인터넷 기반 멀티미디어 응용을 위한 UQoS 관리 미들웨어 프레임워크

윤 은 영[†]·김 수 중[††]·윤 용 익[†††]·김 성 훈[††††]·장 철 수[††††]

요 약

본 논문에서는 인터넷 기반 멀티미디어 응용 서비스가 요구하는 상호운용성과 사용자에게 높은 품질의 서비스를 지원할 수 있는 UQoS 관리(UQoSM : User Quality of Service Management) 미들웨어 프레임워크를 제안한다. UQoS 관리 미들웨어 시스템은 멀티미디어 응용 시스템들의 사용자 요구사항들을 지원하기 위해 기존의 이벤트 서비스 모델에 리플렉션(reflection) 기법을 적용하여 이벤트 모니터, 리플렉티브 이벤트 필터링, 실시간 관리 등의 컴포넌트를 포함하는 구조를 가진다. 특히, 본 논문에서는 인터넷 기반 멀티미디어 응용 서비스 사용자의 다양한 요구사항을 지원할 수 있는 리플렉티브 이벤트 필터링을 제공하는 것에 중점을 두었다. 이로써 사용자는 보다 높은 서비스를 제공받게 되고 이 과정을 통해 전체적인 네트워크 트래픽이 감소되는 효과를 얻을 수 있다.

UQoS Management Middleware Framework for Internet-Based Multimedia Application

Eun-Young Yoon[†] · Soo-Joong Ghim[††] · Yong-Ik Yoon[†††]
Sung-Hoon Kim[††††] · Chul-Soo Chang[††††]

ABSTRACT

This paper proposes a UQoSM (User Quality of Service Middleware) framework for multimedia application systems. UQoSM system is extended the existing event service model added to the event monitoring, reflective event filtering and event dispatcher for supporting multimedia application systems. Especially, this paper is concentrated on providing suitable reflective event filtering function for multimedia application service system in order to meet various user requirements under inter-based environment. It means this system provides high QoS to users. In addition, it results in decreasing network traffic as unnecessary event information is filtered from network.

키워드 : 이벤트 서비스(event service), 리플렉션(reflection), 미들웨어(middleware), 필터링(filtering)

1. 서 론

미래의 인터넷은 장소와 장치에 관계없이 사용자가 요구하는 정보를 제공할 수 있는 응용 서비스 환경을 제공하여야 한다.

지금까지 이 기종의 분산 환경 시스템들을 통합된 환경처럼 제공하기 위해 많은 연구들이 진행되고 있으며, 이러한 통합된 환경을 기반으로 안정되고 질 높은 서비스를 제공하기 위한 연구가 국내외 일부 연구기관에서 활발히 진행되고 있다. 그러나, 차세대의 멀티미디어 응용 시스템에서 요구하는 서비스들은 기존의 요구 조건에 보다 정확한 시간에 사용자가 자신이 원하는 서비스만을 선택해서 제공받을 수 있도록 요구하고 있다. 또한, 멀티미디어 기술의 발달로 높은 품질의 서비스를 필요로 한다.

인터넷 기반 환경에서 적응성을 갖춘 멀티미디어 응용 서비스를 지원하기 위해서는 응용 서비스가 실행환경의 변화를 검출할 수 있어야 하며 이러한 변경된 사항들이 수용될 수 있도록 하부의 미들웨어 기능 요소들을 재구성할 수 있어야 한다. 현재 이러한 문제를 해결하기 위한 방안으로 리플렉션 기법(Reflection) 적용에 대한 연구가 미들웨어 분야에서 활발히 진행되고 있다. 리플렉션 기법은 컴포넌트를 검사하여 수정할 수 있고 미들웨어의 동작을 변경할 수 있다. 리플렉션 미들웨어 기술은 응용 프로그램 또는 사용자가 명백히 제어할 필요 없이 핵심적인 소프트웨어/하드웨어 기법을 동적으로 적응시킴으로써 응용 프로그램 작동의 자율적인 변화가 가능하도록 하는 기술이다[3].

본 논문에서는 이러한 리플렉션 기법을 적용하여 사용자에게 높은 품질의 서비스를 지원할 수 있는 컴포넌트 기반의 UQoS(User Quality of Service) 관리 미들웨어 프레임워크를

† 준 회 원 : 숙명여자대학교 대학원 컴퓨터과학과
†† 정 회 원 : 숙명여자대학교 대학원 컴퓨터과학과
††† 종신회원 : 숙명여자대학교 전산학과 교수
†††† 정 회 원 : 한국전자통신연구원
논문접수 : 2002년 7월 29일, 심사완료 : 2002년 10월 6일

제안한다.

UQoS 관리 미들웨어 시스템은 인터넷 기반 멀티미디어 응용 서비스가 요구하는 상호운용성 및 재사용성 등을 지원할 수 있는 표준화된 분산 시스템 프레임워크인 CORBA(Common Objcet Request Broker Architecture)를 채택하였으며 이벤트의 비동기 전송을 지원하는 CORBA 이벤트 서비스를 사용자의 요구사항을 지원할 수 있도록 리플렉션(Reflection) 기법을 적용하여 사용자에게 높은 품질의 서비스를 제공할 수 있는 리플렉티브 이벤트 필터링 연구에 중점을 두었다.

본 논문의 구성은 다음과 같다. 2장에서 관련연구로 이벤트 서비스 모델을 분석하고 3장에서는 인터넷 기반 멀티미디어 응용 서비스의 요구사항을 분석하여 이를 지원할 수 있는 UQoS 관리 미들웨어 프레임워크를 설계하고 설계한 객체 컴포넌트들의 기능을 설명한다. 4장에서는 UQoS 관리 프레임워크의 검증을 위해 멀티미디어 응용 서비스 시스템에 UQoS 관리 미들웨어 적용방안을 제시하고, 테스트 프로그램 구현에 대해 설명한다. 마지막으로 5장에서는 본 연구의 결론 및 향후 연구 과제를 제시한다.

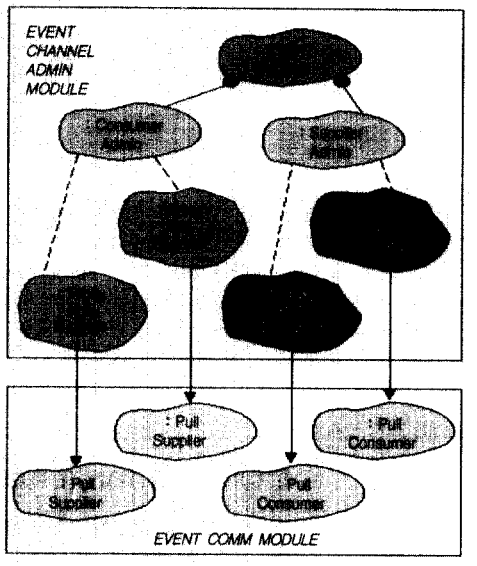
2. 이벤트 서비스

기존의 CORBA 애플리케이션들 사이에서 발생하는 통신의 형태는 상대방의 인터페이스를 이용해서 직접 전달하는 방법을 사용한다[2]. 그러나 CORBA의 이벤트 서비스는 ORB상에 존재하는 애플리케이션들간의 비동기적인(asynchronous) 통신을 위한 모델을 정의하고 있다. 다시말해서, 클라이언트는 특정 서버에 있는 객체의 연산을 직접 호출해서 데이터를 전달하는 대신, 무특정 다수의 객체들에 전달될 수 있는 이벤트를 전달한다. 이벤트를 전달하는 애플리케이션을 이벤트 공급자(supplier)라고 하고 이벤트를 전달 받아서 처리하는 애플리케이션을 이벤트 소비자(consumer)라고 한다. CORBA 이벤트 서비스는 CORBA 통신에 이벤트의 개념을 도입한 것이다. 이벤트는 이벤트 공급자에 의해서 발생하며 다수의 이벤트 소비자에게 전달된다. 공급자는 다수의 소비자에 대한 정보 또는 식별자를 갖지않으며 소비자 역시 전달된 이벤트의 공급자에 대한 정보를 갖지 않는다.

이 모델을 지원하기 위하여 CORBA 이벤트 서비스는 이벤트 채널(Event Channel)이라는 모델을 도입하여 공급자와 소비자 사이에 발생하는 이벤트를 중재하고 연결한다.

공급자와 소비자는 중간 매개인 이벤트 채널에 연결되었기 때문에 서로 직접 연결되지 않는다. 따라서, 서로에 대한 정보 없이도 이벤트를 전달할 수 있다. 공급자의 관점에 있어서, 이벤트 채널은 단일 소비자처럼 보이며, 소비자 관점에서는 이벤트 채널이 단일한 공급자처럼 보인다. 이러한 방식으로, 이벤트 채널은 공급자와 소비자를 연결한다.

(그림 1)은 CORBA 이벤트 모델을 Push 모델과 Pull 모델로 구분하여 공급자, 이벤트 채널, 소비자와의 이벤트 전송 과정을 나타내고 있다. CORBA는 공급자와 소비자간의 이벤트 전송을 시작하는 두가지 방법을 규정하고 있다. 이러한 접근 방식들은 Push 모델과 Pull 모델로 불린다.



(그림 1) CORBA 이벤트 모델

Push 모델에서 공급자는 이벤트가 발생하면 소비자에게 이벤트를 전달함으로써 이벤트 통신을 시작한다. Pull 모델에서는 소비자가 이벤트를 요구함으로써 이벤트 통신을 시작한다.

공급자와 소비자는 이벤트 채널에 의해서 완전히 분리되기 때문에 Push 모델과 Pull 모델은 단일 시스템에서 혼용되어서 사용될 수 있다. 예를 들어, 공급자는 Push 모델을 사용해서 이벤트 채널에 연결하는 반면, 소비자는 Pull 모델을 이용하여 이벤트 채널에 연결하는 경우이다.

이러한 경우, 이벤트 통신이 이루어지기 위해서는 공급자와 소비자가 모두 능동적으로 참여해야 한다. 공급자는 이벤트 채널에 있는 객체의 연산을 호출해서 이벤트 채널에 이벤트를 전송한다. 공급자는 이벤트 채널에 다른 연산을 호출함으로써 채널로부터 이벤트 데이터를 전송받는다. 이벤트 채널은 Pull 모델의 소비자가 이벤트를 요구할때까지나 Push 모델의 소비자가 이벤트 채널에 연결하기 전까지 공급자에 의해서 전달받은 이벤트를 저장한다.

2.1 이벤트 타입별 통신

CORBA 이벤트 서비스는 이벤트를 연산 호출의 파라미터 또는 반환값의 형태로 전달한다. 다시 말하면 Push 모델에서 공급자는 이벤트 채널의 연산을 호출할때 파라미터로 이벤트를 전달하고, 이벤트 채널은 소비자의 연산을 호출할때 파라미터로 이벤트를 전달해서 결과적으로 이벤트를 공급자에서 소비자에게 전달하게 된다.

이벤트 데이터는 각각의 애플리케이션에 종속적이며, 이벤트 통신은 이벤트 데이터의 형태에 따라서 untyped나 typed 형태 중에 하나를 갖는다.

2.1.1 Untyped 이벤트 통신

untyped 이벤트 통신에서, 이벤트 데이터는 any 타입으로 써 공급자에서 소비자에게 전달된다.

pull( ) 연산은 파라미터를 갖지 않는 대신, any 타입의 반환 값을 통해서 이벤트를 전송 받는다. 이 경우에서 공급자와 소비자 애플리케이션들은 any 타입의 내용에 대해서 동의해야 하며, 이 데이터가 유용한 것이면 이벤트 데이터를 전달한다.

2.1.2 Typed 이벤트 통신

type 이벤트 통신은 프로그래머에 의해서 이벤트 데이터의 형태가 정해지는 경우를 말한다. 따라서 이벤트가 전달되는 애플리케이션에 종속적인 IDL 인터페이스를 정의하게 된다. any 타입을 이용해서 데이터를 전송하는 push( )와 pull( ) 연산을 사용하는 대신 프로그래머는 이벤트 통신의 목적으로 사용되는 인터페이스를 정의하고, 이 인터페이스를 통해서 소비자와 공급자는 이벤트를 전달한다. 인터페이스상에서 정의된 연산들은 적합한 IDL 데이터 타입으로 정의된 이벤트를 파라미터로 가질 수 있다.

Push 모델에서 이벤트 통신은 이 인터페이스에 정의된 연산을 호출함으로써 시작된다. Pull 모델에서는 이벤트 통신이 좀더 복잡하다. 왜냐하면 이벤트 통신이 프로그래머가 정의한 애플리케이션에 종속적인 인터페이스로부터 특별하게 만들어진 인터페이스상의 연산을 호출함으로써 시작되기 때문이다. 이벤트 통신은 이벤트 전달을 위해서 프로그래머가 정의한 인터페이스의 연산을 호출함으로써 이루어진다.

3. UQoS 관리 미들웨어 프레임워크

본 논문에서 제안하는 UQoS 관리 미들웨어는 사용자에게 높은 품질의 서비스를 제공하기 위해 미들웨어 레벨에서 사용자의 요구사항을 관리하여 최적의 서비스를 제공하는 미들웨어로 사용자에게 높은 품질의 서비스를 제공하는 UQoS (User Quality of Service) 미들웨어로 정의하였다.

멀티미디어 응용 서비스 사용자에게 높은 품질의 서비스를 제공하기 위하여 본 논문에서는 기존의 이벤트 서비스 모델에 리플렉션(Reflection)기법을 적용한 UQoS 관리 미들웨어 프레임워크를 제안한다.

UQoS 관리 미들웨어는 인터넷 기반 멀티미디어 응용 시스템이 요구하는 상호운용성 및 재사용성을 지원할 수 있는 CORBA를 기반으로 비동기 이벤트 전송을 지원하는 CORBA 이벤트 서비스 모델에 리플렉션 기법을 적용한 리플렉션 이벤트 필터링 모델에 중점을 두어 사용자의 요구사항을 반영하여 높은 품질의 서비스를 제공하고 필요없는 이벤트는 전송전에 제거하여 이를 통해 네트워크 교통량을 줄이는 효과에 중점을 두어 설계하였다.

3.1 멀티미디어 응용 서비스 요구사항

인터넷 기반의 멀티미디어 응용 프로그램을 구축하는데 있어 여러 솔루션들이 제시되고 있다. 가장 효율적이고 목표 시스템에 적합한 최적의 솔루션을 선택하기 위해서는 목표 시스템의 요구사항을 분석하여 사용자 요구사항을 충족시킬 수 있는 개발 환경을 선택해야 한다.

본 논문에서는 다음과 같은 사용자 요구사항이 있는 것으로 간주한다.

- 목표 시스템은 웹브라우저만을 가지고 인터넷 환경(TCP/IP)에서 운영되어야 한다.
- 데이터베이스를 사용해야 하며 향후 다른 기간 업무들에서 구축된 시스템에서 제공하는 데이터 서비스에 대해 손쉽게 호출이 가능해야 한다.
- 목표 시스템 역시 향후 개발될 다른 기관 업무들이 제공하는 데이터 서비스에 대해 손쉬운 호출이 보장되어야 한다.
- CGI와 같이 서버에 부하를 많이 주는 방식은 사용하지 않으며, 표준화되지 않은 웹서버 API 등은 사용하지 않는다.
- 특정 벤더의 데이터베이스에 종속된 시스템이어서는 안된다.
- 사용자는 자신이 원하는 시간에 원하는 정보(이벤트)를 전송 받아야 한다.
- 사용자는 자신이 원하지 않는 정보(이벤트)는 전송 받지 않아야한다.

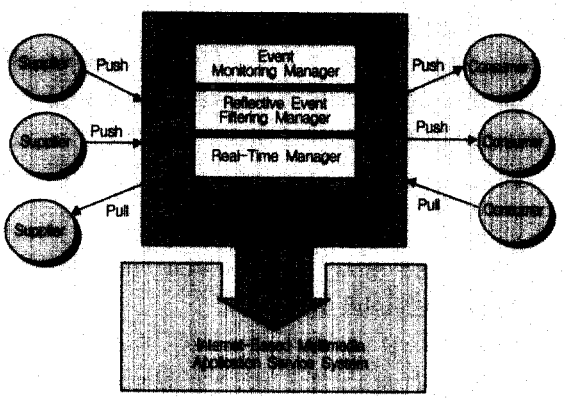
요구 분석 단계에서 고려한 내용에 따르면 결국 표준화된 연동방식을 제공하는 분산 시스템이어야 하며 인트라넷 환경에서 운영되어야 한다.

본 논문에서는 이러한 요구사항을 지원하기 위해 표준화된 분산 시스템인 CORBA 환경에서 데이터베이스에 독립적인 시스템을 구현하기 위해 JDBC를 사용하고 구현에 있어서는 비즈니스 계층을 분리하고 웹브라우저에서 모든 기능이 동작할 수 있도록 하기 위해 자바 애플릿을 클라이언트로 사용하도록 한다.

3.2 UQoS 관리 미들웨어 설계

인터넷 기반 환경에서 적응성을 갖춘 멀티미디어 응용 서비스를 지원하기 위해서는 응용 서비스가 실행환경의 변화를 검출할 수 있어야 하며 이러한 변경된 사항들이 수용될 수 있도록 하부의 미들웨어 기능 요소들을 재구성할 수 있어야 한다. 현재 이러한 문제를 해결하기 위한 방안으로 리플렉션 기법 적용에 대한 연구가 미들웨어 분야에서 활발히 진행되고 있다. 리플렉션 기법은 컴포넌트를 검사하여 수정할 수 있고 미들웨어의 동작을 변경할 수 있다. 리플렉션 미들웨어 기술은 응용 프로그램 또는 사용자가 명백히 제어할 필요 없이 핵심적인 소프트웨어/하드웨어 기법을 동적으로 적응시킴으로써 응용 프로그램 작동의 자율적인 변화가 가능하도록 하는 기술이다.

UQoS 관리 미들웨어는 리플렉션 기법을 적용하여 컴포넌트들을 검사하고 사용자의 요구사항을 반영하여 컴포넌트를



(그림 2) UQoS 관리 미들웨어 프레임워크

수정할 수 있도록 설계하였다. UQoS 관리 미들웨어 프레임워크는 (그림 2)와 같으며 이벤트 모니터 관리, 리플렉티브 이벤트 필터링 관리, 실시간 관리 컴포넌트와 UQoS 저장소로 구성되어 있으며 각 구성요소의 기능은 다음과 같다.

- 이벤트 모니터
  공급자와 소비자로부터 들어오고 나가는 이벤트들의 흐름을 관리하고 모니터하는 컴포넌트
- 리플렉티브 이벤트 필터링
  공급자와 소비자로부터 등록된 정보 및 요구사항을 반영하여 이벤트를 검사하고 요구사항을 지원할 수 있도록 이벤트를 수정하고 필터링하는 컴포넌트
- 실시간 관리
  사용자로부터 입력되는 이벤트의 마감시간, 우선순위와 같은 실시간 요구사항을 바탕으로 이벤트를 스케줄링하고 관리하는 컴포넌트
- UQoS 저장소
  사용자에게 높은 품질의 서비스를 제공하기 위하여 사용자의 요구사항과 기본 정보를 저장하는 저장소

3.3 리플렉티브 이벤트 필터링

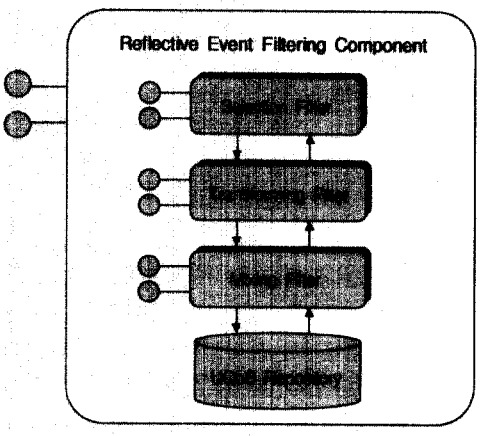
필터는 멀티미디어 응용 서비스에서 광범위하게 쓰이는 용어로 다음 세 가지 대표적인 기능이 있다.

- 선택(Selection)
  두 가지이상의 데이터가 있을때 그 중에 하나 또는 그 중에 몇 개를 선택하고 나머지는 제거하는 기능
- 변환(Transforming)
  하나의 데이터를 다른 요소를 추가하여 새로운 데이터로 변환시키는 기능
- 혼합(Mixing)
  두 가지 이상의 데이터를 혼합하여 새로운 기능의 데이터를 생성하는 기능

필터의 세 가지 기능 중에서 멀티미디어 통신 분야에서는 선택 기능을 적용하여 필요없는 메시지를 제거하는 기술로 광범위하게 응용하고 있다[1].

리플렉티브 이벤트 필터링은 선택 기능 뿐만아니라 변환, 혼합 기능을 모두 적용하여 사용자의 요구사항을 반영할 수 있는 컴포넌트로 설계하였다.

(그림 3)은 리플렉티브 이벤트 필터링 컴포넌트 모델이며 선택 필터, 변환 필터, 혼합 필터 등으로 구성되어 있다.



(그림 3) REF 모델

리플렉티브 이벤트 필터링은 리플렉티브 기법을 적용하여 UQoS 저장소에 저장된 사용자 요구사항 및 관심있는 이벤트의 정보를 바탕으로 이벤트를 검사하고 사용자 요구사항을 반영하여 이벤트를 선택, 변환, 혼합 필터를 이용하여 사용자가 원하는 이벤트만을 전송하도록 필터링한다.

- 선택 필터(Selection Filter)
  이벤트가 입력되면 UQoS 저장소에 있는 마감 시간, 우선순위와 같은 실시간 요구사항 정보를 반영하여 이벤트를 계속 진행시킬 것인지 제거할 것인지 선택하여 필요없는 이벤트 전송을 차단하므로 네트워크 트래픽을 감소시키는 기능 수행
- 변환 필터(Transforming Filter)
  이벤트를 검사하고 사용자의 요구사항을 반영하여 필요시 이벤트를 수정하고 변환하여 전송시키는 기능 수행
- 혼합 필터(Mixing Filter)
  UQoS 저장소에 있는 사용자 정보를 반영하여 하나 이상의 이벤트를 혼합하여 사용자가 필요로하는 새로운 이벤트를 생성하여 제공하는 기능 수행

이러한 필터링 서비스를 제공하기 위해서는 필터를 "어디에 넣을 것인가"를 고려해야하며, "어떻게 사용자의 관심사항을 반영할 것인가"에 대해 고려해야한다. CORBA에서는 이러한 필터 기능을 이벤트 전송 과정의 필요한 시점에 넣어서 수행시키기 위해 ORB 내에 인터셉터(interceptor)라는 객체를 추가하여 여러 가지 기능들을 수행할 수 있도록 한다. 인터셉터는 CORBA 표준 명세서에 정의된 기능이다.

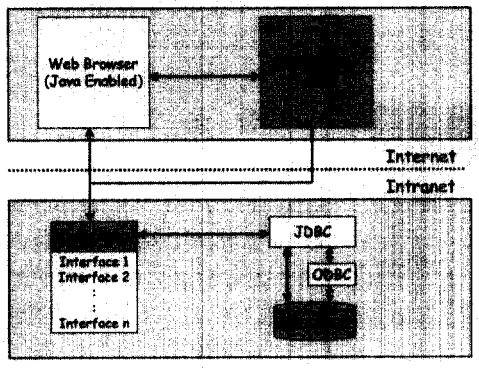
필터는 이러한 인터셉터 기능을 기반으로 수행할 수 있다. 필터에는 두 가지 형태가 있다. 첫 번째로 프로세스 단위 필

터(Per-Process Filter)는 대상 객체에 관계없이 클라이언트나 서버의 주소 공간에 들어오고 나가는 모든 연산과 속성의 호출에서 나타난다. 두 번째로 객체 단위 필터(Per-Object Filter)는 객체 단위로 적용된다.

리플렉티브 이벤트 필터는 인터셉터 개념을 도입하고 객체 단위 필터를 사용한다. 객체 단위 필터는 어느 특정 객체에서만 나타나는 필터로 프로세스 단위 필터에서 클라이언트와 서버간의 연산 수행 과정에서의 10개 시점에서 나타날 수 있는 것과 달리 다른 필터링 포인트를 제공한다. 즉, 프로세스 단위 필터는 처리 순서에서 나타나는 것과 달리 객체 단위 필터는 내부-프로세스 연산 요구를 위해 적용된다.

4. UQoS 관리 미들웨어 테스트 프로그램

UQoS 관리 미들웨어를 검증하기 위한 테스트 프로그램에 대하여 설명한다.



(그림 4) 전체 시스템 구성도

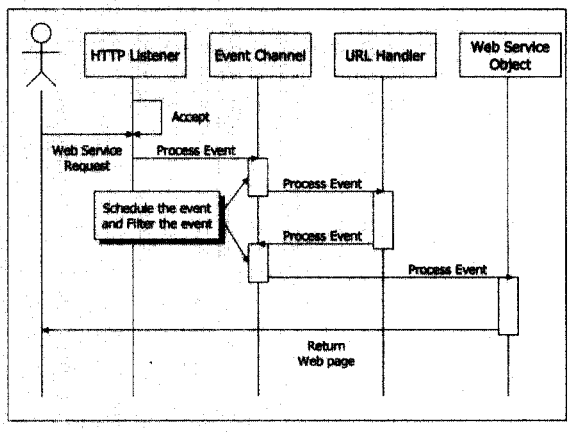
(그림 4)는 인터넷 기반 UQoS 관리 미들웨어를 테스트하기 위한 전체 시스템 구성도이다. 웹서버는 클라이언트와 CORBA 서버간의 중간 역할을 담당하는 기능을 담당하며 웹서버와 클라이언트는 HTTP로 통신하고 CORBA 클라이언트와 서버는 IIOP로 통신한다. HTTP to IIOP Gateway는 자바 애플릿의 서버 참조 요청을 처리하기 위해 웹서버에서 함께 동작하고 있어야 한다.

(그림 5)는 웹서비스에서의 이벤트 흐름을 나타낸 것이다. 사용자로부터 입력된 요청은 이벤트 채널의 이벤트 스케줄링과 이벤트 필터링을 거쳐 처리된다.

Untyped 이벤트는 일반적인 any 타입을 사용한 일반 호출을 이용해서 전달된다. 다른 것은 별로 고려하지 않아도 되며 단지 이벤트 데이터를 전송하기 전에 any 타입으로 묶는 것만 필요하다. 그러나 몇 가지 단점을 가진다.

- 전송되는 모든 정보의 타입이 any 타입이기때문에 필요한 형으로 변형해야 한다. 따라서, 비 효율적이다.
- 이벤트를 필터링해서 소비자를 선택할 수 없다.

COS 이벤트 스펙은 또한 typed 이벤트를 정의하고 있다.



(그림 5) 이벤트 웹서비스 순서도

이것은 untyped에 비해서 사용하기 복잡하나, untyped 이벤트 사용시에 나타나는 단점을 해결할 수 있다.

typed 이벤트를 사용하는 경우, 공급자는 일반적인 CORBA 요구에서처럼 이벤트 채널에서 제공하는 typed 인터페이스 상의 연산을 호출함으로써 이벤트를 전달하며 필터링 기능이 가능하다. REF은 typed 이벤트를 사용하여 필터링 기능을 수행한다.

OrbixWeb 필터 클래스는 IE.Iona.OrbixWeb. Features. Filter에 정의되어 있으며, 동적으로 설치/제거할 수 없게 설계되었다. 그러나 UQoS를 지원할 수 있는 REF(Reflective Event Filtering)를 제공하기 위해서는 실행중에 새로운 필터를 추가/삭제할 수 있어야 한다. REF을 제공하기 위하여 마스터/슬레이브 설계방식을 적용하였다. 마스터/슬레이브 설계방식을 사용하면 실행 중에 새로운 필터를 추가/삭제 할 수 있다. 마스터 필터를 제거하고 등록할 수는 없으나, 마스터 필터는 여러 개의 슬레이브 필터를 관리할 수 있는 필터 백터를 사용한다. REF는 슬레이브 필터를 동적으로 마스터 필터에 등록하고 제거하는 방법을 사용하여 사용자의 요구사항을 반영하는 필터를 제공한다. (그림 6)은 선택 필터(selection filter)를 동적으로 등록하는 코드이다.

```
import java.lang.reflect.* ;
public class MasterFilter extends Filter {
        private Vector FilterVector ;
        public MasterFilter( ) {
         super (true) ;
         FilterVector = new Vector( ) ;
        }
        public viod register (Filter selectionf) {
         if (FilterVector.contains (selectionf)
         else Filter.Vector.addElement (selectionf) ;
        }
```

(그림 6) 필터 등록 및 제거

(그림 7)은 등록된 모든 필터를 이용하여 이벤트를 필터링하는 코드이다.

```
public boolean outRequestPreMarshal
                (org.omg.CORBA.Request r) {
        for (int i = 0 ; i < FilterVector.size ( ) ; i++ {
        return ((Filter)FilterVector.elementAt
            (i)).outRequestPostMarshal (r) ;
        }
    return ;
    }
public boolean outRequestPostMarshal
                (org.omg.CORBA.Request r) {
        for (int i = 0 ; i < FilterVector.size ( ) ; i++ {
        return ((Filter) FilterVector.elementAt
            (i)).outRequestPostMarshal (r) ;
        }
    return ;
    }
```

(그림 7) 등록된 필터를 필터링하는 코드

5. 결 론

인터넷 기반의 멀티미디어 응용 시스템에 있어서 효율적이고 사용자 요구사항을 반영할 수 있는 구현 방안을 위해 세계적으로 많은 연구가 진행되고 있다. 인터넷 기반 멀티미디어 응용 시스템들은 상호운용성 및 사용자의 요구사항을 반영할 수 있는 서비스를 요구한다.

본 논문에서는 이러한 요구사항을 지원할 수 있는 UQoS 관리 미들웨어 프레임워크를 제안하였다. UQoS 관리 미들웨어는 사용자의 요구사항을 반영시키기 위하여 리플렉션 기법을 적용하여 이벤트 필터링을 수행하도록 설계하는데 중점을 두었다.

리플렉션 기법은 컴포넌트를 검사하여 필요시 컴포넌트를 수정하고 미들웨어의 동작을 변경할 수 있는 기술로 이 기법을 필터링에 적용하면 사용자가 원하는 이벤트를 제공하는데 효과가 있다.

참 고 문 헌

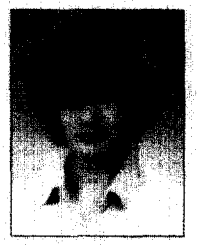
[1] N. Yeadon, F. Garria, D. Hutchison and D. Shepherd, "Filters : Qos Support Mechanisms for Multipeer Communications," IEEE Jour on Selected Areas In Communications, Vol.14, No. 7, pp.1245, September, 1996.

[2] G. Liu and A. Mok, "An Event Service Framework for Distributed Real-Time Systems," Proceedings of IEEE Workshop on Middleware for Distributed Real-time Systems and Services, December, 1997.

[3] Jacques Ferber. Computational Reflection in Class Based Object Oriented Languages. In Proceedings of OOPSLA'89, ACM, pp.317-326, October, 1989.

[4] Jacques Ferber. Computational Reflection in Class Based Object Oriented Languages, In Proceedings of OOPSLA'89, ACM, pp.317-326, October, 1989.

[5] Francois-Nicola Demers and Jacques Malenfant, Reflection in Logic, Functional and Object-Oriented Programming : a Short Comparative Study, In Proceedings of the IJCAI'95 Workshop on Reflection and Metalevel Architectures and Their Applications in AI, pp.29-38, August, 1995.



윤 은 영

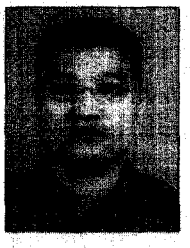
e-mail : foxdiver@sookmyung.ac.kr
1997년 숙명여자대학교 전산학과(이학사)
1999년 숙명여자대학교 대학원 전산학과 (이학석사)
1999년~현재 숙명여자대학교 대학원 컴퓨터과학과 박사과정
관심분야 : 인터넷 응용서비스, 분산 시스템, 분산 미들웨어 시스템, CORBA, 멀티미디어 통신

김 수 중

e-mail : sjghim@sookmyung.ac.kr
1995년 숙명여자대학교 전산학과(이학사)
1998년 숙명여자대학교 대학원 전산학과 (이학석사)
1998년~현재 숙명여자대학교 대학원 컴퓨터과학과 박사과정
관심분야 : 분산 미들웨어 시스템, 모바일 시스템

윤 용 익

e-mail : yiyoon@sookmyung.ac.kr
1983년 동국대학교 통계학과(이학사)
1985년 한국과학기술원 전산학과(공학석사)
1994년 한국과학기술원 전산학과(공학박사)
1998년~현재 숙명여자대학교 전산학과 교수
관심분야 : 정보통신, 멀티미디어통신, 분산시스템, 실시간 처리 시스템, 분산 미들웨어 시스템, 분산 데이터베이스 시스템, 실시간 OS/DBMS



김 성 훈

e-mail : saint@etri.re.kr
1995년 광운대학교 전자공학과(학사)
1997년 광운대학교 전자공학과(공학석사)
1996년~1998년 시스템공학연구소 연구원
1998년~현재 한국전자통신연구원 선임 연구원
관심분야 : 모바일 미들웨어, 응용서버, 분산시스템, CBD

장 철 수

e-mail : jangcs@etri.re.kr
1995년 인하대학교 전자계산공학과(학사)
1997년 광주과학기술원 정보통신공학과 (석사)
1997년~1998년 시스템공학연구소 연구원
1998년~현재 한국전자통신연구원 연구원
관심분야 : 모바일응용서버, 웹기술, 데이터베이스 등