공간 모바일 장치를 위한 내장형 공간 MMDBMS의 설계 및 구현†

Design and Implementation of an Embedded Spatial MMDBMS for Spatial Mobile Devices

박지웅*, 김정준**, 윤재관**, 한기준**
Ji-Woong Park, Joung-Joon Kim, Jae-Kwan Yun, Ki-Joon Han

요 약 최근에 무선 통신의 발달과 더불어 모바일 컴퓨팅에 대한 관심이 높아지고 있다. 모바일 컴퓨팅은 사용자가 PDA, 노트북 등의 이동 가능한 모바일 장치를 휴대하고 무선 통신을 통해서 서버 컴퓨터와 자원을 함께 공유하는 환경이라 할 수 있다. 모바일 데이타베이스는 이러한 모바일 장치에 사용되는 데이타베이스를 말한다. 모바일 데이타베이스의 응용 분야로는 보험업무, 금융업무, 의료 등이 있지만, 특히 사용자의 위치 정보를 활용하는 위치 기반 서비스(LBS: Location Based Service)가 중요한 응용 분야로 등장하고 있다. 이러한 모바일 환경에서 위치 기반 서비스를 제공하기 위해서는 공간 모바일 장치에서 대용량의 공간 데이타를 효율적으로 관리하기 위한 내장형 공간 MMDBMS(Main-Memory Database Management System)가 필요하다.

이에 본 논문에서는 기존의 PC용 MMDBMS인 HSQLDB를 확장하여 공간 모바일 장치에서 공간 데이타를 효율적으로 관리할 수 있는 내장형 공간 MMDBMS를 설계 및 구현하였다. 내장형 공간 MMDBMS는 ISO(International Organization for Standardization)의 공간 데이타 모델을 따르며, 공간 데이타 특성에 적합한 압축 기법인 산술 연산 코딩 기법을 제공하고, 공간 모바일 장치에 적합한 MBR 압축 및 해싱 기법을 이용한 공간 인덱스를 지원한다. 그리고, 공간 모바일 장치의 낮은 성능의 프로세서에서 공간 데이타 디스플레이 기능을 제공하고, 내장형 공간 MMDBMS와 GIS 서버 사이에서 공간 데이타 수입/수출의 성능 향상을 위한 데이타 캐싱과 동기화 기능을 지원한다.

Abstract Recently, with the development of wireless communications and mobile computing, interest about mobile computing is rising. Mobile computing can be regarded as an environment where a user carries mobile devices, such as a PDA or a notebook, and shares resources with a server computer via wireless communications. A mobile database refers to a database which is used in these mobile devices. The mobile database can be used in the fields of insurance business, banking business, medical treatment, and so on. Especially, LBS(Location Based Service) which utilizes location information of users becomes an essential field of mobile computing. In order to support LBS in the mobile environment, there must be an Embedded Spatial MMDBMS(Main-Memory Database Management System) that can efficiently manage large spatial data in spatial mobile devices.

Therefore, in this paper, we designed and implemented the Embedded Spatial MMDBMS, extended from the HSQLDB which is an existing MMDBMS for PC, to manage spatial data efficiently in spatial mobile devices. The Embedded Spatial MMDBMS adopted the spatial data model proposed by ISO(International Organization for Standardization), provided the arithmetic coding method that is suitable for spatial data, and supported the efficient spatial index which uses the MBR compression and hashing method suitable for spatial mobile devices. In addition, the system offered the spatial data display capability in low-performance processors of spatial mobile devices and supported the data caching and synchronization capability for performance improvement of spatial data import/export between the Embedded Spatial MMDBMS and the GIS server

주요어 : LBS, 공간 모바일 장치, MMDBMS, 공간 인덱스
KeyWords : LBS, Spatial Mobile Devices, MMDBMS, Spatial Index

† 본 연구는 정보통신부 및 정보통신 연구진흥원의 대학 IT연구센터 육성, 지원사업의 연구결과로 수행되었음.

* 경기공업대학 컴퓨터정보시스템과 jiwpark@kinst.ac.kr
** 건국대학교 컴퓨터공학부 {jjkim9, jkyun, kjhan}@db.konkuk.ac.kr

1. 서 론

최근 정보통신 기술의 발전에 기인하여 편리성과 신속성을 추구하기 위한 휴대형 정보 기기를 통한 업무처리의 활용이 전산업에 걸쳐 많이 요구되고 있다. 이러한 추세에 발맞추어 노트북, 개인정보 단말기(PDA), 인터넷 접속이 가능한 스마트 폰 등의 기술과 이동 통신 기술이 비약적으로 발전하여 모바일 장치가 널리 보급되었고, 이러한 모바일 장치들의 발전은 언제 어디서나 원하는 정보를 접근할 수 있는 이동 컴퓨팅 시대를 이끌었다[1][2][3].

모바일 데이타베이스는 이러한 모바일 장치에 사용되는 데이타베이스를 말한다. 이것은 단지 클라이언트 시스템에서만 운영되는 데이타베이스라는 의미가 아니라 원격지에 있는 데이타베이스 서버로부터 모바일 장치에 있는 모바일 데이타베이스까지 자연스럽게 연결이 되고, 복제를 통해서 데이타를 늘 갱신할 수 있어야함을 의미한다. 모바일 데이타베이스 응용 분야로서는 보험업무, 금융업무, 의료업무 등이 있지만 GIS(Geographic Information System)와 연계된 사용자의 위치 정보를 활용하는 위치 기반 서비스가 중요한 응용 분야로 등장하고 있다[3][4]. 특히, 본 논문에서 공간 모바일 장치는 위치 기반 서비스를 위해 공간 데이타를 사용하는 모바일 장치를 의미한다.

현재 공간 모바일 장치는 기존 PC에 비해 작은 용량의 저장 공간과 낮은 성능의 프로세서를 사용하고 있다. 그리고, 공간 모바일 장치는 자체 화일 시스템을 기반으로 데이타를 처리하고 있다. 이는 작은 용량의 데이타를 처리하기는 편리할지 모르나 나날이 증가하고 있는 대용량의 공간 데이타를 관리하거나 매우 복잡하고 다양한 공간 질의를 처리하기에는 큰 문제가 있다[5][6]. 이러한 문제는 공간 모바일 장치에서 내장형 공간 MMDBMS(Main-Memory Database Management System)를 이용해 공간 데이타를 관리함으로써 해결될 수 있다[2].

본 논문에서는 메모리 기반 관계형 DBMS인 HSQLDB[7]에 공간 데이타 타입과 공간 연산자, 공간 데이타 특성에 적합한 산술 코딩 압축 기법, 공간 모바일 장치에 적합한 MBR 압축 및 해쉬 기법을 이용한 공간 인덱스, GIS 서버와의 동기화, 그리고 공간 모바일 장치에서 디스플레이 기능과 서버와의 통신 성능을 높이기 위한 데이타 캐쉬 기능을 추가 및 확장하여 공간 모바일 장치에서 공간 데이타를 효과적으로 관리할 수 있는 내장형 공간 MMDBMS를 설계 및 구현하였다.

본 논문의 구성은 다음과 같다. 1장의 서론에 이어 2장에서는 관련 연구로 공간 모바일 장치용 내장형 공간 MMDBMS의 요구사항, ISO/DIS 19107 명세, HSQLDB에 대해 살펴본다. 그리고, 3장에서는 내장형 공간 MMDBMS의 구조도와 내장형 공간 MMDBMS의 기능을 설명하고, 4장에서는 내장형 공간 MMDBMS의 각 모듈에 대한 상세한 설명과 성능 테스트, 그리고 결과 화면에 대해 기술한다. 마지막으로 5장에서는 결론 및 향후 연구 과제에 대하여 언급한다.

2. 관련 연구

본 장에서는 공간 모바일 장치용 내장형 공간 MMDBMS의 요구 사항, ISO/DIS 19107 명세, 그리고 HSQLDB에 대해 설명한다.

2.1 내장형 공간 MMDBMS의 요구 사항

공간 모바일 장치를 위한 내장형 공간 MMDBMS는 초경량 실시간 DBMS 기술 분야에 해당하며, 공간 모바일 장치의 단점인 느린 처리 속도와 작은 저장 공간의 한계를 극복하기 위한 기술이 필요하다. 또한, 다른 시스템에 비해 매우 큰 용량의 공간 데이타를 처리하는 GIS 기능을 공간 모바일 장치에서 수행하기 위해서는 중앙 서버의 대용량 데이타를 공간 모바일 장치에 적합한 데이타로 변환하고 압축하는 공간 데이타 변환[1] 및 동기화 기술[8][9]이 필요하다. 그리고, 공간 모바일 장치에서 활용할 수 있는 데이타로 변환하기 위한 공간 데이타 필터링 기술, 공간 모바일 장치에서 고밀도로 압축된 데이타를 빠르게 검색하기 위한 모바일 공간 인덱스 기술[10], 공간 모바일 장치에서 데이타의 디스플레이 성능을 높이기 위한 지도 캐쉬 기술[2] 등이 요구된다. 특히, 내장형 공간

MMDBMS는 공간 데이타 지원을 위한 공간 데이타 타입과 공간 연산자[11]를 지원해야 한다.

2.2 ISO/DIS 19107 명세

본 절에서는 ISO/DIS 19107 명세에 제시된 8개의 공간 데이타 타입과 18개의 공간 연산자를 소개한다[11]. ISO/DIS 19107에서 제공하는 공간 데이타 타입과 공간 연산자는 5개의 Geometry 팩키지와 3개의 Topology 팩키지, 그리고 12개의 연산으로 구성된다. Geometry 팩키지는 좌표 기하를 위한 다양한 데이타 유형들을 제공하는 팩키지이다. <표 1>은 Geometry 팩키지를 보여준다.

<표 1> Geometry 팩키지

Geometric root package	GM_Object
Geometric primitive package	GM_Boundary, GM_ComplexBoundary, GM_PrimitiveBoundary, GM_Point, GM_Curve, GM_Solid
Coordinate geometry package	GM_PointRef, GM_Position, GM_LineString, GM_Arc, GM_Circle, GM_Polygon
Geometric aggregate package	GM_MultiPoint, GM_MultiCurve, GM_MultiSurface, GM_MultiSolid
Geometric complex package	GM_CompositePoint, GM_CompositeCurve, GM_CompositeSurface, GM_CompositeSolid

Topology 팩키지는 기하 연산이 많은 양의 계산을 필요로 하므로 기하 연산을 신속하게 실행하기 위한 팩키지이다. <표 2>는 Topology 팩키지를 보여준다.

<표 2> Topology 팩키지

Topology root package	TP_Object
Topological primitive package	TP_Boundary, TP_Ring, TP_Edge, TP_Solid, TP_Face
Toplogical complex package	TP_Complex

연산은 Geometry 팩키지와 Topology 팩키지에 정의된 데이타 유형을 위한 연산자 팩키지이다. <표 3>은 연산을 보여준다.

<표 3> 연산

representativePoint	객체에 위치하는 점 객체를 돌려주는 연산
boundary	객체의 경계에 위치하는 점들을 포함하는 객체 유한 집합을 돌려주는 연산
dimension	객체의 고유 차원 값을 돌려주는 연산
envelope	객체의 최소경계사각형을 돌려주는 연산
centroid	객체의 수학적 중심을 연산
contain	두 객체간의 포함 관계를 연산
intersects	두 객체간의 교차 관계를 연산
equals	두 객체간의 동일성 연산
intersection	두 객체간의 교집합 연산
union	두 객체간의 합집합 연산
difference	두 객체간의 차집합 연산

2.3 HSQLDB

2.3.1 HSQLDB 소개

HSQLDB는 순수 자바로 만들어진 메모리 기반 관계형 DBMS이다[7]. HSQLDB는 Hypersonic SQL 프로젝트로 시작하여 1988년 처음 배포되었다. 그리고, Hypersonic SQL의 몇몇 개발자들에 의해 HSQLDB 개발 그룹이 형성되었고, 2004년 7월에 1.7.2 버전이 배포되었다. HSQLDB의 특징은 순수 자바로 구현되었기 때문에 시스템에 독립적이며 표준 ANSI-92 SQL과 JDBC 인터페이스를 지원한다. 그리고, 빠른 검색을 위한 B+-tree를 지원하며, 데이타 무결성을 위해 트랜잭션 지원과 Outer Join, Inner Join을 지원한다. 또한, View, Trigger와 Order by, Group by, Having을 지원하며, Count, Sum, Min, Max, Avg 함수를 제공한다. 그리고, 사용자는 SQL script 화일을 이용하여 데이타베이스를 생성할 수 있다.

2.3.2 HSQLDB 구성

HSQLDB는 각각 3개의 팩키지와 2개의 인터페이스, 그리고 21개의 클래스로 구성된다. <표 4>는 HSQLDB의 팩키지, 인터페이스, 클래스에 대한 각각의 기능을 보여준다.

<표 4> HSQLDB의 팩키지, 인터페이스, 클래스

팩키지

Org.hsqldb	JDBC를 이용한 연결 구현
Org.hsqldb.lib	HsqlDateTime,StringInputStream,HsqlDeque에 대한 구현
Org.hsqldb.util	인터페이스에 대한 구현

Org.hsqldb 인터페이스

HsqlSocketRequestHandler	HsqlSocketRequestHandler 인터페이스 구현
Trigger	Trigger 인터페이스 구현

Org.hsqldb.util 인터페이스

ZaurusComponent	ZaurusComponent 인터페이스 구현

Org.hsqldb 클래스

JdbcConnection	database 연결 구현
JdbcDriver	주어진 URL을 이용하여 database 연결 구현
Server	client-server 모드의 연결 구현
Servlet	applet을 이용한 client / server 모드 연결 구현
Trace	error messages 와 throwing SQLException 처리 구현
WebServer	HTTP를 이용한 client / server 모드 연결 구현

Org.hsqldb 클래스

Index	B-tree 인덱스 구현
Transaction	transaction 처리 구현
View	select query에 기반한 views 구현
User	사용자 인증을 위한 id, password 구현
Log	a .properties file, a .script file a .data file and a .backup file 로 구성된 로그파일 구현
Function	SQL function 와 stored procedure calls 구현

Org.hsqldb.lib 클래스

ArrayUtil	배열의 operations을 위한 정적 메소드들의 집합 구현
HsqlDateTime	date / time / datetime 구현
StringInputStream	String 형태의 bytes를 가져오기 위한 subclass 구현

Org.hsqldb.util 클래스

DatabaseManager	JDBC database관리를 위한 AWT Tool 구현
QueryTool	query 생성 및 실행을 위한 구현
ScriptTool	sql script 생성 및 실행을 위한 구현
Transfer	JDBC를 이용하여 다른 databases와 table 변환 구현
ZaurusConnectionDialog	zaurus GUI를 위한 컴포넌트 구현
ZaurusEditor	zaurus를 위한 search/view/update/insert editor 구현

<표 4>에서와 같이 팩키지는 JDBC와 인터페이스들이 정의된 org.hsqldb, org.hsqldb.lib, ogr.hsqldb.util로 구성되어 있고, org.hsqldb 인터페이스와 org.hsqldb.util 인터페이스는 서버와의 소켓과 트리거에 대한 인터페이스 등이 정의되어 있고, org.hsqldb 클래스는 HSQLDB의 실행 모드에 대한 클래스들과 질의 처리, 데이타베이스 복구, 사용자 인증에 관한 클래스들로 구성된다. 그리고, org.hsqldb.lib 클래스와 org.hsqldb.util 클래스는 HSQLDB에서 사용되는 라이브러리들과 사용자 인터페이스에 대한 클래스들로 구성된다.

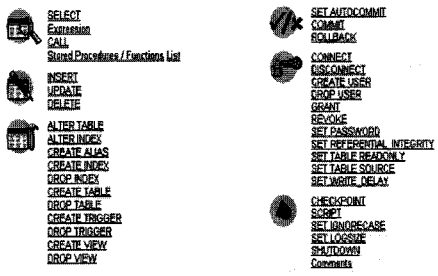
2.3.3 HSQLDB의 데이타 타입과 SQL

HSQLDB는 18개의 데이타 타입을 지원하고 SQL(DML, DDL, DCL)을 지원한다. <표 5>는 HSQLDB 데이타 타입들의 이름과 범위, 그리고 데이타 타입들을 정의하기 위한 자바 언어 타입을 보여주고 있다.

<표 5> HSQLDB의 데이타 타입

NAME	RANGE	JAVATYPE
Integer | int	As java type	"int" | "java.lang.integer"
Double[precision] | float	As java type	"double" | "java.lang.double"
Varchar	Integer.MAXVALUE	"java.lang.string"
Char | character	Integer.MAXVALUE	"java.lang.string"
Longvarchar	Integer.MAXVALUE	"java.lang.string"
Date	As java type	"java.sql.date"
Time	As java type	"java.sql.time"
Timestamp | datetime	As java type	"java.sql.timestamp"
Decimal	No limit	"java.math.bigdecimal"
Numeric	No limit	"java.math.bigdecimal"
Bit	As java type	"boolean" | "java.lang.boolean"
Smallint	As java type	"short" | "java.lang.short"
bigint	As java type	"long" | "java.lang.long"
Real	As java type	"double" | "java.lang.double"
Binary	Integer.MAXVALUE	"byte[]"
Varbinary	Integer.MAXVALUE	"byte[]"
Longvarbinary	Integer.MAXVALUE	"byte[]"
Other | object	Integer.MAXVALUE	"java.lang.object"

<그림 1>은 HSQLDB에서 지원하는 SQL을 보여준다.



<그림 1> HSQLDB의 SQL

HSQLDB는 데이타를 검색하거나 변경하는 데이타 조작 언어(DML)인 insert, update, delete와 데이타의 구조를 정의하는 데이타 정의 언어(DDL)인 create, alter, drop, 그리고 데이타베이스 사용자에게 부여한 특권을 정의하는 데이타 제어 언어(DCL)인 grant, revoke, commit 등을 지원한다.

2.3.4 HSQLDB의 개선 사항 .

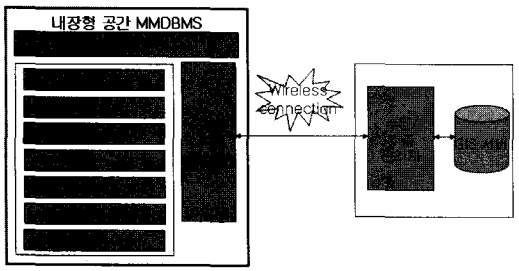
본 논문에서는 2.1절의 요구사항을 충족하기 위해 HSQLDB를 공간 모바일 장치를 위한 내장형 공간 MMDBMS로 확장하였다. 즉, HSQLDB에 ISO/DIS 19107[11]의 공간 데이타 타입 및 공간 연산자, GIS 서버와 공간 데이타 수입/수출시 효율성을 높이기 위해 공간 데이타에 적합한 산술 코딩 압축 기법과 데이타 캐쉬 기능, GIS 서버와의 동기화 기능, 그리고 빠른 검색을 위해 공간 모바일 장치용 공간 인덱스에 적합한 MBR 압축 및 해쉬 기법을 이용한 공간 모바일 장치용 공간 인덱스를 추가하였다.

3. 내장형 공간 MMDBMS의 설계

본 장에서는 공간 모바일 장치용 내장형 공간 MMDBMS의 구조, 공간 데이타 타입 및 공간 연산자, 공간 데이타 압축, 공간 인덱스에 대해서 설명한다.

3.1 내장형 공간 MMDBMS의 구조

본 논문에서 개발한 내장형 공간 MMDBMS의 주된 기능은 모바일 환경에서 GIS 서버의 공간 데이타를 수입/수출하여 내장형 공간 MMDBMS에 공간 데이타를 저장하고 검색하는 것이다. 내장형 공간 MMDBMS은 크게 인터페이스 관리자, 디스플레이 관리자, 데이타 캐쉬 관리자, 트랜잭션 관리자, 공간 인덱스 관리자, 데이타 압축 관리자, 수입/수출 관리자, 동기화 관리자로 구성된다. <그림 2>는 내장형 공간 MMDBMS의 전체 구조를 보여준다.



<그림 2> 내장형 공간 MMDBMS의 전체 구조

수입/수출 관리자와 데이타 캐쉬 관리자는 내장형 공간 MMDBMS와 GIS 서버 간에 데이타 요청시 공간 데이타 교환 기능을 제공하고, 인터페이스 관리자와 디스플레이 관리자는 사용자로부터 질의 입력과 디스플레이기능을 제공한다. 데이타 압축 관리자는 공간 데이타 수입/수출시 공간 데이타 압축 기능을 제공하며, 질의 처리 관리자와 트랜잭션 관리자는 SQL문을 검사, 분석 및 처리하는 기능을 제공한다. 그리고, 동기화 관리자는 내장형 공간 MMDBMS와 GIS 서버 사이에 데이타를 일치시켜 주는 기능을 제공하고, 공간 인덱스 관리자는 공간 모바일 장치를 위한 공간 인덱스를 제공한다.

3.2 공간 데이타 타입 및 연산자

<표 6>은 ISO/DIS 19107 공간 데이타 타입 및 공간 연산자와 내장형 공간 MMDBMS의 공간 데이타 타입 및 공간 연산자간의 관계를 보여준다.

<표 6> 공간 데이타 타입 및 공간 연산자 관계

공간 데이타 타입 관계

ISO/DIS 19107	내장형 공간 MMDBMS
GM_Point	point
GM_LineSegment	simpleline
GM_Curve	polyline
GM_PolyhedralSurface	polygon
GM_Envelope	rectangle
GM_Circle	circle
GM_Polygon	hpolygon
GM_Complex	shape

공간 연산자 관계

ISO/DIS 19107	내장형 공간 MMDBMS
contain	contain
buffer	cover
disjoint	disjoint
equals	equal
intersects	crossover
intersection	overlap
touch	touch
area	area
centroid	center
distance	distance
length	length
nearest	nearest
edges	edges
transform	translation
union	union
difference	difference
envelope	boundary

<표 7>은 내장형 공간 MMDBMS에 추가된 공간 데이타 타입과 공간 데이타 타입 표현을 보여주고, <표 8>은 내장형 공간 MMDBMS의 공간 연산자들의 정의를 보여준다. 공간 연산자는 위상 연산자(contain, cover, disjoint, equal, crossover, overlap, touch)와 기하 연산자(area, center, distance, direction, length, nearest, edges)로 구성된다.

<표 7> 공간 데이타 타입 표현

공간 데이타 타입	공간 데이타 타입 표현
point	POINT(3,7)
simpleline	SIMPLELINE((1,2), (3,5))
polyline	POLYLINE((2,1), (3,4), (6,8) ... , (4, 1))
polygon	POLYGON((1,1), (5,7), (10,15) ... , (1,1))
rectangle	RECTANGLE((2, 5), (4, 10))
circle	CIRCLE((1,5), 3.568)

<표 8> 공간 연산자 정의

공간 연산자 (위상 연산자)	정의
operand1 **contain** operand2	첫번째 피연산자가 두번째 피연산자를 포함하고 있으며 경계값 간의 교집합이 없는 경우 참을 반환
operand1 **cover** operand2	첫번째 피연산자가 두번째 피연산자를 포함하고 있으며 경계값 간의 교집합이 있는 경우 참을 반환
operand1 **disjoint** operand2	두 피연산자간의 교집합이 없으면 참을 반환
operand1 **equal** operand2	두 피연산자가 동일하면 참을 리턴
operand1 **crossover** operand2	두 피연산자가 내부값간의 교집합의 차원이 두 피연산자 중 큰 차원을 갖는 피연산자의 차원보다 한차원 작아야 하며, 두 피연산자 사이의 포함관계가 없어야 함
operand1 **overlap** operand2	두 피연산자가 같은 차원을 가져야 하며 내부값간의 교집합의 차원이 피연산자의 차원과 동일하여야 하며, 두 피연산자 사이의 포함관계가 없어야 함
공간 연산자 (거하 연산자)	**정의**
area operand	공간객체의 면적을 구함
center operand	공간객체의 무게중심을 구함
operand1 **distance** operand2	두 공간객체간의 최소거리를 구함
operand1 **direction** operand2	두 공간객체의 무게중심을 이은 직선과 X 축과의 각을 구함
length operand	선이나 면 객체타이를 가진 operand 의 길이나 둘레길이를 구함
operand1 **nearest** operand2 n	operand2 를 기준으로 하여 operand1에서 가장 거리가 가까운 n개의 객체를 리턴
edges operand	polyline 이나 polygon 객체타입을 가진 operand 로부터 n 번째 변을 구함

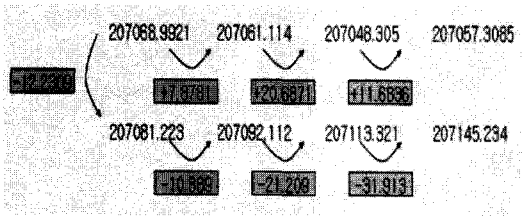
3.3 공간 데이타 압축

본 논문에서는 공간 데이타 압축 방법으로 기준점과의 거리차를 이용한 산술 연산 코딩 기법을 적용하였다[1][12]. 일반적으로 GIS에서 가장 많은 크기를 차지하는 것은 Point나 SimpleLine이 아닌 MultiLine이나 Polygon이다. 이러한 MultiLine이나 Polygon은 특정 부위에 대하여 클러스터링된 형태를 띠게 된다. 그러므로, 클러스터링되어 있는 MultiLine이나 Polygon을 기준으로 하여 거리의 차이를 구하게 되면 실제 기준점의 데이타는 8바이트를 차지하게 되지만 그 이후에 나타나는 차이의 값은 점차적으로 정수측의 값이 작아지게 된다.

예를 들어 {207068.9921, 451486.2931}, {207061.114, 451469.1631}, {207048.305, 451477.8571}, {207057.3085, 451495.4036}, {207068.9921, 451486.2931}과 같은 5개의 점으로 이루어진 Polygon이 있을 경우 맨 처음 값인 207068.9921과 이어지는 다음 X 좌표 값들의 차는 7.878100, 20.687100, 11.683600, 0.000000과 같은 값을 가지게 된다. 이러한 경우 앞의 정수 측의 값은 비트 연산을 이용하면 충분히 줄일 수 있게 되는데 공간 모바일 장치에 전송하기 전에 가감산 연산을 수행하고 데이타를 전송한 후 공간 모바일 장치에서 이 데이타를 이용하여 double 형태의 데이타를 만들어 내게 되면 무선 인터넷을 이용하여 데이타를 전송할 경우 그 크기를 줄일 수 있게 된다.

예를 들어, 위의 좌표 값의 차이인 7.878100을 다음과 같이 표현할 수 있다. 맨 앞의 플래그는 양수인지 음수인지를 표시하고, 각 길이에 대한 플래그는 실제 값이 차지하는 비트의 크기를 표시한다. 즉, 1은 실제 값이 4비트, 2는 8비트, 3은 12비트, 4는 16비트, 5는 20비트의 크기를 가지고 있다는 의미이다. 각 비트 단위마다 값은 15, 255, 4095, 65535, 1048575의 크기까지 표현할 수 있다. 7.878100에서의 정수부인 7은 양수이기 때문에 signed를 표시하는 0, 정수부 길이의 플래그 001, 그리고 7을 4비트로 표현하면 0111, 즉 00010111을 값을 가지게 된다. 그리고, 소수아래 부분은 878100이므로 소수부 길이 플래그는 0101, 실제 값은 11010110011000010100을 가지게 된다. 이와 같은 과정을 거쳐서 7.878100을 표현하기 위한 8바이트를 00010111 01011101 01100110 00010100의 4바이트로 표시할 수 있다. 즉, double 형으로 하나의 점을 표시할 때 16바이트가 소모되는 것을 최소 4바이트, 최대 10바이트로 표현할 수 있게 된다.

본 논문에서는 압축률을 더욱 높이기 위해 산술 연산 코딩 기법을 개선하였다. 이전 산술 연산 코딩 기법은 각 객체의 첫 번째 좌표점들은 크기를 줄일 수 없는 단점을 가지고 있다. 그래서 개선된 산술 연산 코딩 기법에서는 두 번째 객체들부터 각 객체의 첫 번째 좌표점을 이전 객체의 첫 번째 좌표점과의 거리차를 구함으로써 각 객체들의 첫 번째 좌표점들도 정수측 값을 줄여 압축 성능을 더욱 향상시켰다. 이러한 방법은 객체들이 밀집되어 있을 경우 더욱 높은 압축률을 가질 수 있게 된다. <그림 3>은 개선된 산술 연산 코딩 기법을 이용한 공간 데이타 압축 예를 보여준다.



<그림 3> 공간 데이타 압축 예

첫 번째 Polygon의 X좌표 207068.9921과 두 번째 Polygon의 X좌표 207081.223의 차이 -12.2309를 본 논문에서 제시한 압축 구조로 표현하게 되면 기존의 산술 연산 코딩 기법에서 각 Polygon의 첫 번째 좌표들의 값은 줄일 수 없었던 단점을 해결하여 더욱 압축 성능을 높일 수가 있게 된다.

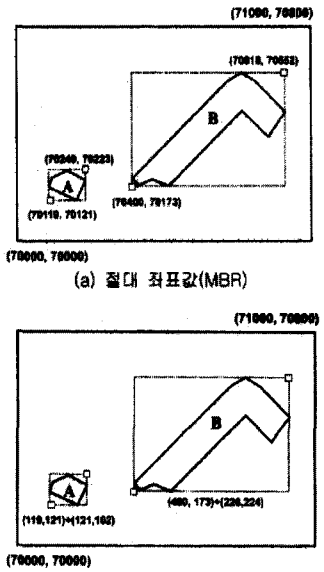
3.4 공간 인덱스

R*-tree는 공간 효율성 측면에서 보면 균형 트리를 만들기 위해 노드를 분할하고 병합하는 과정에서 디스크의 사용률이 낮아진다[13]. 이러한 R*-tree의 특징 때문에 제한된 용량을 가진 메모리 기반 시스템에서는 적절하지 않다. 또한, 메인 메모리 기반 공간 인덱스인 CR-tree는 MBR 압축 기법을 사용하고 있다[14]. 그러나, 정량화 방법을 사용한 MBR 압축 기법은 공간 효율성은 좋지만 여과 효율의 저하로 공간 모바일 장치에서는 적절하지 않다. 그러므로, 본 논문에서는 공간 데이타의 변경보다는 단순 디스플레이 및 점, 영역 질의 위주의 공간 데이타 검색에 효율적인 MBR 압축 및 해쉬 기법을 이용한 공간 모바일 장치용 공간 인덱스[10]를 사용하였다. 공간 모바일 장치용 공간 인덱스에 사용된 해쉬 함수는 다음과 같다.

$$Hx(x) = int[(x - Xmin)/(Xmax - Xmin) * Nx]$$

$$Hy(y) = int[(y - Ymin)/(Ymax - Ymin) * Ny]$$

여기서 Xmax, Xmin, Ymax, Ymin는 전체 MBR을 의미하고 Nx, Ny는 각축의 버켓 개수를 의미한다. 오버플로우 발생시에는 min, max 값을 오버플로우가 발생한 버켓의 MBR로 대치하여 2차 해쉬을 수행하여 하나의 버켓에 들어가는 객체의 개수를 일정하게 유지하도록 하였다. 본 논문에서 MBR의 압축기법은 RMBR(Relative MBR)과 QMBR(Quantization MBR)을 혼합한 HMBR(Hybrid MBR) 기법을 사용하였다. HMBR 표현법은 좌하점은 상대 좌표계처럼 표현하되 우상점은 크기로 표현하는 표현법이다. 이 방법은 MBR의 정확도를 적절히 유지하면서 데이타의 양을 줄일 수 있다. 실제 서너가지 다른 공간 인덱스와 성능 평가를 실시해 본 결과 HMBR 기법이 상대적으로 우수함을 알 수 있었다. <그림 4>는 HMBR 압축 예를 보여준다.



<그림 4> HMBR 압축 예

<그림 4> (a)에서 객체 A의 MBR 크기는 X축으로 121이고 Y축으로 102이다. 객체 B는 X축으로 418, Y축으로 379이다. 그러므로 객체 A의 HMBR은 <그림 4> (b)에서와 같이 (119,121,121,102)로 표현된다. 반면, 객체 B의 크기값은 1바이트의 범위를 벗어나기 때문에 정량화해서 표현하게 되면 <그림 4> (b)에서와 같이 (400,173,226,224)로 표현된다.

4. 내장형 공간 MMDBMS의 구현

본 장에서는 공간 모바일 장치용 내장형 공간 MMDBMS 각각의 모듈에 대한 상세한 구현과 성능 테스트에 대해 기술하고, 주요 결과 화면을 보여준다.

4.1 수입/수출 관리자와 데이타 캐쉬 관리자

수입/수출 관리자는 무선 네크워크에서 소켓을 이용하여 내장형 공간 MMDBMS와 GIS 서버간에 데이타 요청시 데이타 교환 기능을 제공하며, 선택 사항으로 RSA(Rivest-Shamir-Adleman) 암호화 전송 기능을 제공한다.

수입/수출 과정은 다음과 같이 각각 네 단계로 수행된다. 수입 과정의 첫 번째 단계는 소켓을 이용하여 내장형 공간 MMDBMS와 GIS 서버를 연결하고, 두 번째 단계는 GIS 서버의 질의 형식을 이용하여 공간 데이타를 추출한다. 세 번째 단계는 추출된 데이타를 산술 연산 코딩 기법으로 압축하여 내장형 공간 MMDBMS에 전송하고, 네 번째 단계는 전송된 공간 데이타를 내장형 공간 MMDBMS에 입력한다. 그리고 수출 과정의 첫 번째 단계는 소켓을 이용하여 내장형 공간 MMDBMS와 GIS 서버를 연결하고, 두 번째 단계는 내장형 공간 MMDBMS의 질의 처리 관리자를 이용하여 공간 데이타 추출한다. 세 번째 단계는 추출된 데이타를 산술 연산 코딩 기법으로 압축하여 GIS 서버에 전송하고, 네 번째 단계는 전송된 공간 데이타를 GIS 서버에 입력한다.

데이타 캐쉬 관리자는 두 가지 기능을 제공한다. 첫째, 수입/수출시 사용자가 질의한 영역을 먼저 전송한 후 질의 영역보다 30% 큰 영역을 사용자의 다음 질의를 위해 미리 캐쉬하여 전송받는 기능을 제공한다. 둘째, 사용자에게 디스플레이 되는 영역보다 30% 큰 메모리 영역에 공간 및 비공간 데이타를 먼저 디스플레이한 후 사용자에게 필요한 영역만 공간 모바일 장치에 디스플레이 하는 기능을 제공한다.

4.2 데이타 압축 관리자

데이타 압축 관리자는 내장형 공간 DBMS와 GIS 서버 간에 공간 데이타 수입/수출시 네트워크 효율성을 높이기 위해 산술 연산 코딩 기법을 이용한 공간 데이타 압축 기능을 제공한다. 압축 관리자에 사용된 산술 연산 코딩 기법에 대한 성능 테스트는 공간 모바일 장치에서 zip 압축과 압축률, 압축 시간을 비교하였고 성능 테스트에 사용된 데이타는 sequoia 2000 벤치마크[15] 데이타와 강동구 건물 데이타를 사용하였다. Sequoia 2000 벤치마크 데이타는 균등 분포 데이타이며, 객체 수는 79,607개이고, 각 객체는 최소 3개 최대 38,400의 포인터수를 가진다. 그리고, 데이타 크기는 45,060KB이다. 강동구 건물 데이타 객체 수는 12,873개이고 각 객체는 최소 8개 최대 8,174개의 포인터 수를 가진다. 그리고, 데이타 크기는 3,909KB이다.

<표 9>와 <표 10>은 산술 연산 코딩 기법의 압축률과 압축 시간에 대한 성능 테스트 결과를 보여준다.

<표 9> 산술 연산 코딩 기법 압축률

데이타 종류 / 압축방법	sequoia 2000 벤치마크 데이타	강동구 건물 데이타
zip 압축	15135 KB	1327 KB
산술 연산 코딩 기법의 개선 방법	11352 KB	996 KB

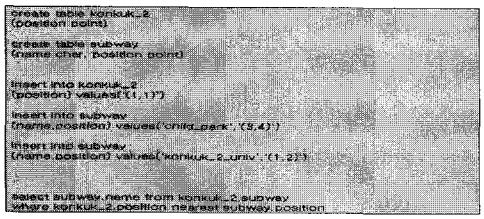
<표 10> 산술 연산 코딩 기법 압축 시간

데이타 종류 / 압축방법	sequoia 2000 벤치마크 데이타	강동구 건물 데이타
zip 압축	19578 ms	2610 ms
산술 연산 코딩 기법의 개선 방법	13060 ms	1739 ms

<표 9>과 <표 10>에서 알 수 있듯이 산술 연산 코딩 기법이 zip 압축에 비해 압축률은 25%, 압축 시간은 30% 향상된 결과를 보여주었다.

4.3 질의 처리 관리자와 트랜잭션 관리자

질의 처리 관리자는 사용자로부터 입력된 SQL 질의문을 인터페이스 관리자로부터 넘겨 받아 SQL문을 검사, 분석 및 처리하는 기능을 제공하며, 내장형 공간 MMDBMS에서 제공하는 공간 데이타 타입과 공간 연산자를 이용하여 공간 데이타에 대한 검색, 삽입, 삭제, 갱신 연산을 수행한다. <그림 5>는 간단한 테이블을 생성하여 공간 데이타를 입력한 후 공간 연산자 nearest를 이용한 검색 예이다.



<그림 5> 공간 연산자 nearest 검색 예

<그림 5는> point형 position 컬럼을 진 konkuk_2와 char형 name와 point형 position 컬럼을 가진 subway 테이블을 생성하고, 각 테이블에 간단한 공간 데이타를 입력한다. 그리고, 공간 연산자 nearest를 이용하여 konkuk_2의 위치에서 가장 가까운 subway 위치를 검색하는 SQL 예이다.

트랜잭션 관리자는 COMMIT과 ROLLBACK 처리를 지원하여 데이타 무결성을 보장한다. 즉, 마지막 시스템 접속 시간, 시스템 버전 정보, 시스템의 정상적인 종료 여부를 기록한 log.properties 화일과 COMMIT된 SQL 구문을 기록한 log.script 화일을 제공하여 데이타 무결성 기능을 지원한다. 또한, 시스템이 비정상적으로 종료되었을 경우 복구 기능도 지원한다.

4.4 디스플레이 관리자

디스플레이 관리자는 데이타 캐쉬 관리자로부터 넘겨 받은 공간 데이타를 공간 데이타 파싱 모듈에 의해 파싱한 후 공간 모바일 장치 좌표로 변환하여 화면에 디스플레이 하는 기능을 제공한다. 공간 모바일 장치의 디스플레이 화면에 출력하기 위해서는 공간 모바일 장치의 디스플레이 영역에 맞도록 좌표를 변환하여야 한다. 즉, GIS 서버 공간 좌표는 실제 크기의 좌표로 이루어져 있으므로 공간 모바일 장치 화면상에 디스플레이 하기에는 일반적으로 좌표 영역이 넓다. 그러므로, 좌표 변환 모듈을 사용하여 실제 크기의 좌표를 공간 모바일 장치 화면상에 디스플레이 가능한 수치로 변환한다. <그림 6>은 본 논문에서 사용한 좌표 변환 공식을 나타내고 있다.

X 좌표 변환 식

x_ratio = (디스플레이 될 윈도우의 폭) / (x_최대좌표 - x_최소좌표)

화면상에 디스플레이 될 x_좌표 = (x_실제좌표 - x_최소좌표) * x_ratio

Y 좌표 변환 식

y_ratio = (디스플레이 될 윈도우의 높이) / (y_최대좌표 - y_최소좌표)

화면상에 디스플레이 될 y_좌표 = (디스플레이 될 윈도우의 높이) -

[(y_실제좌표 - y_최소좌표) * y_ratio]

<그림 6> 좌표 변환 공식

<그림 6>에서 보듯이 좌표 변환에 사용되는 좌표 변환 공식을 간단하게 설명하면 실 좌표당 디스플레이될 수 있는 화면의 x_ratio와 y_ratio를 설정하고, x_ratio와 y_ratio에 각각의 실 좌표를 곱하여 디스플레이 화면에 반영한다. 그러나, 공간 데이타의 Y 좌표의 시작값과 화면 Y 좌표의 시작값은 정 반대로 되어 있기 때문에 화면에 맞는 좌표로 출력하기 위해서는 Y 좌표의 시작 위치를 바꿔야 한다. 그러므로 <그림 6>의 Y 좌표 변환식과 같이 화면상에 디스플레이될 y_좌표는 디스플레이될 윈도우의 높이에서 (y_실제좌표-y_최소좌표)*y_ratioy을 뺀 값이 된다.

4.5 인터페이스 관리자

인터페이스 관리자는 사용자로부터 질의 입력 기능과 지도 디스플레이 및 줌 인/줌 아웃, 팬 기능을 제공한다. 줌 인/줌 아웃 모듈은 공간 모바일 장치에서 지도 화면을 확대/축소하는 모듈이다. 줌 인/줌 아웃 모듈은 ZoomIn(), ZoomOut()의 함수로 구성되어 있으며

공간 모바일 장치에서 디스플레이 되는 공간 데이타를 확대/축소하여 디스플레이될 수 있도록 질의를 생성한다. 즉, 일정한 화면 크기에 전의 영역 좌표값보다 축소된 영역 좌표값을 적용하면 디스플레이시 확대되고, 일정한 화면 크기에 전의 영역 좌표값보다 확대된 영역 좌표값을 적용하면 디스플레이 시 축소된다.

<그림 7>은 본 논문에서 사용한 확대/축소 공식을 보여준다.

확대 수행 공식

newXmax = oldXmax + (oldXmax - oldXmin) * 0.1

newYmax = oldYmax + (oldYmax - oldYmin) * 0.1

newXmin = oldXmin - (oldXmax - oldXmin) * 0.1

newYmin = oldYmin - (oldYmax - oldYmin) * 0.1

축소 수행공식

newXmax = oldXmax - (oldXmax - oldXmin) * 0.1

newYmax = oldYmax - (oldYmax - oldYmin) * 0.1

newXmin = oldXmin + (oldXmax - oldXmin) * 0.1

newYmin = oldYmin + (oldYmax - oldYmin) * 0.1

<그림 7> 확대/축소 공식

팬 모듈은 클라이언트에 디스플레이된 지도 화면을 동, 서, 남, 북으로 영역 이동하는 기능을 제공한다. 팬 모듈은 East(), West(), South() North() 함수로 구성되어 있으며, 클라이언트에서 디스플레이되는 공간 데이타를 동, 서, 남, 북으로 이동하여 디스플레이할 수 있도록 질의를 생성한다. <그림 8>은 본 논문에서 사용한 이동 공식을 보여준다.

동 : (newXmin + dX , newYmin) , (newXmax + dX , newYmax)

서 : (newXmin - dX , newYmin) , (newXmax - dX , newYmax)

남 : (newXmin , newYmin + dY) , (newXmax , newXmax + dY)

북 : (newXmin , newYmin - dY) , (newXmax , newYmax - dY)

<그림 8> 이동 공식

4.6 공간 인덱스 관리자

공간 인덱스 관리자는 직접 탐색 방법인 해쉬을 이용하여 검색 효율성을 높이고 MBR를 압축하여 공간 효율성을 높인 MBR 압축 및 해쉬 기법을 이용한 공간 모바일 장치용 공간 인덱스를 제공한다. 공간 인덱스 관리자의 성능 테스트는 공간 모바일 장치에서 동적 해쉬과 인덱스의 크기, 공간 데이타 삽입과 검색에 대한 성능을 비교하였고 성능 테스트에 사용된 데이타로는 sequoia 2000 벤치마크 데이타와 강동구 건물 데이타를 사용하였다. <표 11>은 내장형 공간 MMDBMS에 적용된 공간 인덱스와 동적 해쉬간의 인덱스 크기를 비교한 그림이다.

<표 11> 공간 인덱스 크기

인덱스 구조 \ 데이타 종류	sequoia 2000 벤치마크 데이타	강동구 건물 데이타
공간 인덱스	783 KB	96 KB
동적 해쉬	624 KB	78 KB

<표 11>에서 보여주는 바와 같이 인덱스의 크기는 내장형 공간 MMDBMS에 사용된 공간 인덱스의 크기가 동적 해쉬보다 더 크다는 것을 알 수 있다. 공간 인덱스와 동적 해쉬간의 삽입과 검색 시간을 테스트한 결과 <표 12>와 <표 13>에서와 같이 공간 데이타 삽입과 검색 시간에서 공간 인덱스가 동적 해쉬에 비해 약 25-30% 성능 향상을 보였다.

<표 12> 공간 인덱스 삽입 시간

인덱스 구조 \ 데이타 종류	sequoia 2000 벤치마크 데이타	강동구 건물 데이타
공간 인덱스	2136.5 ms	305.4 ms
동적 해쉬	2847.3 ms	406.5 ms

<표 13> 공간 인덱스 검색 시간

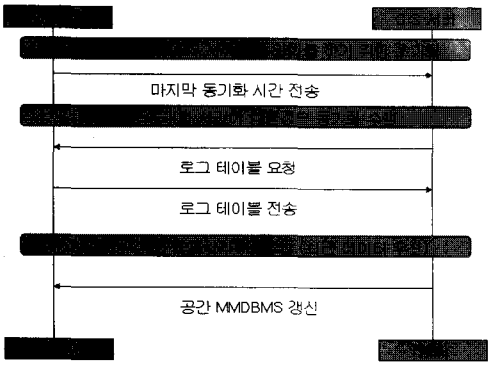
인덱스 구조 \ 질의영역		PointQuery	영역 질의		
			0.1%	0.4%	1%
강동구 건물 데이타	공간 인덱스	6.6 ms	64.5 ms	101.8 ms	147.5 ms
	동적 해쉬	8.9 ms	87.0 ms	148.5 ms	210.7 ms
sequoia 2000 벤치마크 데이타	공간 인덱스	46.8 ms	451.8 ms	782.9 ms	1103.0 ms
	동적 해쉬	63.1 ms	607.9 ms	1046.7 ms	1487.9 ms

4.7 동기화 관리자

동기화 관리자는 내장형 공간 MMDBMS와 GIS 서버 사이간에 데이타를 일치시켜 주는 기능을 제공한다. 동기화 방법으로는 point-update 동기화 방법을 사용하였는데, 이것은 양방향 동기화 기능을 제공한다. point-update 동기화 방법은 마지막 동기화 이후에 변화된 부분만 갱신하는 방법이며, 동기화중에는 내장형 공간 MMDBMS와 GIS 서버에 트랜잭션 LOCK이 적용된다. 그리고, 동기화중 충돌 해결 방법으로는 최근 데이타 우선 방법을 사용하였다.

동기화를 하기 위해서는 로그 테이블 정보가 필요한데, 로그 테이블 정보에는 변경된 테이블에 대한 테이블명, PK 컬럼명, PK 값, 데이타 타입, 변경 후 값, DML 타입, time 등 변경된 데이타에 대한 정보가 기록된다. 여기서 DML 타입에는 insert, update, delete가 있으며, time은 트랜잭션 수행 시간을 의미한다.

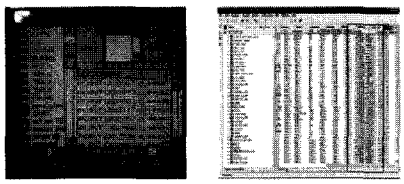
<그림 9>는 동기화 프로토콜을 보여준다.



<그림 9> 동기화 프로토콜

4.8 결과 화면

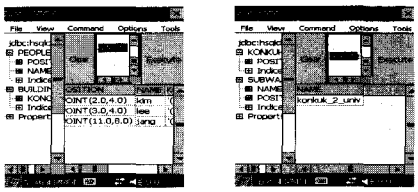
본 논문에서는 시스템 구현시 개발 언어로 Java를 사용하였고, 컴파일 툴로는 ANT를 사용하였다. GIS 서버는 ZEUS를 사용하였고 공간 모바일 장치는 Compaq iPAQ 5450을 사용하였다. 그리고, OS는 Pocket PC 2002와 Embedded Linux 환경에서 Personal Java 1.1과 Embedded Java 1.3을 설치하여 사용하였다. 테스트 데이타는 Sequoia 2000 벤치마크 데이타와 서울시 강동구 건물 데이타를 이용하였다. <그림 10>은 내장형 공간 MMDBMS와 ZEUS간의 수입/수출 실행 화면으로 내장형 공간 MMDBMS에서 무선 통신을 이용하여 ZEUS에서 강동구 건물 데이타를 수입/수출한 결과 화면을 보여준다.



내장형 공간 MMDBMS　　ZEUS

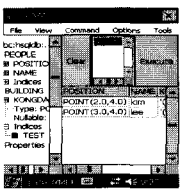
<그림 10> 수입/수출 실행

<그림 11>은 공간 연산자중 contain 연산자를 이용하여 특정 영역에 포함되어 있는 사람들을 검색한 결과 화면과 nearest 연산자를 이용하여 가장 가까운 주변 역을 검색한 결과 화면을 보여준다.



<그림 11> 질의 처리 (contain과 nearest)

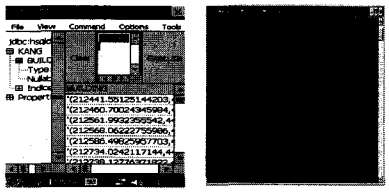
<그림 12>와 <그림 13>은 각각 데이타를 입력하여 공간 인덱스를 생성한 후 데이타 검색 시 공간 인덱스를 이용하여 검색한 화면과 검색된 공간 데이타를 공간 모바일 장치 화면에서 맵으로 디스플레이한 화면을 보여준다.



<그림 12> 공간 인덱스를 이용한 검색

<그림 13> 맵 디스플레이

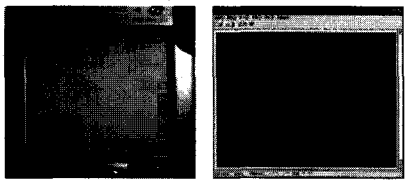
<그림 14>는 내장형 공간 MMDBMS에서 제공하는 JDBC 드라이버를 이용하여 클라이언트에서 내장형 공간 MMDBMS에 접속한 후 공간 데이타를 삽입 및 검색하는 화면을 보여준다.



내장형 공간 MMDBMS　　클라이언트

<그림 14> JDBC 드라이버를 통한 내장형 공간 MMDBMS 접속

<그림 15>는 공간 모바일 장치의 Embedded Linux에 Embedded Java를 설치한 후 Porting한 화면을 보여준다.



내장형 공간 MMDBMS　　하이퍼터미널

<그림 15> Embedded Linux Porting

5. 결론 및 향후 과제

최근에 무선 인터넷과 이동 컴퓨팅 기술이 발전하고, 휴대 전화, PDA와 같은 모바일 장치가 보편화됨에 따라 사용자의 위치 정보를 활용하는 위치 기반 서비스가 다양한 분야에서 제공되고 있다. 모바일 장치는 기존 PC에 비해 작은 용량의 저장 공간과 낮은 성능의 프로세서를 가지고 있으며, 자체 화일 시스템을 기반으로 데이타를 처리하고 있다. 이러한 자체 화일 시스템은 작은 용량의 데이타를 처리하기는 편리할지 모르나 나날이 증가하고 있는 대용량의 공간 데이타를 관리하거나 매우 복잡하고 다양한 공간 질의를 처리하기에는 큰 문제가 된다. 그러므로 위치 기반 서비스에서 대용량의 공간 데이타를 신속하게 처리하고 효과적으로 관리하기 위해서는 공간 모바일 장치의 메모리에서 공간 데이타를 효율적으로 처리할 수 있는 내장형 공간 MMDBMS가 필요하다.

본 논문에서는 메모리 기반 관계형 DBMS인 HSQLDB에 위치 기반 서비스에서 필요한 공간 데이타를 지원하기 위해 공간 데이타 타입과 공간 연산자를 추가하였고, 내장형 공간 MMDBMS와 GIS 서버간에 공간 데이타 수입/수출시 공간 데이타의 양을 줄일 수 있도록 공간 데이타 특성에 적합한 산술 연산 코딩 압축 기법을 추가하여 효율적으로 공간 데이타를 수입/수출할 수 있도록 하였다. 또한, 공간 데이타의 빠른 검색을 위해 공간 모바일 장치에 적합한 MBR 압축 및 해싱 기법을 이용한 공간 인덱스를 적용하였고, 공간 모바일 장치에서 디스플레이 기능과 GIS 서버와의 통신 성능을 높이기 위한 데이타 캐쉬 기능을 추가하여 공간 모바일 장치에서 공간 데이타를 효과적으로 관리할 수 있는 내장형 공간 MMDBMS를 설계 및 구현하였다.

본 논문에서 개발한 내장형 공간 MMDBMS는 지능형 교통 시스템이나 M-CRM(Mobile CRM) 등 다양한 위치 기반 서비스에서 공간 및 위치 데이타를 효율적으로 관리할 수 있기 때문에 공간 모바일 장치에서 위치 기반 서비스의 성능 향상에 큰 도움을 줄 수 있다. 앞으로 향후 연구 과제로는 여러 다른 GIS와의 표준화된 동기화 방법에 관한 연구가 필요하겠다.

참고문헌

[1] Lee, K.Y., Kim, D.O., Yun, J.K., and Han, K.J., "A Real-time Mobile GIS based on the HBR-tree," Proc. of the 33rd Int. Conf. on Computers & Industrial Engineering, 2004.

[2] Yun, J.K., Kim, D.O., and Han, K.J., "Development of a Real-Time Mobile GIS supporting the Open Location Service," Proc. of Geotec Event Conference, Canada, 2003.

[3] 윤재관, 장염승, 한기준, "모바일 GIS를 위한 위치 기반 서비스," 한국정보과학회 데이타베이스 연구,

18권1호, 2002, pp.3-15.

[4] 양영규, "위치기반 서비스(LBS: Location Based Service)기술 현황 및 전망," 한국정보처리학회 정보처리학회지, 8권6호, 2001, pp.4-5.

[5] Greene, D., "An Implementation and Performance Analysis of Spatial Data Access Methods," IEEE Transaction on Knowledge and Data Engineering, Vol.5, No.1, 1989, pp.606-615.

[6] Gueting, R.H., "An Introduction to Spatial Database Systems," The VLDB Journal, Vol.6, No.1, 1994, pp.357-399.

[7] HSQLDB, http://hsqldb.sourceforge.net.

[8] 이상윤, 박순영, 이미영, 김명준 "이동 DBMS의 데이타 동기화 기술 분석," 한국정보과학회 데이타베이스 연구, 17권3호, 2001, pp.29-31..

[9] 최우영, 이경아, 염태진, 진성일 "내장형 DBMS를 위한 동기화 서버 시스템," 한국정보과학회 논문지, 31권1호, 2004, pp.429-437.

[10] 김진덕, "최소경계사각형 압축 및 해쉉 기법을 이용한 PDA용 공간색인," 한국정보과학회 데이타베이스 연구, 18권4호, 2002, pp.11-21.

[11] ISO TC/211, 19107 Geographic Information - Spatial Schema, http://www.isotc211.org.

[12] 조형주, 정진완, "다차원 색인 구조를 위한 효율적인 압축 방법," 한국정보과학회 논문지, 30권5호, 2003, pp.429-437.

[13] Beckmann, N., Kriegel, H.P., Schneider, R., and Seeger, B., "R*-tree : An Efficient and Robust Access Method for Points and Rectangles," Proc. of Int. Conf. on ACM SIGMOD, 1990, pp.322-331.

[14] Kim, K.H. Cha, S.K., and Kwon, K.J., "Optimizing Multidimensional Index Trees for Main Memory Access," Proc. of Int. Conf. on ACM SIGMOD, 2001, pp.139-150.

[15] Stonebraker, M., Frew, J., Gardels, K., and Meredith, J., "The SEQUOIA 2000 Strorage Benchmark," Proc. of Int. Conf. on ACM SIGMOD, 1993, pp.2-11.

박지웅

1992년 배재대학교 전자계산학과 (이학사)

1994년 건국대학교 컴퓨터공학과 (공학석사)

2004년 ~ 현재 건국대학교 컴퓨터공학과 박사과정

2001년 ~ 현재 경기공업대학 컴퓨터정보시스템과 조교수

관심분야 : GIS, LBS, gCRM/mCRM, 데이타마이닝

김정준

2003년 건국대학교 컴퓨터공학과 졸업 (공학사)

2004년 건국대학교 대학원 컴퓨터공학과 졸업예정(공학석사)

2005년 ~ 현재 건국대학교 대학원 컴퓨터공학과 박사과정

관심분야 : GIS, LBS, MMDBMS, 위치 데이타 인덱스

윤재관

1997년 건국대학교 컴퓨터공학과 졸업 (공학사)

1999년 건국대학교 대학원 컴퓨터공학과 졸업(공학석사)

2003년 건국대학교 대학원 컴퓨터공학과 졸업 (공학박사)

2003년 ~ 현재 건국대학교 컴퓨터공학부 강의교수

관심분야 : 실시간 데이타베이스, LBS, 위치 데이타 인덱스, 분산 위치 저장 시스템

한기준

1979년 서울대학교 수학교육학과 졸업 (이학사)

1981년 한국과학기술원 전산학과 졸업 (공학석사)

1985년 한국과학기술원 전산학과 졸업(공학박사)

1990년 Stanford 대학 전산학과 visiting scholar

1985년 ~ 현재 건국대학교 컴퓨터공학과 교수

2004년 ~ 현재 한국공간정보시스템학회 회장

2004년 ~ 현재 한국정보시스템감리사협회 회장

관심분야 : 공간데이타베이스, GIS, LBS, 텔레매틱스, 정보시스템 감리