유비쿼터스 환경을 위한 위치 기반 이벤트 서비스

김선욱° 윤희용
성균관대학교
{sunux°, youn}@ece.skku.ac.kr

Location-based Event Services for ubiquitous environment

Sun Uk Kim0 ,Hee Young Youn
School of Information and Communications Engineering
Sungkyunkwan University, Suwon, Korea

요 약

유비쿼터스 환경으로 접어들면서 분산 컴퓨팅 기술의 발전과 더불어 무선통신 기술의 발달이 가속화 되고 있다. 무선통신 기술의 발전으로 분산 컴퓨팅 기술에서 이동성을 지닌 객체의 위치 정보에 대한 서비스 지원 문제가 이슈로 떠오르고 있다. 하지만 기존의 분산 컴퓨팅 미들웨어는 이동 객체의 변화하는 위치에 적응적인 정보 제공 서비스가 미약했다. 본 논문에서는 이러한 부분에 대한 하나의 해결 방법으로 위치 기반 분산 컴퓨팅 기술을 제안한다. 제안된 구조는 기존의 이벤트 서비스에서 위치 정보를 필터링 할 수 있는 모듈을 구현함으로써 기존의 이벤트 서비스보다 사용자의 위치 상황에 유동적으로 정보를 제공할 수 있도록 한다. 성능평가는 시뮬레이션 툴을 이용하여 위치 정보를 발생시키고 그 정보들을 이용하여 위치 필터링을 통과하여 두 객체의 데이터 통신에 중점을 두었고 데이터 크기 및 Supplier 수의 변화에 따른 메시지의 지연시간을 측정, 평가 하였다.

1. 서 론

널리 알려진 유비쿼터스 환경이 우리 생활에 점점 가까워지면서 분산 컴퓨팅 기술의 중요성이 더욱 증가하고 있다. 또한 무선통신 기술의 발달로 인해 이동성을 지닌 객체들의 위치 정보 또한 분산 컴퓨팅 기술의 중요한 요소로 부각되고 있는 추세이다. 이러한 무수히 늘어나는 객체들 간의 통신을 위해서는 미들웨어가 필요할 것이다. 그리고 객체들의 이동 정보에 따라 제공되는 서비스들도 틀려질 것이다. 따라서 위치 정보를 기반으로 각 객체들에 대한 서비스를 제공하는 미들웨어가 절실히 요구되어 지고 있다.

대용량 통신 처리와 비동기 통신, 고가용성으로 인하여 메시지 기반 미들웨어(MOM: Message-oriented Middleware)는 유비쿼터스 환경에서 많이 쓰이고 있는 미들웨어의 한 형태이다. 이러한 메시지 기반 미들웨어의 대표적인 것에는 Microsoft의 MSMQ, IBM의 WebSphere, 자바 기반의 JMS, OMG의 CORBA 이벤트 서비스, CORBA Notification 서비스 등이 있으며 각 종 학회를 통하여 다양한 미들웨어 들이 나타나고 있다[1][2][3][4]. 그리고 이동성을 지원하는 하나의 방법으로 위치 기반 서비스(Location-based Services)도 다양하게 연구되고 있다[5].

하지만 앞에서 언급한 여러 미들웨어는 사용자의 위치에 대한 적응적인 서비스를 고려하지 않고 동작한다. 이러한 기존의 MOM 미들웨어는 사용자의 위치 환경을 고려하지 않은 정보들까지, 즉 현재의 위치에서 사용자에게 불필요한 정보를 제공하는 문제점이 있다.

본 논문에서는 이러한 문제점을 해결하고자 위치 기반 이벤트 서비스를 제안한다. 우선 이동 객체에 대한 위치 정보를 알아내기 위해 일반적으로 많이 쓰이는 GPS를 이용한다. GPS는 사용자의 위치를 구할 때 정확도가 떨어진다는 단점이 있지만 수신기 하나로 비교적 사용자의 위치를 쉽게 구할 수 있다는 장점을 가지고 있다. GPS로부터 수신된 위치 정보를 이벤트 채널에 함께 전달하면 미들웨어의 내부에서는 그 정보를 필터링 할 수 있는 모듈을 이용하여 위치 정보를 필터링 한다. 이렇게 필터링 된 정보를 바탕으로 사용자의 위치를 파악하여 보다 나은 서비스를 사용자에게 제공할 수 있게 해준다.

논문의 구성은 서론에 이어 2 장에서는 위치 정보에 사용되고 있는 GPS와 MOM 미들웨어에서 자주 쓰이는 COS 이벤트 서비스에 대해 알아보고, 3 장에서는 본 논문에서 제안 하는 위치 기반 이벤트 서비스에 대한 방법을 소개한다. 그리고 4 장에서는 간단한 시뮬레이션 툴을 이용하여 JMS와 Information Bus와의 성능 결과를 보여준 뒤 마지막 5 장에서는 본 논문의 결론과 향후 연구 과제에 대해서 논하도록 한다.

2. 관련 연구

2.1 GPS(Global Positioning System)

GPS[6]는 위성을 이용한 항법지원시스템으로 3차원 위치정보 및 정확한 시간정보를 제공한다. GPS는 지구 전역에서 날씨에 영향 없이 24시간 사용 가능하며, 일반에게 공개된 표준 측위 서비스는 약 10m 이내의 오차를 갖는 정밀한 위치정보를 획득할 수 있다.

기본원리 및 구성: 눈에는 보이지 않지만 우리 머리 위로는 인공위성이 여러대 떠 있다. 그중에는 미 국방성에서 개발한 GPS위성이 잇는 전 지구를 총 24개의 위성으로 감싸고 있다. 이는 지구 어디에서나 네 개 이상의 위

성을 수신할 수 있도록 설계된 것이다. 일반적으로 위성이 네 개 이상이 수신되어야 현 위치를 계산해 낼 수 있는데 이는 삼각 측량의 원리와 비슷하다. 2차원 상에서 삼각 측량법은 위치를 알고 있는 두 점을 각각 a와 b라 하고 미지의 한점을 X라고 했을 때 a, b의 위치, 이 두 점과 X 사이의 거리를 이용해 미지의 점 X의 위치를 구하는 방법이다.

3차원 상에서는 위치를 알고 있는 3개의 점이 필요한 것이고 이에 해당하는 것이 GPS위성이다. 그리고 우리가 알고자 하는 위치가 미지의 한 점이 되면 위성과 미지의 점 사이의 거리정보를 제공하는 것이 GPS기술이다. 간단하게 생각하면 위성의 정확한 위치를 알고 다른 세 위성으로부터의 거리를 정확히 알면 자신의 위치를 계산할 수 있다는 것이다. 나머지 한 대는 시각을 동기화 하는데 사용된다. 아래의 그림 1은 위치 계산을 단계별로 간략하게 도식화 하였다.

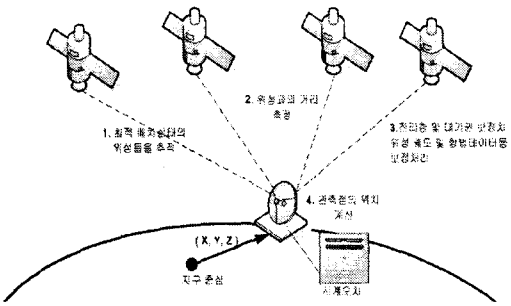

그림 1 위치 계산의 단계

동작 원리: GPS는 위성에서 보내주는 항법 메시지를 받아 수신기에서 위치를 계산함으로써 위치를 파악하는 위성 항법 시스템이라 말할 수 있다. 수신기에서는 자신이 받은 신호가 24개의 위성 중 어떤 위성으로부터 왔는지를 알아야 하는데, 각 위성에서 모두 똑같은 주파수로 각자의 데이터를 실어 보내기 때문에 주파수로 구분이 불가능하다.

따라서 각 위성마다 독립적인 ID코드를 할당하여 발신 위성을 파악하는데, 이러한 방식이 CDMA(Code Division Multiple Access)방식이다. 수신기는 위성으로부터의 신호가 들어오면 가지고 있는 모든 위성의 ID중 서로 맞는 ID를 찾아낸다. 이렇게 위성으로부터 온 신호인지를 파악하게 되면 항법 데이터를 얻게 되고 위치를 계산하게 된다.

GPS 통신 프로토콜: 위성으로부터의 시차와 거리 등이 모든 것은 사실 GPS 수신기에서 알아서 다 해주고 우리에게 NMEA(Nation Electronics Association)[7]에서 제안한 데이터 포맷으로 결과를 볼 수 있다. NMEA에서 제안한 데이터 포맷으로 실제 GPS에서 수신된 내용을 1초 내지 수초마다 한번씩 텍스트 형태로 볼 수 있다. GPS 수신기의 설정에 따라 수신 받을 수 있는 정보들이 있다. 아래의 그림 2는 몇 가지 수신정보의 예이다.

```
$GPGSA,A,3,04,05,,09,12,,,24,,,,,2.5,1.3,2.1*39
$GPGSV,2,1,08,01,40,083,46,02,17,308,41,12,07,344,39,14,22,228,45*75
$GPGGA,123519,4807.038,N,01131.000,E,1,08,0.9,545.4,M,46.9,M,,*47
$GPRMC,123519,A,4807.038,N,01131.000,E,022.4,084.4,230394,003.1,W*6A
$GPGLL,4916.45,N,12311.12,W,225444,A,*31
$GPRMB,A,0.66,L,003,004,4917.24,N,12309.57,W,001.3,052.5,000.5,V*20
```

그림 2 GPS 수신 정보

수신된 정보들 중 본 논문에서 사용하는 정보는 GPRMC라는 항목이다. GPRMC 항목은 Recommended Minimum Specific GNSS Data의 약자로 네비게이션에서 일반적으로 필요한 데이터 항목을 모두 갖추고 있다.

표 1 GPRMC 데이터 형식

	필드	예	설명
1	Sentence ID	$GPRMC	Recommended Minimum Specific GNSS Data
2	UTC Time		hhmmss.sss
3	Status	A	A=Valid, V=Invalid
4	Latitude	3558.0468	ddmm,mmmm
5	N/S indicator	N	N=North, S=South
6	Longitude	12657.5067	ddmm,mmmm
7	E/W indicator	E	E=East, W=West
8	Speed over ground	0	Knots
9	Course over gorund	0	Degrees
10	UTC Date	240303	DDMMYY
11	Magnetic variation	-	Degrees
12	Checksum	*17	-
13	Terminator	CR/LF	

위의 표 1에서 나와 있는 GPRMC의 2번 'UTC TIME'으로 위성에서 보낸 시간으로 아주 정확한 시간을 나타낸다. 3번은 A와 V로 표시 하는데 V일 경우 위치정보를 신뢰할 수 없는 경우이다. 4번과 6번은 경도와 위도를 나타낸다. 일반적으로 경위도에는 도/분/초를 사용한다. 6번과 7번은 동경과 서경, 북위와 남위를 나타내므로 우리나라에서는 N, E밖에 볼 수 없다. 8번은 속도이다. NMEA가 해양용이라 노트(Knots)단위로 되어 있다. 10번은 날짜 항목과 같이 계산하면 완전한 날짜와 시간을 얻을 수 있다.

2.2 CORBA 이벤트 서비스

CORBA 이벤트 서비스는 OMG(Object Management Group) 의해 정의 및 표준화된 분산 객체 컴퓨팅 미들웨어 명세서이다[8]. 유비쿼터스 환경에서 사용되는 수많은 분산 애플리케이션들은 이벤트 기반 기법을 이용한 비동기 통신을 필요로 한다[9,10]. 기본적으로 CORBA 모델은 하나의 Supplier와 Consumer가 동기적

(Synchronous)으로 연결되는 형태인 반면 CORBA 이벤트 서비스는 다중 응용 프로그램간의 비동기적인 (Asynchronous) 통신을 지원한다.

일반적으로 CORBA에서는 ORB를 통해 원격 객체 호출(Remote object invocation)을 하는데 이 개념은 전통적인 RPC(Remote Procedure Call)와 유사하지만 중요한 차이점은 일반적으로 전통적인 RPC가 허용하는 것보다도 사용되는 규모가 다양하고, 프로세스보다는 객체로서 전달된다는 것이다.

아래의 그림 3은 이벤트 서비스에서 Suppliers와 Consumers 사이의 이벤트 통신을 보여준다. 이벤트 서비스에서의 통신 방법에는 Push 모델과 Pull 모델, 두 가지 방법이 있다.

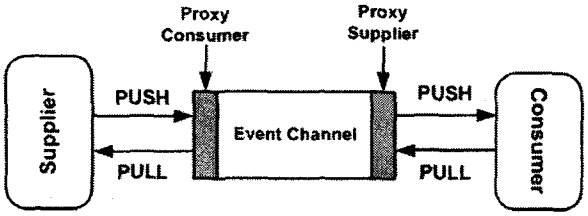

그림 3 CORBA 이벤트 서비스 구조

Push 모델: 이벤트의 Supplier가 Consumer들에게 이벤트 데이터 전송을 초기화 시킨다. 이러한 능동적인 Supplier는 자신이 보내고자 하는 곳의 채널에 데이터를 전송하면 이벤트 채널은 수동적인 Consumer들에게 데이터를 전송하는 방식이다.

Pull 모델: Consumer가 Supplier로부터 이벤트를 요구하는 것을 말한다. 능동적인 Consumer들은 이벤트 채널을 통해 수동적인 Supplier로부터 데이터를 전송 받을 수 있는 방식이다.

이벤트 채널: 이벤트 채널은 Supplier와 Consumer의 연결 시켜주는 중개자 역할을 담당한다. Supplier는 채널에 데이터를 전송하고 자신은 다른 작업을 계속 수행할 수 있다. 또한 이벤트 채널은 Supplier와 Consumer의 객체 주소를 관리하는데 채널의 양 끝단에는 proxy supplier와 proxy consumer가 존재한다. proxy consumer는 실제 supplier와의 연결을 지원하기 위해 사용되며, proxy supplier는 실제 consumer와의 연결을 지원하기 위해 사용된다.

이벤트 서비스의 통신 모델은 이벤트의 전달 연산자와 인자의 형에 따라 두 가지로 구분할 수 있다. 일반 모델의 경우에는 모든 통신이 이벤트 데이터를 위한 하나의 인자를 가진 일반 push()와 pull() 연산자에 의해서 수행된다. 형 모델의 경우에는 통신이 OMG IDL()에 정의된 인터페이스 연산자에 의해서 이루어진다.

3. 제안된 위치 정보를 이용한 이벤트 서비스

3.1 동기

이동성을 지닌 객체가 주를 이루는 유비쿼터스 환경에서 객체의 위치에 따라 제공되는 정보가 달라진다. 기존의 이벤트 서비스는 채널 기반으로 데이터를 제공한다. 이는 항상 변화하는 이동하는 객체의 상황에 적응적인 정보를 제공하는데 문제점이 있다. 그래서 본 논문에서는 이동객체가 이벤트 서비스를 위치에 따라 좀 더 적응적으로 서비스를 제공하는 방법을 제시한다.

3.2 전체적인 위치 기반 이벤트 서비스 구조

본 논문에서 제안하는 방법은 그림 2와 같다.

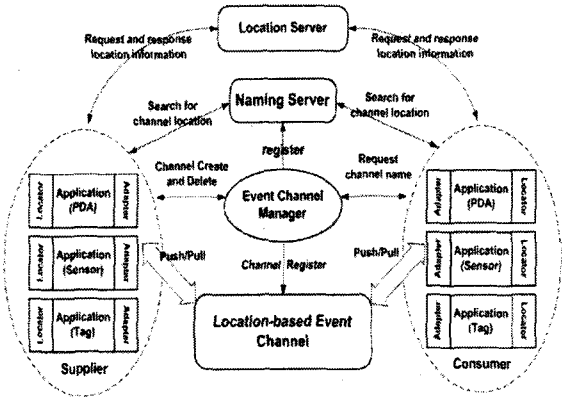

그림 4 위치기반 이벤트 서비스 구조

위의 그림에서 Supplier와 Consumer는 이동성, 비이동성을 지닌 모든 객체(PDA, Sensor, Tag 등)가 다 포함된다. 각각의 Supplier와 Consumer에는 Adapter 모듈과 Locator 모듈이 존재한다. Adapter 모듈은 이벤트 채널을 생성 및 삭제하고 Supplier와 Consumer간 데이터를 주고받기 위한 효율적인 인터페이스를 제공한다. Locator 모듈은 Location Server에서 제공되는 위치 정보를 GPS 수신기를 통해 위치 정보를 텍스트 형식으로 제공 받은 뒤 그 정보에서 사용자에게 필요한 위치 정보를 추출하기 위해 존재한다. 이렇게 추출된 위치 정보는 채널에 데이터가 들어가기 직전 Location-based Event Channel 내에 존재하는 Location Filtering Module에 의한 필터링에 사용된다. Naming Server는 채널들의 이름과 주소 값을 가지고 있어서 Supplier와 Consumer는 채널 이름을 가지고 정보를 요청하게 된다. 이때 Server는 그 이름을 가지고 Naming Server내에 존재하는 이름을 찾아 그에 상응하는 주소 값을 전달해 주게 된다. Event Channel Manager는 실제적인 이벤트 채널의 생성과 삭제를 담당하며 Naming Server에 생성되는 채널들의 이름과 주소를 등록하는 역할을 한다. Location-based Event Channel 내에는 여러 개의 Channel들이 존재하는데 이 채널들은 Supplier와 Consumer간의 데이터를 중개하는 역할을 한다.

3.3 위치 기반 이벤트 채널

그림 5는 그림 4에서 보여 지는 Location-based Event Channel의 세부 그림이다. 이벤트 채널 내에는 수많은 채널 리스트들이 존재하며 이러한 채널들은 위치 정보를 기반으로 하고 있다.

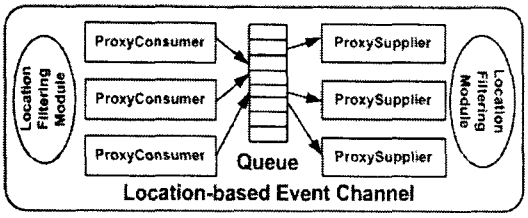

그림 5 Location-based Event Channel 구조

Location Filtering Module은 Supplier와 Consumer로부터의 받은 위치 정보를 필터링하여 proxyConsumer에게 전달하여 채널 내에 정보를 삽입하고 채널을 통해 proxySupplier로부터 정보를 가져와 전달하는 역할을 한다. 이 모듈을 통하여 사용자는 현재 자신의 영역에서 필요한 채널들을 선택할 수 있게 된다.

그리고 기존의 이벤트 서비스에서 제공되지 않는 채널 내부에 큐를 제공한다. 이 큐는 Supplier와 Consumer간의 데이터 통신에 있어서 데이터의 손실이 발생 했을 때나 시스템 자체의 문제로 인해 손실된 데이터를 다시 제공 받을 수 있게 하여 신뢰성을 지원하기 위해 존재한다.

위치 필터링을 위해 본 논문에서는 Supplier와 Consumer간의 신뢰성을 보장하기 위해 그림 6와 같이 데이터 포맷을 정하였다.

```
struct EventMsg{
    bool              ACK;
    unsigned short    EventName;
    unsigned short    EventSize;
    long              Timestamp;
    long              Expiration;
    struct            EventProperty;
}

struct EventProperty
{
    int               Number;
    string            Message;
    bool              mobility;
    long              Longitude;
    long              Latitude;
}
```

그림 6 이벤트 메시지 형식

Supplier와 Consumer는 상호간에 약속된 메시지 형식을 사용함으로서 데이터 통신에 있어서 데이터 사용의 편리함과 일관성을 제공할 수 있게 한다. 위에서 제시한 데이터 형식을 간단히 설명하자면, ACK는 데이터가 도착했을 경우 그 응답에 사용하는 변수이고 EventName은 이벤트 채널의 이름, EventSize는 이벤트데이터의 총 크기, Timestamp와 Expiration는 데이터의 라이프 사이클과 관련된다. EventProperty 구조체는 사용자의 정보들이 들어 있는데 여기에는 위치 정보와 보내고자하는 데이터의 정보가 들어가게 된다. 위치 정보를 가지고 있는 부분은 EventProperty 구조체의 Longitude와 Latitude 변수이다. GPS 수신기에 의해 수신되어진 위치 정보 텍스트를 축출하여 이 두 변수에 넣어주게 된다. 위에서 제시한 데이터 포맷을 이용하여 Supplier와 Consumer는 위치에 맞는 데이터들을 받을 수 있게 되는 것이다.

아래의 그림 7은 이벤트 채널로 데이터를 전송할 때 사용되는 소스 코드이다. 현재 사용자의 위치 정보를 축출한 뒤 사용자가 받고 자하는 곳의 영역 경계 변수 boundary를 설정한 뒤 아래와 같이 조건문을 통해 데이터를 전송하거나 전송 받으면 된다.

```
If(Channel.Longitude+boundary < current_location &&
Channel.Longitude-boundary > current_location )
{
        ...
        uTMessage_Send( instance,* szMsg, size);
        // 채널 내에 메시지 삽입
        ...
}

If(Longitude+boundary< current_location && Longitude-boundary>
current_location && Latitude+boundary< current_location &&
Latitude-boundary> current_location)
{
        ...
        uTMessage_Receive(instance);
        // 채널로부터 메시지 얻어옴
        ...
}
```

그림 7 위치 기반 메시지 송/수신 코드

3.3 동작 방법

Supplier 관점

① Supplier는 Location Server로부터 정해진 시간에 한번씩 위치 정보를 제공 받게 된다.

② Supplier는 Naming Server에게 이벤트 채널을 요청하게 된다. 만약 채널이 없을 경우 Event Channel Manager를 통해 채널을 생성하게 되고 있을 경우는 존재하는 채널의 주소 값을 전달하게 된다.

③ 채널 생성이 완료 되면 그때부터 Supplier는 생성된 채널로 데이터를 보내게 되는데 보내어진 데이터는 Location-based Event Channel내의 Location Filtering Module을 통하여 위치 정보를 분석하고 현재의 위치 내에서 돌어온 정보가 유효한지를 결정하고 채널의 큐에 데이터를 넣게 된다.

④ Supplier의 위치가 더 이상 정보를 제공하지 못하는 곳에 있다면 Supplier는 Adapter를 이용하여 Event Channel Manager에게 채널 삭제를 요청하게 된다.

⑤ 채널 삭제를 요청 받은 Event Channel Manager는 채널을 삭제하기 전 채널 내에 큐에 들어있는 모든 데이터를 Consumer에게 전송한 뒤 채널을 완전히 삭제하게 된다.

Consumer 관점

① 역시 Consumer도 Location Server로부터 정해진 시간에 한번씩 위치 정보를 제공 받게 된다.

② Naming Server에 제공 받고자 하는 채널이 있는지 여부를 요청한다. 있을 경우는 채널의 주소 값을 전달받게 된다.

③ Naming Server로부터 전해 받는 채널의 주소 값을 이용하여 채널에 접근 하게 된다. 그리고 자신의 위치 정보를 접근한 채널에게 보내주게 된다.

④ Location Filtering Module은 Consumer의 위치 정보를 분석하여 Consumer에게 필요한 정보만을 필터링

하여 전해주게 된다.

⑤ Consumer의 위치가 더 이상 정보를 제공하지 못하는 곳에 있다면 Adapter를 이용하여 채널과의 연결을 끊게 된다.

4. 성능평가

그림 8은 GPS 수신기에서 제공하는 정보를 가상으로 만들어 줄 수 있는 시뮬레이션 툴이다. 이 툴을 이용하여 객체들의 위치 정보를 만들고 전체의 이벤트 서비스 시스템에서 정해진 영역 별로 제공되는 채널 및 정보를 알아올 수 있도록 만들었다.

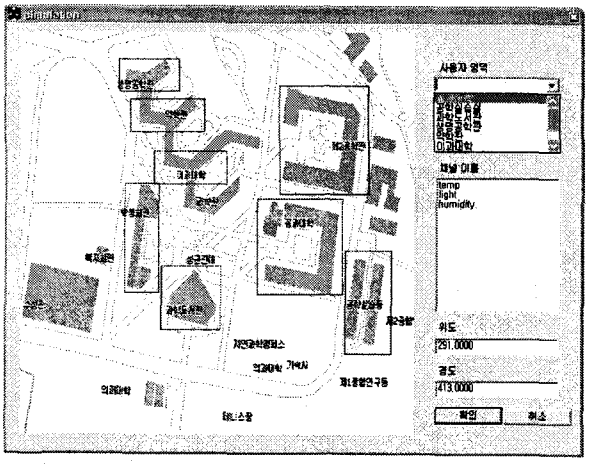

그림 8 위치 정보와 이벤트 채널 정보를 보여주는 시뮬레이션 툴

성능 평가는 위의 툴을 이용하여 이동하는 객체가 어떤 한 영역에서 데이터를 보낼 때의 지연시간과 같은 영역에서 여러 개의 Supplier가 데이터를 보낼 때의 지연시간을 측정하였다. 본 논문에서 제안된 구조를 Information Bus와 JMS에 적용하여 비교하였다. Information Bus는 Omnievent를 기반으로 하여 구현한 메시지 기반 미들웨어이다. 테스트는 CPU 2.1Ghz, RAM 512Mb, PC는 윈도우 XP기반 PC의 환경에서 실행하였다.

먼저 그림 9는 이동 객체가 정해진 한 영역에서 일대일 연결이 설정되었을 경우 이벤트 데이터 크기를 증가시키면서 이벤트 데이터의 전달 지연 시간을 측정하였다. 측정 결과 전달하는 이벤트의 크기가 클수록 위치 정보를 필터링에 상관없이 지연시간이 늘어난 것을 볼 수 있고 JMS 보다는 Information Bus가 속도 면에서는 더 좋은 성능을 내고 있다.

그림 10은 똑같은 이벤트 데이터 크기 256B를 사용하여 다대일 연결에 따른 이벤트 지연 시간을 측정하였다. Supplier수가 증가 할수록 이벤트 지연 시간도 증가하였고 일대일 연결과 비슷하게 필터링을 한 것 보다는 하지 않는 것이 JMS보다는 Information Bus가 더 좋은 성능 결과를 보였다.

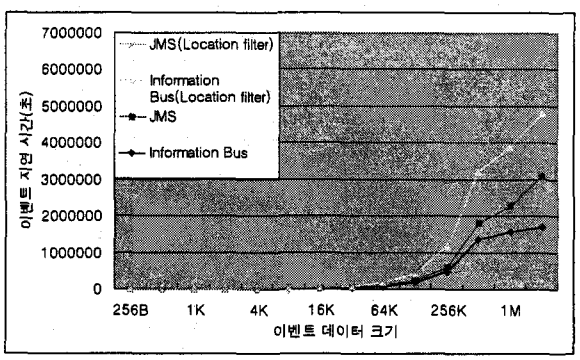

그림 9 일대일 연결에서 이벤트 데이터 크기에 따른 이벤트 지연 시간

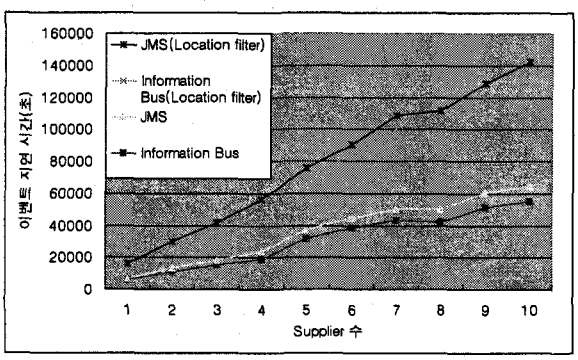

그림 10 다대일 연결에서 Supplier 수에 따른 이벤트 지연 시간

5. 결론

유비쿼터스 환경이 발전됨에 따라 이동 객체의 수와 제공되는 정보의 양은 기하급수적으로 늘어날 것이다. 그래서 이동 객체의 위치 정보를 바탕으로 한 서비스들 또한 늘어날 것으로 보인다. 이러한 추세에 맞추어 본 논문에서는 이동 객체의 위치를 기반으로 이벤트 채널의 정보를 보다 효율적으로 제공받기 위한 방법을 제시하였다.

향후에는 본 논문을 바탕으로 여러 개의 Naming Server가 운영되는 곳에 적용 시킬 것이다. 여러 개의 Naming Server 운영은 사용자에게 보다 넓은 영역에서의 정보를 제공하는 하나의 방법이 될 것이다. 하지만 그러한 시스템의 운영에는 이벤트 채널의 중복성, Supplier와 Consumer의 채널에 대한 연결 문제 등 많은 복잡함을 가진다. 따라서 현재의 연구를 바탕으로 메시지 기반 미들웨어에 대한 연구 및 위치기반 미들웨어의 조사가 필요 할 것이다.

참고문헌

[1] Microsoft Corporation MSMQ, 2006, http://www.microsoft.com/msmq/

[2] Sun Microsystems, Inc., Java Message Service (JMS), 2006, http://java.sun.com/products/jms/

[3] IBM Corporation, 2006, http://www-306.ibm.com /software/websphere/

[4] Object Management Group: CORBA Notification Service Specification, v1.1, 2004. http://www.omg.org/cgi-bin/doc?formal /2004-10-13

[5] E Kaasinen, "User needs for location-aware mobile services", Personal and Ubiquitous Computing, vol. 7, 2003.

[6] D.We::s, "Guide to GPS Positioning", N.B.Frederiction; Univ. of New Brunswick Graphic Service, 1978

[7] World's Most Popular GPS Information Resource http://www.gpsinformation.org/dale/nmea.htm

[8] Object Management Group. Event Service Specification, October 2004. v1.2, http://www.omg.org/cgi-bin/apps/doc?formal/04-10-02.pdf.

[9] F. Peschanski, "A versatile event-based communication model for generic distributed interactions", Distributed Computing Systems Workshops, 2002. Proceedings. 22nd International Conference on 2-5 July 2002, pp.503 - 510.

[10] R.E. Gruber, B. Krishnamurthy, E. Panagos, "READY: a high performance event notification service", Data Engineering, 2000. Proceedings. 16th International Conference on 29 Feb.-3 March 2000, pp.668-669.