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

JMX 를 이용한 동적 서버 클러스터링의 관리

채희성*, 윤수현*, 송하윤*, 김한규*, 이기철*, 박중기**
*홍익대학교 컴퓨터공학과
**한국전자통신연구원 모바일 응용 서버 연구팀
e-mail : {hschae, shyoun, song, hkim, lee}@cs.hongik.ac.kr, jkp@etri.re.kr

The Management Technology of Dynamic Server Clustering Using JMX

Hee-Sung Chae*, Su-Hyeoun Youn*, Ha-Yoon Song*, Han-Gyoo Kim*,
Kee-Cheol Lee*, Joong-Ki Park**
*School of Information and Computer Engineering, Hongik University
**ETRI, Mobile Application Server Research Team

요 약

인터넷 서비스 환경은 모바일 단말기들을 지원하는 비즈니스 및 서비스 환경으로 급변하고 있다. 이러한 변화에 따라 기존 유선 인터넷 뿐 아니라 모바일 환경의 서비스에서도 쉽게 사용할 수 있는 모바일 응용서버에 관한 연구가 필요하다. 서버 환경에서는 시스템 접근 요구가 많은 대규모 서비스를 위해 상시 가동을 기본적으로 필요로 하며, 큰 트래픽의 효과적인 분산을 통해 부하분산, 확장성, 가용성 등의 기능을 제공해 주기 위해서 클러스터링 기법을 이용한다. 이 논문에서는 클러스터링의 효과적인 관리를 위해 JMX(Java Management Extensions) 프레임웍을 이용하여 서버의 동적 클러스터링을 관리하는 새로운 기법을 제시한다.

1. 서론

이동통신 기술의 발전에 따라 셀룰러 폰, PDA, PostPC 와 같은 모바일 단말기들이 광범위하게 보급되어 활용되고 있다. 더불어 이를 응용한 비즈니스 및 서비스 환경으로 인터넷 서비스 환경이 급변하고 있다. 이러한 환경변화에 따른 모바일 서비스의 요구를 효과적으로 서비스하기 휘해서는 기존의 유선 인터넷을 기반으로 구축되어 운영중인 서비스를 모바일 환경에서도 쉽게 사용할 수 있도록 함은 물론 모바일과 관련된 지식이 없이도 새로운 서비스를 쉽게 개발 할 수 있는 유무선 통합 모바일 응용서버에 대한 연구가 절대적으로 필요하다[1].

이 논문에서는 유무선 환경을 통해 전달되는 대규모의 요청을 효율적으로 분산시켜 시스템 성능을 높일 수 있도록 하기 위하여 JMX(Java Management Extensions)를 이용한 모바일 응용서버의 동적 클러스터링 방법에 대하여 설명한다.

2. 관리 프로토콜

2-1. SNMP

단순 망 관리 프로토콜은 TCP/IP 프로토콜을 사용하는 네트워크에 존재하는 장치들을 관리하기 위한 기본 구조로서, 네트워크의 상태를 감시하고 유지·보수하기 위한 기본적인 사항을 제공한다.

SNMP(Simple Network Management Protocol)에서는 관리자, 관리 에이전트, 관리 정보 베이스, 네트워크 관리 프로토콜의 4 가지 구성 요소와 GetRequest, GetNextRequest, GetResponse, SetRequest, Trap 의 5 가지

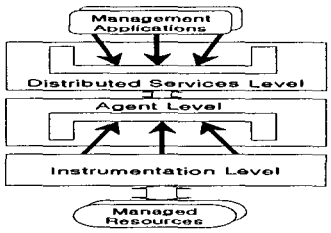
의 메시지의 형식을 이용하여 네트워크 관리를 하게 된다[6].

2-2. CMIP

CMIP(Common Management Internet Protocol)는 SNMP 에 비해서 보안성, 이벤트 발생 기능에서 우월하며, 인증, 접근제어, 보안 로그 등 관리 정보에 대한 보안 체제를 갖추고 있는 새로운 프로토콜로 안정성이 뛰어나다. SNMP 와 달리 CMIP 에서는 이벤트 발생 기능을 지원하므로, 관리 프로세스가 관리대상을 계속해서 감시(polling)할 필요가 없이 보다 능률적으로 네트워크를 관리할 수 있다. 이 시스템 관리를 목적으로 정보와 명령을 교환하기 위해서 이용하는 기능이 CMISE(Common Management Information Service Element)이고, 이는 사용자와의 인터페이스 서비스를 담당하는 CMIS(Common Management Information Service)와 PDU(Protocol Data Unit)및 관련 절차를 규정한 CMIP 으로 이루어진다[6].

3. JMX

JMX 는 썬 마이크로 시스템에서 제안한 관리 프레임웍이다[2][4]. JMX 는 3 개의 구분된 계층으로 구성되어 있다. 관리를 받는 데이터와 응용 프로그램은, 관리를 수행하는 관리 계층에 독립적으로 구성되어질 수 있다. JMX 는 기본적으로 3 개의 계층 외에도 추가로 관리 프로토콜의 API 들을 가지고 있다. JMX 의 기본 구성도는 그림 1 에 나타나 있다.



<그림 1> JMX 의 기본 구성도

3-1. Instrumentation Level

자바 기술을 기반으로 하는 자원들을 직접적으로 표현하는 계층으로서 MBeans(Managed Beans)와 Notification Model 그리고 MBean Metadata Classes 로 구성되어 있다. MBean 은 Standard MBean, Dynamic MBean, Open MBean, Model MBean 의 4 가지 종류로 구분되어 진다.

3-2. Agent Level

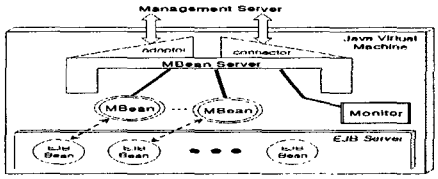
이 계층은 관리를 위한 핵심 부분인 MBean Server 와 Agent Services 로 구성 되어 있다. MBean Server 는 MBean 들이 등록되어지는 컴포넌트로서, 각각의 Mbean 들을 위한 관리 인터페이스를 제공함으로써, 관리 시스템이 이 MBean 들의 존재를 알 수 있게 하는 역할을 한다. Agent Services 는 MBean Server 에 등록되어 있는 MBean 에 대해서 관리 연산을 수행할 수 있는 객체로서 Dynamic Class Loading, Monitors, Timers, The Relation Service 의 4 가지 종류가 있다.

3-3. Distributed Services Level

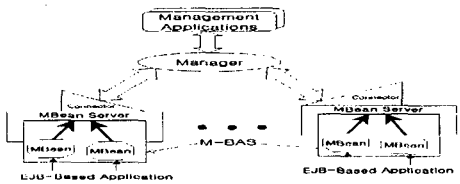
Distributed Services Level 은 다수의 에이전트들로부터 오는 관리 정보를 최종적인 관리 응용 프로그램이 사용할 수 있도록 제공해 주는 계층이다. 커넥터 클라이언트를 통해서 에이전트 측의 커넥터 서버와 통신을 해서 관리 정보를 얻어오는 방법이 있고, 부가적으로 제공되어지는 관리 프로토콜들의 API 를 이용해서 만든 어댑터를 통해서 관리 정보를 얻어오는 방법이 있다[3].

4. 아키텍쳐

JMX 를 이용해서 관리하는 대상이 되는 객체는 EJB(Enterprise JavaBeans) 등의 자바 관련 기술을 기반으로 만들어져 있는 각각의 응용 프로그램들이다. 각 응용 프로그램들은 MBean 에 매핑되고 각 MBean 은 응용 프로그램이 동작하고 있는 동일한 자바 가상 머신에 있는 MBean server 에 등록되게 된다. 동시에 이 MBean server 에서는 4 가지의 Agent Services 가 동작을 하게 된다. Agent Services 의 하나인 Monitor 가 각 MBean 들을 각각의 상태 정보에 의해 관리한다. Monitor 가 상태 정보를 관리 중에 특정 상황(MBean 이 처리할 수 있는 한계의 초과, MBean 의 동작 중단 등)을 감지하면, Monitor 는 이벤트를 발생시켜 매니저 시스템이 문제 상황을 파악해서 대처할 수 있도록 구성된다. 에이전트와 매니저 시스템 간의 대표적인 통신방법은 2 가지의 선택이 있다. 첫 번째는 SNMP APIs 를 이용해서 SNMP 방식의 관리를 이용하는 방법이다[3]. 두 번째는 에이전트에 커넥터 서버(connector server)를, 관리 시스템에 커넥터 클라이언트(connector client)를 설치해서 통신을 하는 방식이다. 그림 2 는 JMX 로 구성한 기본적인 클러스터링 환경이다.



<그림 2> EJB 로 구축된 응용 서버의 내부 관리 구조



<그림 3> JMX 를 이용한 기본적인 클러스터링 환경

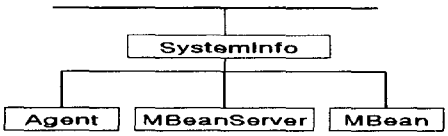
그림 3 은 JMX 를 통해 관리 되어지는 EJB 기반의

M-BAS(Mobile-Business Application Server)의 모습을 보여준다. 관리를 위한 매니저 시스템은 1 개로 되어 있고, 그 매니저 시스템이 전체 클러스터링 시스템을 관리한다. 이러한 구조는 간단하게 클러스터링 시스템을 유지할 수 있으며, 성능상의 결함을 보이지 않는다. 그러나, 매니저의 시스템에 문제가 발생하면 SPoF(Single Point of Failure)의 대표적 경우로서 전체 클러스터링 시스템이 제 기능을 할 수 없는 최악의 상황에 처한다. SPoF 문제를 해결하기 위해서 다수의 매니저 시스템을 구축하여 관리하는 구조가 제시 될 수 있다.

5. 관리 객체 및 메시지

5-1. 관리 객체

JMX 를 이용하여 동적인 클러스터링을 구축 하는데 필요한 기본적인 객체들의 구조도는 그림 4 와 같이 제시된다.

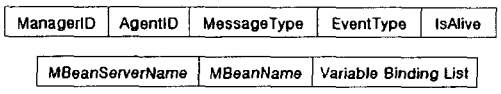


<그림 4> 관리 객체 구조

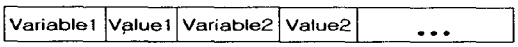
SystemInfo 밑에 관리 대상이 되는 서버의 핵심 구성 요소들의 정보를 다루기 위해 에이전트, MBeanServer, MBean 의 3 개의 그룹을 정의한다. 각 컴포넌트는 관리하고 있는 에이전트, MBeanServer, MBean 의 정보를 관리하기 위한 요소이다. 각각의 그룹에 관한 정보는 표 1, 표 2, 표 3 에서 보여주고 있다.

5-2. 관리를 위한 메시지의 형식

본 논문에서 제시한 아키텍처 상에서 매니저와 에이전트간의 통신에 이용 되는 메시지의 형식은 그림 5 와 같다. 그림 6 은 그림 5 에 나타난 Variable Binding List 의 세부적 표현을 보여준다.



<그림 5> 메시지 형식



<그림 6> Variable Binding List 의 형식

5-3. 메시지의 종류

상태 정보 요구, 상태 정보 응답, 이벤트 발생 공지, MBean 관련 정보 요구, MBean 관련 정보 응답의 5 가지의 메시지 형식이 있다.

<표 1> Agent 그룹

Id	변수 이름	형태	설명
1	N_Agent	Counter	Agent 의 총 수
2.1.1	AgentID	String	Agent 의 식별자
2.1.2	AgentName	String	Agent 의 이름
2.1.3	AgentIP	String	Agent 가 있는 서버의 IP Address
2.1.4	N_MBeanServer	Counter	MBeanServer 의 수

<표 2> MBeanServer 그룹

Id	변수 이름	형태	설명
1	Total_MBeanServer	Counter	MBeanServer 의 총 수
2.1.1	MBeanServerID	String	MBeanServer 의 식별자
2.1.2	MBeanServerName	String	MBeanServer 의 이름
2.1.3	N_MBean	Counter	MBean 의 수

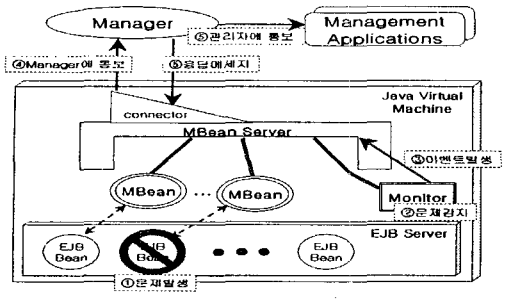
<표 3> MBean 그룹

Id	변수 이름	형태	설명
1	Total_MBean	Counter	MBean 의 총 수
2	Run_MBean	Counter	동작중인 MBean 수
3	Idle_MBean	Counter	Idle 한 MBean 수
4	CF_MBean	Counter	처리 용량이 꽉 찬 MBean 수
5	Stop_MBean	Counter	기능을 잃은 MBean 의 수
6.1.1	MBeanID	String	MBean 의 식별자
6.1.2	MBeanName	String	MBean 의 이름
6.1.3	TC_MBean	Counter	MBean 의 총 서비스 처리 용량
6.1.4	URate_MBean	Counter	MBean 의 서비스 처리 용량의 사용률
6.1.5	MBeanServerID	String	MBean 을 관리하고 있는 MbeanServer 의 식별자

6. 시나리오

응용 서버들에서 발생할 수 있는 대표적인 상황으로 서버의 추가, 서버의 제거, 서버에 발생한 문제의 확인 등을 들 수 있다. JMX 를 통한 관리 기법을 이용했을 때, EJB 기반의 응용 서버들에서 발생 할 수 있는 상황들에 대한 대처 방법을 보여주는 시나리오들을 설명한다.

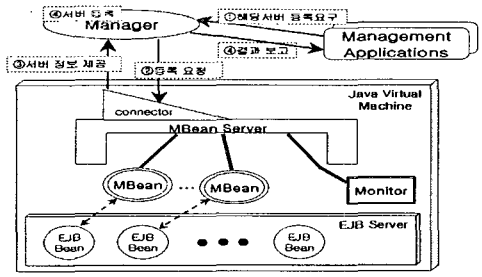
6-1. MBean 에 문제가 발생한 상황



<그림 7> MBean 에 문제가 생긴 시나리오

① 문제발생
② Monitor 에서 문제 감지
③ 에이전트로 이벤트 발생을 알림
④ 에이전트에서 매니저로 이벤트가 발생했음을 알리는 메시지를 전송한다. 응답이 없으면 재전송을 한다.
⑤ 매니저는 에이전트에게 이벤트를 접수했다는 응답을 하기 위해 필드에 [해당 매니저, 해당 에이전트, 이벤트발생공지, 수신확인]의 내용을 기록하여 에이전트에게 보내게 된다. 동시에 관리 응용 프로그램에 이벤트 발생을 통보함으로써 관리자에게 MBean 에 발생한 문제에 대해 알려주고, 해당 에이전트가 있는 서버로 새로운 서비스 요청이 배정되지 않도록 한다.

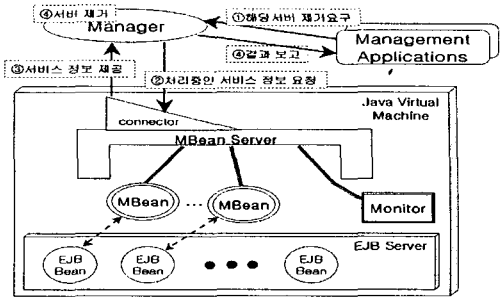
6-2. 새로운 서버가 추가 되는 상황



<그림 8> 새로운 서버가 추가되는 시나리오

① 관리 응용 프로그램을 통해서 새로운 서버가 등록될 매니저에게 추가 시키려는 서버의 IP 정보와 AgentID 를 알려준다.
② 매니저는 새로운 서버의 에이전트에게 등록을 요청하는 메시지를 보낸다.
③ 매니저에 서버를 등록함과 동시에 서버가 가진 MBeanServer 와 MBean 들의 정보를 매니저에게 전송하게 된다.
④ ③의 메시지를 수신한 매니저는 등록과정을 마치게 되고, 관리 응용 프로그램에 등록이 완료되었음을 알리게 된다. 만약 ③의 과정이 실패하게 되어 응답이 오지 않는 경우엔 관리 응용 프로그램에 서버 등록 실패를 알리게 된다.

6-3. 서버를 제거하는 상황



<그림 9> 서버를 제거하는 시나리오

① 관리 응용 프로그램을 통해서 매니저에게 제거하려는 서버의 IP 정보와 AgentID 를 알려준다.
② 매니저는 제거될 서버에서 처리중인 서비스의 존재 여부를 확인을 위해 에이전트에게 메시지를 전송한다.
③ 에이전트는 매니저에게 처리되고 있는 서비스가 있는지의 여부를 알려준다.
④ ③에서 받은 정보에 처리중인 서비스가 없으면 해당 서버를 관리 목록에서 제거하고 관리 응용 프로그램에게 제거했음을 알린다. 만약 처리중인 서비스가 존재한다면 새로운 서비스의 요청이 해당 서버로 배정 되지 않도록 처리를 한 후 관리 응용 프로그램에게 추후에 제거 작업을 처리할 것임을 알린다.
⑤ 단계 ②~④가 서비스가 존재하지 않을때까지 지속적으로 반복된다.

7. 결론

본 논문에서는 새로운 프레임웍인 JMX 에 기반한 응용서버의 동적인 클러스터링 관리를 지원하는 아키텍처와 관리 객체를 제시 하였다. 즉, 자바 기반의 서버환경에서의 클러스터링 관리를 위한 새로운 기법을 의미한다.

본 기법의 검증을 위하여 본 논문에 제시된 아키텍처와 관리 객체 등을 기반으로 구체화된 시스템을 구성, 실제 실험을 진행하여 JMX 가 동적인 클러스터링 관리에 미치는 효율성을 파악하는 것이 추후의 연구 목표이다.

참고문헌

[1] ETRI, "유무선 통합 모바일 응용서버에 관한 연구", 정보과학회지 제 20 권 제 6 호, 2002. 06.
[2] Sun Microsystems, "Java Management Extensions Instrumentation and Agent Specifications, v1.1", 2002.05.
[3] Sun Microsystems, "Java Management Extensions SNMP manager APIs", 1999. 08.
[4] H. Kreger, "Java Management Extensions for application management", IBM Systems Journal, Vol 40, No 1, 2001.
[5] Jyoti Batheja, Manish Parashar, "Adaptive Cluster Computing using JavaSpace", IEEE International Conference on Cluster Computing, 2001.
[6] William Stalling, SNMP, SNMP v2, and CMIP, Addison Wesley, 1993.