제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

이동객체를 위한 위치정보 관리 시스템의 설계

조대수*, 민경욱, 이종훈
한국전자통신연구원, 공간정보기술센터
e-mail: {junest, kwmin92, jong}@etri.re.kr

Design of Location Information Management System for Moving Objects

Dae-Soo Cho*, Kyoung-Wook Min, Jong-Hun Lee
Spatial Information Technology Center, ETRI

요 약

최근에 이동객체의 위치정보를 활용한 위치기반서비스(LBS, Location Bases Services)에 대한 관심이 급증하고 있다. 이동객체는 시간에 따라 위치 정보가 계속 변경되는 공간 객체를 의미한다. 전통적으로 고정된 위치정보를 갖는 공간 객체는 GIS 서버에서 저장, 관리되었다. 그러나, 이동객체의 경우, 위치의 변화가 매우 빈번하며, 과거의 위치정보도 관리해야 하므로, 전통적인 GIS 서버로는 관리가 어렵다. 이 논문에서는 위치기반서비스를 위해 위치정보를 저장, 관리하기 위한 위치정보 관리 시스템을 설계한다. 위치정보 관리 시스템은 위치획득 서브시스템, 위치저장 서브시스템, 그리고, 위치질의 서브시스템으로 구성된다. 이 논문의 기여는 이동객체에 대한 위치획득, 저장, 질의 등 각각의 연구 결과를 통합함으로써, 실제 응용에 적용할 수 있는 시스템을 개발하는데 있다.

1. 서론

무선 인터넷이 가능한 이동 단말의 보급과 GPS 로 대표되는 측위 기술의 발달에 따라 다양한 종류의 위치기반 응용서비스에 대한 관심이 급증하고 있다. 위치기반 응용서비스를 제공하기 위해서는 지속적으로 위치가 변하는 이동객체(MO, Moving Objects)를 저장, 처리하기 위한 효과적인 방법이 요구되고 있다. 현재의 상용 데이터베이스 및 GIS 시스템으는 이동객체를 효과적으로 처리하기 어렵다[8]. 왜냐하면, 이동객체를 처리하기 위해서는 이동 객체 데이터 모델, 이동객체 질의어, 이동객체 색인 등의 기능이 추가되어야 하기 때문이다.

기존의 이동객체에 대한 연구는 이동객체 데이터 모델 및 질의어에 대한 연구[1,5,8,9]와 이동객체 색인에 대한 연구[3,4,6,7,12]로 크게 구분되고 있다. 또한, 이동 객체의 현재 위치를 기반으로 미래 위치를 다루는 연구[1,4,8,12]와 과거 위치를 다루는 연구[5,9,3,6,7]로 구분될 수 있다. 그밖의 연구로는 다양한 종류의 이동객체에 대해서 실제 위치정보를 얻기 어려운 문제를 해결하기 위해서, 가상의 위치 데이터 셋 생성에 대한 연구[2,11,13]등이 있다.

이 논문은 이동객체에 대한 다양한 연구들을 통합함으로써, 실제 응용에 적용할 수 있는 위치정보 관리 시스템(LIMS, Location Information Management System)을 개발하기 위해 전체 시스템 구조와 각각의 서브시스템을 설계하는데 목적이 있다. 위치정보 관리 시스템의 전체 구조는 2 장에서 설명하고, 각 서브시스템은 3, 4, 5 장에서 설명하겠다. 그리고, 6 장에서는 결론과 향후 연구과제에 대해서 설명하겠다.

2. 위치정보 관리 시스템(LIMS)의 구조

이 논문에서 제안하는 위치정보 관리 시스템의 구조는 그림 1과 같으며, 세 개의 서브시스템으로 나뉜다. 각각의 서브시스템은 3, 4, 5 장에서 상세히 설명한다.

- 위치획득 서브시스템(LAS):
  위치획득 전략에 따라서, 이동객체의 현재위치

를 획득하여, 위치저장 서브시스템에 알린다.

- 위치저장 서브시스템(LSS):
  위치획득 서브시스템으로부터 보고된 위치정보는 메모리 버퍼를 통해서, 위치색인과 위치저장소에 저장된다.

- 위치질의 서브시스템(LQS):
  이동객체를 표현하기 위한 데이터구조와 연산자를 정의한 이동객체 모델을 기반으로 위치질의를 수행한다.

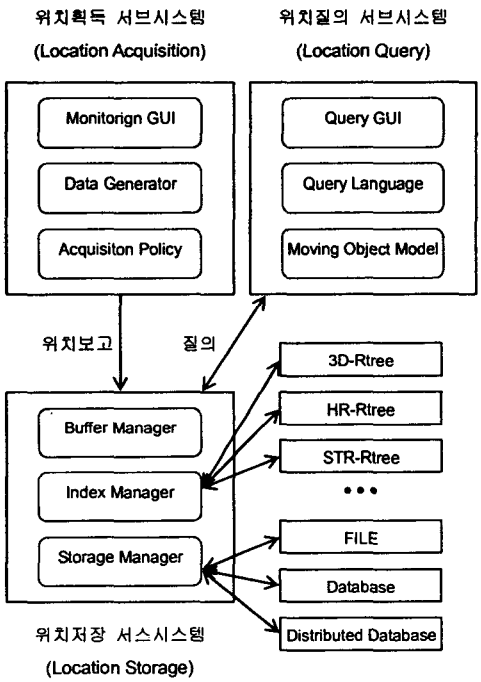

그림 1 위치정보 관리 시스템 구조

3. 위치획득 서브시스템

위치 획득 서브시스템은 이동객체의 위치정보를 획득하는 역할을 담당한다. 위치 획득 서브시스템은 다음과 같이 구성된다.

- Monitoring GUI:
  각 이동객체의 위치정보를 모니터링한다.

- Data Generator:
  실제 이동객체에 대한 위치 데이터 셋을 구하는 것은 어렵기 때문에, 이 논문에서는 GSTD[2,11], City Simulator[13] 등을 사용하여 이동객체에 대한 위치 데이터 셋을 생성한다.

- Acquisition Policy:
  이동객체의 위치정보를 획득하기 위한 전략을 제공한다.

위치획득 전략의 목적은 대용량 이동객체의 위치획득에 발생하는 통신 비용을 최소화하는데 있다. 위치획득 전략의 기본 개념은 이동객체의 위치정보 변동량이 적으면 위치획득 간격을 늘리고, 변동량이 크면 위치획득 간격을 줄이는 것이다. 즉, 과거의 위치정보를 기반으로 미래의 위치정보의 변화를 예측함으로써, 위치획득 요청을 줄이는 것이다. 위치획득 전략은 다음과 같이 구분된다.

- 정적 위치획득 정책:
  위치획득 시간 간격을 일정하게 정하여 위치를 획득하는 방법이다.

- 거리기반 위치획득 정책:
  이동객체의 이동거리의 변화를 이용하여 획득 시간 간격을 조정하는 것이다.

- 그룹기반 위치획득 정책:
  이동객체가 특정 시간대에 특정 지역에 군집하였다가 그 지역을 벗어날 때까지의 획득시간 간격을 늘여서 통신비용를 줄이는 방법이다.

- 예측기반 위치획득 정책:
  이동객체의 과거 우치정보를 기반으로 방향과 속도를 계산하여 미래의 위치를 예측함으로써, 위치획득을 비용을 줄이는 방법이다.

이 논문에서 제안하는 위치획득 서브시스템은 사용자가 위치획득 정책을 변경할 수 있도록 위치획득 프레임웍을 제공하고 있다. 특정 응용에 적합한 위치획득 정책이 존재할 경우에는, 사용자는 미리 정해진 규칙에 따라서 새로운 위치획득 정책을 쉽게 추가할 수 있는 특징이 있다. 예를 들어, 이동 객체의 이동 경로를 미리 알 수 있는 경우에는 이 정보를 활용하여, 위치획득 비용을 줄일 수 있다. 또는 실시간 교통정보를 활용할 수 있는 응용에서 차량의 위치정보는 획득할 경우에는 교통흐름에 따라 위치획득 간격을 조절함으로써 위치획득 비용을 줄일 수 있다.

4. 위치저장 서브시스템

위치저장 서브시스템은 버퍼 관리자, 색인 관리자, 저장 관리자로 구성된다.

4.1. 버퍼 관리자 (Buffer Manager)

메모리 버퍼는 각각의 이동객체를 MORow 객체를 통해 저장, 관리한다. MORow 객체는 그림 2와 같이 이동객체 구분자(MOID), 저장된 위치정보의 개수, 저장된 위치정보에 대한 최소경계사각형(MBR, Minimum Bounding Rectangle), 가장 오래된 위치정보의 획득 시간(From Time), 가장 최근의 위치정보 획득 시간(To Time), 위치정보로 구성된다. MORow 에 저장가능한

위치정보의 최대 개수(MaxLocation)는 고정되어 있으며, 저장된 위치정보의 개수가 최대 개수가 될 경우에는, 해당 MORow 를 저장 관리자를 통해 디스크에 저장한다.

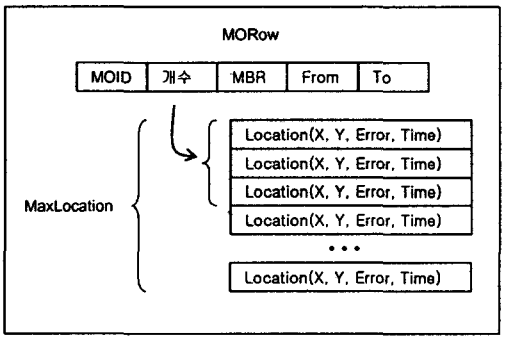

그림 2 MORow 구조

메모리 버퍼의 구조는 그림 3과 같이 각 이동객체별로 위치정보를 저장하고 있는 MORow 의 집합으로 구성된다. 그리고, 메모리 버퍼는 이동객체 식별자(MOID)를 통해서 해당 이동객체의 MORow 에 빠르게 접근하기 위해서 B-tree 색인을 가지고 있다.

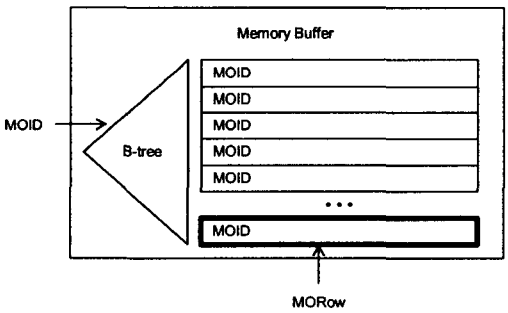

그림 3 메모리 버퍼의 구조

4.2. 색인 관리자 (Index Manager)

이동객체의 위치정보를 위한 색인은 크게 세가지 종류로 구분된다.

- 현재위치 색인:
  이동객체의 현재 위치정보만을 관리하는 색인[4,12]으로써, 대용량 이동객체의 위치정보에 대한 빈번한 갱신이 요구된다.

- 시간간격질의를 위한 과거위치 색인:
  현재 및 과거의 위치정보에 대해서 시간간격질의를 효과적으로 수행하기 위한 색인으로써, 3DR-tree[7], HR-tree[6] 등이 있다.

- 궤적질의를 위한 과거위치 색인:
  이동객체의 과거 위치정보에 대해서 궤적질의를 효과적으로 수행하기 위한 색인으로써, STR-tree[3], TB-tree[3] 등이 있다.

이 논문에서 설계한 색인 관리자는 모든 색인의 기반 클래스가 되는 MOIndex 를 정의하여, 인터페이스를 표준화함으로써, 전체 시스템의 변경없이 서로 다른 종류의 색인이 사용될 수 있도록 하였다.

4.3. 저장 관리자 (Storage Manager)

각각의 이동객체에 대해 저장될 과거 위치정보의 개수는 시간이 지남에 따라서 계속해서 증가하기 때문에, 저장 관리자는 위치정보를 저장하는 단위를 결정해야 한다. 이 논문에서는 그림 4와 같이 일정한 간격(Packaing Interval)과 미리 정의된 이동객체 식별자 리스트를 갖는 패키지를 저장단위로 사용한다.

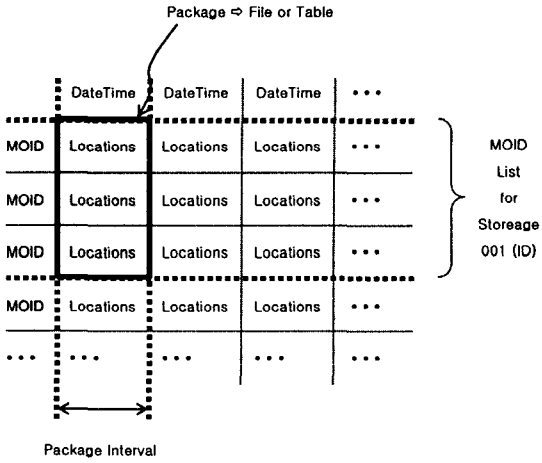

그림 4 위치정보 저장 단위 (패키지)

버퍼 관리자의 메모리 버퍼에서 관리되던 이동객체의 위치정보는 저장 관리자를 통해서 다양한 위치저장소에 저장된다. 이때, MORow 의 위치정보는 압축되어 저장된다. 위치저장소에는 파일, 데이터베이스, 분산데이터베이스가 있다.

- FILE:
  하나의 패키지를 하나의 파일에 저장한다.

- Database:
  하나의 패키지를 하나의 테이블에 저장한다.

- Distributed Database:
  저장단위를 사용하지 않는다. 즉, 패키지로 구분하여 저장되지 않는다. 분산데이터베이스 내부의 분산저장 정책에 따라서 이동객체의 저장방법이 결정된다.

5. 위치질의 서브시스템

위치질의 서브시스템에서는 이동객체 모델과 질의어를 정의해야 한다. 이동객체 모델은 이동객체를 위한 데이터 구조와 연산자를 의미한다. 이 논문에서는 [5,9]의 연구내용을 기반으로 MPoint, MLineSegment, MPolygon 을 정의하였다. 그림 5는 이동객체의 계층구조를 보이고 있다. IMoving, ITimeSeries 는 이동객체를 위한 연산자를 정의하고 있으며, 모든 이동객체가 이를 상속받아 구현하도록 설계하였다.

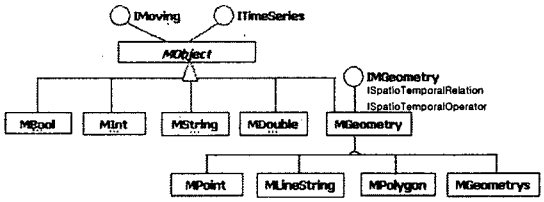

그림 5 이동객체 계층구조

위치질의어는 SQL 구문을 확장하여 그림 6와 같이 정의하였다. 위치질의는 기존의 SQL 질의와 이동객체를 위해 추가된 MOSELECT, MOWHERE 를 서로 분리해서 처리한다. 즉, 기존 SQL 질의는 속성 질의로써, 위치질의 서브시스템에서 처리하지 않고, 속성 정보를 저장하고 있는 데이터베이스를 통해서 처리한다.

```
SELECT          select_target_list
MOSELECT        moselect_target_list
FROM            table_list
WHERE           where_string
MOWHERE         mowhere_string

moselect_list   = function
                  | function , moselect_list
mowhere_string  = function
                  | function and mowhere_string
                  | not ( mowhere_string )
function        = function_name ( parameter_list )
function_name   = overlaps | snapshotbyvalidtime | slicebyvalidtime | ...
parameter_list  = parameter
                  | parameter, parameter_list
parameter       = function | object | const
object          = point() | line() | area() | period() | time() | mosmbr()
                  | string(geometry_column_name) | ...
```

그림 6 이동객체 질의어

6. 결론

이 논문에서는 대용량 이동객체를 위한 위치정보 관리 시스템을 설계하였다. 이 논문에서 제안한 위치정보 관리 시스템에서는 이동객체와 관련된 기존의 연구 결과물을 통합하였을 뿐아니라, 기존의 연구를 기반으로 이동객체 모델, 이동객체 질의어, 이동객체 저장 방법 등을 새롭게 제시하고 있다. 제안한 시스템은 다양한 위치획득 전략, 위치 색인, 위치 저장소등을 지원하며, 특히, 응용의 종류에 따라서 각각의 컴포넌트를 조합할 수 있는 특징을 갖고 있으므로, 다양한 위치기반서비스에서 활용될 수 있을 것으로 기대된다. 향후 연구과제로는 제안한 시스템을 구현하고, 실제 응용에서 적용될 수 있도록 다양한 실험환경에서 시스템의 성능을 평가하는 것이다.

참고문헌

[1] A. Prasad Sistla, Ouri Wolfson, Sam Chamberlain, and Son Dao, "Modeling and Querying Moving Objects," ICDE, pp.422-432, 1997

[2] Dieter Pfoser and Yannis Theodoridis, "Generating Semantics-Based Trajectories of Moving Objects," International Workshop on Emerging Technologies for Geo-Based Applications, Ascona, Switzerland, 2000

[3] Dieter Pfoser, Christian S. Jensen, and Yannis Theodoridis, "Novel Approaches in Query Processing for Moving Object Trajectories," VLDB 2000, 395-406

[4] George Kollios, Dimitrios Gunopulos, and Vassilis J. Tsotras, "On Indexing Mobile Objects," ACM Symp. on Principles of Database Systems, pp261-272, 1999

[5] Luca Forlizzi, Ralf H. Güting, Enrico Nardelli, Markus Schneider, "A Data Model and Data Structures for Moving Objects Databases," Proc. ACM SIGMOD Conf. (Dallas, Texas), pp. 319-330, May 2000.

[6] M. A. Nascimento and J. R. O. Silva, "Towards historical R-trees," ACM SAC, 1998

[7] Michalis Vazirgiannis, Yannis Theodoridis, and Timos K. Sellis, "Spatio-Temporal Composition and Indexing for Large Multimedia Applications," Multimedia Systems 6(4), 284-298 (1998)

[8] Ouri Wolfson, Bo Xu, Sam Chamberlain, and Liqin Jiang, "Moving Objects Databases: Issues and Solutions," SSDBM 1998, 111-122

[9] R.H. Güting, M.H. Böhlen, M. Erwig, C.S. Jensen, N.A. Lorentzos, M. Schneider, and M. Vazirgiannis, "A Foundation for Representing and Querying Moving Objects," FernUniversität Hagen, Informatik-Report 238, September 1998, ACM Transactions on Database Systems, 25(1):1-42, 2000

[10]Y. Tao and D. Papadias, "MV3R-Tree: A Spatio-Temporal Access Method for Timestamp and Interval Queries," VLDB, 2001

[11]Yannis Theodoridis, Jefferson R. O. Silva and Mario A. Naschimento, "On the Generation of Spatiotemporal Datasets," CHOROCHRONOS Technical Report CH-99-01, Proceedings of the 16th Int'l Symposium on Spatial Databases (SSD), 1999

[12]Zhexuan Song 1 and Nick Roussopoulos, "Hashing Moving Objects," MDM 2001, LNCS 1987, pp. 161-17, 2001

[13]http://www.alphaworks.ibm.com/tech/citysimulator