OpenGIS의 Simple Feature Geometry컴포넌트 구현+

(An Implementation of Simple Feature Geometry Component of OpenGIS)

황 정래*, 강 혜경*, 강 인수**, 이 기준***
(Hwang JungRae*, Kang HaeKyong*, Kang InSoo**, Li KiJoune***)

초 록

지리 정보 시스템을 위한 공간 데이터 베이스 관리시스템 간의 이질성을 최소화하고 공간 데이터의 호환성이나 상호 연동성을 높일 수 있도록 OGC에서 OpenGIS의 사양을 정의하였다. 본 연구에서는 OpenGIS에서 정의된 공간 데이터 모델을 지원하는 컴포넌트를 구현한 것이다. 즉, OpenGIS에서 요구되어진 사양에 따라 SFG(Simple Feature Geometry) 컴포넌트를 구현하였다. 본 논문에서는 컴포넌트 구현상의 발생되었던 문제점과 이 해결 방법을 제시한다. 또한 구현 도중에 발견된 OpenGIS SFG의 사양 자체의 문제점도 함께 제시한다.

키워드

지리정보시스템, 컴포넌트, , OGC, OpenGIS, Simple Feature Geometry, Dimensionally Extended 9-Intersection Model

+본 연구는 ETRI 연구과제로부터 지원받음.
* 부산대학교 지형정보협동과정
** 한국통신정보기술
*** 부산대학교 전자계산학과

1. 서론

현재, 지리 정보 시스템 및 공간 데이터 베이스 관리 시스템의 기술은 통합된 대형시스템에서 컴포넌트로 구성된 시스템으로 변화하여 가는 패러다임 변화기에 해당한다. 시대적 변화에 대응하기 위해 많은 지리정보시스템(GIS: Geographic Information Systems) 생산업체와 연구기관에서는 패러다임 변화기에서 요구되는 기술을 선점하려고 노력하고 있다.

이 기술 중에서 가장 중요한 것 중의 하나는, 지리 정보 시스템에서 요구되는 공간 데이터의 관리 기능을 컴포넌트화하는 것이다. 이 기능은 세계적으로 요구되고 있는 것일 뿐 만 아니라, 국내 지리 정보 시장을 위해서도 반드시 필요한 부분이다. 이러한 요구를 위하여 세계적인 주요 GIS 생산기관 및 연구소, 학교를 회원으로 하는 OGC(OpenGIS Consortium)[1]이 발족하여 OpenGIS라는 사양을 발표하였다. OpenGIS에 의하면, GIS가 컴포넌트화된 여러 개의 구성요소로 나누어진다. 그 중에서 단순화된 공간 데이터 모델을 통하여 기

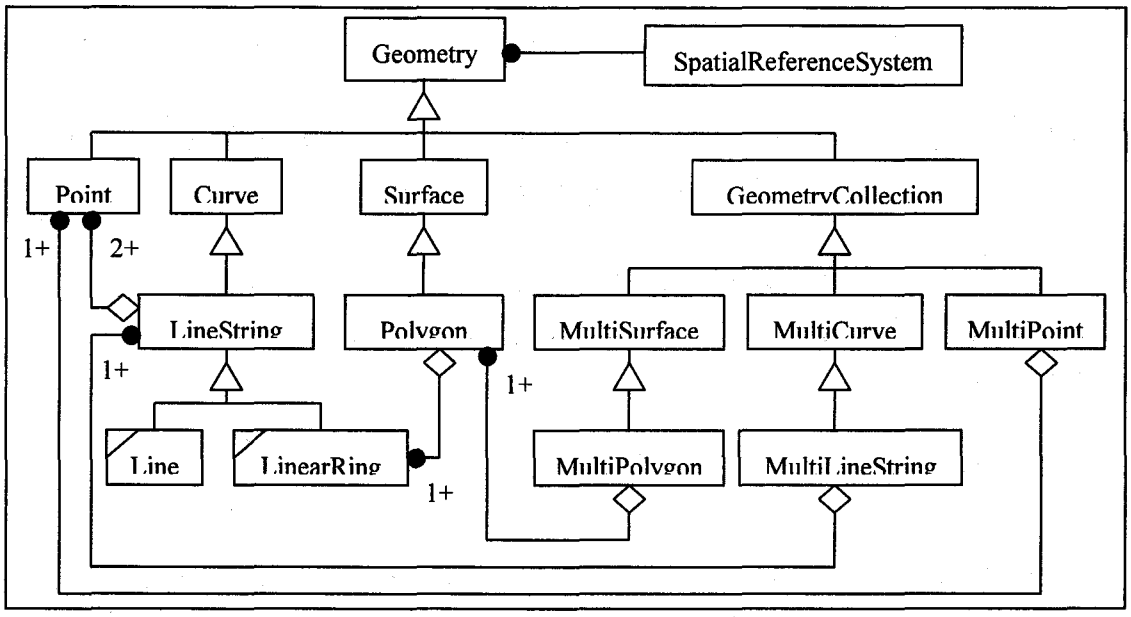


그림 1. Simple Feature Geometry의 기하 객체 모델

하학적인 연산을 처리하는 컴포넌트가 있는데, 이를 Simple Feature Geometry(SFG)라고 한다. 이 컴포넌트는 GIS를 구현하는데 매우 중요한 것으로서 독립된 컴포넌트로 구성되었을 때, 상당한 재사용의 이득을 얻을 수 있다.

본 연구에서는 OpenGIS 사양을 바탕으로 SFG 컴포넌트를 구현한다. 이 컴포넌트는 기하 객체 모델(Geometry Object Model)에서 정의된 공간 기하 객체들에 대해 공간 연산 기능을 제공한다. 본 논문에서는 SFG를 구현하면서 발생되었던 문제점과 해결책을 제안한다. 또한 OpenGIS 사양에 정의된 원칙을 따르면 틀린 결과가 제시되는 경우가 발견되었는데, 이를 기술함으로써 사양의 보완점을 제시하고자 한다.

이를 위해서 OpenGIS에서 제시하는 기하 객체 모델과 SFG 컴포넌트 사양을 조사하여 2장에서 기술한다. 3장에서는 SFG 구현 흐름도를 제시하고, 구현 내용과 방법 그리고 구현 결과도 함께 제시하며, 4장에서는 구현상의 문제점과 해결 방법도 제시한다. 마지막으로 5장에서는 결론을 제시한다.

2. OpenGIS Simple Feature Geometry 구현명세

본 장에서는 본 논문에서 구현하기 위해 기본이 된 OpenGIS에서 제안하는 SFG 모델 및 컴포넌트 명세에 대해 간단히 살펴본다.

2.1 기하 객체 모델 (Geometry Object Model)

SFG 컴포넌트에 대한 사양을 정의하기 위해서는 우선 기본적인 공간 데이터 모델이

제시되어야 한다. OpenGIS에서는 이를 위하여 아래 그림 1과 같이, 기하 객체 모델을 제시하였다. 아래의 그림에서 공간참조 시스템(*SpatialReferenceSystem*)을 제외한 클래스 (또는 기하 객체)들은 실제 사용자가 사용 가능한 공간 데이터 타입이며 사용자들에게 공간 데이터에 대한 인터페이스를 제공한다. 각각의 기하 객체들을 간단히 살펴보면 다음 표 1과 같다[3].

기하객체	차원	내용
Point	0	좌표 공간상에 단일 위치로 표현된다.
Curve	1	순서가 있는 점들의 집합으로 표현된다.
Line	1	시작점과 끝점을 가지는 단일 선 객체로 표현된다.
LineString	1	연결된 하나 이상의 Line 세그먼트로 표현된다.
Surface	2	면적을 가지는 2차원 기하 객체이면서, 2.5차원 기하 객체도 포함한다.
Polygon	2	단순하면서 위상적으로 닫힌 면으로 표현된다.
LinearRing	1	닫혀 있는 LineString 객체이다.
MultiPoint	0	순서화 되지 않은 Points의 집합으로 표현된다.
MultiLineString	1	LineStrings의 집합으로 표현된다.
MultiPolygon	2	Polygon의 집합으로 표현된다.
GeometryCollection	0 이상	하나 이상의 기하 객체의 집합으로 표현된다.

표 1. 기하 객체들에 대한 설명

2.2 SFG 컴포넌트 구현 명세

OpenGIS에서는 SFG에 대한 구현 명세를 인터페이스 형태로 제공한다. 인터페이스는 컴포넌트 개체가 사용자에게 자신의 기능을 노출시키는 방법이다. 구현 명세에서 제시하는 인터페이스의 종류는 표 2와 같다[3].

ISpatialOperator, ISpatialRelation, IspatialRelation2, IWks, IGeometryFactory 를 제외한 인터페이스들은 SFG의 데이터 모델의 기하 객체들과 일대일 관계를 가진다. *ISpatialOperator, ISpatialRelation, IspatialRelation2* 인터페이스는 기하객체의 지원이 필요하다. *IWks*는 기하객체와는 직접 관계가 없는 내부 자료구조 변환을 위한 인터페이스이다. *IGeometryFactory* 역시 기하 객체와는 직접 관계가 없이 기하객체를 생성하는 기능을 한다. 그리고, 위의 표2에서 제시한 각각의 인터페이스가 모여서 하나의 컴포넌

트를 구성한다. 이러한 컴포넌트를 SFG 컴포넌트라고 한다. 다음 절에서는 SFG 컴포넌트의 구현 방법에 대하여 설명한다.

3. Simple Feature Geometry 컴포넌트 구현

아래 그림 2는 SFG 컴포넌트의 인터페이스인 *SpatialRelation, SpatialOperator* 가 생성되는 과정의 흐름도를 간단히 제시하였다.

공간 데이터 인터페이스	상위 인터페이스	설명
IGeometry	없음	공간 기하 객체 최상위 인터페이스
IGeometryFactory	없음	공간 기하 객체를 WKB(WKT)를 이용하여 생성하기 위한 인터페이스
IWks	없음	공간 기하 객체와 WKB(WKT)간의 변환을 위한 인터페이스
IPoint	IGeometry	점 기하 객체에 대한 인터페이스
ICurve	IGeometry	선 기하 객체에 대한 인터페이스
ILineString	ICurve	선 기하 객체에 대한 인터페이스
ILinearRing	ILineString	면 기하 객체를 구성하는 선 기하 객체 인터페이스
ISurface	IGeometry	면 기하 객체에 대한 인터페이스
IPolygon	ISurface	면 기하 객체에 대한 인터페이스
IGeometryCollection	IGeometry	집합 기하 객체를 위한 인터페이스
IMultiPoint	IGeometryCollection	점 집합 기하 객체를 위한 인터페이스
IMultiCurve	IGeometryCollection	선 집합 기하 객체를 위한 인터페이스
IMultiLineString	IMultiCurve	선 집합 기하 객체를 위한 인터페이스
IMultiSurface	IGeometryCollection	면 집합 기하 객체를 위한 인터페이스
IMultiPolygon	IMultiCurve	면 집합 기하 객체를 위한 인터페이스
ISpatialOperator	없음	기하연산, 집합연산을 위한 인터페이스
ISpatialRelation	없음	위상연산을 위한 인터페이스
ISpatialRelation2	없음	위상연산을 위한 보조 인터페이스

표 2. SFG 컴포넌트에 대한 인터페이스

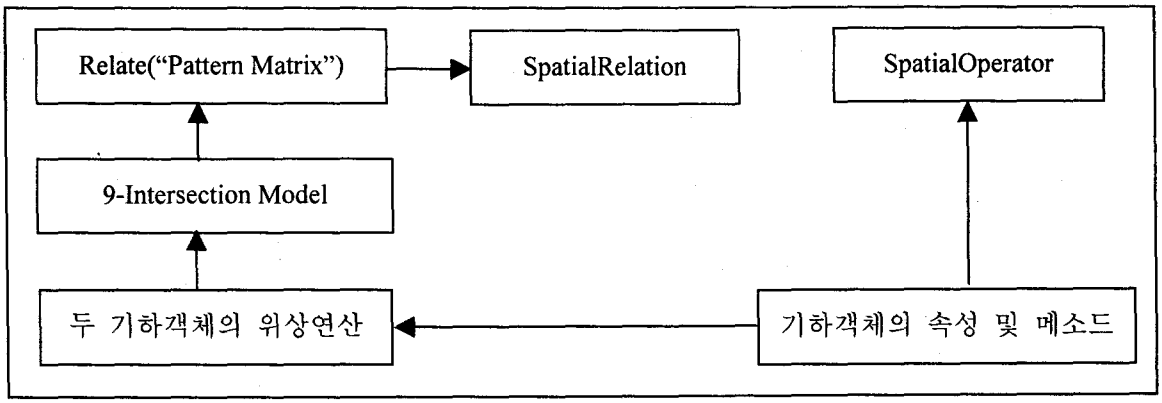


그림 2. 공간 연산자 생성과정 흐름도

3.1 구현 내용

OpenGIS에서는 각 기하 객체에 대하여 표 3에서 요약한 세가지 종류의 인터페이스를 정의하고 있다. 표 3의 공간연산자 (*ISpatialOperator*)와 두 가지 공간관계연산자 (*ISpatialRelation, IspatialRelation2*) 인터페이스의 기능은 선택적으로 제공 가능하지만, 본 논문에서는 모두 구현하였으며, 기능의 종류는 표 3에 구체적으로 나열되어 있다.

SpatialOperator	SpatialRelation	SpatialRelation2
Boundary	Contains	Relate
Buffer	Cross	
Convexhull	Disjoint	
Distance	Equal	
Intersection	Intersect	
Difference	Overlap	
SymmetricDifference	Touch	
Union	Within	

표 3. 공간 연산 기능 사항

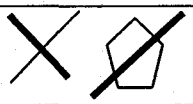
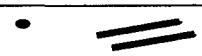
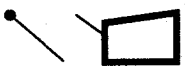
위상 관계	설 명	예
Contains	주어진 기하 객체에 다른 기하 객체가 완전히 포함되는 것을 말한다.	
Within	Contains와 반대되는 개념이다.	
Cross	두 기하 객체가 교차는 것으로, 두 기하 객체의 최대 차원보다 작은 기하 객체 값을 가진다.	
Disjoint	두 기하 객체가 완전히 떨어져 있는 것을 말한다.	
Equal	두 기하 객체가 완전히 동일한 것을 말한다.	
Overlap	두 기하 객체가 교차하는 것으로, 주어진 기하 객체와 동일한 차원을 가진다.	
Touch	두 기하 객체간에 만나는 점들로, 그들의 Boundary에 놓인다.	
Intersect	Disjoint을 제외한 모든 경우를 말한다.	Disjoint의 반대 개념

표 4. 기하 객체 사이의 위상 관계

우선, OpenGIS에서는 8가지의 공간연산자(*ISpatialOperator*)를 정의하여 각 기하객체에 대하여 인터페이스를 제공하도록 명시되어 있다. 이들 공간연산자에 대한 자세한 설명은 3.4.2절에서 설명한다. 또한 OpenGIS에서는 공간관계연산자(*Ispatial-Relation*)를 8가지인 포함(Contains), 피포함(Within), 교차(Cross), 분리(Disjoint), 동등(Equal), 중복(Overlap), 접촉(Touch), 비분리(Intersect)로 구별하여 각각의 인터페이스에서 구현할 기하 객체 사이의 위상 관계를 정의하였는데, 이를 간단히 설명하면 위의 표 4와 같다.

공간관계연산자중에 가장 많이 사용되는 것을 *ISpatialRelation*로 정의하였다면, OpenGIS사양에서는 나머지 일반적인 공간관계연산자를 *ISpatialRelation2* 이라는 다른 종류의 그룹으로 분류하였다. 이 공간관계연산자를 위한 인터페이스의 결과는 차원 확장된 9-교차모델(Dimensionally Extended 9-Intersection Model)의 행렬값으로 된다. 차원 확장된 9-교차모델에 대한 설명은 다음 절에서 설명하기로 한다.

3.2 공간연산자 및 공간관계연산자

기하 객체에 공간관계연산자를 적용시키기 위해서는 공간 기하 객체의 관계를 정의하는 기준이 있어야 한다. 공간 데이터 모델에서 가장 많이 사용되는 방법 중에 9-교차모델(9-Intersection Model)이 있

다.[10] 9-교차모델은 공간객체의 내부/경계/외부가 교차하는지 여부를 이진값(0/1)으로 표현한다. 그리고 공간관계연산자의 결과는 공간객체의 내부/경계/외부의 교차여부를 나타내는 3x3 행렬값으로 표현된다. 예를 들어서 아래의 그림 3과 같은 두개의 기하 객체 사이의 분리(Disjoint)라는 위상관계는 다음과 같이 표현된다[7].

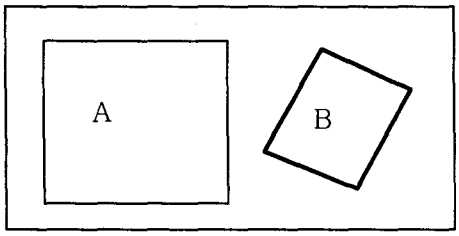


그림 3. 두개의 기하객체사이의 위상관계와 9-교차모델

$$R(A,B) = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

그러나, OpenGIS에서는 9-교차모델의 표현력을 보다 높이기 위하여 두개의 기하객체사이의 위상관계를 차원 확장된 9-교차모델(Dimensionally Extended 9-Intersection Model)로 표현하도록 되어있다. 차원 확장된 9-교차모델에서는 공간객체의 내부/경계/외부가 교차하는지 여부를 이진(0/1)대신 True/False와 기하객체의 차원(0,1,2)으로 표현한다.[3] 결국, 9-교차모델이 차원으로 확장된다는 의미로 차원 확장된 9-교차모델이라고 한다. 차원 확장된 9-교차모델에서 공간관계연산자를 정의하는 방법은 9-교차모델과 동일하다. 즉, 차원확장 9-교차모델에서도 공간연산자는 공간객체의 내부/경계/외부의 교차여부를 나타내는 3x3 행렬로 정의한다. 이렇게 정의된 9개의 값을 패턴 행렬(Pattern Matrix)이라고 부른다. 이 행렬 구성하는 값의 범위는 {T,F,*,0,1,2}이며, 각 값의 의미는 다음과 같다.

T : $dim(x) \in \{0, 1, 2\}$, i.e. $x \neq \varnothing$
F : $dim(x) = -1$, i.e. $x = \varnothing$
$*$: $dim(x) \in \{-1, 0, 1, 2\}$
0 : $dim(x) = 0$
1 : $dim(x) = 1$
2 : $dim(x) = 2$

예를 들어, 그림 3과 같이 두개의 기하객체 A, B사이의 분리(Disjoint)라는 위상관계를 나타내는 공간관계연산자는 다음 표 5와 같이 나타나며, 표현은 아래와 같다.

A \ B	내부	경계	외부
내부 (Interior)	F	F	2
경계 (Boundary)	F	F	1
외부 (Exterior)	2	1	2

표 5. 차원 확장 9-교차모델의 예

A.Disjoint(B)
$\Leftrightarrow (I(A) \cap I(B) = \varnothing) \wedge (I(A) \cap B(B) = \varnothing) \wedge (B(A) \cap I(B) = \varnothing) \wedge (B(A) \cap B(B) = \varnothing)$

⇔ A.Relate(B, "FF*FF****")

나머지 공간관계연산자 역시 위와 같이 표현하였으며, 구현한 결과의 행렬값은 3.4.1에서 보여준다.

3.3 구현 방법

본 장에서는 앞 절에서 설명한 OpenGIS 사양을 본 연구에서 구현할 때 발생되는 문제와 해결방법, 구현결과에 대하여 살펴본다.

3.3.1 공간 기하 객체 사이의 위상 관계

다음 표 6은 본 연구에서 공간 기하 객체 사이의 위상 관계를 나타내기 위해서 패턴 행렬을 구현한 결과이다. 이 표 6에는 단순한 단일 점, 선 그리고 면 객체에 대한 관계만을 나타내었다. 기하 객체 모델에는 단일 기하 객체들 뿐 아니라, 복합 기하 객체(Geometry Collection)에 대한 객체들도 정의되어 있다. 이들 복합 기하 객체들은 단일 객체집합으로 변경되므로, 복합 기하 객체들 사이의 위상관계는 단일 기하 객체 사이의 위상관계로부터 유도될 수 있다. 예를 들어서, 그림 4와 같이 단일 기하 객체 A는 복합 기하 객체 B1과 분리(Disjoint)하고, 복합 기하 객체 B2와도 분리(Disjoint)한다는 것을 알 수 있다.

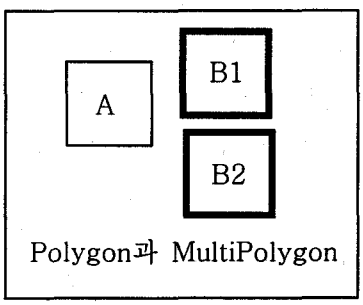


그림 4. 복합기하객체의 공간위상관계의 예

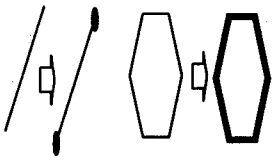
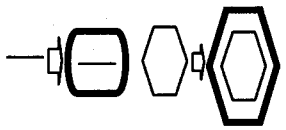
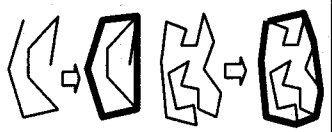
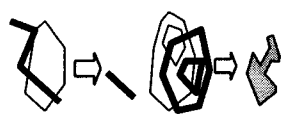
따라서, 본 연구에서는 우선 단일 기하 객체 사이의 위상관계를 위한 공간관계연산자를 구현하고, 이를 이용하여 복합 기하 객체사이의 위상관계를 계산하는 공간관계연산자를 구현하였다. 물론, 성능상의 문제를 고려하면, 단일 기하 객체의 공간관계연산자를 이용하지 않고 직접 구현하는 것이 좋을 수 있다. 그러나, 전체적인 시스템의 단순성을 위하여, 단순 기하 객체의 공간관계연산자를 이용하였고, 성능상의 문제를 보완하기 위해서는 이를 직접 구현한 모듈로 쉽게 교환이 가능하다.

기하객체	공간관계연산자	기하객체		
		Point	**LineString**	**Polygon**
Point	Contains	"0** *** ***"	False	False
	Cross	False	False	False
	Disjoint	"F** *** ***"	"FF* *** ***"	"FF* *** ***"
	Equal	"0** *** ***"	False	False
	Intersect	! Disjoint	! Disjoint	! Disjoint
	Overlap	"0** *** ***"	False	False
	Touch	False	"F0* *** ***"	"F0* *** ***"
	Within	"0** *** ***"	"0** *** ***"	"0** *** ***"
LineString	Contains	"0** *** ***"	"10* F** ***"	False
	Cross	False	"0** *** ***"	"10* *** ***"
	Disjoint	"F** F** ***"	"FF* FF* ***"	"FF* FF* ***"
	Equal	False	"1F* F0* ***"	False
	Intersect	! Disjoint	! Disjoint	! Disjoint
	Overlap	False	"10* 0** ***"	False
	Touch	"F** 0** ***"	"F0* *** ***" "F** 0** ***" "F** *0* ***"	"FT* *** ***" "F** *0* ***"
	Within	False	"1F* 0** ***"	"1F* 0** ***"
Polygon	Contains	"0** *** ***"	"10* F** ***"	"21* F** ***"
	Cross	False	"1** 0** ***"	False
	Disjoint	"F** F** ***"	"FF* FF* ***"	"FF* FF* ***"
	Equal	False	False	"2F* F1* ***"
	Intersect	! Disjoint	! Disjoint	! Disjoint
	Overlap	False	False	"21* 10* ***"
	Touch	"F** 0** ***"	"F** T** ***" "F** *0* ***"	"F** *T* ***"
	Within	False	False	"2F* 1** ***"

표 6. 각 위상관계와 공간관계연산자의 결과 행렬

3.3.2 공간 기하객체 사이의 기하 연산 및 집합 연산

OpenGIS에서는 공간연산자로 다양한 기하 연산 및 집합 연산을 제공하고 있다. 본 연구에서는 여기서 정의되는 모든 공간연산자를 구현하였다. 즉, 경계(Boundary), 완충(Buffer), 거리(Distance), 볼록다각형(ConvexHull), 교집합(Intersection), 차집합(Difference,) 대칭차집합(SymmetricDifference), 합집합(Union)의 모든 공간연산자가 구현되었는데, 다음 표 7은 공간 기하객체 사이의 기하 연산 및 집합 연산 기능에 대해 간단한 설명과 예를 들어 보았다.

연산 기능	설 명	예
Boundary	Boundary는 점 객체는 Boundary가 없고, 선 객체는 그 선의 시작점과 끝점을 나타내며, 면 객체는 면 내부를 뺀 면을 이루는 LinearRing으로 표현한다.	
Buffer	Buffer는 source 기하 객체의 Boundary로부터 주어진 distance내에 있는 모든 점들을 포함하는 Polygon으로 표현되며, 점 객체에서는 Buffer를 원으로 구현하였고, 선 객체는 시작점과 끝점을 처리하기 위해 점 객체에서 Buffer를 만드는 것을 사용하여 처리하였으며, LineString인 경우에는 각 선 객체의 Buffer를 구한 후 Union을 가지고 처리하였다. 그리고, 면 객체는 LineString과 마찬가지로 처리한 반면, 내부 Buffer는 고려하지 않았다.	
Distance	Distance는 source 기하 객체와 주어지는 다른 기하 객체 사이의 최소 거리를 말하며, 구현은 각각 기하 객체의 거리를 비교한 후 가장 작은 값으로 나타내었다. 단, 두 기하 객체 가 교차하는 경우에는 Distance를 0으로 나타내었다.	
ConvexHull	ConvexHull은 source 기하 객체를 바깥으로 둘러싸는 최소 볼록 다각형으로 나타내었다.	
Intersection	Intersection은 source 기하 객체와 주어지는 기하 객체 사이에서 교차하는 부분으로 점이나 선, 또는 면으로 나온 결과를 처리하였다.	

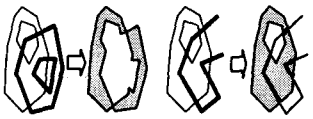
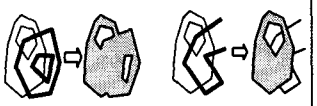
연산 기능	설 명	예
Difference	Difference는 source 기하 객체로부터 주어지는 기하 객체를 뺀 나머지 기하 객체를 나타내었다.	
Symmetric Difference	SymmetricDifference는 source 기하 객체로부터 주어지는 기하 객체를 뺀 나머지 기하 객체와 주어지는 기하 객체에서 source 기하 객체를 뺀 나머지 기하 객체의 합집합을 나타내었다.	
Union	Union은 source 기하 객체와 주어지는 다른 기하 객체사이에서의 합집합 부분을 나타내었다.	

표 7. 공간 기하 객체 사이의 기하 연산 및 집합 연산

4. 문제점과 해결방법

본 절에서는 OpenGIS 사양의 몇 가지 문제점에 대하여 살펴보기로 한다. 이러한 문제점은 사양자체의 문제점과 구현상의 문제점으로 나누어 진다. 이들 문제점은 OpenGIS 사양이 완전하여지기 위하여서는 보완되어야 할 것들이다.

4.1 공간 기하객체 사이의 기하 연산 및 집합 연산에서의 문제점과 해결책

OpenGIS구현사양에 나온 공간연산자는 기하 연산자와 집합 연산자로 나누어진다. 기하 연산자는 주어진 기하 객체의 넓이나, 길이를 구하는 비교적 단순한 것에서부터, 복합 객체의 완충(Buffer)과 같이 비교적 구현이 까다로운 것까지 다양하게 정의되어 있다. 또한 집합 연산자는 주어진 두개의 기하 객체의 교집합, 합집합된 새로운 기하 객체를 구하는 것 등으로 정의되어 있다.

이와 같은 공간연산자를 구현하는데 가장 까다로운 작업은 집합 연산이나 기하 연산의 결과로 생성되는 객체를 하나의 객체로 만들어주는 것이다. 예를 들어, 그림 5와 같이, 두 개의 단순 기하 객체로 구성된 하나의 복합 객체에 대하여 완충 연산을 하려고 한다고 가정하자. 이 경우, 복합 객체를 구성하고 있는 단순 객체에 대하여 완충 연산을 하는 것으로 작업이 끝나는 것이 아니다. 만일 단순 객체에 대한 완충 연산 결과가 분리된 여러 개의 단순 객체로 구성되어 있으면 이들을 하나의 복합 기하 객체로 정의하면 된다. 그러나, 이들이 서로 겹쳐지는 영역이 있으면, 이들을 통합하여 하나의 단순 기하 객체로 만들어 주어야 한다.

그림 5에서와 같이, 두개의 선으로

구성된 복합 객체 A에 대한 완충 연산을 할 경우, 단순 객체에 대한 완충 연산의 결과로 두개의 기하 객체 A_1과 A_2가 만들어진다. 이들 두 객체는 서로 분리되어 있지 않으므로 B라는 하나로 통합된 단순 기하 객체를 만들어주어야 한다.

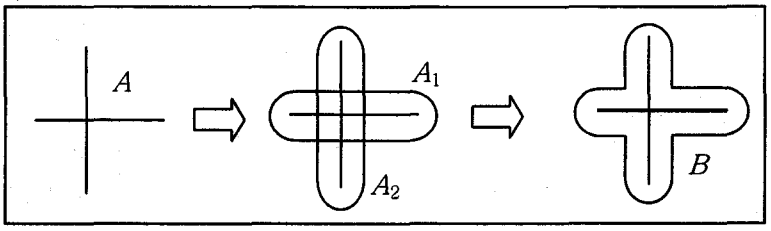


그림 5. 완충 연산을 한 후 합집합으로 처리한 예

본 연구에서는 완충 연산뿐 아니라, 기하 연산이나, 집합 연산의 결과가 마찬가지로 여러 개의 기하 객체로 생성될 경우에도 위에서 설명한 것과 같이, 단순 기하 객체에 대한 연산결과를 하나의 복합 객체나, 하나의 단순 객체로 전환하는 작업을 구현하였다.

4.2 기하 객체 사이의 위상 관계에서 문제점과 해결책

기하 객체 사이의 위상 관계를 구현하면서 발견된 문제점이 크게 세 가지로 구분될 수 있다.

문제 1: OpenGIS SFG 사양에는 명시되어 있는데, 실제로는 나타나지 않는 경우

OpenGIS SFG 사양에는 교차관계가 Point와 Linestring, Point와 Polygon도 다음과 같이 정의가 되어 있다. a가 Point이고, b가 LineString이거나 Polygon일 경우는 다음과 같이 표현된다.

a,Cross(b)

$\Leftrightarrow (I(a) \cap I(b) \neq \varnothing)$

$\wedge (I(a) \cap E(b) \neq \varnothing)$

$\Leftrightarrow$ a.Relate(b, "T*T******")

하지만 본 연구에서 살펴본 바에 의하면, 아래와 같이 어떠한 경우에도 위와 같은 결과가 나올 수가 없었다. 즉 아래의 그림 6은 모든 가능한 Point와 LineString, Point와 Polygon간의 위상관계를 나타내지만 어떠한 경우도 교차위상관계로 성립될 수 없다. 따라서, 본 연구에서는 이런 경우는 구현하지 않았다.

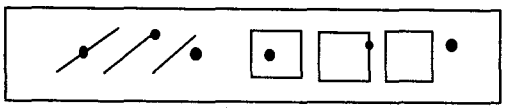


그림 6. Point와 LineString, Point와 Polygon간의 가능한 모든 위상관계

문제 2: 두 기하 객체 사이에서 너무 많은 위상관계로 발생하는 모호성

예를 들면, 다음과 같은 그림 7은 공간 데이터 간의 위상 관계가 포함인지 교차인지 접촉인지가 구별이 되지 않는 경우이다. 이런 경우들은 OpenGIS SFG 사양에 따라 위상 관계를 비분리로 처리하였다.

그러나, 다음 그림 8과 같은 공간 데이터 간의 위상 관계는 OpenGIS SFG 사양에는 명확하게 명시 되어 있지 않았다. 따라서, 본 논문에서는 이러한 경우에는 비분리와 포함, 두 가지의 위상 관계가 나타난 것으로 처리하였다.

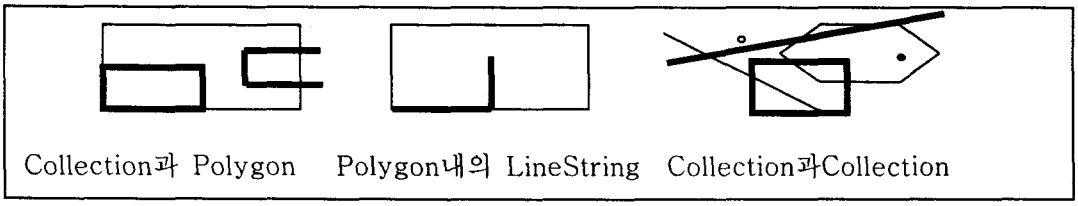


그림 7. 비분리로 처리한 공간 기하 객체 사이의 위상 관계 예

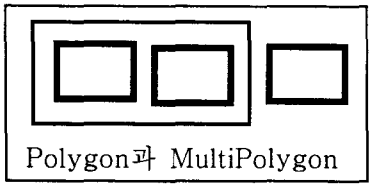


그림 8. 두 가지의 공간위상관계가 나타나는 경우

문제 3: 차원확장 9-교차모델의 결과 행렬에 대한 적용

OpenGIS 사양에는 기하 객체 사이의 위상관계를 차원 확장된 9-교차모델을 이용하여 표현하도록 되어 있으나, 실제 8가지의 공간관계연산자를 정의할 때는 이를 이용하지 않고, 기존의 9-교차모델을 이용하였다. 따라서, 본 연구에서는 OpenGIS에서 제시하는 차원 확장된 9-교차모델을 그대로 이용하여 8가지의 공간관계연산자를 정의하고 구현하였다

5. 결론

지리정보시스템을 구축하는 매우 효과적인 방법은 각 기능을 컴포넌트화하여 이를 재사용하는 것이다. 이를 위하여 OGC에서는 OpenGIS 사양을 정의하였다. OpenGIS사양에 나타난 컴포넌트 중에서 매우 자주 사용되는 것은 공간객체에 대한 연산인데, 이는 SFG(Simple Feature Geometry)와 각 기하객체의 공간연산자를 위한 인터페이스로 정의되어 있다.

본 연구에서는 OpenGIS에서 정의된 SFG의 기능을 컴포넌트로 구현하였다.

구현에서 발생된 여러 가지 문제점이나, 고려해야 할 점들을 본 논문에서는 그 해결책들과 함께 제시하였다. 또한, OpenGIS의 사양에 나타난 문제점들도 구현 도중 발견되었는데, 이에 대한 설명과 해결방법도 본 논문에서 제시하고 구현하였다.

본 연구에서 개발된 OpenGIS SFG 컴포넌트는 다른 지리정보시스템을 구축하는데 매우 효과적으로 재사용될 수 있으리라고 예상된다. 본 연구에서 개발된 몇 가지 공간연산자의 성능은 다른 접근방법으로 개선될 수 있으리라고 예상된다. 따라서, 앞으로의 연구는 몇 가지 연산자에 대한 성능개선을 포함하여야 한다.

참고문헌

[1] Open GIS Consortium, *Open GIS Simple Feature Specification for OLE/COM Revision 0*, 1997

[2] Open GIS Consortium, *Open GIS Simple Feature Specification for OLE/COM Revision 1.0*, 1998

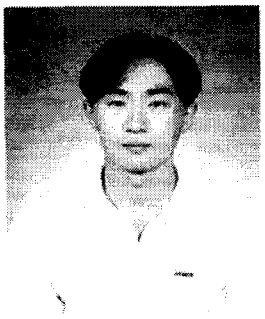
[3] Open GIS Consortium, *Open GIS Simple Feature Specification for OLE/COM Revision 1.1*, 1999

[4] Open GIS Consortium, *The OpenGIS Abstract Specification Model version 3*, 1998

[5] Open GIS Consortium, *The OpenGIS Abstract Specification Model version 4*, 1999

[6] C. Eliseo, S. Jayant, and M. J. Egenhofer, "Modeling Topological Spatial Relations Strateges for Query Processing", *Computers and Graphics* 18(6), 1994.

[7] M. J. Egenhofer, "Reasoning about Binary Topological Relationships." Proc. *SSD'91*, pp.143-160

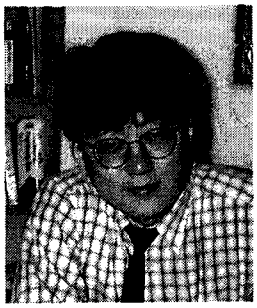


황 정 래

1999년 경성대학교 컴퓨터공학과 졸업(학사)

1999년-현재 부산대학교 지리정보시스템학과 석사 과정

관심분야: GIS, Clearinghouse, Metadata



이 기 준

1984년 서울대학교 계산통계학과 졸업(학사)

1986년 서울대학교 계산통계학과 졸업(석사)

1992년 프랑스 국립 응용과학원(INSA) 전산학 박사

1993년-현재 부산대학교 전자계산학과 교수

관심분야: 공간데이타베이스, GIS

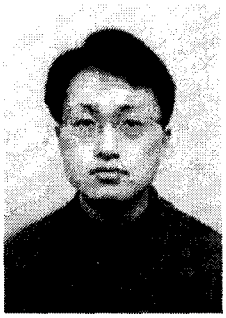
강 혜 경

1995년 창원대학교 행정학과 졸업(학사)

1998년 부산대학교 지리정보시스템학과
졸업(석사)

2000년-현재 부산대학교
지리정보시스템학과 박사과정

관심분야: GIS, FrameworkData, Data model



강 인 수

1996년 부산외국어대학교 컴퓨터공학과
졸업(학사)

1998년 부산대학교 전자계산학과 졸업(석사)

2000년 부산대학교 전자계산학과 (박사수료)

2000년-현재 한국통신 정보기술

관심분야: WebGIS, Data Mining