전자교통행정서비스를 위한 멀티미디어 메시징 엔진의 설계

Design for the Multimedia Messaging Engine for Electronic Traffic Administration Services

궁상환*, 고현희**

천안대학교 정보통신학부*, 숙명여자대학교 컴퓨터과학과**

Sang-Hwan Kung(kung@cheonan.ac.kr)*, Hyon-Hee Koh(hhkoh@sookmyung.ac.kr)**

요약

향후 전자정부 시대에는 행정업무 담당자가 PDA와 같은 단말기를 이용하여 언제, 어디서나 업무처리가 가능하게 되고, 일반 사용자 역시 어느 장소에서나 최신의 정보 서비스를 제공받을 수 있는 편리한 시대가 도래할 것으로 예측되고 있다. 교통행정분야도 텍스트뿐 아니라 카메라 이미지, 동영상과 같은 디지털 자료를 이용함으로써 기관 업무 및 민원 서비스의 획기적인 개선이 이루어지고 있다. 이러한 전자교통행정 서비스의 공통된 요구사항 중의 하나는 다양한 멀티미디어 정보를 수시로 등록하고, 언제, 어디서나 다양한 단말기를 이용하여 구독해 볼 수 있도록 하는 미들웨어인 메시징(MOM: Message Oriented System) 서비스를 필요로 하고 있다. 이러한 배경으로 본 연구는 유선 및 무선의 분산 환경에서 수행되는 다양한 전자정부의 교통행정서비스를 지원하는 멀티미디어 컨텐츠의 유통시스템인 메시징 엔진 미들웨어 소프트웨어를 아키텍처 설계방법론에 따라 설계한 내용을 다루었다. 또한, 설계과정에서 비대칭 채널 패턴과 메시지 스위치 패턴, 쓰레드 풀 패턴의 3가지 아키텍처 패턴의 활용에 대해 설명하며, 아울러 다른 엔진과의 구조적 차이를 비교 및 평가를 수행하였다.

■ 중심어 : | 메시징 엔진 | 메시징 서비스 | 전자 교통 행정 서비스 |

Abstract

It may not be strange to anticipate that the forth-coming epoch of electronic government would bring us lots of convenience on account of ubiquitous environment with mobile devices. Such environment enables us to access the up-to-date information anytime and also a worker to carry out administration services to perform one's line of duty at anytime with electronic, handy devices such as PDA. What we now recognize from the field of electronic traffic administration services is that the office work flow as well as the public services has been remarkably improved by using of not only text but video and image information under the distributed, mobile environment. One of key requirements of this kinds of services is the feature of exchanging various information among multiple information publishers and subscribers. We call this feature Messaging Services generally. The study in this paper focuses on the architecture design of the Messaging Engine software, complying with the pattern-oriented software architecture design methodology. And our contribution also goes to the discovery of three architecture patterns found in the design process and the efficient multi-threading architecture compared to one of the Messaging Engine solutions.

■ Keyword : | Messaging Engine | Messaging System | Electronic Traffic Administration Services |

* 본 연구는 한국학술재단의 지원에 의하여 연구되었습니다.(KRF-2004-003-D00347)

접수번호 : #050516-002

접수일자 : 2005년 05월 16일

심사완료일 : 2005년 06월 20일

교신저자 : 고현희, e-mail : hhkoh@sookmyung.ac.kr

I. 서 론

인터넷의 확산 및 활용 증가와 함께 기업간, 개인간의 정보 교환이 빈번하게 이루어지고 있으며, 이러한 정보 교환을 지원하는 메시징 시스템의 요구도 점차 증대되고 있다. 대표적으로 B-to-B와 같이 기업간, 기관간 워크플로우를 기반으로 한 다양한 정보의 교환이나 차세대 뱅킹 환경에서 Front와 Back office간 메시징 시스템을 이용한 거래 정보의 교환, iCalendar, iCard와 같은 인터넷 상에서의 불특정 개인정보의 상호 교환, 무선 환경에서의 개인 간 채팅, 게임 등 다양한 정보의 교환 및 공유 등에서 메시징 시스템을 필요로 하고 있다.

이러한 기업간 거래나 개인 간의 정보교환을 위해 분산 환경에서의 소프트웨어 프로세스 간 다양한 데이터의 교환을 위해 미국 IBM 등에서는 메시징 시스템을 별도로 패키지화 하여 상품화하고 있다.

특히 메시징 엔진은 SMS(Short Message Service)와 같이 핸드폰 등 모바일 단말이 활발하게 이용되면서 더욱 필요성이 증대될 것으로 예상된다. 예를 들어, 향후 전자정부 시대의 교통서비스와 같은 경우, 정보의 전송 및 교환 서비스는 시간이나 장소에 불문하고 지속적으로 요구하게 될 것으로 예상되고, 특히 이동 중에 있는 버스, 승용차, 트럭 등 다양한 교통수단과 승객, 운전자와 같은 사람, 또한 거리 곳곳에 설치되어 있는 정류장 등으로 교통정보, 배차정보, 도착정보 등 다양한 정보가 실시간으로 전달되어야 할 것이다.

따라서 이러한 메시징 솔루션을 구축할 경우, 교통행정서비스의 전자화 및 자동화의 범위를 확장할 뿐 아니라, 여러 가지 다른 응용 서비스에도 다목적의 활용이 가능하다고 할 수 있다.

전자교통행정서비스의 공통된 요구사항 중의 하나는 다양한 멀티미디어 정보를 수시로 등록하고, 또한 등록된 정보를 언제, 어디서나 다양한 단말기를 이용하여 구독해 볼 수 있도록 하는 미들웨어인 메시징(MOM: Message Oriented System) 서비스를 공통적으로 필요로 하고 있다.

본 연구에서는 유선 및 무선의 분산환경에서 수행되는 다양한 전자정부의 교통행정서비스를 지원하는 멀티미디어 컨텐츠의 유통시스템인 메시징 엔진의 미들웨어 소프트웨어를 아키텍처 설계방법론에 따라 설계한 내용을 다룬다.

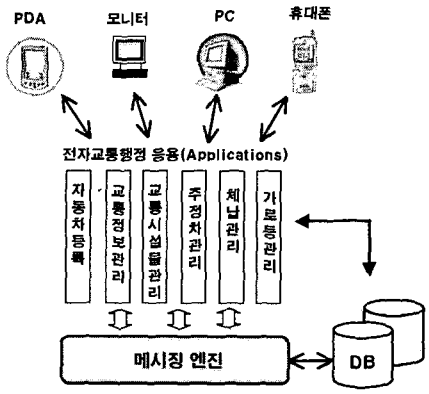


그림 1. 전자교통행정응용과 메시징 엔진

1. 연구 내용

본 논문의 구성은 다음과 같다. II장에서는 본 연구를 위한 관련연구로 메시징 시스템의 기능 비기능 요구사항을 식별하기 위하여 메시징 시스템 관련 기술 분석 및 국내외 메시징 시스템 사례 분석을 하였고, III장에서는 II장에서 분석된 요구사항을 기반으로 메시징 엔진의 구조설계 및 개발에 대한 연구 내용을 기술하였다. 특히, 메시징 엔진의 S/W 구조 결정은 프로그램의 구현 이후 성능 등 중요한 시스템의 품질을 좌우한다는 점을 고려하여, 예상 가능한 아키텍처 패턴을 분석하고, 프로토타이핑 시스템 구현 과정에서 대안 간의 상호 비교 및 평가를 통해 최적의 구조가 설계에 반영되도록 하였다. IV장에서는 메시징 엔진의 아키텍처 설계결과에 대한 평가를 갖고, 끝으로 V장에서는 향후 기술의 활용 방안이나 확장 연구에 대해서 기술한다.

II. 관련연구

1. 메시징 시스템

컴퓨터와 사람은 메시징 시스템을 사용하여 정보통신망을 통해 메시지를 교환함으로써 상호 대화를 할 수

있다. 오늘날 가장 많이 애용되는 다재다능한 메시징 시스템은 역시 떨어진 사람들 간 대화를 편리하고 용이하게 지원하는 e-mail이라고 할 수 있다. 그러나 e-mail은 사람과 사람 사이의 메시지 시스템으로 중요한 역할을 하지만 다른 위치에 있는 응용 소프트웨어 간 정보의 교환을 요구하는 경우를 위해 설계된 것은 아니다. 이와 같이 기업 시스템들 사이에 응용 소프트웨어 간 정보를 교환하는 데 사용되는 메시징 시스템은 일반적으로 기업 메시징 시스템 또는 Message-Oriented Middleware(MOM)이라고 부른다[12].

기업 메시징 시스템은 둘 이상의 응용이 메시지 형태로 정보를 교환할 수 있도록 해준다. 여기서 메시지는 기업의 자료와 통신망에서의 정보를 전달하는 라우팅에 대한 헤더 정보를 갖는 패키지이다.

기업 메시징의 중요한 개념은 메시지는 한 시스템에서 다른 시스템으로 메시지가 전달될 때 비동기식(asynchronous)으로 전달된다는 것이다[12]. 비동기식 전달방식에서 송신자는 송신된 메시지가 수신자에 의해 수신되거나 처리되기를 기다리지 않는다는 것이다. 따라서 메시지는 독립적인 단위로 간주되어야 하며, 각 메시지는 그 메시지를 처리하는 기업업무(business logic)가 필요로 하는 자료와 상태에 대한 모든 정보를 전달한다.

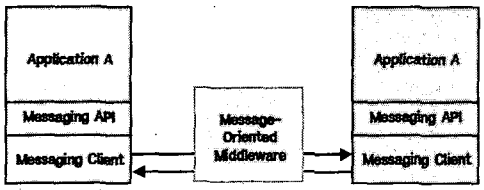


그림 2. Message-Oriented Middleware

1.1 메시징 시스템 구조

메시징 시스템은 중앙 집중식 구조와 분산형 구조로 나눌 수 있다. 중앙 집중형 구조를 사용하는 기업 메시징 시스템은 메시지 서버에 의존한다. 메시지 라우터 또는 브로커(broker) 라고도 불리는 메시지 서버는 한 메시징 클라이언트로부터 다른 메시징 클라이언트로 메시지를 전달하는 책임을 갖는다. 중앙 집중식 구조는 hop-and-spoke 토폴로지를 사용하며, 중앙에 메시지 서버와 이 서버에 모든 클라이언트가 연결된다.

분산형 구조는 통신망계층에서 IP 멀티캐스트를 활용하는 것이 현재의 추세이다. 멀티캐스트에 기반을 둔 메시징 시스템은 중앙 집중 서버를 갖지 않는다. 서버의 일부기능(자료저장, 거래처리, 보안)은 클라이언트의 로컬부분에 내장되는 한편 메시지 라우팅은 통신망 계층의 IP 멀티캐스트 프로토콜을 활용하여 그룹 내의 모든 클라이언트에게 메시지를 분배하는 형태로 운영된다.

1.2 메시징 서비스 모델

메세징 서비스에는 크게 publish-and-subscribe와 point-to-point의 두 가지 서비스 모델이 있다. publish-and-subscribe는 일대다 형태로 메시지를 분배하는 데 활용되며, point-to-point는 메시지의 일대일 전달을 목적으로 한다.

한편 메시지를 만들어 내는 메시징 클라이언트는 생산자(producer)로, 그리고 메시지를 수신하는 클라이언트는 소비자(consumer)라고 부른다. 하나의 메시징 클라이언트는 동시에 생산자와 소비자가 될 수도 있다.

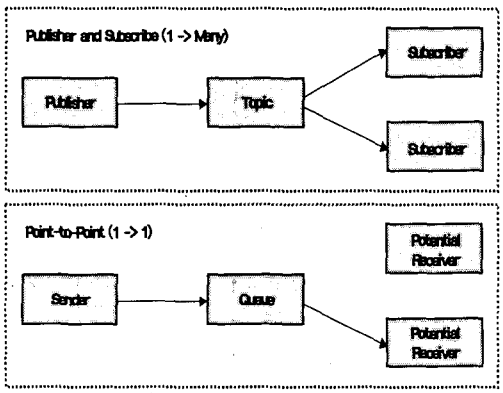


그림 3. 메시징 서비스 모델

publish-and-subscribe 모델에서 하나의 메시지 생산자는 topic이라고 불리는 가상채널을 통해 하나의 메시지를 여러 소비자에게 전송할 수 있다. 메시지를 수신하는 소비자는 특정한 topic을 구독하기 위해 선택할 수 있다. 특정한 topic으로 향하는 임의의 메시지들은 그

topic의 모든 소비자에게 전달된다.

point-to-point 모델에서는 메시징 클라이언트가 queue라고 불리는 가상채널을 이용하여 동기식뿐 아니라 비동기식으로 메시지를 송신하거나 수신하는 것이 가능하도록 한다. 이 모델은 전통적으로 pull- 또는 polling 기반의 모델이므로 메시지는 클라이언트에게 자동적으로 push 되는 대신에 queue로부터 요청을 받게 되면 전달된다.

2. 관련 연구 및 개발

JMS는 선 마이크로시스템사에 의해 만들어진 기업 메시징을 위한 Java API이다. JMS는 기본적으로 Connection, Session, Service라는 세 가지 연결계층을 제공하여 특정한 JMS의 응용이 요구하는 다중 접속 서비스를 제공하고 있다.

한편, JMS는 그 자체가 하나의 메시징 시스템은 아니며, 메시징 클라이언트가 메시징 시스템을 통해 통신할 때 필요로 하는 추상화된 인터페이스와 클래스라고 할 수 있다. 즉, JMS가 다양한 운영체제와 통신 인터페이스로부터 응용을 격리시켜 MOM 시스템들에 대한 접근을 추상화 하고 있기 때문에, 메시징 응용의 메시징 클라이언트들은 JMS를 사용하므로서 MOM 제품들 사이에서 포팅이 용이하게 된다[11].

외국의 메시징 시장은 IBM의 MessageQ뿐 아니라 Fiorano사의 FioranoMQ, Sonic사의 SonicMQ 등 이미 여러 회사의 제품이 상호 기능과 성능을 비교해 가면서 경쟁적으로 제품을 개발하고 있다[7].

SUN에서는 응용서비스들의 메시지 생성, 전송, 수신, 접근을 지원하기 위해 JMS 규격을 개발함과 아울러 *J2EE Connector Architecture*를 release하였고, IBM은 메시징 서비스를 위해 MQ Series 제품을 상품화하여 국내외 시장에서 활발히 활용되고 있다. Fiorano사는 JMS를 기반으로 한 최초의 상품화를 실시하여 FioranoMQ를 출시하고, AT&T의 무선통신, KPMG, Fed Ex, Motorola, JP Morgen 등의 기관에 판매하였다, Fiorano는 메시징 서버에 대한 대규모 동시 사용자의 접근을 지원하기 위해 Scalable 연결 관리 모듈을 개발하여 Pure JAVA 연결 관리 모듈의 adapter로 사용 가능하도록 개발하고 또한 IP 멀티캐스트가 가능한 근거리 통신망 환경에서 FioranoMQ 멀티캐스트 버전을 개발하였다. SoftWired사는 망간 연동 환경에서 페이저, 모바일 폰, PC, 워크스테이션에 이르는 다양한 JAVA 디바이스들의 실시간 처리 및 신뢰성 있는 멀티캐스트가 가능한 iBus 메시징 제품을 개발하였다[3].

이들 대부분의 상용 메시징 시스템은 2,000명 정도의 동시 사용자를 처리할 수 있는 시스템으로 미래에 예상되는 수천, 수만 명의 대규모의 동시 사용자가 접속할 경우 메시징 서버의 성능저하를 방지하기 위한 방법들이 현재 연구되고 있다.

Fiorano 등에서 성능 향상을 목적으로 사용하는 한 가지 방법은 쓰레드풀을 생성하고 관리하는 방법을 상요한다. 이것은 JAVA와 같이 멀티쓰레드 프로그래밍 환경에서는 클라이언트의 요청 시 마다 쓰레드를 생성할 경우 쓰레드 생성에 시간의 지연이 발생하는 문제점을 해결하기 위해 사용하는 방식이다. 즉, 미리 일정한 수의 쓰레드를 초기화 단계에서 생성하고 쓰레드의 생성이 필요한 때 마다 이미 생성된 쓰레드와 연결시켜 주는 방식이다[1][5].

한편, 국내에서는 ETRI에서 JMS 규격에 충실한 MoIM-Message 엔진을 개발하였다[4]. 이 엔진은 JMS의 클라이언트에 대해, 한 응용에서 단 하나의 객체생성 만을 허락하는 Singleton 디자인 패턴을 적용하여 유일한 클라이언트 서비스만을 받도록 하고 있다. 이것은 서버측에서는 클라이언트들이 많은 연결을 생성하는 것을 제한하도록 하여 전체적인 서버의 부하를 줄이고자 하는 것이다[8]. 또한 이 엔진에서는 각각의 Connection 별로 2개의 쓰레드를 제공하여 실시간 메시징 서비스가 가능하도록 하고 있다.

3. 소프트웨어 아키텍처 설계방법론

본 연구의 초점은 메시징 엔진에 적합한 소프트웨어 아키텍처를 설계하는 데 있다. 대표적인 소프트웨어 아키텍처 설계방법에는 ADD(Attribute-Driven Design)과 ABD(Architecture-Based Design)이 있다[6][9][10]. 두 방법론이 다소간의 차이가 있다 할지라고 궁극적인 방법론에는 별 차이가 없다고 하겠으며, 핵심적인 설계절차는 다음과 같다.

1) 분해할 모듈을 선택한다.
2) 선택한 모듈을 분해하고 정제한다.
   ① 아키텍처 드라이버에 따른 아키텍처 스타일을 선택한다.
   ② 스타일에 기능을 할당한다.
   ③ 품질속성을 점검한다.
3) 모듈의 분해가 필요치 않을 때까지 이상의 작업을 반복한다.

이 연구의 설계과정도 이러한 과정을 통해 설계되며, 설계의 결과로 새로 발견된 아키텍처 패턴의 새로운 패턴지식으로 DB화 된다.

III. 메시징 엔진의 아키텍처 설계

1. 메시징 엔징의 설계 개념

1.1 메시지 송수신 지원 모델

메시징 엔진의 가장 중요한 기능은 우선 송신자와 수신자 간의 메시지를 전달하는 서비스 즉, 메시지를 교환(switching)해 주는 기능이다. 특히 여기서 메시지를 교환하기 위해 어떠한 모델이 사용되는 가가 결국은 목표 엔진이 얼마나 응용에 효율적으로 편리하게 사용될 수 있는 것인가를 의미하므로 매우 중요한 요소가 된다. 본 연구에서 개발하고자 하는 메시징 엔진의 데이터 전송 유형이 [그림 4]에 도시되어 있다.

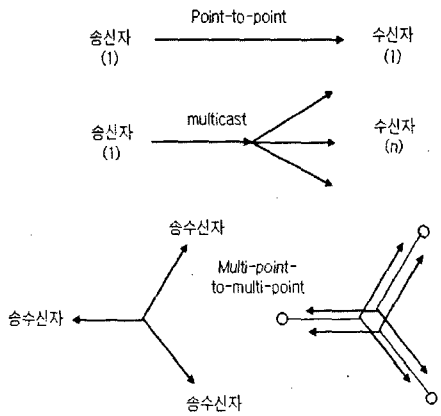


그림 4. 데이터 서비스 송수신 유형

송신자와 수신자 간 가장 간단한 서비스는 일대일(Point-to-Point) 데이터 전달 서비스이다. 다음의 서비스는 하나의 송신자의 정보를 그룹에 포함된 다수의 수신자가 동시에 데이터를 수신 받는 서비스이다. 특정 그룹에 속한 멤버들에게 방송형태의 데이터 서비스를 제공하는 경우이다. 개발엔진이 목표로 하는 최종 목표 서비스는 세 번째 그림인 다대다(multi-point-to-multi-point) 서비스이다. 이 서비스의 특징은 하나의 참여자가 다수의 송신자의 데이터를 하나의 접근점으로부터 수신할 수 있다는 점이다.

1.2 계층적 송수신 서비스 구조

앞서 설명한 송수신 유형은 기본적으로 다음의 4가지 계층을 토대로 한 구조 위에서 실현된다. 즉, 메시징 엔진에 포함되는 Connection(연결) 계층과 Session(세션)계층, Service(서비스) 계층, 그리고 메시징 엔진에서 제공하는 서비스를 이용하는 Application(응용) 계층이다. 이러한 계층구조는 메시징을 이용하는 다양한 응용을 효과적으로 지원하기 위해서이다.

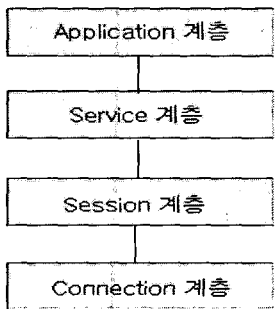


그림 5. 송수신 서비스의 계층구조

계층구조에 대한 이해를 돕기 위해 한 가지 응용 즉, 영상회의를 예시하면서 계층의 의미와 필요성을 살펴본다. 영상회의를 위해서는 영상이나 음성, 문자와 같이 몇 가지 미디어 정보가 동시에 전달되어야 한다. 이것은 하나의 멀티미디어를 위한 특정 세션에 여러 개의 미디어 서비스가 제공되는 것이라고 할 수 있다. 한편, 영상회의에는 미디어 정보 외에도 참가자들이 참가나 탈퇴 등과 같은 제어정보를 전달할 필요를 갖는다. 이것은 영

상제어 세션을 이용하여 서비스를 제공받을 수가 있는 것이다. 즉, 실제 데이터인 메시지는 서비스 계층을 이용하여 송수신된다는 것을 알 수가 있다.

그러나, 멀티미디어 정보전달을 위한 세션이나 또는 제어정보를 위한 세션은 모두 하나의 연결을 통해 이루어질 수 있다. 여기서 연결은 쉽게 일반적으로 통신에서 API로 활용되는 Socket을 이용하는 것이라고 할 수 있다. 다시 말해 하나의 연결은 하나의 Socket과 매핑된다고 할 수 있다. 물론 응용에서는 여러 개의 연결 즉, Socket으로 각각의 세션을 프로그램 할 수도 있다. 그러나 본 시스템의 기본 모델은 여러 개의 세션도 하나의 연결을 통해 쉽게 구현이 가능하도록 하고 있으며, 이것은 서버의 입장에서 볼 때 각각의 클라이언트가 여러 개의 연결 즉, Socket을 사용함으로써 통신자원을 낭비하게 되는 것을 방지하고자 모델에 반영하고 있는 것이다.

따라서 응용의 입장에서는 하나의 연결을 만들고, 연결 내에 하나 이상의 세션을 만들고, 다시 하나의 세션 내에 복수의 서비스를 만들며, 각각의 서비스 별로 메시지 즉, 데이터의 송신 또는 수신을 담당하도록 하는 것이다.

2. 메시징 엔진의 구조

2.1 구조 개요

메시징 엔진의 구조는 아키텍처 설계방법론에 따라 전체 시스템이 1차적으로 중요한 모듈로 분해되고, 각각각의 모듈을 위한 아키텍처 패턴을 적용하는 순서로 설계된다. 따라서 설계의 첫 작업은 메시징 엔진을 크게 서버 부분과 클라이언트 부분으로 구분하는 것으로부터 착수된다. 서버 부분은 메시징을 위한 고성능의 독립 컴퓨터에서 동작하는 소프트웨어인 반면, 클라이언트 부분은 PC나 PDA 등 단말에서 동작하는 소프트웨어이다. 서버에서 동작하는 구성 요소로는 연결 관리와 메시지 관리, 쓰레드 관리가 포함되며, 클라이언트에서 동작하는 구성요소에는 연결 관리와 메시징 서비스를 이용할 메시징 응용이 포함된다.

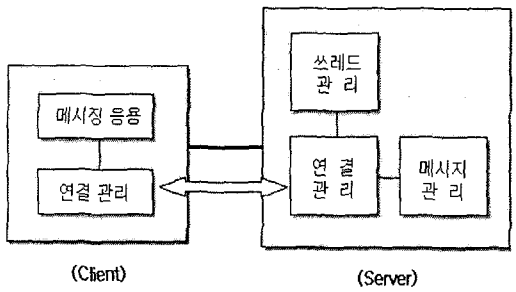


그림 6. 메시징 엔진의 구조

연결관리는 앞 절에서 설명한 메시지의 송수신 서비스를 계층적 구조로 구현한 것이며, 메시지 관리는 송신자로부터 입력된 메시지를 원하는 하나 또는 여러 수신자에게 배달해주는 기능이며, 쓰레드 관리는 쓰레드로 동작할 객체를 위한 쓰레드 풀(pool)을 생성하여 요구 객체에 반영해 주는 기능이다.

2.2 연결 관리

연결 관리는 기본적으로 계층적 송수신 서비스를 위해 클라이언트 측에는 Connection, Session, Service 객체를 운용하며, 서버 측에는 이에 대응하는 ServerConnection, ServerSession, ServerService 객체들이 상호 대칭되어 peer간 통신모델을 운영하게 된다.

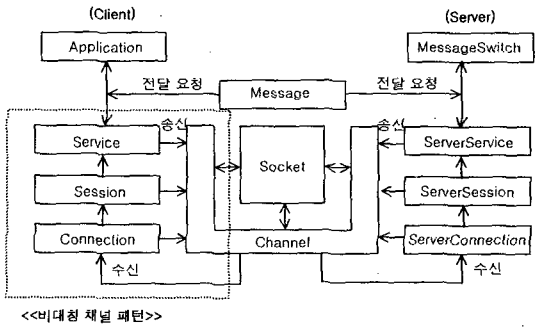


그림 7. 연결 관리와 관련 클래스

Connection과 ServerConnection은 Channel이라는 객체를 이용하여 메시지 스트림을 송수신한다. 이 Channel은 하나의 Socket을 이용하여 연결을 설정하며, 메시지를 교환하기 위해 각 Socket에 대한 입력스트림과 출력스트림 객체를 생성하여 메시지의 통신에

활용한다. 특히 메시지 수신의 항상 수신 상태로 대기해야 하기 때문에 Channel은 ChannelListener라는 쓰레드로 구현된 객체를 포함하고 있다.

Session과 ServerSession은 임의의 Connection과 ServerConnection 객체가 생성된 이후 그 안에서 생성된다. 다만 이 세션은 별도의 능동적인 동작을 할 필요가 없고 통신을 지속적으로 사용해야 하는 필요성이 없기 때문에 쓰레드의 형태로 동작하지는 않는다. 한편, 세션은 복수의 서비스 즉, 데이터 서비스를 위해 세션에 포함되는 메시지를 Message ID라는 표시자를 사용하여 관리한다. 따라서 동일한 메시지 그룹으로 전달 또는 수신하고자 할 때 동일한 메시지 ID를 가짐으로써 다른 메시지 그룹과 구분하게 되는 것이다.

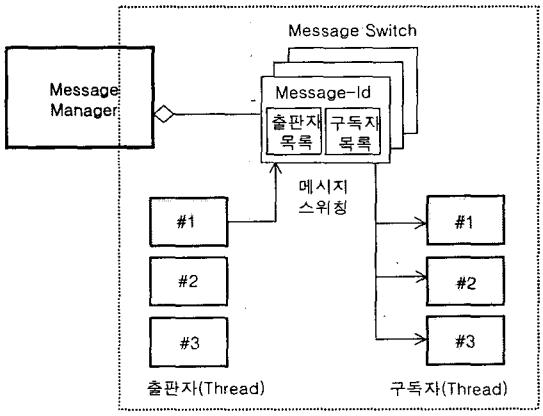
Session과 ServerSession 안에서 생성되는 Service와 ServerService 객체는 정보를 제공하는 역할(Publisher)과 정보를 제공받는 역할(Subscriber)의 객체로 구분되어 운영된다. Server와 ServerService 객체는 데이터의 송수신을 위한 함수를 제공하고 있고, 지속적인 정보의 송수신에 활용되기 때문에 활성화되어 있는 기간 동안 쓰레드로 동작하게 된다.

연결 관리를 도시하는 그림에서는 메시지의 송신과 수신이 서로 다른 방법으로 이루어짐을 보여 준다. 즉, 송신의 경우는 Channel을 통해 직접 Socket의 write 기능을 이용하지만, 수신의 경우는 Connection/SeverConnection, Session/ServerSession, Service/ServerService 객체를 통과하면서 순서적으로 전달된다. 이것은 수신의 경우에서는 이러한 계층적인 연결서비스가 1:1로만 국한되지 않으며, 1:다의 형태로 제공하기 때문이다.

2.3 메시지 관리

메시지 관리는 메시징 클라이언트간 송수신 되어야 하는 메시지를 저장하고 전달하는 역할을 의미한다. 특히 메시지 송신자와 수신자 간에 원하는 Message-ID를 교환해 주는 기능을 위해 모든 메시지를 Message-ID 별로 구분하여 MessageEntry 내에 보관한다. 여기서 보관의 의미는 임시적인 저장이나 전달을 의미한다. MessageEntry에는 ID 외에 메시지 출판자와 메시지 구독자 객체의 주소를 보관한다. 즉, 하나의 출판자가 전달하고자 하는 메시지가 있으며, MessageEntry 안의 모든 구독자에게 해당 메시지를 전달하게 된다. 따라서 하나의 출판자가 정보를 모든 구독자에게 전달/분배하는 동안 다른 출판자는 로킹(locking) 상태에 있게 된다.

MessageEntry 내에 출판자와 구독자의 객체 정보를 관리하는 또 다른 이유는 이 엔트리의 수명과도 관련이 있다. 즉, 하나의 출판자 또는 구독자가 있는 동안만 이 엔트리는 보관된다. 하나의 엔트리가 메시지 관리자에 의해 메시지 테이블에 등록된 이후 각각의 출판자와 구독자들이 등록되지만, 반대로 각각의 출판자와 등록자가 서버로부터 빠져 나가게 되어 최종 하나의 출판자 또는 구독자도 없게 되면 관련된 엔트리는 메시지 테이블에서 제거되는 결과를 초래한다.



<<메시지 스위치 패턴>>

그림 8. 메시지 관리

2.4 쓰레드 관리

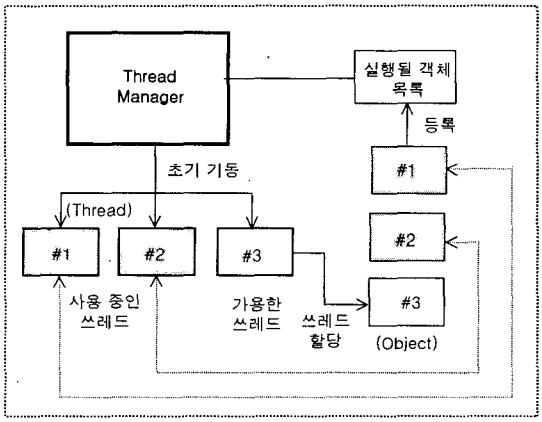
메시징 엔진의 설계에서 경제적인 측면과 성능을 고려한 결과, 쓰레드의 형태로 능동적인 동작을 수행하는 객체에는 ChannelListener와 서버에서 동작할 ServerService(출판자 또는 구독자)가 있다.

메시징 엔진에서 수용해야 할 클라이언트의 규모가 커질수록 고려해야 되는 한 가지는 메시징 서비스를 위한 성능 요구사항이며, 성능 품질은 쓰레드의 생성과 관련이 있다. 이것은 쓰레드를 생성하는 데 소요되는 시간

으로 인해 메시징 서비스의 지연을 초래하는 요인이 되기 때문이다.

통상 쓰레드를 생성하는 데는 일정한 시간이 소요된다. 따라서 만약 메시징 엔진이 필요로 하는 쓰레드를 모두 미리 생성하고, 쓰레드로 동작할 객체를 이 쓰레드에 할당해 준다면 전체적인 성능 개선에 매우 도움이 될 것으로 본다.

[그림 9]는 본 연구에서 활용하는 쓰레드 풀의 관리방법을 설명하고 있다. 쓰레드 관리자는 메시징 엔진의 파라미터로 관리되는 개수만큼의 쓰레드를 메시징 서버의 기동시 미리 생성해 둔다. 그리고 클라이언트가 메시징 서비스를 요청할 때 마다 해당 객체를 쓰레드로 기동될 객체 목록에 등록을 시킨다. 쓰레드 관리자는 가용한 쓰레드의 여유가 있을 때 순서적으로 객체 목록에서 객체를 호출하여 기동 중인 쓰레드로 동작시키는 것이다.



<<쓰레드 풀 패턴>>

그림 9. 쓰레드 관리

이러한 구조의 결정을 위한 두 가지 실험이 선행되었다. 한 가지는 쓰레드 생성시간의 측정을 위한 실험이며, 다른 한 가지는 설계패턴에 대한 대안의 제시와 성능실험을 통한 대안의 선정이다.

후자의 설계패턴에 대한 대안은 크게 세 가지로, 첫째는 쓰레드를 사용하지 않는 Event-dispatching 패턴과 쓰레드를 활용하는 Thread-per-message 패턴, 그리고 쓰레드를 미리 생성하는 Thread-pool 패턴이다. 이 세 가지의 패턴의 실험결과, 세 번째의 Thread-pool 패턴이 가장 우수한 성능을 보여 주고 있음이 확인되었다.

3. 메시지 형식의 설계

메시징 엔진은 다양한 메시지를 이용하여 궁극적인 메시징 서비스를 제공한다. 메시징 엔진이 활용하는 메시지는 크게 제어정보를 수록하는 메시지와 데이터 정보를 수록하는 메시지로 구분된다. 다만 이러한 모든 메시지는 Message라는 하나의 최상위 객체로부터 상속을 받는다.

[그림 10]에서 중요한 제어정보와 이러한 제어 메시지의 활용을 Connection, Session, Service의 생성 및 종료와 관련하여 도시한다.

이외에도 데이터 메시지는 일종의 Payload 형태의 정보이며, 기본적으로 세 가지 유형의 정보를 수록한다. 본 설계 엔진에서는 TextMessage, ByteMessage, ObjectMessage를 지원하는 데, 이 세 가지 유형은 대부분의 유용한 정보를 수록하는 데 충분하리라고 사료된다.

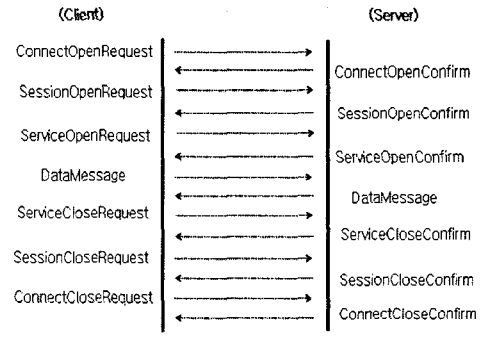


그림 10. Client-Server 통신 프로토콜

V. 연구 결과에 대한 평가

지금까지 일대일뿐 아니라 다자간 데이터 통신을 제공하는 메시징 엔진의 설계과정과 설계결과에 대한 설명을 하였다. 이 엔진은 기 개발된 메시징 규격을 완벽히 충족하고 있지는 않지만, 오히려 핵심적인 개념설계를 토대로 구현함으로써 다른 응용과 통합되어 활용될 때 이용의 편리성과 높은 성능 품질을 기대할 수 있다는 이점을 제공한다고 할 수 있다.

연구결과의 상대적 평가는 ETRI에서 개발된 MoIM-

message와의 비교를 통해 정리하면 다음과 같다.

표 1. 메시징 엔진 비교평가

	Thread 생성		Thread Pool
	클라이언트	서버	
MoIM-Message	Connection 마다 2개	클라이언트마다 2개	지원 안함
교통행정 메시징엔진	Connection 마다 1개, 단, Service는 응용이 선택	Connection마다 1개, Service 마다 1개	지원 함

[표 1]에서 보는 바와 같이 교통행정 메시징 엔진은 MoIM-Message과 비교할 때 Thread Pool을 지원하여 대규모 클라이언트에서 성능에서 커다란 차이를 보인다. 설계 엔진의 다른 객관적 평가는 MoIM-Message가 클라이언트와 서버 측에서 하나의 Connection 마다 두개의 쓰레드를 생성하는 것에 비해 기본적으로 Connection마다 하나의 쓰레드를 생성하여 쓰레드 생성의 부하를 줄여 준다는 장점이 있다. 특히 이 엔진은 클라이언트와 서버에서의 쓰레드 생성 전략을 달리하고 있는 데 서버의 경우는 클라이언트가 Connection에 속한 Service를 open할 때마다 이것을 쓰레드로 서비스하여 최대의 성능을 보장하는 것이고, 클라이언트의 경우는 클라이언트의 형편에 따라 쓰레드로 생성하거나 아니면 그냥 객체의 형태로 서비스를 받게 할 것이냐를 응용 프로그램에서 선택할 수 있도록 하였다. 이것은 클라이언트가 PC와 같이 성능이 양호한 환경에 있을 수도 있고 또한 핸드폰이나 PDA처럼 보다 열악한 환경에 있을 수 있게 때문에 환경에 따른 최적의 성능을 보장하기 위한 전략이라고 하겠다.

이러한 연구 성과 외에도 개발된 엔진을 이용하여 얻을 수 있는 기대 성과를 살펴보면 다음과 같다.

기술적 측면의 기대 효과로 첫째 메시징 엔진을 통한 기술적인 이점은 분산 환경에서 유선 단말이나 무선 단말을 이용하여 정보를 교환해야 하는 다양한 메시지 송수신 응용서비스들을 신속하게 개발할 수 있도록 지원한다.

둘째 이제까지의 메시징 엔진은 주로 주식 정보와 같은 텍스트 정보에 국한하였으나 본 연구에서는 텍스트 뿐 아니라 영상 및 음성을 포함하는 멀티미디어 정보의 전송에 이용이 가능하도록 복수의 미디어 채널과 제어 채널이 연계된 프로그래밍이 가능하다.

셋째 본 연구에서 발견된 비대칭 채널 패턴과 메시지 스위치 패턴, 쓰레드 풀 패턴의 3가지 아키텍처 패턴은 유사한 응용의 개발에 많이 활용될 수 있다는 점에서 또 다른 기여가 될 수 있다고 하겠다.

V. 결 론

본 논문에서는 유선 및 무선의 분산 환경에서 수행되는 다양한 전자 정부의 교통 행정 서비스를 지원하는 멀티미디어 컨텐츠의 유통시스템인 메시징 엔진 미들웨어 소프트웨어를 개발을 위한 아키텍처 설계를 모델링하였고, 설계 엔진에 대한 평가와 분석을 수행하였다.

설계된 엔진은 메시지 버퍼관리에 대한 부분이 설계되어 있지 않은 데, 설계전략에는 이 버퍼관리 즉, 메모리 버퍼나 파일 버퍼에 대한 저장관리 부분을 엔진의 내부 기능으로 보지 않고, 또 다른 하나의 메시징 응용으로 보아 별도로 설계한다는 개념이 포함되어 있다. 즉, 모든 중요한 모듈을 컴포넌트화 하고 패턴화 한다는 취지이며, 아울러 핵심엔진은 가장 경량화 시키고자 한다는 목적을 실행한 것이다.

본 연구의 결과는 다양한 교통행정이나 기업 메시징 응용을 위한 메시지 교환 엔진으로 활용될 수 있을 것으로 사료된다. 또한, 컴퓨터와 인터넷을 이용한 자료의 제출, 수집, 처리 등과 같은 웹 환경에서의 자유로운 문서의 출판 및 구독 서비스에 활용 가능하고, 직접적인 응용서비스, 예를 들어 교통행정 서비스나 메시지 단문 서비스, 주가변동정보 통보서비스와 같은 응용서비스와 통합하여 최신의 정보를 알려 주는 서비스에도 이용이 가능할 것으로 예상한다. 아울러 인터넷 방송, 인터넷 교육 등의 실시간 동영상 서비스에 멀티미디어 정보 전송을 위한 스트리밍 엔진으로도 활용 가능하며, 특히 광범위한 VOD(Video On Demand) 서비스를 위한 컴퓨터기반 라우터 서비스에도 활용할 수 있다.

참 고 문 헌

[1] 진광일, 박성철 역, "*자바스레드*", 한빛미디어, 2000.

[2] 천영환 역, "*자바스레드*", 인포북, 2001.

[3] 한국전자통신연구원 연구보고서, "*모바일 인터넷 환경에서 Dynamic, Scalable 메시지 성능에 관한 연구*", 2002.

[4] 한국전자통신연구원 연구보고서, "*신뢰성 보장 이동분산처리기술 개발*", 2002.

[5] 홍상욱 역, "*자바 퍼포먼스 튜닝*", 한빛미디어, 2001.

[6] L. Bass, P. Clements, and R. Kazman. *"Software Architecture in Practice"*, Addison Wesley, 1998.

[7] *"Benchmarking JMS Based E-Business Messaging Providers"*, Java Developer's Journal, Mar. 2001.

[8] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *"Design Patterns: Elements of reusable object-oriented software"*, Addison-Wesley professional computing series, Massachusetts, 1995.

[9] F. Bachmann, L. Bass, G. Chastek, P. Donohoe, and F. Peruzzi, *"The Architecture Based Design Method"*, Software Engineering Institute, Carnegie Mellon University, Technical Report CMU/SEI-2000-TR-001, ESC-TR-2000-001, 2000.

[10] L. Bass and R. Kazman, *"Architecture-Based Development"*, Software Engineering Institute, Carnegie Mellon University, Technical Report CMU/SEI-99-TR-007, ESC-TR-99-007, 1999.

[11] P. Giotta, S. Grant, M. Kovacs, S. Maffies, K. Morrison, G. Raj, and M. Kunnumpurath, *"Professional JMS Programming"*, 2000.

[12] R. Monson-Hoefel and D. A. Chappel, *"Java Message Service"*, O'Reilly, 2001.

저 자 소 개

궁 상 환(Sang-Hwan Kung) 정회원

- 1977년 2월 : 숭실대학교 전자계산학과(이학사)
- 1983년 2월 : 고려대학교 전자정보차리학과(경영학석사)
- 1998년 2월 : 충북대학교 전자계산학과(이학박사)
- 1981년 : 제2군수지원사령부 제원처리실 프로그램장교
- 1998년 : 한국전자통신원 책임연구원
- 1997년 : 미국 스탠포드 연구소(SRI) 초빙연구원
- 2000년 : 중소기업청 정보화지원과 전산사무관
- 2000년 : 미국 카네기 멜론 대학교 MSE 과정 수료
- 2001년~현재 : 천안대학교 정보통신학과 조교수

<관심분야> : 소프트웨어 구조, 분산시스템

고 현 희(Hyon-Hee Koh) 준회원

- 1992년 2월 : 숙명여자대학교 전자계산학과(이학사)
- 2000년 2월 : 숙명여자대학교 컴퓨터과학과(이학석사)
- 2005년 8월 : 숙명여자대학교 컴퓨터과학과(이학박사)
- 1993년 : 코오롱 정보통신 연구원
- 1999년 : LG산전연구소 주임연구원
- 2002년 : LG전자 정보통신 연구소 선임연구원
- 2002년 : 미국 카네기 멜론 대학교 MSE 과정 수료
- 2001년~현재 : 숙명여자대학교 대학원 컴퓨터과학과 박사과정

<관심분야> : 소프트웨어 아키텍처, 설계 패턴