Special Edition

# 텔레매틱스 데이터 관리

## Database Management Systems for Telematics

**이 현 익** / Hyonik Lee
한국과학기술원
Korean Advanced Institute of Science and Technology

**이 진 원** / Jinwon Lee
한국과학기술원
Korean Advanced Institute of Science and Technology

**송 준 화** / Junehwa Song
한국과학기술원
Korean Advanced Institute of Science and Technology

### 개요

이동 통신 기술의 발달과 더불어, 점점 더 많은 사람들이 자동차 안에서 각종 정보 서비스[1]를 제공받기 위해 텔레매틱스를 이용하고 있다. 텔레매틱스 환경에서의 다양한 서비스를 위해서는 데이터들이 효과적으로 저장, 전달되어야만 한다. 또한, 이들 데이터에 대한 질의들도 빠르고 효율적으로 처리되어야 한다.

본 장에서는 텔레매틱스 특집의 일환으로 텔레매틱스 환경에서의 데이터 관리에 적합한 세종류의 데이터 관리 시스템에 대해 살펴볼 것이다. 우선, 단말기 측의 데이터관리 시스템인 모바일 DBMS에 대해서 알아본다. 모바일 DBMS는 모바일 단말기내의 데이터를 관리해줌으로써 이동 중에도 쉽게 데이터를 사용할 수 있도록 해주는 기능을 한다. 다음으로, 서버 측의 이동객체 데이터관리 시스템인 이동객체 DBMS에 대해서 알아본다. 이동객체 DBMS는 주로 차량이나 모바일 사용자들의 위치 정보를 관리해 준다. 마지막으로, 서버 측의 스트림 데이터관리 시스템인 스트림 DBMS에 대해서 알아본다. 이는 센서 데이터처럼 지속적으로 생성되어 입력되는 스트림 데이터를 처리하는데 사용된다.

1) 교통정보, 원격차량 진단, 모바일 전자상거래 등

### 모바일 DBMS (Mobile DBMS)

#### 모바일 DBMS란?

모바일 DB는 모바일 기기에 사용되는 데이터베이스로써 다른 말로는 내장형 데이터베이스(Embedded Database), 초소형 데이터베이스(Ultra Tiny Database) 또는 자바 데이터베이스(Java Database)라고도 불린다. 최근 분석에 따르면 2006년까지 전세계적으로 10억개의 모바일 기기들이 존재할 것으로 예상되고 있다. 물론 어떤 응용 프로그램들이 주로 사용되게 될지는 누구도 말하기 힘들지만 이들 중의 상당수가 어떤 종류던 간에 모바일 데이터베이스를 사용할 것이라는 것은 분명하다.

#### 기존 DBMS와의 차이점

모바일 DBMS를 기존의 DBMS와 비교해 보면 다음과 같은 특징을 발견할 수 있다. 우선, 굉장히 가벼

위야 한다. 물론, 모바일 기기가 노트북처럼 대용량의 메모리와 컴퓨팅 능력을 가질 수도 있겠지만, PDA나 핸드폰 같은 대부분의 모바일 기기들은 그렇지 못하기 때문이다. 따라서 모바일 DBMS는 작은 크기로도 만족할 만한 성능을 낼 수 있어야 한다. 둘째로 모바일 DBMS는 기존 데이터베이스 서버와의 동기화가 중요하다. 일반적으로 기업에는 기존에 사용되던 데이터베이스 서버가 존재한다. 모바일 기기가 항상 데이터베이스 서버와 연결된 상태에서 어떤 정보를 열람한다는 것은 실질적으로 불가능 하기 때문에 필요한 만큼의 데이터는 모바일 기기 내에 복제되어 있다. 이 데이터가 의미 있게 사용되기 위해서는 데이터베이스 서버와 모바일 데이터베이스의 데이터 동기화를 통해서 자료를 늘 갱신하고 일치시킬 수 있어야만 한다. 하나의 특징을 더 들자면, 모바일 DBMS는 사용상의 편의를 위해 간단하고 빠르게 다운로드 받아 운영할 수 있어야 한다.

### 모바일 DBMS의 예

최근 들어 오라클, IBM, 마이크로소프트, 사이베이스 등 대부분의 데이터베이스 전문 기업들은 모바일 솔루션의 일환으로 저마다의 모바일 데이터베이스를 앞다투어 내놓고 있다. 오라클의 Lite 10g이나 IBM의 DB2 에브리플레이스 등을 예로 들 수 있을 것이다. 특히 요즘에는 Windows CE나 PalmOS 뿐 아니라 Linux에서도 설치 구동 될 수 있도록 지원되고 있다.

## 이동 객체 DBMS(Moving Object DBMS)

### 이동 객체 DBMS란?

이동 객체 DBMS은 이동체의 위치정보를 관리하며 위치 기반의 질의를 처리 해주는 시스템이다. 차량 관제나 물류/택배, 사람 찾기 등의 사례에서 볼 수 있듯이 이동 객체 DBMS는 주로 이동통신망을 기반으로 사람이나 사물의 위치를 정확하게 파악하고 이를 활용하는 위치기반 서비스(LBS : Location-Based Service)를 위해서 사용된다.

### 특징 그리고 기존 DBMS의 문제점

LBS 환경에서의 데이터베이스는 대용량의 데이터를 다룬다는 특성이 있다. 또한, 데이터 및 색인이 실시간으로 갱신되고 시공간 질의, 궤적질의 등의 새로운 형태의 질의도 가능하다는 특성도 있다.

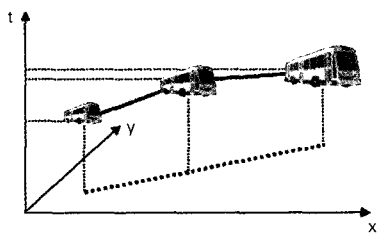
LBS 환경에서의 이런 특징들은 이동 객체를 기존의 DBMS로 관리할 때 다음과 같은 문제점들을 유발시킨다. 첫째, 이동체가 연속적으로 위치 정보를 변경하기 때문에 시스템 성능에 악영향을 끼치게 된다. 기존의 DBMS에서는 객체들에 색인을 두어 빠르게 질의를 처리할 수 있었다. 비록 객체정보를 변경하면 색인도 같이 변경시켜줘야 하지만 기존의 DBMS에서는 객체정보가 거의 변하지 않았기 때문에 문제가 별로 없었다. 하지만, 연속적으로 객체들이 위치정보를 변경하는 상황이라면 색인 수정에 병목현상이 생겨 시스템의 성능을 크게 저하시키게 된다. 둘째, 기존의 DBMS에서는 이동체를 모델링 하기가 쉽지 않다. 이동체의 위치는 시간에 따라서 계속 변한다. 이때 과거의 위치와 미래의 위치에 대한 질의를 수행하기 위해서는 시간에 따른 이동체의 궤적을 표현해줘야 한다. 하지만, 기존의 DBMS에서는 객체를 정적으로 모델링 하기 때문에 위와 같은 이동체를 모델링 하기 어렵다. 셋째, 기존의 DBMS에서는 이동체에 대한 질의를 처리하기 어렵다. 기존의 DBMS에는 이동체에 대한 질의연산 모델이 정의되어 있지 않다. 따라서, 궤적 질의나 Time-Slice 질의 등을 지원하지 못한다.

### 이동 객체 DBMS의 요소 기술

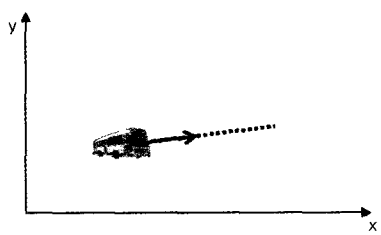
위와 같은 문제점을 극복하기 위하여 이동 객체 DBMS가 갖춰야 할 요소 기술로는 크게 이동체 모델링, 이동체에 대한 연산자 및 질의, 이동체 색인, 이동체 위치정보의 불확실성 보완 이렇게 네 가지로 나눠 볼 수 있다.

#### 이동체 모델링

이동체는 시간이 변화함에 따라서 객체의 위치나 모양이 연속적으로 변하는 시공간 데이터를 말한다. 이때 계속 변하는 이동체의 위치를 모두 저장할 수는 없기 때문에 이동체를 모델링하여 데이터베이스에 저장한다. 이동체 모델링 방식은 크게 Spatio-Temporal 모델링 방식과 MOST(Moving Objects Spatio-Temporal) 모델링 방식으로 나뉜다〈그림 1, 그림 2〉. Spatio-Temporal 모델링 방식은 과거 및 현재 의 이동정보만을 모델링 한다. 반면 MOST는 현재 및 미래의 이동정보만을 모델링 한다.




〈그림 1〉 Spatio-Temporal 모델




〈그림 2〉 MOST 모델

#### 이동체에 대한 연산자 및 질의

이동체에 대한 연산자 및 질의는 기존의 DBMS에서 제공하던 것들과는 다르다. 이동체에 대한 연산자는 다음 표와 같다.

〈표 1〉 이동체에 대한 연산자

| 연산자 | 연산자 의미 |
|---|---|
| Mdistance | 모든 시간에 대해 두개의 이동 점 간의 거리를 구함 |
| Visits | 이동 점이 이동영역의 내부에 있을 때 이동 점의 위치를 구함 |
| Trajectory | 주어진 이동 점의 궤적을 선으로 구함 |
| Length | 주어진 선의 총 길이를 구함 |
| Velocity | 주어진 시간에 이동 점의 위치를 구함 |
| Attime | 어떤 시간 t에 대한 이동 점의 위치를 구함 |
| Minvalue, Maxvalue | 최소, 최대 값을 구함 |
| Mintime, Maxtime | 최소, 최대일 때의 시간을 반환 |

이동체에 대한 질의는 질의 형태와 질의 언어로 나누어 살펴보도록 하자. 질의 형태로는 점(Point) 질의, 영역 (Range) 질의, 근처 이웃 (Nearest Neighbor) 질의, Time-slice 질의, 궤적 (Trajectory) 질의 등이 있다. 질의 언어로는 현재와 미래에 대한 질의 언어인 FTL (Future Temporal Logic)과 과거에 대한 질의 언어인 STQL (Spatio-Temporal Query Language)가 있다.

#### 이동체 색인

이동체에 대한 질의는 대부분 위치 정보에 대해서 이루어진다. 따라서 질의를 빠르게 처리하기 위해서는 이동체의 위치 정보에 대해 색인을 두어야 한다. 이러한 색인은 빈번한 위치 정보 갱신에도 성능이 떨어지면 안 되며 과거/현재/미래의 위치 검색도 지원해 주

어야 한다. 이러한 특성을 만족하는 색인으로는 현재/미래의 위치정보를 위한 TPR-tree와 과거의 위치정보를 위한 3DR-tree, STR-tree, TB-tree 등이 있다.

#### 이동체 위치정보의 불확실성 보완

모델링되어 저장된 이동체의 위치 정보는 불확실할 수 있다. 불확실한 위치정보는 측정 장비 자체에 오류가 발생하거나 위치정보의 갱신주기가 길어서 발생한다. 측정 장비자체에서 발생하는 위치정보 오류는 예외로 처리하면 되지만 위치정보 갱신주기 때문에 생기는 불확실성은 데이터베이스의 성능과 밀접한 관련이 있다. 데이터베이스에 정확한 위치 정보를 자주 갱신하면 위치정보의 불확실성은 떨어지나, 갱신이 늘어남에 따라 데이터베이스의 성능은 저하된다. 이러한 이동체 위치정보의 불확실성은 이동체모델과 질의를 확장함으로써 보완될 수 있다. 데이터베이스에는 위치정보의 불확실성에 관련된 부속속성을 두고 사용자는 질의에 (May, Must) 등과 같이 정보의 정확성에 대한 명시적인 요구조건을 기술한다. 이를 통해 위치정보의 불확실한 정도에 따라 질의를 처리한다.

## 스트림 DBMS(Stream DBMS)

### 스트림 DBMS란?

스트림 DBMS란 쉽게 말해서 스트림 데이터를 관리하고 이에 관한 질의를 처리해 주는 시스템이다. 여기서 스트림 데이터란, 지속적으로 생성되어 주기적으로 보고되는 데이터를 말한다. 센서에 의해 주기적으로 감지된 자동차의 위치나 속도, 엔진의 온도, 타이어의 상태 정보 등을 예로 들 수 있을 것이다. 스트림 DBMS는 이렇게 실시간으로 끊임없이 입력되는 데이터에 대한 질의를 효과적으로 지원해 준다.

### 특징 그리고 기존 DBMS의 문제점

이런 환경은 다음과 같은 특징이 있다. 우선 데이터들이 네트워크를 통해 실시간으로 계속 입력된다는 점이다. 이는 프로세싱이나 스토리지 면에서 시스템에 상당한 부담이 된다. 두번째로 데이터의 형태가(예를 들면, 온도, 습도, 인터넷의 XML 문서 등) 매우 다양하다는 점을 들 수 있다. 또한 기존의 DBMS에서는 거의 요구되지 않았던 영속 질의(Continuous Query)가 많이 요구된다. 영속 질의란 일정 시간 동안 데이터베이스 시스템에 존재하면서 질의 조건에 맞는 새로운 데이터가 들어왔을 때마다 처리되는 질의를 말한다. 예를 들면 다음과 같은 질의를 들 수 있다.

**앞으로 3개월 동안, 대학 구내에서 시속 30km 이상 속도를 내는 차를 알려달라.**

위와 같은 특징들로 인해 기존의 DBMS가 스트림 데이터를 처리하기에는 여러 가지 문제가 있다. 우선 기존의 DBMS는 이렇게 빠른 속도로 끊임없이 유입되는 대용량의 데이터를 처리하도록 설계되어 있지 않다. 기존의 DBMS는 이미 저장되어 있는 데이터를 효율적으로 관리하고, 이에 대한 질의의 처리에 최적화되어 설계되어 있기 때문이다. 또한, 기존의 DBMS는 영속 질의를 처리를 하는데 효율적이지 못하다. 종전에는 데이터베이스에 미리 저장된 데이터에 대한 질의들이 대부분이었기 때문에 바로 처리해서 결과값을 돌려주었다. 하지만 영속 질의는 끊임없이 입력되는 스트림 데이터에 대한 질의이기 때문에 한번에 처리하고 끝낼 수는 없다. 텔레매틱스 환경에서는 이런 영속 질의들이 빈번히 요청되기 때문에 데이터베이스는 많은 수의 영속 처리를 동시에 처리해야 한다.

### 기본적인 연산과 요소기술

스트림 데이터에 대한 영속 질의를 수행하기 위해 가

장 기본적으로 사용될 연산들을 정리하면 다음과 같다.

**〈표 2〉 영속 질의를 위한 연산자**

| 연산자 | 연산자 설명 |
|---|---|
| Selection | 스트림 데이터에 대한 필터링 |
| Multiplexing & Demultiplexing | 물리적으로 하나인 스트림을 여러 개의 논리적 스트림으로 분리하거나 그 반대로 통합 |
| Frequent Item Count | 가장 많이 나타나는 아이템을 찾음 |
| Join | 다수의 스트림 사이의 Join 또는 정적인 메타정보와의 Join |
| Windowed 질의 | 위의 모든 질의 타입은 윈도우 크기에 의해 연산이 제한될 수 있다 |

또한, 위와 같은 연산을 수행하기 위해 스트림 DBMS가 갖춰야 할 요소 기술로는 스트림을 위한 데이터 모델과 질의언어, 스트림 연산자 구현, 스트림 연산자의 수행 구성과 최적화 등이 있다.

### 스트림 DBMS의 예

대표적인 스트림 DBMS로는 코넬(Cornell) 대학의 Cougar Device DBMS, 위스콘신(Wisconsin) 대학의 NiagaraCQ, 버클리(Berkeley) 대학의 TelegraghCQ 그리고 브라운(Brown) 대학, 브랜다이스(Brandeis) 대학, MIT 세학교가 공동 개발한 Aurora를 들 수 있다. 이들은 제시된 스트림 DBMS의 요구사항을 해결하기 위해 각자 고유한 방법을 제안하고 있다. 하지만 이들 각각에 대한 자세한 설명은 지면 관계상 허락이 되지 않기 때문에 생략하도록 하겠다.

(송준화 교수 : Junesong@cs.kaist.ac.kr)

## 참고문헌


1. 송준화 외 공저. 텔레매틱스 개론. 홍릉과학출판사
2. J. Lee, S. Kang, S. Choi, H. Jin, S. Choe, Y. Lee, J. Song, LARI : Locality-Aware Range query Index for High Performance Data Stream Processing, KAIST Technical Report CS-TR-2004-202, August 2004
3. http://www.cs.brown.edu/research/aurora/
4. Stanford Stream Data Management(STREAM) Project. http://www-db.stanford.edu/stream
5. B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and Issues in Data Stream Systems. In Proc. ACM SIGMOD/SIGA CT Conf. on Princ. of Database Syst., pages 1 16, Madison, Wisconsin, USA, June 2002.
6. P. Bonnet, J. Gehrke and P. Seshadri, "Querying the Physical World," IEEE Pers. Commun., vol. 7, Oct. 2000, pp. 10-15.
7. Yong Yao, J. E. Gehrke The Cougar Approach to In-Network Query Processing in Sensor Networks. Sigmod Record, Volume 31, Number 3 September 2002.
8. J. Chen, D. DeWitt, F. Tian, and Y. Wang. NiagaraCQ : A scalable continuous query system for internet databases. In Proc. of the ACM SIGMOD Conf. on Management of Data, 2000.
9. Sailesh Krishnamurthy, Sirish Chandrasekaran, Owen Cooper, Amol Deshpande, Michael J. Franklin, Joseph M. Hellerstein, Wei Hong, Samuel R. Madden, Vijayshankar Raman, Fred Reiss, and Mehul A. Shah. TelegraphCQ : An Architectural Status Report. IEEE Data Engineering Bulletin, Vol 26(1), March 2003
10. Samuel R. Madden, Mehul A. Shah, Joseph M. Hellerstein and Vijayshankar Raman. Continuously Adaptive Continuous Queries over Streams. ACM SIGMOD Conference, Madison, WI, June 2002.
11. Samuel R. Madden and Michael J. Franklin. Fjording the Stream : An Architecture for Queries over Streaming Sensor Data ICDE Conference, February, 2002, San Jose.
12. Joseph M. Hellerstein and Ron Avnur. Eddies : Continuously Adaptive Query Processing. In SIGMOD 2000.
13. D. Carney, U. Cetintemel, M. Cherniack, C. Convey, S. Lee, G. Seidman, M. Stonebraker, N. Tatbul, and S. Zdonik. Monitoring Streams : A New Class of Data Management Applications. In proceedings of the 28th International Conference on Very Large Data Bases (VLDB 02), Hong Kong, China, August 2002
14. D. Abadi, D. Carney, U. Cetintemel, M. Cherniack, C. Convey, S. Lee, M. Stonebraker, N. Tatbul, S. Zdonik. Aurora : A New Model and Architecture for Data Stream Management. In VLDB Journal, August 2003