이동객체를 위한 질의처리 컴포넌트의 설계 및 구현

Design and Implementation of Query Processor for Moving Objects

김경숙*, 권오제**, 변희영**, 조대수***, 김태완****, 이기준*****
Kyoung-Sook Kim, O-Je Kwon, Hee-Young Byun, Dae-Soo Jo, Tae-Wan Kim, Ki-Joune Li

요 약 무선통신망과 GPS(Global Positioning System)를 탑재한 모바일 단말기의 발달로 사람이나 사물의 위치정보를 파악하고 이용하는 위치기반 서비스의 영역이 확대되고 있다. 위치기반서비스와 같은 응용분야에서는 시간에 따라 연속적으로 움직이는 이동객체를 효율적으로 저장하고 처리할 수 있는 데이터베이스가 주요한 기술이다. 본 논문에서는 이동객체 데이터베이스를 개발하기 위한 하나의 서브시스템으로서, 이동객체에 대한 질의를 처리하기 위한 컴포넌트를 설계 및 개발한다. 이동객체에 대한 질의를 처리하기 위해서 대표적인 질의표현 및 처리방법을 조사하고, 기존에 개발된 이동객체에 대한 데이터모델과 연산자를 기반으로 SQL형태의 이동객체 질의어를 새로이 정의한다. 사용자는 본 연구에서 제공하는 이동객체 질의어를 이용하여 이동객체의 위치정보에 대한 영역질의, 위상질의, 궤적질의, 최근접질의 등을 표현할 수 있다. 이동객체 질의처리 컴포넌트는 각 질의 들을 분석한 후 이를 효율적으로 처리하기 위한 모듈 들을 설계하고 구현한다. 또한, 다른 이동객체 응용시스템을 개발할 때 본 시스템의 활용을 높일 수 있도록 ADO.NET 인터페이스를 제공하고 XML을 이용하여 질의의 결과를 표현할 수 있는 기능을 제공한다.

ABSTRACT With the growth of wireless communication networks and mobile devices taking in GPS, Location-Based Service(LBS) is becoming an integral part of mobile applications. LBS can deal with location-aware features such as persons holding mobile phones or vehicles equipped with GPS, and provide the users with the location information of the features. Thus it is necessary to develop moving object database systems to store, manage, and query moving objects which change their locations continuously as time passes. In this paper, we design and implement a query processing component which deals with moving objects as a key data type. For this component, we define a new SQL-like query language(called MOQL) and as a consequence, design and implement modules that analyze and execute queries. It supports various types of operators that process range queries, infer topological relations, compute trajectories, and find k-nearest neighbors. It can be used as a subsystem of other application systems which deal moving objects and also supports ADO.NET interface that can be used to interact end-users.

주요어 : 위치기반서비스, 이동객체, 질의어, 질의처리기, 컴포넌트
Key word : Location-Based Service, Moving Object, Query Language, Query Processor, Component

1. 서 론

위치 기반 서비스(Location-Based Service; LBS)는 현재 많은 모바일 분야에서 이용되고 있다. 무선통신망과 GPS(Global Positioning System)를 탑재한 모바일 단말기의 발달로 사람이나 자동차의 위치 파악이 용이해짐으로써 위치기반 서비스는 안전보호, 생활정보 제공 등의 응용서비스 영역으로 확대되

* 부산대학교 전자계산학과 박사과정 ksookim@pusan.ac.kr
** 부산대학교 정보컴퓨터공학부 석사과정 {ojkwon, hybyun@isel.cs.pusan.ac.kr}
*** 한국전자통신연구원 텔레매틱스연구단 LBS연구팀 선임연구원 junest@etri.re.kr
**** 부산대학교 컴퓨터 및 정보통신연구소 전임연구원 twkim@quantos.cs.pusan.ac.kr
***** 부산대학교 정보컴퓨터공학부 부교수 lik@pusan.ac.kr

고 있다[1]. 위치기반 서비스를 구축하기 위해서는 위치측위기술, 무선통신망, LBS 서버기술, LBS 응용서비스 기술 등의 기반기술이 확보되어야 한다. 특히, 다양한 서비스를 제공하기 위해서는 LBS 서버가 실시간 또는 과거시간의 위치정보를 저장, 관리, 검색할 수 있어야 한다. 현재 LBS 서버는 기존의 지리정보시스템에서 많이 활용되던 공간데이터베이스 기술을 활용하여 개발되고 있다. 그러나, 이러한 위치기반 서비스의 대상이 되는 객체는 건물과 도로와 같은 고정적인 위치정보를 가지고 있는 공간객체뿐만 아니라 자동차나 사람과 같이 움직이는 이동객체[2]도 포함하고 있다. 이동객체는 일반적으로 시간이 변함에 따라 위치정보가 연속적으로 변하는 객체를 말한다. 따라서, 기존의 공간데이터베이스의 기술이나 이론을 직접 이동객체에 적용하는 것은 부적절하다. 즉, 이동객체를 효율적으로 처리하기 위해서는 시간에 따라 변하는 위치정보의 특성을 고려한 이동객체 데이터베이스 시스템[3] 개발이 필요하다. 이동객체 데이터베이스 시스템은 방대한 위치정보를 저장하기 위한 저장시스템, 과거 및 현재의 위치정보에 대한 색인, 그리고 사용자의 질의를 처리할 수 있는 질의처리기 등으로 크게 구성된다.

대용량 위치데이터 관리를 위한 정보시스템은 위치획득 서브시스템, 위치저장 서브시스템, 그리고 위치질의 서브시스템으로 구성된다[4]. 이들 중 기존의 위치질의 서브시스템은 이동객체의 가하적 속성을 다루는 데이터모델, 기본 연산자, 간단한 질의어를 정의 및 구현 하였다. 따라서, 실제 사용자들이 이러한 사양들을 사용하여 응용시스템을 개발하기에는 기술적인 어려움이 많이 존재한다. 본 연구에서는 사용자 인터페이스를 지원하기 위하여 이동객체 질의처리 컴포넌트를 설계하고 구현한다. 질의처리 컴포넌트는 1)이동객체에 대한 데이터 모델과 기본연산자 정의 및 구현, 2) 질의언어의 정의, 3) 질의분석 및 실행, 4) 질의최적화 등의 기능성이 필요하다. 이동객체를 표현하는 데이터모델과 기본연산자는 이전 연구[5]에서 자세히 기술되었다. 본 연구에서는 OpenGIS 확장SQL[6]과 이동 객체에 관련된 연구[7][8]들을 참조하여 질의언어를 정의하고, 질의언어를 처리하기 위한 질의분석 및 실행 기능을 개발한다. 그리고, 좀더 효율적으로 처리할 수 있는 질의최적화 방법도 함께 제시한다.

본 논문은 다음과 같이 구성된다. 2장에서는 이동객체 데이터모델링과 질의처리에 대한관련연구 및 이동객체 컴포넌트를 소개하고 3장에서는 사용자 인터페이스로 제공되는 이동객체 질의어를 정의한다. 4장에서는 이동객체 질의처리 컴포넌트의 구조와 각 모듈의 구현에 대해 살펴보고, 질의처리에 대한 예를 보인다. 마지막으로 5장에서 결론을 맺는다.

2. 관련연구

이동객체를 위한 질의처리 컴포넌트를 개발하기 위해서는 우선적으로 이동객체를 표현할 수 있는 데이터모델과 이에 대한 질의처리 방법의 연구가 필요하다. 본 장에서는 현재까지 연구된 대표적인 연구 몇 가지를 소개하고, 기존에 개발된 이동객체 컴포넌트에 대해 간단히 살펴본다.

2.1 이동객체 데이터모델과 질의처리 방법의 기존 연구

이동객체는 시간에 따라 연속적으로 변하는 일종의 시공간객체이다. 시공간데이터베이스[9]는 시간과 공간의 개념을 모두 포함하고 있지만 단지 이산적 시간적 속성을 가지는 공간객체에 국한되어 있어, 연속성을 가지고 있는 이동객체를 처리하기에는 적합하지 못하다. 따라서, 이동객체에 대한 새로운 모델 및 질의어, 그에 따른 질의 처리 방법 들을 개발해야 한다.

이동객체에 대한 데이터 모델과 질의처리방법에 대한 연구는 CHOROCHRONOS[10][11][12]와 DOMINO[8][14]에서 대표적으로 보여주고 있다. CHOROCHRONOS에서는 이동객체를 표현하기 위

〈표1〉 CHOROCHRONOS 이동객체 기본연산자

종 류	연산자
Project to Domain/Range	defttime, rangevalue, location, trajectory, ...
Interaction with Domain/Range	atinstant, atperiods, at, present, ...
Rate of Change	derivative, speed, turn, velocity

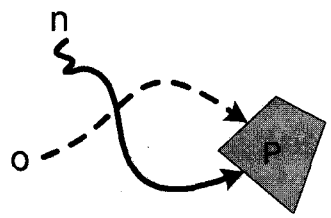
한 데이터 형으로 이동점객체(moving point), 이동면객체(moving polygon) 등을 정의하고, 표1과 같이 각 데이터 형에 대해 정사영 연산자, 상호연산자, 변화율에 대한 연산자 등을 이동객체 기본연산자로 정의하였다.

[12]에서는 이동객체 데이터 모델을 개념적인 추상적 데이터모델과 실제 구현할 수 있는 이산적 데이터 모델로 나누어 데이터 타입을 정의하였다. 그리고, 이산적 데이터 모델을 바탕으로 이동객체 사이의 거리나 위상관계와 관련된 몇 가지 기본연산자를 구현하기 위한 알고리즘을 보여주고 있다. 그리고, [7][13]에서는 이러한 모델에 대하여 SQL 기반의 이동객체 질의어인 STQL(Spatio-Temporal Query Language)을 제안하였다. 특히, STQL에서는 시공간 프리디키트(Spatio-Temporal Predicates)를 정의하여 이동객체에 대한 위상적 관계뿐만 아니라 시간적 선택(selection) 질의, 집합(aggregation) 질의를 표현하고 있다.

DOMINO[14]에서는 이동객체의 연속적인 공간 속성을 표현하기 위한 MOST(Moving Object Spatio-Temporal)데이터 모델[8]을 사용하여 이동객체를 표현하였다. CHOROCHRONOS의 데이터모델은 이동객체의 과거에서 현재까지 시간에 따라 연속적으로 움직인 궤적을 표현하는 반면 MOST 데이터 모델은 이동객체의 미래의 위치를 예측할 수 있도록 현재 위치정보를 시간에 대한 함수형태로 표현하였다. 이는 시간에 따라 변하는 이동객체의 현재위치를 저장하기 위해서 일어난 빈번한 갱신 문제를 고려한 모델이다. 또한, 이러한 시간함수에 대한 연산자를 정의하여 QUEL 기반 FTL(Future Temporal Logic) 질의언어를 제안하고, 이동객체에 대한 질의를 처리하는 방법을 보여주고 있다. <그림1>은 이동객체에 대한 질의를 STQL과 FTL을 사용하여 표현하였다.

그 외에도 SQL^{ST}[15]는 시공간 데이터에 대한 모델링과 질의 방법을 기존의 관계형 데이터베이스에서 SQL을 이용하여 처리하는 방법을 보여주고 있다. SQL^{ST}은 시간을 위한 점기반 시간모델과 공간적인 부분을 위한 삼각측량법을 사용한다. 즉, 단일 시간단위로 공간객체를 나누어 질의를 처리하고 있다. 그러나, SQL^{ST}은 이동객체에 대한 연속적인 부분에 대해서는 잘 고려하지 못 하였다. 그리고, [16]에서는 이동객체에 대한 위상 변화에 대하여 개념적 및 수학적 방법으로 모델링하고, [17]에서는 이를 기반으로 CQL(Constraint Query Language)을 이용한 거리기반의 질의어를 정의하여 이동객체의 최근접 질의를 표현하고 있다. [18]에서도 이동객체에 관련된 12가지 구문의 질의어를 설계하였다. 그러나, 대부분의 연구는 질의어 정의를 위주로 작업하였으며, 실제 질의어를 처리하기 위한 시스템 개발은 그리 많지 않다. 본 논문에서는 OGC의 확장SQL형태의 이동객체 질의어를 정의하고 이를 응용시스템에서 사용할 수 있도록 질의처리기 함께 개발한다.

Retrieve the pairs of objects o and n such that the distance between o and n stays within 5 miles until they both enter the polygon P



STQL(Spatio-Temporal Query Language)

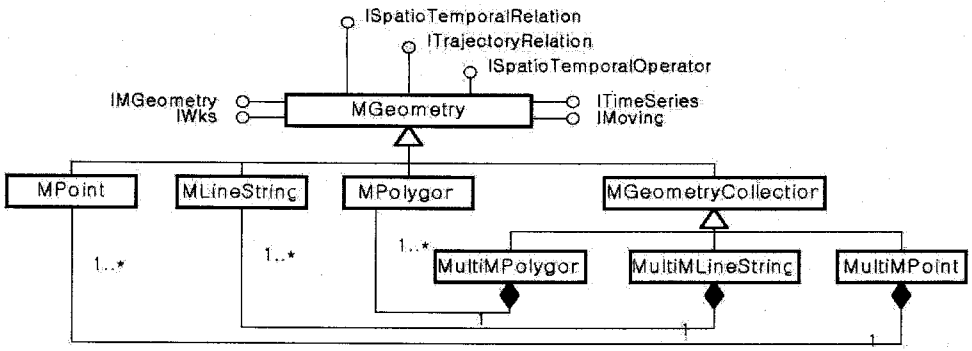
```
Select o,n
From MO
Where beginTime(Dist(o,n)<=5)<=now and
      endTime(Dist(o,n)<=5)>=
      beginTime(Inside(o,P) and Inside(n,P))
```

FTL (Future Temporal Logic)

```
Retrieve o,n
Where Dist(o, n) <= 5
Until (Inside(o,P) and Inside(n,P))
```

<그림 1> 이동객체 질의어 예

2.2 이동객체 컴포넌트

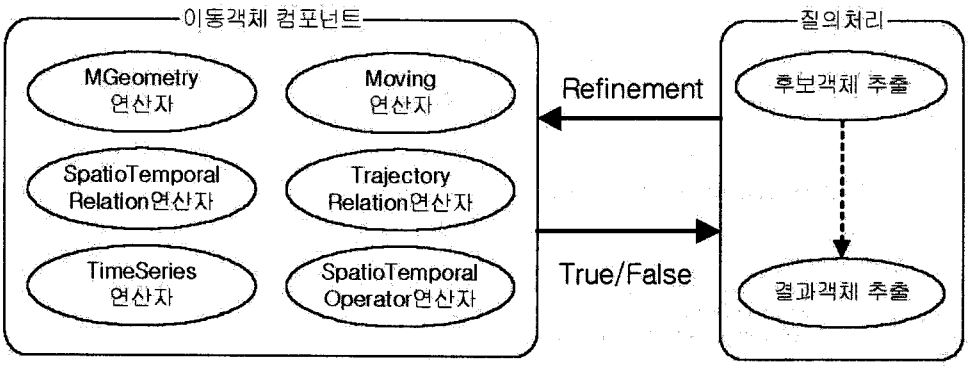


〈그림 2〉 이동객체 컴포넌트

이동객체 컴포넌트[5]는 〈그림 2〉와 같이 이동객체에 대한 데이터 형과 기본연산자를 제공하고 있다. 이 모델은 OGC의 SFG(Simple Feature Geometry)의 모델[6]을 기반으로 확장되었다.

이동객체에 대한 데이터 형은 이동점객체(MPoint), 이동선객체(MLineString), 이동면객체(MPolygon)와 독립적인 이동객체가 하나이상인 복합이동점객체(MultiMPoint), 복합이동선객체(MultiMLineString), 복합이동면객체(MultiMPolygon), 그리고 이들의 상위 클래스인 이동복합공간객체(MGeometry-Collection)가 있다. 그리고 각 데이터 형의 기본연산자를 위한 인터페이스들이 구현되었다. 기본연산자로는 이동객체의 기하학적 속성에 관련된 연산자(IMGeometry), 이동객체의 이산적인 시계열 데이터를 다루기 위한 연산자(ITimeSeries), 연속적으로 변하는 이동객체에 대한 기본함수로 시간과 공간에 대한 상호·정사영 관련 연산자(IMoving), 이동객체 사이의 위상관계를 처리하기 위한 연산자(ISpatioTemporalRelation), 이동객체의 움직인 궤적에 대한 연산자(ITrajectoryRelation), 그리고 이동객체의 시공간 기하학적 계산을 위한 연산자(ISpatio-TemporalOperator) 등을 제공하고 있다. IWks 인터페이스는 OGC의 WKB(Well Known Binary)를 확장한 것이고 이동객체를 다른 컴포넌트에 전달할 수 있는 연산자를 제공한다.

본 논문에서 개발하는 이동객체 질의처리기는 이동객체 컴포넌트에서 제공하고 있는 데이터 형과 기본연산자를 바탕으로 이동객체 질의어를 정의한다. 이동객체 질의어는 다음 장에서 자세히 설명하겠다. 그리고, 표현된 이동객체에 대한 질의는 〈그림 3〉과 같이 기본연산자들을 수행한 결과를 조합하여 처리하게 된다.



〈그림 3〉 이동객체 질의처리와 기본연산자

3. 이동객체 질의어

이동객체 질의처리 컴포넌트는 이동객체에 대한 사용자의 질의를 표현하기 위해서 기존의 SQL의 형태를 바탕으로 이동객체 질의어(Moving Object Query Language-이하 MOQL)를 새로이 정의하였다. 이 장에서는 본 논문에서 정의한 이동객체 질의어에 대해 설명한다.

3.1 MOQL 데이터 타입

이동객체에 대한 데이터베이스는 논리적 테이블 형태인 레이어의 집합으로 구성된다. 레이어의 필드는 사용자가 정의 가능하고 본 논문에서 사용하는 레이어는 이동객체의 고유번호(**M**oving **O**bject **ID**entification, MOID), 이동궤적, 그리고 속성필드로 구성되어 있다고 가정한다. 예를 들어, 모바일 폰 사용자에 대한 레이어를 생성하기 위해서는 다음과 같은 MOQL문을 사용한다.

CREATE TABLE *CelluarPhoneUser*(*id* MOID, *position* **MPOINT**, *name* **VARCHAR**);

MOQL에서는 사용자의 레이어 정의를 지원하기 위하여 다음 〈표 2〉와 같이 5가지 유형의 데이터 타입을 지원한다. 기본 데이터 타입형은 정수형이나 실수형과 같은 기본 데이터 타입에 MOID타입과 MSSQL타입을 추가하였다. MOID타입은 각각의 이동객체를 식별할 수 있는 번호를 나타내고 MOID타입의 스키마는 하나의 레이어에 적어도 하나 이상 반드시 존재해야 한다. MSSQL타입은 질의처리 컴포넌트가 비공간 속성을 식별하기 위한 내부데이터 타입으로 질의문에 명시적으로 표현하지 않는다. 본 논문에서는 MicroSoft SQL Server(이하 MSSQL 서버)에서 제공하고 있는 모든 데이터 타입 들을 이동객체 위치정보와 구별되도록 MSSQL타입으로 변환하여 처리한다. 사용자는 MSSQL 서버에서 지원하는 데이터 타입을 그대로 질의문에 사용할 수 있다. 그리고, 나머지 유형의 시간, 공간, 이동값, 이동객체에 대한 데이터 타입은 앞에서 설명한 이동객체 컴포넌트의 데이터 모델과 일대일 대응된다. 단, RECTANGLE과 MRECTAGLE은 POLYGON 및 MPOLYGON의 특수화 된 자료형으로 추가하였다.

MOQL문에서 나타나는 데이터 타입의 인스턴스 중에서 공간 데이터 타입의 인스턴스는 OGC의 SFG의 WKT(Well-Known Text)포맷을 그대로 사용하고, 나머지 데이터 타입에 대해서는 〈표 3〉과 같이 WKT 포맷을 응용하여 표현하였다.

예를 들어, 생성된 레이어에 데이터를 삽입하기 위한 MOQL문은 다음과 같이 표현된다.

INSERT INTO *CelluarPhoneUser*(111, *MPOINT('2003-5-8T11:56:10', 10, 15,*
'2003-05-08T11:56:20', 13, 26,
'2003-5-8T11:56:30', 15, 37...));

〈표 2〉 MOQL 데이터 타입

타입유형	데이터 타입
기본값	MOID, BOOL, LONG, DOUBLE, MSSQL
시간객체	INSTANT, PERIOD, INTERVAL, TEMPORALCOLLECTION
공간객체	POINT, LINESTRING, POLYGON, RECTABGLE, MULTIPOINT, MULTILINESTRING,MULTIPOLYGON,GEOMETRYCOLLECTION
이동값	MBOOL, MINTEGER, MDOUBLE
이동객체	MPOINT, MLINESTRING, MPOLYGON, MRECTANGLE, MULTIMPOINT, MULTIMLINESTRING, MULTIMPOLYGON, MGEOMETRYCOLLECTION

〈표 3〉 MOQL 인스턴스 표현

타 입	표 현
INSTANT	'yyyy-mm-ddTh:m:s' [or 'h:m:s'] [=>과거의 순간시간] NOW [=>현재시간]
PERIOD	PERIOD(yyyy-mm-ddTh:m:s, yyyy-mm-ddTh:m:s)
TINTERVAL	TINTERVAL(yyyy-mm-ddTh:m:s[now], DAY+[-]n) TINTERVAL(yyyy-mm-ddTh:m:s[now], WEEK+[-]n) TINTERVAL(yyyy-mm-ddTh:m:s[now], MONTH+[-]n) TINTERVAL(yyyy-mm-ddTh:m:s[now], YEAR+[-]n)
MBOOL	MBOOL((bv_1, t_{11}, t_{12}), (bv_2, t_{21}, t_{22}), ...) [=> t_{ij}는 instant를 말함]
MINTEGER	MINTEGER((iv_1, t_1), (iv_2, t_2), ...)
MDOUBLE	MDOUBLE((dv_1, t_1), (dv_2, t_2), ...)
MPOINT	MPOINT (t_1, x_1, y_1, t_2, x_2, y_2, ...)
MLINESTRING	MLINESTRING ((t_1, x_{11}, y_{11}, x_{12}, y_{12}, ...), (t_2, x_{21}, y_{21}, x_{22}, y_{22}, ...))
MPOLYGON	MPOLYGON ((t_1, (x_{111}, y_{111}, x_{112}, y_{112}, ...), (x_{121}, y_{121}, x_{122}, y_{122}, ...)), (t_2, (x_{211}, y_{211}, x_{212}, y_{212}, ...)), ...)
MULTIMPOINT	MULTIMPOINT ((t_1, x_1, y_1, t_2, x_2, y_2, ...), (t_1, a_1', b_1, t2, a_2, b_2, ...), ...)
MULTI-MLINE STRING	MULTIMLINESTRING (((t_1, x_{11}, y_{11}, x_{12}, y_{12}, ...), (t_2, x_{21}, y_{21}, x_{22}, y_{22}, ...)), ((t_1, a_{11}, b_{11}, a_{12}, b_{12}, ...), (t_2, a_{21}, b_{21}, a_{22}, b_{22}, ...)))
MULTI-MPOLY GON	MULTIMPOLYGON (((t_1, (x_{111}, y_{111}, x_{112}, y_{112}, ...), (x_{121}, y_{121}, x_{122}, y_{122}, ...)), (t_2, (x_{211}, y_{211}, x_{212}, y_{212}, ...)), ...), ((t_1, (a_{111}, b_{111}, a_{112}, b_{112}, ...), (a_{121}, b_{121}, a_{122}, b_{122}, ...)), (t_2, (a_{211}, b_{211}, a_{212}, b_{212}, ...)), ...))
MGEOMETRY-COLLECTION	MGEOMETRYCOLLECTION (mgeometry wkt)
MRECTANGLE	MRECTANGLE(t_1, t_2, x_1, x_2, y_1, y_2)
RECTANGLE	RECTANGLE (x_{11} x_{21}, y_{11} y_{21})

3.2 MOQL 문법

MOQL은 레이어를 생성하거나 소멸하기 위한 데이터 정의 언어와 생성된 레이어에 이동객체를 삽입, 삭제, 그리고 검색할 수 있는 데이터 관리 언어로 구성된다.

데이터 정의 언어는 기존의 SQL과 같은 형태로 사용되고, 데이터 관리 언어는 이동객체를 위하여 몇 가지 추가하였다. 우선, 데이터 삽입 명령문은 궤적을 삽입하는 경우와 현재위치를 갱신하는 경우 두 가지로 표현된다. 이동객체의 궤적 전체를 삽입하는 경우는 앞 절에서 보여준 예와 같이 이동객체-예를 들면, MPOINT- 인스턴스를 삽입하고, 현재 위치를 계속적으로 갱신해 나가는 경우는 다음과 같이 데이터 삽입 문에 현재 시간(NOW)에 해당하는 공간객체-예를 들면, POINT- 인스턴스를 삽입한다.

INSERT INTO *CelluarPhoneUser* (**111, *POINT(20, 40), NOW***);

데이터의 삭제 명령문에서는 이동객체의 궤적에서 임의의 시간까지 궤적을 삭제하는 TO 키워드가 추가되었다. 예를 들어, '번호가 111인 모바일 사용자의 궤적에서 A시간까지의 이동 궤적을 삭제하라' 라는 명

령문은 다음과 같이 표현된다.

DELETE FROM *CelluarPhoneUser* WHERE id = 111 **TO 'A'** ;

본 논문에서 사용하는 이동객체 위치저장시스템[19]은 궤적의 중간부분을 삭제하는 것을 허용하지 않고 있다. 그리고, 궤적의 중간부분을 삭제하는 것은 이동객체의 연속성을 보장하지 못하게 되어 이동객체에 대한 연산자를 수행 시 잘못된 결과를 초래 할 수 있다. 따라서, MOQL에서는 기본적으로 삭제문에서 데이터가 처음 저장된 시간을 기준을 내포하고 있으므로 명시적인 시작시간은 지정하지 않는다. TO 키워드가 존재하지 않는 명령문은 WHERE 조건에 부합하는 이동객체의 전체 궤적뿐 만 아니라 다른 모든 정보까지 삭제하라는 의미이다.

데이터 검색을 위한 명령문은 세 개의 키워드인 SELECT, FROM, 그리고 WHERE 절로 구성된다. FROM 절은 질의를 하고자 하는 테이블 이름을 나타내며, 다수의 테이블을 사용할 수 있고 AS 키워드를 이용하여 테이블에 대한 별명(alias) 역시 사용 가능하다. SELECT 절에는 조건에 부합하는 레코드의 원하는 컬럼 들이나 본 논문에서 정의한 MOQL 함수(Function)이 올 수 있고, WHERE 절은 MOQL 연산자를 이용하여 조건을 표현한다. 이때, 이동객체의 위치정보에 대한 연산자 외에도 속성에 관련된 조건도 표현 가능하다. MOQL 함수와 MOQL 연산자에 대해서는 3.3절과 3.4절에서 자세히 설명하겠다. 〈그림 4〉는 이동객체 질의어의 문법의 일부이다.

3.3 MOQL 연산자

MOQL에서 정의한 연산자는 논리형 연산자(Boolean operator)로 이동객체의 위치 정보에 대한 시공간 관계연산자, 궤적 관계연산자, 최근접 연산자와 시간 관계연산자, 그리고 이동객체에 대한 함수나 컬럼에 대한 비교연산자가 있다. 각 연산자들의 결과는 논리형(boolean) 또는 이동논리형(mboolean) 값 형태로 제공함으로써 질의처리시 주어진 명시적 조건에 부합하는 이동객체를 추출하게 된다. 시공간 관계연산자와 궤적 연산자는 이동공간객체와 이동공간객체 사이 또는 이동공간객체와 공간객체 사이에서 시간에 따라 변하는 공간적 위상관계-분리(Disjoint), 포함(Contain), 교차(Overlap), 접촉(Touch) 등-와 움직인 궤적사이의 위상관계-진입(Enter), 이탈

```
<moql> : <statement> <semicolon>
<statement> : <ddl statement> | <dml statement>
<ddl statement> : <create statement> | <drop statement>
<dml statement> : <insert statement> | <delete statement> | <select statement>
<select statement> : SELECT <select list> <table expression>
<select list> : <column> [ COMMA <select list> ]
<column> : <asterisk> | <moving function> | <temporal function> | <column name>
<table expression> : <from clause> [ <where clause> ]
<from clause> : FROM <table list>
<table list> : <table> [ COMMA <table list> ]
<table> : <table name> [ AS <alias name> ]
<where clause> : WHERE <search condition>
<search condition> : <BOOL term> | <BOOL term> OR <search condition>
<BOOL term> : <BOOL factor> | <BOOL factor> AND <BOOL term>
<BOOL factor> : <mopredicate> <comp op> <comp value> | <attribute predicate>
<mopredicate> : <trajectory relation> | <nearest> | <temporal relation> |
                <moving function> | <column name>
이하 생략
...
```

〈그림 4〉 이동객체 질의어

(Leave), 상주(Inside) 등-를 표현한다. 최근접 연산자는 주어진 이동객체나 공간객체에서 가장 가까운 k개의 객체를 추출하기 위한 질의를 표현하고, 시간관계연산자는 이동객체사이의 생명주기(Lifetime)의 위상관계를 나타낸다. 〈표 4〉는 WHERE 절에 올 수 있는 MOQL 연산자를 나타낸다.

〈표 4〉 MOQL 연산자

종 류	연 산 자
시공간 관계연산자	EQUALS, DISJOINTS, MEETS, TOUCHES, CONTAINS, WITHINS, CROSSES, OVERLAPS, INTERSECTS
궤적 관계연산자	ENTERS, LEAVES, PASSES, INSIDES
최근접 연산자	NEAREST
시간 관계연산자	PRECEDE, AFTER, MEET, OVERLAP, DURING

다음은 MOQL 연산자를 이용하여 이동객체에 대한 질의를 표현한 예이다.

(Q1) B영역에 진입했던 객체를 모두 검색하라

```
SELECT  id, position
FROM  CelluarPhoneUser
WHERE  ENTERS(position, rectangle
  (3000, 10000, 3000, 10000)) = TRUE;
```

(Q2) 임의 A시간에 P지점에서 가장 가까운 객체 5개를 검색하라.

```
SELECT  id, position
FROM  CelluarPhoneUser
  WHERE  NEAREST(snapshot(position,
    '2003-11-23T12:30:40'),point(3000,
                3000), 5) = TRUE;
```

3.4 MOQL 함수

MOQL에서는 〈표 5〉와 같이 공간객체관련, 시계열관련, 이동객체관련, 시공간위상관련, 시공간연산관련, 시간객체관련, 이동점객체관련, 이동선객체관련, 이동면객체관련 함수를 제공한다.

이러한 함수는 조건에 부합하는 이동객체의 현재 또는 과거의 특정시점이나 시간구간에 대한 부분 위치정보나 움직인 궤적정보, 이동객체의 공간적 또는 시간적 속성에 대한 정보를 추출하고 이동객체 사이의 거

〈표 5〉 MOQL 함수

종 류	연 산 자
공간객체관련 함수	DIMENSION, ISEMPTY, ISSIMPLE, ENVELOPE
시계열관련 함수	SUBSEQUENCE
이동객체관련 함수	PROJECT, LIFETIME, SNAPSHOT, SLICE, SNAPSHOTBYVALUE, SLICEBYVALUE
시공간위상관련 함수	EQUALS, MEETS, DISJOINTS, CONTAINS, WITHINS, CROSSES, OVERLAPS, INTERSECTS
시공간연산관련 함수	DISTANCE, BUFFER, BOUNDARY, UNION, INTERSECTION, DIFFERENCE
시간객체관련 함수	DURATION, UNION, DIFFERNCE, INTERSECTION
이동점객체관련 함수	SPEED, DIRECTION, ACCELERATION
이동선객체관련 함수	LENGTH
이동면객체관련 함수	PERIMETER, AREA

리를 계산하거나 위치정보에 대한 버퍼영역을 구하는데 사용한다. 각 함수들은 이동객체, 공간객체, 시간객체, 이동값 등을 결과값으로 제공한다.

함수는 SELECT 절의 추출정보뿐만 아니라 WHERE 절의 연산자의 파라미터로 위치할 수 있다. 다음은 MOQL 함수를 이용하여 이동객체에 대한 질의를 표현한 예이다.

(Q3) B영역에 포함되었던 객체의 궤적과 생명주기를 구하라.

```
SELECT  PROJECT(position), LIFETIME
        (position)
FROM  CelluarPhoneUser
WHERE  WITHINS(position, rectangle
  (3000, 10000, 3000, 10000)) = TRUE;
```

(Q4) A시간 사이에 서로 만난 객체의 교차 위치를 검색하라.

```
SELECT  PROJECT(INTERSECTION
        (S1.position, S2.position))
FROM  CelluarPhoneUser as C1, Celluar
        PhoneUser as C2
WHERE  MEETS(SLICE(C1.position,
        '2003-11-23T12:30:40','2003-
        11-23T12:30:40'),
        SLICE(C2.position,'2003-11-23
        T12:30:40','2003-11-23T
        12:30:40')) = TRUE  and
        C1.id != C2.id;
```

4. 이동객체 질의처리 컴포넌트

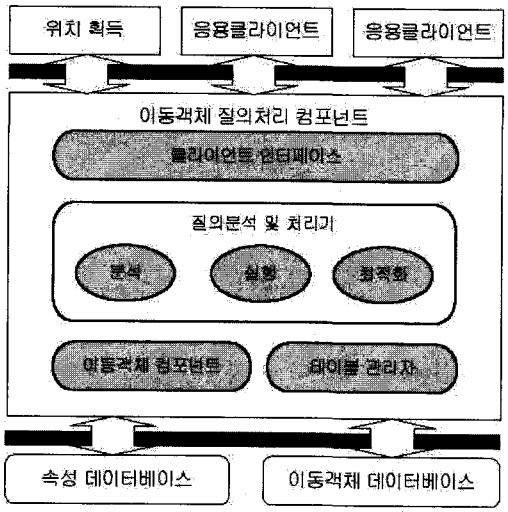
이동객체 질의처리 컴포넌트는 앞장에서 정의된 질의어를 분석하여 테이블 생성 및 삭제, 데이터 삽입 및 삭제, 그리고 조건에 부합하는 데이터를 검색하는 기능을 제공한다. 사용자의 조건질의가 주어지면, 우선적으로 이동객체에 대한 속성질의와 위치정보에 대한 질의로 분해하여 속성질의는 속성정보가 저장된 데이터베이스-예를 들어, 오라클이나 MSSQL 서버와 같은 상용데이터베이스-에서, 이동객체의 위치질의는 이동객체의 색인을 이용하거나 색인이 존재하지 않을 경우는 직접 위치저장시스템에서 제공하는 인터페이스들을 이용하여 후보객체의 정보를 추출하게 된다. 본 논문에서 구현된 질의처리 컴포넌트는 속성데이터베이스로 MSSQL 서버를 사용하고, 위치정보의 저장은 이동객체 위치저장시스템[19]이 담당한다. 그리고, 과거궤적 색인[20]과 현재위치 색인[21]을 이용하여 질의처리 시 좀더 효율적으로 질의를 처리하도록 하였다. 이동객체 컴포넌트에서 제공하는 기본연산자 들은 후보객체를 읽어온 후 정제 단계를 수행 할 때 사용된다. 그리고 정제된 질의 결과는 ADO. NET 인터페이스를 통하거나 XML 형태의 문서로 변환되어 사용자에게 제공하게 된다.

4.1 질의처리 컴포넌트 설계

이동객체 질의처리 컴포넌트는 클라이언트 인터페이스, 질의 분석, 실행 및 최적화 모듈, 테이블 관리 모듈, 이동객체 컴포넌트 모듈로 크게 구성된다.

- 클라이언트 인터페이스 - 클라이언트가 이동객체 질의처리 컴포넌트에 질의를 요청하고 결과를 참조할 수 있는 ADO.NET 인터페이스를 제공한다. 클라이언트는 이 인터페이스를 통하여 MOQL문장을 실행할 수 있고, 결과객체를 참조할 수 있다. 그리고, 웹서비스를 위하여 결과데이터를 XML형태의 문서로 생성하고 서버와 클라이언트 사이의 통신방법은 .NET 리모팅을 이용한다.
- 질의 분석 및 처리기 - MOQL질의문을 분석하고 처리하기 위한 인터페이스를 제공한다. 질의처리 모듈은 질의문을 분석하기 위한 파서(parser)와 분석된 질의를 수행하기 위한 질의 실행자(executer), 그리고 질의 수행시 최적화를 위한 모듈로 구성된다.
- 테이블 관리기 - 저장시스템과 연계되어 실제 데이터를 접근하는 인터페이스를 제공한다. 테이블 관리기는 질의처리시 필요한 이동객체에 대한 위치정보, 속성정보의 메타정보를 관리하고 속성 조건, 시간 조건, 이동객체 ID정보 등을 이용하여 저장시스템에서 이동객체 위치정보를 읽어와서 질의를 처리하기 위한 자료구조로 변환한다. 그리고 이동객체의 위치정보와 속성정보를 연결하여 하나의 객체로 만들어주는 역할을 담당하게 된다.
- 이동객체 컴포넌트 - 이동 점객체, 이동 선객체, 이동 면객체 등의 데이터타입과 기본연산자를 제공한다.

〈그림 5〉는 이동객체 질의처리 컴포넌트의 개략적인 구조를 도식화한 것이다.



〈그림 5〉 이동객체 질의처리 컴포넌트 구조도

4.2 질의처리 컴포넌트 구현

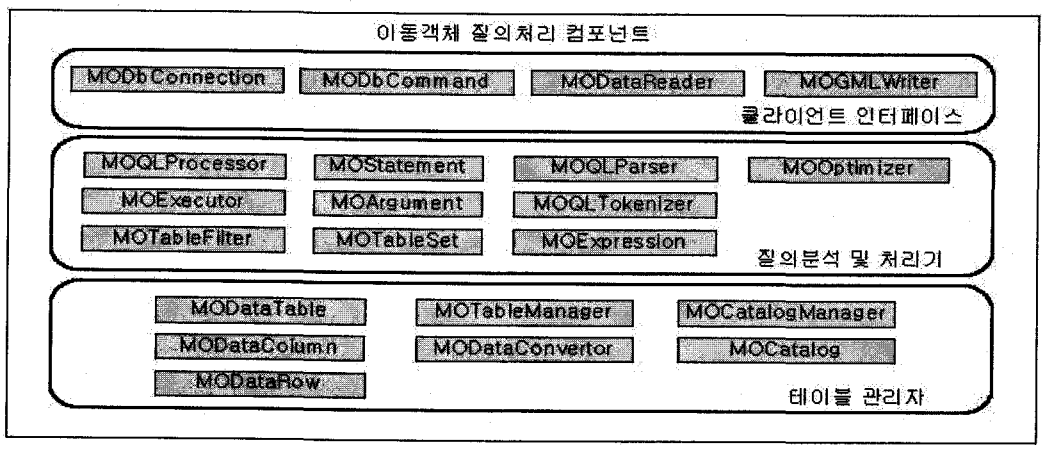
본 논문의 이동객체 질의처리 컴포넌트의 각 모듈들은 〈그림 6〉의 클래스 들로 구성된다.

클라이언트 인터페이스 모듈은 클라이언트와의 연결을 위한 MODBConnection클래스, 사용자의 MOQL 질의문을 나타내기 위한 MOQLCommand클래스, 질의처리기에서 명령을 실행하여 얻은 결과에 대해 읽을 수 있는 수단을 제공하는 MODataReader클래스, 질의 결과를 XML형태의 문서를 생성하기 위한 MOGMLWriter클래스가 있다.

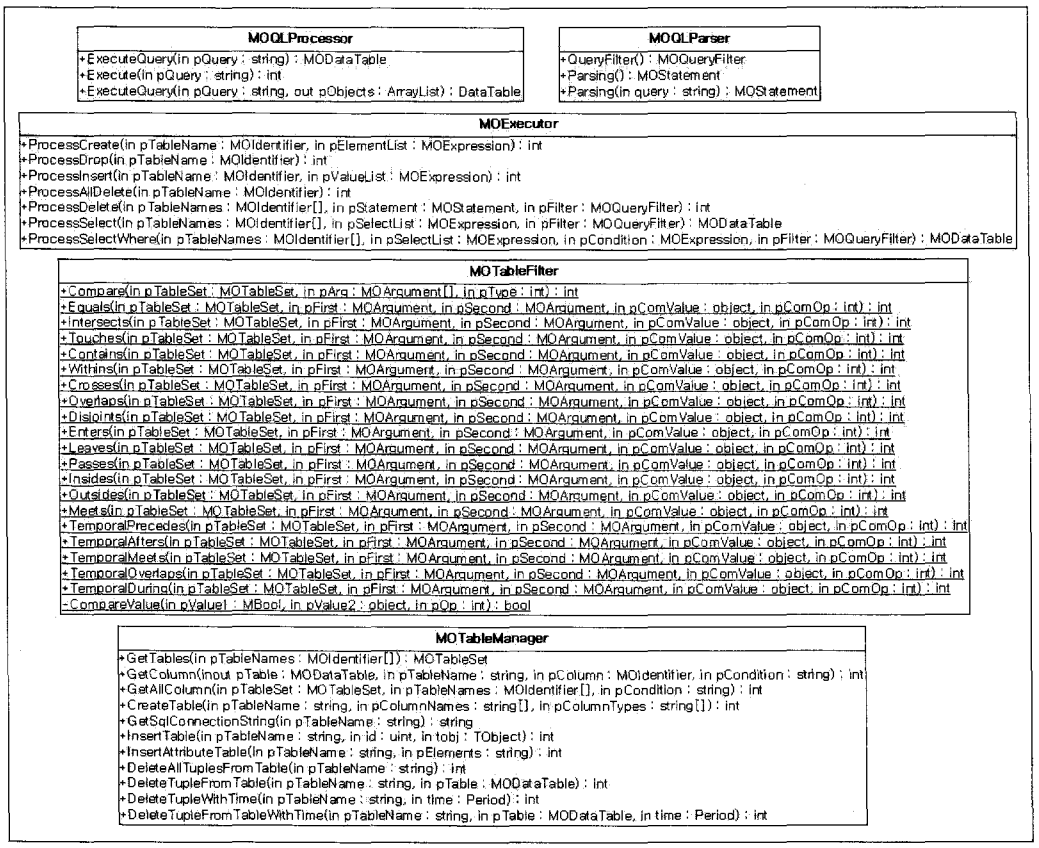
질의분석 및 처리기 모듈에 속하는 클래스는 MOQL 질의문 분석과 실행을 위한 인터페이스를 제공하는 MOQLProcessor클래스, MOQL 질의문을 파싱하여 트리형태로 변환하는 MOQLParser클래스, MOQL 질의문에서 토큰을 잘라내기 위한 MOQLTokenizer클래스, 분석된 트리 구조의 질의문을 나타내는 MOStatement 클래스와 MOExpression클래스, 질의 실행을 담당하는 MOExecutor클래스와 질의 최적화를 위한 MOOptimizer클래스, 질의처리의 대상인 후보객체 들을 저장하는 MOTableSet클래스, 그리고 후보객체에서 조건에 부합하는 데이터를 정제하기 위한 MOTableFilter 클래스로 구성된다.

테이블 관리자 모듈에 속하는 클래스로는 각 레이어에 대한 메타정보를 나태내기 위한 MOCatalog클래스, 각 카탈로그를 관리하기 위한 MOCatalogManager클래스, 레이어의 생성·소멸 및 데이터의 삽입·삭제를 관리해주는 MOTableManger클래스, 그리고 레이어를 질의처리기에 전달하기 위한 MODataTable클래스, MOData Column클래스, MODataRow클래스가 존재한다.

다음 절에서는 〈그림 7〉에서 나타난 주요한 클래스를 바탕으로 질의처리 과정에 대해 기술한다.



〈그림 6〉 이동객체 질의처리 컴포넌트 구성 클래스



〈그림 7〉 주요 클래스의 인터페이스

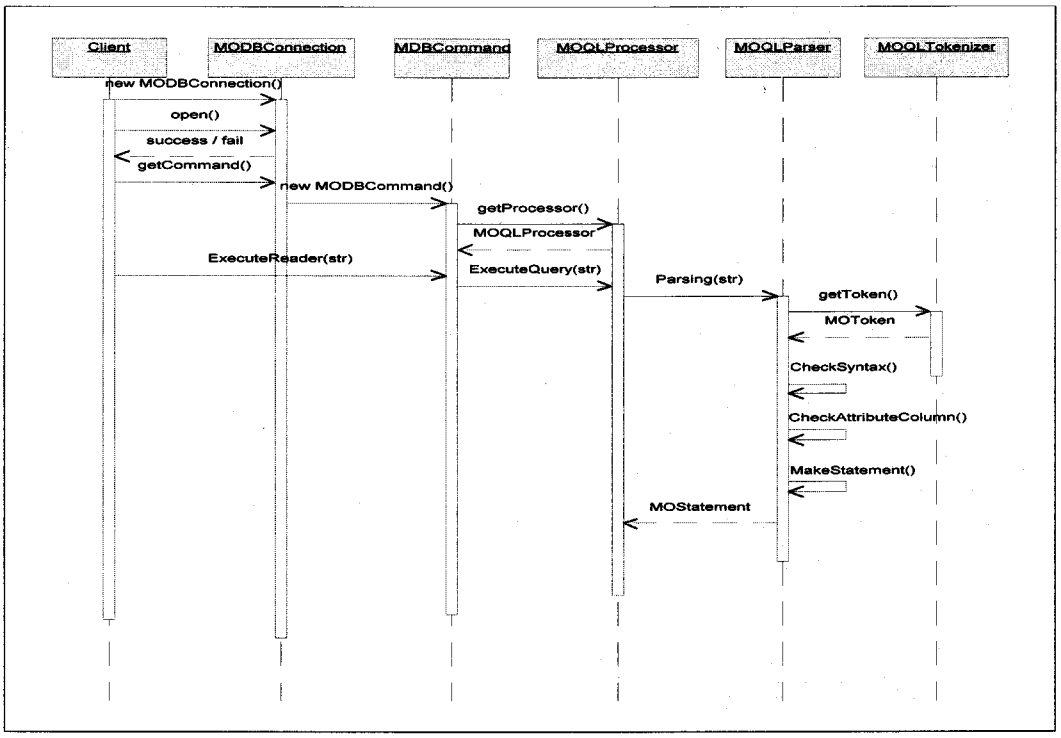
4.2.1 질의 분석

MOQL 질의문은 클라이언트에서 MODBConection 객체와 MODBCommand객체를 통하여 질의처리기의 MOQLProcessor객체에 전달된다. MOQLProcessor객체는 MOQLParser객체를 이용하여 질의를 분석한다. MOQLParser객체는 질의문의 문법이 올바른지 검사하거나 MOQLTokenizer객체를 통하여 잘못된 토큰이 존재하는지 검사해 나가면서 파싱트리(parsing tree)를 만들어 나간다. 이때, 이동객체 레이어의 필드이름에 해당하는 문자열은 비공간속성에 관련된 필드인지 검사하여 질의수행시 비공간 속성 관련 부분과 위치 관련 부분을 구분한다. 본 이동객체 질의처리 컴포넌트는 MOQL 질의문의 비공간 속성부분에 대해서는 직접적인 분석을 수행하지 않고, 내부적으로 하나의 문자열(string)을 생성하여 질의처리 시 비공간 속성을 담당하는 MSSQL 서버의 파라미터로 넘긴다. 분석된 파싱트리는 MOStatement객체로 표현되며 생성명령문, 소멸명령문, 삽입명령문, 삭제명령문, 선택명령문, 조건명령문의 6가지 타입 중 하나의 타입으로 정해진다. 질의분석 과정을 〈그림 8〉과 같다.

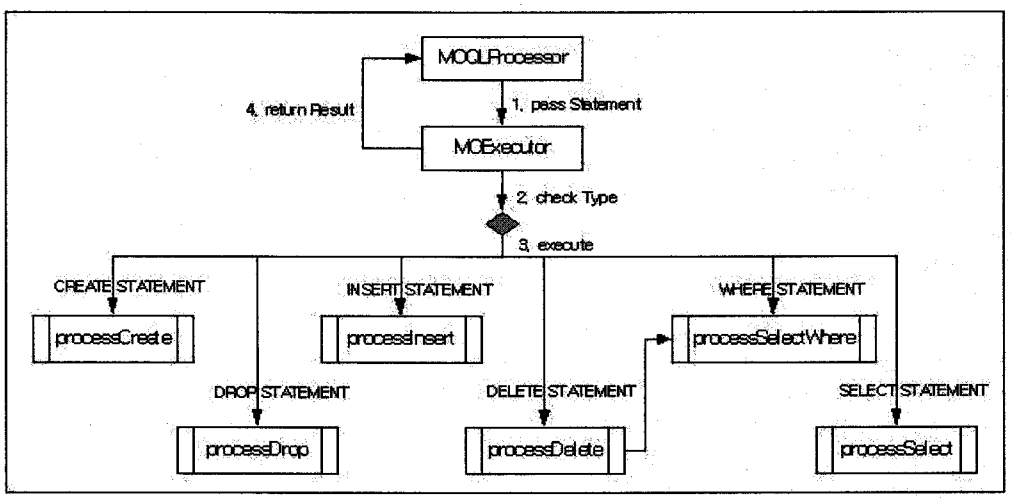
4.2.2 질의 수행

실제 질의 수행은 그림9에서와 같이 MOExecutor 객체가 담당한다. MOExecutor객체는 〈그림 9〉와 같이 분석된 질의문의 타입에 따라 해당 명령문을 수행한다. 〈그림 9〉에서 사각형은 클래스를 의미하고 삼중첩된 사각형은 프로세스를 의미한다.

레이어를 생성하기 위해서는 위치데이터베이스와 속성데이터베이스에 각각 새로운 테이블을 추가하고 위치정보와 속성정보의 연계를 위해서 레이어의 메타정보(Catalog)를 MOCatalogManager객체에 등록한다. 레이어 소멸은 생성 과정의 역순으로 제거해 나간다.



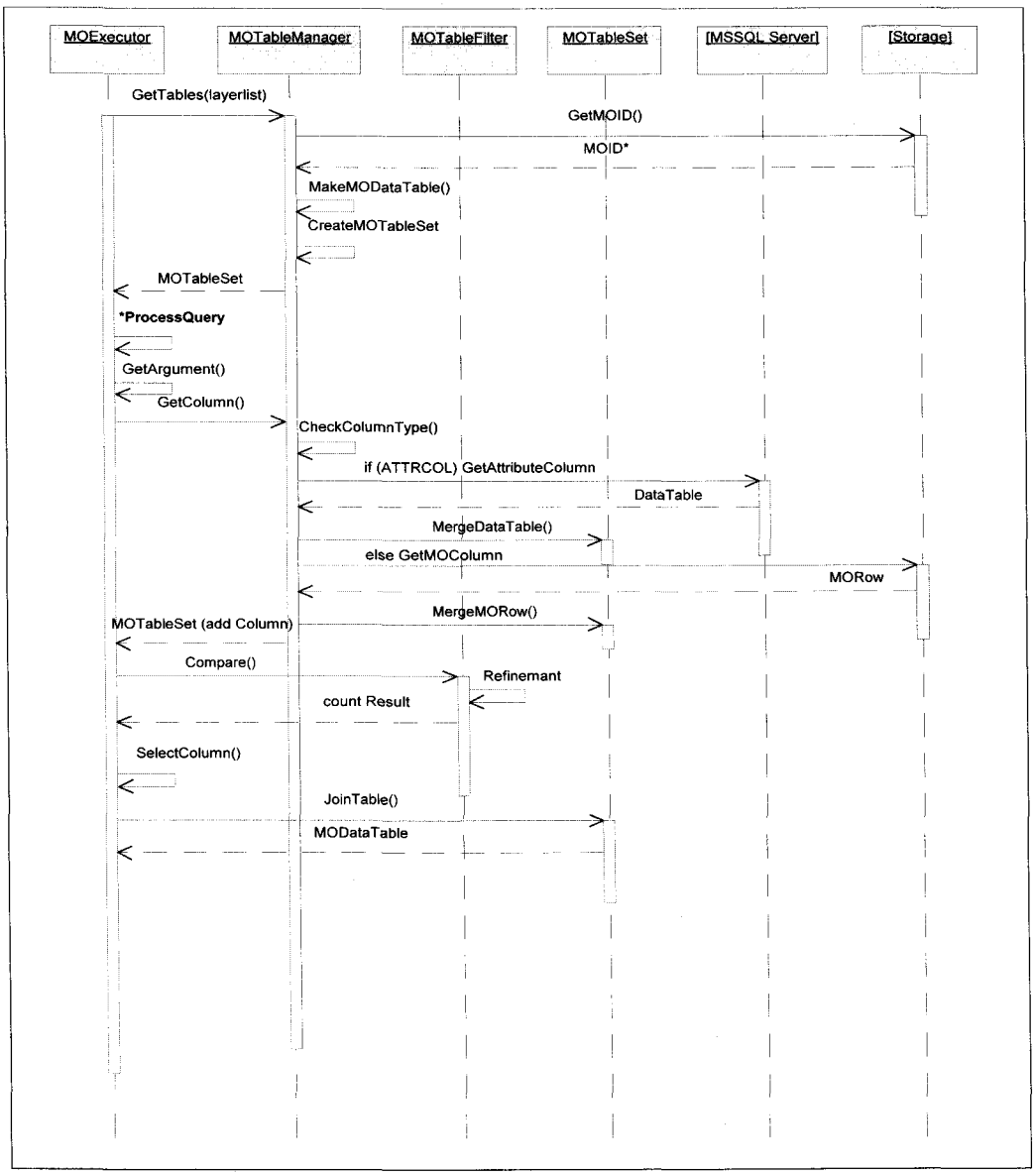
〈그림 8〉 질의 분석과정



〈그림 9〉 질의 수행과정

레코드 삽입은 이동객체 전체 궤적을 한꺼번에 삽입하는 경우와 기존에 입력된 이동객체의 위치정보를 계속 갱신해 나가는 경우가 존재한다. 전자의 경우는 새로운 레코드를 생성하여 위치저장시스템의 삽입 인터페이스를 통하여 레코드를 추가하고, 과거궤적에 대한 색인정보가 존재하는 경우에는 궤적색인의 삽입 인터페이스를 이용하여 궤적정보를 삽입한다. 후자의 경우는 미리 저장된 이동객체 레코드를 찾아서 마지막 저장위치 다음에 새로운 위치정보를 추가하고, 현재위치정보에 대한 색인이 존재하는 경우 현재위치색인의 갱신 인터페이스를 이용하여 위치정보를 갱신한다. 그리고, 마지막으로 비공간 속성부분을 삽입한다.

레코드의 삭제는 WHERE절 조건과 TO절 조건에 부합하는 레코드를 찾아서 삭제한다. WHERE절만



〈그림 10〉 질의처리기와 테이블관리자 상호관계

존재하는 경우는 조건에 부합하는 레코드의 해당 아이디(MOID)를 이용하여 위치정보와 비공간 속성정보를 모두 저장시스템에서 삭제시킨다. 그러나, TO절이 존재하게 되면 전체 정보를 모두 지우지 않고 과거 위치 데이터 저장이 시작된 시간에서 해당시간까지의 위치정보인 궤적의 일부를 삭제한다. 이때, 비공간속성에 관련된 데이터는 삭제하지 않는다.

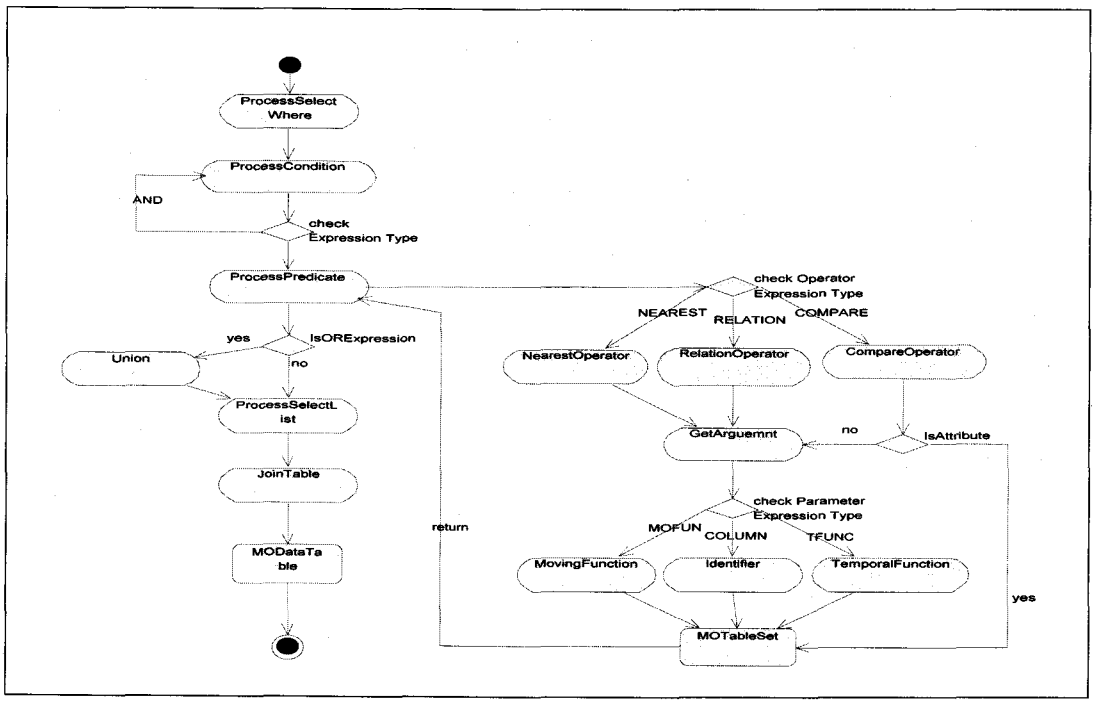
SELECT, FROM, WHERE로 구성된 조건질의는 FROM절에 명시된 레이어들에서 우선 MOID만을 읽어들여 후보테이블 셋을 구축하고 WHERE절의 비공간 속성과 관련된 조건을 처리한 후 위치정보를 처리한다. 〈그림 10〉은 질의처리기가 테이블 관리자에게 후보테이블을 요청하는 시점과 컬럼정보를 요청하는 순서를 보여주고 있다.

MOID만으로 우선 처리하는 이유는 궤적에 대한 연산자를 수행하기 위해서 조건에 따라 대용량의 궤적정보를 저장시스템에서 읽어오는 것이 부하가 심하고, 시공간 위상관계 연산자(SpatioTemporalRelation), 거리 연산자(Distance), 버퍼 연산자(Buffer)과 같은 연산자 들은 연산해야 할 궤적정보에 따라서 질의 성능을 저하시킬 수 있기 때문이다. 따라서, MOID에 관련된 조건 또는 비공간 속성에 관련된 조건을 미리 처리하여 후보객체를 최대한 줄여서 이동객체의 위치정보에 대한 연산자 수행의 비용을 줄이고자 한다. WHERE절의 수행 결과 후보테이블 셋에서 조건에 부합하는 레코드들만 남아있게 된다. 그리고 SELECT절에서 명시된 컬럼 정보를 추출하여 하나의 결과 테이블을 생성한다. 〈그림 11〉은 실제 질의처리를 담당하는 MOExecutor객체에서 조건질의가 어떻게 처리되는지를 도식화 하였다.

4.2.3 질의 최적화

본 이동객체 질의처리 컴포넌트는 세 가지를 고려하여 좀더 효율적으로 질의를 처리하도록 하고 있다.

첫째는 앞 절에서 간단히 설명한 바와 같이 질의를 수행하기 전에 후보테이블 생성 시 저장시스템으로부터 레이어의 MOID정보만 읽어오고 나머지 컬럼들은 제외시킨다. 만약, 질의를 처리하기 위해서 테이블 전체 또는 테이블에서 이동객체의 궤적정보를 모두 읽어온다면 읽어오는 시간뿐만 아니라 기본연산자를 수행하는데 있어서도 심각한 비용을 야기시킨다. 따라서, 질의처리시 가상적인 테이블을 대상으로 조건절을 수행하고 조건절을 계산하기 위한 컬럼정보만 읽어와서 처리하게 된다.



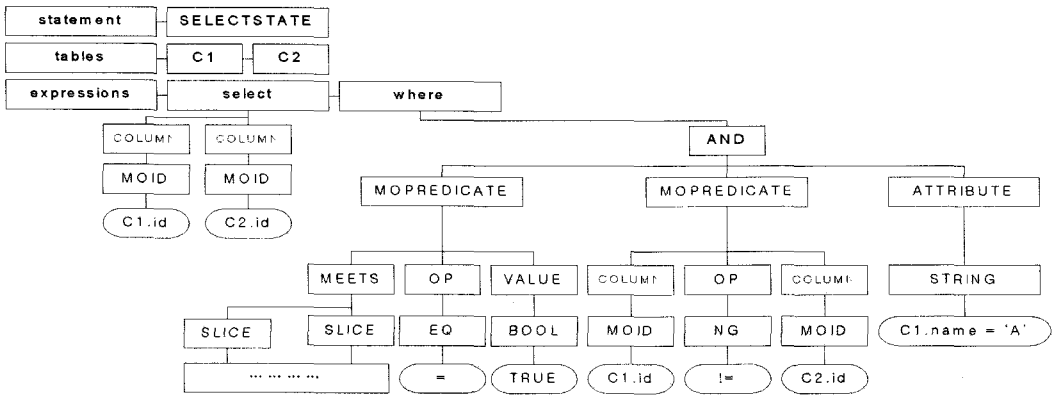
〈그림 11〉 조건질의의 수행과정

둘째는 조건절의 연산자가 하나 이상 나오는 경우는 우선순위에 따라 높은 우선순위의 연산자를 먼저 계산하도록 한다. 연산자 우선순위는 다음과 같다.

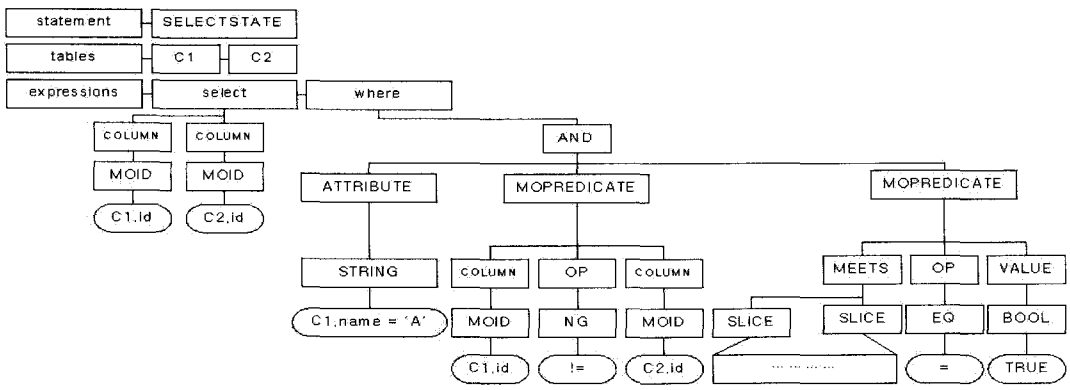
attribute 〉 moid column 〉 temporal relation 〉 trajectory relation 〉 sptiotemporal relation 〉 nearest

즉, 비공간 속성정보에 대한 조건이 존재하는 경우는 MSSQL서버을 이용하여 질의를 처리한 결과를 후보테이블 셋과 비교하여 서로 부합되는 MOID만 남겨두고 제거시킨다. 이때, MSSQL서버에서 구해온 정보를 후보테이블 셋에 복사한다. 마찬가지로 MOID관련된 조건절을 수행하여 후보객체를 줄여나간다. 그리고 나서, 이동객체에 대한 시간관계연산자, 위치정보에 대한 궤적연산자, 시공간관계연산자, 최근접연산자 순으로 처리한다. 예를 들면, 다음의 질의는 〈그림 12〉의 (a)와 같이 분석되고, 이를 연산자 우선순위에 따라 (b)의 형태로 재구성한다.

```
SELECT  C1.id, C2.id
FROM    CelluarPhoneUser as C1, Celluar
        PhoneUser as C2
WHERE   MEETS(SLICE(C1.position,
        '2003-11-23T12:30:40','2003-11-
        23T12:30:40'),
        SLICE(C2.position,'2003-11-23T
      12:30:40','2003-11-23T12:30:40'))
        = TRUE and
        C1.id != C2.id and name = 'A';
```



(a) 분석된 질의문



(b) 재구성된 질의문

〈그림 12〉 연산자 우선순위에 따른 분석질의문

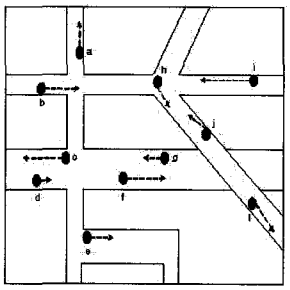
마지막으로 조인에 관련된 부분이다. 테이블 간의 조인 연산은 테이블의 후보객체가 많을수록 그 비용은 증가하게 된다. 따라서, 본 이동객체 질의처리 컴포넌트에서는 질의 조건 중 조인에 관여하는 테이블과 관여하지 않는 테이블을 구분하여 조인 연산을 수행하기 전에 각 테이블의 규모를 최대한 줄이고, 사용자가 원하는 정보만 컬럼들만 추출하여 조인연산을 수행한다.

4.3 실행결과

지금까지 이동객체 질의어를 정의하고 이동객체 질의 처리 컴포넌트의 수행 과정을 살펴 보았다. 이 절에서는 실제 질의문 몇 가지에 대하여 이동객체 질의처리 컴포넌트에서 어떻게 수행되는지 그 결과를 보여준다. 예를 들어, 다음과 같이 모바일 폰 사용자에 대한 레이어를 생성한다.

```
CelluarPhoneUser ( id MOID, position
                   MPOINT, name VARCHAR,
                   phonenumber VARCHAR )
```

생성된 레이어는 이동객체의 고유번호(즉, ID필드), 이동궤적(즉, POSITION 필드), 속성필드들인 이름(즉, NAME필드), 전화번호(즉, PHONENUBER필드)를 필드르 가진다. 〈그림 13〉 도로위의 모바일 폰 사용자에 대한 레이어를 도식화한 것이다.



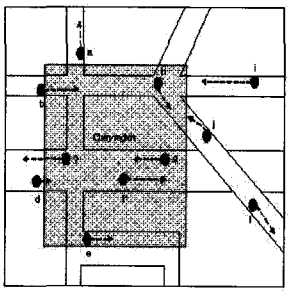
ID	POSITION	NAME	PHONENUMBER
111	{…, (4050, 10200, '2004-1-30T22:59:40), … }	a	013-111-2233
112	{…, (2950, 9800, '2004-1-30T22:59:40), … }	b	013-512-9832
113	{…, (3100, 5020, '2004-1-30T22:59:40), … }	c	013-863-1234
114	{…, (2800, 4890, '2004-1-30T22:59:40), … }	d	013-888-3498
…	……	……	……

〈그림 13〉 도로위의 모바일 폰 사용자 레이어 예

〔질의1〕 모바일 폰 사용자 중에서 특정시간 〔2004-1-30T22:59:40〕에 질의영역(3000, 10000, 3000, 10000) 안에 존재하는 사용자를 검색하라. (그림 14에서 화살표가 질의시간의 위치를 말한다.)

```
Select  id, position
From    CelluarPhoneUser
Where Withins( Snapshot
        (position,'2004-1-30T22:59:40'),
        RECTANGLE(3000,10000,3000,
              10000) ) = TRUE;
```

〈그림 14〉는 질의1을 수행한 결과를 도식화 한 것이다.



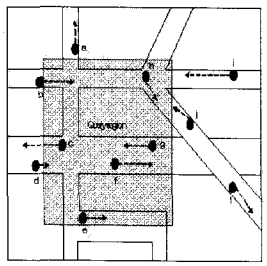
ID	POSITION	NAME	PHONENUMBER
111	{…, (4050, 10200, '2004-1-30T22:59:40), … }	a	013-111-2233
112	{…, (2950, 9800, '2004-1-30T22:59:40), … }	b	013-512-9832
113	{…, (3100, 5020, '2004-1-30T22:59:40), … }	c	013-863-1234
114	{…, (2800, 4890, '2004-1-30T22:59:40), … }	d	013-888-3498
…	……	……	……

〈그림 14〉 질의 1의 영역질의 예

〔질의2〕 모바일 폰 사용자 중에서 특정시간구간 〔2004-1-30T22:59:40,2004-1-30T23:00:00〕 사이에 질의점(10000,5500)에서 가장 가까운 사용자 3명을 검색하라.

```
SELECT  id, position
FROM    CelluarPhoneUser
WHERE   Nearest ( Slice(position,
        '2004-1-30T22:59:40',
        '2004-1-30T23:00:00'),
        POINT(10000,5500), 3) = TRUE;
```

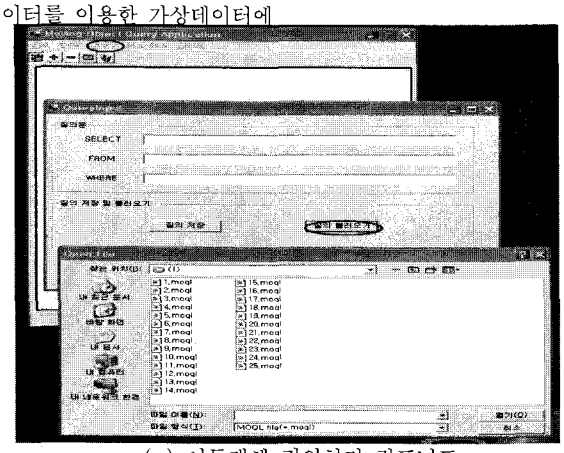
〈그림 15〉는 질의2을 수행한 결과를 도식화 한 것이다.



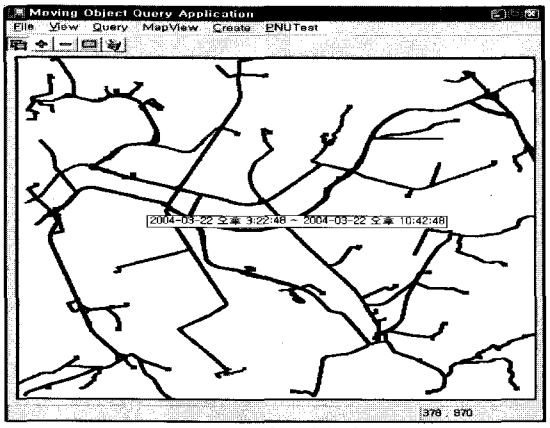
ID	POSITION	NAME	PHONENUMBER
111	{…, (4050, 10200, '2004-1-30T22:59:40), … }	a	013-111-2233
112	{…, (2950, 9800, '2004-1-30T22:59:40), … }	b	013-512-9832
113	{…, (3100, 5020, '2004-1-30T22:59:40), … }	c	013-863-1234
114	{…, (2800, 4890, '2004-1-30T22:59:40), … }	d	013-888-3498
…	……	……	……

〈그림 15〉 질의 2의 최근접질의 예

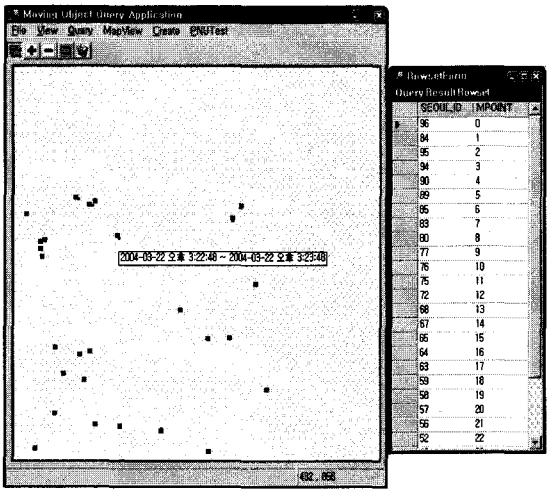
다음 〈그림 16〉은 실제 구현된 이동객체 질의처리 컴포넌트에서 서울시 도로데이터를 이용한 가상데이터에 대하여 질의를 처리한 결과를 보여주고 있다.



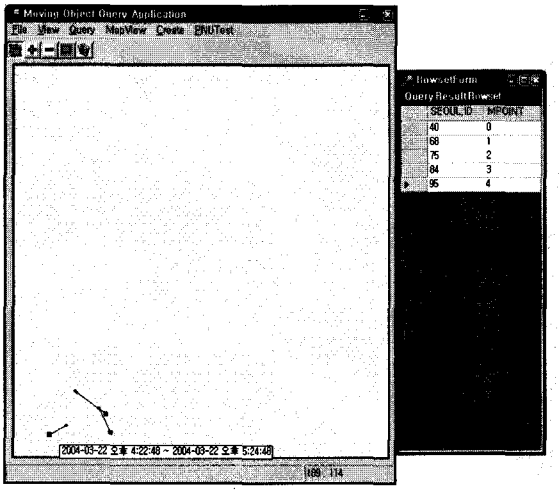
(a) 이동객체 질의처리 컴포넌트



(b) 전체 이동객체의 궤적



(c) 영역 질의 결과



(d) 최근접 질의 결과

〈그림 16〉 실제 데이터에 대한 질의 수행 결과

5. 결론

최근 위치기반 서비스와 같은 응용분야가 대두되면서 시간과 공간의 속성을 모두 가진 이동객체에 대한 기술적 요구가 많아졌다. 특히, 이동객체는 시간에 따라 공간적 속성이 연속적으로 변하므로 기존의 공간데이터베이스나 시간데이터베이스의 기술만으로는 그 특수성을 처리하기 적합하지 못하다. 따라서, 이동객체를 위한 새로운 데이터베이스 개발이 요구되어진다.

본 논문에서는 이동객체 데이터베이스를 위한 질의처리기를 개발하였다. 이동객체에 대한 질의를 처리하기 위해서는 이동객체에 대한 모델링뿐만 아니라 질의를 표현할 수 있는 질의어가 필요하다. 본 논문에서는 기존의 이동객체에 대한 질의어에 관한 연구를 바탕으로 SQL형태의 이동객체 질의어(MOQL)을 정의하였다. 정의된 질의어는 이동객체에 대한 기하학적 연산 질의뿐만 아니라 위상질의, 시공간질의, 최근접질의, 속성질의 등의 질의를 표현할 수 있다. 그리고, 정의된 질의어를 분석하여 질의처리 계획을 수립하고 수립된 계획을 바탕으로 질의를 처리해 나가는 질의처리기를 설계하고 구현하였다. 구현된 질의처리기는 이동객체에 대한 기본연산자를 조합해 나가며 질의조건에 부

합하는 결과를 추출하였다. 이때, 사용되는 기본연산자들은 이동객체 컴포넌트에서 구현된 연산자를 이용하였다. 또한, 질의를 좀도 효율적으로 처리하기 위해서 제공되는 이동객체를 위한 색인방법을 고려하고 있으며, 분리되어 있는 저장시스템의 위치정보와 속성정보를 서로 연결시켜 사용자에게 하나의 논리적인 테이블을 제공하도록 하였다. 본 논문에서는 이동객체에 대한 많은 응용분야에 질의처리기를 활용할 수 있도록 컴포넌트 형태로 개발되었고 기존의 ADO.NET인터페이스를 제공함으로써 그 활용 범위를 크게 기대할 수 있다. 향후 연구과제로는 웹 환경에서 다양한 서비스 제공을 위하여 이동객체를 표현하고 질의처리 할 수 있도록 XML 질의처리 언어인 XQuery를 바탕으로 확장해 나가도록 할 예정이다.

참고문헌

[1] 최혜옥, "위치기반서비스 (LBS, Location-Based Services)," 제3회 공간정보 워크샵, 2002, pp.5-22

[2] L. Forlizzi, R.H. Güting, E. Nardelli and M. Schneider, "A Data Model and Data Structures for Moving Objects Databases", Proc. ACM SIGMOD, 2000, pp.319-330

[3] O. Wolfson, S. Chamberlain, L. Jiang, "Moving Objects Databases: Issues and Solutions", Proc. SSDBM, 1998, pp.111-122

[4] 조대수, 남광우, 이재호, 민경욱, 장인성, 박종현, "대용량 위치 데이터 관리를 위한 정보 시스템", 한국정보과학회 데이터베이스연구회지, 18권 4호, 2002, pp.11-22

[5] 김경숙, 임복자, 남광우, 이기준, "이동객체 컴포넌트 설계 및 구현", 한국정보과학회 데이터베이스연구회지, 18권 4호, 2002, pp.50-57

[6] Open GIS Consortium, Inc., OpenGIS Simple Features Specification For SQL Revision 1.1, 1999

[7] M. Erwig and M. Schneider , "STQL: A Spatio-Temporal Query Language", Chapter 6 of Mining Spatio-Temporal Information Systems (eds. R. Ladner, K. Shaw, and M. Abdelguerfi), Kluwer Academic Publishers, 2002 , pp.105-126

[8] A. P. Sistla, O. Wolfson, S. Chamberlain and S. Dao, "Modeling and Querying Moving Objects", Proc. ICDE, 1997, pp.422-432

[9] T. Abraham and J.F. Roddick, "Survey of Spatio-Temporal Databases", GeoInformatica 3(1), 1999, pp.61-99

[10] T. K. Sellis, "CHOROCHRONOS: Research on Spatiotemporal Database Systems", DEXA Workshop, 1999, pp.452-456

[11] M. Erwig, R.H. Güting, M. Schneider and M. Vazirgiannis, "Spatio-Temporal Data Types: An Approach to Modeling and Querying Moving Objects in Databases", GeoInformatica 3(3), 1999, pp. 269-296

[12] R.H. Güting, M.H. Böhlen, M. Erwig, C.S. Jensen, N. A. Lorentzos, M. Schneider and M.V. Athens, "A Foundation for Representing and Querying Moving Objects", ACM TODS, Volume 25, 2000, pp.1-42

[13] M. Erwig and M. Schneider, "Developments in Spatio-Temporal Query Languages", IEEE Int. Workshop on Spatio-Temporal Data Models and Languages, 1999, pp.441-449

[14] O. Wolfson, S. Chamberlain, A.P. Sistla, B. Xu, J. Zhou, "DOMINO: Databases fOr MovINg Objects tracking", Proc. ACM SIGMOD, 1999, pp.547-549

[15] C.X. Chen and C. Zaniolo, "SQLST : A Spatio-Temporal Data Model and Query Language", Proc. ER, 2000, pp.96-111

[16] J. Su, H. Xu, and O. Ibarra, "Moving objects: Logical relationships and queries", Proc. SSTD, 2001, pp.3-19

[17] H. Mokhtar, J. Su and O. Ibarra, "On Moving Object Queries", Proc. ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, 2002, pp.188-198

[18] 이현아, 이혜진, 김동호, 김진석, "이동체 관리 시스템을 위한 이동체 질의어 설계", 정보과학회 2003년 추계학술대회 VOL. 30 NO. 2-2, 2003, pp.148 -150

[19] 윤재관, 장유정, 한기준, "대용량 위치 데이타를

위한 분산 위치 저장 컴포넌트의 개발," 한국정보과학회 데이타베이스 연구, 18권4호, 2002, pp. 67-80

[20] 전희철, 안성우, 김진덕, 홍봉희, "이동체 데이터베이스에서 과거, 현재 및 미래위치 질의를 위한 통합 색인", 한국정보과학회 2003 가을 학술발표논문집(II) 제30권2호, 2003, pp121-123

[21] 박현규, 김명호, "위치 기반 서비스에서 연속 근사 질의 처리", 한국정보과학회 데이터베이스연구회지, 18권 4호, 2002, pp.58-65

김경숙

1999년 부산대학교 전자계산학과 졸업(학사)

2001년 부산대학교 전자계산학과 졸업(석사)

2001년~현재: 부산대학교 대학원 전자계산학과 박사과정.

관심분야: 이동객체DB, GIS, Telematics

권오제

2003년 부산대학교 컴퓨터공학과 졸업(학사)

2004년~현재 부산대학교 컴퓨터공학과 석사과정

관심분야: 이동객체DB, GIS

변희영

2003년 부산외국어대학교 컴퓨터공학과 졸업(학사)

2004년~현재 부산대학교 컴퓨터공학과 석사과정

관심분야: 이동객체DB, GIS

조대수

1995년 부산대학교 컴퓨터공학과 (학사)

1997년 부산대학교 컴퓨터공학과 (석사)

2001년 부산대학교 컴퓨터공학과 (박사)

2001년~현재 한국전자통신연구원 텔레매틱스 연구단 LBS 연구팀 선임연구원

관심분야 : GIS, 공간DB, LBS, 이동체DB

김태완

1988년 연세대학교 경영학과(학사)

1991년 미국 피츠버그대학교 전자계산학과(학사)

1994년 미국 텍사스 A&M대학교 전자계산학과(석사)

2003년 부산대학교 대학원 전자계산학과 이학박사

2003년~현재 부산대학교 컴퓨터 및 정보통신 연구소 전임연구원

관심분야: 다차원색인, GIS, 시공간DB

이기준

1984년 서울대학교 계산통계학과 졸업(학사).

1986년 서울대학교 계산통계학과 (전자계산학 전공)졸업(석사).

1992년 프랑스 응용과학원 전자계산학과 전산학박사.

1993년~현재 부산대학교 전자전기정보컴퓨터 공학부 부교수

관심분야: 공간DB, 시공간DB, GIS, Telematics 및 Ubiquitous Computing