분산객체 환경에 기반한 GIS 엔진개발의 연구

A Study on Development of GIS engine based on Distributed Object & Component Environment

이정수(Lee JeongSoo)
서울시립대학교 도시과학대학원

제 1 절 연구배경

하드웨어 발달과 소프트웨어 발달의 불균형이 심화되면서 소프트웨어 발전의 새로운 전환이 요구됨에 따라 지난 80년대 대두된 객체지향 기술의수용으로 분산에 대한 요구와 객체지향에 대한 반향을 동시에 포함하는 획기적인 컴퓨팅 환경으로 발전하였다. 분산객체환경은 이미컴퓨팅 기술의 핵심 솔루션을 자리잡았고 국내에서도 다양한 시스템, 서비스에 적용되어 기술적, 상업적 가치가 검증되었다. 네트워크 성능의 기하급수적인 발전과 사용자 요구의 다양성 및 이에 상응하는 시스템의 복잡도가 지속적으로 늘어나고 있어 분산객체환경의 적용과 확산은 전체 컴퓨팅 환경을 대상으로 한 당연한 발전방향으로 인식되고 있다.

분산객체 환경은 기존 아키텍처만으로는 기술적인 포화상태에 직면한 GIS가 한층 범용적이고 안정적인 시스템으로 발전할 수 있는 획기적인 전기를 마련할 수 있다. 따라서 본 연구는 최신 분산객체 환경을 이용하여 GIS를 구축하고자 할 때 먼저 고려되어야 할 기술적 요소를 검토하고 분산객체 GIS 의 효과적인 구현 방안을 도출하고자 한다. 그리고 분산객체 환경으로 GIS 엔진을 개발하였을 경우 발생하게 될 문제점을 고찰하고 최적의 분산객체형 GIS 엔진 개발을 위한 기술적인 해결방안을 제시하고자 한다.

제 2 절 도입의 문제점

대부분의 경우 GIS는 다른 IT 분야보다 새로운 컴퓨팅 환경을 적용하는데 소극적이다. GIS의 적은 수요나 인력 등의 시장요소들이 주요 원인으로 먼저 고려되어야 하겠지만 GIS 자체의 특성이 일반적인 시스템들과는 상당히 다른 즉 기술적인 이질성도 원인이 되고 있다.

범용적인 활용을 목적으로 디자인된 각종 컴퓨팅 환경은 GIS에 적용되기 이전에 상당부분 수정되거나 최적화되어야 하며 이때 GIS 도입의 문제점들이 구체적으로 분석되어야 한다. 분산객체가 GIS에 도입되므로 해서 GIS의 활용도가 개선되는 것은 사실이지만 실상 이러한 개선효과를 가져오기 위해서는 기존 GIS의체계를 상당부분 개선할 필요가 있다.

1. GIS의 문제점

GIS는 데이터의 크기가 매우 크고 하나의 기능 수행을 위해 동시에 처리해야 하는 데이터 량도 여타의 시스템에 비해 많다. 특히 시스템간 연동을 위해서는 네트워크를 통한 데이터의 전송이 전체 수행 부하에 큰 비중을 차지하는데 이때 이동될 데이터의 크기가 크면 즉시 시스템의 성능 문제로 귀착될 수 밖에 없다. 이러한 데이터의 크기 문제를 극복하기 위해서는 아래 세 가지의 기술적인 개선이 필요하다.

◆ 전송 데이터의 경량화
◆ 데이터 전송의 최소화
◆ 대상 도면 범위의 최적화

전송 데이터의 경량화는 각 도형데이터에서 불필요하거나 중복된 부분을 제거하는 방법이다. 도형 내부에서 중복된 부분을 제거하는 것은 데이터의 품질에 많은 영향을 미치는 것이 일반적이다. 물론 중복 제거의 범위에 따라 원 데이터를 손상시키지 않도록 할 수 있지만 이러한 경우는 경량화 정도가 미미하다. 데이터의 전송을 최소화 하기 위해서는 서버와 클라이언트간의 오퍼레이션 호출을 최소화하여야 한다. 특별한 경우가 아니면 도형 데이터 자체의 이동은 배제하고 가급적 최종적으로 얻고자 하는 결과값만을 클라이언트에 리턴 될 수 있어야 한다. 특히 한번의 호출에 최종 결과로 단일 값이 리턴 되는 경우가 가장 이상적이라 할 수 있다.

데이터의 규모 문제 외에도 시스템 자체가 가지는 기술적인 특수성으로 인해 발생하는 문제가 있다. 초기에 도입된 GIS는 자체시스템에 컴퓨팅 기술의 하부 구조로부터 비즈니스 모델에 이르는 상위 구조까지 자체적인 규격을 가지고 개발된, 말하자면 독립성이 강한 구조가 대부분이다. 이러한 문제점은 매우 지난한 작업 방법을 가지는데, 결국 기존 시스템을 내부에 두고 그 밖을 범용적, 표준적인 인터페이스로 둘러싸는 일종의 Wrapper를 개발하는 것이다. Wrapper내부의 코드는 기존 GIS와 완전하게 연결될 수 있도록 기존 GIS가 사용하는 프로토콜을 준수한다. 대신 Wrapper에 의해 외부에 노출되는 인터페이스는 OGC 표준 인터페이스를 철저하게 준수한다.

물론 이러한 방법은 Wrapper 내부에서 데이터의 변환 작업이 반복적으로 발생하므로 기본적으로 중복된 부하를 가져온다. 이러한 실행 부하는 데이터 변환작업의 최적화, 예를 들면 데이터 변환 요청 및 결과에 대한 캐쉬 처리 등을 이용하여 최적화할 수 있다.

2. 분산객체환경의 문제점

분산객체환경이 가지는 문제점으로는 첫째로, 퍼포먼스의 문제를 들 수 있다. 왜냐하면 이질적인 환경을 하나의 표준 규격으로 묶기 위해서는 당연히 Conversion을 위한 리소스의 소모가 필수적이기 때문이다. 특히 메시지의 전송 즉, 마샬링을 네트워크를 통해 호출 시 매번 수행하는 것은 공유된 메모리를 통해서 Native 형의 바이너리 데이터를 직접 전송하는 것 보다 데이터 자체의 오버헤드도 존재할 뿐만 아니라 네트워크 부하를 피할 수 없다.

이러한 문제점에 대하여 "시간이 모든 부하를 해결한다" 또는 "상호운영성의 개선은 퍼포먼스의 희생을 압도한다"라는 주장은 경험적으로는 그 동안 컴퓨팅 환경의 발전 속도를 볼 때 일면 설득력이 있으나 GIS를 포함한 비교적 사용 리소스의 크기가 매우 큰 분야에서는 만족할 만한 해답이 되지는 않는다. 물론 이러한 환경적인 문제가 그 동안GIS 에 신기술 적용이 늦어진 원인이기도 하다.

둘째, 분산객체환경의 표준화 문제이다. 현재 OGC가 제시한 표준화 작업의 결과 즉, Simple Specification v1.1은 실제 GIS 엔진의 구현에 적용되기 이전에 상당 부분 확장되어야 한다. 또OLE/COM과 CORBA의 효과적인 연동은 많은 노력이 투자되어야 얻어질 수 있는 것으로 아직 각 분산객체환경이 제안하는 완전한Inter-Operability는 어느 누구도 지원할 수 없는 상태이다.

제 3 절 Open GIS와의 연관성

OGC 규격은 객체 인터페이스의 표준안을 제공하기도 하지만 주로 데이터 교환 포맷의 통일을 통해 GIS 데이터베이스를 상

호 활용하고자 하는 목표가 가장 크다고 할 수 있으며 분산객체환경을 위한 접근은 COM과 CORBA를 위한 기본 설계만을 제시하고 있다. OGC는 GIS 데이터의 교환 포맷으로 WKB(Well Known Binary)를 규정하고 있다. WKB는 순수한 바이너리 포맷으로 포맷 자체에 수치데이터의 엔디언(Endian) 방식을 표시하고 유형 및 구성요소의 수, 각각의 좌표 값들을 가지고 있는 공개된 포맷이다.

그 동안의 GIS는 플랫폼에 매우 의존적이고 폐쇄적인 경향이 있어 구축된 데이터베이스 간에 데이터연동이 완전하지 못하였다. 데이터베이스의 스키마가 다를 경우 이로 인해 데이터의 누락과 변이가 매우 많았다. WKB의 활용은 데이터베이스 구축 시 일종의 표준화된 제한을 강제하는 것으로 이로 인해 원 데이터 자체를 WKB로 저장하거나 아니면 적어도 원 데이터가 WKB로 오류 없이 변환될 수 있는 형태로 데이터베이스에 저장될 수 있도록 한다. 이를 통해 한 시스템의 데이터(특히 도형데이터)가 타 시스템에서 동일한 값으로 사용될 수 있도록 한다. 다만 WKB는 교환포맷으로는 규격자체가 상당히 정교하며 효율적이나 네트워크 전송을 위한 전송포맷으로는 매우 커서 압축 및 간소화 작업이 필요할 수 있다.

제 4 절 효율적인 엔진의 설계

표준 인터페이스 설계는 다양한 플랫폼을 기준으로 하고 또 대상으로 삼지만 본 연구에서는 개발 플랫폼을 Microsoft의 DCOM으로 결정했다. OGC의 DCOM 규격은 CORBA 규격에 비해 다소 체계적이지 못하고 상세하지 않은 문제가 있으므로 경우에 따라 CORBA의 규격도 일부 수용하였다.

연구의 성과는 크게 운용성 측면에서의 개선과 수행 성능의 개선을 중심으로 평가한다. 다만, 운용성의 개선은 정량적인 측정이 불가능하므로 운용성의 개선을 관측할 수 있는 활용 사례를 통하여 개선된 효과를 기술한다. 그리고 수행 성능의 측정은 GIS에서 가장 많이 호출될 수 있는 오퍼레이션을 중심으로 오퍼레이션의 사이클을 데이터의 전송량과 시간을 기준으로 측정, 비교한다.

1. 시스템 운용성의 개선

단일 컴퓨터에서 실행되던 GIS는 여러 대의 컴퓨터로 분리되어 운영될 수 있으며 이를 통해 부하를 분산시켜 전체적으로 퍼포먼스가 개선될 수 있다. 분산객체기술은 이러한 환경을 구성을 가능하게 한다. 서버 시스템의 분리는 두 가지의 중요한 장점을 제공한다.

- ◆ 특정 목적에 할당된 전용 서버에 대한 전문적인 관리가 가능하다.
- ◆ 부하의 분담을 통해 서비스 안정성이 증가된다.

즉, 시스템의 관리가 용이하고 서비스의 퍼포먼스를 개선할 수 있다. 서비스 퍼포먼스의 개선은 단순히 하나의 컴퓨터가 여러 개의 컴퓨터로 대응되는 즉, 시스템 파워의 확장을 통해 얻어질 수도 있지만 적절한 Load balancing에 의해 최적의 성능을 발휘할 수도 있다.

완성된 어플리케이션을 제공하여 한정적인 방법을 통해 커스터마이징을 하도록 하는 기존 범용 GIS이 지향하는 방법은 사용자에게 초기 접근을 매우 쉽도록 하는 장점이 있다. 사용자는 간단한 스크립트를 통해 메뉴를 추가하고 기존에 제공하는 기능을 조합하여 확장 기능을 작성, 사용한다. 그러나 이러한 방식은 기능 보완에 대한 제한적인 수단만을 제공하여 개발 가능성을 매우 제한한다. 반대로 소스코드 수준으로 제공하는 템플릿 기반의 툴들은 개발자에게 많은 사전 지식을 요구하여 개발 가능성을 역시 제한하고 있다. 그러므로 분산객체환경이 제공하는 컴포넌트 형태의 제공 방식은 개발자가 기본적인 개발환경만을 이해한다

면 수정가능범위도 넓고 개발도 용이한 두 가지 장점을 모두 제공하는 것이다.

GIS는 데이터베이스를 포함한 시스템의 각 리소스가 일반 사용자로서는 접근하기 어렵기 때문에 전문인력이 아니고서는 충분한 사용 효과를 얻어낼 수 없다.

2. 수행 성능의 개선

분산객체형 GIS 엔진의 성능에 가장 큰 영향을 미치는 요소는 GIS의 기본 부하를 제외한다면 실행 중 오퍼레이션의 호출 부하라 할 수 있다. 분산객체환경이 여타의 개발, 실행 환경과 차이를 가지는 것은 오퍼레이션의 호출을 위한 마샬링을 스스로 해결해준다는 것인데 이러한 범용적인 구조가 모든 오퍼레이션에 동일한 효과를 부여하지는 않는다. 오퍼레이션의 호출이 매우 자주 일어나는 경우 마샬링 자체의 부하가 시스템의 성능을 떨어트리는 주요한 요인이 될 수 있다. 그러므로 본 연구에서는 마샬링이 발생할 수 있는 빈도를 최소한으로 줄이는 노력이 필요했다. 예를 들어 하나의 도형을 데이터베이스로부터 액세스되어 마샬링 후 클라이언트 화면에 최종 도시할 때까지의 절차를 살펴볼 필요가 있다. OGC의 인터페이스 설계는 이 문제에 대한 고려는 생략되어 있기 때문에 실제 시스템의 설계에 활용되기 위해서는 많은 개선이 필요하다. 도형을 구성하는 좌표값들을 일일이 액세스 하는 인터페이스 이외에 화면 도시에 필요한 기타 정보를 포함하여 한꺼번에 액세스할 수 있는 인터페이스를 추가해야 한다.

다음으로 수행성능에 미치는 영향이 매우 큰 요소로 GIS의 특성이라고 할 수 있는 전송데이터의 크기 문제가 있다. 분산객체 환경에서는 전송 데이터의 크기가 클 경우 본래의 마샬링 이외에 추가적인 마샬링 필터를 개발하여 적용하는 커스텀 마샬링을 응용할 수 있다. 커스텀 마샬링에 압축 필터를 적용하여 마샬링 단계에서 전송 데이터를 압축하고 클라이언트에서 전송 받은 후 다시 복원하여 사용한다. 인코딩/디코딩 단계에서 추가적으로 소요되는 부하가 시스템 퍼포먼스에 대해 마이너스 효과로 작용할 수 있으나 대부분의 경우 네트워크 부하를 줄여 전체 서비스의 성능에 대해서는 긍정적인 효과를 제공한다.

3. 개선 효과의 검증

성능 비교의 항목은 다음과 같이 분류하였다.

◆ 화면 도시를 위한 정보의 액세스 속도
◆ 마샬링으로 전달되는 물리적인 데이터의 양

비교 항목		OGC 규격	개선 후
완료 시간		33.4sec	4.8sec
호출회수	총계	1,286,509	280,960
	행정 경계	17,380	72
	하천	6,199	13
	지하철	20,561	92
	건물	713,331	170,908
	아파트	510,772	100,740
	심볼	18,266	9,135

<표 1> 개선의 효과

테스트에 필요한 데이터는 서울시 데이터로 6개의 레이어만 선별하였다. 위 검증 결과를 기준으로 분산객체형 GIS 엔진의 설계 및 개발에서 퍼포먼스의 개선에 집중할 것은 오퍼레이션의 호출과 각 오퍼레이션이 가지는 물리적인 부하를 최소화하는 것으로 볼 수 있다. 또 이러한 개선 작업은 가급적 단일 개체가 많은 수의 좌표데이터로 구성되어 있는 레이어에서 가장 큰 효과를 가진다. 데이터의 압축은 무시할 수 있는 범위 내에서라도 일종의 데이터 열화가 발생하는

것임에 주의하여야 하고 또 압축필터의 적용이 그만큼 퍼포먼스의 저하를 가져오는 것은 당연하다고 하겠다. 그러므로 개체 구성 좌표가 매우 적은 즉, Point 개체들로 구성된 레이어에서는 압축결과도 좋지 안고 전송 성능의 개선에서도 커다란 혜택을 가져올 수 없으므로 신중하게 적용되어야 한다.

제5절 결론

초기의 범용 GIS 시스템은 사용자를 GIS 전문가로 가정한 다기능의 솔루션이었다. IT의 확산으로 일반 사용자들에게도 GIS를 접할 수 있는 다양한 기회를 제공할 수 있으나 여전히 범용 툴 형태의 접근으로 사용자가 요구하는 서비스를 제공하는 것은 불가능하다. 뿐만 아니라 다양한 서비스와의 연계를 통해 복합정보를 제공하고자 하는 시도가 활발한 가운데 일반 개발자가 쉽게 서비스 개발에 이용할 수 있는 단순하고 쉬운 GIS 엔진의 보급이 필요하다. 이러한 목적을 달성하는데 분산객체환경은 바람직한 개발 모델을 제시한다. 분산객체형 GIS 엔진을 활용하면 서비스 개발자는 GIS에 대한 구체적인 기술을 알지 못하더라도 서비스에 응용할 수 있고 또 다양한 방향으로 응용분야를 확대할 수 있다. 기술의 성능 우위를 보더라도 분산객체환경은 향후 GIS의 기본 개발환경으로 정착될 것이다.