이동 사무환경을 위한 메신저 프로토콜 설계 및 서버의 구현 및 분석☆

Design of Messenger Protocol, Implementation of Server and Analysis for Mobile Office Environment

온 진 호* 최 완** 조 기 환*** 이 문 근****
Jin-Ho On Wan Choi Gi-Hwan Cho Moon-Kun Lee

요 약

최근 이동사무환경에 대한 관심이 증대되고 있다. 이를 위한 '유비쿼터스 이동사무환경 시스템'의 메신저 시스템은 기존 메신저 프로토콜에서 지원하지 않았던 파일 요청, 사무환경 전송/ 요청 / 브로드캐스팅 등 이동 사무환경 메신저 시스템을 위한 여러 가지 새로운 개념들이 필요하다. 본 논문은 이러한 이동사무환경 메신저에 맞는 새로운 프로토콜을 설계하였으며, 기존의 프로토콜과의 차이점을 분석하였다. 또한 MOE-MP에 최적화된 서버를 구현하고 이에 대한 효율성 분석을 수행하였다. 새롭게 설계한 MOE-MP는 XMPP의 기본적인 프로토콜을 사용하였고, 이동사무환경에 맞도록 확장하였다.

Abstract

Recently, interest about Mobile Office Environment is increasing. Messenger systems of 'Ubiquitous Mobile Office Environment' need various kinds new concept such as request/broadcasting/transmission of working environment that existent messenger protocol does not support. Design correct new protocol to Mobile Office Environment messenger in this treatise, and analyzed existentprotocol and comparison. Also, examine characteristic of rendezvous server that was optimized in MOE-MP protocol and analyzed efficiency. MOE-MP that design newly based on basic syntax of XMPP optimized to Mobile Office Environment.

keyword : 이동사무환경, 메신저 프로토콜, 메신저 비동기 서버

1. 서 론

최근 효율적인 업무처리를 위해 이동사무환경의 관심이 증대되고 있다. 여기서 이동이란, 현재 작업 중이던 작업이 사용자의 위치적 이동이나 다른 사용자로의 전송 등으로 인해 다른 PC로 이동되어 원래의 상태로 복원되는 것을 말한다. [1]에서 제안한 '유비쿼터스 이동사무환경 시스템'의 메신저 시스템은 효율적인 사무를 위한 공동작업[2] 메신저 시스템이다.

메신저 시스템의 구현을 위해 사용할 수 있는 프로토콜은 XML기반 공개형 프로토콜인 Jabber의 XMPP[3]와 SIP의 SIMPLE[4] 프로토콜 등이 존재한다. 이 프로토콜은 일반적인 메신저 응용프로그램을 위한 프로토콜로, 사용하는 네트워 모델과 개발방식, 적용대상 등에서 큰 차이를 보이는 대표적인 메신저 프로토콜이라 할 수 있다.

사무환경전송[5][6]을 위한 메신저 시스템은 지금까지의 파일전송 개념이 아닌, 사무환경 전송을

* 정 회 원 : 전북대학교 컴퓨터공학 박사과정
jjinghott@gmail.com
** 정 회 원 : 한국전자통신연구원 연구원
wchoi@etri.re.kr
*** 정 회 원 : 전북대학교 전자정보공학부 부교수
ghcho@chonbuk.ac.kr
**** 정 회 원 : 전북대학교 전자정보 공학부 교수
moonkun@chonbuk.ac.kr
[2007/08/01 투고 - 2007/08/09 심사 - 2007/11/15 심사완료]
☆ 본 연구는 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술개발사업의 일환으로 수행하였음. [2007-S-015-01, SaaS기반 이동형 개인맞춤 사무환경 구축 기술 개발]

원하는 사용자로부터 상대 사용자의 현재 작업상태[7]를 전송 받는 전혀 다른 개념이 적용된다. 이러한 개념의 핵심 기능은 사무환경의 전송 요청과 파일의 요청, 요청된 환경의 전송선택, 전송된 환경의 복구 기능 등이 있다[5]. 이러한 요구조건 등은 앞에서 언급한 일반적인 메신저 프로토콜에서는 지원하지 않는 기능이다. 그러므로 본 논문에서는 기존의 XMPPP 문법을 기반으로 이동사무환경에 적합한 프로토콜을 설계하였다. 또한 안정적인 사용자 처리를 위하여 Coarse-grained 작업 기반 모델[8]을 바탕으로한 랑데부 서버를 설계, 구현 하였다.

논문은 크게 메신저 프로토콜과 랑데부 서버의 구현으로 구성되었다. 메신저 프로토콜 파트에서는 앞에서 언급한 대표적인 메신저 프로토콜에 대해 간단히 살펴본 후 이동사무환경 메신저의 요구사항에 대한 분석과 새롭게 정의한 Mobile Office Environment Messenger Protocol (MOE-MP)의 특징, 그리고 기존 프로토콜과의 비교를 통해 MOE-MP를 분석하도록 한다.

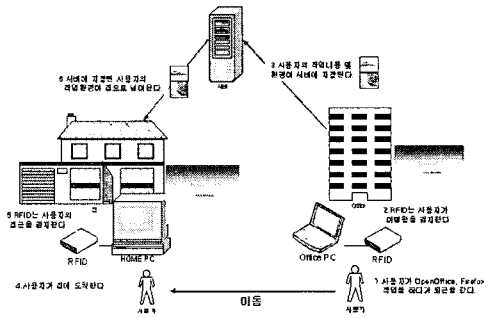
랑데부 서버의 구현에서는 랑데부 서버의 요구사항과 이를 해결하기 위한 메시지 처리 모델을 살펴보고 랑데부 서버의 아키텍처와 메시지 처리 방식을 살펴본 후 구현된 랑데부 서버의 성능을 분석하도록 하겠다.

2.관련연구

2.1 유비쿼터스 이동사무환경

유비쿼터스 이동사무환경은 사용자의 개인사무환경에 대한 이동성을 위한 시스템으로 개인과 작업환경의 자유로운 이동과 각 사용자들 사이의 원활한 작업환경 공유를 위해 제안되었다.

사무환경 소프트웨어인 한글[9], MS 오피스[10], 인터넷 익스플로러[11] 등을 사용 중인 상태 그대로 저장했다가 원하는 자리에서 복원할 수 있는 기능을 제공하며, RFID[12]를 이용한 서버 기반과 이동식 저장장치인 USB메모리 기반의 서비스를 제공한다. 사용자는 간편한 로그인 과정만 거치면 USB 메모리 장치만으로도 사무환경을 복원, 작업을 계속할 수 있어 사무환경 복원 시 데이터를 저장해 다닐 필요가 없게 된다.



(그림 1) 유비쿼터스 이동사무환경

(그림 1)은 유비쿼터스 이동사무환경에 대한 간단한 시나리오이다. RFID를 사용하는 사용자가 사무실에서 작업을 수행하던 중 사무실을 이탈하게 되면 사용자가 사용 중이던 회사의 작업은 서버에 저장되게 되고, 사용자 집에 있는 PC에서 RFID 태그가 인식되면 서버에서 사용자의 작업 중이던 환경을 그대로 복원해준다.

전체 시스템 중 메신저 시스템은 공동 작업 사용자들의 작업의 공유 및 확인을 위한 시스템으로 메신저 시스템을 기반으로 하며 각 사용자의 진행중인 작업에 대해 요청/이주/복원의 기능을 수행하는 시스템이다.

2.2 메신저 프로토콜

2.2.1 SIP/SIMPLE

SIP 는 매우 간단한 텍스트 기반의 응용계층 제어 프로토콜로서, 하나 이상의 참가자들이 함께 세션을 만들고, 수정하고, 종료할 수 있게 한다. 이러한 세션들에는 인터넷을 이용한 원격회의, 전화, 면회, 이벤트 통지, 인스턴스 메시징 등이 포함된다.

SIP는 하위에 있는 패킷 프로토콜(TCP, UDP, ATM, X.25)에 독립적이다. SIP는 클라이언트들이 호출을 시작하면 서버가 그 호출에 응답을 하는 클라이언트-서버 구조에 기반을 두고 있다. SIP는 이러한 기존의 텍스트 기반 인터넷 표준들에 따름으로써, 고장 수리와 네트웍 디버깅 등이 쉽다.

SIMPLE은 SIP 프로토콜 중 응용프로그램 계층의 메신저 프로토콜로 이를 사용하는 대표적인 메신저로는MSN 메신저[13]가 있다.

SIMPLE의 장점으로는 기업들이 중심이 되어 채택하고 있고, P2P 모델을 사용하므로 부하가 적다. 하지만, 모니터링, 데이터 기록등과 같은 서버기반의 기능을 수행하기 어렵다는 단점이 있다[14].

2.2.2 XMPP

XMPP는 실시간에 가까운 메시지 전달(near real-time messaging), 상태교환(presence)[14], 요구 응답 전송 서비스를 위한 공개된 XML 프로토콜이다. 기본 문법과 구문은1999년에 Jabber 오픈 소스 커뮤니티에서 개발되어 2002년에 XMPP WG(Working Group)이 IETF[15] instant messaging (IM) and presence 기술에 적합한Jabber 프로토콜을 수용하여 개발하는 것을 공인했다.

XMPP는 특정 네트웍 구조와 밀접한 관계를 갖지는 않지만, 일반적으로 전형적인 클라이언트-서버 구조를 통해 구현되어왔다. XMPP를 사용하는 클라이언트는 TCP 소켓을 통해 서버에 접근한다.

2.3 메시지 처리 모델

접속기반 모델은 현재 구현되어진 웹 서버들에 많이 구현되어진 모델로서, 각 요청마다 개별 스레드가 생성되는 구조로 하나의 연결을 통해 파이프라인으로 들어오는 복수개의 요구를 어떻게 처리하느냐에 따라, Coarse-grained 모델[8]과 fine-grained 모델[8]로 세분화 된다.

Coarse-grained 접속기반 모델은 각 연결마다 새로운 스레드를 생성하여 각 연결로부터 요구들을 연속적으로 읽어 들이고 그 응답을 연속적으로 전송한다. 이 방식의 문제점은 각 스레드들이 경쟁상태를 일으키는 문제가 발생한다는 것이다.

Fined-grained 접속기반 모델은 각 연결마다 스레드를 만드는 대신 각 요청마다 이를 처리하는 스레드를 만드는 것이다. 각 요구와 응답 사이에 순서는 지킬 필요가 없으나, 요구와 응답이 한번에 하나씩 이뤄져야 정보가 손실되지 않는다.

작업기반 모델은 서버에 부가되는 접속이나 요구의 수와 무관하게 일정한 수의 스레드가 생성/유지되는 모델로서, 요구를 처리하는 각 단계들을 데몬 스레드로 구성하는 구조를 가진다. 각 스레드들은 접속기반 모델과는 달리 서버 초기화 시에 생성되어 서버가 정지할 때까지 유지된다.

Fine-grained 작업 기반 모델은 구성하는 각 스레드들이 각각 요구를 처리작업의 협소한 부분만을 수행하도록 요구처리 단계를 세분화하여 각 단계마다 스레드를 부여하여 많은 스레드를 생성하게 되는 구조를 가진다.

Coarse-grained 작업기반 모델은 앞에 설명한 Fine-grained 작업 기반 모델의 문제점들의 영향을 줄이기 위해 요구처리 단계를 상대적으로 보다 적게 세분화함으로써, 병행성을 희생하는 대신 동기화 처리를 위한 부담을 줄일 수 있게 한 모델이다.

Part I. 이동 사무환경 메신저 프로토콜

3. 이동 사무환경 메신저 프로토콜

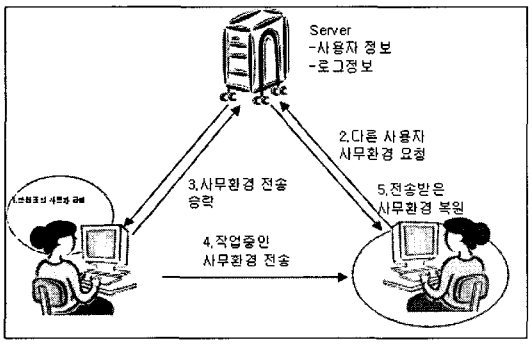
일반적인 메신저 프로토콜은 각 사용자의 메시지 전송과 파일전송을 핵심 기능으로 구성되어 있다. 일반적인 메신저 시스템과는 달리 이동 사무환경 메신저는 사무환경의 수집/전송/복원을 위해 사무환경 전송, 파일전송 요청, 사무환경 브로드캐스팅 등의 새로운 개념이 적용되어야 한다.

본 논문에서 제안하는 MOE-MP는XMPP의 기본 프로토콜을 확장하여 이동사무환경 메신저 시스템의 모든 요구사항을 만족하도록 설계하였다.

4. 이동사무환경 메신저 프로토콜의 요구사항

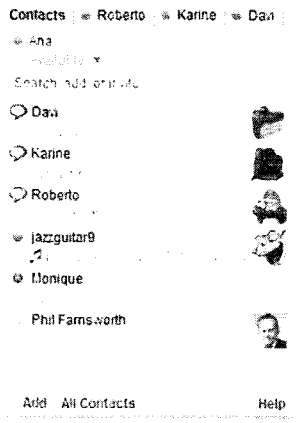
4.1 일반적인 메신저와의 차이점

이동사무환경 메신저는 각 사용자의 작업 상태에 대한 정보 유지가 필요하며, 작업중인 작업들에 대한 선택적 전송과 요청, 브로드캐스팅이 필요하다. 또한 전송된 상대방의 작업에 대한 복원도 필요하게 된다.



(그림 2) 이동사무환경 메신저 시나리오

(그림2)는 사용자가 작업 중이던 사용자의 작업 환경을 다른 사용자에게 전송하여 복원하는 시나리오를 나타낸 그림이다. (그림3)의 Google talk [16]메신저와 같이XMPP 프로토콜을 사용하며, XMPP를 사용하는 다양한 클라이언트와 호환되어 사용될 수 있다. 이러한 일반적인 메신저는 단순한 파일 전송을 위한 기능만 포함되어 있다. 사무환경 전송을 위해서는 사무환경을 요청하는 사용자가 다른 사용자의 현재 진행중인 작업을 확인할 수 있어야 하며, 요청을 받은 사용자는 진행중인 작업 중 현재 원하는 작업만을 선택적으로 전송할 수 있어야 한다. 또한 전송이 결정된 작업에 대해서 세션을 저장하고, 요청자에게 전송 후 복원까지 자동으로 수행되어야 하지만, 단순한 파일전송 기능으로는 이러한 작업을 지원할 수 없다.



(그림 3) XMPP를
사용하는Google talk 메신저

4.2절에서는 Google talk 메신저와 같은 일반적인 메신저가 지원하지 않는 이동사무환경을 위한 메신저가 포함해야 할 기능들을 요약하였다.

4.2 사무환경 전송에서의 프로토콜 제약사항

이동사무환경 메신저의 사무환경전송은 다음과 같은 요구사항들이 만족되어야 한다.

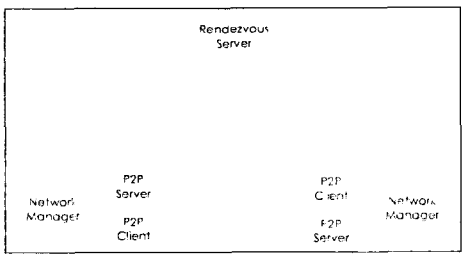
- 사무환경 전달 기능
  - 전송자는 작업중인 작업리스트 안에서 원하는 작업을 전송할 수 있어야 한다.
  - 수신자는 자신이 받고자 하는 작업들을 선택할 수 있어야 한다.
  - 사무환경 전송이 완료되면 선택적으로 작업을 복구 시킬 수 있어야 한다.
- 사무환경 요청 기능
  - 요청자는 상대방에게 작업전송을 요청할 수 있다.

- 전송자는 요청자의 요청에 자신의 작업리스트 중 원하는 작업만을 선택하여 리스트를 전송할 수 있어야 한다.
- 전송자의 작업리스트를 확인 후 요청자는 자신이 원하는 작업을 선택적으로 전송 받을 수 있어야 한다.
- 사무환경 전송이 완료되면 선택적으로 작업을 복구시킬 수 있어야 한다.

• 선택 사용자에 대한 그룹 전송 기능 (브로드캐스팅)
- 전송자는 현재 자신의 친구리스트 중, 활성화 되어 있는 사용자들을 대상으로 자신의 작업을 전송할 수 있어야 한다.

5. 기본 네트웍 구현 모델

(그림 4)와 같이 MOE-MP는 사무환경 전송을 위해 XMPP의 전통적 네트웍 모델인CS(Client-Server) 모델[17]을 사용하지 않고, P2P[18]와 CS모델의 혼합 모델을 사용하여 사무환경 전송을 수행한다. 클라이언트-서버 모델을 사용하는 가장 큰 이유는 정보의 중앙 관리이다. 각 사용자들의 모든 작업은 서버에 의해서 통제되며, 사무환경 전송과 전송요청에 대한 모든 정보는 서버에 기록된다. 이를 통해 각 사용자의 작업에 대한 트레이스 정보를 관리할 수 있는 장점이 있다. 서버의 부하를 줄이기 위해 파일전송과 사무환경 전송은 P2P의 형태로 전송된다. 이 경우에도 전송 데이터는 서버에 의해 통제되며, 전송 실패와 재전송의 경우는 서버의 재전송 명령에 의해 수행된다.



(그림 4) 네트웍 연결구조

6. MOE-MP의 추가 프로토콜

(그림 2)의 시나리오와 같이 사무환경 전송을 수행하기 위해서는 XMPP에서 제공되지 않는 몇 가지의 프로토콜이 필요하다.

4.2절에서 설명한 요구사항을 만족하기 위해 추가된 프로토콜은 다음과 같다.

가) 환경설정 파일 복원
랑데부 서버에 저장되어 있는 사용자의 개인 환경설정을 로그인 시 복원한다.

나) 사무환경 요청, 전달 기능
A. 사무환경 요청 기능
B. 사무환경 전달 기능
C. 브로드캐스팅 기능

다) 그룹에 대한 관리
XMPP의 특성상 그룹에 대한 기능이 이동사무환경 메신저와는 맞지 않다.

라) XMPP의 Roster구조상 독립적으로 그룹추가를 수행할 수 없음.

마) 그룹 명 변경 기능을 추가함.

바) 그룹 전체 삭제 기능을 추가함.
XMPP는 각 사용자만 삭제할 수 있도록 정의되어 있다.

사) 사용자 추가 시 동료 추가 확인
XMPP는 동료 추가 시 확인 절차를 거치지 않고 바로 추가된다. 이러한 문제를 해결하기 위해 동료 추가 시 추가된 사용자의 확인 절차를 추가 하였다.

6.1 전송데이터 타입

사무환경 전송을 위한 데이터는 사용자가 작업중인 파일과 파일의 현재 환경 정보, 그리고 사용 시스템 환경에 대한 정보가 저장된다. 이러한 정보들이 하나의 파일로 묶어져 상대 PC로 전송 되게 된다.

이와 같은 전송을 수행하는MOE-MP의 명령어

는 대부분의 경우 XMPP와 동일한 형태로 구성되어 있다.

사무환경 요청의 경우를 통해 MOE-MP 프로토콜의 기본 형태를 설명하도록 하겠다.

사용자가 파일 전달 요청(P2P Stream Initiation)

- 클라이언트는 상대방(Romeo)에게 파일 요청을 송신한다.
- from은 자기 ID를 to는 상대방 ID를 표기하고 요청 Type은 set을 적는다. file element와 address의 자신의 p2p 서버 주소를 보낸다

• 파일요청

```
<iq to='romeo@example.net'
from='juliet@example.com/balcony'
type='get' xml:lang='en'>
<query xml:ns='http://rts.chonbuk.ac.kr/
protocol/p2psi'>
<address ip='사용자IP' port='사용자 Port' />
<file name='파일이름' size='파일길이'
date='최근수정일' digest='해쉬값' />
</query>
</iq>
```

상대방(Romeo)은 파일 전달 요청에 대한 응답은 수락 시 type에 result를 기술하고 address element에 전달인지 요청인지에 대한 표시로 type에 receive를 기술한다. 그리고 각 address, file에 관한정보를 다시 한번 기술하여 이를 알린다.

• 파일요청 응답

```
<iq to='juliet@example.com' from='romeo@
example.net' type='result' xml:lang='en'>
<query xml:ns='http://rts.chonbuk.ac.kr/
protocol/si'>
<address ip='사용자IP' port='사용자 port'
type='receive' />
<file name='파일이름' size='파일길이'
date='최근수정일' digest='해쉬값' />
</query>
</iq>
```

위 과정을 통하여 상대방(Romeo)은 클라이언트의 P2P 정보를 알고 있으므로 전송 과정과 같이 파일을 요청하게 된다.

• P2P 정보 송신

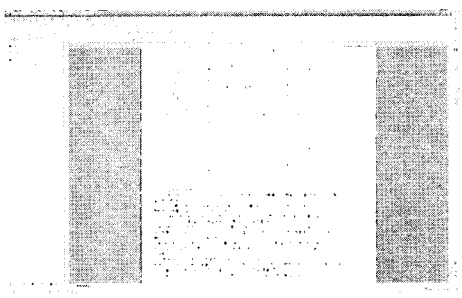
```
<iq to='juliet@example.com' from='romeo@
example.net' type='get' xml:lang='en'>
<query xml:ns='http://rts.chonbuk.ac.kr/
protocol/p2psi'>
<address ip='사용자IP' port='사용자 Port'/>
<file name='파일이름' size='파일길이'
date='최근수정일' digest='해쉬값' />
</query>
</iq>
```

• 사무환경 전송완료 송신

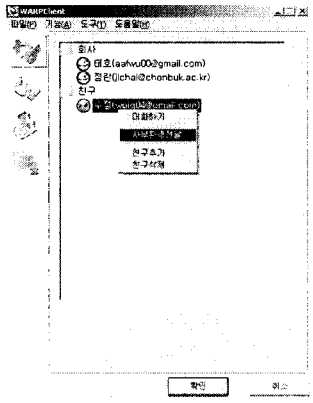
```
<iq to='romeo@example.net' from=' juliet@
example.com' type='result' xml:lang='en'>
<query xml:ns='http://rts.chonbuk.ac.kr
/protocol/si'>
<address ip='사용자IP' port='사용자 port'
type='send' />
<file name='파일이름' size='파일길이'
date='최근수정일' digest='해쉬값' />
</query>
</iq>
```

6.2 메신저 동작 예

현재 작업중인 파일이 (그림 5)와 같다면,

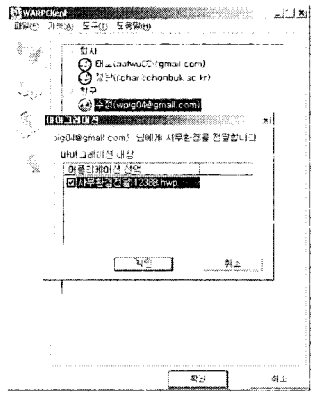


(그림 5) 현재 작업환경



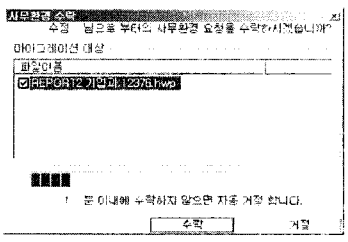
(그림 6) 단일 사용자에게
사무환경 전송

다른 사용자로의 사무환경 전송은 (그림6)와 같다.



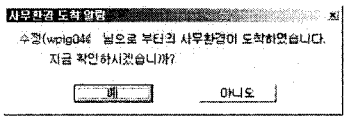
(그림 7) 사무환경 전달목록

(그림 7)과 같이 사무환경 전송 시 자신이 전송을 원하는 작업에 대해 전송을 수행한다.



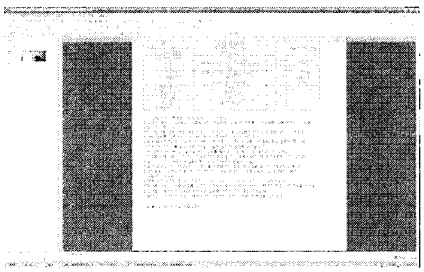
(그림 8) 사무환경 수신목록

(그림 8)과 같이 수신자측에는 전송자측에서 보내온 사무환경 리스트가 출력되고, 수신자는 선택하여 사무환경을 전송 받는다.



(그림 9) 사무환경 도착 알림

(그림9)은 전송이 완료된 후 복원을 원하는 작업에 대한 확인 메시지이다. 이 선택에 의한 전송된 작업은 원래의 작업 상태로 복원된다. 다른 사용자 PC에서 복원된 사무환경은 (그림10)과 같다.



(그림 10) 다른 사용자 PC에서 복원된
사무환경

6.3 MOE-MP와 기존 프로토콜과의 비교

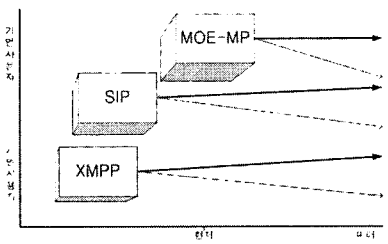
기존의 메신저 프로토콜과 MOE-MP를 비교하면 (표1)과 같은 차이점을 보인다.

(표1) MOE-MP와 기존 프로토콜 비교

		XMPP	SIP	MOE-MP
기술측면	모 델	Client-Server	Peer-To-Peer	Client-Sever + P2P
	소프트웨어 공개 방식	Open Source	Closed Source	Open Source
	Syntax	xml 기반	자체 Syntax	xml 기반
	암호화방식	SASL/ TLS	TLS	SHA1, MD5

기능측면	채팅		O		
	파일전송	일반 파일 전송	O	O	O
		사무 환경 전달	X	X	현재 작업 중인 문서의 상태 그대로를 상대방에 전송 후복원
		사무 환경 요청	X	X	상대방의 작업리스트 확인후 원하는 작업에 대한 전송 요청을 수행, 상대방이 수락하면 요청자의 PC에 작업환경 복원.
	그룹관리	그룹 추가	X	그룹 홀로 존재 가능	사용자가 존재해야만 그룹이 존재
		그룹 변경		O	O
		그룹 삭제		O	O

MOE-MP이 다른프로토콜과 가장 큰 차이점은 현재 작업중인 공동 문서에 대해서 현재 상태를 다른 사용자에게 전송하고, 브로드캐스팅을 수행할 수 있다는 것이다. MOE-MP는 현재 이러한 목적을 수행하기 위한 최적화된 프로토콜이라고 할 수 있다.



(그림 11) 각 메신저의 사용 도메인

위의 (그림11)은 현재 대표적인 프로토콜들의 사용 도메인에 대한 그림이다. 위의 그림에서 검은색은 화살표는 지금까지 개발된 프로토콜이 추구하는 방향이고, 하늘색의 화살표는 여러 가지 사용목적에 따라 추가되고 있는 프로토콜의 추구하는 방향을 나타낸다. XMPP는 현재 다른 프로토콜보다 개인 사용자를 위한 응용프로그램에 많이 적용되고 있다. SIP의 SIMPLE 프로토콜은 XMPP 보다 기업이 주도하여 개발하고 있는 프로토콜이기 때문에 그 사용 영역도 기업의 사내 메신저나 다른 시스템에 도입되어 사용되는 경우가 많다. MOE-MP는 이동사무환경 지원 메신저의 프로토콜이라는 목적에 맞게 개인 사용자보다 사무환경에서 응용되어 사용될 수 있는 응용계층 메신저 프로토콜이다.

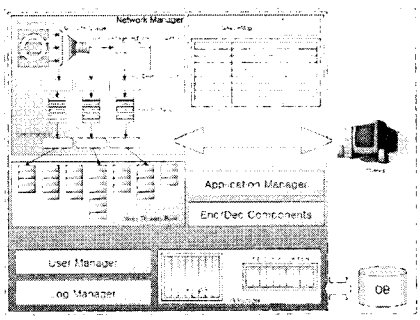
Part II. 랑데부 서버

7. 서버 아키텍처

MOE-MP는 '워프클라이언트' 와'랑데부서버'에서 구현되었다.

서버는 사용자의 사무환경 정보와 다수의 사용자들에 대한 정보들을 관리하며 클라이언트간의 메시지 전달을 위한 중간자적인 역할을 수행한다. 서버의 가장 큰 기능은 각 클라이언트간의 안정적인 메시지 전송이며, 다수의 클라이언트를 접속관리하고, 클라이언트 간의 사무환경 전송에 대한 정보를 기록하는 것이다.

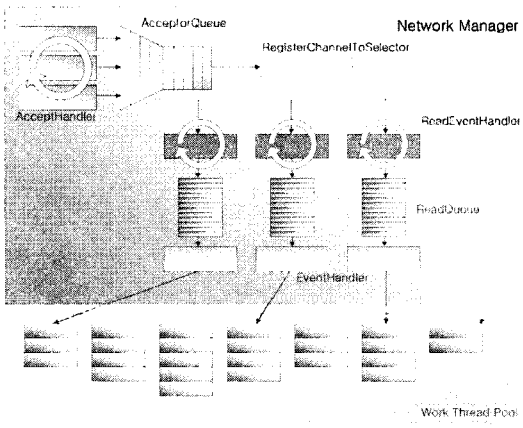
랑데부 서버는 Java NIO[19]의 Non-Blocking 모델[20]을 사용한 비동기 통신[21]을 수행하며, 성능의 향상을 위해 메시징 큐잉, Database Connection Pooling, Object Pooling[22], Work Thread Pool[23], Java Reflection 기법[24] 등이 적용되었다.



(그림 12) 랑데부 서버 아키텍처

서버의 아키텍쳐는 (그림12)과 같다. 서버는 스레드의 생성을 제한하고, 효율적인 스레드풀링을 적용하여 CPU 자원을 효율적으로 분배하도록 설계하였다.

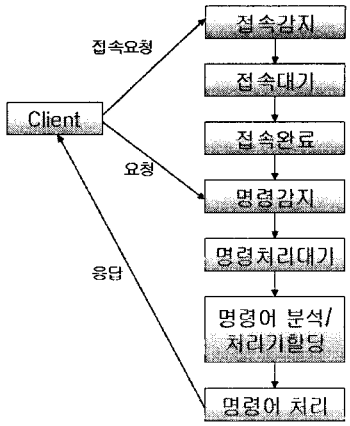
자바의 NIO는 일반적인 select() 메소드를 사용하지만, JVM[25]의 구현에 따라 각 운영체제에서 다른 메시지 처리 방식으로 처리된다. 대표적인 방식으로는 xBSD의 kqueue[26], 리눅스의 epoll [27] 방식 등이 있다.



(그림 13) 네트웍 매니저

랑데부 서버의 네트웍을 담당하는 Network Manager의 구조는 (그림 13)과 같고, 클라이언트의 접속과 메시지 이벤트의 처리 사이클은 (그림 14)와 같다. 운영체제의 메시지 처리 방식은 JVM에서 매핑되기 때문에 자바에서는 모두 select방식으로 구현되게 된다. 접속요청은 Accept처리 모듈에서 사용자의 접속을 감지하게되며, 이를 ReadEvent 처리 모듈에 등록하는 방식으로 수행된다.

ReadEvent 처리 모듈은 클라이언트의 요청을 대기하며, 요청이 감지되면 사용자로부터 명령을 읽어 들여 명령어 분석/처리기인 EventHandler에 명령을 전달한다. EventHandler는 사용자의 명령을 분석하고, WorkThreadPool에서 해당하는 처리 스레드에 명령처리를 지시하는 역할을 수행한다.



(그림 14) 랑데부 서버의 메시지 처리 사이클

운영체제의 메시지 처리 방식은 JVM에서 매핑되기 때문에 자바에서는 모두 select 방식으로 구현되게 된다. 접속요청은 Accept처리 모듈에서 사용자의 접속을 감지하게 되며, 이를 ReadEvent 처리 모듈에 등록하는 방식으로 수행 된다.

ReadEvent 처리 모듈은 클라이언트의 요청을 대기하며, 요청이 감지되면 사용자로부터 명령을 읽어 들여 명령어 분석/처리기인 EventHandler에 명령을 전달한다. EventHandler는 사용자의 명령을 분석하고, WorkThreadPool에서 해당하는 처리 스레드에 명령처리를 지시하는 역할을 수행한다.

Coarse-grained 작업 기반 모델을 기반으로 한 랑데부 서버는 상대적으로 적은 수의 스레드를 동시 실행하여 병행성은 줄이는 대신 스레드 사이에 필요한 동기화를 줄이며, 부가적인 스레드 사이의 문맥 교환의 추가 부담을 줄일 수 있도록 설계되었다.

작업 기반 모델에서 각 스레드들은 순차적으로 작업을 처리한다. Coarse-grained 작업 기반 모델의 병행성은 요구 처리 작업의 일부만을 각 스레드가 수행하게 함으로써, 전체 작업 처리가 파이프라인 형식으로 됨으로써 이루어 진다. 하지만, 이 파이프라인 형식의 처리는 작업들 중 하나의 처리 시간이 길어질 경우 이후의 작업들의 처리

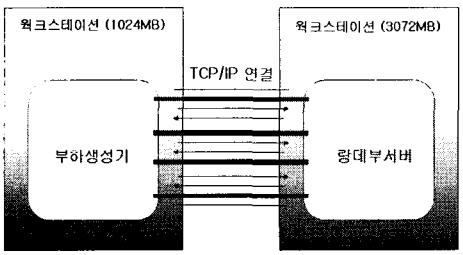
가 지나치게 지체되는 문제를 야기시킨다. 이러한 문제점을 해결하기 위해 랑데부서버는 처리 순서가 중요한 경우는 순차적으로 처리를 하고, 순서가 중요하지 않은 명령에 대해서는 길어지는 부분의 처리와 함께 이후 작업들을 병렬적으로 실행하는 방식을 선택하였다.

8. 서버 효율성 분석

8.1 실험환경

(그림 15)은 실험 구조를 설명하는 그림이다. 실험에 이용된 구조를 살펴보면 크게 3가지로 분류된다. 첫 번째는 요구를 보내주는 부하생성기, 두 번째는 그 요구들을 처리하는 랑데부 서버, 마지막으로 세 번째는 네트웍 부분이다.

부하생성기가 탑재된 컴퓨터는 운영체제로 Windows 2003을 사용하였고, 1024MB의 메모리를 사용하고 있다. CPU는 AMD MP 1700+ 2개를 사용하는SMP(Symmetric Multiprocessing)[28] 구조를 가진 시스템이다.



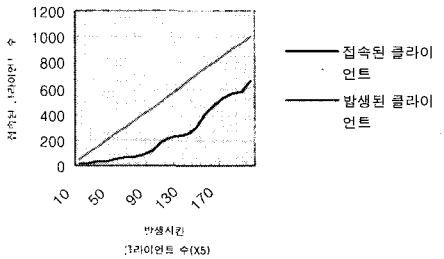
(그림 15) 실험 환경

랑데부 서버가 탑재된 시스템은 운영체제로 Windows 2003을 사용했으며, 3072MB의 메모리, 그리고 SMP를 지원하는 AMD MP 2800+ 2개를 가졌고, 15K의 SCSI를 시스템 디스크로 사용하였다.

마지막으로 네트웍은 100Mbps의 내부 네트웍을 사용하였다.

8.2 사용자 폭주로 인한 동시 접속자수 증가 실험

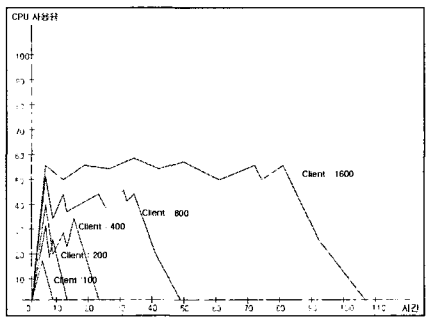
부하생성기는 2초에 한번씩 5번에 걸쳐 100 개의 메시지를 서버로 전송하는 클라이언트를 정해진 숫자만큼 생성한다. 생성된 스레드는 모든 메시지를 전송한 후 소멸한다.



(그래프 1) 접속증가에 따른 랑데부 서버의 동시접속 허용량

랑데부서버는 접속한 총 클라이언트의 숫자가 한번에100개씩 접속을 시도한 순간부터 동시접속자수가 증가하기 시작한다. 한 개의 클라이언트가 100개씩의 에코 메시지를 전송하도록 되어 있으므로 발생시킨 클라이언트의 수가 100이라면, 동시 최대 접속자의 수는 총 발생된 500중 187개임을 나타내고 이고, 랑데부 서버는 최소 187개에서 최대 18700개의 메시지를 처리하고 있다는 것을 알 수 있다.

8.3 CPU 사용률



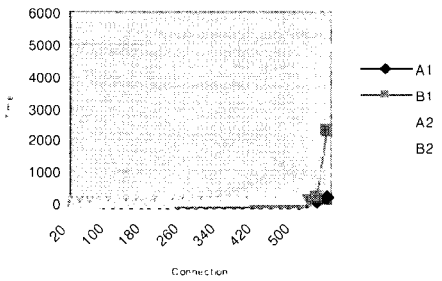
(그래프 2) 동시접속 클라이언트 별 CPU 사용률

랑데부서버가 처리하는 클라이언트가 증가할수록 서버의 CPU 점유율이 증가하게 된다. (그래프 2)를 살펴보면, 클라이언트가 증가하여도 CPU 점유율 60% 미만을 유지하며 사용자의 요청을 처리하게 된다.

8.4 모델 별 성능평가

서버는 시스템의 자원을 최대한 활용하여 최소의 자원소모에서의 최대 성능을 유지할 수 있어야 한다. 이를 위해 랑데부서버 모델과 비동기 모델에 큐를 적용한 비교서버를 구현하여 랑데부 서버의 효율성을 분석하였다.

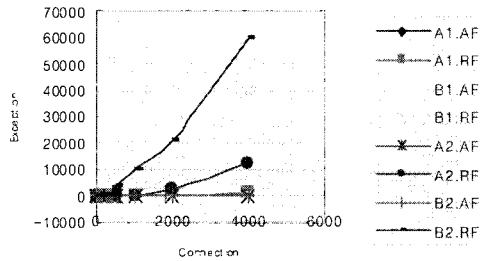
A1: 랑데부 서버 - 부하가 적을 때
B1: Non-Blocking / Queue - 부하가 적을 때
A2: 랑데부 서버 - 과부하 적용
B2: Non-Blocking / Queue - 과부하 적용



(그래프 3) 모델별 동시접속자

저부하 테스트의 경우, (그래프 3)에서 보는 것과 같이 접속자 1000이상부터 급격한 성능차이를 보이고, 고부하 테스트에서는 500명을 기점으로 B2에서 발생한 많은 오류로 인해 연결된 많은 클라이언트가 해제되었다. 이로 인해 연결된 클라이언트수가 최초 부하발생기에서 생성한 클라이언트보다 적어지게 되며 훨씬 적은 수의 메시지를 처리하게 된다. (그래프 3) B2 그래프의 성능그래프는 접속자가 많아질수록 성능이 높게 표현되었다.

AF: 접속오류
RF: 명령어 읽기 오류



(그래프 4) 모델 별 동시접속 오류 발생

(그래프 4)는 (그래프3)와 연관하여 분석할 수 있는 그래프로 서버의 내부 로직이 복잡해질수록 B의 경우에서 오류횟수의 급격한 증가를 보이고 있다는 것을 알 수 있다.

9. 결론

본 논문은 이동사무환경에 대한 간단한 설명과 이 환경에 필요한 메신저 시스템의 프로토콜 설계를 위해 기존 프로토콜을 분석/비교하고 새로운 MOE-MP를 설계하였다. MOE-MP는 이동사무환경의 사무환경 공유 개념을 적용하여 다자간의 작업을 공유할 수 있도록 설계된 프로토콜이다. 기존의 메신저 프로토콜과의 비교를 통해 사무환경에서 가지는 MOE-MP의 우위를 입증하였다.

구현적인 측면에서는 MOE-MP를 만족하는 클라이언트/서버를 설계/구현하였고, 구현한 서버의 특징과 이동사무환경 서비스를 제공할 때의 서버 효율성 분석을 수행하였다. 향후 연구에는 현재 고정적인 시스템에서 이뤄지는 MOE-MP를 이동 시스템에 맞도록 개선하는 과정이 필요할 것이다.

참 고 문 헌

[1] 국가기술은행, "유비쿼터스 이동사무환경 시스템기술" http://www.ntb.or.kr/front/techTrans/

SaleTechView.asp?techCode=S2006000015&seq=165&thisTime=129

[2] Wikipedia, Collaboration, "http://www.wikipe dia.com/collaboration"

[3] Jabber Software Foundation, "http://xmpp.org", Retrieved 15 October 2006.

[4] The SIP Center, "http://www.sipcenter.com/ sip.nsf/html/What+Is+SIP+Introduction", Retrieved 15 October 2006.

[5] Dag Johansen, Havard Johansen, Robbert van Renesse, "Environment Mobility-Moving the Desktop Around," ACM, Vol.77, pp.150-154, 2004

[6] Byeong-thaek Oh, Taein Hwang, Hojin Park, "A Study on Common Application Context Migration Method for Mobile Environment," Digital Contents Society, Vol.6, No.1, 2005

[7] Taein Hwang, Hojin Park, "Desktop Migration System Based on Dynamic Linking of Application Specified Libraries", Proceedings of International Conference on Advanced Communication Technology, 2006

[8] 제영희, 낭종호 "멀티스레드 웹 서버의 개념적모델링"한국정보처리학회 춘계학술발표논문집 제24권 제2호, 1997. pp.685-688.

[9] Hangul Office, http://www.haansoft.com/

[10] MicroSoft Office, http://en.wikipedia.org/wiki / Microsoft_Office

[11] MS Internet Explorer, http://www.microsoft.com/korea/windows/products/winfamily/ie/default.mspx

[12] Magrassi, P., & Berg, T. (2002). A world of smart objects: The role of auto-identification technologies. Gartner Dataquest. Reference Number: R-17-2243.

[13] MSN Messenger, http://get.live.com/messenger/overview

[14] presence http://en.wikipedia.org/wiki/Instant_ messaging

[15] IETF, http://www.ietf.org/overview.html

[16] Google talk, http://www.google.com/talk/

[17] Hirsch, B.: Building an Open, Client/Server Application, Interact, Volume 12, Issue 10, 100 - 115. 1992

[18] Dejan S. Milojicic, Vana Kalogeraki, and Rajan Lukose, er al. : Peer-to-peer Computing. HP Laboratories, HPL-2002-57, March 8th.2002

[19] O'Reilly On Java.com, "http://www.onjava.com/pub/a/onjava/2004/09/01/nio.html?page=1," 2006. 10.

[20] Non-blocking, http://en.wikipedia.org/wiki/ Non-blocking_I/O

[21] R. Felderman and L. Kleinrock. "An upper bound on the improvement of asynchronous versus synchronous distributed processing." In SCS Multiconference on Distributed Simulation V.22, pages 131 - 136, Jan. 1990.

[22] Object Pool, "http://en.wikipedia.org/wiki/ Object_pool"

[23] Thread Pool, "http://en.wikipedia.org/wiki/ Thread_pool_pattern"

[24] Sun, Java Reflection Document. http://java. sun.com/javase/6/docs/technotes/guides/reflection/index.html 2006. 10.

[25] JVM, http://en.wikipedia.org/wiki/Java_ Virtual_ Machine

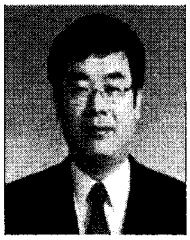
[26] Kqueue, http://wiki.netbsd.se/kqueue_tutorial

[27] Epoll , http://www.xmailserver.org/linux-patc hes/nio-improve.html

[28] P. Stenstrom, E. Hagersten, D.J. Li, M. Martonosi, and M. Venugopal. Trends in Share d Memory Multiprocessing. IEEE Computer, vol. 30, no. 12 pp. 44-50, December 1997.

◐ 저 자 소 개 ◑

온 진 호(Jin-Ho On)
2006년 원광대학교 컴퓨터정보통신공학 학사
2008년 전북대학교 컴퓨터공학 석사
현재 전북대학교 컴퓨터공학 박사과정
관심분야 : 분산/병렬처리시스템, 정형기법



최 완(Wan Choi)
1983년KAIST공학석사
1985~현재 한국전자통신연구원 연구원
관심분야 : 소프트웨어 공학, 미들웨어, 소프트웨어 스트리밍

조 기 환(Gi-Hwan Cho)
1985년 전남대학교 계산통계학과 학사
1987년 서울대학교 계산통계학과 석사
1996년 영국 Newcastle 대학교 전산학과 박사
1987년 ~ 1997년 한국전자통신연구원 선임연구원
1997년 ~ 1999년 목포대학교 컴퓨터과학과 전임강사
1999년 ~ 현재 전북대학교 전자정보공학부 부교수
관심분야 : 이동컴퓨팅, 컴퓨터통신, 무선네트워크 보안, 센서네트워크, 분산처리시스템

이 문 근(Moon-Kun Lee)
1989년 The Pennsylvania State University, Computer Science 학과 이학사
1992년 The University of Pennsylvania, Computer and Information Science 학과 이공학석사
1995년 The University of Pennsylvania, Computer and Information Science 학과 이공학박사
1992년 5월~1996년 1월 : 미국, Computer Command and Control Company, Computer Scientist 로 근무
1996년 ~ 1998년 전북대학교 컴퓨터과학과 전임강사
1998년 ~ 1999년 전북대학교 컴퓨터과학과 조교수
1999년 ~ 2007년, 전북대학교 전자정보 공학부 부교수
현재 전북대학교 전자정보 공학부 교수
관심분야 : 정형기법, 소프트웨어 재·역공학, 실시간 시스템, 운영체제, 형식언어, 병렬함수 언어, 컴파일러 등