViews의 핵심 클래스들과 MapObjects의 API를 비교한 것이다.

OpenViews는 MVC(Model-View-Controller) 아키텍처를 수용하였으며, 패턴을 쉽게 인지할 수 있도록 클래스 명명을 되도록이면 패턴에 맞추도록 하였다. GraphicView는 GraphicModel에서 처리한 데이터를 디스플레이하는 부분으로 실제 매핑이 이루어지며 앞에서 설명된 멀티 뷰가 가능하도록 설계되었다. 각각의 공간 데이터들은 ShapeObject 인스턴스로 표현되며, 심볼 객체를 통해 캔버스 상에 디스플레이되도록 하였다. ShapeSymbol은 각각의 ShapeObject 유형에 맞도록 정의되었으며, ShapeObject와 ShapeSymbol 모두 사용자에 의해 확장 가능하도록 설계되었다. 예를 들어, 서울의 행정구역 경계 레이어를 표현하는 객체들은 PolygonShape로부터 상속을 받아 만들어진 클래스로부터 생성될 수 있다.

그림 5은 OpenViews에서 사용하는 기본 공간 객

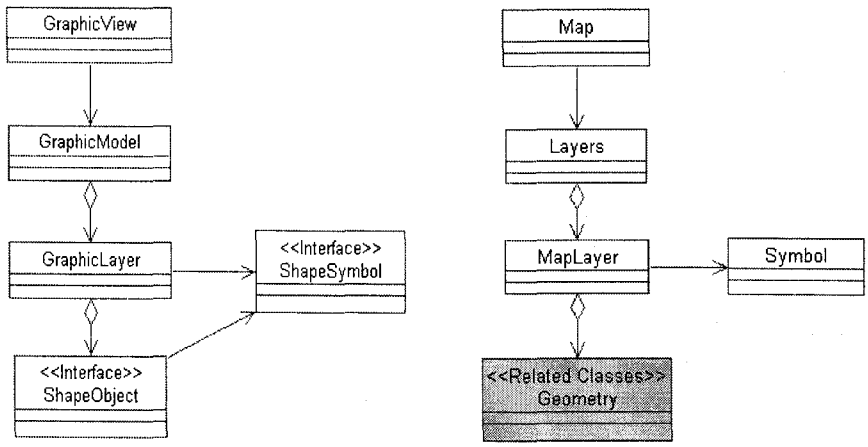

그림 4. 핵심 모델의 비교 (OpenViews 클래스(왼쪽), MapObjects API(오른쪽))

체인 ShapeObject의 정의 클래스와 관련된 시스템 클래스들을 보여주고 있다. 이 클래스들은 OpenGIS 명세에서 정의하고 있는 공간 객체들과 일대일로 매핑될 수 있으며 그래픽 디스플레이를 위한 객체들을 부가적으로 정의하고 있다.

3.4 확장 지리공간 데이터 인터페이스

OpenGIS 명세에 정의된 Geometry를 사용하는 경우 공간 객체의 검색을 위해 공간 데이터베이스에 저장된 각각의 공간 객체에 대해 순차적인 검색(enumeration)을 위한 메소드 호출이 일어나므로 데이터 송수신이 지연되는 문제가 생길 수 있다.

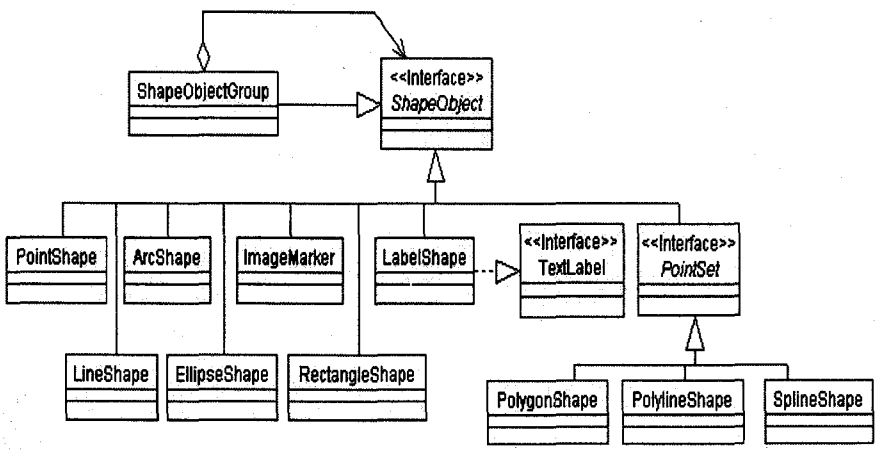

그림 5. ShapeObject 인터페이스와 기본 시스템 클래스 계층도

그림 6은 서울 동경계를 표현하는 공간 객체들을 매핑하는 경우에 사용자가 새로 클래스와 심볼을 정의하는 예를 보여주고 있다. 특정 도메인에 맞는 디스플레이를 위해서는 앞에서 정의된 기본 클래스(base class)들을 사용하는 방식과 함께 다른 객체들과 구별될 수 있는 사용자 정의 클래스를 정의하는 방식을 사용할 수 있다.

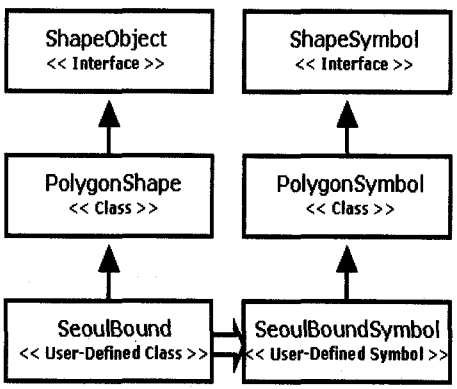

그림 6. 사용자 클래스와 사용자 심볼의 정의 예 (서울 동 경계 오브젝트)

이와 같은 문제를 고려하여 기본적인 데이터 송수신에는 OpenGIS Feature 모델을 준수하고 많은 공간 객체를 포함하는 레이어를 획득하는 경우에는 아래와 같은 새로운 인터페이스를 정의하여 사용하였다. 이 인터페이스는 GeoDataConnection 클래스를 정의하고 있는데 이 클래스는 데이터베이스의 유형에 관계없이 GeoDataSet이라는 동일한 데이터 포맷으로 지리공간 데이터를 넘겨주도록 설계되었다. 즉 GeoDataSet 클래스는 OpenGIS Geometry에 기반하고 있으면서 서버 객체가 어떤 데이터베이스와 링크되어 있는지에 관계없이 동일한 방식으로 공간 데이터에 접근할 수 있는 메커니즘을 제공한다.

```
interface OpenDataProvider {
  exception DataProviderException {string why;};
  // the name of the provider
  void getName(out wstring name);
  // set DataBase in case of Data Provider for
  shapefile
  void setDataBase(in wstring data_base);
  // list GeoDataSets
  void listGeoDataSet(out StringSeq names);
```

```
// get GeoDataSet
void getGeoDataSet(in wstring name, out Open
        GeoDataSet dataset)
        raises (DataProviderException);
// query feature by geometry
void queryByGeometry(in wstring source, in
        OGIS::WKSGeometry geometry,
        in wstring target, in wstring relation,
        out OpenGeoDataSet dataset) raises
        (DataProviderException);
// query feature by attribute
void queryByAttribute(in wstring statement,
        out OpenGeoDataSet dataset)
        raises (DataProviderException);
};
```

3.5 심볼 서비스 인터페이스

기존의 GIS 시스템들은 나름대로의 심볼 체계를 이용하였으므로 분산 환경에서 표준 인터페이스를 통해 심볼을 공유하는 것이 거의 불가능하다. 그러나, 실제로 심볼을 분산 객체로 만들고 표준 인터페이스를 통해 접근하는 방식을 사용하였을 경우에 시스템 운영에 있어 상당한 효율성과 비용 절감의 효과를 가져올 수 있다. 예를 들어, 국가 수치지도(National Digital Map)를 디스플레이하는 경우 정의된 코드체계에 맞게 심볼을 사용해야 하는데 이 코드 체계가 업데이트 되는 경우 새로운 심볼 체계를 시스템마다 다시 정의해서 사용해야 하는 불편이 있다. 만약 심볼을 인터넷/인트라넷 상에서 접근할 수 있다면 정해진 심볼을 인증 기관이 한번 업데이트하면 모든 시스템이 원하는 시간에 자유롭게 사용할 수 있으므로 심볼 객체에 대한 접근 메커니즘을 정의하는 것은 유지 관리의 측면에서 중요하다.

그러나, 앞에서도 언급된 바와 같이 심볼 인터페이스는 아직 표준으로 정의되지 않았으므로 심볼 서비스를 위해 새로운 인터페이스를 고안하였다. Open Views는 기본적으로 자바 2D API를 확장하여 심볼을 정의하므로 자바 2D API를 이용할 수 있는 심볼 인터페이스를 정의하였다. 정의된 심볼 인터페이스는 다음과 같다.

```
interface SymbolServiceProvider {
    exception ServiceProviderException {string
    why;};
    // the name of the provider
    void getName(out wstring name);
    // list symbols
    void listSymbol(out StringSeq names);
    // get symbol size
    void countSymbol(in wstring name, out
    long count);
    // get OpenSymbol
void getSymbol(in wstring name, out Open
        SymbolSeq symbol)
```

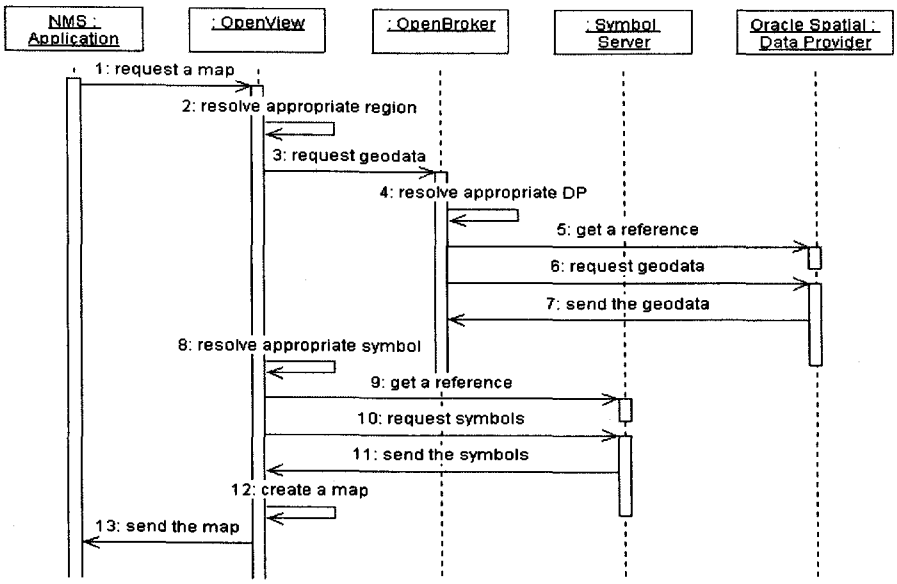

그림 7. OpenViews를 이용한 일반적인 지도 생성 메커니즘

```
                raises (ServiceProviderException);
};
```

그림 7은 NMS(Network Management System) 시스템이 특정 지역의 장비 분포를 알아보기 위해 공간 데이터와 심볼 서비스를 이용하는 시나리오(scenario)를 시퀀스 다이어그램(sequence diagram)으로 표기한 것이다. NMS 애플리케이션을 구성하는 공간 데이터 처리 객체는 OpenViews 인스턴스를 통해 지도 생성을 요구할 수 있다. 실제 지도 생성 작업은 OpenViews 인스턴스가 담당하므로 생성된 인스턴스는 먼저 ORB를 통해 데이터 제공자에게 원하는 스케일과 지역 정보를 전송하여 공간 데이터를 획득하고 원하는 심볼을 심볼 제공자에게 요청하여 그에 해당하는 심볼이 넘겨받게 된다. 최종적으로 이 정보들을 이용하여 디스플레이하고 그 결과를 클라이언트에게 넘겨주는 과정을 거친다.

3.6. OpenViews 구현

설계된 아키텍쳐에 기반하여 구현된 OpenViews 객체들은 기본적으로 ORB와 자바 2 런타임을 필요로 한다. 클라이언트와 서버간 통신은 IIOP로 이루어지며, 데이터 서비스는 OpenBroker*를 이용한 서버 오브젝트가 담당한다.

그림 8은 OpenViews 컴포넌트를 클라이언트 웹 브라우저에 배치했을 경우를 보여주며 데이터 제공자는 오라클 8i의 공간 엔진(ORACLE Spatial)을 데이터 저장소로 사용한다. 오라클의 경우 대부분의 공간 연산(9IM[13] and Spatial Operator)이 데이터베이스 수준에서 이루어지므로 OpenBroker는 단순 미들웨어의 역할만 수행하게 된다.

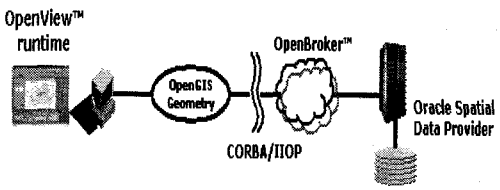

그림 8. 클라이언트 웹 브라우저 상의 OpenViews 컴포넌트 배치

그림 9은 심볼을 분산시켜 놓은 후 가져오기 위한 시스템 배치의 예로 디스크 상에 저장된 심볼은 심볼 서비스 에이전트를 통해 획득될 수 있다. 심볼은 앞에서 기술된 인터페이스를 통해 접근가능하며, 위의 데이터 제공자로부터 가져온 오브젝트에 등록할 수 있다.

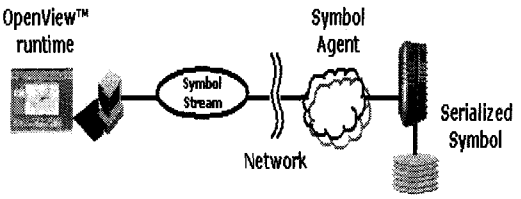

그림 9. 사용자 정의 심볼의 분산과 획득을 위한 시스템 배치

그림 10은 분산되어 있는 심볼 서버 오브젝트로부터 획득한 심볼을 디스플레이한 것으로 오라클 서버 객체로부터 얻어진 서울 지하철 노선도와 또다른 네트워크 노드상에 위치한 심볼 서버 객체로부터 얻어진 서울 지하철 노선도 심볼을 이용하였다.

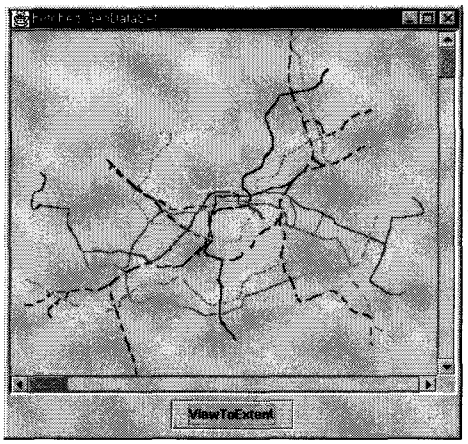

그림 10. 서울 지하철 노선도 심볼의 획득과 디스플레이

4. GIS 애플리케이션 서버 설계

4.1. 애플리케이션 서버 기술

4.1.1 EJB 프레임워크

매핑 커널을 이용하는 실제 시스템의 설계에는 좀더

* CORBA/Java 기반의 서버 오브젝트를 만드는 데 사용되는 공통 기능을 정의한 오브젝트들로 OpenStore와 ShapeFile 데이터 제공자는 이 오브젝트들의 기능을 그대로 이용한다.

다양한 IT기술이 사용될 수 있다. 특히, EJB 프레임워크를 ORB와 함께 사용하는 시스템은 분산 환경에서 비즈니스 객체들을 자유롭게 배치시킬 수 있는 아키텍처를 설계할 수 있도록 한다. EJB는 자바 구현 기술을 기반으로 서버 쪽의 객체 조작이 가능하도록 만든 표준 API[14]로서 웹 기술이 소개되면서부터 웹 상에 정보를 제공할 수 있도록 개발되어 온 다양한 애플리케이션 서버의 표준 역할을 하고 있다. 특히 웹 애플리케이션을 제공하는 시스템의 서버 객체가 유연성을 가짐과 동시에 커스터마이징이 가능하다는 장점을 포함하고 있다.

그림 11은 EJB를 채용하여 웹 상의 전자지도 출판을 구현하는 경우에 가능한 시스템 구성을 보여주고 있다.

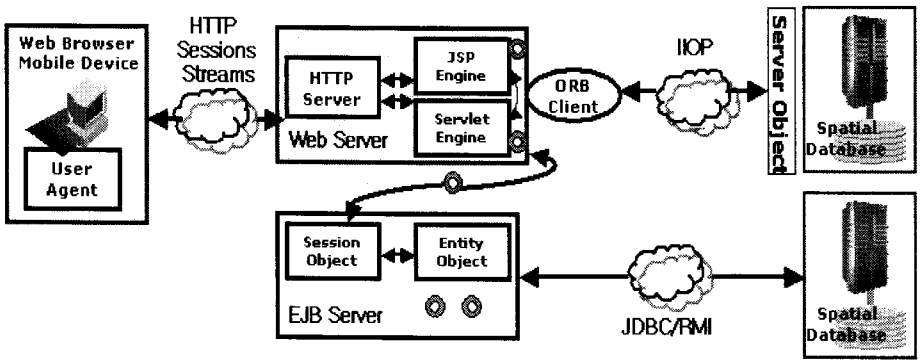

그림 11. EJB와 ORB를 이용한 분산 객체들의 배치와 객체간 통신

클라이언트가 이미지 스트림을 전송받는 경우에는 JSP와 Servlet을 이용한 HTTP 프로토콜이 사용되며, 벡터 스트림을 전송받아 OpenViews 런타임을 클라이언트 웹 브라우저에서 구동시키는 경우에는 ORB에 기반한 IIOP 채널을 사용한다. 그림에서 볼 수 있는 바와 같이 미들웨어와 서버 객체를 분산시킬 수 있을 뿐만 아니라 클라이언트에서 다수의 서버 객체에 접근할 수 있는 채널이 다양하므로 EJB와 ORB를 지원하는 어떤 시스템 구성으로도 구현이 가능하다.

4.1.2 능동적 애플리케이션 서버

현재 C/S 환경에서 분산 환경으로 시스템을 마이그레이션 하는 과정에서 CORBA/COM 기술이 많이 사용되고 있으나, 단순한 2-레이어 시스템은 사용 가능한 서비스(functional server)의 수가 증가하는 데 대해 유연하게 대처하기 어렵다. 이것은 글의 앞부분에서 제시된 요구사항을 수렴하기 어렵다는 것을 의미하므로, 중개자(mediator)를 이용한 새로운 대안이 필요하다[15]. GIS 애플리케이션 서버는 네트워크 상에 존재하는 데이터 제공자와 서비스 제공자에 대한 정보를 제공하는 것은 물론, 매핑을 통해 클라이언트가 원하는 유형의 지도를 출판하는 기능을 하는 중개자라고 정의될 수 있다.

애플리케이션 서버(application server)라는 용어는 서버의 확장 모듈(server extension), 미들웨어 서버(middleware server), 오브젝트 캐쉬(object cache), 객체 서버(object server) 등의 의미로 다양하게 사용되고 있다. 이러한 애플리케이션 서버를 기능적으로 분류하는 경우에는 크게 웹 정보 서버(Web Information Server), 수동적 컴포넌트 서버(Passive Component Server), 능동적 애플리케이션 서버(Active Application Server)로 구분될 수 있다[16]. 웹 정보 서버는 주로 브라우저의 데이터베이스에 접근과 웹 출판(web publishing)에 관심을 두며, 수동적 컴포넌트 서버는 데이터 접근과 컴포넌트에 대한 런타임 프레임워크를 제공하는 데 초점을 둔다. 이에 반해서 능동적 애플리케이션 서버는 백-오피스(back-office) 애플리케이션과 새로운 상위 레벨의 로직을 위한 프레임워크에 관심을 둔다.

GIS 애플리케이션 서버가 오브젝트 웹 환경에서 효율적으로 운용될 수 있도록 하려면, 실제로 stateless한 현재의 웹 서비스를 stateful하게 만들고 오브젝트간 협력을 원활히 만드는 메커니즘의 정의가 필요하므로 GIS 애플리케이션 서버 아키텍처는 다음과 같은 특징들을 수용할 수 있어야 한다.

- **능동성(Active GIS Application Server)**

공간 데이터 처리와 관련된 복잡한 로직들을 모두

수용할 수 있어야 한다. 예를 들어, 공간 통계를 이용한 질병 데이터의 분석 및 디스플레이가 필요할 경우 GIS 애플리케이션 서버는 공간 통계와 관련된 기능을 매핑 작업 이전에 수행할 수 있어야 한다. EJB 프레임워크를 이용하여 이와 같은 비즈니스 객체를 구성할 수 있다.

• 이식성(Portable GIS Application Server)

멀티-tier 환경에서 운용되는 GIS 애플리케이션 서버는 네트워크 상의 어떤 노드에서도 운용 가능해야 한다. 따라서, 클라이언트 오브젝트, 미들웨어 오브젝트, 서버 오브젝트를 모두 수용할 수 있어야 한다.

4.2 PASS†설계

4.2.1 동적 지도 생성

GIS 애플리케이션 서버의 설계에 있어 가장 크게 고려해야 할 사항은 다수의 클라이언트가 요구하는 사항들을 효율적으로 처리할 수 있어야 한다는 점이다. 이는 동적으로 로딩되는 지리공간 데이터의 양이 일반 데이터와 달리 대용량이라는 점에서 주로 기인한다. 웹 상에서 지리정보를 서비스하는 기존의 시스템들은 미리 정의된 데이터와 심볼들을 이용하므로 데이터의 동적인 로딩이 그리 필요하지 않다. 그러나, 능동적이고 효율적인 GIS 애플리케이션 서버를 위해서는 동적인 지도 생성과 서비스가 필요하므로 이 경우에 성능 문제는 매우 중요하다.

4.2.2 멀티 뷰

멀티 뷰는 하나의 공유하고 있는 하나의 공간 데이터를 서로 다른 사용자에게 동일 혹은 상이한 뷰로 보여주는 기술로 데이터 공유를 위한 기반 기술이다 [17].

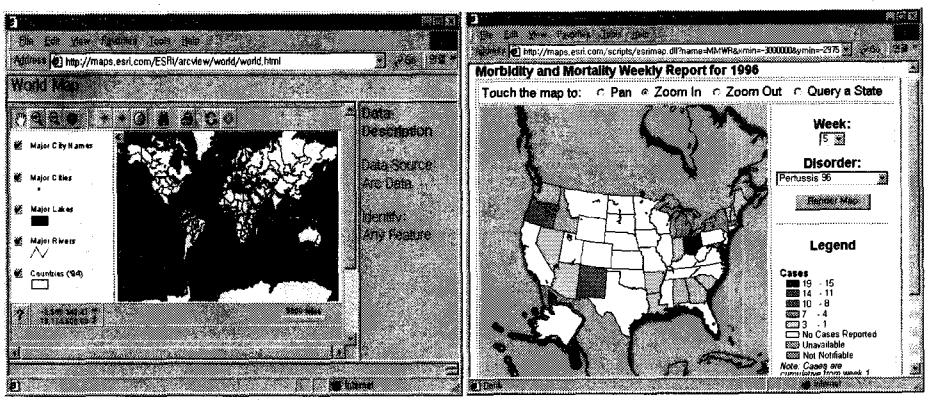

그림 12. ESRI사의 지도 서비스 (미리 정의된 데이터와 심볼을 이용)

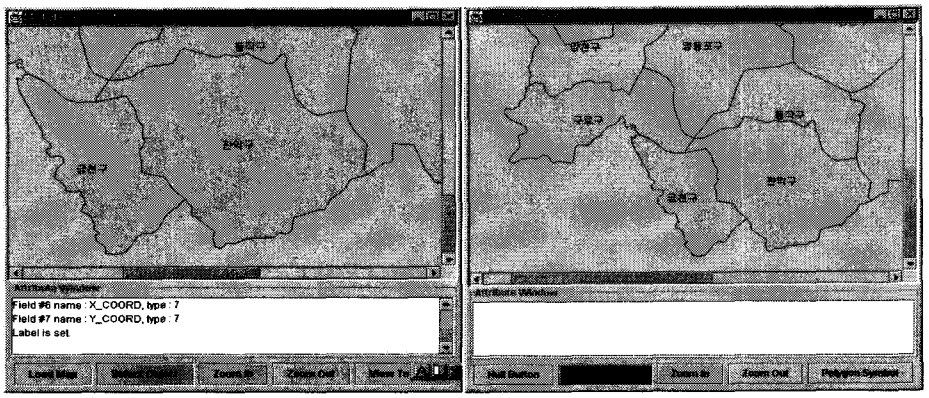

그림 13. 두 개의 뷰가 서로 다른 방식으로 확대/축소되는 경우

† Portable & active GIS 애플리케이션 서버를 설계 및 구현한 것

그림 13과 그림 14은 서울 구 경계를 담고 있는 공간 데이터를 로딩한 후에 두 개의 서로 다른 뷰로 디스플레이한 것을 보여준다. 이 두개의 뷰는 하나의 데이터를 공유하고 있으므로 실제 담고 있는 객체들은 동일하지만 확대/축소나 형태의 변화는 다른 방식으로 디스플레이될 수 있다. 멀티 뷰를 이용하는 경우 확대/축소 과정이 뷰마다 독립적으로 실행되거나 객체를 선택하고 편집하기 위해 두 개의 뷰가 동일한 상황을 디스플레이하도록 만들 수도 있다.

클라이언트 요구에 따라서 어떻게 유지할 것인가에 대한 전략을 비교한 매트릭스이다. 매니저 객체의 상태 1은 Thread per Request, 2는 Thread Pool, 3은 Shared Object를 나타내며, 매핑 커널의 A는 멀티 뷰를 사용하지 않는 경우, B는 뷰가 Thread per Request, C는 뷰가 Thread Pool, D는 뷰가 Shared Object 전략을 사용하는 경우를 나타낸다.

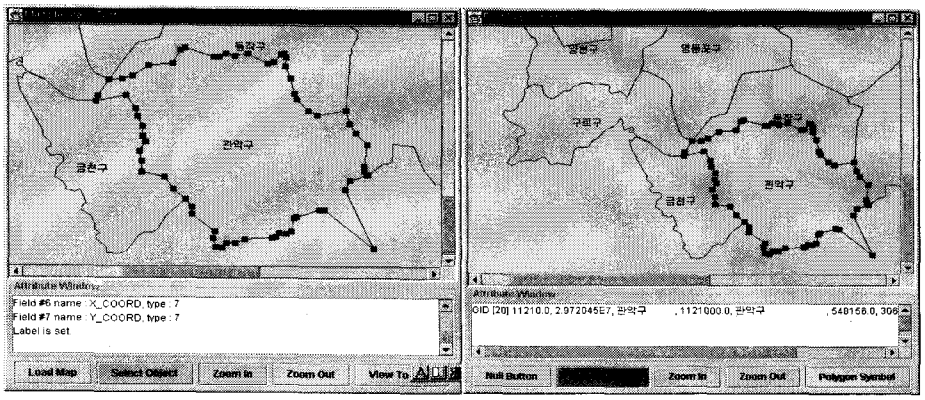

그림 14. 두 개의 뷰가 똑같이 편집되는 경우

4.2.3. 성능 개선을 위한 컴포넌트 구성 전략

오브젝트 웹을 고려한 분산 환경에서는 객체들을 어떤 방식으로 분산시키는가에 의해 성능이 크게 좌우된다. 특히 대용량의 공간 데이터를 자료원으로 사용하는 경우에는 네트워크 부하량과 서버 또는 미들웨어 구성 객체의 데이터 조작 능력이 중요하다.

N개의 클라이언트 객체가 미들웨어 객체에 대해 지도 이미지 스트림을 요구하는 경우를 생각해 보면 Connection Pool을 어떻게 유지하는가의 전략과 함께 멀티 뷰의 구성 방안, 그리고 데이터 제공자와의 관계가 문제가 된다. 그림 15은 매핑 커널과 이를 조작하는 애플리케이션 서버 내부의 매니저 객체 상태를

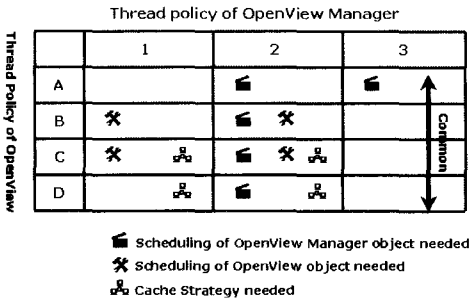

그림 15. 매핑을 위한 쓰레드 전략

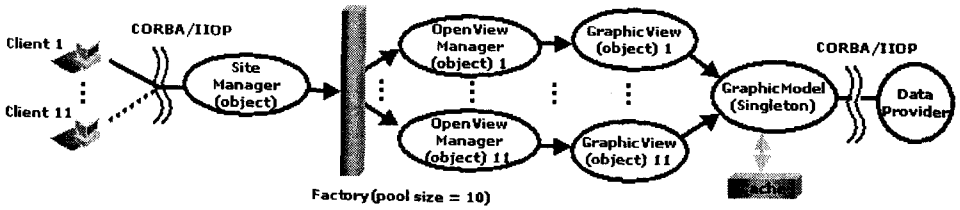

그림 16. 애플리케이션 서버 객체의 쓰레드 정책 2/D

이러한 열 두가지 쓰레드 정책 중에서 2/D의 방법이 클라이언트의 수가 늘어나도 안정적인 서비스할 수 있는 가장 좋은 정책 중의 하나이다. 그림 16은 2/D 전략을 사용하는 경우의 객체 배치를 보여주고 있다.

지도 이미지를 생성하는 뷰들은 멀티 뷰이므로 동일한 모델에 들어있는 공유 데이터를 이용한다. 공유 데이터를 이용하는 경우, 특정 클라이언트에 의해 데이터 제공자로부터 획득된 공간 데이터는 캐쉬로 저장되어 재사용될 수 있다.

4.3 PASS 구현

그림 17은 구현된 GIS 애플리케이션 서버의 주요 구성요소들을 다이어그램으로 보여주고 있다. 사이트 매니저(Site Manager)는 주로 지도 이미지의 생성에 사용되는 것으로 웹 서버와 서블릿, JSP 등을 이용하여 구성되었다.

그림 18은 Shapefile 데이터 제공자, Oracle Spatial 데이터 제공자, OpenStore† 데이터 제공자로부터 공간 데이터를 넘겨받아 동일한 캔버스에 디스플레이한 것으로 왼쪽의 트리는 현재 이용 가능한 객체(available objects)의 이름과 이용 가능한 데이터 및 심볼들을 보여주고 있다.

그림 18에서 Eagles 서버 오브젝트는 Shapefile 데이터 제공자, Unicorns 서버 오브젝트는 Oracle Spatial 데이터 제공자, Dragons 서버 오브젝트는 OpenStore 데이터 제공자, Sharks 서버 오브젝트는 OpenStore에 저장된 심볼 서비스 제공자를 구현한 것으로 서울시 구 경계 데이터는 Eagles 데이터 제공자로부터, 지하철 노선도는 Unicorns 데이터 제공자로부터, 그리고 CCTV 설치 지점은 Dragons 데이터 제공자로부터 받아 디스플레이한 것이다.

서로 다른 데이터 제공자에 접근할 수 있는 메커니

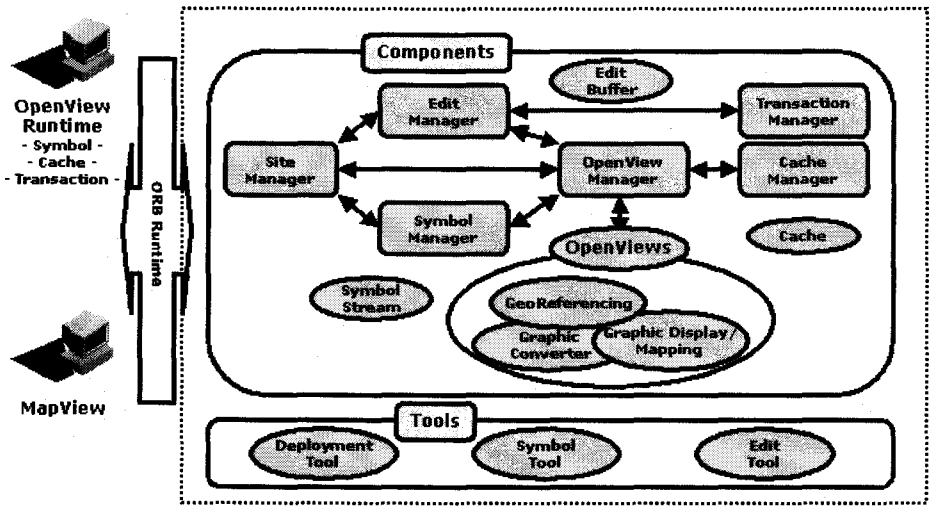

그림 17. PASS의 주요 구성요소

매핑 커널을 제어하는 매니저 객체(Openview Manager)와 네트워크 상의 심볼 서버들에 접근할 수 있는 심볼 매니저(Symbol Manager)는 지도 생성을 위해 가장 중요한 역할을 담당한다. 캐쉬 매니저(Cache Manager)는 데이터 제공자로부터 취득한 데이터를 캐쉬해 두며 편집 매니저(Edit Manager)는 데이터 갱신을 위한 작업을 진행한다.

즘은 SDE, GEUS, Geomania등과 같은 공간 데이터베이스에 대해 확장 설계될 수 있으며, 이런 접근 방식을 통해 기존의 이질적 시스템을 통합할 수 있다.

5. 제한점 및 앞으로의 연구방향

본 논문에서는 오브젝트 웹 GIS의 관점에서 새로운

† SUN GIS Lad.에서 개발한 순수자바 기반으로 오브젝트 저장소로 Btee와 Quadtree 인덱싱을 사용한다.

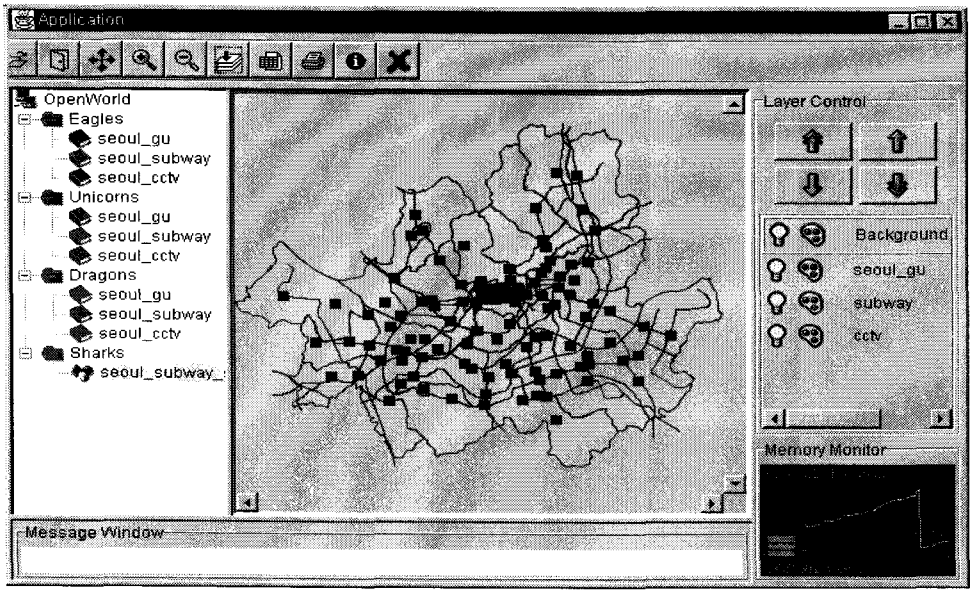

그림 18. PASS를 통한 지도 디스플레이

개방형 시스템 아키텍처의 필요성을 기술하고 최신 정보 기술을 고려한 개방형 GIS 시스템 아키텍처를 제시하였다. 새로운 아키텍처의 핵심이 되는 매핑 커널은 100% 순수 자바를 기반으로 하여 ORB를 통해 통신하므로 이식성과 이동성이 뛰어나다. 분산 환경에서 다양한 데이터 제공자와 함께 매핑 커널이 동작할 수 있도록 표준 인터페이스를 사용하였으며, 심볼 객체를 분산시키기 위한 방법으로 심볼 인터페이스 표준을 제시하였다.

OpenViews 매핑 커널을 내장한 GIS 애플리케이션 서버는 웹 서버와 서블릿, JSP등과 함께 EJB, CORBA/자바 기반의 컴포넌트들로 구성되도록 설계되었다. 특히, OpenViews가 갖는 멀티 뷰를 이용하여 시스템 성능 향상 기법을 제안하였다. 그러나, 아직까지 충분한 벤치마킹 및 시스템 운영 테스트가 이루어지지 않아 앞으로 안정 커널의 개발과 사용자 중심의 편집 툴을 보강하는 것이 필요하다.

끝으로 본 시스템 컴포넌트들은 오브젝트 웹 기술을 중심으로 전자지도 출판(digital map publishing)과 분산 환경에서의 퍼스널 GIS(Personal GIS)를 위한 기본 시스템으로 운영 가능하며, 리눅스 기반의 서버 시스템에서도 문제없이 동작할 수 있으므로 저비용의 공간 데이터 통합 및 서비스가 가능하다는 장점을 가지고 있다.

참 고 문 헌

[1] Landgraf, G., *Evolution of EO/GIS Interoperability towards an Integrated Application Infrastructure*, In: Schek, H., et al, Proceedings of Second International Conference, INTEROP '99, Springer-Verlag, 1999.

[2] The Open GIS Consortium : http//www.opengis.org/.

[3] Orafali, R., Harkey, D., Edwards, J., *Instant CORBA*, John Wiley & Sons, Inc., 1997.

[4] Fletcher, D. R, *Interoperable Enterprise*, Chapter 2 in Enterprise GIS Edited by Meyer N. R. and R. Scott Oppman, URISA, 1999.

[5] Nelson, J., *Programming Mobile Objects with Java*, John Wiley & Sons, 1999.

[6] Cockayne, W. and Zyda, M., *Mobile Agents*, Prentice Hall., 1997.

[7] NCGIA Research Initiatives : http//www.ncgia.ucsb.edu/.

[8] Microsoft Press, *Microsoft OLE DB 2.0 Programmer's Reference and Data Access*

SDK, 1998.

[9] Gio Wiederhold, *Mediation to Deal with Heterogeneous Data Sources*, in Interoperating Geographic Information Systems, Second International Conference, INTEROP '99 Proceedings, Springer-verlag, 1999.

[10] 김민수 외, *응용 시스템 구축을 위한 OLE/COM 기반의 GIS 데이터 제공자 컴포넌트 시스템에 관한 연구*, 한국 GIS 학회지, 제7권 제2호, 1999.

[11] Booch, G., *Object-Oriented Design with Applications (2nd Edition)*, Benjamin/Cummings, Redwood City, California, 1994.

[12] Grand, M., *Patterns in Java*, John Wiley & Sons, New York, 1998.

[13] Max Egenhofer, John Herring, *A Mathematical Framework for the Definition of Topological Relationships*, in K Brassel & H Kishimoto, Proceedings of the 4th International Symposium on Spatial Data Handling, Zurich, 1990.

[14] Danny A. et al, Professional Java Server Programming, Wrox Press Ltd., 1999.

[15] OpenStream consulting Inc., *Analysis of the Application Server Market.*, 1998.

[16] Frank, A. and Timpf, S., *Multiple Representations for Cartographic Objects in a Multi-scale Tree-An Intelligent Graphical Zoom. Computers and Graphics* 18(6), 1994.

[17] Reifer, Donald J., *Practical Software Reuse*, John Wiley & Sons Inc., New York., 1997.

[18] Szyperski, C., *Component Software*, ACM Press, New York., 1998.

[19] Goodchild, M., et al., *Interoperating Geographic Information Systems*, Kluwer Academic Publishers, London., 1999.

[20] Vckovski, A., *Interoperable and Distributed Geoprocessing in GIS*, Taylor & Francis, London., 1998.

[21] Buehler K., Mckee L., *The OpenGIS Guide : Introduction to Interoperable Geoprocessing and the OpenGIS Specification*, 3rd ed., Open GIS Consortium, Inc., 1998.

[22] Genesereth, Micheal R., *Interoperability : An Agent-Based Framework*, AI Expert, Vol. 10(3), 1995.

[23] Powersoft, *Business Application Development for the Internet*, Powersoft White Paper, http://www.sybase.com/powersoft/products, 1996.

박기호

1982 서울대학교 건축학과 (학사)

1985 Univ. of Washington 도시계획학과 (석사)

1994 Univ. of California, Santa Barbara 전산학과 (박사)

1995~현재: 서울대학교 지리학과 조교수

관심분야: 개방형 GIS, 분산 객체처리, 데이터웨어하우스, 계량분석기법

정재곤

1996년 서울대학교 지리학과 졸업 (학사)

1998년 서울대학교 지리학과 졸업 (석사, GIS 전공)

1998년~현재 서울대학교 지리학과 박사과정 (GIS 전공)

관심분야 : CORBA/Java 기반 개방형 GIS 시스템, 공간 데이터베이스, 공간 통계, 전자지도 출판