GML 데이타를 지원하는 확장된 DOM
(An Extended DOM for GML Data)

반 재 훈 † 조 정 희 †† 문 상 호 ††† 홍 봉 희 ††††
(ChaeHoon Ban) (JungHee Jo) (SangHo Moon) (BongHee Hong)

요 약 OGC(Open GIS Consortium)는 웹 GIS 환경의 상호 운용을 제공하기 위해, 웹 매핑 테스트베드를 통한 맵서버와 GML 명세를 정의함으로써 새로운 웹 매핑 기술을 제안하고 있다. 이 환경에서 맵서버는 다양한 데이타 서버들의 지리정보를 GML로 변환하여 클라이언트에게 제공하고, 클라이언트는 전송받은 GML 데이타를 웹 브라우저에 출력한다. 이러한 웹 매핑 테스트베드는 웹 GIS 환경에 분산 저장된 GIS 정보의 발견, 접근, 통합 및 출력 방법은 제공하지만 GIS에서 필수적인 공간연산을 수행하는 방법은 제공하지 않는다.

본 논문은 웹 환경의 다양한 데이타 서버들로부터 중첩되어 전송된 GML 데이타에 대한 공간연산을 수행하는 방법을 제시한다. 이를 위해 W3C에서 XML 문서의 분석을 위해 제시한 DOM과 OGC에서 개방형 GIS 시스템의 구현을 위해 제시한 OpenGIS 단순 피처(Simple Features) 명세를 기반으로, 웹 GIS 환경에서 GML 데이타에 대한 공간연산을 지원하는 GDOM(Geographic Document Object Model)을 설계하고 구현한다. 그리고 웹 매핑 테스트베드 환경에서 클라이언트가 GDOM을 이용하여 공간연산을 수행하는 과정 및 결과를 보인다.

키워드 : OpenGIS, 웹 매핑, DOM, 공간연산, XML, GML

Abstract The OpenGIS Consortium has proposed a new web-mapping technology to support interoperability in web GIS environment by developing the specifications of MapServer and GML. In this environment, the MapServer transforms legacy spatial data into GML data, and clients display them on standard web browsers. This web-mapping testbed proposes methods for discovering, accessing, integrating and displaying GIS information except processing of spatial operations which are essential services in GIS environment.

This paper proposes the method for executing spatial operations on GML data which are overlays of different map layers in legacy data servers. To support spatial operations on GML data in web GIS environment, this paper designs and implements GDOM based on the W3C's DOM Specifications and OGC's Simple Features Specifications. This paper shows the specifications and implementation of GDOM and the process of spatial operations in web-mapping testbed environment.

Key words : OpenGIS, Web Mapping, DOM, Spatial Operation, XML, GML

1. 서 론

월드와이드웹(World Wide Web)이 인터넷을 중심으로 급속도로 발전함에 따라 웹 환경의 수많은 정보들을 효율적으로 이용하기 위한 연구의 필요성이 제기되고 있다. GIS 분야에서도 웹 환경에 분산되어 있는 이질적인 지리정보에 대한 상호 운용을 제공하기 위해 웹 GIS에 대한 연구와 기술 개발이 진행 중이다. OGC에서는 이러한 동향에 따라 웹 매핑 테스트 베드를 통해 기존 OpenGIS(Open Geodata Interoperability Specification)의 상호 운용성을 웹 환경에서 검증하고, 지리 정보의 표현 및 접근을 위한 새로운 웹 매핑 명세[1]를 정의한다. 웹 매핑 명세는 첫째, URL 형식의 클라이언트 질의를 수행하여 GML, JPEG, SVG, GIF 형식의 지리정보를

† 정 회 원 : 경남정보대학 인터넷상거래과 교수
chban@kit.ac.kr
†† 비 회 원 : LG전자 차세대단말연구소 연구원
jhjo@lge.com
††† 정 회 원 : 부산외국어대학교 컴퓨터전자공학부 교수
shmoon87@taejo.pufs.ac.kr
†††† 종신회원 : 부산대학교 컴퓨터공학과 교수
bhhong@pusan.ac.kr
논문접수 : 2001년 11월 12일
심사완료 : 2002년 7월 20일

제공하기 위한 OpenGIS 웹 맵서버 인터페이스 구현 명세[2]와 둘째, 웹 환경의 이질적인 지리정보를 XML 문서[3]로 변환하여 통합하기 위한 GML(Geography Markup Language)명세[4]가 있다.

GIS에서는 일반적으로 공간연산을 이용한 질의를 수행하여 공간객체간의 공간관련성 정보를 획득한다. 예를 들어, 그림 1과 같이 웹 브라우저에 출력된 GML 데이타에 대해 *"A지점에서 반경 500m내의 모든 병원을 출력하라"* 또는 *"특정 지하철 노선이 지나가는 도로를 화면에 출력하라"*와 같은 질의는 공간연산을 이용한 질의이다. 그러나 웹 매핑 테스트베드에서는 맵서버와 GML 명세를 통해 웹 환경에서 GIS 정보의 발견, 접근 및 출력 방법은 제시하지만 획득한 GML데이타에 대한 공간연산을 수행하는 작업은 제시하지 않는다. 따라서 GML 데이타에 대해, 공간 질의를 수행하기 위한 공간연산 수행 방법의 제시가 필요하다.

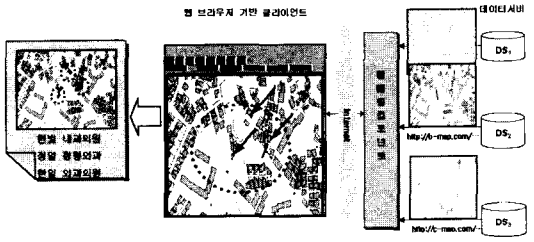

그림 1 웹 매핑 테스트베드 환경에서 지도출력을 위한 구조

웹 환경의 다양한 서버들로부터 전송된 GML 데이타에 대해 공간연산을 수행하기 위해서는GML 형식으로 표현된 공간객체의 기하 및 속성 정보를 문서로부터 검색하고 추출해야 한다. W3C에서 제안한 DOM은 XML 요소의 접근 및 조작을 위한 표준 API를 제공하므로 GML 데이타에 대한 공간연산 작업에 이용할 수 있다. 그러나 DOM은 광범위한 XML 응용에 공통적으로 적용될 수 있는 사항을 고려하여 정의된 인터페이스이므로, XML의 특정 응용인 GML로 표현된 공간객체에 대한 공간연산을 지원하기 위해서는 GIS 환경에서 요구되는 기능을 고려하여 DOM을 확장하는 작업이 필요하다.

본 논문은 웹 환경의 다양한 데이타 서버들로부터 전송된 GML 데이타에 대해 공간연산을 수행하는 방법을 제시한다. 이를 위해 W3C에서 XML 문서의 분석을 위해 제시한 DOM과 OGC에서 개방형 GIS 시스템의 구현을 위해 제시한 OpenGIS 단순 피처 명세를 기반으로, 웹 GIS 환경에서 GML 데이타에 대한 공간연산을 지원하는 확장 DOM인 GDOM(Geographic Document Object Model)을 설계하고 구현한다. 그리고 웹 매핑 테스트베드 환경에서 클라이언트가 GDOM을 이용하여 공간연산을 수행하는 과정 및 결과를 보인다.

본 논문의 구성은 다음과 같다. 먼저 2장에서 관련 연구를 설명하고 3장에서 GDOM의 설계 내용을 기술한다. 그리고 4장에서는 GDOM의 구현 내용을 설명하며 5장에서는 결론 및 향후 연구를 기술한다.

2. 관련 연구

이 장에서는 OGC에서 웹 GIS 환경을 구축하기 위한 웹 매핑 테스트베드와, XML 문서의 분석을 위해 W3C에서 제안한 표준 DOM및 특정 응용에서 사용하기 위해 표준 DOM을 확장한 확장 DOM에 대해 설명한다.

2.1 OGC의 웹 매핑 테스트베드

웹 환경에서 OpenGIS의 상호 운용성을 검증하기 위한 웹 매핑 테스트베드는 기존의 웹 브라우저를 사용하여 다양한 데이타 서버에 저장된 지리 정보를 발견, 접근하는 것을 목적으로 한다. 웹 매핑 테스트베드 시스템을 구성하고 있는 GIS 컴포넌트들의 역할은 표 1과 같다.

표 1 웹 매핑 테스트베드 환경의 GIS 컴포넌트의 역할

GIS 컴포넌트	역할 정의
Publisher 클라이언트	맵서버에 데이터 셋 등록 및 메타정보 요청
Viewer 클라이언트	미디에이터로부터 전송받은 GML의 분석 및 출력
미디에이터	맵서버로부터 전송받은 GML 형식의 레이어들의 통합
맵서버	데이터 제공자로부터 전송받은 지리정보를 GML로 변환
데이터 제공자	데이터 서버로부터 전송받은 지리정보를 OpenGIS 명세를 기반으로 변환
데이터 서버	기존의 공간 데이터 서버

웹 매핑 테스트베드에서 클라이언트는 그림 2와 같이 질의를 통해 이질적인 데이타 서버들에 저장된 기하 정보를 4단계에 거쳐 획득하게 된다. 1단계에서, 다수의 데이타 서버(DS_1, DS_2, ···, DS_n)들은 기하정보를 OGC 단순 피처 명세에 정의된 정립된 이진 기하(WKB Geometry)로 변환하여 맵서버에 전송한다. 2단계에서, 맵서버는 각 데이타 서버들로부터 전송받은 정립된 이진 기하를 GML로 변환하여 미디에이터에 전송한다. 3단계에서, 미디에이터는 전송받은 GML 문서들을 하나의 GML 문서로 통합하여 클라이언트에 전송한다. 이때 전송되는 GML 문서는 클라이언트가 요청한 레이어들을 중첩한 전체 지도이다. 마지막으로, 클라이언트는 전송받은 GML을 분석하여 웹 브라우저에 출력한다. 이와 같은 웹 환경에서의 지리정보 통합은 데이타 서버와 클

라이언트의 응용인 웹 브라우저 사이에 위치한 맵서버가 이질적인 지리정보를 시스템에 독립적인 데이타 구조인 GML로 변환함으로써 수행된다.

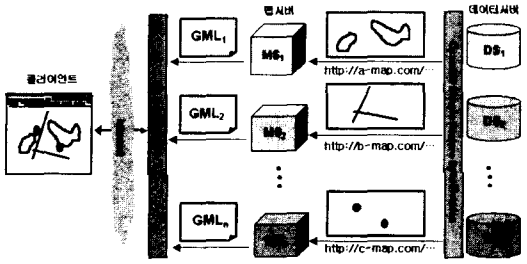

그림 2 상호 운용을 지원하는 웹 매핑 기술

2.2 DOM 및 확장 DOM

GML로 표현된 지리정보를 출력하기 위해서는 GML 문서의 요소를 검색하여 공간객체들의 좌표를 추출하는 작업이 필요하다. 이를 위해 XML 문서에 대한 접근 및 조작을 위한 공통 API로서 W3C에서 제안한 DOM[5]을 이용한다.

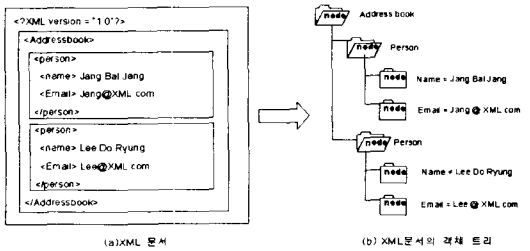

그림 3 DOM을 이용한 XML 문서의 객체 구조의 트리

DOM은 W3C가 웹 환경의 XML 문서를 다루기 위해 표준화한 인터페이스로서 XML 문서 내의 태그의 요소들은 DOM을 통해 객체화되며, 객체화된 각각의 노드들은 그림 3과 같이 XML 문서의 논리적인 구조를 나타내는 트리 형태로 구성된다. 개발자는 DOM 명세에 정의된 인터페이스를 기반으로 트리의 노드에 접근하여 XML 요소의 검색뿐만 아니라 추가, 수정, 삭제의 작업을 수행한다.

표준 DOM은 일반적인 XML 문서를 분석하기 위해 모든 응용에 적용할 수 있는 공통적인 부분을 고려하여 설계한 표준 인터페이스로, 특정 응용에 종속적인 기능은 포함하고 있지 않다. 그러므로 각 XML 응용 분야에서는 해당 응용에 적합한 기능을 추가하여 DOM을 확장하는 작업을 수행하고 있다. 그 예로, 수학 분야의 MathML[6]을 지원하는 MathML DOM[7], 화학 분야의 CML[8]을 지원하는 CML DOM[9], 그래픽 분야의 SVG[10]를 지원하는 SVG DOM[11] 등이 있다.

3. GDOM의 설계

이 장에서는 GIS 환경에서 공간연산을 지원하는 확장된 DOM인 GDOM의 정의 및 인터페이스 명세의 설계 내용과 구성을 설명한다.

3.1 GDOM의 정의

GDOM은 웹 GIS 환경에서 맵서버로부터 전송받은 GML 데이타에 대해 공간연산 기능을 지원하기 위해 표준 DOM을 확장한 것이다. GDOM의 인터페이스 명세는 GIS 표준화 기관에서 제시하는 데이타 모델을 기반으로 설계해야 하며, GIS 분야의 표준화 기관으로는 CEN/TC 287, ISO/TC211, OGC가 있다. 본 논문은 OGC의 웹 매핑 테스트베드가 기반이 되므로 OGC의 단순 피처 기하 모델에서 제시하는 공간연산자의 인터페이스를 표준 DOM에 추가하여 GDOM으로 확장한다. 표 2는 본 논문에서 고려되는 OGC의 공간연산자를 분류한 것이다.

표 2 OGC의 공간연산자 분류

ISpatialRelation	Equals, Touches, Contains, Within, Disjoint, Crosses, Overlaps, Intersects	진리값
ISpatialRelation2	Relate	
ISpatialOperator	Distance	실수
	Boundary, Buffer, Intersection, Union, Difference, SymmetricDifference, ConvexHull	기하정보

3.2 GDOM의 인터페이스 명세 설계

OGC의 OpenGIS Simple Features Specification for OLE/COM 명세에 의하면, 클라이언트는 OpenGIS OLE DB 인터페이스를 호출하여 공간 데이타를 정립된 이진 기하의 형태로 획득한다. 그리고 이 명세의 기하 컴포넌트 인터페이스를 호출하여 정립된 이진 기하로부터 기하 COM 객체를 생성한다. 이 때 생성하는 기하 COM 객체는 단순 피처 기하 클래스 구조를 기반으로 한다.

OGC의 기하 데이타 모델을 기반으로, GDOM을 설계 시 고려해야 할 사항은 다음과 같다. OpenGIS 기하

컴포넌트 명세는 그림 4(a)와 같이 클라이언트로부터 정립된 이진 기하를 입력 받아, 임의 기하 타입의 Open GIS COM 객체를 생성하는 것이다. 그러므로 이 명세를 그림 4(b)와 같이 웹 환경의 GML 문서에 적용하기 위해서는, OpenGIS 기하 컴포넌트 명세에서 제시하는 인터페이스와 메소드 중에서 정립된 이진 기하에 의존적인 요소는 GML을 고려하여 보완하고, 부족한 부분은 추가해야 한다.

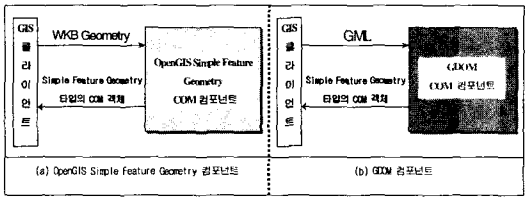

그림 4 OpenGIS 단순 피처 기하 컴포넌트와 GDOM 컴포넌트의 비교

표 3은 이러한 사항을 고려하여 W3C의 DOM의 명세와, OGC의 OpenGIS Simple Features Specification for OLE/COM 명세를 기반으로 GDOM의 인터페이스 명세를 설계한 것이다. 이텔릭체로 강조한 인터페이스 및 메소드는 웹 환경의 GML을 고려하여 기존 Open GIS Simple Features Specification for OLE/COM 명세를 확장한 것이다. 즉, IGeometryFactory, IGML DOMDocument, IWks, IGeometryCollection은 웹 환경의 GML을 고려하여 확장한 인터페이스이며, 그 이외의 인터페이스는 기존 OpenGIS Simple Features Specification for OLE/COM 명세와 동일하다.

3.3 GDOM의 구성도

OpenGIS Simple Features Specification for OLE/COM 명세를 기반으로 표준 DOM을 확장한 GDOM의 구성도는 그림 5와 같다.

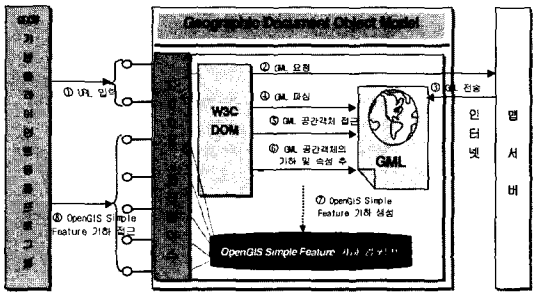

그림 5 GDOM의 구성도

이 구성도에서 클라이언트가 GDOM을 사용하여 맵서버에 URL을 통해 접근한 뒤, OpenGIS 기하 COM 객체의 인터페이스를 획득하는 과정은 다음과 같다.

- URL 입력(①): GDOM 기반의 클라이언트 응용 프로그램은 GDOM의 인터페이스를 호출하여 맵서버로부터 GML을 요청하기 위한 URL을 전달한다.
- GML 요청, 전송 및 파싱(②~④): GDOM은 내부에서 표준 DOM 객체를 생성한 후, 클라이언트로부터 입력받은 URL을 이용하여, 맵서버에 GML을 요청하여 전송 받는다. 그리고 이를 파싱하여 메모리에 올린다.
- GML 공간객체의 접근, 기하 및 속성 추출(⑤, ⑥): 파싱된 GML에 접근하여 공간객체의 기하 및 속성 정보를 검색한 후 추출한다.
- OpenGIS 단순 피처 기하 COM 객체의 생성 (⑦): 추출된 공간객체의 정보를 이용하여 OpenGIS 단순 피처 기하 COM 객체를 생성하여 기하 COM 객체의 인터페이스를 클라이언트에게 제공한다.
- OpenGIS 단순 피처 기하 COM 객체의 접근(⑧): GDOM 기반의 클라이언트 응용 프로그램은 GDOM으로부터 제공되는 OpenGIS 기하 COM 객체의 인터페이스에 접근하여 기하 COM 객체의 정보를 획득하고 공간연산을 수행한다.

3.4 GDOM 명세의 검토

웹 GIS 환경에서 상호운용성을 제공하는 GML에서의 공간연산을 지원하기 위해 본 논문에서 제시하는 GDOM 명세는 다음과 같은 특징을 갖는다. 첫째, GML 데이타에 대한 공간연산을 수행하는 방법을 제시한다. OGC는 웹 매핑과 GML 명세를 통해 웹 환경에서 지리 정보의 발견, 접근 및 출력 방법은 제시하지만 GML 데이타에 대한 공간연산을 수행하는 방법은 제시하지 않는다. 둘째, 본 논문에서 제시하는 GDOM 명세는 OGC의 단순 피처 기하 모델을 기반으로 한다. 즉, 표준화된 단순 피처 기하 모델에서 제시하는 공간연산자의 인터페이스를 지원한다. 따라서 공간연산자에 대한 OGC의 명세가 변경된다면 GDOM 명세도 변경해야 한다. 마지막으로 GDOM 명세는 공간연산을 지원하기 위하여 표준 DOM을 확장한다. 공간연산의 지원은 여러 방법을 이용할 수 있지만, GML이 XML을 기반으로 확장되었기 때문에 XML을 조작하는 표준 DOM을 이용하여 확장하는 것이 가장 효율적이다. 또한 GDOM은 표준 DOM을 확장하였기 때문에 확장된 부분만을 컴포넌트화 할 수 있다. 그러나 위에서 언급한 것과 같이 표준 DOM의 인터페이스가 변경되는 경우 GDOM의 명세도 변경되어야 한다.

표 3 GDOM의 인터페이스 명세

<그룹 1 – General Interface>

```
[ object, uuid(6A124031-FE38-11d0-BECE-00805F7C4268) ]
interface IGeometry : IUnknown{
[propget] HRESULT Dimension([out, retval] long * dimension);
HRESULT Envelope([out, retval] IGeometry** envelope);
HRESULT Extent2D([out] double* minX, [out] double* minY,
                 [out] double* maxX, [out] double* maxY);
};
[ object, uuid(6A124032-FE38-11d0-BECE-00805F7C4268) ]
interface IWks : IUnknown{
HRESULT ImportFromCoord([in]BSTR coord,[in]long* dimension);
};
[ object, uuid(B96EBDD9-B2D6-11d4-8BD9-009027A67FA2) ]
interface IGMLDOMDocument : IUnknown{
HRESULT ImportFromGML([in]BSTR URL);
};
[ object, uuid(6A124033-FE38-11d0-BECE-00805F7C4268) ]
interface IGeometryFactory : IUnknown{
HRESULT CreateFromGML(BSTR URL , IGeometry** geometry);
HRESULT CreateFromCoord
(BSTR coord, long* dimension, IGeometry** geometry);
};
[ object, uuid(6A12403A-FE38-11d0-BECE-00805F7C4268) ]
interface IGeometryCollection : IGeometry{
HRESULT NumGeometryCollection([out,retval] long *numberof);
HRESULT NumGeometries([in]long geomcollection_index,
[out, retval]long* numberof_geometries);
HRESULT Geometry([in]long geomcollection_index, [in]long geom_index,
[out, retval]IGeometry ** geometry);
HRESULT NameGeometryCollection([in]long geomcollection_index,
[out, retval]BSTR* geomcollection_name);
};
```

<그룹 2 – Geometry 관련 Interface>

```
[ object, uuid(6A124035-FE38-11d0-BECE-00805F7C4268) ]
interface IPoint : IGeometry{
HRESULT Coords([out] double* x, [out] double* y);
[propget] HRESULT X([out, retval] double *pVal);
[propget] HRESULT Y([out, retval] double *pVal);
};

[ object, uuid(6A124036-FE38-11d0-BECE-00805F7C4268) ]
interface ICurve : IGeometry{
[propget] HRESULT Length([out, retval] double* value);
HRESULT StartPoint([out, retval] IPoint** sp);
HRESULT EndPoint([out, retval] IPoint** ep);
[propget] HRESULT IsClosed([out, retval] VARIANT_BOOL * isClosed);
HRESULT Value([in] double t, [out, retval] IPoint** p);
};
[ object, uuid(6A124037-FE38-11d0-BECE-00805F7C4268) ]
interface ILineString : ICurve{
[propget] HRESULT NumPoints([out, retval] long *pVal);
HRESULT Point([in]long index, [out,retval]IPoint** point);
};
[ object, uuid(6A124038-FE38-11d0-BECE-00805F7C4268) ]
interface ILinearRing : ILineString{
};
[ object, uuid(6A124039-FE38-11d0-BECE-00805F7C4268) ]
interface ISurface : IGeometry{
[propget] HRESULT Area ([out, retval] double* area);
HRESULT Centroid ([out, retval] IPoint** result);
HRESULT PointOnSurface ([out, retval] IPoint** result);
};
[ object, uuid(6A12403C-FE38-11d0-BECE-00805F7C4268) ]
interface IPolygon : ISurface{
HRESULT ExteriorRing([out, retval] ILinearRing ** exteriorRing);
[propget] HRESULT NumInteriorRings([out, retval] long * count);
HRESULT InteriorRing([in] long index, [out] ILinearRing ** interiorRing);
};
[ object, uuid(6A12403D-FE38-11d0-BECE-00805F7C4268) ]
interface IMultiCurve : IGeometryCollection{
[propget] HRESULT Length ([out, retval] double* length);
[propget] HRESULT IsClosed([out, retval] VARIANT_BOOL * isClosed);
};
[ object, uuid(6A12403F-FE38-11d0-BECE-00805F7C4268) ]
interface IMultiSurface : IGeometryCollection{
[propget] HRESULT Area ([out, retval] double* area);
HRESULT Centroid ([out, retval] IPoint** result);
HRESULT PointOnSurface ([out, retval] IPoint** result);
};
```

<그룹 3 – Relationship관련 Interface>

```
[ object, uuid(6A124040-FE38-11d0-BECE-00805F7C4268) ]
interface ISpatialRelation : IUnknown{
HRESULT Equals ([in] IGeometry* other, [out, retval] VARIANT_BOOL * equals);
HRESULT Touches ([in] IGeometry* other, [out, retval] VARIANT_BOOL * touches);
HRESULT Contains([in] IGeometry* other, [out, retval] VARIANT_BOOL * contains);
HRESULT Within ([in] IGeometry* other, [out, retval] VARIANT_BOOL * within);
HRESULT Disjoint([in] IGeometry* other, [out, retval] VARIANT_BOOL * disjoint);
HRESULT Crosses ([in] IGeometry* other, [out, retval] VARIANT_BOOL * crosses);
HRESULT Overlaps([in] IGeometry* other, [out, retval] VARIANT_BOOL * overlaps);
HRESULT Intersects([in] IGeometry* other, [out, retval] VARIANT_BOOL * overlaps);
};
[ object, uuid(6A124042-FE38-11d0-BECE-00805F7C4268) ]
interface ISpatialOperator : IUnknown{
HRESULT Distance([in] IGeometry* other, [out, retval] double *distance);
HRESULT Boundary([out, retval] IGeometry** boundary);
HRESULT Buffer([in] double distance,[out, retval] IGeometry** result);
HRESULT ConvexHull([out, retval] IGeometry** result);
HRESULT Intersection([in] IGeometry* other,[out] IGeometry** result);
HRESULT Union([in] IGeometry* other,[out] IGeometry** result);
HRESULT Difference([in] IGeometry* other,[out] IGeometry** result);
HRESULT SymmetricDifference([in] IGeometry* other,[out] IGeometry** result);
};
```

4. GDOM의 구현

4.1 GDOM을 이용한 공간연산

GDOM을 이용하여 GML로 표현된 공간객체에 대한 공간연산을 수행하기 위해서는 OpenGIS 기하 COM 객체의 생성, OpenGIS 기하 COM 객체의 접근, OpenGIS 기하 COM 객체간의 공간연산 과정이 필요하다. 즉, GML로 표현된 공간객체간의 공간연산을 수행하기 위해서는, 연산의 대상이 되는 객체를 GML 문서로부터 생성하고 초기화한 후, 해당 COM 객체로부터 공간연산 인터페이스의 포인터를 반환 받아 호출하여 연산을 수행한다. 그림 6은 이러한 과정을 통해 두 GeometryCollection을 대상으로 GDOM의 ISpatialRelation 인터페이스의 Intersects 연산을 수행하기 위한 클라이언트 프로그램의 예제 코드의 일부이다.

```
VARIANT_BOOL result;
IGeometry* pIGeometry_this = NULL;
IGeometry* pIGeometry_other = NULL;
ILinearRing* pILinearRing = NULL;
IPoint* pIPoint = NULL;
ISpatialRelationPtr m_pISpatialRelation;

return_pIGeometryCollection->NumGeometries(geom_index,&numgeometries_this) ;
for(int i=0 ; i < numgeometries_this ; i++){
  return_pIGeometryCollection->Geometry(geom_index , i , &pIGeometry_this),
  m_pISpatialRelation = pIGeometry_this;
  return_pIGeometryCollection->NumGeometries(geom_index,&numgeometries_other) ;
      for(int j=0 ; j < numgeometries_other ; j++){
        return_pIGeometryCollection->Geometry(geom_index , j , &pIGeometry_other);
        result = m_pISpatialRelation->Intersects(pIGeometry_other);
      }
```

그림 6 OpenGIS 기하 COM 객체 간의 공간연산 예제 코드

4.1.1 OpenGIS 기하 COM 객체의 생성 시나리오

GDOM이 GML로부터 OpenGIS 기하 COM 객체를 생성하는 과정은 그림 7과 같다.

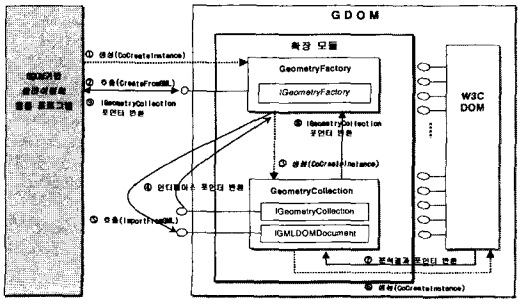

그림 7 OpenGIS 기하 COM 객체의 생성

- GeometryFactory 생성(①): 클라이언트는 CoCreate Instance 함수를 호출하여 GDOM의 Geometry Factory COM 객체를 생성한 후에 IGeometry Factory 인터페이스를 획득한다.
- IGeometryFactory의 CreateFromGML 호출(②): 맵서버에 지도를 요청하기 위한 URL을 매개변수로 GDOM에게 넘기고 결과로서 OpenGIS 기하 COM 객체인 GeometryCollection COM 객체의 포인터를 반환한다.
- GeometryCollection 생성(③, ④): GDOM 내부의 GeometryFactory COM 객체에서는 GML을 분석한 후 공간객체들을 저장하기 위한 Geometry Collection COM 객체를 생성하고IGMLDOM Document 인터페이스를 획득한다.
- IGMLDOMDocument의 ImportFromGML 호출(⑤~⑨): GeometryFactory로부터 전달 받은 URL을 이용하여 GML 문서를 전송 받는다. 그리고 DOM 컴포넌트를 이용하여 추출한 공간객체를 저장하고, Geometry Factory에게 분석 결과의 포인터를 반환한다. Geometry Factory는 GDOM 내부에서 생성된 GeometryCollection의 포인터를 클라이언트에게 반환한다.

4.1.2 OpenGIS 기하 COM 객체의 접근 시나리오

1) Point

GDOM이 GML로부터 생성한 OpenGIS 기하 COM 객체의 타입이 Point일 경우 OpenGIS Point COM 객체에 접근하여 좌표를 획득하는 과정은 그림 8과 같다.

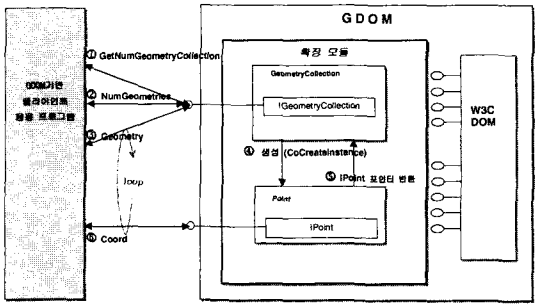

그림 8 OpenGIS 기하 COM 객체의 접근(Point의 경우)

세부 획득과정을 설명하면 다음과 같다.

- GetNumGeometryCollection(①): GDOM으로부터 반환 받은 GeometryCollection의 포인터를 이용하여 GML을 구성하는 GeometryCollection의 개수를

획득한다.

- NumGeometries(②): GDOM으로부터, 공간연산의 대상인 GeometryCollection이 유지하는 공간객체의 총 개수를 획득한다.
- Geometry(③~⑤): 인덱스에 해당하는 Point COM 객체에 대한 포인터를 획득한다.
- Coord (⑥): Point COM 객체에 대한 포인터를 이용해 Point의 좌표를 획득한다.

2) LineString

GDOM이 GML로부터 생성한 OpenGIS COM 객체의 타입이 LineString일 경우, OpenGIS LineString COM 객체에 접근하여 좌표를 획득하는 과정은 그림 9와 같다.

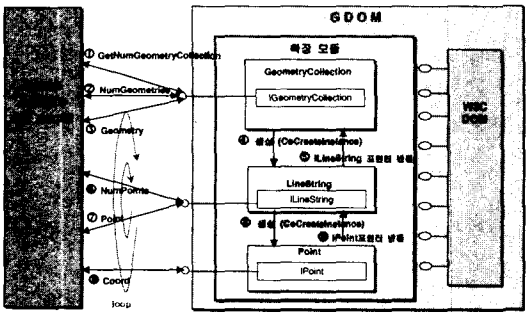

그림 9 OpenGIS 기하 COM 객체의 접근(LineString의 경우)

세부 획득과정을 설명하면 다음과 같다.

- GetNumGeometryCollection, NumGeometries, Geometry(①~⑤): 해당 OpenGIS COM 객체의 타입이 Point의 경우와 동일하다.
- NumPoints(⑥): LineString COM 객체를 구성하는 Point의 개수를 획득한다.
- Point(⑦~⑨): 인덱스에 해당하는 Point COM 객체에 대한 포인터를 획득한다.
- Coord(⑩): Point COM 객체에 대한 포인터를 이용해 LineString의 좌표를 획득한다.

3) Polygon

GDOM이 GML로부터 생성한 OpenGIS COM 객체의 타입이 Polygon일 경우 OpenGIS Polygon COM 객체에 접근하여 좌표를 획득하는 과정은 그림 10과 같다.

세부 획득과정을 설명하면 다음과 같다.

- GetNumGeometryCollection, NumGeometries, Geometry(①~⑤): 해당 OpenGIS COM 객체의 타입이 Point의 경우와 동일하다.

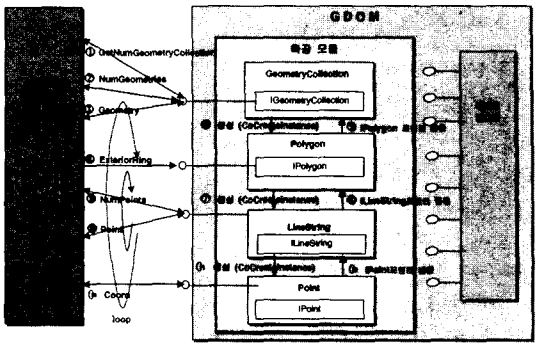

그림 10 OpenGIS 기하 COM 객체의 접근(Polygon의 경우)

- ExteriorRing(⑥~⑧): Polygon COM 객체를 구성하는 LinearRing COM 객체의 포인터를 획득한다.
- NumPoints(⑨): LinearRing COM 객체를 구성하는 Point의 개수를 획득한다.
- Point(⑩~⑫): 인덱스에 해당하는 Point COM 객체에 대한 포인터를 획득한다.
- Coord(⑬): Point COM 객체에 대한 포인터를 이용해 Polygon의 좌표를 획득한다.

4.1.3 OpenGIS 기하 COM 객체 간의 공간연산 시나리오

GDOM이 GML로부터 생성한 OpenGIS Polygon COM 객체에 접근하여 획득한 좌표 정보를 이용하여 공간연산을 수행하는 과정은 그림 11과 같다.

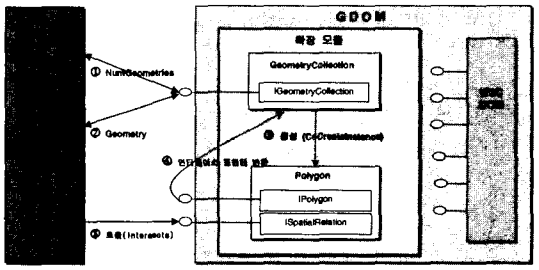

그림 11 OpenGIS 기하 COM 객체간의 공간연산

세부 획득과정을 설명하면 다음과 같다.

- NumGeometries(①): GDOM으로부터 반환 받은 GeometryCollection의 포인터를 이용하여 공간연산의 대상이 되는 GeometryCollection에 포함된 전체 공간객체의 수를 획득한다.
- Geometry(②~④): GDOM은 공간연산의 대상이 되는 GeometryCollection이 유지하는 공간객체를 내부에서

OpenGIS 기하 COM 객체의 타입으로 생성한 후, 클라이언트에게 해당 OpenGIS 기하 COM 객체의 포인터를 반환한다. 그림 11에서는 공간연산의 대상이 되는 GeometryCollection의 타입이 Polygon인 경우이다. 이 경우에는 Polygon COM 객체의 IPolygon 인터페이스의 포인터를 클라이언트로 반환한다.

- Intersects(⑤): GDOM으로부터 획득한 IPolygon 인터페이스의 포인터로 *ISpatialRelation* 인터페이스의 포인터를 획득하여 intersect 메소드를 호출해 공간연산을 수행한다.

4.2 구현 예제

GDOM은 COM 컴포넌트로 구현되어 클라이언트 응용에게 DLL로 제공되며 내부 인터페이스의 구현에 사용되는 표준 DOM 컴포넌트는 Microsoft사의 MSXML을 이용한다*[12]*. *MSXML* 파서는 *W3C DOM Level* 1의 명세를 기반으로 Microsoft사에서 구현한 COM 컴포넌트이며, DOM 명세에서 지원하지 않는 XML 문서의 전송, 저장 등 실제 응용에 필요한 내용을 부가적으로 구현하고 있다. GDOM을 이용하는 클라이언트의 어플리케이션은 Windows98 환경에서 ActiveX 컨트롤로 구현하며, 클라이언트에게 GML을 제공하는 맵서버는 Windows NT 환경에서 Active Server 컴포넌트로 구현한다. 구현에 사용하는 지도 데이타는 상용 GIS 데이타 서버인, MGE 데이타 서버와 사이버맵 데이타 서버의 도시 데이타이며 클라이언트 질의 결과를 출력하기 위한 웹 브라우저는 IE 5.0을 이용한다. 그림 12는 클라이언트가 상용 GIS 데이타 서버인 사이버맵 데이타 서버의 지도를 요청하기 위해 GDOM의 인터페이스를 호출한 후, GDOM이 GML로부터 생성한 OpenGIS 기하 COM 객체에 접근하여 부산시 영도구의 도시지역 데이타를 웹 브라우저인 IE 5.0에 출력한 화면이다.

그림 12 GDOM을 이용한 사이버맵 데이타 서버의 공간 데이타 출력

그림 13은 그림 12의 지도를 대상으로, GDOM의 공간연산 인터페이스인 ISpatialOperator::Buffer와 ISpatial Relation::Within을 이용하여 공간연산을 수행하기 위해 point 객체를 입력하고, 그 지점을 중심으로 반경 200m 내의 교회 레이어에 속하는 객체를 GDOM을 이용하여 추출한 결과 화면이다.

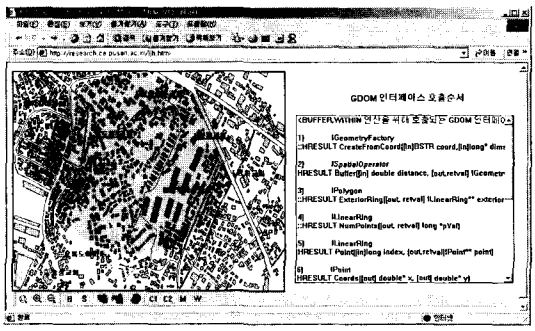

그림 13 공간연산의 결과(Buffer, Within)

GDOM은 상호 운용을 지원하는 컴포넌트이므로 특정 데이타 서버에 종속적이지 않다. 이를 입증하기 위해 사이버맵 데이타 서버와 다른, MGE 데이타 서버로부터 전송받은 GML 데이타에 대해서 GDOM을 이용하여 ISpatialRelation::Intersects 연산을 수행하였다. 이를 위해, GDOM을 이용하여 그림 14와 같이 MGE 데이타 서버의 공간객체를 맵서버로부터 GML 형식으로 전송받아 웹 브라우저에 출력하였다.

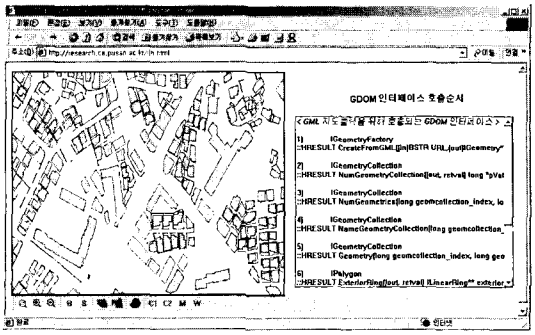

그림 14 GDOM을 이용한 MGE 데이타 서버의 공간 데이타 출력

그림 15는 GDOM의 ISpatialRelation::Intersects 연산자를 이용하여 클라이언트가 선택한 영역과 교차하는 일반건물 객체들을 추출한 결과 화면이다.

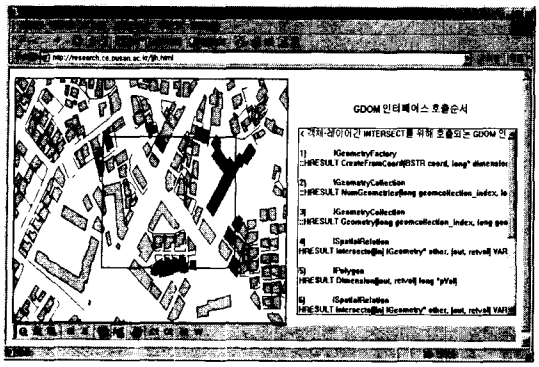

그림 15 공간연산의 결과(Intersect)

5. 결론 및 향후 연구

웹 GIS 환경의 상호 운용을 제공하기 위한 OGC의 웹 매핑 테스트베드에서는 맵서버와 GML 명세를 제안하여 웹 GIS 환경에 분산 저장된 GIS 정보의 발견, 접근, 통합 및 출력 방법은 제공하지만 GIS에서 필수적인 공간연산을 수행하는 방법은 제공하지 않는다. 본 논문에서는 웹 환경의 다양한 데이타 서버들로부터 중첩되어 전송된 GML데이타에 대한 공간연산을 수행하는 방법을 제시하였다. 이를 위해 W3C에서 XML 문서의 분석을 위해 제시한 DOM과 OGC에서 개방형 GIS 시스템의 구현을 위해 제시한 OpenGIS 단순 피처 명세를 기반으로, 웹 GIS 환경에서 GML 데이타에 대한 공간연산을 지원하는 GDOM을 설계하고 구현하였다.

향후 연구과제로는 GDOM 컴포넌트 사용에 의해 지도출력 및 공간연산의 속도가 저하되는 문제점을 고려하여 공간 인덱싱 기법을 이용한 GDOM의 성능 개선 방안의 연구가 필요하다.

참 고 문 헌

[1] OGC Web Mapping Testbed Public Page, http://www.opengis.org
[2] Web Map Server Interfaces Implementation Specification Revision 1.0.0, http://www.opengis.org/techno/specs.htm
[3] Extensible Markup Language(XML) 1.0, http://www.w3c.org/TR/REC-xml
[4] Geography Markup Language-Recommendation (GML), http://www.opengis.org/specs.htm
[5] Document Object Model (DOM), http://www.w3.org/DOM/
[6] MathML(Mathematical Markup Language), http://www.w3.org/Math/
[7] MathML DOM(Mathematical Markup Language Document Object Model), http://www.w3.org/TR/MathML2/appendixd.html
[8] CML(Chemical Markup Language), http://www.xml-cml.org/
[9] CML DOM(Chemical Markup Language Document Object Model), http://www.xml-cml.org/
[10] SVG(Scalable Vector Graphics), http://www.w3.org/Graphics/SVG/Overview.htm8
[11] SVG DOM(Scalable Vector Graphics Document Object Model), http://www.w3.org/TR/SVG/svgdom.html
[12] MicroSoft MSXML, http://msdn.microsoft.com/xml/default.asp
[13] ScoreML, http://caristudenti.cs.unibo.it/ids/gruppo_7/doc/displet/dtd/index.html
[14] OpenGIS Consortium, OpenGIS Simple Features Specification for OLE/COM Revision 1.1, 1999.
[15] 정소영, 홍은지, 유석인, "OpenGIS 공간연산자 구현시의 모호성 분석", 한국정보과학회 가을 학술발표논문집, Vol.25, No.2, pp.105-107, 1998.
[16] 윤우진, 조대수, 홍봉희, "OLE/COM을 기반으로 한 OpenGIS 미들웨어 설계", 개방형지리정보시스템 학술회의 논문집, Vol. 2 , No. 2, pp.95-106, 1999.

반 재 훈
1997년 2월 부산대학교 컴퓨터공학과 공학사. 1998년 2월 부산대학교 컴퓨터공학과 공학석사. 2001년 부산대학교 컴퓨터공학과 박사과정 수료. 2002년 ~ 현재 경남정보대학 인터넷상거래과 교수. 관심분야는 이동객체, 지리정보시스템(GIS)

조 정 희
1999년 2월 부산대학교 컴퓨터공학과 공학사. 2001년 2월 부산대학교 컴퓨터공학과 공학석사. 2001년 1월 ~ 현재 LG 전자 차세대단말연구소 무선인터넷팀 주임연구원. 관심분야는 무선인터넷, 위치정보서비스, GPS, 지리정보시스템(GIS)

문 상 호

1991년 2월 부산대학교 컴퓨터공학과 공학사. 1991년 ~ 1992년 한국기계연구원 전산시스템실 연구원. 1994년 2월 부산대학교 컴퓨터공학과 공학박사. 1998년 3월 ~ 2002년 8월 위덕대학교 컴퓨터멀티미디어공학부 교수. 2002년 9월 ~ 현재 부산외국어대학교 컴퓨터전자공학부 교수. 관심분야는 지리정보시스템(GIS), 공간뷰, 객체지향데이타베이스, GIS 표준화, 정보시스템 감리

홍 봉 희

1982년 서울대학교 전자계산기공학과졸업(공학사). 1984년 서울대학교 대학원 전자계산기공학과 졸업(공학사). 1988년 서울대학교 대학원 전자계산기공학과졸업(공학박사). 현재 부산대학교 공과대학 컴퓨터공학과 정교수. 관심분야는 병렬공간 DB, 분산공간 DB, 개방형 GIS