이동체의 실시간 관리를 위한 시공간 DSMS†
A Spatio-Temporal DSMS for the Real-time Management of Moving Objects

김장우*, 박춘걸, 김동오, 한기준
Jang-Woo Kim*, Chun-Geol Park, Dong-Oh Kim, Ki-Joon Han
건국대학교 컴퓨터·정보통신공학과
{zwkim*, cgpark, dokim, kjhan}@db.konkuk.ac.kr

요 약

오늘날 모바일 장치 기술, 위치 측위 기술, 무선 통신 기술 등이 급속도로 발달하고 이동체 위치 데이타가 널리 활용됨에 따라 차량 관리 시스템, 차량 배차 및 제어 시스템 등과 같은 이동체 위치 데이타를 실시간으로 서비스하기 위한 시스템이 개발되고 있다. 그러나 이러한 시스템에서 기반 시스템으로 사용되는 MO(Moving Object) DBMS 같은 이동체 관리 시스템은 이동체의 실시간 스트림 관리에 비효율적이고, 기존의 DSMS(Data Stream Management System)와 같은 스트림 관리 시스템은 공간 데이타를 효율적으로 처리하지 못하고 있다.

따라서, 본 논문에서는 이동체 위치 데이타의 효율적인 실시간 관리를 위한 시공간 DSMS를 설계 및 구현하였다. 본 논문에서 구현한 시공간 DSMS는 스탠포드 대학의 STREAM(STanford stREam dAta Manager)을 기반으로 이동체 위치 데이타의 실시간 관리와 공간 및 시공간 질의 처리 기능을 지원하는 시스템이다. 특히, 시공간 DSMS에서 사용하는 시공간 함수는 호환성을 위해서 OGC에서 제시한 "SQL을 위한 심플 피쳐 명세"를 따르는 표준 인터페이스를 지원한다. 마지막으로 본 논문에서 구현한 시공간 DSMS를 이동체 위치 데이타의 실시간 관리가 필요한 실시간 모니터링 분야에 적용해 봄으로써 시스템의 효용성을 입증하였다.

1. 서 론

오늘날 RFID, GPS 등을 통한 위치 측위 기술의 발달로 이동체 위치 데이타 수집 수단이 획기적으로 발전하고 그 활용 범위가 다양해지고 있다. 또한 여러 분야에서 이동체 위치 데이타를 이용한 서비스가 활용됨에 따라 차량 관리 시스템, 차량 배차 및 제어 시스템 등과 같은 이동체 위치 데이타를 실시간으로 서비스하기 위한 시스템이 개발되고 있다[1,2].

이러한 이동체 위치 데이타를 실시간으로 원활히 서비스하기 위해서는 위치 데이타를 실시간으로 처리할 수 있는 기반 시스템이 필요하다. 그러나 이러한 기반 시스템으로 MODBMS와 같은 이동체 관리 시스템을 사용할 경우 지속적으로 입력되는 실시간 스트림 관리에 비효율적이며[3], 기존 DSMS와 같은 스트림 관리 시스템을 사용할 경우 위치 데이타 처리가 비효율적이라는 문제점을 가지고 있다[4].

† 본 논문은 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음.

따라서 본 논문에서는 이동체 위치 데이타의 효율적인 실시간 관리를 위해 시공간 DSMS를 설계 및 구현하였다. 본 논문에서 개발한 시공간 DSMS는 스탠포드 대학의 STREAM[5]을 기반으로 이동체 위치 데이타의 실시간 관리와 공간 및 시공간 질의 처리 기능을 지원하는 시스템이다. 특히, 시공간 DSMS에서 사용하는 시공간 함수는 호환성을 위해서 OGC에서 제시한 "SQL을 위한 심플 피쳐 명세[6]"를 따르는 표준 인터페이스를 지원한다. 마지막으로 본 논문에서 구현한 시공간 DSMS를 이동체 위치 데이타의 실시간 관리가 필요한 실시간 모니터링 분야에 적용해 봄으로써 시스템의 효용성도 입증하였다.

본 논문의 구성은 다음과 같다. 제 2장에서 DSMS에 대해 간단히 소개하고, 기존 STREAM에 대하여 알아본다. 제 3장에서는 시공간 DSMS의 설계에 대하여 설명하며, 제 4장에서는 시공간 DSMS의 구현에 대해 설명하고, 가상 시나리오를 통해 본 시스템의 효용성을 검증한다. 마지막으로 제 5장에서는 결론에 대하여 언급한다.

2. 관련 연구

본 장에서는 DSMS에 대한 일반적인 특징과 구조를 설명하고, 본 논문에서 시공간 DSMS을 구현하기 위해 사용한 기반 시스템인 STREAM에 대하여 살펴본다.

2.1 DSMS

DSMS는 지속적으로 끊임없이 입력되는 데이타 스트림을 실시간으로 처리하기 위한 스트림 관리 시스템이다[3,5]. DSMS는 효율적인 데이타 스트림 처리를 위해 기존 DBMS와 같은 데이타를 로드한 후에 인덱스를 생성하고 질의하는 과정을 탈피한다. 즉, 데이타를 미리 로드하여 처리하는 것이 아니라 실시간으로 처리 하기위해 질의를 미리 등록하여 실시간으로 들어오는 연속적인 데이타 스트림을 처리한다.

일반적인 DSMS의 세부구조는 그림 1과 같다[3].

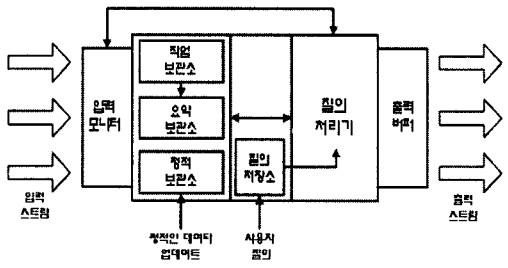

그림 1. DSMS의 세부구조

일반적으로 DSMS는 지속적인 입력을 제어하는 입력 모니터, 지속적 질의를 저장하는 질의 저장소, 이를 지속적으로 처리하는 질의 처리기, 소스의 위치 정보와 같은 시스템에 필요한 정적인 정보를 저장하는 정적 보관소, 질의 대상이 되는 데이타를 저장하는 작업 보관소와 메모리의 요약정보를 저장하는 요약 보관소, 처리 결과를 출력하는 출력 버퍼 등으로 구성된다.

2.2 STREAM

STREAM은 데이타 스트림을 처리하기 위하여 스탠포드 대학에서 개발한 DSMS이다. 현재 서버 0.6 버전과 클라이언트로 구성되었으며 소스는 BSD 라이센스 형태로 공개되어 있다[5]. 서버는 Linux 기반의 C++로 구현되어 있으며, 연속적이고 무한히 생성되는 데이타 스트림을 효과적으로 처리한다.

STREAM은 저장되어 있는 데이타 집합에 대한 연속 질의를 지원하며, 연속 질의를 위한 질의 언어로 SQL과 유사한 질의 언어인 CQL(Continuous Query Language)을 사용한다[7]. 클라이언트는 JAVA로 구현되어 있으며, 데이타 스트림 입력 및 질의 입력 기능 등을 제공한다.

그림 2는 STREAM의 동작 방식을 보여주고 있다.

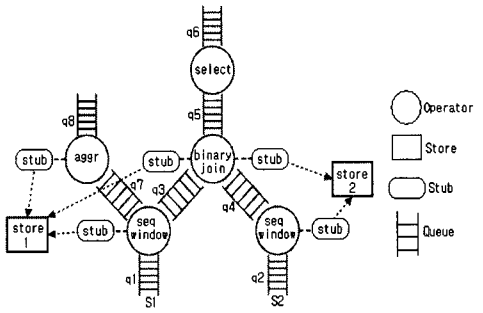

그림 2. STREAM의 동작 방식

그림 2에서 보는 바와 같이 사용자가 등록한 연속 질의는 Select, Project, Join 등의 연산자가 트리 형태로 모여 있는 실행 계획으로 변환되어 저장 및 처리된다. 각 연산자는 필요에 따라 저장소를 생성하게 되며 이 저장소에는 질의 처리의 중간 결과가 저장되어 여러 연산자에 의해 공유된다. 각 연산자는 Stub라는 상태 저장 모듈을 갖고 있어서 저장소에 대한 참조 값을 보관하며, 연산자들은 Queue로 연결되어 있어서 입력 큐의 데이타를 처리하여 출력 큐에 저장한다.

3. 시스템 설계

본 장에서는 시스템 전체 구조에 대해 설명하고, 시공간 DSMS와 클라이언트의 각 관리자에 대해 자세히 설명한다.

3.1 시스템 전체 구조

시공간 DSMS의 시스템 전체 구조는 그림 3과 같다.

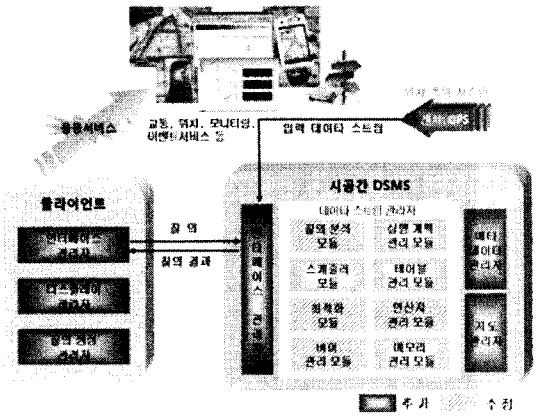

그림 3. 시스템 전체 구조

그림 3에서 보는 바와 같이 시공간 DSMS는 기존 STREAM의 질의 분석 모듈, 연산자 관리 모듈, 최적화 모듈, 버퍼 관리 모듈을 수정하여 확장한 데이타 스트림 관리자, 본 논문에서 추가한 인터페이스 관리자, 메타데이타 관리자, 지도 관리자, 그리고 질의 요청 및 결과 반환을 위한 클라이언트로 구성되어 있다.

3.2 시공간 DSMS의 관리자

시공간 DSMS는 인터페이스 관리자, 데이타 스트림 관리자, 메타데이타 관리자, 지도 관리자로 구성된다.

인터페이스 관리자는 센서, GPS 등과 같은 위치 측위 장치로 부터 데이타 스트림을 입력받아 버퍼 관리 모듈에 전달하며, 클라이언트로부터 받은 질의를 질의 분석 모듈에게 전달하기 위해 네트워크 연결을 관리하는 역할을 담당한다. 입력 질의로 사용될 수 있는 데이타 타입은 "SQL을 위한 심플 피쳐 명세"에 제시된 데이타 타입과 공간 함수를 확장한 시공간 함수를 포함한다.

본 시스템에서 지원하는 공간 데이타 타입은 표 1과 같다.

표 1. 지원하는 공간 데이타 타입

Geometry	Type SQL Text Literal Representation
Point	POINT (10 10)
LineString	LINESTRING (10 10, 20 20, 30 40)
Polygon	POLYGON ((10 10, 10 20, 20 20, 20 15, 10 10))
MultiPoint	MULTIPOINT (10 10, 20 20)
MultiLineString	MULTILINESTRING ((10 10, 20 20) (15 15, 30 15))
MultiPolygon	MULTIPOLYGON (((10 10, 10 20, 20 20, 20 15, 10 10)), ((60 60, 70 70, 60 60, 60 60)))

표 1에서 보는 바와 같이 점 데이타를 표현하는 Point, 선을 표현하는 LineString, 다각형을 표현하는 Polygon, 다중 점 데이타를 표현하는 MultiPoint, 다중 선 데이타를 표현하는 MultiLineString, 다중 다각형 데이타를 표현하는 MultiPolygon 타입을 지원한다.

본 시스템에서 추가한 시공간 함수의 목록은 표 2와 같다.

표 2. 확장된 시공간 함수 목록

종 류	이 름	설 명
시공간 관계 함수	ST_Equals	두 시공간 객체가 동일하면 참을 반환함
	ST_Disjoint	두 시공간 객체가 교집합이 없으면 참을 반환함
	ST_Touches	두 시공간 객체의 경계선만 닿았으면 참을 반환함
	ST_Within	첫 번째 시공간 객체가 두 번째 시공간 객체를 포함하면 참을 반환함
	ST_Overlaps	두 시공간 객체가 같은 차원이면서 겹쳐 있으면 참을 반환함
	ST_Crosses	두 시공간 객체가 교차하면서 교차된 영역의 차원이 같거나 줄어들면 참을 반환함
	ST_Intersects	두 시공간 객체가 교차하면 참을 반환함
	ST_Contains	두 번째 시공간 객체가 첫 번째 시공간 객체를 포함하면 참을 반환함
	ST_Relate	두 시공간 객체의 관계를 DE-9IM 문자열로 반환함
시공간 분석 함수	ST_Distance	두 시공간 객체간의 최단거리를 계산함
	ST_Intersection	두 시공간 객체가 겹치는 부분을 시공간 객체로 반환함
	ST_Difference	첫 번째 시공간 객체에서 두 번째 시공간 객체를 뺀 부분을 반환함
	ST_Union	두 시공간 객체를 합친 시공간 객체를 반환함
	ST_Buffer	주어진 시공간 객체 주변을 주어진 크기만큼 증가시킨 시공간 객체를 반환함

데이타 스트림 관리자는 연속적으로 입력되는 이동체의 데이타 스트림을 효율적으로 처리하는 역할을 담당하는데, 등록된 질의를 분석하기 위한 질의 분석 모듈, 분석된 질의를 실행 계획으로 변경하기 위한 실행 계획 관리 모듈, 공간 및 시공간 함수를 효율적으로 수행하기 위한 연산자 관리 모듈, 등록된 실행 계획들을 연산자 단위로 스케줄링 하여 처리하기 위한 스케줄러 모듈, 테이블을 생성하고 생성된 테이블의 스키마를 관리하기 위한 테이블 관리 모듈, 메모리와 같은 시스템 자원을 미리 확보하여 작업에 대한 최적화를 수행하기 위한 최적화 모듈, 사용할 메모리들을 관리하기 위한 메모리 관리 모듈로 구성된다.

메타데이타 관리자는 위치 측위 장치의 메타 정보를 관리하며, 등록된 질의에 대한 로그 정보를 기록하여 시스템 오류와 같은 이벤트 발생시 최적화된 작업처리를 지원하는 역할을 담당한다. 지도 관리자는 지도의 특정한 지역을 확대나 축소하기 위한 확대 축소 모듈, 지도의 특정 영역으로 이동하기 위한 영역 이동 모듈로 구성되는데, 사용자의 위치에 적합한 환경에 맞는 지도 정보를 효율적으로 제공하는 역할을 담당한다.

3.3 클라이언트의 관리자

클라이언트는 질의 생성 관리자, 인터페이스 관리자, 디스플레이 관리자로 구성되어 있다. 질의 생성 관리자는 실제 사용자로부터 입력받는 질의를 파싱한 후 시공간 DSMS측 인터페이스 관리자에게 전달하기 위해 클라이언트의 인터페이스 관리자에게 전달하는 역할을 담당한다.

인터페이스 관리자는 클라이언트가 시공간 DSMS에 접근하여 네트워크 통신이 가능하도록 연결을 유지한다. 또한 질의 생성 관리자로 부터 전달받은 질의를 시공간 DSMS에 전달하며, 시공간 DSMS로부터 전달받은 질의 처리 결과를 화면에 디스플레이 하기 위해 디스플레이 관리자에게 전달하는 역할을 담당한다. 디스플레이 관리자는 인터페이스 관리자로부터 넘겨받은 데이타 스트림을 파싱한 후 화면에 디스플레이 하기 위한 역할을 담당한다.

4. 시스템 구현

본 장에서는 시스템의 구현 환경과 구현 상세 내용을 설명하고, 가상 시나리오를 통해 시스템의 효용성을 실험한다.

4.1 구현 환경

본 논문에서 운영체제는 Ubuntu 6.06을 사용하였고, 개발 도구는 g++ 3.2.3 버전을 사용하였다. 클라이언트는 Windows XP Professional을 사용하였고, 개발 도구는 JAVA 1.5 버전을 사용하였다.

4.2 시공간 DSMS 구현

본 논문에서는 시공간 질의 처리를 위해 기존 STREAM을 기반으로 시공간 연산을 처리할 수 있도록 질의 분석 모듈, 최적화 모듈, 연산자 관리 모듈, 버퍼 관리 모듈 등을 보완 및 수정하였다.

질의문에 포함된 시공간 함수는 질의 분석 모듈을 통해 분석된 후 실행 계획 관리 모듈에 의해 실행 계획으로 변환되어 수행되며, 연산자 관리 모듈의 요청에 따라 적절한 시공간 함수를 호출하여 처리한 뒤 그 결과를 반환하게 된다.

구현된 시공간 DSMS의 테스트를 위한 질의로 공간 객체 A가 공간 객체 B를 포함하는지를 반환하는 시공간 함수인 ST_Contains를 사용하여 질의문을 생성하였으며, 그림 4는 질의 처리 결과를 보여준다.

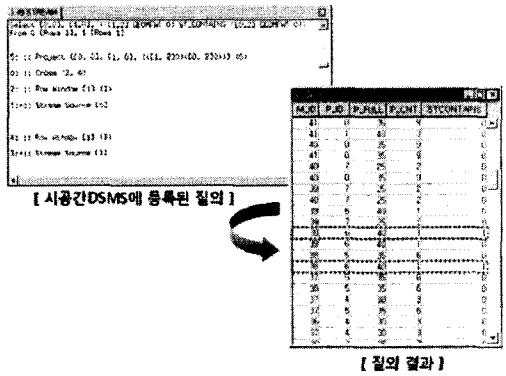

그림 4. 질의 처리 결과 화면

그림 4에서는 38번 ID를 가진 이동체가 6번 ID를 가진 공간에 진입하여 ST_Contains의 결과로 True값을 반환한 것을 확인할 수 있다. 그림 5는 시공간 DSMS와 클라이언트의 실행 화면을 보여준다.

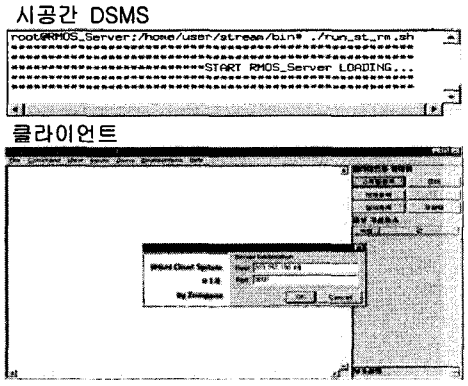

그림 5. 시스템 실행 화면

그림 5의 클라이언트내의 다이얼로그 박스는 시공간 DSMS에 연결하기 위해 호스트 IP 주소와 포트를 설정하기 위한 것이다.

4.3 가상 시나리오

시공간 DSMS의 효용성을 검증하기 위해 실시간 주차 모니터링에 시공간 DSMS를 적용하였으며, 또한 이동 차량과 주차구역 차량 인식 센서 등을 임의로 생성하여 그에 따른 주차구역에 진입하는 이동 차량의 감지, 주변 주차구역의 검색 등을 실험하였다.

실험을 위해 이동 차량과 주차구역의 테이블을 생성하였으며, 생성한 이동 차량 (M) 테이블에는 이동 차량 id와 현재 위치 정보가 있고, 주차구역 (P) 테이블에는 주차구역 id와 공간 정보 외에 전체 주차공간의 수량과 남은 주차공간의 수량 정보가 있다.

본 실시간 주차 모니터링에서 사용한 질의 목록은 표 3과 같다.

표 3. 사용된 질의 목록

이 름	질 의
질의 1 (Geo_m)	Select id, GeomFromText(M.loc, 0) From M [range 2 seconds];
질의 2 (Geo_p)	Select id, GeomFromText(P.loc, 0), full, cnt From P [range 2 seconds];
질의 3 (st_cont)	Select M.id, P.id, P.full, P.cnt, ST_Contains(GeomFromText(P.loc, 0), geomFromText(M.loc, 0)) From M, P;
질의 4 (st_dist)	Select M.id, P.id, ST_Distance(GeomFromText(P.loc, 0), geomFromText(M.loc, 0)) From M, P;
질의 5	Select M_id, P_id, From st_dist Where s_dist < 500;

<질의 1>는 이동차량의 위치를 모니터링하고, <질의 2>는 근처 주차구역의 현황을 모니터링 하고, <질의 3>은 현재 이동차량이 어느 주차구역에 진입했는지를 모니터링하고, <질의 4>는 주변 주차구역과의 거리를 구하고, <질의 5>는 반경 500m 내에 있는 주변 주차구역을 검색한다.

실시간 주차 모니터링은 표 3에서 사용된 질의를 통해 입력 데이타 스트림을 처리하게 된다. 그림 6은 등록된 질의에 대한 질의 처리 결과를 보여준다.

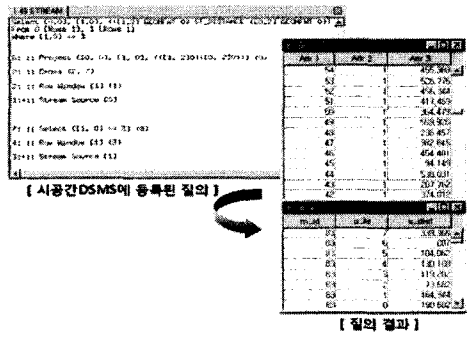

그림 6. 질의 처리 결과 화면

그림 6은 이동 차량과 주차구역의 거리를 구하는 <질의 4>의 처리 결과와 구해진 거리를 기반으로 이동 차량의 반경 500m 내에 있는 주차구역을 검색해서 출력하는 <질의 5>의 처리 결과를 보여주고 있다.

그림 7은 질의 처리 결과에 따른 클라이언트의 디스플레이 관리자 처리 화면이다.

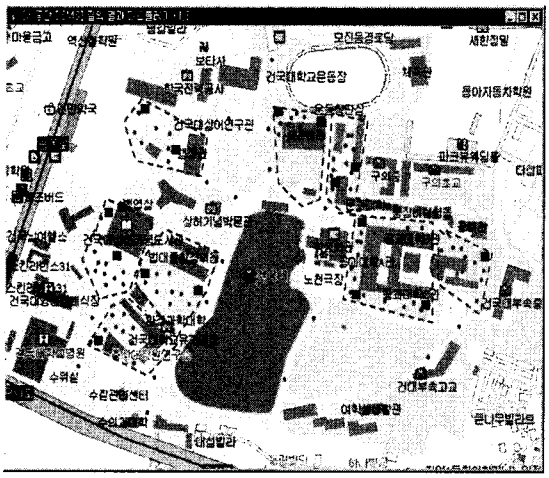

그림 7. 질의 처리 결과 디스플레이

그림 7은 <질의 1>과 <질의 2>의 처리 결과인 이동 차량의 현재 위치와 주차구역의 위치, 주차구역내의 전체 주차 공간의 수 및 주차가능 공간의 수를 디스플레이 관리자를 통해 디스플레이 화면으로 보여주고 있다. 이 때 화면 상의 원은 이동 차량, 사각형은 센서, 점선은 주차 구역을 나타낸다.

5. 결 론

본 논문에서는 기존 MODBMS와 같은 이동체 관리 시스템이 갖는 비효율적인 실시간 스트림 관리 문제와 기존 DSMS에서 공간 및 시공간 데이타를 처리하지 못하는 문제를 해결하기 위해 이동체 위치 데이타의 효율적인 실시간 처리를 지원하는 시공간 DSMS를 개발하였다.

시공간 DSMS는 스탠포드 대학의 STREAM을 기반으로 질의 분석 모듈, 연산자 관리 모듈, 최적화 모듈, 버퍼 관리 모듈을 수정하여 다양한 공간 및 시공간 질의 처리 기능을 지원할 수 있도록 확장하였다. 또한 시공간 DSMS를 실시간 이동체 관리가 필요한 가상 시나리오에 적용해 봄으로써 시스템의 효용성도 검증하였다.

참고문헌

[1] 지능형교통시스템(ITS)전담기구, 교통·전자·통신등을 통한 실시간 교통정보 시스템, http://www.itskorea.or.kr/its/definition.asp, 2006.

[2] 원종호, 이미영, 김명준, "유비쿼터스 컴퓨팅 환경을 위한 RFID 기반 센서 데이타 처리 미들웨어 기술 동향," 전자통신동향분석, 19권 5호, 2004, pp. 21-29.

[3] Golab L., and Özsu M. T., "Issues in Data Stream Management," SIGMOD Record, Vol.32, No.2, 2003, pp. 5-14.

[4] 강홍구, 박치민, 홍동숙, 한기준, "공간 센서 데이타의 효율적인 실시간 처리를 위한 공간 DSMS의 개발," 한국공간정보시스템학회 논문지, 2007(게재 예정)

[5] Arasu, A., Babcock, B., Babu, S., Cieslewicz, J., Datar, M., Ito, K., Motwani, R., Srivastava, U., and Widom, J., *STREAM: The Stanford Data Stream Management System,* http://dbpubs.stanford.edu/pub/2004-20, 2004.

[6] Open Geospatial Consortium Inc., *OpenGIS Implementation Specification for Geographic information - Simple Feature Access - Part 2: SQL option*, 2005.

[7] Arasu, A., Babu, S., and Widom, J., "The CQL Continuous Query Language: Semantic Foundations and Query Execution," The VLDB Journal, Vol.15, No.2, 2006, pp. 121-142.