# 지식관리시스템 연계를 위한 미들웨어 기반 다중 전송 서비스

남덕윤° 김평 최기석 이동만+
한국과학기술정보연구원
+한국정보통신대학교
{dynam°, pyung, choi}@kisti.re.kr, dlee@icu.ac.kr

# Middleware-based multipoint-to-multipoint service for the integration of knowledge management systems

Dukyun Nam°, Pyung Kim, KiSeok Choi, and Dongman Lee+
Korea Institute of Science and Technology Information
+Information and Communications University

요 약

지식관리시스템은 기관 및 회사들에서 자신들의 조직에 맞는 시스템으로 구축하여 기사용하고 있다. 조직별 지식관리시스템에서는 하나의 조직에서 발생할 수 있는 지식만을 축적하고 있다. 최근 지식 고도화 및 혁신사업의 하나인 '지식관리'를 위해 다른 조직의 지식관리시스템과의 연계를 통한 지식 공유 및 기존 지식들에서 양질의 지식을 창출할 수 있는 체계적 지식관리시스템 구축에 주안점을 두고 있다. 본 논문에서는 별도로 개발된 다수의 지식관리시스템 사이의 연계를 위한 통신 방식을 살펴보고, 시스템 구축의 편의를 도모할 수 있는 미들웨어인 CORBA와 Java Message Service (JMS)를 활용한 메시지 다중 전송 서비스 연계방식을 제안한다.

## 1. 서 론

지식관리시스템은 다수의 기관 및 회사들에서 자신들의 조직에 맞는 시스템으로 구축하여 이미 사용하고 있다. 조직별 지식관리시스템에서는 하나의 조직에서 발생할 수 있는 지식만을 축적하고 있는 반면, 지식 고도화를 위해서는 다른 조직의 지식관리시스템과의 연계를 통해, 지식을 공유하고 기존 지식들에서 양질의 지식을 창출할 수 있는 범조직적인 지식관리시스템 구축에 대한 필요성이 대두되고 있다.

별개의 시스템들을 통합하는 데 있어서 어려운 점은 기존의 시스템들이 기개발되어 있고, 통합을 위해 기존 시스템들이 수정되지 않아야 한다는 점이다. 이에 시스템의 하드웨어와 소프트웨어의 이종성(Heterogeneity)에 대해 별다른 고민없이, 쉽게 통합할 수 있는 방법이 분산 미들웨어(Distribution middleware)[1]를 사용하는 것이다. 미들웨어는 운영체제와 응용프로그램 사이에 위치하며, [1]에서 미들웨어를 호스트 인프라스트럭처 미들웨어 (Host infrastructure middleware), 분산 미들웨어 (Distribution middleware), 공통 미들웨어(Common middleware), 도메인별 미들웨어 (Domain-specific middleware)로 구분하고 있다. 이 중에서 분산 미들웨어는 프로그래밍 언어, 운영체제 플랫폼, 통신 프로토콜, 하드웨어에 대한 의존성 없이 목적지의 객체를 호출할 있게 해주며, 분산 미들웨어가 우리가 일반적으로 알고 있는 통상의 미들웨어이다. 예를 들어, CORBA[2], Java RMI[3], Microsoft DCOM[4] 등이 있다. 미들웨어를 사용하여 기존의 지식관리시스템 사이의 통신을 하게 되면, 소켓과 같은 네트워크 인터페이스를 활용한 세부 구현에 대해 신경 쓸 필요가 없다.

지식관리시스템 사이의 통신을 통해, 최종적으로 이루고자 하는 것은 서로 공유하고자 하는 내용을 보다 효율적으로 교환하는 것이다. 가장 초보적인 방법은 각 시스템 별로 일대일 통신을 함으로써 정보를 교환할 수 있을 것이다. 그러나 다수의 시스템 사이에서 정보 교환이 이루어지고 공유된다는 점에서, 그룹 통신 서비스 (Group communication service)[5]를 활용한다면, 우리는 보다 효율적인 정보 공유를 할 수 있을 것이다. 그룹 통신 서비스는 여러 개의 프로세스들을 하나의 프로세스 그룹으로 이용할 수 있도록 함으로써, 결함 감내(Fault tolerance)와 고 가용성(High availability)을 향상시켜 주는 유용한 메커니즘으로 잘 알려져 있다 [6, 7]. 그룹 통신 시스템의 중요한 역할은 그룹의 멤버들이 공유하고 있는 그룹 상태 정보의 일관성을 유지하는 것이다.

분산미들웨어에 그룹통신을 지원하는 통신 방식으로 CORBA에 그룹통신 시스템을 지원하는 방식[8]과 Publish/Subscribe 방식[9]의 Topic 기반 자바 메시지 서비스 (Java message service; JMS)[10]를 들 수 있다.

본 논문에서는 지식관리시스템 사이의 정보 공유를 위해 미들웨어를 활용한 다중 전송 서비스로 그룹 통신을 어떻게 지원하는지 살펴보고, 기존의 지식관리시스템 사이의 통합에는 어떤 방식이 현실적인지 토론한다.

## 2. 관련연구: 그룹통신

그룹 통신 서비스는 멤버쉽 서비스와 신뢰적 다중 전송 서비스로 구성된다. 멤버쉽 서비스는 그룹을 형성하고 있는 멤버간의 일관성을 보장하기 위해, 멤버들의 모임인 뷰(view)로 멤버쉽을 관리한다. 즉 멤버들이 결함이 나거나, 네트워크 단절이 있는 경우에 이를 반영함으로써, 실제 동작하고 있는 멤버들로 이루어진 뷰를 모든 멤버들에게 알려주는 것이다. 다중 전송 서비스에서는

메시지 순서화 (Message ordering)을 지원하며, 이를 통해 멤버들 간의 상태 일관성을 지원한다. 그룹 통신 서비스의 수행은 1) 그룹 구성과 2) 메시지 전송으로 나눠볼 수 있으며, 본 논문에서는 이에 대해 실제 적용 방식을 설명코자 한다. 그룹 통신 서비스에 대한 보다 자세한 내용은 [11]에서 확인할 수 있다.

## 3. CORBA 기반 다중 전송 방식

그룹 통신 서비스는 객체 복제를 지원하는.주요 기술 중 하나이다. CORBA Rev. 2.3[2]에서는 그룹통신을 지원하는 것은 CORBA가 기본적으로 RPC (Remote procedure call) 방식 즉, 동기화 통신을 지원하기 때문에, 그룹통신 지원을 위해서는 확장이 필요했다. RPC(Remote procedure call) 모델은 로컬 호스트의 프로세스와 상호작용(interaction) 하는 것처럼, 원격 호스트와 상호작용 하도록 지원하는 모델로, 동기화 통신(Synchronous communication)을 기본으로 함으로써, 송신자는 수신자로부터 응답이 있을 때까지 아무 것도 수행하지 않은 상태(block)로 기다려야 한다는 단점이 있다.

이에 Fault tolerant CORBA (FT-CORBA)[11]에서 그룹 통신 지원에 대한 인터페이스를 정의했으며, 본 논문에서는 [8]에서 제안한 FT-CORBA compliant 구조 기반의 그룹 구성과 메시지 전송을 간략히 설명하고자 한다.

### 3.1. 그룹 구성

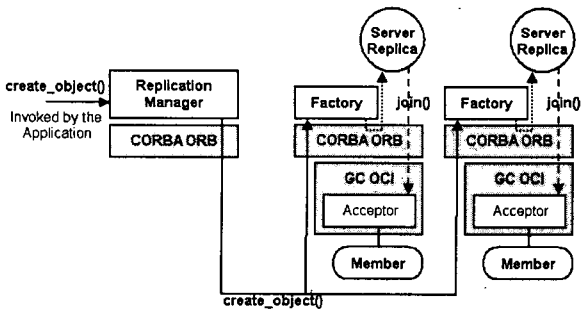


그림 1. CORBA 객체의 그룹 구성.

그림 1에서 서버 복제 객체(Server replica)들이 하부 그룹 통신 서비스의 프로세스 그룹 멤버와 일대일 대응이 됨으로써, 그룹통신에서의 멤버쉽 서비스를 활용함을 보여준다. 이를 통해 CORBA 기반의 그룹 구성은 멤버 객체 각자가 동일한 멤버 리스트를 파악하고 있다. 멤버가 자신이 속한 그룹의 멤버들을 알고 있다는 점이 다음 장에서 설명할 JMS 기반의 그룹 구성 방식과 가장 큰 차이점이다.

### 3.2. 메시지 전송

그림 2는 클라이언트의 객체는 하부 그룹통신 서비스를 활용하여, 서버 객체들에게 다중 전송을 하는 그림으로, FT-CORBA 명세서에서 게이트웨이를 활용하지 않은 메시지 전송 방식이다.

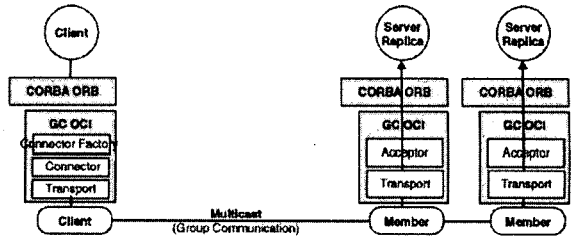


그림 2. CORBA 객체 그룹에 대한 메시지 전송.

## 4. JMS 기반 다중 전송 방식

JMS의 Topic 기반 Publish/Subscribe 방식 또한 그룹 개념을 사용함으로써, 그룹 통신[5]을 지원한다. RPC 방식과 달리 Publish/Subscribe 방식은 비동기화 통신(Asynchronous communication)을 지원한다. 일반적으로 Publish/Subscribe 모델에서는 메시지를 송신자를 Publisher, 메시지를 수신자를 Subscirber라고 하나, 본 논문에서는 CORBA 기반의 방식과의 일관성을 위해, 클라이언트와 서버로 지칭한다.

### 4.1. 그룹 구성

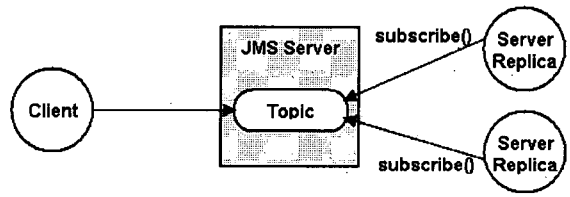


그림 3. JMS Topic을 활용한 그룹 구성.

멤버 관리에 있어서, CORBA 기반의 그룹 통신과의 차이점은 종단(End host)인 클라이언트와 서버들은 자신이 속한 그룹에 누가 있는지를 모른다는 점이다. JMS 제공자(vendor)에 따라 다르긴 하지만, 대부분은 중앙의 JMS 서버에서는 누가 가입되어 있는지 알 수 있다. 그림 3은 메시지를 받고자 하는 멤버들이 JMS 서버의 해당 Topic을 구독 신청함으로써, 그룹 구성이 이루어짐을 보여준다.

### 4.2. 메시지 전송

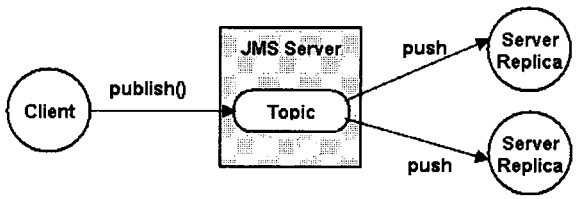


그림 4. JMS Topic을 통한 메시지 전송.

그림 4는 클라이언트가 JMS Topic에 전송하고자 하는 메시지를 발행함으로써, 구독 신청이 되어 있는 멤버들에게 JMS 서버가 전달해 주는 것을 보여준다.

## 5. 토론

지금까지 CORBA와 JMS를 활용한 그룹 구성 및 다중 전송에 대해 살펴보았다. 본 절에서는 이들의 장단점에 대해 결함 감내 및 일관성 보장의 관점에서 토론한다. 앞서 설명한 두 방식을 편의상 CORBA 방식, JMS 방식으로 지칭하고자 한다.

### 5.1. 결함 감내

멤버인 호스트가 동작하지 않는 경우에 어떻게 되는지 살펴본다. CORBA 기반 그룹 통신에서는 멤버쉽 관리를 하고 있기 때문에, 결함이 난 멤버를 제외한 멤버들끼리 다시 멤버쉽을 구성한 후, 동작한다. 결함났던 멤버가 복구되었을 시에는 그룹에 재가입하게 되며, 그룹 탈퇴 기간동안 전달 받지 못한 메시지에 대해서는 개별의 메시지들이 아닌, 상태 전송(state transfer)[13]을 통해, 응용 프로그램의 상태를 전달받는다.

JMS 방식에서는 Durable subscriber로 Topic 구독을 한 경우, 복구된 멤버에 대해 지원한다. 멤버쉽을 멤버 개개인이 관리하는 것이 아니기 때문에, 결함이 발생한 멤버에 대해 다른 멤버들은 신경쓸 필요가 없다. 결함 멤버가 복구되어 재가입하게 되면, 서버에서는 보관중인 메시지들을 전달해 준다. 한편 JMS 방식에서는 JMS 서버를 활용하기 때문에, 중앙 서버가 결함 나는 경우에는 "single point of failure"이다. 물론 중앙 집중식의 공통된 문제로, 이를 피하기 위해 다중 서버를 구성하는 노력이 필요하며, 로그(log)를 남기고, 별도의 방법으로 서버를 복구 하는 방식이 있어야 한다. 이와 같은 결함 감내를 위한 방법은 JMS 제공자(vendor) 별로 지원하며, JMS 명세서에서는 기술하고 있지 않다.

결론적으로 결함감내에 대해서는 두 방식 모두 기술적인 지원이 가능하다고 판단된다.

### 5.2. 일관성 보장

CORBA 방식에서는 각 호스트들이 그룹을 구성하고 있는 멤버들을 모두 알고 있는 반면, JMS 방식에서는 제공자 및 가입자들은 통신되는 내용이 누구한테 오는 것인지, 누구에게 가는 것인지 알지 못하며, 다만 JMS 서버에서 제공자 및 가입자를 알 수 있다.

별도로 관리되고 있는 종단의 지식관리시스템에서는 허가 받은 기관의 시스템으로 내용이 전달된다는 것만 보장된다면, 매 전송마다 누구에게 보내지는지는 관심이 없다. 또한 CORBA 방식처럼 그룹 가입자에게 메시지 전달이 되었는지 확인해야 하는 부담을 각 시스템들이 지고 싶어 하지 않는다. 이에 메시지 전달 및 일관성 보장 측면에서는 중앙집중식의 JMS 서버에서 책임지는 것이 보다 효율적인 방법이 될 수 있다.

## 6. 결론

본 논문에서 우리는 기개발된 지식관리시스템 사이의 지식 공유를 위한 미들웨어 기반의 다중 전송 서비스에 대해 살펴보았다. CORBA 방식에서는 그룹 멤버쉽을 각 멤버들이 파악하고 있고, 멤버들로의 메시지 전송을 책임짐으로써 상태 일관성을 보장하는 반면, JMS 방식에서는 JMS 서버에서 멤버 관리 및 메시지 전달을 보장하고 있다.

각 기관들의 지식관리시스템에서는 다른 시스템과의 연계를 위해 자신들의 시스템이 수정되는 것을 바라지 않으며, 연계에 대한 책임을 꺼린다. 이에 가장 현실적이면서 효율적인 구현 방식은 공개 데이터에 대한 별도의 데이터베이스를 제공하고, JMS를 통해 지식관리시스템들을 연계함으로써, 지식 공유를 실현하는 것이다.

## 참고문헌


[1] R.E. Schantz and D.C. Schmidt, "Middleware for Distributed Systems: Evolving the Common Structure for Network-centric Applications," *Encyclopedia of Software Engineering*, Wiley & Sons, 2001.

[2] Object Management Group, The Common Object Request Broker: Architecture and Specification, Rev. 2.3, OMG Document formal/98-12-01, June 1999.

[3] *Java Remote Method Invocation Specification*, Sun Microsystems, 2000.

[4] M. Horstmann and M. Kirtland, DCOM architecutre, 1997. Avaiable online at www.microsoft.com/com/tech/DCOM.asp.

[5] D. Powell (ed.), "Group Communication," Special Issue on Group Communications, *Communications of the ACM*, 49(4), April 1996.

[6] K. Birman, "The Process Group Approach to Reliable Distributed Computing," *Communications of the ACM*, 36(12): 37-53, December 1993.

[7] K. Birman, "A Review of Experiences with Reliable Multicast," *Software-Practice and Experience* 29(9): 741-774, July 1999.

[8] D. Lee, D. Nam, H.Y. Youn, and C. Yu, "OCI-based Group Communication Support in CORBA," *IEEE Transactions on Parallel and Distributed Systems*, 14(11): 1126-1139, November 2003.

[9] P.Th. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys*, 35(2): 114-131, June 2003.

[10] *Java Message Service Specification, ver. 1.1*, Sum Microsystems, April 2002.

[11] G.V. Chockler, I. Keidar, and R. Vitenberg, "Group communication specifications: a comprehensive study," *ACM Computing Surveys*, 33(4): 427-469, December 2001.

[12] Object Management Group, *Fault Tolerant CORBA*, OMG Document formal/2001-09-29, Sep. 2001.

[13] Y. Amir, G.V. Chockler, D. Dolev, and R. Vitenberg, "Efficient State Transfer in Partitionable Environments," in *Proceeding of the European Research Seminar in Advanced Distributed Systems (ERSADS'97)*, pages 183-192, Zinal, Valais, Switzerland, March 1997.