제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

Clustered EJB 서버에서의 멀티캐스트 프레임워크 연구

김수형, 정승욱, 서범수, 노명찬, 김중배
한국전자통신연구원 전자거래연구부
e-mail : {lifewsky, swjung, bsseo, mcroh, jjkim}@etri.re.kr

A Study on the Multicast Framework for the Clustered EJB Server

Soo-Hyung Kim, Seung-Woog Jung, Beom-Su Seo, Myung-Chan Roh, Joong-Bae Kim
Dept of Electronic Commerce,
Electronics and Telecommunications Research Institute

요 약

EJB 서버의 클러스터 구성을 위해서는 빈 인스턴스의 상태 복제 및 클러스터링 구성에 따른 적절한 정적/동적 빈 배포 그리고 노드들의 상태를 감지하기 위한 노드 상태 탐지 기능 등이 요구된다. 본 연구는 이들 기능을 지원하기 위해 기본적으로 요구되는 멀티캐스트 프레임워크에 대해 논의하고자 한다.

1. 서론

대중들의 인터넷 접속이 일반화된 오늘날의 인터넷 서비스들은 보다 많은 사용자에게 중단 없이 안정적으로 제공되는 것을 요구 받고 있다. 특히 대규모의 비즈니스 서비스를 제공하는 경우에는 이러한 요구사항이 필수적으로 만족되어야 하는데, 보다 많은 계산을 안정적으로 수행하기 위해 이전에는 주로 단일 슈퍼컴퓨터에 의존하였으나 현재는 소형 서버들의 클러스터링을 통해 이러한 요구가 충족되고 있다.

클러스터는 두 대 이상의 컴퓨터를 마치 하나의 시스템처럼 행동하도록 연결하여, 대규모의 서비스 요청 처리가 필요하거나, 중단 없는 서비스를 수행하고자 할 때 사용하는 시스템으로, 대용량 전산 처리를 위한 슈퍼컴퓨터에 비해 상대적으로 저렴한 비용으로 필요한 컴퓨팅 파워나 기능을 구현 할 수 있는 장점을 갖는다.

클러스터 시스템에서 대규모 서비스 요청 처리를 위해서는 클러스터 구성 노드들의 상태 정보를 바탕으로 적절하게 부하를 분배하는 부하 분배 서버가 필요하며 클러스터 구성 노드들 각각은 사용자의 입장에서 하나의 서버가 서비스를 제공하는 것처럼 하기 위해 노드들간 상태 정보 공유가 필수적이다.

특히 EJB 서버의 클러스터 구성을 위해서는 빈 인스턴스의 상태 복제 및 클러스터링 구성에 따른 적절한 정적/동적 빈 배포 그리고 노드들의 상태를 감지하기 위한 노드 상태 탐지 기능 등이 요구되는데, 본 연구에서는 이들 기능을 지원하기 위한 멀티캐스트 프레임워크에 대해 논의하고자 한다.

2 장에서는 IP 멀티캐스트(Multicast)의 신뢰성 문제를 극복할 수 있는 대안 중 하나인 JavaGroups 의 기본 구조와 기능을 설명한다. 3 장에서는 멀티캐스트 프레임워크를 위한 고려 사항 및 요구 사항에 대해 정의한다. 4 장에서는 본 연구의 멀티캐스트 프레임워크에서 제공하는 기본 기능을 설명하고, 설계된 기본 구성 요소들에 대해 살펴본다. 마지막으로 5 장에서는 본 연구의 결론과 향후 연구해야 할 과제에 대해 이야기한다.

2. JavaGroups

본 연구에서는 IP Multicast 의 신뢰성 문제를 극복하기 위해 JavaGroups 를 사용하여 노드 관리와 상태 복제를 수행할 수 있는 방법을 고려하고 있다. 또한

EJB 서버 중 하나인 JBoss 도 JavaGroups 를 기초로 하여 클러스터를 구성하고 있는데[2], 이는 JavaGroups 에서 제공되는 기능들이 클러스터링 구성을 위해 요구되는 기능과 맞아 떨어지기 때문이다.

JavaGroups 은 신뢰성 있는 멀티캐스트 통신을 제공하는 툴킷으로, IP Multicast 를 기반으로 하고 있지만 메시지 송 수신에 대한 신뢰성과 그룹 멤버십 관리 기능 등을 제공한다. 같은 그룹을 구성하는 노드에 전송된 메시지는 메시지 송신 노드에 의해 수신이 확인되며, 메시지는 크기에 상관없이 송신된 순서에 따라 정확히 그룹 내 노드에 전달된다. 또한 그룹에 처음 접속되는 노드 정보와 그룹 탈퇴 노드 정보가 모든 노드들에 송신됨으로 클러스터 노드 관리에 필요한 기본적인 기능을 제공한다[1].

JavaGroups 은 그림 1 과 같이 구성되어 있는데, 채널의 하위 프로토콜 스택은 위에서 설명한 JavaGroups 의 기능을 제공하기 위한 다수의 프로토콜들로 이루어져 있다. 채널은 요구되는 통신 환경/특성에 따라 프로토콜 스택을 구성하고 멀티캐스트 송 수신을 위한 API 를 제공한다. 그리고 마지막으로 Building Block 은 Listener 를 통한 풀(Pull) 타입 메시지 리시버 구성, 동기식(Synchronous) 메시지 송 수신, RPC 형태의 원거리 메소드 호출 등을 제공할 수 있는 서비스 블락들이다.

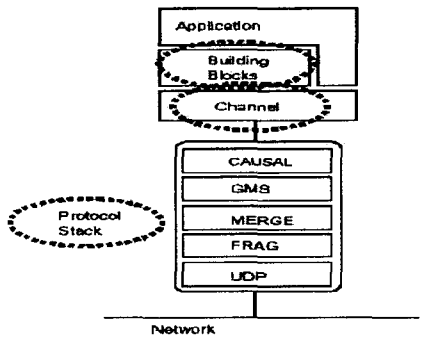

그림 1 JavaGroups 의 구성

JavaGroups 의 핵심 기능 중 하나인 그룹 관리를 좀 더 자세히 살펴보면, JavaGroups 은 그룹에 새로운 멤버가 추가되거나 기존 멤버가 빠져나갈 때, 그룹 내 멤버가 이상 징후를 보일 때, 그룹에 처음 참여한 멤버(Coordiantor)가 현재의 그룹 멤버 리스트(View)를 그룹 내 모든 노드에 전송하여 모든 멤버가 동일한 멤버 리스트를 유지할 수 있도록 하고 있다. JavaGroups 의 이러한 그룹 관리 기능은 클러스터를 구성하는 노드들의 관리에 그대로 사용될 수 있다고 보나, 본 연구에서는 JavaGroups 에 종속적이지 않기 위해 JavaGroups 이외의 다양한 멀티캐스트 채널을 수용할 수 있는 구조를 갖는다. 이는 JavaGroups 의 성능상의 문제가 일부분 고려되었다.

3. 멀티캐스트 프레임워크 요구 사항

클러스터를 구성하는 목적은 대규모 서비스 요청으로 인한 서버의 과부하를 방지하고 좀 더 원활한 서비스를 제공하기 위해 서비스 요청을 다수의 서버들에 분산시키기 위해서이며, 또한 하나의 서버가 다운되는 경우에도 서비스가 중단 없이 제공되기 위해 서비스를 클러스터 내 모든 서버에서 제공하기 위함이다. 이와 같은 클러스터 구성 목적을 만족 시키기 위해서는 다음과 같은 클러스터 고려 사항들[3],[4]이 멀티캐스트 프레임워크에서도 또한 고려되어야 한다.

3.1 부하 분배(Load Balancing)

부하 분배란 사용자의 서비스 요청들을 부하 분배 알고리즘(Round-Robin, Random, Weight-based, Load-based, etc)에 따라 다수의 서버에 분배하는 것을 의미한다. 부하 분배를 위해 부하 분배 서버는 클러스터를 구성하는 서버들에 대한 정보를 알아야 하는데, 동적으로 서버 정보를 확인하고 구성하기 위해 본 연구에서는 멀티캐스트 프레임워크에 멤버 관리 메커니즘을 제공하고 있다.

3.1 고가용성(High availability, H/A)

고가용성이란 어느 하나의 노드에 장애가 생겨도 서비스를 중단 없이 수행하도록 구성된 시스템을 의미하며, 대개 LVS (Linux Virtual Server)의 Load balancer 와 같이 구성되어 있기 때문에 LVS 와 H/A 는 거의 같은 뜻으로 사용된다. 하지만 정확히 말하면 LVS 는 대규모의 transaction 처리를, H/A 는 시스템이나 소프트웨어 장애에 초점을 둔 것이다. 본 연구에서는 고가용성을 위해 노드의 상태 정보를 동적으로 감독· 유지할 수 있는 메커니즘을 가지며, 노드들의 빈 인스턴스(Instance) 상태 정보를 클러스터 내 모든 노드들에 전달하여 이를 관리할 수 있는 구조를 고려하고 있다.

3.2 Failover

Failover 는 사용자의 개입 없이 사용자의 서비스가 노드 장애 시에도 계속될 수 있도록 하는 기능으로, 이를 위해 본 연구에서는 서비스 대체 서버 선택을 위한 노드 정보 제공 및 상태 복제 등에 필요한 기본 기능을 제공하는 구조를 갖도록 하고 있다.

3.3 클러스터링 구조

클러스터링 구조는 클러스터를 구성하는 WebServer, JSP/Servlet Engine, EJB 서버 들의 위상을 의미하는데, 클러스터링 구조는 비즈니스 요구 사항에 따라 다양하게 설계될 수 있으며 설계된 구조에 따라 클러스터 노드에서 제공하여야 하는 기능의 범위는 달라진다. 본 연구는 그림 2 와 같은 클러스터링 구조 모델을 기준으로 하고 있으며, HTTP 서버의 Proxy Plug-in 과 통신하여 동적인 노드 정보 전달이 가능한 구조를 갖도록 한다.

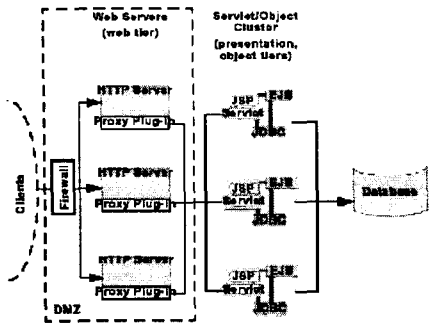

그림 2 클러스터링 시스템 구조

3.4 기타 사항

이외에 네이밍(Naming), 스마트 스텁(Smart Stub), 리소스(JDBC, JMS Queue/Topic, Transaction) 복제 등의 EJB 서버 클러스터링을 지원하기 위한 고려 사항 들이 있으나 본 연구에서는 클러스터링을 위한 통신 프레임워크를 다루고 있으므로 논의에서 제외한다.

4. 멀티캐스트 프레임워크 설계

위에서 소개한 요구 사항들을 만족시키기 위해 전체 멀티캐스트 프레임워크를 그림 3 과 같이 구성하였다. EJB 서버의 클러스터링을 제공하기 위한 모듈인 Cluster Deployer, State Replicator, Cluster Monitor 등에 대한 상세 내역은 논외로 하며 본 장에서는 이들을 지원하기 위한 멀티캐스트 프레임워크에 대해 설명하고자 한다. 멀티캐스트 프레임워크에서 지원해야 할 기능은 앞서 설명한 바와 같이, 클러스터 내 노드들의 상태 정보 파악, 노드 내 빈 인스턴스들의 상태 복제를 위한 상태 오브젝트 송· 수신 등이다.

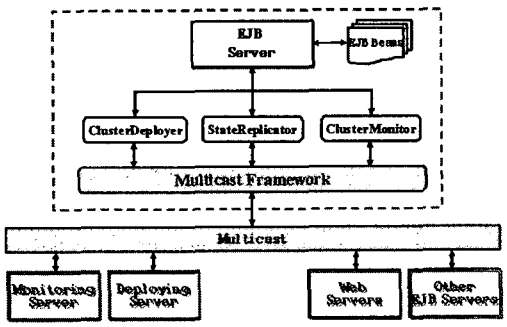

그림 3 EJB 서버 클러스터링을 위한 시스템 구성도

그림 3 에서 보는 바와 같이, 멀티캐스트 프레임워크는 노드 내 Cluster Deployer, State Replicator, Cluster Monitor 에서 요구되는 서비스를 제공 하며, 외부의 모니터링 서버, 빈 배포 서버, 웹 서버 및 다른 EJB 서버와 통신하여 필요한 정보를 수집 또는 제공한다.

아래에 프레임워크를 구성하는 클래스들과 그 기능에 대해 좀 더 상세히 살펴보도록 한다.

4.1 Cluster Manager

클러스터 매니저는 클러스터에 필요한 기본 정보 및 관련 객체들을 관리 하며 EJB 서버의 클러스터링 기능 제공 모듈들에 노드 정보와 노드 간 상태 복제와 관련된 API 를 제공한다.

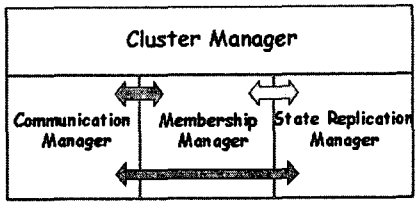

그림 4 멀티캐스트 프레임워크 개념도

클러스터 매니저는 그림 4 와 같이 Communication Manager, Membership Manager, State Replication Manager 를 두고 있는데, 각각의 매니저에서 요구되는 초기 설정 값은 클러스터 환경 정의 파일에서 가져온다. 환경 정의 파일은 EJB 서버의 배포 관리자에 의해서 정의되며 멀티캐스트 통신 환경과 클러스터 구성 방법 등에 대한 정보를 가지고 있다.

Membership Manager 와 State Replication Manager 는 필요한 통신 기능을 얻기 위해 Communication Manger 로부터 통신 채널을 얻어온다. 그리고 State Replication Manager 는 상태 복제 대상 노드를 결정하기 위해 Membership Manager 로부터 클러스터 구성 노드들에 대한 정보를 가져온다.

4.2 Communication Manager

그림 5 는 Communication Manager 와 관련된 클래스들의 상관 관계를 보여주고 있는데, Communication Manager 는 클러스터에 속한 전체 노드들에 대한 정보 전달을 위해 기본적으로 멀티캐스트 채널을 관리한다. 하지만 특정 노드에 대한 정보 전달을 위해 유니캐스트 채널을 관리할 수 있다.

Transport Connector 는 채널에 대한 속성, 채널 관련 클래스와 해당 정보등을 관리하며, Transport Interface 를 구현한 다양한 Transport 모듈들을 생성하여 사용할 수 있다.

Transport 모듈은 앞서 소개한 JavaGroups 뿐만 아니라 IP Multicast, 기타 다른 통신 서비스로 수행될 수 있으며 해당 Transport 에 적합한 메시지 리시버를 포함하고 있다.

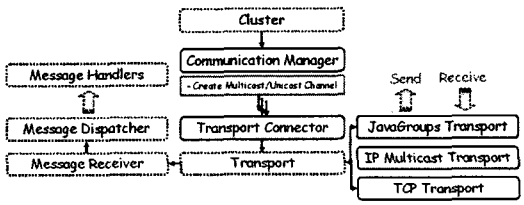

그림 5 Communication Manager

다른 노드로부터 수신된 메시지는 메시지 리시버에게 전달되고 전달된 메시지는 메시지 분배자(Dispatcher)에 의해 각각의 메시지 타입에 따라 해당 핸들러에게 분배된다. 메시지 타입은 멤버십 정보, 클러스터 관리정보 및 상태 정보 등으로 구분된다.

4.3 Membership Manager

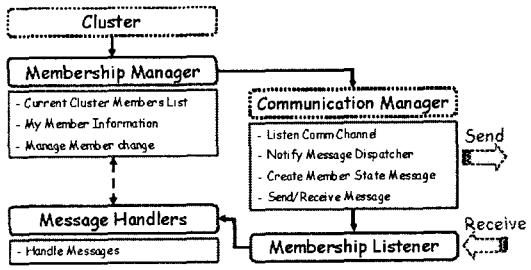

그림 6 Membership Manager

그림 6 에서 보는 바와 같이 Membership Manager 는 동일 클러스터에 속하는 노드들의 정보를 관리하는데, Communication Manager 가 관리하는 Transport Connector 를 통해 자신의 상태 정보(Join, Leave, Crash, etc)를 전송하며 다른 노드로부터 전송된 상태 정보를 수신한다. 수신된 상태 정보는 Membership Manager 의 현재 클러스터 노드 정보 리스트에 의해서 관리되고 상위 멤버 관리 모듈에게 필요한 정보를 전달한다.

또한 Membership Manager 는 모니터링 서버와 필요한 정보를 주고 받을 수 있으며, 자신의 노드 상태를 변경하거나 서버의 동작을 결정하기 위해 모니터링 서버로부터 명령을 수신할 수도 있다.

4.4 State Replication Manager

그림 7 은 State Replication Manager 와 연관된 클래스들을 보여주는데, Membership Manager 와 같이 Transport Connector 를 통해 Stateful Session Bean 인스턴스의 상태를 복제하기 위해 요구되는 정보를 송 수신한다. Communication Manager 에 의해 의뢰하여 생성된 Transport Connector 는 필요에 따라 멀티캐스트/유니캐스트 통신 채널을 지원한다.

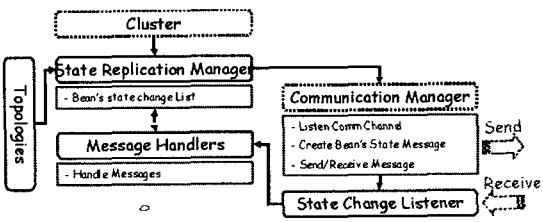

그림 7 State Replication Manager

State Replication Manager 는 상태 복제와 관련된 상위 모듈에 상태 복제 서비스를 제공하기 위해 상태 복제를 위한 복제 대상 리스트를 관리하고 다른 노드의 State Replication Manager 로부터 수신한 상태 복제 정보도 함께 리스트에 보관한다. 이러한 상태 복제 리스트는 컨테이너가 관리하는 해당 빈 인스턴스의 상태를 결정하기 위해 사용되며 사용자 요청에 대한 서비스 결과가 서비스 제공 노드에 따라 다르게 나타나는 것을 방지한다.

토폴로지 클래스는 상태 복제 시 상태 복제가 이루어질 노드 정보를 관리하는 클래스로, 클러스터 노드들은 다음과 같은 토폴로지로 구성될 수 있다.

- HomogeneousTopology : 모든 클러스터 노드에 상태를 복제한다.
- PrimarySlaveTopology : 특정 slave 노드에 상태를 복제 한다.
- HeterogeneousTopology : 몇 개의 지정된 특정 노드에 상태를 복제한다.

5. 결론

대부분의 EJB 서버 제공 업체들은 대규모의 서비스 요청 처리를 위해 EJB 서버들을 클러스터링 할 수 있는 방법을 제공하고 있다[2][3][4]. 클러스터는 그 구조에 따라 다양한 방법으로 이루어 질 수 있지만, 클러스터링을 위해 기본적으로 요구되는 것은 노드들의 정보 관리와 Stateful Session Bean 인스턴스의 상태 복제와 관련된 기능이다. 본 연구에서는 이러한 노드 정보 관리와 상태 복제를 위한 멀티캐스트 프레임워크에 대해 논의 했으며, 설계된 프레임워크 구조에 대해 설명하였다.

본 연구에서는 신뢰성 있는 멀티캐스트 통신을 위해 JavaGroups 을 사용하고 있지만, JavaGroups 의 성능상의 문제 때문에 IP multicast 도 또한 하나의 방법으로 고려되어 설계되었다. 향후 과제는 이들의 구현 과정에서 IP multicast 의 신뢰성 문제를 어떻게 처리할 것인지에 대한 정책과 방법을 제시하는 것이다. 그리고 클러스터 노드들을 관리 감독할 수 있는 방법도 함께 고려되어 구체적인 연구가 진행 되어야 할 것으로 보인다.

참고문헌

[1] Bela Ban, "JavaGroups User's Guide," http://www.javagroups.com/javagroupsnew/docs/newuser.zip, 2001

[2] Sacha Labourey, Bill Burke, "JBoss 3.0 Clustering," http://www.jboss.org/docs/#Clustering, 2002

[3] Abraham Kang, "J2EE Clustering," http://www.javaworld.com/jw-02-2001/jw-0223-extremescale.html, 2001

[4] "Using WebLogic Server Clusters," http://e-docs.bea.com/wls/docs70/cluster/index.html, 2002