유비쿼터스 컴퓨팅 환경에 적합한 우선순위 기반 지니 리스 스케줄러의 구현

김기섭, 김영윤
삼성전자 디지털미디어 연구소
e-mail : brendan.kim@samsung.com

The Implementation of Priority Based Jini Lease Scheduler for Ubiquitous Computing Environment

Ki-Sup Kim, Young-Yoon Kim
Software Platform Lab., Digital Media R&D Center, Samsung Electronics Co., Ltd.

요 약

유비쿼터스 컴퓨팅 환경 안에서 모바일/임베디드 장치는 하드웨어적인 제약이 있어 충분한 저장 공간을 갖기가 어렵다. 지니 기술은 네트워크 가능한 모바일/임베디드 장치가 필요로 하는 서비스를 작업 그룹 내에서 검색하고 시스템의 재구성 없이 네트워크를 통해 일정 기간 동안 사용할 수 있도록 해줌으로써 이러한 문제를 해결해준다. 기존의 지니 기술에 서비스를 사용하는 사용자에게 우선순위를 부여하고 이에 따라 서비스 리스 사용 시간을 스케줄해주는 기능을 추가함으로써 효율적인 자원 관리를 해줄 수 있다.

1. 서론

유비쿼터스 컴퓨팅 환경(Ubiquitous Computing Environment)안에서 네트워크를 통해 공유할 수 있는 서비스와 자원(Resource)의 효율적인 연결을 위해 자원을 제공하는 서비스 제공자(Service Provider)와 자원을 필요로 하는 클라이언트(Client)를 연결해 주는 브로커(Broker)의 역할을 담당하는 서비스를 생각할 수 있다. 선 마이크로시스템즈(Sun Microsystems) 사의 지니 기술(Jini Technology)은 이러한 기능을 가능하게 준다.

유비쿼터스 컴퓨팅 환경 안에서 모바일/포터블 디바이스(Mobile/Portable Device)나 네트워크 가능한 임베디드 디바이스(Embedded Device)들은 하드웨어적인 제약이 있어서 충분한 저장 공간을 가지고 있지 못하다.

지니 기술은 필요로 하는 서비스에 대하여 클라이언트가 서비스 프로그램의 인스톨(Installation)이 없이 네트워크 환경을 통해 서비스 제공자의 자원을 일정 기간 동안 사용할 수 있도록 하여 작은 용량의 저장 공간을 갖고 있는 디바이스에 적합하다. 그러나 분산된 네트워크 환경(Distributed Network Environment)에서 클라이언트들은 기본적으로 한정된 자원을 무한히 점유하고자 하는 특성이 있다.

본 논문에서는 기존 지니 기술에 클라이언트들의 속성(Attribute)을 분류함으로써 미리 사용자의 우선순위(Priority)를 정하고 조건에 따라 사용기간을 정의하여 서비스 리스(Service Lease)를 제공하는 리스 스케줄러(Lease scheduler) 기능에 대하여 설명한다. 이는 유비쿼터스 컴퓨팅 환경 안에서 다양한 클라이언트의 서비스 점유 요구에 대하여 서비스 제공자의 자원을 제한적으로 제공해줄 수 있도록 한 자원 분배 방법의 하나이다.

2. 지니 기술

지니 서비스는 자바(Java) 언어 기반의 서비스 프록시(Service Proxy)를 필요로 하는 클라이언트와 서비스 프록시를 제공하는 서비스 제공자 그리고 이를 연결해 주는 룩업 서비스(Lookup Service)로 구성되어 있다. 지니 기술은 다음과 같은 기능을 제공한다.

룩업 서비스: 서비스 제공자와 클라이언트는 룩업 서비스에 등록되어 있는 서비스 레지스트라(Service Registrar) 객체를 찾게 된다. 이후 룩업 서비스는 클라

이언트가 원하는 서비스와 서비스 제공자가 갖고 있는 프록시 객체의 서비스 엔트리(Service Entry)가 일치하면 양쪽 자바 가상 기기(JVM: Java Virtual Machine) 안에서 실행되는 프록시 객체를 네트워크를 통해 연결해 줌으로써 서비스가 실행 된다.

디스커버리와 조인 프로토콜(Discovery and Join Protocols): 클라이언트와 서비스 제공자는 디스커버리 프로토콜을 통해 클라이언트가 필요로 하는 서비스나 서비스 자신을 필요로 하는 클라이언트에 대한 정보를 갖고 있는 룩업 서비스를 찾게 된다.

디스커버리 프로토콜은 두 가지 종류가 있는데, 작업 그룹 내에서 정확한 위치를 알 수 없는 룩업 서비스를 찾기 위한 경우에 사용되는 멀티캐스트 디스커버리 프로토콜(Multicast Discovery Protocol) 과정과 이미 룩업 서비스의 IP 어드레스(IP Address)를 알고 있는 경우에 사용하게 되는 유니캐스트 디스커버리 프로토콜(Unicast Discovery Protocol) 과정으로 나뉘어 지게 된다. 멀티캐스트 디스커버리 프로토콜의 경우, 사용 가능한 룩업 서비스는 해당 서비스를 요청한 클라이언트에게 자신의 위치를 알려줌(Multicast Announcement Protocol)으로써 서비스를 시작한다[1].

클라이언트와 서비스 제공자는 이를 통해 필요로 하는 서비스 레지스트라 프록시를 룩업 서비스에 등록(Join)하게 되고, 서비스 레지스트라 프록시는 서비스 제공자와 클라이언트가 룩업 서비스와 통신할 수 있는 환경을 제공해 주게 된다.

마지막으로 클라이언트는 디스커버리 프로토콜을 통해 찾게 된 룩업 서비스의 서비스 레지스트라 객체를 통해 서비스 제공자와 연결된다. 이때 서비스 객체는 고유한 아이디(Service ID)를 갖고 있어 해당 작업 그룹 내에서 필요로 하는 서비스를 식별하게 된다.

분산 이벤트(Distributed Events): 일반적으로 자바 가상 기기를 탑재하고 있는 기기가 또 다른 자바 가상 기기를 탑재하고 있는 기기에 이벤트를 보내어 의사 소통을 하기 위한 방법을 말한다.

지니 기술은 분산 환경에서의 객체가 자신의 상태 변경이나 알리고자 하는 이벤트가 있을 때 다른 기기의 어플리케이션 프로그램이 반응하기를 원하는 경우 예측 가능하고 신뢰성이 있는 이벤트의 처리 기능을 제공해 준다. 또한 서로 다른 기기들간의 이벤트 처리 결과를 빠르게 알려준다(Event Notification)[2].

리스 매니지먼트(Lease Management): 한정된 자원의 효율적인 관리를 위해 지니 기술은 서비스 리소스(Service Resource)를 일정기간 동안만 제공해 주는 리스 매니지먼트를 수행한다.

분산 트랜잭션(Distributed Transactions): 지니 기본 구조에서는 서비스를 제공하는 과정에서의 문제점 해결에 대한 메커니즘이 존재하지 않는다. 이를 위해 지니 배포본(Jini Distribution)에는 트랜잭션 매니저(Transaction Manager)를 서비스 형태로 구현하고 있다. 이를 이용해 시스템의 상태를 로그 디렉토리(Log Directory) 안에 저장할 수 있다.

이는 일반적인 지니 서비스와 마찬가지로 지니 연합체(Jini Federation)에 등록되어 있는 서비스(Mahalo Service)를 검색하여 사용함으로써 가능하다[3].

3. 지니 리스 메커니즘

제한된 자원을 사용하기 위해서 지니 기술은 리스 메커니즘(Lease Mechanism)을 사용한다. 리스란 등록된 서비스 프록시(Service Proxy)에 대한 일정기간 동안의 사용권으로써 서비스에 대한 정보 및 속성 등 서비스 사용에 필요한 자원을 제한된 시간 동안만 룩업 서비스에 저장하여 제공함으로써 불필요한 메모리와 저장 공간의 낭비를 방지할 수 있도록 해준다.

리스 지속기간(Lease Duration): 서비스 사용자는 특정한 리스 지속기간을 서비스 제공자와 룩업 서비스에게 요청함으로써 사용하고자 하는 서비스의 사용 시간을 보장 받는다. 이는 리스 협상 과정(Lease Negotiation Process)에 의해 결정되고 지속 시간의 결정은 리스 제공자에 의해 최종 결정된다(예: 20 seconds)[4].

리스 재사용 요청(Lease Renewal): 리스가 제공되면 서비스 사용자는 리스가 종료되기 전에 다시 재사용을 요청할 수 있다. 현재의 지니 기술이 제공하는 메커니즘에는 기본적으로 이러한 리스의 재사용 기간에 대한 제약이 없으므로 제공되는 서비스에서 기간을 제한해주거나 룩업 서비스에서 이를 제한해 줘야만 한다[5].

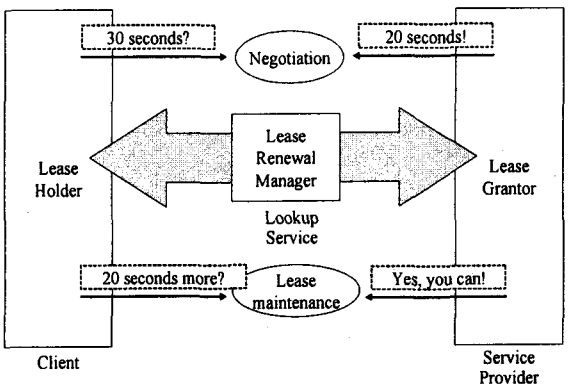

그림 1. 리스 리뉴얼 매니저와 리스 재사용 요청

리스 맵(Lease Map): 기본적으로 클라이언트들은 가능한 많은 시간 동안 서비스를 점유하고자 하기 때문에 과다한 리스의 재사용의 요청은 네트워크의 과부하를 야기할 수 있게 된다. 이를 위해 리스 리뉴얼 매니저(Lease Renewal Manager)는 기본적으로 하나의 서비스 제공자가 제공하는 서비스들에 대하여 동시에 재사용 가능하게 해 주거나 종료해주는 기능을 제공

한다[6].

4. 우선순위 기반 지니 리스 스케줄러

분산환경에서 모든 클라이언트들은 기본적으로 제한된 자원을 무한정 사용하기 원한다. 기존 지니 메커니즘에서의 리스 리뉴얼 매니저는 서비스 리스의 재사용을 가능하게 하여 사용자가 특정 서비스에 대한 리스의 사용을 계속 원할 경우, 이를 점유할 수 있게 된다. 따라서 모든 클라이언트들이 특정 서비스에 대하여 계속 리스의 재사용을 요청하게 되는 경우가 발생하게 된다.

이를 보완하기 위해 서비스를 사용하고자 하는 클라이언트에게 우선순위를 부여하여 사용 기간의 제한을 두어 정해진 기간 동안만 해당 서비스를 사용할 수 있도록 해줄 수 있다. 이로써 사용하고자 하는 서비스를 다른 클라이언트가 이용하고 있다고 해도 사용자의 서비스 점유 우선순위에 따라 리스의 최대 사용 시간을 미리 정해 놓음으로써 특정 클라이언트의 해당 서비스에 대한 점유를 막을 수 있게 된다.

클라이언트 분류: 현재 우선순위 기반 리스 스케줄러에서는 우선순위 그룹에 따라 0, 1, 2 의 단계를 두어서 우선순위 0 그룹인 경우 특정 서비스의 리스 재사용을 지속적으로 가능하게 하고, 우선순위 1 그룹인 경우는 리스 재사용을 제한적(재사용 요청 1 회 가능)으로 가능하게 하였다. 우선순위 2 그룹의 경우에는 해당 서비스의 리스를 요청한 처음 한번만 제공 받을 수 있게 된다. 이는 필요에 따라 재 분류가 가능하다.

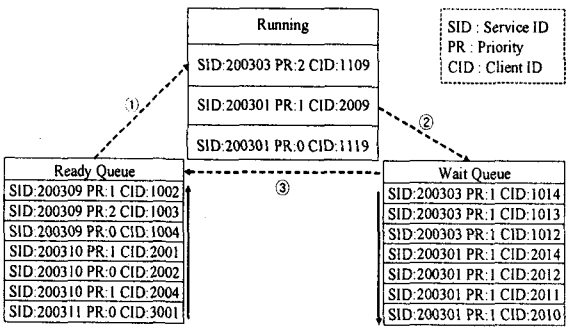

그림 2. 우선순위에 따른 서비스 리스 제공

1. 준비 큐(Ready Queue): 클라이언트들로부터의 등록된 서비스 리스에 대한 요청을 받게 되면 등록된 순서대로 서비스 아이디와 우선순위, 그리고 클라이언트의 아이디가 리스 스케줄러내의 준비 큐에 저장된다. 이후 스케줄러는 클라이언트와 서비스 제공자에게 이벤트를 보내고 클라이언트가 서비스를 제공 받은 이후부터 해당 서비스는 일정기간 동안 실행된다. 서비스 리스가 제공되는 순서는 준비 큐에 들어오는 순서대로(FIFO: First In First Out) 이루어지게 된다. 동일한 서비스를 요청한 클라이언트가 있는 경우에는 해당 서비스 리스의 제공이 종료된 후 준비 큐에 있는 다음 클라이언트에게 서비스 리스를 제공하게 된다.

2. 실행(Running): 준비 큐로부터 받은 서비스 제공 요청에 대하여 서비스 제공자들은 각 클라이언트가 요청하는 서비스 리스의 사용기간(Duration)만큼 해당 서비스에 대한 리스를 제공하게 된다. 이때 각 클라이언트의 우선순위에 따라 리스 제공에 대한 정책이 다르게 적용되게 되는데, 그림에서 실행 시 CID(Client ID) 1109 의 경우 우선순위가 2 이므로 한번의 서비스 리스의 사용 후 재사용을 할 수 없게 된다.

CID 2009 의 경우 우선순위가 1 이므로 일정기간 동안 서비스 리스의 사용한 후 대기 큐로 옮겨져서 이후 해당하는 서비스 제공자로부터 한번 더 서비스 리스를 제공 받게 된다.

CID 1119 의 경우에는 클라이언트가 서비스 리스의 재사용을 요청하면, 서비스 제공자가 재사용에 대한 특별한 제약(예: 최대사용 시간 5 분)을 두지 않는다면 새로 재사용을 요청하여 사용자가 원하는 기간만큼 계속 사용할 수 있다. 이때 해당 서비스 리스는 클라이언트가 원하는 기간만큼 계속 실행 상태에 있게 된다.

3. 대기 큐(Wait Queue): 초기의 우선순위가 1 인 경우 정해진 기간 동안의 서비스 리스가 종료되면 대기 큐로 옮겨져 다시 한번 서비스 리스를 기다리게 된다. 이후 대기 큐에서 다시 준비 큐로 옮겨지게 되는데 이때 우선순위는 2 로 변환되게 된다. 준비 큐에서 실행 단계로 넘어가면 우선순위가 2 로 재 분류되었기 때문에 마지막 한번의 서비스 리스를 제공 받고 요청된 서비스를 종료하게 된다. 이후 다시 서비스를 사용하기 위해서는 새롭게 서비스 리스를 요청해야 한다.

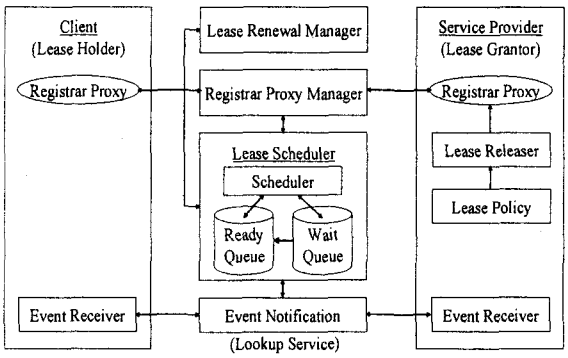

그림 3. 우선순위 기반 리스 스케줄러

우선순위 정책(Priority Policy): 네트워크상에서 지니 연합체에 등록된 클라이언트와 서비스 제공자는 클라이언트의 우선순위에 대한 정책을 정해 놓아야 한다. 우선순위를 분류하는 방법에는 여러 가지가 있을 수 있다.

예를 들어, 이전에 해당 모바일/임베디드 기기 자신에게 서비스를 제공했던 클라이언트가 필요로 하는 서비스를 요청할 경우 우선순위 0 을 부여하거나, 처

리 속도가 빠른 CPU 를 갖고 있는 기기에 높은 우선순위를 부여하는 방식으로 임의의 우선순위 부여 방법을 선택할 수 있을 것이다.

서비스 제공자 내부의 우선순위 기반 스케줄러: 또 다른 우선순위에 기반한 서비스 제공 방법으로는 각각의 서비스 제공자 내부에 스케줄러를 내장함으로써 허용된 범위 내에서 서비스 제공자 자신의 리소스를 사용할 수 있도록 하는 것이다. 이는 룩업 서비스 내부에 스케줄러를 위치하는 방식에 비해 서비스 제공자의 자원 소모 및 관리를 보다 효율적으로 해 줄 수 있게 된다.

그러나 모든 서비스가 현재 룩업 서비스에 내장되어 있는 스케줄러의 기능을 제공하기 위해서는 서비스 제공자 마다 이를 추가로 구현해야 하는 부담이 있다. 이는 네트워크를 통한 서비스 제공 시 룩업 서비스에 내장되어 있는 경우보다 과부하를 야기시키게 된다.

이벤트 알림(Event Notification): 룩업 서비스에서 리스 스케줄링 수행 결과를 서비스 제공자와 클라이언트가 인식할 수 있도록 룩업 서비스는 클라이언트와 서비스 제공자에게 이벤트로 알려주게 된다. 이를 통해 룩업 서비스는 리스 서비스를 제공 받은 클라이언트들에게 현재 사용하고 있는 리스 서비스를 재 사용할 수 있는지, 사용 할 수 없는지 등의 정보를 제공해 주게 된다.

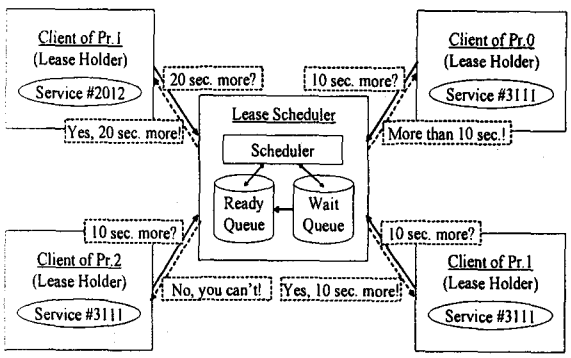

그림 4. 이벤트 알림

이와 더불어 룩업 서비스는 우선순위에 따라 해당 서비스 리스가 종료되기 전에 클라이언트에게 서비스 리스 종료 시간을 알려 줌으로써 필요에 따라 클라이언트가 다시 리스를 요청할 수 있도록 해준다.

5. 결론 및 차후과제

유비쿼터스 컴퓨팅 환경에서 임베디드 / 모바일 기기들이 필요로 하는 서비스와 기능이 증가하고 있다. 필요로 하는 다양한 서비스들을 제공하기 위해서는 이를 저장할 수 있는 더 많은 공간이 필요하다. 지니 기술을 임베디드/모바일 기기에 적용하면 네트워크를 통한 분산된 자원의 공유를 가능케 함으로써 저장 공간의 제약을 해결해 줄 수 있다. 그러나 제공 가능한 서비스 자원 보다 사용자의 요구가 많다면 모든 사용자에게 똑같이 서비스를 제공할 수는 없게 된다. 이를 위해 사용자에게 서로 다른 우선순위를 부여함으로써 필요로 하는 서비스를 제한적으로 사용할 수 있도록 하였다.

우선순위 기반의 스케줄링은 일반 운영체제(Operating System)나 실시간 운영체제(Real Time Operating System)에서 적용되고 있다. 본 논문에서 구현한 우선순위 기반 스케줄링 기법은 기본적으로 운영체제에서 커널(Kernel)이 프로세스(Process)를 스케줄하는 방법과는 차이가 있지만 필요한 서비스를 원하는 사용자 그룹에 우선순위를 부여함으로써 유비쿼터스 컴퓨팅 환경에서 다수의 클라이언트들의 사용 요구에 비해 한정된 서비스 자원에 대한 효율적 관리를 가능하게 하였다.

참고문헌

[1] Sun Microsystems, Inc., "Discovery and Join", Jini Technology Core Platform Specification, 2003

[2] Sun Microsystems, Inc., "Distributed Events", Jini Technology Core Platform Specification, 2003

[3] Sing Li, "Professional Jini", pp 298 - 301, Wrox Press Ltd., 2000

[4] Sun Microsystems, Inc., "Distributed Leasing", Jini Technology Core Platform Specification, 2003

[5] Sing Li, "Professional Jini", pp 267, Wrox Press Ltd., 2000

[6] Sing Li, "Professional Jini", pp 269 - 270, Wrox Press Ltd., 2000