2000년 한국정보처리학회 추계 학술논문발표집 제7권 제2호

Java RMI 비동기 이벤트 전달 시스템의 설계

남한석* , 윤태진** , 안광선*
*경북대학교 컴퓨터공학과
**경운대학교 소프트웨어공학과
e-mail:jesus@knight.ce.knu.ac.kr

Asynchronous Event Delivery System of Java RMI

Han Soek Nam* , Tae jin Yun** , Kwang Sun Ahn*
*Dept. of Computer Engineering, Kyungpook National University
**Dept. of Software Engineering, Kyungwoon University

요약

본 논문에서는 비동기 통신을 할 수 있는 Java RMI 이벤트 전달 시스템을 제안한다. 이 시스템은 이벤트 채널과 프락시 객체를 사용하여 소비자와 공급자 객체가 서로에 대한 정보를 알 필요 없이 독립적으로 구현되어질 수 있도록 하며, 모든 등록된 소비자 프락시 객체에게 이벤트를 전달해야하는 부담을 제거하기 위해 공급자 타입 시스템을 사용하여 이벤트 필터링을 제공한다. 또한 Java.rmi.Activation 클래스를 사용하여 이벤트 공급자 프락시 객체로 인해 발생하는 시스템부하를 줄이고, 발생하는 이벤트 정보를 객체로 정의함으로써 효율적인 이벤트 전달이 이루어지게 한다.

1. 서론

컴퓨터와 데이터통신 기술의 발전으로 기존의 중앙 집중형 시스템이 네트웍 중심의 분산 처리 시스템으로 전환됨에 따라 OMG에서는 분산객체 기술의 해결방안으로 OMA 구조를 제안하였다. OMA의 핵심기술인 CORBA와 함께 분산객체를 지원하는 기술로서는 Java RMI가 있다. Java는 플랫폼 독립성을 가지고 있으며 멀티쓰레딩을 지원하는 등의 장점을 가지고 있다[1].

Java에서 분산환경을 제공하는 Java RMI와 OMG의 CORBA는 동기 호출을 사용한다. 동기 호출은 클라이언트가 호출을 하고 응답을 받을 때까지 블록킹되며, 클라이언트와 서버 객체사이의 통신이 일대일 통신으로 이루어지기 때문에 클라이언트와 서버가 서로에 대한 통신시 필요한 모든 정보를 가지고 있어야 하는 단점을 가진다.

이런 단점을 해결하기 위해 OMG 의 CORBA는 이벤트 서비스를 정의하고 있다. 하지만 CORBA는 객체간의 통신을 위해 IDL이라는 인터페이스 언어로 인터페이스를 정의해 주어야 한다. 자바객체간의 통신에 있어서도 CORBA를 사용한다면 IDL로 인터페이스를 정의해 주어야 하는 것이다[10]. 따라서 본 논문에서는 Java RMI를 사용하여, 기존의 자바객체로만 구성된 분산객체시스템에서 효율적으로 사용할 수 있는 Java RMI 비동기 이벤트 전달 시스템을 제안한다. Java RMI 비동기 이벤트 전달 시스템은 이벤트 채널을 통해 RMI객체간의 비동기 통신을 제공하고, 이벤트 필터링 기능을 제공하며, RMI Activation 클래스를 사용하여 이벤트 공급자 프락시 객체로 인해 발생하는 시스템부하를 줄인다. 전달되는 이벤트 정보를 객체로 정의하여 효율적인 전달이 이루어지게 한다.

2. 관련연구

본 장에서는 Java RMI의 기본배경과 CORBA에서 제공하는 비동기 이벤트 서비스에 대해 살펴보도록 하겠다.

(1) Java RMI

자바에서 제공하는 일반적인 클라이언트/서버 간 통신 방법은 소켓(Socket) 프로그래밍이다. 이 방법을 사용하는 프로그래머는 기본적으로 TCP나 UDP 같은 프로토콜을 잘 이해해야 하며, 클라이언트와 서버간의 메시지 구조와 이를 해석하는 부분 등 통신에서 발생하는 많은 오버헤드를 손수 감당해야만 한다[2]. 이런 문제점 때문에 자바에서는 RMI라는 메커니즘을 제공하고 있다[3,4,5,6,7,11]. Java RMI는 순수 자바 객체로만 구성된 분산 시스템을 구성할 때 빠르고 효율적인 방법을 제공한다. 로컬에 있는 자바 객체는 원격지에 있는 객체의 함수를 RMI

라이브러리를 사용하여 로컬에 있는 것처럼 호출할 수 있다. RMI의 계층구조와 호출메커니즘은 [2]를 참조하기 바란다.

(2) CORBA 이벤트 서비스

CORBA 이벤트 서비스는 이벤트 채널을 통해 제공되어진다. 공급자 객체는 이벤트 채널에 참가하고 이벤트 채널에 이벤트를 제공한다[8]. 소비자 객체도 유사한 방법으로 이벤트 채널에 참가하고 이벤트를 받는다. 공급자와 소비자는 이벤트 전달과정에서 능동적이거나 수동적일 수 있다. CORBA 모델에서 소비자 객체와 공급자 객체는 이벤트 채널에 참가하고 있는 기간동안 이벤트 채널을 대표하는 프락시 객체를 획득한다. 프락시 객체는 이벤트 채널 객체에 의해 생성되어지고, 참가하고 있는 소비자와 공급자에게 프락시 객체에 대한 참조를 반환한후, 공급자 객체와 소비자 객체가 이벤트 채널과 상호작용을 하는 유일한 인터페이스가 된다. 그림1는 CORBA 이벤트 서비스의 구조를 보여준다.

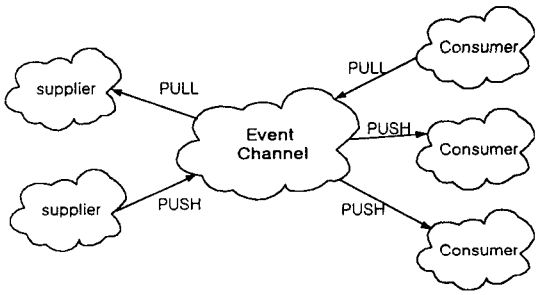

그림 1 CORBA 이벤트 서비스 구조

3. Java RMI 비동기 이벤트 전달 시스템

분산 환경에서 원격자바 객체를 사용할 수 있는 메커니즘을 제공하는 Java RMI는 객체간 비동기 통신을 지원하지 않는다. 자바는 메시지 서비스와 같은 비동기 통신 능력을 제공하는 추가적인 분산객체 서비스를 제공하지만, Java RMI와 호환되지 않는다는 단점이 있다. CORBA의 이벤트 서비스를 사용할 수도 있지만 RMI상에서 자바로 구현된 인터페이스를 IDL로 변환해주어야 하는 부담을 가지게 되며, 하부통신을 담당하는 ORB가 필요하게 된다[8,10]. 이런 단점 때문에 기존의 Java RMI로 구현된 분산 객체 시스템을 지원하면서 비동기 통신을 제공하는 시스템이 필요하다. 그리고 Java RMI 비동기 이벤트 전달 시스템을 구현할 때 Java RMI에서 제공하는 기본적인 일대일 통신을 사용한다면 이벤트 소비자 객체와 이벤트 공급자 객체가 서로에 대한 모든 정보를 알아야 한다는 단점이 생긴다. 따라서 Java RMI 비동기 이벤트 전달 시스템은 이벤트 채널을 사용함으로써 이를 해결하고 있다.

현재 분산환경에서 비동기 통신을 제공하는 CORBA 이벤트 서비스는 필터링 기능을 정의하고 있지 않다[9]. 다시 말해 이벤트 공급자 객체에서 생성된 이벤트가 등록된 모든 이벤트 소비자 객체에게 전달된다는 의미이다. 이것은 이벤트 채널뿐만 아니라 이벤트 소비자 객체에서도 부가적인 처리시간을 요구하게 된다. 더욱이 이벤트 채널에서 이벤트 소비자 객체로의 이벤트 전달은 원격호출을 해야 하기 때문에 훨씬 더 통신상의 부하가 커지게 된다.

이러한 문제를 해결하기 위해 본 논문에서는 비동기 통신과 이벤트 필터링 기능을 제공하는 Java RMI 비동기 이벤트 전달 시스템을 설계한다. 시스템의 구조는 그림2와 같다.

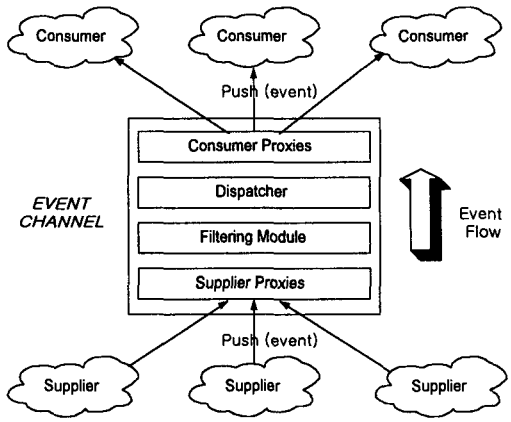

그림 2 RMI 이벤트 채널의 구조

(1) 공급자(Supplier) 객체

공급자 객체는 이벤트를 생성하는 RMI객체로서 이벤트 공급자 프락시 객체를 획득한 후 프락시 객체와의 연결을 통해 발생한 이벤트를 이벤트 채널로 전달한다. 공급자 객체는 발생하는 이벤트를 구별하기 위한 유일한 ID를 가진다. 프락시 객체를 생성시킬 때 SubscriptionManager를 통해 이벤트 채널에 ID를 등록한다.

(2) 소비자(Consumer) 객체

소비자 객체는 이벤트를 처리하는 RMI객체로서 이벤트 소비자 프락시 객체를 획득한 후 프락시 객체와의 연결을 통해 발생한 이벤트를 이벤트 채널로부터 전달받는다. 소비자 객체는 특정한 공급자 객체로부터 이벤트를 받을 수 있다. 프락시 객체를 생성시킬 때 SubscriptionManager를 통해 프락시 객체의 참조와 특정 공급자 객체의 ID를 등록한다.

(3) 이벤트 객체

Java RMI 비동기 이벤트 전달 시스템에서 발생한 이벤트가 실제적으로 저장되어지고 전달되는 객체이다. Java에서 제공하는 Serialize 기법을 사용하여 구현함으로써 Java에서 제공하는 객체뿐만 아니라 사용자가 정의한 객체까지도 투명하게 전달하도록 설계하였다. 단 사용자가 정의한 객체도 Serializable 인터페이스를 구현한 객체이어야 한다.

(4) RMI 이벤트 채널

이벤트 채널은 표준 RMI객체이고, 이벤트 채널과의 통신은 표준 RMI 호출로 이루어진다. 이벤트 채널은 이벤트 공급자 객체와 이벤트 소비자 객체 사이에서 이벤트 중개자 역할을 하기위해 이벤트 채널상에 공급자 객체와 소비자 객체의 동일한 기능을 하는 프락시 객체를 사용한다. RMI 이벤트 채널은 이벤트 소비자 및 이벤트 공급자 관리 객체를 획득하기 위한 인터페이스인 이벤트 소비자와 이벤트 공급자 프락시 인터페이스 그리고 RMIEventChannel 인터페이스를 포함한다.

소비자는 이벤트 채널상에 받고 싶은 이벤트를 등록하고 이벤트 채널은 이벤트 공급자 객체로부터 이벤트가 도착하면 해당 이벤트 소비자 프락시에게 이벤트를 전달한다. RMI 이벤트 서비스의 핵심이 되는 RMI 이벤트 채널의 구조는 그림2에서 보여준다.

1) 소비자 프락시 객체

Consumer Proxy 객체는 RMIProxySupplier 인터페이스를 지원하는 객체로서 이벤트 채널상에서 이벤트 소비자 객체의 대리인 기능을 수행한다. RMIProxySupplier 인터페이스는 이벤트 채널과의 연결을 설정하거나 연결을 해제하기 위해 이벤트 소비자 객체에 의해 사용되어진다.

2) 공급자 프락시 객체

Supplier Proxy 객체 RMIProxyConsumer 인터페이스를 지원하는 객체로서 이벤트 채널상에서 이벤트 공급자 객체의 대리인 기능을 수행한다. RMIProxyConsumer 인터페이스는 이벤트 채널과의 연결을 설정하거나 연결을 해제하기 위해 이벤트 공급자 객체에 의해 사용되어진다. 시스템 부하를 줄이기 위해 java.rmi.Activation 클래스를 사용하여 프락시 객체를 구현한다.

3) Filtering 모듈

공급자 객체에 의해 발생하는 이벤트를 전달하는 가장 쉬운 방법은 이벤트 채널에 등록된 모든 소비자 객체에게 브로드캐스트하는 것이다. 하지만 이 방법은 단점을 가지고 있다. 소비자가 발생한 이벤트의 일부만을 받고자 하는 경우 각각의 소비자 객체가 이런 기능을 모두 구현해야 하며, 버려지는 이벤트가 많아질수록 필요없는 대역폭과 처리시간을 낭비하게 된다. 따라서 RMI 이벤트 채널에서는 Filtering 모듈을 둠으로써 이 문제를 해결한다.

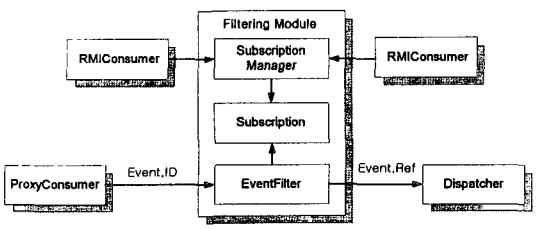

그림 3 Filtering 모듈의 구조 및 동작

이 모듈은 소비자 객체가 받고자 하는 이벤트를 Subscription Manager를 통해 Subscription에 등록할 수 있게 하고, 발생하는 이벤트를 먼저 이벤트 채널의 EventFilter에서 검사하여 해당 소비자에게 보낸다. 이벤트 필터링을 제공하기 위해서는 이벤트를 위한 잘 정의된 타입 시스템이 필요하다. 이벤트 타입시스템에 대한 전체적인 설명은 본 논문의 영역을 벗어나므로 [12]를 참고하기 바란다.

본 논문에서는 공급자 기반 필터링을 사용하여 이벤트 채널을 설계할 것이다. 그림3과 같이 Filtering 모듈은 Subscription , Subscription Manager 그리고 EventFilter 로 구성되어 있다.

4) Dispatcher 모듈

Filtering 모듈에서 이벤트와 전달되어야 할 소비자 프락시 객체에 대한 참조가 넘어오면 해당 소비자 프락시 객체에게 이벤트를 전달하는 기능을 수행한다. 이벤트 객체가 로컬객체인지 원격객체인지를 구별하여 알맞은 전달과정을 선택하여 이벤트를 전달한다.

(4) 프락시 객체 등록 및 이벤트 전달

먼저 이벤트 공급자 객체가 이벤트 채널상에서 obtain_consumer() 메소드를 호출하여 공급자 프락시 객체를 획득하고 이벤트 채널상의 SubscriptionManager를 통해 이벤트 공급자 객체 ID를 Subscription에 등록한다. 생성된 프락시 객체를 RMIProxyConsumer라고 명명하는 것은 이벤트 공급자 객체로부터 이벤트를 받는 역할을 하기 때문이다. 공급자 프락시 객체가 생성되면 이벤트 공급자 객체가 공급자 프락시 객체와의 연결을 설정하기 위해 공급자 프락시 객체상에서 connect_supplier() 메소드를 호출한다. 이때 이벤트 공급자 객체의 참조와 프락시 객체의 참조가 서로 전달되어진다.

다음으로 이벤트 소비자 객체가 이벤트 채널상에서 obtain_supplier() 메소들를 호출하여 소비자 프락시 객체를 획득하고 이벤트 채널상의 SubscriptionManager를 통해 받고자 하는 이벤트에

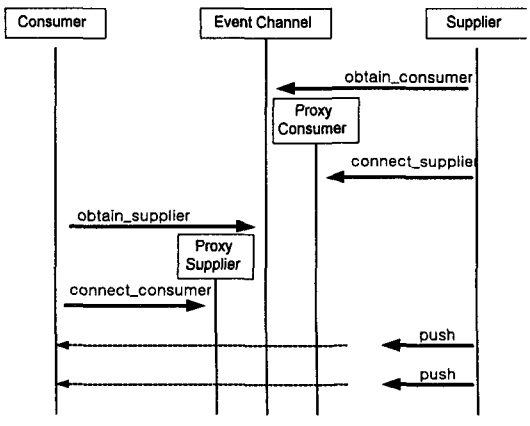

그림 4 프락시 객체등록 과정

대한 이벤트 공급자 객체 ID, 이벤트 프락시 객체 참조를 Subscription에 등록한다. 마찬가지로 생성된 프락시 객체를 RMIProxySupplier라고 명명하는 것은 이벤트 소비자 객체에게 이벤트를 전달해주는 역할을 하기 때문이다. 소비자 프락시 객체가 생성되면 이벤트 소비자 객체가 소비자 프락시 객체와 연결을 설정하기 위해 소비자 프락시 객체상에서 connect_consumer() 메소드를 호출한다. 이때 이벤트 소비자 객체의 참조와 프락시 객체의 참조가 서로 전달되어진다. 그림4는 프락시 객체 등록 과정을 보여준다.

그림5와 같이 이벤트 공급자 객체는 이벤트 공급자 프락시 객체 상에서 push() 메소드를 호출하여 발생한 이벤트와 자신의 ID를 전달한다. 이벤트 공급자 프락시 객체는 전달받은 Event 와 ID를 이벤트 채널로 전달하고 이벤트 소비자 프락시 객체는 이벤트 채널로부터 Event를 전달받게 된다. 그리고 이벤트 소비자 프락시 객체는 이벤트 소비자 객체에게 Event를 전달하게 된다.

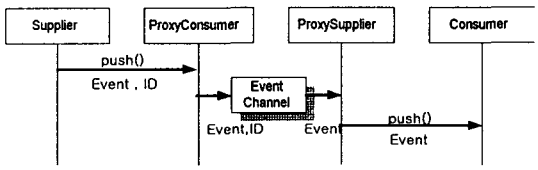

그림 5 이벤트 전달과정

이벤트 채널내에서의 처리과정은 그림6과 같다. 이벤트 채널 내부의 FilterEvent가 이벤트 공급자 프락시 객체로 부터 Event와 ID를 받고 Subscription 정보를 사용하여 발생한 이벤트에 관심이 있는 이벤트 소비자를 찾게 된다. 이때 EventFilter에 의해 찾아지는 정보는 이벤트 소비자 프락시 객체에 대한 참조이며 이것은 이벤트 소비자 객체가 SubscriptionManager를 통해 Subscription에 등록해 놓은 정보이다. 검색된 참조는 이벤트와 함께 Dispatcher에게 전달되어지게 된다. 그러면 Dispatcher는 이벤트 소비자 프락시 객체의 참조를 이용하여 소비자 프락시 객체에게 Event를 전달하는 것이다.

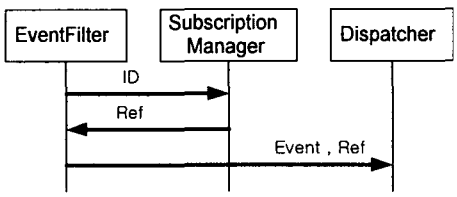

그림 6 이벤트 채널내의 이벤트 처리과정

4. 결론

본 연구에서는 Java RMI에서 비동기 통신을 제공하기 위한 방법으로 이벤트 채널을 이용한 Java RMI 이벤트 전달 시스템을 제안하였다.

Java RMI 이벤트 전달 시스템은 기존의 RMI 객체와의 호환성을 유지하기 위해 Java RMI 객체인 이벤트 채널객체를 두고, 이 객체를 통해 소비자와 공급자 객체가 서로 통신할 수 있게 한다. 따라서 현재 비동기 통신을 제공하는 CORBA 이벤트 서비스를 이용할 때 요구되는 인터페이스 재작성의 노력을 줄여준다.

이 시스템은 이벤트 채널과 프락시 객체를 사용하여, 프락시 객체가 소비자와 공급자 객체의 참조를 저장하고 대리인 역할을 하게 한다. 이것은 소비자와 공급자 객체가 통신을 위해서는 서로에 대한 정보를 알아야 한다는 부담을 해결해 주며, 두 객체가 독립적으로 구현되어질 수 있도록 한다. 등록된 모든 소비자에게 이벤트를 전달해야 하는 단점을 해결하기 위해 공급자 타입 시스템을 사용하여 이벤트 필터링을 제공한다. 공급자 기반 필터링을 사용하므로 시스템내에서 이벤트 공급자 객체는 유일한 ID를 유지해야 한다. RMI Activation 클래스를 사용하여 이벤트 공급자 프락시 객체로 인해 발생하는 시스템 부하를 줄이고, 전달되는 이벤트 정보를 객체로 정의함으로써 효율적인 이벤트 전달이 이루어지게 한다.

향후 보다 향상된 전달 시스템을 구현하기 위해 현재 이벤트 전달시 소비자 객체마다 원격호출을 하는 방식에서 원격호출을 줄일 수 있는 방법에 대한 연구가 이뤄져야 한다.

참고문헌

[1] 박성우 "가전제품과 정보기기를 위한 Java 기술" 정보과학회지 제 16권
[2] Qusay H. Mahmoud "Distributed Programming with JAVA" Manning Publication Co.
[3] Sun Microsystems "Java(TM) Remote Method Invocation Specification"
[4] M. Hughes and C. Hughes "Java Network Programming" Manning Publications Co.
[5] P. Sridharam Advanced Java Networking Prentice Hall PTR
[6] Sun Microsystems "Java Enterprise Computing " Sun Microsystems
[7] Troy Bryan Downing "Java RMI: Remote Method Invocation" IDG Books Worldwide Inc.
[8] R. Orfali and D. Harkey "Instant CORBA" Addison Wesley Publication
[9] OMG "CORBAServices" OMG Document
[10] OMG "The Common Object Request Broker : Architecture and Specification (Revision 2.2)" OMG Document
[11] Cay S. Horstmann and Gray Cornell "core JAVA: Advanced Features" Sun Microsystems
[12] D. C. Schmidt, D. L. Levine and T. H. Harrison "An ORB Endsystem Architecture for Hard Real-Time Scheduling" , OMG Document