이벤트 서비스를 이용한 컨텍스트 기반 동적 채널 관리

유영권° 윤희용
성균관대학교
{youykwon°, youn }@ece.skku.ac.kr

Context-based Dynamic Channel Management for efficient context processing using Event Service

Young Kwon Youn° Hee Yong Youn
School of Information and Communications Engineering
Sungkyunkwan University, Suwon, Korea

요 약

유비쿼터스 환경에서 사용되는 수 많은 분산 애플리케이션들은 이벤트 기반 기법을 이용한 비동기 통신을 필요로 한다. OMG에서 정의된 비동기 메시지 처리를 위해 정의된 CORBA 이벤트 서비스는 유연성 있는 이벤트 채널을 통하여 분산 오브젝트 간의 비동기 통신을 지원한다. 기존의 이벤트 서비스에서 이벤트 서비스를 제공하기 위해서는 사용자나 개발자에 의해 수동적으로 채널이 생성되며, 생성된 채널을 통하여 이벤트 서비스를 제공한다. 그러나, 수동적인 채널관리 시스템은 분산 애플리케이션이 제공하는 다양한 이벤트 서비스를 제공하기에 기능적 오버헤드를 가지고 있다. 본 논문에서는 동적 채널 생성, 삭제 그리고 컨텍스트 기반의 채널 관리 구조를 제시하며, 제안된 구조의 이벤트 서비스의 시뮬레이션 결과는 기존의 이벤트 서비스의 이벤트 처리 시간보다 더 나은 결과를 보여준다.

1. 소 개

1991년에 Mark Weiser의 논문이 발표된 이후, 유비쿼터스 컴퓨팅은 현실 세계에 접근하고 실생활의 정보 처리를 위한 주요 패러다임이 되었다[1]. 유비쿼터스 컴퓨팅은 다양한 컴퓨터가 사용자를 중심으로 동작하는 것이라고 말할 수 있는데, 유비쿼터스는 라틴어로 "언제어디서나", "동시에 존재한다"라는 의미를 갖는다. 이를 위하여 상황인지와 통신 기능을 가진 칩, 센서 및 컴퓨터가 주위의 모든 사물 또는 공간에 보이지 않게 내장되어 사람, 사물 및 기계 등 무엇이든지 서로 접속하여 실시간 으로 어떠한 정보든지 주고받을 수 있으며, 이러한 방법을 통하여 컴퓨팅(Computing), 커뮤니케이션 (Communication), 접속(Connection), 콘텐츠 (Contents), 조용함(Calm)등 5C의 5Any화(Anytime, Anywhere, Anynetwork, Anydevice, Anyservice)를 제공 함으로써 인간에게 최적의 서비스를 제공할 수 있는 차세대 기술이다.

컨텍스트는 유비쿼터스 환경에세 가장 중요한 요소 중 하나이다. Schilit와 Hilbert는 위치, 사람과 오브젝트들의 식별 그리고 오브젝트들의 변화를 컨텍스트로 언급했다[2]. 컨텍스트의 다른 정의는 엔티티의 상태를 특징화 할 수 있는 정보이다. 엔티티는 사람, 지역 혹은 사용자와 애플리케이션 간의 상호 작용과 관계될 수 있는 하나의 오브젝트라 할 수 있다[3]. 또 다른 정의들은 간단하게 컨텍스트의 개념을 나타낼 수 있는 것이다. 컨텍스트는 일반적인 우리의 생활의 복잡성을 줄이기 위해 실제와 가상 공간을 통합할 수 있다. 그와 같은 환경은 명백하게 일반적인 것이 아니라 사용자에게 매우 특유한 것이다.

OMG의 이벤트 서비스는 생성된 이벤트 채널을 통해 이벤트 서비스를 제공한다[4][5]. 여러 개의 센서에서 발생하는 다양한 종류의 이벤트를 전송하기 위해 기존의 이벤트 서비스를 사용할 때 약간의 단점을 가진다. 우선 이벤트 채널 생성과 삭제는 채널 관리자나 개발자에 의해 특정 목적을 위해 처리된다. 각각의 생성된 채널은 사용과 범위는 사용자나 개발자에 의해 정해진다. 따라서, 이와 같은 방식은 통합 관리 차원에서 불편하며 효율적이지 못하다. 다른 단점은 특정 채널을 많은 수의 Supplier와 Consumer가 사용함으로써 생기는 네트워크 트래픽이다. 예를 들어, 거대한 센서 네트워크 환경에서 온도, 조도, 습도등의 정보를 발생하는 많은 장치들이 Supplier가 되었을 경우이다. 만약 매 시간 거대한 양의 정보가 하나의 채널을 통해 흐르게 되면 분산 시스템의 전통적인 문제인 병목 현상이 발생 할 수 있다. 이와 같은 현상으로 기존의 시스템에서 하나의 채널 통한 서비스는 지연 현상이 생길 수 있다. 그러므로, 우리는 컨텍스트 분류를 통한 동적 채널 관리 구조를 제안한다. 제안된 구조의 이벤트 서비스의 시뮬레이션 결과는 기존의 이벤트 서비스의 이벤트 처리 시간보다 향상 된 결과를 보여준다.

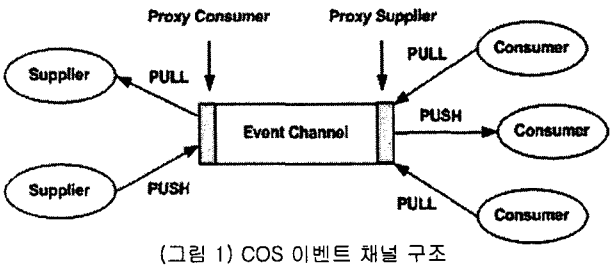
본 논문의 구성은 다음과 같다. 2장에서는 기존의 이벤트 서비스에 대하여 설명하고, 3장에서는 본 논문에서 제안하는 컨텍스트 기반의 동적 채널 관리 구조를 소개한다. 마지막으로 4장에서는 논문의 결론을 제시한다.

2. 기존 연구

2.1 Event Service

CORBA는OMG(Object Management Group) 의해 표준화된 분산 객체 컴퓨팅 미들웨어 명세서이다[6]. 유비쿼터스 환경에서 사용되는 수 많은 분산 애플리케이션들은 이벤트 기반 기법을 이

용한 비동기 통신을 필요로 한다[7,8,9]. 비동기 통신을 지원하기 위해 OMG는 CORBA 객체 서비스로 CORBA 이벤트 서비스 컴포넌트를 정의 했다. 이벤트 서비스는 비동기 메시지 커뮤니케이션을 지원하고 하나 이상의 comsumer에서 하나 이상의 supplier 간의 메시지 전달을 지원한다. (그림 1)은 COS 이벤트 채널 구조를 보여준다.



(그림 1) COS 이벤트 채널 구조

Suppliers andConsumers: Consumer는 Supplier에 의해 발생된 이벤트 메시지의 최종 목적지이다. Supplier와 Consumer 둘 다 수동적이거나 능동적 기능을 가진다. 능동적인 push suppliers는 수동적인 push consumer에게 이벤트 메시지를 넣어준다. 그리고, 수동적인 pull supplier는 능동적인 pull consumer로부터 이벤트 메시지를 가져오기 위해 대기한다.

Event Channel: COS 이벤트 서비스 관점에서 이벤트 채널은 supplier와 consumer의 중개자 역할을 담당한다. 이벤트 채널은 supplier와 consumer의 객체 레퍼런스를 관리한다. 채널의 한 측면에 있는 proxy consumer는 실제 supplier와의 연결을 지원하며, 다른 측면의 proxy supplier는 실제 consumer와의 연결을 지원한다. *Push supplier*는 이벤트 채널을 사용하여 *push* consumer에게 데이터를 전송한다. 더욱이, pull consumer는 명시적으로 pull supplier로부터 데이터를 가져올 수 있다. 이벤트 전달에서 push와 pull은 consumer와 supplier가 표준 CORBA의 두 가지 방식의 커뮤니케이션 모델의 지나친 동기화 방식의 제약을 해결해 준다. 이벤트 채널은 하나 또는 그 이상의 supplier가 다중의 consumer에게 replicator, broadcaster 또는multicaster를 이용하여 그룹 커뮤니케이션을 수행 할 수 있다.

본 논문은 컨텍스트 기반의 동적 채널 관리에 초점을 맞추고 있다. 제안된 구조에서 supplier는 이벤트 메시지의 컨텍스트에 따라 채널이 생성되며, 각각의 consumer는 받고자 하는 이벤트 메시지의 컨텍스트에 의해 적합한 채널에 연결된다. 서비스를 제공하는 supplier가 동작을 종료할 때 자동적으로 해당 채널은 삭제된다. 다음장에서 우리는 제안된 컨텍스트 기반의 동적 채널 관리 구조를 보여준다.

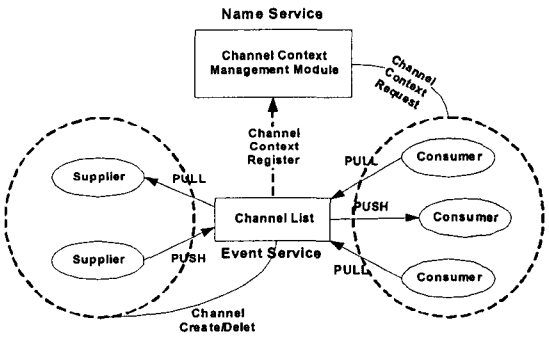
3. 제안된 컨텍스트 기반의 동적 채널 관리

3.1 동기

유비쿼터스 환경에서 사용되는 분산 시스템을 설계하고 구현하는데 있어서 많은 시스템들이 비 동기적인 통신 방법을 필요로 한다. 분산 CORBA는 객체들간의 비 동기적인 통신을 지원하기 위해서 Event Service를 지원한다. 기존의 이벤트 서비스는 정적으로 생성된 채널을 통하여 supplier와 comsumer간의 이벤트 데이터를 처리하게 된다. 하지만 유비쿼터스 환경에서의 컨텍스트는 사용자 및 기기에 부착된 센서의 종류 및 역할에 따라 다양한 형태로 발생하게 된다. 한명의 사용자에게서 서로 다른 센서에 의한 컨텍스트도 발생하게 될것이며, 빈번한 이동성을 갖는 사용자나 기기에 부착된 센서에 의한 컨텍스트 또한 이동성을 가지게 되고, 유비쿼터스 환경의 특성상 많은 양의 컨텍스트가 발생한다. 그러므로 이러한 동적으로 컨텍스트 정보들을 처리하는 센서간의 통신에서 센서가 발생하는 이벤트 내용에 정적 채널 생성을 통한 이벤트 처리는 이벤트 데이터의 구분 처리 및 동적 채널 생성등의 기술을 필요로 하게된다. 그러므로 본 논문에서는 유비쿼터스 환경에 적합한 컨텍스트 기반의 동적 채널 관리를 제안하고자 한다. 제안된 이벤트 서비스는 이벤트 채널을 효율적으로 관리 하기 위하여 이벤트 서비스에 채널 컨텍스트 관리 모듈을 추가하여 supplier와 consumer간에 사용되는 이벤트에 알맞은 채널을 부여하여 차별화된 서비스를 가능하게 하는 제안 사항을 중심으로 설명한다.

3.2 제안된 이벤트 서비스 구조

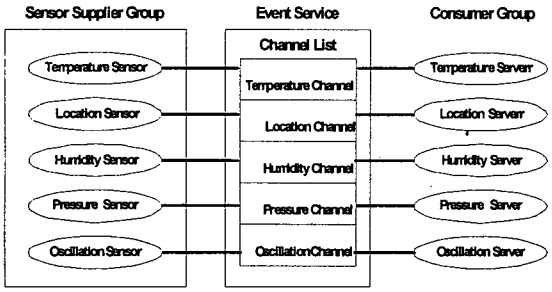
이전 장에서 설명한 것과 같이 기존의 이벤트 서비스는 정적 채널 생성, 이벤트 데이터에 따른 채널 관리가 필요하다. 따라서 본 논문에서는 동적 채널 생성과 컨텍스트 채널 관리를 통하여 유비쿼터스 환경에 알맞은 이벤트 서비스의 컨테스트 기반의 동적 채널 관리 구조를 제안하고자 한다. (그림 2)는 제안된 이벤트 서비스의 구조를 보여준다. 구조는 이벤트 네임 서비스와 이벤트 서비스로 구성되어 있다. Supplier와 consumer는 생성된 채널을 통하여 커뮤니케이션이 이뤄진다. 컨텍스트 채널 관리 모듈은 네임 서비스 모듈에 추가 되었으며 생성된 채널 목록을 관리한다.



(그림 2) 제안된 이벤트 서비스 구조

제안된 채널 관리 구조의 사용 예는 (그림 3)과 같다. Supplier 그룹은 다양한 센서들로 구성되어 있으며, consumer 그룹은 받고자 하는 컨텍스트에 따라 서로 다른 이벤트 메시지를 받게 된다.

이와 같은 구조는 push 모델뿐만 아니라 pull모델에도 적용된다. 제안된 구조는 이벤트 의해 구분되는 채널 관리를 통하여 효율적인 네트워크 트래픽을 분산하는 것이다.

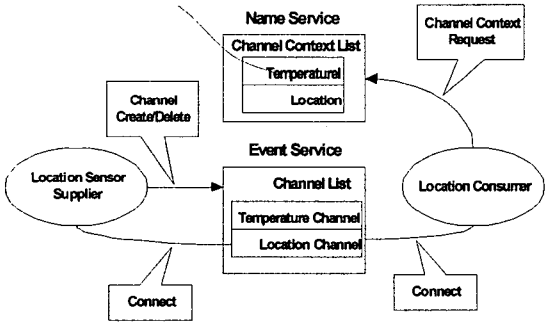


(그림 3) 컨텍스트 기반의 동적 채널 관리 사용 예

제안된 이벤트 서비스를 구현하기 위해서 omniORB Event Service를 사용하였으며, 채널 리스트 관리를 위해서 omniName Service를 이용하였다. 동적 채널 생성,삭제 기능은 Event Service 인터페이스를 이용하여 구현 하였으며, 채널 컨텍스트 리스트 관리,요청, 응답 처리는 Name Service 의 내부에 IDL 인터페이스를 추가하여 구성하였다.

3.3 제안된 이벤트 서비스 동작

(그림 4)는 제안된 이벤트 서비스의 동적 채널 생성 및 삭제 과정을 보여준다. 이 예는 Location정보를 발생하는 Sensor node가 생성 되면 Location 이라는 컨텍스트와 관련된 채널 생성을 요청한다. 생성된 채널은 채널 context list 관리를 위하여 Name Service에 등록된다. 요청된 채널을 통하여 Location sensor supplier와 consumer는 이벤트 데이터를 처리하게 된다. Location sensor가 제거 되면, 채널 삭제 요청 후에 Name Service에서 관련된 채널을 제거 요청을 하게 된다.



(그림 4) 동적 채널 생성/삭제

시스템 구조는 Name Service 모듈과 Event Service 모듈, 그리고 그에 따른 Supplier와 Consumer로 구성되어 있다. 이러한 시스템의 각 모듈과 동작 과정을 보면 다음과 같다.

① Location Sensor가 생성되면 Location관련 context를 이용하여 Event Service에게 채널 생성을 요청한다.

② Name Service의 channel context list에 생성된 채널을 추가한다.

③ 채널 생성이 완료 되면 Location Sensor는 Supplier로서 해당 채널에 접속을 하여 pull model이나 push model로서 발생된 Location Event를 Event Channel의 Proxy Consumer에게 전송하게 된다.

④ Location 이벤트를 받고자 하는 Consumer는 Name Service에게 Location Channel Context를 요청한다.

⑤ Consumer는 전송된 Channel Context를 이용하여 해당 채널로 접속 하여 Location Event등을 Proxy Supplier로부터 전송 받게 된다.

⑥ Location Sensor Supplier가 가용하지 않게 되면 채널 삭제를 요청하면, Event Service는 채널을 삭제하고 다음에Name Service에 등록된 Channel Context List에서 해당 채널 context를 제거 한다.

4. 결 론

최근 빠른 속도로 컴퓨터와 컴퓨터, 사람과 컴퓨터 사이의 네트워킹을 넘어 사람, 사물, 컴퓨터가 각각 독립적으로 연동되는 유비쿼터스 시대를 맞이하고 있다. 이러한 유비쿼터스 컴퓨팅은 언제, 어디서나, 어떠한 정보라도 편리하게 전송 및 처리가 가능하다는 점에서 미래의 핵심 기반 기술로 자리 매김하고 있다.

본 논문에서는 유비쿼터스 환경에서 사용 되는 OMG COS(Common Object Service)의 Event Service 구조에 센서가 발생시키는 이벤트에 대하여 채널 컨텍스트 관리 기능을 추가하여 동적 채널 생성 채널 및 채널 병목 현상을 해결 시킬 수 있는 효율적인 방안을 제시하였다.

향후 연구 방향으로는 제안된 구조의 최적화와 센서와 이벤트 서비스간의 직접적인 연결 할 수 있는 인터페이스 기술의 연구 개발이 필요하다.

참 고 문 헌

[1] M. Weiser, "The Computer for the Twenty-First Century", *Scientific American*, September 1991, pp.94-100.

[2] Schilit, B., Theimer, M. Disseminating Active Map Information to Mobile Hosts. *IEEE Network*, 8(5) (1994) 22-32.

[3] Anind K. Dey, Gregory D. Abowd, Towards a Better Understanding of Context and Context-Awareness, *Proceedings of HUC'99*, Karlsruhe, Germany, September 1999.

[4] Object Management Group. Event Service Specification, October 2004. Version 1.2, *http://www.omg.org/cgi-bin/apps/doc?formal/04-10-02.pdf.*

[5] Object Management Group. *http://www.omg.org/.*