DOI: 10.3745/KIPSTD.2008.15-D.5.599

위치 및 RFID 기반의 물류 환경을 위한 이벤트 통지 시스템의 설계 및 구현

이 용 미[†]·남 광 우[††]·류 근 호[†††]

요 약

유무선 통신기기 및 센서 기술의 발달은 물류 환경에서도 화물 및 주변 환경의 온도, 습도, 무게, 위치 등의 상황 정보를 실시간으로 수집하는 것을 가능하게 한다. 또한, 사용자는 언제 어디서든 자신이 관심 있는 화물에 대한 상황 정보를 실시간으로 얻기 원한다. 이러한 요구를 만족시키기 위해, 응용은 상황 정보를 실시간으로 수집하여 분석하고, 원하는 사용자에게 전달할 수 있는 서비스를 제공하여야 한다. 이벤트 기반의 서비스는 이러한 요구를 만족시킬 수 있는 방법 중의 하나이다. 이 논문에서는 위치 및 RFID 기반의 물류 환경에 초점을 맞추어 이벤트 통지 시스템을 설계한다. 이를 위해, XML 기반의 이벤트의 표현 모델과 ECA 기반의 프로파일 정의 모델을 제시하고, 2단계로 분리된 효율적인 이벤트 필터링 기법을 제안한다. 또한, 이를 기반으로 구현된 시스템은 물류 환경뿐만 아니라, RFID나 GPS 장치를 기반으로 하는 지능형 교통관리 시스템 등에 광범위하게 적용할 수 있다.

키워드 : 이벤트 통지 시스템, 이벤트 필터링, 프로파일 정의어

Design and Implementation of Event Notification System for Location- and RFID-based Logistics Environment

Yongmi Lee[†]· Kwang Woo Nam[††]· Keun Ho Ryu[†††]

ABSTRACT

Advanced wireless network and sensor technologies are capable of collecting information such as temperature, humidity, weight, and location about objects at real time in logistics area. Besides, users want to be notified of contextual information about interest of objects whenever they want it and wherever they want it. To satisfy these requirements, applications should collect and analyze contextual information at real time, and must support a service that can notify it to wanted users. Event-based service is one of the way to satisfy these requirement of users. In this paper, we design an event notification system focused on location- and RFID-based logistics area. To do this, we present XML-based event expression model, ECA-based profile definition model, and an algorithm that has high scalability by distinguishing event filtering in two steps. Based on these designs, our implemented system can apply to not only logistics area but also intelligent traffic control system based on RFID or GPS devices.

Keywords : Mobile Event Notification System, Event Filtering, Profile Definition Language

1. 서 론

최근의 유·무선 통신기기 및 센서 기술의 발달과 함께, USN (Ubiquitous Sensor Network) 환경하의 시스템들은 지능화된 사물로부터 온도, 습도, 무게, 위치 등의 실시간으로 변하는 상황 정보를 수집하는 것이 가능해졌다. 변화하는 환경과 더불어 응용도 객체에 대한 상황 정보를 실시간으로 전달하는 서비스를 제공하기 위해 변화하고 있다. 이러한 변화는 우편, 항만 등의 물류 환경 분야에서도 예외는 아니어서 물류 차량, 화물, 팔레트 등에 RFID 센서를 부착하고 객체에 대한 상황 정보를 자동으로 식별하는 기반 환경을 구축하고 있다[1].

USN 응용으로서 물류 환경을 고려할 때, 사용자가 알기 원하고, 서비스로서 제공되어야 하는 정보는 관심 있는 화물 또는 차량과 같은 객체에 대한 상황 및 위치 정보일 것이다. 그러나 기존의 온라인과 오프라인이 분리된 환경에 운영되는 대부분의 시스템들은 주로 운영자의 키보드나 화면 조작을 통해 객체에 대한 정보를 얻게 되므로 변화된 환경에 적응하는데

※ 이 논문은 2007년도 충북대학교 학술연구지원사업의 연구비 지원에 의하여 연구되었음.
† 준 회 원 : 충북대학교 전자계산학과 박사과정
†† 정 회 원 : 군산대학교 컴퓨터정보공학과 조교수
††† 종신회원 : 충북대학교 전기전자컴퓨터공학부 교수(교신저자)
논문접수 : 2007년 9월 3일
수 정 일 : 1차 2008년 5월 26일
심사완료 : 2008년 6월 13일

한계가 있다. 이러한 요구들은 객체에 대한 상황 정보를 수집하기 위해 RFID 외에 위치 정보를 인식할 수 있는 GPS 장치를 도입하고, 응용이 이벤트 통지 서비스(Event Notification Service)를 지원함으로써 해결이 가능하다. 이 서비스에서는 사용자가 원하는 정보를 프로파일로 정의하면, 시스템은 각 객체로부터 발생한 이벤트를 수집하고 사전에 정의된 프로파일과의 필터링을 통하여 원하는 사용자에게 통지 메시지를 전달하게 된다[2, 3, 4].

이벤트 기반의 서비스를 제공하는 시스템은 다양한 이벤트 소스를 정확하게 표현하고 전달할 수 있는 이벤트 표현 모델과 객체에 대한 프로파일을 단순하고 정확하게 정의할 수 있는 프로파일 정의 모델을 제공하여야 한다. 또한, 시스템은 성능을 저하시키는 원인 중의 하나인 이벤트 필터링의 비용을 감소시킬 수 있는 효율적인 알고리즘을 제공하여야 한다.

따라서 이 논문에서는 RFID 및 GPS 장치에 기반한 물류 환경에 응용 가능한 이벤트 통지 시스템을 위한 이벤트 및 프로파일의 표현, 알고리즘, 시스템 구조의 설계 및 구현에 초점을 맞춘다. 첫째, 이기종의 플랫폼 간에 상호운용성(interoperability)을 보장할 수 있는 API로서 가장 많이 사용되는 XML에 기반한 이벤트 표현 모델과 이벤트 발생시 평가되어야 할 조건 및 그 결과로서 취해져야 하는 행위를 기술할 수 있는 ECA 패러다임에 기반한 프로파일 정의 모델을 제시한다. 둘째, 이벤트 통지 시스템의 내부 구조를 설계하고, 셀렉션 프리디킷과 함수 프리디킷을 분리한 2단계 이벤트 필터링 기법을 제안한다. 이 기법은 데이터베이스 기반의 필터링 기법을 사용함으로써 비교 연산자(comparison operator)들을 폭넓게 수용할 수 있을 뿐만 아니라, 좀 더 복잡한 계산을 요구하는 부분을 분리함으로써 필터링에 소요되는 시간을 단축시킬 수 있다.

이 논문의 구성은 다음과 같다. 제2장에서는 데이터의 표현과 이벤트 필터링 기법에 관련된 이전의 연구들을 분석하고, 제3장에서는 이벤트 표현 및 프로파일 정의 모델에 대하여 기술한다. 제4장에서는 이를 기반으로 시스템의 구조를 설계하고, 이벤트 필터링의 효율성을 높이기 위한 2단계 필터링 기법을 제안한다. 제5장에서는 제안된 시스템의 구현 결과를 보이고, 제6장에서 결론을 맺는다.

2. 관련 연구

이벤트 통지 시스템을 설계하고 구현하는 데 있어서 연구되어야 할 내용은 응용이 물리적인 환경으로부터 객체에 대한 상황 정보인 이벤트를 어떻게 표현하고 수집할 것인가, 사용자의 관심인 프로파일을 어떻게 정의하도록 할 것인가, 그리고 그 데이터들을 바탕으로 시스템 내부에서 어떤 필터링 기법을 사용할 것인가에 관한 것이다. 이것은 대용량의 연속적인 데이터를 실시간으로 수집해야만 하는 환경에서는 이러한 요인들이 시스템의 성능에 많은 영향을 끼칠 수 있기 때문이다.

2.1 이벤트 및 프로파일의 표현

이벤트란 객체에 대한 상태 변화를 의미하며[2], 특정한 구조를 가진 메시지로 표현되어 시스템으로 전달된다. 이벤트를 표현하는 가장 단순한 방법은 (속성, 값)의 쌍으로 이루어진 프리디킷의 조합을 사용하여 속성을 열거하는 것으로, 대표적인 시스템으로 Yeast[3], Siena[4], Lesubscribe[5] 등이 있다. YFilter[6]와 같은 시스템들은 이벤트를 XML 기반의 계층적인 구조를 가진 메시지로 표현한 이벤트 모델을 제시하고 있다. 또한, IBM의 Auto-ID Center에서는 EPC (Electronic Product Code) 네트워크를 구성하는 컴포넌트들 사이에 센서 기반의 환경에서 측정된 이벤트를 표현하고 전달하기 위한 스키마(schema)로서 PML(Physical Markup Language) Core[7]을 제시하고 있다.

프로파일이란 클라이언트에 의해 정의된 이벤트에 대한 사용자의 관심을 의미한다. 시스템은 객체에 대한 이벤트가 발생하면, 시스템에 저장된 모든 프로파일에 대해 필터링을 시도한다[2]. 따라서 프로파일의 정의 방법은 시스템에 의해 제공되는 이벤트 필터링의 능력에 영향을 끼치게 된다. LeSubscribe[5]는 단순하지만 대부분의 질의를 지원하는 프리디킷을 결합하는 $x\ \theta\ y$ 또는 x *contains* y 형태의 프로파일 정의어를 지원하고 있는데, x는 속성, y는 x의 도메인 값을 의미한다. 그 외에도 Siena[4], JEDI[8], Elvin4[9] 등이 (속성, 값)의 쌍으로 된 프로파일 정의어를 지원한다. Yeast[3]와 NiagaraCQ[10] 등의 ECA 패러다임에 기반을 둔 프로파일 정의어는 능동 데이터베이스(Active Databases)의 규칙(rule)에 기반을 둔다. 따라서 프로파일에는 이벤트(E)가 발생할 때, 평가되어야 할 조건(C)과 결과로서 취해져야 할 행위(A)들이 기술된다. 그 외에 최근의 몇몇 XML 기반의 연구들[6, 11, 12]는 XPath[13] 표현식(expression)을 이용하여 프로파일을 정의하기도 하며, A-MEDIAS[2]나 Jung[14]와 같이 별도의 프로파일 정의어를 제공하기도 한다. 이 방법은 XML 문서의 내용보다는 구조적인 매칭에 집중하기 때문에, 프로파일의 정의에 동등연산자('=')만을 제공하는 제한된 면을 보인다.

이벤트 및 프로파일의 표현에 있어서 (속성, 값) 쌍에 의한 표현은 단순한 반면, 너무 제한된 표현을 제공한다. 따라서 이 논문에서는 좀 더 많은 표현력을 제공할 수 있는 XML 기반의 이벤트 표현 모델과 ECA 패러다임에 기반을 둔 셀렉션 프리디킷의 연산자들(>=, >, <, <=, ==, <>) 및 함수 프리디킷을 수용하는 프로파일 정의 모델을 고려한다.

2.2 이벤트 필터링

이벤트 필터링은 시스템이 제공하는 필터 엔진의 시멘틱에 기반하고, 프로파일의 정의도 필터 엔진의 능력에 의해 제한된다. 따라서 프로파일 정의어와 필터의 시멘틱 사이에는 피할 수 없는 상충관계(trade-off)가 존재할 수 있다.

데이터베이스 기반의 필터링은 데이터로서 프로파일을 처리하기 때문에 이벤트와 프로파일의 매칭은 데이터베이스의 조인 질의에 의해 평가된다. 초기의 시스템들은 그룹 질의 최적화나 연속 질의 시스템의 문맥에서 필터링을 다루고 있다. Sellis는 공통된 계산을 공유하기 위해 유사한 질의의 그룹을 최적화하였고[15], NiagaraCQ[10]는 유사한 연속 질의를 그룹화하기 위해 시그내쳐(signature)를 사용하고 질의의 그룹

을 동시에 평가하기 위해 조인과 함께 질의로부터 추출된 상수(constant)들의 테이블을 사용하였다.

메모리 기반의 필터링은 메시지에 대한 필터로서 프로파일을 처리하기 때문에, 프로파일을 시스템의 메인 메모리에 유지한다. A-MEDIAS[2], SIFT[16]와 같은 시스템과 Aguilera[17]는 프로파일의 저장 및 이벤트 필터링을 위해 메모리 기반의 트리를 사용하였다. 또한, 몇몇 XML 기반 연구들[18, 19, 20]은 XPath 표현식으로 기술된 프로파일로부터 메인 메모리에 프로파일당 하나의 FSA(finite state automaton)를 생성한 후, 시스템이 이벤트 메시지를 수신하면, 이벤트 구동(event-driven) 방식의 파서가 호출되어 이벤트의 각 요소를 메모리에 저장된 FSA로 통과시키면서 이벤트의 구조가 프로파일과 매치하는지를 조사한다. FSA가 프로파일당 하나의 FSA를 유지하는 반면, YFilter[6]은 계산의 공유를 제공하기 위해 NFA(Non-deterministic Finite Automata)를 이용한 기법을 제안하였다. YFilter는 모든 프로파일에 대해 하나의 NFA만을 구성한 후, 첫 단계에서 NFA와의 구조적인 매칭만을 시도하고, 두 번째 단계에서 프리디킷에 대한 매칭을 시도한다.

데이터베이스 기반의 필터링 기법들은 조인 연산을 수행하기 위해 비싼 비용을 지불하는 대신 프로파일의 수에 따른 메모리의 크기에 크게 제한을 받지 않는다. 반면에, 메모리 기반의 필터링 기법은 프로파일의 수가 메모리의 크기에 제한을 많이 받기 때문에, 프로파일의 수가 많은 응용에는 적합하지 않다. 따라서 이 논문에서는 시스템이 미들웨어 수준의 서비스라는 것을 고려하여, 이벤트 필터링의 효율성을 높이기 위해 데이터베이스 기반의 필터링을 제안한다.

3. 이벤트 표현과 프로파일 정의 모델

센서 기반의 환경에서 다양한 소스로부터 발생한 이벤트는 특정한 형식을 가진 메시지로 표현되어 시스템으로 전달되고, 특정한 객체에 대한 관심은 프로파일로 정의되어 시스템에 저장된다. 따라서 시스템은 다양한 이벤트 소스를 정확하게 표현하고 전달할 수 있는 이벤트 모델과 객체에 대한 관심을 정확하게 정의할 수 있는 프로파일 정의어를 제공해야 한다.

3.1 XML 기반의 이벤트 표현 모델

사용자의 관심은 객체에 대해 어떠한 변화가 발생하는가에 있다. 그들은 자신을 유일하게 구분할 수 있는 식별자(identifier)에 의해 식별되며, 일정한 도메인을 가진 속성(attribute)에 의해 기술된다. 또한 객체는 속성 값에 의해 식별될 수 있는 상태(state)를 가지고 있으며, 일정기간 그 상태를 유지한다. 예를 들면, 어떤 연구실에 배치된 온도 센서가 18°C에서 20°C로 변화된 온도를 감지한 경우, 온도의 상태는 18°C를 지속하던 상태와 20°C로 변화된 두 가지의 상태가 존재한다.

이 때 객체에 대한 상태의 변화는 이벤트의 발생으로 간주될 수 있다[2]. 이벤트는 상태와는 달리 지속기간(duration)을 가지지 않지만, 그 발생을 구분할 수 있는 타임스탬프 T를 가진다. 이벤트의 실제 발생을 이벤트 인스턴스 e라고, 발생 시간을 t라 할 때, 하나의 이벤트 e는 $t(e)$로 나타내고, 이벤트 집합 E는 (식 1)과 같이 정의한다.

$$E = \{t_1(e), t_2(e), ..., t_n(e)\} \quad \text{(식 1)}$$

제1장에서 기술한 바와 같이, 이 논문에서는 물류 상황 정보를 관찰하기 위한 방법으로 RFID와 GPS의 두 가지 센서 기반 기술을 고려한다. 물류 차량에 대한 이동 정보는 GPS 위성으로부터 객체의 위치를 수신하는 GPS 기반 기술을 이용하고, 화물에 대한 상황 정보는 모든 화물에 RFID 태그를 부착하여 RFID 리더가 감지하는 신호를 객체에 대한 상황 정보인 이벤트로 식별하도록 한다. RFID 태그를 각 화물과 차량에 부착하고 RFID 리더를 차량과 창고에 배치한다고 가정할 때, 물류 환경에서 발생할 수 있는 이벤트의 유형 및 상태는 <표 1>과 같다.

RFID 리더가 감지한 신호 중, 화물에 부착된 RFID 태그의 감지 신호는 화물의 적재와 하역에 대한 “Loading”과 “Unloading” 상태를, 차량에 부착된 태그의 감지 신호는 차량의 출발과 도착에 대한 “Departure”, “Arrival” 상태를 가지는 “Detect” 이벤트로 인식된다. 한편, 차량에 부착된 GPS 수신 장치는 이동하는 위치에 대한 주기적인 정보를 갖는 이벤트로서 식별된다.

물리적인 환경으로부터 수집된 이벤트는 시스템으로 전달되기 전에 특정한 형식의 메시지로 표현되어야 하는데, XML은 이기종의 플랫폼 간에 상호운용성(interoperability)을 보장할 수 있는 API로서 가장 많이 사용된다. 또한 이벤트 표현 모델은 객체보다는 RFID 리더, 온도 센서, 습도 센서, GPS 장치와 같은 관찰자를 기반으로 모델링하는 것이 바람직하다[7]. 이것은 센서 기반의 환경에서의 이벤트는 물리적인 속성이나 객체에 대한 관찰 능력이 있는 센서에 의해 관찰되고, 하나의 센서는 다수의 객체에 대한 이벤트를 관찰할 수 있기 때문이다. 만약, 객체를 기반으로 모델링한다면, 이벤트 표현의 일반적이고도 유연한 프레임워크를 제한할 수 있다.

(그림 1)은 BNF[21] 구문으로 표현한 XML 기반의 이벤트 표현 모델이다. BNF 구문은 “*symbol* ::= *expression*” 형태로 심볼(symbol)을 묘사하기 때문에, XML 구조로 표현되는 모델을 설명하기가 용이하다. 각 심볼은 XML 구조의 요소에 대응된다.

(그림 1)에서 첫 번째 줄의 “Sensor ::= <ID><Observation>”

〈표 1〉 물류 환경에서의 이벤트의 유형 및 상태

이벤트	객체	유형 (Type)	상태 (State)	발생 시점
RFID	화물	Detect	Loading	화물이 차량에 실릴 때
			Unloading	화물이 차량으로부터 내려질 때
	차량		Departure	차량이 운송을 시작할 때
			Arrival	차량이 복귀했을 때
GPS		Location	-	차량이 이동 중일 때

```
Sensor ::= <ID><Observation>
ID ::= <StringLiteral>
Observation ::= <Type><Datetime>(<State><Tag> | <Position>)
Type ::= 'Detect' | 'Location'
State ::= 'Loading' | 'Unloading' | 'Departure' | 'Arrival'
Datetime ::= <StringLiteral>
Position ::= <Tag><Latitude><Longitude><Velocity><Course>
Tag ::= <EPC>{<EPC>}
EPC ::= <StringLiteral>
Latitude ::= <Degree><Minute><Second>
Longitude ::= <Degree><Minute><Second>
Velocity ::= <StringLiteral>
Course ::= <StringLiteral>
Degree ::= <StringLiteral>
Minute ::= <StringLiteral>
Second ::= <StringLiteral>
```

(그림 1) BNF로 표현한 XML 기반의 이벤트 표현 모델

```
<?xml version="1.0" encoding="UTF-8"?>
<Sensor>
   <ID>01.000000A.00000B.00000000D</ID>
   <Observation Type="Location">
      <Datetime>2006-11-01T13:00:00</Datetime>
      <Position>
         <Tag>
            <EPC>01.000000A.00000B.00000000F</EPC>
         </Tag>
         <Latitude>
            <Degree>36</Degree>
            <Minute>19</Minute>
            <Second>1.5</Second>
         </Latitude>
         <Longitude>
            <Degree>127</Degree>
            <Minute>24</Minute>
            <Second>2.07</Second>
         </Longitude>
         <Velocity>33.1</Velocity>
         <Course>120.1</Course>
      </Position>
   </Observation>
</Sensor>
```

(그림 2) 'Location' 이벤트에 대한 XML 기반의 메시지의 예

은 센서(<Sensor>)가 유일한 식별자(<ID>)를 가졌으며, 관찰(<Observation>)할 능력을 가지고 있다는 것을 의미한다. 세 번째 줄의 "Observation ::= <Type><Datetime>(<State><Tag>|<Position>)"은 관찰(<Observation>)된 각 이벤트들이 "<Type><Datetime><State><Tag>" 또는 "<Type><Datetime><Position>" 중의 하나로 구성된다는 것을 의미한다. 전자는 RFID 리더가 관찰(<Observation>)한 시간(<Datetime>)과 태그(<Tag>)를, 후자는 GPS 장치가 수신한 좌표값(<Position>)이 기술된다. 여기에서 <Type>과 <State>는 <표 1>의 이벤트 유형과 상태 중의 하나에 해당한다. "<EPC>"는 IBM의 Auto-ID Center에서 개발한 개별 상품을 식별할 수 있는 코드를 의미한다[22]. "<StringLiteral>"은 문자와 숫자가 결합된 문자열을 의미하는 것으로, <String-Literal>을 구성하는 추가적인 표현식의 기술은 생략한다. (그림 2)는 GPS 수신 장치에 의해 관찰된 객체(차량)가 이동하고 있는 위치와 속도, 방향등을 XML 구조로 기술한 예를 보여준다.

3.2 ECA 기반의 프로파일 정의 모델

프로파일이란 사용자에 의해 정의된 객체에 대한 관심을 의미한다. 프로파일을 정의함으로써, 사용자는 객체와 관련된 상황 정보를 얻을 수 있다. 사용자에 의해 정의된 객체

```
Profile ::= 'Event' <EventExpr>
            'Condition' < ConditionExpr>
            'Action' <ActionList>
EventExpr ::= 'Location' | 'Detect'
ConditionExpr ::= <SelectionList> [<LogicalOp> <Function>]
SelectionList ::= <Selection> {<LogicalOP> <Selection>}
Selection ::= <Literal> <ArithmeticOP> <Literal>
LogicalOP ::= 'AND' | 'OR'
ArithmeticOP ::= '>' | '>=' | '=' | '<>' | '<' | '<='
Function ::= <DistanceExpr> | <IntervalExpr>
DistanceExpr ::= 'Distance('<NumLiteral> ',' <NumLiteral> ','
                 <NumLiteral>')'
IntervalExpr ::= 'Interval(' <NumLiteral> ')'
ActionList ::= 'Send('<Protocol>, <StringLiteral>')'
               {', Send('<Protocol>, <StringLiteral>')'}
Protocol ::= 'SMS' | 'E-mail'
Literal ::= <StringLiteral> | <NumLiteral> | <DateLiteral>
NumLiteral ::= <FloatLiteral> | <IntegerLiteral>
```

(그림 3) BNF 구문으로 표현된 EAC 기반의 프로파일 정의 모델

에 대한 관심을 프로파일(p)이라 할때, 프로파일 집합 P는 (식 2)와 같이 정의한다.

$$P = \{p_1, ..., p_n\} \quad \text{(식 2)}$$

이벤트와 마찬가지로, 객체에 대한 관심을 정의하는 프로파일도 특정한 형식의 모델로 표현할 수 있다. 이 논문에서는 프로파일을 정의하기 위한 모델로서 능동 데이터베이스의 규칙에 사용되는 ECA(Event-Condition-Action) 패러다임에 기반을 둔 프로파일 정의 모델을 설명한다. 이 패러다임에서는 어떠한 객체에 대한 이벤트(E)가 발생하면, 평가되어야 할 조건(C)과 결과로 취해져야 할 액션(A)이 기술된다. 능동 데이터베이스의 규칙과 이벤트 통지 시스템의 프로파일의 차이점은 처리되는 데이터의 측면에 있다. 규칙은 삽입(insert), 갱신(update), 삭제(delete) 등의 데이터베이스 연산에 행해지는 반면, 프로파일은 시스템을 구성하는 컴포넌트들 사이에 교환되는 메시지(Message)에 기반을 둔다.

(그림 3)은 BNF 구문으로 표현된 ECA 기반의 프로파일 정의 모델이다. 첫 번째 줄은 프로파일이 ECA 패러다임에 기반하고 있음을 의미한다. 'Event'절의 <EventExpr>에는 <표 1>에서 구분한 이벤트 유형('Detect' 또는 'Location')을 기술한다. 'Condition'절의 <Condition Expr>에는 관심 있는 객체의 식별자가 무엇이며, 어떠한 속성과 값에 관심이 있는가를 기술한다. 'Action'절에는 평가된 결과를 누구에게 어떤 방법(프로토콜)으로 전달해야 하는지가 표현된다. 다섯 번째 줄의 "ConditionExpr ::= <SelectionList>[<LogicalOp><Function>]"은 프로파일 정의어가 관계연산(>=, >, <, <=, =, <>)에 필요한 셀렉션 프리디킷 뿐만 아니라, 함수로 구성된 프리디킷도 수용하고 있음을 보여준다. 만약, 조건절에 기술할 조건이 이벤트에 나타난 속성만으로 비교가 가능한 프리디킷이라면, '<SelectionList>' 부분에 기술한다. 반면에, 이벤트에 주어진 값을 기준으로 좀 더 복잡한 계산이 필요한 경우라면, '<Function>' 부분에 표현할 수 있다.

이 논문에서는 단순화를 위해 '<Function>'의 유형으로 Distance()와 Interval()만을 고려한다. Distance()는 이벤트가 관찰된 지점과 임의의 지점 사이의 거리를 계산하며, Interval()은 이벤트가 관찰된 시간과 임의의 시간 사이의

P1: 객체가 특정 지점(36.1234, 127.36)의 반경 2Km내로 진입하면 휴대폰(012-3456-7890)과 이메일(gdhong@dblab.cbnu.ac.kr)로 알려 주시오.

```
Event   Location
Condition EPC = '01.000000A.00000B.00000000E' AND
                Distance(36.1234, 127.36, 2)
Action Send('SMS', '홍길동', '012-3456-7890'),
       Send('E-mail', '홍길동', 'gdhong@dblab.cbnu.ac.kr')
```

P2: 객체가 이동하는 위치를 10분마다 휴대폰(012-3456-7890)으로 알려 주시오.

```
Event   Location
Condition EPC = '01.000000A.00000B.00000000E' AND
                Interval(10)
Action Send('SMS', '홍길동', '012-3456-7890')
```

(그림 4) 프로파일 정의의 예

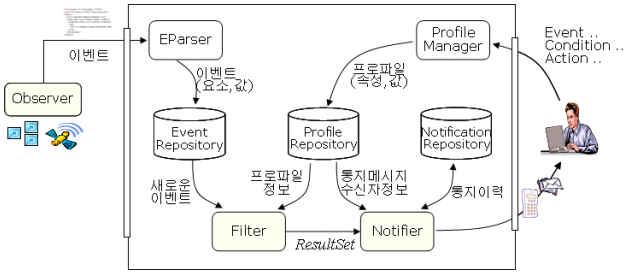
간격을 계산하기 위한 프로파일에서의 함수적 표현이다.

(그림 4)는 (그림 3)의 프로파일 정의 모델에 따라 프로파일을 정의한 예로, 사용자가 EPC의 식별자가 '01.000000A.00000B.00000000E'인 객체의 이동하는 위치에 관심이 있음을 보여주고 있다.

4. RFID와 GPS 기반의 이벤트 통지 시스템

수집되는 이벤트에 대하여 시스템 내에 저장된 프로파일과의 필터링을 수행하고, 사용자에게 이를 제공하는 것은 이벤트 통지 시스템의 역할이다. 일반적으로, 이벤트 통지 시스템은 정보의 제공자와 사용자를 연결하기 위해 Observer, Filter, Notifier의 기본적인 세 가지 모듈로 구성된다[2]. Filter는 이벤트와 프로파일의 필터링을 수행하는 가장 핵심이 되는 모듈이며, Notifier는 필터링의 결과를 이용하여 통지 메시지를 생성하고 전달한다. 그러나 일반적으로 Observer는 시스템 외부의 물리적인 환경으로부터 객체의 상황 정보를 수집하므로 시스템의 구조에는 포함되지 않는다. 그 외에, 제안하는 시스템에서는 XML 기반의 이벤트 메시지를 수신하는 EParser, 사용자가 프로파일을 쉽게 정의할 수 있도록 도와주는 Profile Manager등이 있다. (그림 5)는 이들 기본적인 모듈들을 바탕으로 한 시스템의 구조 및 각 모듈 사이의 데이터의 흐름을 보여주고 있다.

(그림 5)에서 보는 바와 같이 사용자의 관심은 Profile Manger를 통하여 프로파일로 정의되고 (속성, 값) 쌍으로 파싱되어 Profile Repository에 저장된다. 시스템 외부의 Observer를 통하여 수집된 이벤트 메시지가 EParser를 통하여 (요소, 값) 쌍으로 파싱되어 Event Repository에 저장되면, Filter는 이 이벤트 메시지와 사전에 정의된 프로파일과의 필터링을 수행한다. 프로파일과 매칭된 이벤트는 Notifier에 의하여 사용자에게 SMS 메시지 혹은 전자메일을 통하여 전달된다.



(그림 5) 제안된 이벤트 통지 시스템의 구조도

4.1 이벤트 메시지의 파싱

EParser는 시스템이 수신한 XML 기반의 이벤트 메시지를 (요소, 값)의 쌍으로 된 이벤트 요소 집합으로 파싱하고, 이벤트 메시지의 유형에 따라 이벤트 레파지토리인 *EREP*(Event REPository에 저장한다. 이 때, 이벤트 필터링의 효율성을 증가시키기 위해 RFID 기반의 이벤트(*EventDET*)와 GPS 기반의 이벤트(*EventLOC*)를 구분하여 저장하는데, 이것을 정의하면 (식 3)과 같다.

$$\begin{aligned} EventLOC &= \{EventID{:}v_i,\ EPC{:}v_i,\ Datetime{:}v_i,\ SensorID{:}v_i,\\ &\quad LatDeg{:}v_i,\ LatMin{:}v_i,\ LatSec{:}v_i,\ LongDeg{:}v_i,\\ &\quad LongMin{:}v_i,\ LongSec{:}v_i,\ Velocity{:}v_i,\ Course{:}v_i\} \\ EventDET &= \{EventID{:}v_i,\ EPC{:}v_i,\ Datetime{:}v_i,\ SensorID{:}v_i\} \end{aligned} \quad \text{(식 3)}$$

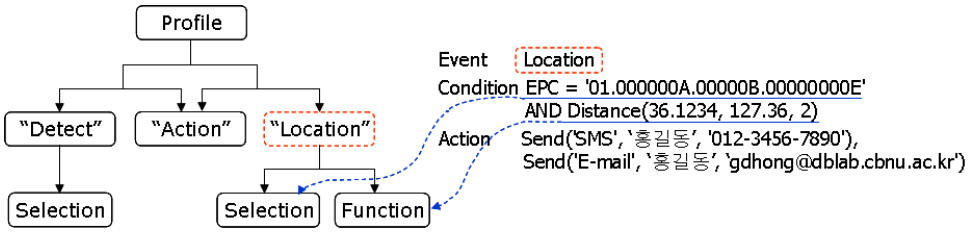
여기에서, v_i는 n개의 이벤트가 있다고 할 때, i번째 이벤트에서 각 요소의 값 v를 의미한다. 이 저장 구조는 이벤트의 유형에 따라 이벤트 식별자(*EventID*:v_i), 객체의 ID(*EPC*:v_i), 이벤트가 관찰된 시간(*Datetime*:v_i), 센서(RFID리더 혹은 GPS 장치)의 식별자(*SensorID*:v_i), 시·분·초로 이루어진 위도(*LatDeg*:v_i, *LatMin*:v_i, *LatSec*:v_i)와 경도(*LongDeg*:v_i, *LongMin*:v_i, *LongSec*:v_i) 속도(*Velocity*:v_i), 방향(*Course*:v_i) 등으로 구성된다.

4.2 프로파일의 관리

사용자가 프로파일을 쉽게 정의하고, Filter가 이벤트 필터링을 효율적으로 수행할 수 있도록 프로파일을 저장 관리하는 것은 Profile Manager의 역할이다. Profile Manager는 (그림 3)의 ECA 기반의 프로파일 정의 모델을 이용하여 정의된 프로파일 구문으로부터 (속성, 값)의 쌍으로 된 속성 집합을 생성하여 *PREP*(Profile REPository)에 저장한다. 이 때, 파싱된 프로파일의 속성 집합은 크게 조건과 액션의 두 부분으로 나뉘어지고, 조건 부분은 다시 이벤트와 마찬가지로 RFID 객체 기반의 프로파일(*ProfileDET*)와 GPS 객체 기반의 프로파일(*ProfileLOC*)로 구분되는데, 이를 정의하면 (식 4)와 같다.

$$\begin{aligned} ProfileLOC &= \{ProfileID{:}v_i,\ Selection{:}v_i,\ Function{:}v_i,\ Datetime{:}v_i,\\ &\quad Status{:}v_i\} \\ ProfileDET &= \{ProfileID{:}v_i,\ Selection{:}v_i,\ Datetime{:}v_i,\ Status{:}v_i,\} \end{aligned} \quad \text{(식 4)}$$

여기에서, v_i는 n개의 프로파일이 있다고 할 때, i번째 프로파일에서 각 요소의 값 v를 의미한다. 이 저장구조는 프로파일의 유형에 따라 프로파일 식별자(*ProfileID*:v_i), 조건의 셀렉션 프리디킷 부분(*Selection*:v_i)과 함수 프리디킷 부분(*Function*:v_i), 프로파일 정의 시간(*Datetime*:v_i), 상태값(*Status*:v_i,) 등으로 구성된다. (식 4)에서 정의된 바와 같이, 조건 부분은 이벤트 필터링의 성능을 향상시키기 위해 셀렉



(그림 6) 프로파일의 저장

```
Algorithm: Filter()
Input: EventType // EventType='Location' or 'Detect'
       EventID  // 이벤트의 식별자
       EREP // EREP={EventLOC_Set, EventDET_Set | EventLOC_Set
                ← a set of EventLOCs ∩ EventDET_Set
                ← a set of EventDETs}
       PREP // PREP={ProfileLOC_Set, ProfileDET_Set | ProfileLOC_Set
                ← a set of ProfileLOCs ∩ ProfileDET_Set
                ← a set of ProfileDETs}
       NREP // NREP={Notification_Set | Notification_Set
                ← a set of Notifications}
Output: ResultSet: 필터링의 결과

Method:
    ResultSet = {};

    // 1.셀렉션프리디킷의평가
    PSet ← {p'.ProfileID, p'.Selection}, where p'∈PREP corresponding to
           EventType;
    FOREACH (p ∈ PSet) {
        ESet = {e.EventID | e.EventID∈EventID ∩ p.Selection⊂e ∩ e∈EREP
               corresponding to EventType};
        IF (ESet ⊄ {})
            ResultSet[p.ProfileID] = "T";
    }
    IF (ResultSet ∈ {})
        return ResultSet;

    IF (EventType is "Detect")
        return ResultSet;

    //2.함수(Function)프리디킷의평가
    PSet ← {p''.ProfileID, p''.Function}, where p''∈PREP;
    FOREACH (p ∈ PSet) {
        IF ("Distance"⊂p.Function)
            pos ← Parse(p.Function);
            ESet ← {e.Pos_X, e.Pos_Y}, where e.EventID∈EventID ∩ e∈EREP
            IF (CompDistance(pos.X, pos.Y, ESet.X, ESet.Y) <= pos.Radius)
                ResultSet[p.ProfileID] = "T";
            ELSE
                ResultSet[p.ProfileID] = {};
        ELSE
            Minute ← Parse(p.Function);
            ESet ← e.Datetime, where e.EventID∈EventID ∩ e∈EREP
            NSet ← Max(Datetime_Set), where Datetime_Set={n.Datetime | n∈
                   EREP
                                        ∩ n.ProfileID p.ProfileID};
            IF (CompInterval(ESet.Datetime, NSet.Datetime) >= Minute);
                ResultSet[p.ProfileID] = "T";
            ELSE
                ResultSet[p.ProfileID] = {};
    }

    return ResultSet;
```

(그림 7) 이벤트 필터링 알고리즘

션 프리디킷으로 처리되어야 하는 부분과 함수 프리디킷으로 처리되어야 할 부분으로 나뉘어 각각 저장되는데, 이러한 구분은 물류 환경의 이벤트 소스의 특성상 GPS 객체 기반의 프로파일에만 해당되며 RFID 객체 기반의 프로파일은 셀렉션 프리디킷과 관련된 부분만이 존재한다.

(그림 6)은 (그림 4)에서 예시한 프로파일 P1를 레파지토리에 저장하는 예를 보여주고 있다. 이 예제에서 "EPC='01.000000A.00000B.00000000E'"은 셀렉션 프리디킷을 위해 저장되어야 할 문자열이고, "Distance(36.1234, 127.36, 2)"는 함수 프리디킷을 위해 저장되어야 할 문자열이다.

4.3 이벤트 필터링 기법

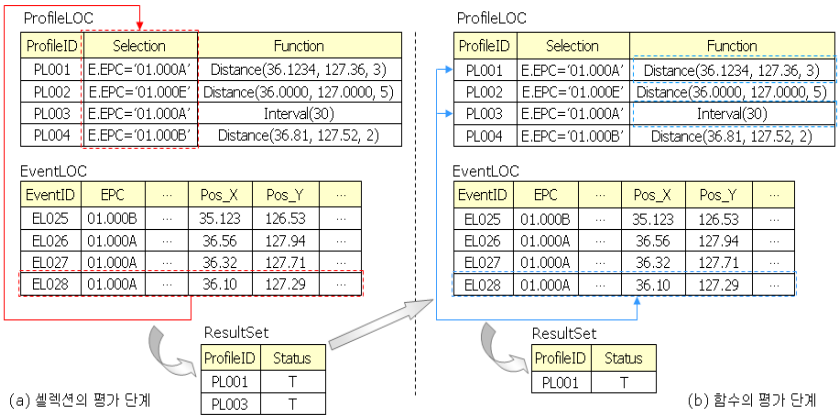
이벤트 통지 시스템은 사용자가 정의한 프로파일에 대하여 새롭게 발생한 이벤트와의 효율적인 필터링을 제공해야 한다. 제안된 시스템에서 Filter는 데이터베이스 기반의 필터링을 위해 셀렉션 프리디킷과 함수 프리디킷의 평가를 분리한다. 이것은 이벤트 필터링에서 셀렉션 프리디킷만으로 처리 가능한 조건을 1단계에서 미리 처리함으로써 필터링의 성능 향상을 의도하려는 것이다. 즉, 제안하는 이벤트 필터링 기법이 데이터베이스 연산에 기반하기 때문에, 프로파일의 저장시 셀렉션 프리디킷을 그대로 저장한다면 Psoup[23]에서와 마찬가지로 추가적인 연산 없이 데이터베이스가 제공하는 프리디킷을 이용할 수 있다는 것을 의미한다. 따라서 동등 연산자(==)를 포함하는 모든 비교 연산자(!=, >=, >, <, <=)를 지원하는 것이 가능해진다. 또한, 제안된 필터링 기법은 함수 프리디킷을 고려함으로써 추가적인 연산을 필요로 하는 환경에 적합한 유연성을 제공할 수 있다.

(그림 7)은 이벤트에 대하여 프로파일을 매치시키는 과정을 기술한 알고리즘이고, (그림 8)에서 이 알고리즘을 도식화하여 설명하고 있다. 이때 입력은 이벤트 메시지로부터 추출된 *EventID*와 *EventType*이며, 출력은 *ReusltSet*으로 이벤트 필터링을 만족하는 *ProfileID*와 *Status*를 포함한다. *ResultSet*의 데이터 구조는 (식 5)과 같다.

$$ResultSet = \{\text{ProfileID, Status}\} \quad \text{(식 5)}$$

셀렉션(selection) 프리디킷의 평가: 먼저 이벤트 레파지토리에 저장된 이벤트와 정의된 모든 프로파일의 셀렉션 프리디킷 부분을 비교하여 (그림 8)의 (a)에서와 같이 *ResultSet*에는 셀렉션 프리디킷이 참으로 평가된 프로파일들의 ID과 'T' 값이 저장된다. 만약 이 단계를 만족하는 프로파일이 존재하지 않는다면, Filter 모듈은 널(NULL) 값을 반환하고, 그 이벤트는 버려진다.

함수(Function) 프리디킷의 평가: 이 단계에서는 (그림 8)의 (b)와 같이 첫 번째 단계에서 참으로 평가되어 *ResultSet*에 있는 프로파일들만이 함수의 평가에 참여하고, 필요한 함



(그림 8) 이벤트 필터링의 과정

수들이 내부적으로 호출된다. 이 논문에서는 (그림 7)에 기술된 알고리즘에서 보는 바와 같이, 물류 환경에 적합한 이벤트 필터링을 위해 거리 기반 함수인 *CompDistance(posX1, posY1, posX2, posY2)*와 시간 기반 함수인 *CompInterval(time1, time2)*이 내부적으로 호출된다. 이 때, 거짓으로 평가된 프로파일의 ID가 *ResultSet*으로부터 삭제되고, *ResultSet*이 결과로서 반환된다.

4.4 통지 메시지의 전달

Notifier는 필터링의 결과(*ResultSet*)를 바탕으로 프로파일 레파지토리로부터 수신자 정보를 추출해내고, 통지 메시지(notification)를 생성하는 역할을 한다. 제안하는 시스템에서는 통지 메시지를 시스템뿐만 아니라, 휴대전화의 SMS(Short Message Service) 메시지나 전자메일(e-mail)을 이용하여 전달할 수 있으며, 그 결과는 (식 6)과 같이 통지 메시지 레파지토리인 *NREP*(Notification REPository)에 저장된다.

$$Notification = \{ProfileID{:}v_i,\ Sequence{:}v_i,\ Status{:}v_i,\ Datetime{:}v_i,\ Message{:}v_i\} \quad \text{(식 6)}$$

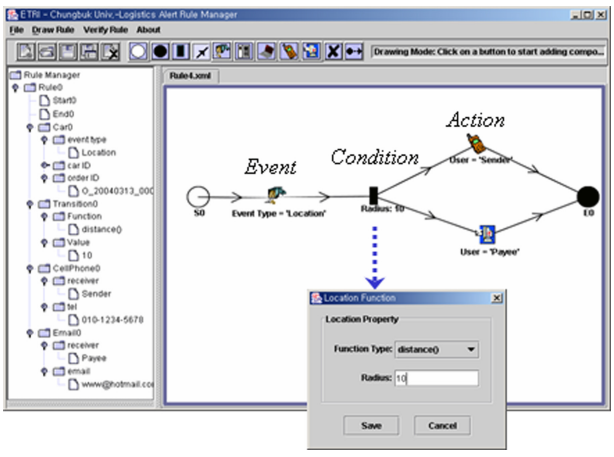
여기에서, v_i는 n개의 통지 메시지가 있다고 할 때, i번째 통지 메시지의 각 속성값 v를 의미한다. 이 저장구조는 프로파일 식별자(*ProfileID* : v_i), 프로파일 수신자의 순서번호(*Sequence* : v_i), 실패(0)와 성공(1)으로 구분되는 통지결과(*Status*:v_i), 통지 시간(*Datetime*:v_i), 통지 메시지(*Message*:v_i) 등으로 구성된다.

5. 시스템의 구현 및 평가

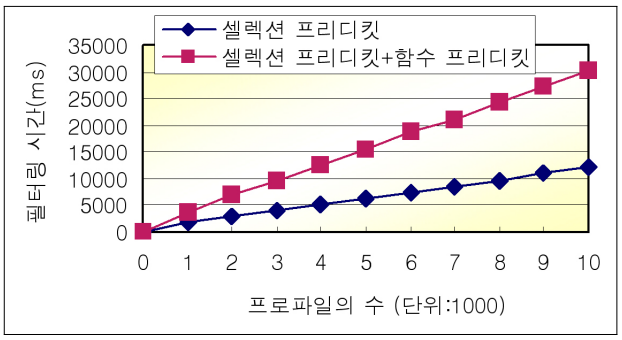
제안된 시스템은 제4장에서 제안한 시스템의 설계를 기반으로 JDK 1.5, DBMS로는 Oracle 10*i*를 사용하였으며, 펜티엄 IV 2.0GHz, RAM 1GB, Windows XP Professional을 기반으로 구현하였다. 또한, 시스템은 XML 구조를 가진 이벤트 메시지의 파싱을 위해 트리 기반의 파서보다는 상대적으로 가벼운 이벤트 기반의 SAX 파서를 사용하였다. XML 기반의 문서를 파싱할 때, 트리 기반의 파서는 문서를 탐색하기 위해 많은 양의 메모리 맵을 만드는 반면, 이벤트 기반의 파서는 문서의 시작과 끝, 요소의 시작과 끝, 텍스트와 같은 특성들을 이벤트로 인식하여 특정한 코드를 실행시킨다. 따라서 작은 크기의 데이터를 파싱할 때는 이벤트 기반의 파서를 사용하는 것이 시간과 공간 측면에서 효율적이다.

(그림 9)는 제 3장에서 제시한 ECA 기반 프로파일의 정의를 용이하게 할 수 있도록 구현된 프로파일 매니저(Profile Manager)이다. 오른쪽 패널은 사용자가 손쉽게 프로파일을 정의하고 수정할 수 있도록 구성된 그래픽 사용자 인터페이스이고, 왼쪽은 사용자가 정의한 프로파일의 목록 및 각 프로파일의 속성들을 트리 형태로 보여주는 속성 패널이다. ECA 기반 프로파일의 각 부분(E-C-A)에 대한 정의는 (그림 9)의 하단에 보이는 속성창과 같이 각 부분별 속성과 값을 정의할 수 있는 별도의 속성창들을 이용하여 정의할 수 있다.

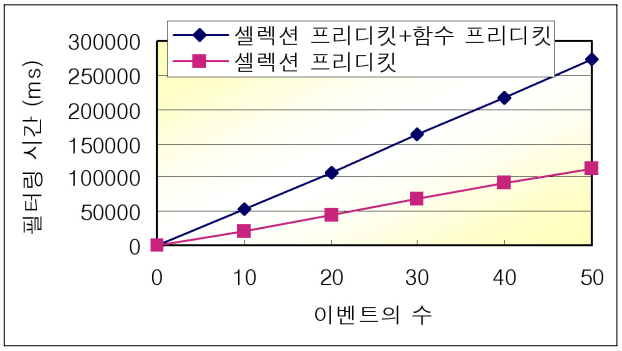
(그림 9)는 (그림 4)의 프로파일 P1을 정의한 예로, 사전에 저장되어 있는 화물의 목적지를 기반으로 사용자가 입력한 반경(radius) 값과의 거리를 계산하여 완전한 프로파일을 구성하게 된다. 정의된 프로파일들은 수정 및 삭제가 용이하도록 DOM 트리 기반의 XML 구조로 변환하여 관리된다. 또한, 제안된 시스템에서는 필터링된 결과(notification)가 SMS 메시지로도 통지될 수 있도록 하여 좀 더 빠르게 사



(그림 9) Profile Manager



(그림 10) 셀렉션 및 함수 프리디킷에 대한 필터링 비교



(그림 11) 이벤트의 증가에 따른 필터링 시간 비교

용자가 원하는 정보를 전달 받을 수 있도록 하였다.

(그림 10)은 프로파일의 수를 1000 단위씩 증가시켰을 때 셀렉션 프리디킷과 함수 프리디킷 기반의 이벤트에 대한 필터링 시간을 보여준다. 평가를 위해 이벤트는 모든 프로파일을 만족하도록 하였으며, 프로파일은 셀렉션 프리디킷과 함수 프리디킷에 대한 조건을 모두 가진 GPS 기반의 프로파일을 사용하였다. (그림 10)에서 볼 수 있듯이, 함수 프리디킷을 평가하는데 걸리는 시간이 셀렉션 프리디킷을 평가하는데 걸리는 시간의 두배 정도인 것을 볼 수 있다.

(그림 11)은 이벤트의 수를 10단위로 증가시킬 때, 셀렉션 프리디킷과 함수 프리디킷 기반의 이벤트의 필터링 시간을 보여준다. 마찬가지로, 이벤트는 모든 프로파일을 만족하도록 실험을 진행하였다. 실험의 결과는 함수 프리디킷 기반의 프로파일의 필터링 시간이 셀렉션 프리디킷 기반의 프로파일의 필터링 시간의 두 배이상 증가함을 볼 수 있다. 이것은 이벤트 필터링에서 함수 프리디킷을 평가하기 위한 시간이 셀렉션 프리디킷을 평가하는 시간보다 훨씬 많은 시간이 소요된다는 것을 의미한다.

따라서 셀렉션 프리디킷만으로 구성된 프로파일이 다수 존재한다면 프리디킷의 평가를 두 단계로 구분하는 알고리즘은 필터링 시간의 측면에서 훨씬 더 좋은 성능을 보일 수 있다.

<표 2>는 이 논문에서 제안하는 RFID와 GPS 기반의 이벤트 통지 시스템과 이전의 A-MEDIAS[2], ELvin4[9], SIFT[16] 등의 이벤트 통지 시스템들을 이벤트, 프로파일, 서버, 통지 방법으로 나누어 비교한 것이다.

ELvin4, SIFT, A-MEDIAS 등은 구체적인 이벤트 표현 모델을 제시하지 않고 있으며, 와 제안된 시스템만이 별도의 프로파일 정의 모델을 제시하고 있다. 프로파일 정의시 지원되는 연산자로는 ELvin4가 동등(==), 비동등(!=), 논리연산자(AND, OR, NOT) 및 문자에 기반한 비교 연산자들을 지원하고 있으며, SIFT와 A-MEDIAS는 동등 연산자만을 지원한다. 이에 반해, 제안된 시스템은 비교연산자(==, !=, >=, >, <, <=) 및 시간과 거리를 계산할 수 있는 함수연산들을 제공하여 물류 환경에서의 위치에 관한 상황정보를 인식할 수 있도록 한다.

서버 측면에서도 다른 시스템들은 트리기반의 알고리즘을 제공하는 반면, 제안된 시스템은 데이터베이스 연산에 기반을 두고 있으므로 프로파일이 셀렉션 프리디킷 만으로 구성

〈표 2〉 다른 이벤트 통지 시스템들과의 비교

구분		ELvin4	SIFT	A-MEDIAS	제안된 시스템
이벤트	표현모델	-	-	-	XML 기반
	데이터	구조화된 문서	텍스트 문서	온도,A/C 동작	RFID(객체), GPS(위치)
프로파일	정의모델	(속성,값)쌍	Keyword 기반	별도의 프로파일 모델	ECA 기반의 프로파일 모델
	지원 연산자	동등, 비동등, 논리연산자 문자비교연산자	동등연산자	동등연산자	비교연산, 함수연산(거리,시간)
서버	알고리즘	트리 기반	트리 기반	트리 기반	데이터베이스 기반
	Observer	X	X	X	X
	Filter	○	○	○	○
	Notifier	○	○	○	○
통지 방법		단순한 라우팅	전자메일	시스템	전자메일, SMS

된 경우 데이터베이스가 제공하는 프리디킷을 그대로 이용할 수 있도록 하였다.

6. 결 론

유·무선 통신기기 및 센서 기술의 발달로 인해, 시스템들은 지능화된 사물로부터 실시간으로 객체의 상황 정보를 수집하는 것이 가능해졌다. 마찬가지로, 물류 환경에서도 GPS나 RFID와 같은 센서 기반 기술을 이용한다면, 실시간으로 객체에 대한 상황 정보를 수집하고, 사용자가 원하는 정보를 전달할 수 있다. 따라서 이 논문에서는 변화된 환경에서 실시간으로 수집되는 물류 환경의 상황 정보를 분석하여 서비스할 수 있는 이벤트 통지 시스템을 제안하였다.

첫째, 이벤트 메시지의 유연한 표현을 위해 센서 중심의 XML 구조를 가진 이벤트 표현 모델을 제시하였고, 셀렉션 프리디킷과 함수 프리디킷을 모두 필터링의 조건으로 기술할 수 있는 ECA 기반의 프로파일 정의어를 제시하였다. 둘째, 데이터베이스 기반의 2단계 필터링 기법을 제시하였다. 1단계에서는 이벤트에 포함된 속성만으로 조건을 평가하는 셀렉션 부분에 대한 필터링을 수행한다. 이때, 제안한 필터링 기법이 데이터베이스 연산에 기반을 두고 있기 때문에 별도의 추가적인 연산 없이 모든 비교연산자(==, !=, >=, >, <, <=)를 수용할 수 있다. 2단계에서는 셀렉션 프리디킷의 평가 결과가 참으로 평가된 프로파일을 대상으로 함수 프리디킷에 대한 필터링을 수행한다. 이와 같이 함수 프리디킷의 필터링을 분리시킴으로써, 프로파일이 셀렉션 프리디킷만으로 구성된 경우 필터링에 소요되는 시간을 좀 더 단축시킬 수 있다. 또한, 물류 환경에서는 함수 프리디킷을 이용하여 상황 정보를 실시간으로 추적하는 것이 가능해진다.

제안된 이벤트 통지 시스템은 변화된 물류 환경뿐만 아니라, 지능형 교통관리 시스템, 통합 물류관리 시스템, 국가 재난 관리 시스템, u-의료 시스템 등과 같은 USN 기반의 여러 분야에 광범위하게 적용될 수 있다.

참 고 문 헌

[1] 전성우, 김기학, 구훈영, 허홍식, 박종홍. “RFID 기반 실시간 우편물류 기술,” 전자통신동향분석, 제22권 3호. 2007, 06.

[2] A. Hinze, “A-MEDIAS: Concept and Design of an Adaptive Integrating Event Notification Service,” PhD Thesis, Freie Universitt Berlin, Institute of Computer Science, 2003, 12.

[3] B. Krishnamurthy and D. S. Rosenblum, “Yeast: A General Purpose Event-Action System,” IEEE Transactions on Software Engineering, Vol.21, No.10, pp.845-857, 1995, 10.

[4] A. Carzaniga, “Architectures for an Event Notification Service Scalable to Wide-area Networks,” PhD Thesis, Politecnico di Milano, Dipartimento di Elettronica e Informazione, 1998.

[5] J. Pereira, F. Fabret, H. Jacobsen, et. al, “LeSubscribe: Publish and subscribe on the web at extreme speed,” In Proceedings of ACM SIGMOD, Santa Barbara, CA, 2001.

[6] Y. Diao, M. Altinel, M. J. Franklin, and H. Zhang, P. M. Fischer, “Path Sharing and Predicate Evaluation for High-Performance XML Filtering,” ACM Transactions on Database Systems (TODS), Vol.28, No.4, pp.467-516, 2003, 12.

[7] C. Floerkemeier, D. Anarkat, T. Osinski, and M. Harrison, “PML Core Specification 1.0,” White Paper. STG-AUTOID-WH-005, IBM Auto-ID Center, 2003.

[8] G. Cugola, E. D. Nitto, and A. Fuggetta, “Exploiting an Event-based Infrastructure to Develop Complex Distributed Systems,” In Proceedings of ICSE 98, Kyoto, Japan, 1998.

[9] B. Segall, D. Arnold, Julian Boot, Michael Henderson, and Phelps, “Content Based Routing with Elvin4,” In Proceedings of AUUG2K, 2000, 07.

[10] J. Chen, D. DeWitt, F. Tian, and Y. Wang, “NiagaraCQ: A scalable continuous query system for internet databases,” In Proceedings of ACM SIGMOD, Dallas, TX, 2000.

[11] F. Tian, B. Reinwald, and H. Pirahesh, T. Mayr, and J. Myllymaki, “Implementing A Scalable XML Publish/Subscribe System Using Relational Database Systems,” In Proceedings of ACM SIGMOD, 2004.

[12] M. Altinel and M. J. Franklin, “Efficient Filtering of XML Documents for Selective Dissemination of Information,” In Proceedings of VLDB, pp.53-64, 2000.

[13] W3C, “XML Path Language(XPath) 2.0,” Available at http://www.w3.org/TR/xpath, 2007.

[14] Y. J. Jung and K. H. Ryu, “The Vehicle Tracking System for Analyzing Transportation Vehicle Information,” In Proceedings of APWeb Workshops 2006, pp.1012-1020, 2006, 01.

[15] T. K. Sellis, “Multiple Query Optimization,” ACM Transactions on Database Systems (TODS), Vol.13, No.1, pp.23-52, 1988, 03.

[16] T. W. Yan and H. Garcia-Molina, “The SIFT information dissemination system,” ACM Transactions on Database Systems (TODS), Vol.24, No.4, pp.529-565, 1999, 02.

[17] M. K. Aguilera, R. E. Strom, D. C. Sturman, et al, “Matching Events in a Content-based Subscription System,” In Proceedings of PODC, 1999.

[18] M. Altinel, and M. J. Franklin, “Efficient Filtering of XML Documents for Selective Dissemination of Information,” In Proceedings of VLDB, pp.53-64, 2000.

[19] T. J. Green, G. Miklau, M. Onizuka, and D. Suciu, "Processing XML Streams with deterministic automata," In Proceedings of ICDT, 2003.
[20] F. Peng, and S. S. Chawathe, "XPath Queries on Streaming Data," In Proceedings of ACM SIGMOD, 2003.
[21] J. W. Backus, F. L. Bauer, J. Green, C. Katz, J. McCarthy, A. J. Perlis, H. Rutishauser, K. Samelson, B. Vauquois, J. H. Wegstein, A. van Wijngaarden, M. Woodger, and P. Naur (ed.), "Revised report on the algorithmic language Algol 60," Communications of the ACM Vol.6, No.1, pp.1-17, 1963.
[22] EPCglobal, "EPC™ Generation 1 Tag Data Standards Version 1.1 Rev.1.27," EPCglobal Inc™, 2005, 10.
[23] S. Chandrasekaran and M. J. Franklin, "PSoup: A System for Streaming Queries over Stream Data," The VLDB Journal, Vol.12, No.2, pp.140-156, 2003.

이 용 미

e-mail : ymlee@dblab.chungbuk.ac.kr
2002년 충북대학교 컴퓨터과학과(이학사)
2005년 충북대학교 전자계산학과 (이학석사)
2005~현 재 충북대학교 전자계산학과 박사과정
관심분야 : 시공간 데이터베이스, 스트림 데이터 처리, 데이터 마이닝

남 광 우

e-mail : kwnam@kunsan.ac.kr
1995년 충북대학교 컴퓨터과학과(이학사)
1997년 충북대학교 전자계산학과(이학석사)
2001년 충북대학교 전자계산학과(이학박사)
2001~2004년 ETRI 텔레매틱스·USN연구단 선임연구원
2004년~현 재 군산대학교 컴퓨터정보공학과 조교수
관심분야 : LBS, geoSensor Network, 이동체 데이터베이스, GIS

류 근 호

e-mail : khryu@dblab.chungbuk.ac.kr
1976년 숭실대학교 전산학과(이학사)
1980년 연세대학교 공학대학원 전산전공 (공학석사)
1988년 연세대학교 대학원 전산전공 (공학박사)
1976~1986년 육군군수 지원사 전산실(ROTC 장교), 한국전자통신연구원(연구원), 한국방송통신대 전산학과 (조교수) 근무
1989~1991년 University of Arizona Research Staff(TempIS 연구원, Temporal DB)
1986년~현 재 충북대학교 전기전자컴퓨터공학부 교수
관심분야 : 시공간 데이터베이스, Temporal GIS, 지식기반 정보검색 시스템, 유비쿼터스 컴퓨팅 및 스트림 데이터 처리, 데이터 마이닝, 데이터베이스 보안, 바이오인포매틱스