JAVA SDK를 이용한 파일 클라우드 서버의 설계 및 구현

이 상 곤*, 김 충 현
*전주대학교 컴퓨터공학과 언어과학실
e-mail : neogures@naver.com, samuel@jj.ac.kr*

Design and Implementation of File Cloud Server by Using JAVA SDK

Samuel Sangkon Lee*, Chung-Hyun Kim
*Dept of Computer Engineering and Engineering, Jeonju University

요 약

개인적인 용도의 파일을 저장하고 이를 여러 디바이스에서 공유하는 클라우드 서비스가 주목을 받고 있다. Dropbox와 OAuth, PACloud를 통해 이와 같은 서비스를 구현할 수 있다. 또한 스레드 폴링을 이용하여 서버에 들어오는 여러 태스크들을 적절하게 처리할 수 있는 구현 기술을 제시하였다. 구현 기술을 설명하기 위해 소프트웨어 공학적인 여러 다이어그램을 제시하였다.

1. 서론

과거 저장 공간에 있는 데이터를 다른 공간으로 옮기기 위해서는 데이터의 이동 매체인 디스크, CD-ROM, 외장 하드 디스크 등에 옮겨야 한다. 그렇기 때문에 불편함은 물론 비용까지 소모된다. 그러나 현대 사회는 유비쿼터스(Ubiquitous)화가 가속되며 많은 변화를 겪고 있다. 그 가속을 부추긴 것은 바로 IPV4(Internet Protocol Version 4)이며, 이 프로토콜은 많은 기기를 하나로 모아 연동할 수 있는 기술이다. 현대에는 스마트 폰 어플을 이용하여 대학 도서관의 좌석을 확인하고 열람실 자리에 대해 대여 신청도 할 수 있다. 이와 같이 이제는 원격으로 모든 것을 제어할 수 있는 세상이며, 사물 인터넷의 저장 공간 또한 유비쿼터스화가 진행 중이다. 이제 저장 공간은 과거의 물리적인 형태에서 가상의 형태로 바뀌고 있으며, 웹 포털 사이트에서도 개인별 웹 하드를 충분히 제공하고 있다. 미래에는 클라우드 시스템을 도입해 자기 PC의 저장 공간 일부를 사이버 상에서 공유하고, 다른 디바이스에서도 곧바로 접근할 수 있는 환경을 만들어 다른 서비스들과 통합하여야 한다.

이와 함께 스마트 폰 이용자 수가 늘어남에 따라 현대인의 생활 패턴도 많이 바뀌고 있다. 기존에는 하나의 장소에서 업무를 보았다면 지금은 이동 중에도, 심지어 길을 걸으면서 장소에 구속받지 않고 업무를 볼 수 있는 환경이 되어 가고 있다. 또한 USB 메모리에 저장하던 개인 자료나 업무 파일들을 인터넷 상의 이메일로 저장하는 이용자도 많아졌다. 하지만 인터넷의 경우 개인 용량의 한계가 있고, 개인적인 문서들이 인터넷 서버에 그대로 남아 있음으로써 자료의 보안성 및 안전성이 떨어진다. 이를 해결하는 방안으로 클라우드 서비스(Cloud Service)가 주목받고 있다. 이 서비스는 앞에서 언급한 용량의 문제를 해결해 주며, 자료의 보안성을 보장하기 위해 개인의 가상 디스크를 제공하는 등 최신 트렌드로 주목받고 있다. 앞으로의 발전 가능성 및 차세대 컴퓨팅 환경에도 이러한 가상 디스크는 많이 사용될 것이며, 능률 및 생산성 향상 측면에서 그 활용성은 매우 높다.

2. 서버의 효율 향상을 위한 스레드 폴링

멀티 스레딩(Multi Threading) 기법은 자바(Java)를 강력한 프로그래밍 언어[1, 2, 3]로 만든 특징 중의 하나이다. 이 기법은 프로그래밍 언어 차원에서 지원되기 때문에 간단한 몇 가지 규칙만으로 쉽게 스레드 프로그램을 만들 수 있다.

풀링(Pooling) 기법 중의 하나인 스레드 폴링(thread pooling)은 일반 객체가 아닌 스레드를 풀링하는 기법으로, 기본 원리는 스레드의 재활용성이다. 할당된 일을 마친 스레드를 소멸시키지 않고, 스레드 풀(thread pool)에 저장해 두었다가 필요할 때 다시 꺼내 쓰는 개념이다. 즉, 이 개념은 스레드의 생성과 소멸에 필요한 비용을 지불하지 않겠다는 것이다.

스레드 풀은 처리해야 할 일이 등록되기 전에 생성되는데, 풀이 생성됨과 동시에 스레드들도 생성되어 풀에 대기하게 된다. 지능적인 풀은 처리해야 할 일의 증가 및 감소에 따라서 풀 안의 스레드 개수를

늘리기도 하고, 줄이기도 한다. 만약에 풀에 존재하는 스레드의 수보다 처리해야 할 일의 수가 많다면, 일이 순서대로 처리되도록 디자인 할 수도 있고, 빠른 일 처리를 위해 추가적인 스레드가 생성되도록 풀을 디자인할 수도 있다.

스레드 풀링은 객체 풀링과 마찬가지로 자원의 소모를 줄이는 효과가 있다. 무엇보다 스레드 생성 시에 많은 자원이 필요하고, 제거 역시 만만한 일이 아니다. 그렇기 때문에 자바와 같이 프로그래밍 언어 차원에서 스레드가 지원된다면 스레드가 필요할 때마다 생성하여 처리하고, 필요없어지면 버리는 일을 반복한다면 효율적인 서버 프로그래밍이 될 수 없다. 특히 스레드라는 것이 보통 외부의 요청을 처리하는 독립적인 역할을 하는 경우가 많기 때문에, 외부 요청이 잦은 경우에 이를 모두 생성하는 방식으로 처리하면 효율이 낮아질 수밖에 없다.

이를 해결하기 위해 스레드 간에 CPU를 점유하기 위한 스레드 스케줄링(thread scheduling)이 필요하다. 모든 스레드는 자신의 기본적인 실행 시간 외에 스레드 스케줄링에 일부 시간을 할애해야 한다. 만일 외부 요청에 의해서 스레드가 한꺼번에 생겼다 사라지면 실제 실행 시간 보다는 스레드 스케줄링에 많은 컴퓨팅 자원이 들어갈 것은 당연한 일이다. 따라서 네트워크 서버와 같이 여러 요청이 동시에 들어오는 응용 프로그램에서는 스레드의 관리가 컴퓨터의 실제 성능을 좌우하게 된다.

첫째로 스레드는 run() 메소드가 끝나지 않으면 종료되지 않는다는 점을 이용한다. 즉 스레드가 run() 메소드 안에 있으면 해당 스레드는 종료되지 않는다. 따라서 한번 만들어진 스레드를 run() 메소드 안에 가두어 놓으면 된다. 둘째, 자바에서 스레드를 특정 상황에서 정지할 수 있다는 점을 이용한다. 스레드의 정지 없이 무한정 실행하게 하면 부하가 심하게 걸리므로 실행 중인 스레드를 태스크에게 할당하기 전에 실행을 정지하면 된다. 이 두 가지 조건이 잘 만족되면, 스레드 폴링을 하는 중 특정 태스크(task)가 생기면 이를 적절히 실행시키면 된다.

3. 클라우드 서버에 적용

앞에서 기술한 스래드 풀링의 기법을 그대로 적용하여 클라우드 서버(cloud server)를 구축하였다. Runnable 인터페이스로 만든 스래드로 2장에서 소개한 태스크 관리를 위한 것이다. 그리고 ThreadPoolManager의 객체인 매니저를 생성하여 클라이언트의 접속 연결인 태스크를 받아들이기 전에 10개의 유휴 스레드를 생성하여 대기한다. while() 의 무한 루프를 시작하면, 서버는 server.accept()로 클라이언트(Client)의 소켓을 통한 접속을 대기하며, 클라이언트가 접속한 이후 통신하며 계속해서 일을 하게 될 태스크 클래스를 생성하고, 이를 ThraedPoolManager인 매니저에게 매개 변수로 전달하여 유휴 스레드에 일을 하도록 execute() 시켜 원활하게 하였다. 위에서 설명한 방법을 이용하여 본 논문에서 제안하는 클라우드 서버를 설계하여야 한다.

클라우드 컴퓨팅은 인터넷 기반으로 하는 컴퓨팅 기술로 인터넷상의 유틸리티 데이터를 서버쪽 프로그램에 두고 이 데이터를 다양한 장치로 불러오는 웹 기반 소프트웨어이다. 클라우드 컴퓨팅에는 다양한 서비스가 있는데, 외국 서비스로는 Google Drive, Dropbox, 애플의 iCloud, 아마존의 AWS가 있고, 국내에는 네이버 N 드라이브, 다음 클라우드, KT ucloud 등이 있다. 클라우드 컴퓨팅은 초기 Web 2.0과 SaaS(Software as a Service)와 같이 잘 알려진 기술들과 연관성을 갖는 일반화된 개념으로 IaaS(Infrastructure as a Service), PaaS(Platform as a Service) 등이 있다.

이런 다양한 클라우드 서비스는 여러 기기에서 사용할 수 있게 애플리케이션을 제공하여 언제 어디서나 사용자가 파일을 관리할 수 있다. 따라서 서비스 개발사에서는 누구나 손쉽게 자사 서비스를 활용할 수 있는 API(Application Programming Interface)를 제공한다. 본 논문에서 서비스 개발사들의 다양한 API를 활용하여 통합된 애플리케이션을 제공하고 사용자가 하나의 프로그램으로 여러 디바이스에서 공유할 파일을 관리하는 프로그램을 개발하였다. 그 중 Dropbox API의 연동 및 인증 위한 Dropbox API[10, 12]에 대해 논의하고자 한다.

4. 구현

본 연구의 구현을 위해 먼저 사용한 개발 모형은 에자일(Agile) 방법론[4, 5]이다. 이 방법론은 프로세스, 도구보다 의사소통을 강조함으로써 요구 사항 변경에 적극적으로 대처할 수 있도록 하는 방법론이다. 문서화 된 계획보다 단위 모듈 개발 결과와 고객의 신속한 요구사항 반영, 그리고 테스트에 중점을 두고 있다. 유스케이스 식별은 세 가지 기능을 구현하였으며, 유스케이스 다이어그램을 제시하였다. 식별 작업은 요구 분석 -> 설계 -> 개발 -> 테스트의 순으로 작성한다. 이 세 가지 기능의 클래스 다이어그램을, 시퀀스 다이어그램을, 그리고 유스케이스 명세서를 제공하여야 한다.

5. 결론

본 논문에서는 여러 디바이스에서 하나의 저장 공간을 공유하는 클라우드 파일 시스템을 설계하여 제작하였다. 이를 이용하면 Android 와 Windows 에서 동일한 저장 공간을 공유할 수 있다. 또한 다양한 디바이스에서 공간의 제약성을 극복하고 데이터의 손실을 방지할 수 있다. 이

시스템은 실시간 데이터 동기화가 가능하고, 타 클라우드 서비스 계정과 동기화가 가능하며, 하나의 뷰(view)에서 여러 클라우드 시스템을 관리할 수 있는 장점이 있다[13].
본 논문에서는 OAuth 인증과 Dropbox API에 대해서도 조사하였다. OAuth가 사용되기 전에는 인증 방식의 표준이 없어 개발사별 서로 다른 인증 방식을 사용하였다. 또한 보안상의 취약성이 존재하였지만, OAuth와 같은 표준화 된 인증 방식을 사용하면 보안 문제가 해결되고 인증 과정도 공유할 수 있다. 이러한 장점으로 인해 대부분의 회사에서 OAuth를 지원한다. Dropbox사에서는 역시 OAuth를 기반으로 하는 다양한 API를 지원하여 개발자가 특정 API를 선택할 수 있다. 따라서 본 논문에서도 이를 이용하여 사용자가 폴더를 지정하여 백그라운드 상에서 자동으로 파일을 업로드 하는 기능을 설계하여 구현하였다. 또한, App key를 이용하여 로그인 하지만 차후 Dropbox 아이디로 직접 로그인을 할 수 있도록 지원하고, OAuth의 최신 버전인 2.0으로 프로그램을 작성하고 이를 소프트웨어 공학적인 방법론을 이용하여 테스트하였다.
직접 UI를 개발하지 않고도 기존의 API를 이용하거나, Open Source를 이용해 자신이 필요한 부분을 바로 사용하고, 이를 수정함으로써 더 좋은 프로그램을 제작할 수 있다고 생각한다. 최신의 모바일 플랫폼의 경우에도 이러한 API만을 사용함으로써 클라우드 서버를 손쉽게 개발할 수 있다. 특히 Dropbox를 자신이 개발하는 애플리케이션에 추가함으로써 데이터 보관의 안정성을 높이고, 사용자에게는 편리함을 제공할 수 있다.
또한 스레드 풀링은 해당 어플리케이션이 이용할 CPU의 자원을 미리 점유함으로써 효율적이고 빠른 성능을 보장하는 좋은 기술이다. 미리 할당 받은 스레드들은 태스크를 더욱 빠르게 처리할 수 있다. 그렇기 때문에 한 순간에 갑자기 여러 태스크가 발생하여도 스레드 풀링을 이용하여 미리 얻은 자원으로 단 시간 내에 일을 처리 할 수 있는 것이다. 이를 적용하여 개발한 5장의 PACloud는 서버에 접속하는 클라이언트들의 태스크를 바로 처리할 스레드를 미리 점유하여 갑작스런 많은 사용자들의 접속 폭주를 완만하게 처리해 낼 수 있게 된다. 주의할 점으로는 미리 점유할 자원에 대해서 서버가 스스로 판단할 수 없기 때문에 프로그래머 자신이 서버 특성을 이해하고, 여러 번의 테스트 과정을 거쳐 점유할 자원을 미리 예측하고 연구하는 과정도 필요하다. 서버가 감당할 수 있는 클라이언트의 동시 접속자 수를 가늠하여 확보하여야 하는데, 그 방법으로 동시 접속자 수의 하루 통계치를 내어 시간대마다 유동적으로 유휴 스레드를 조절하여 CPU의 효율을 극대화하고, 전력 소모를 현저하게 낮출 수 있을 것이다.
추가 개발을 위해 향후에는 다양한 디바이스에서 사용 가능한 애플리케이션으로 개발하여야 한다. 서비스 향상을 위한 서버의 멀티 스레드 지원하고, 다른 클라우드 서비스를 통합 지원이 필요하다. 또한 드롭 박스 로그인 과정을 개선하여야 한다. 그리고 향후 파일을 파편화 하여 동시에 전송하는 멀티 스레드 풀링 기술을 이용한다면 파일을 보다 빠르게, 인터넷 자원을 효율적으로 전송할 수 있는 시스템은 굉장한 파급력을 보일 것으로 전망한다.

참고문헌

[1] Sanghyun Kim, .Net Programming - C#, Win Form, ADO, Game Publishing, 2008. (Korean)

[2] Yunmyung Kim, JAVA Programming for Brain, Hanbit Media Publishing, 2006. (Korean)

[3] Seungbak Kim, Java I/O & NIO Network Programming, Hanbit Media Publishing, 2004. (Korean)

[4] Eunman Choi, Software Engineering, 5th Ed., Jeongiksa Publishing. 2011. (Korean)

[5] Dongho Han, Android Programming with Example - Step by Step, J-Perm, 2011. (Korean)

[6] Jaekon Jeong, Do it! Android App Programming, Easysper Publishing, 2013. (Korean)

[7] Hyun-hee Jang, Programming WPF, Hanbit Media Publishing, 2008. (Korean)

[8] Daum DNA Development Network, “OAuth Look Around”, http://dna.daum.net/apis/oauth.

[9] Wikipedia, “OAuth”, http://ko.wikipedia.org/wiki/OAuth.

[10] Wikipedia, “Dropbox”, http://ko.wikipedia.org/wiki/Dropbox

[11] Wikipedia, “Cloud Computing”, http://ko.wikipedia.org/ wiki/Cloud_Computing.

[12] Dropbox, “Dropbox – Developers”, [Internet], https://www.dropbox.com/developers.

[13] Hakgeon Lee, Changho Yun, Jongwon Park, Yongwoo Lee, "An Analysis of Big Video Data with Cloud Computing in Ubiquitous City," Vol. 15, No. 3, pp. 45-52, 2014. (Korean)