졸업 프로젝트 최종 보고서

유무선 통합 메신저

팀명 : 바이러스 (VIRUS) -12조

담당 : 최준수 교수님

조원 : 김미영 (20017313)

김경미 (20017420)

정희진 (9805076)

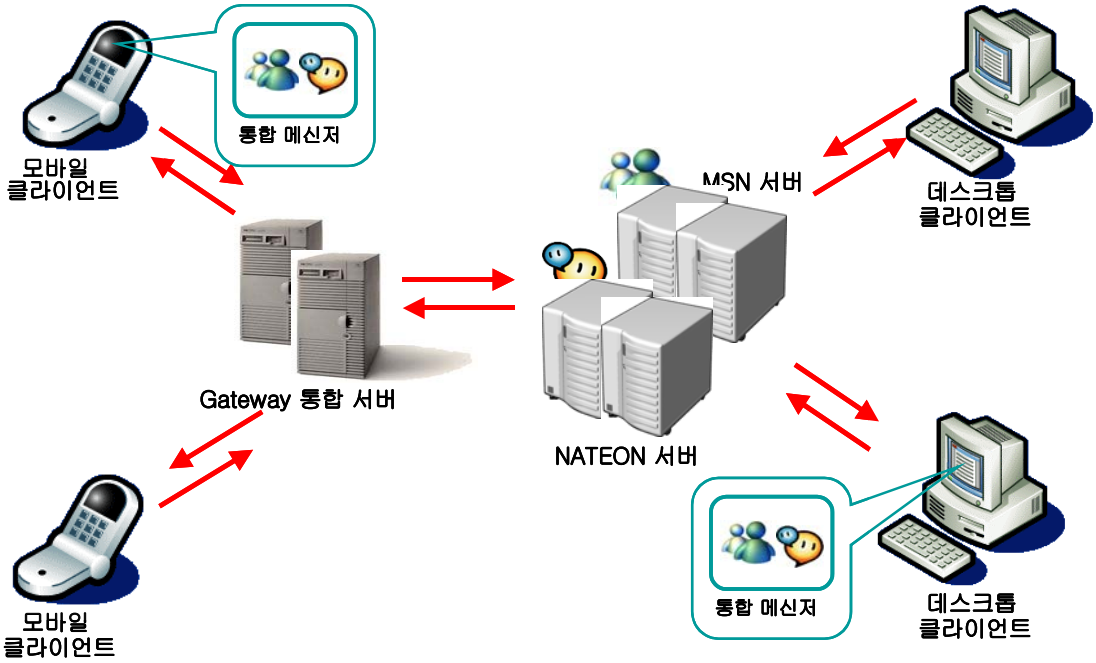
홍성민 (9805086)

정석호 (9805072)

목 차

1. 최종 보고서 요약문

메신저의 보급이 확산됨에 따라, 다양한 메신저들이 나오게 되었고, 메신저 사용자들도 한 두 개 이상의 메신저 계정을 가지게 되었다. 하지만 사용자가 여러 계정을 가지게 되면서, 각각의 메신저 프로그램을 설치하는 것이 번거로울 수 밖에 없고 관리하기에도 불편한 점이 따르게 되었다. 이러한 불편한 점을 해소하기 위해서 사용자에게 통합된 메신저 환경을 제공할 수 있다면 더욱 편리하게 메신저 서비스를 이용할 수 있을 것이다.
또한 모바일 폰의 사용인구수가 급증함에 따라 서비스 공급 회사들은 여러 가지 서비스를 내놓으며 경쟁을 하고 있고 요즘에는 데스크탑 PC와 모바일간 연동을 하려는 움직임이 서서히 일고 있다. 그에 발맞춰 위에서 언급한 통합메신저가 모바일 폰과 연동이 될 수 있다면 굳이 PC앞에서가 아니더라도 언제 어디에서나 메신저 서비스를 즐길 수 있게 된다.



따라서 이번 프로젝트에서의 개발목표는 위의 그림처럼 다양한 메신저를 하나의 클라이언트로 통합하고 데스크탑과 모바일간 커뮤니케이션이 가능하도록 유무선 연동을 하는 것이다.

최종 발표로 준비한 내용은 MSN 라이브러리와 NATEON 라이브러리를 구현하여 PC 통합 메신저와 모바일 통합 메신저, 그리고 Gateway Server를 구현하여 유무선 통합 메신저를 제작하는 것을 목표로 하고 있다.
이와 같은 목표를 위해 지금까지 우리 팀이 한 내용은 다음과 같다.

1. MSN과 NATEON 프로토콜 분석
2. MSN 프로토콜 라이브러리 구현
3. NATEON 프로토콜 라이브러리 구현
3. PC 통합 메신저 제작
4. Mobile 통합 메신저 제작
5. Gateway Server 구현

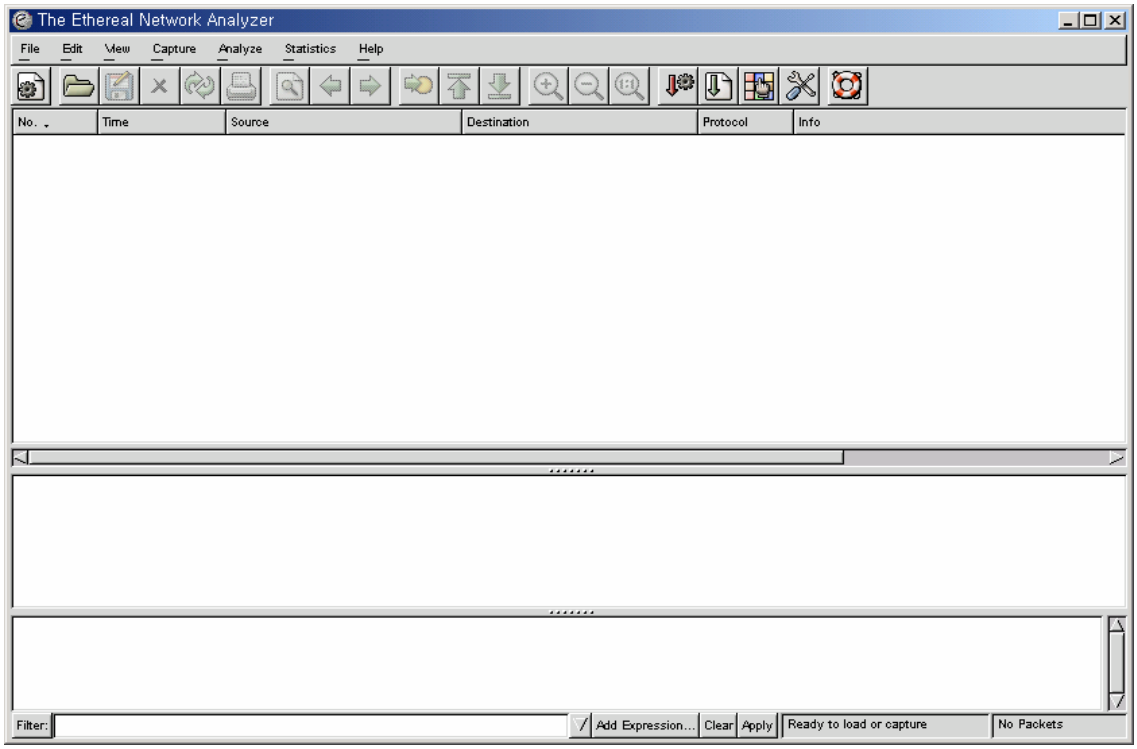
세부적인 내용으로 2장에서 Ethereal이라는 패킷 분석 프로그램을 통해 MSN과 NATEON 프로토콜을 분석하는 방법에 대해서 살펴 본 후, 3장에서 분석한 프로토콜을 통해 MSN 라이브러리를 구현한 내용, 4장에선 NATEON 라이브러리를 구현한 내용을 설명하겠다.
그리고 5장에서 분석한 프로토콜을 통해 구현한 MSN, NATEON 라이브러리를 가지고 PC 통합 메신저를 제작한 내용에 대해서 살펴본 후, 6장에선 무선 파트와의 연동을 위해 Gateway Server와 Mobile Client 간에 프로토콜을 제작하고 라이브러리로 구현한 내용을 알아보도록 하겠다.
그리고 7장에서 자체 제작한 무선 파트 라이브러리를 가지고 Gateway Server를 구현한 내용에 대해서 살펴보고, 마지막으로 8장에서 통합 플랫폼인 위피 플랫폼 기반에서 Mobile 통합 메신저를 제작한 내용에 대해서 자세히 살펴보겠다.

2. MSN과 NATEON 프로토콜 분석

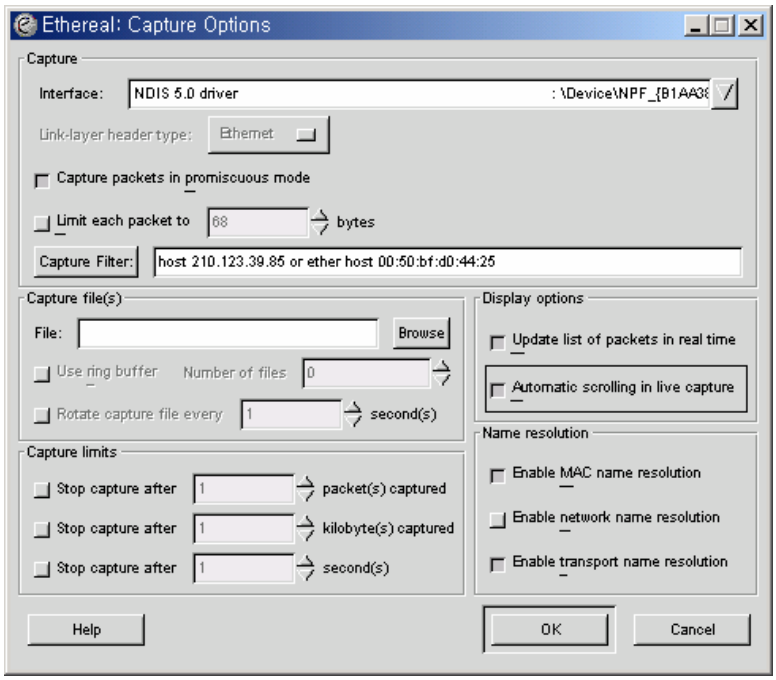
2.1 Ethereal에 사용한 프로토콜 분석

MSN, NATEON 메신저를 구현하기 위해서 가장 먼저 해야 할 일은 클라이언트와 서버가 주고 받는 언어인 프로토콜을 분석하고 분석한 결과를 라이브러리로 구현하는 것이다.
아쉽게도 MSN과 NATEON 모두 자사의 프로토콜을 공개하고 있지 않다. 따라서 우리는 Ethereal이라는 패킷 분석 프로그램을 사용하여 수많은 테스트의 결과로 프로토콜을 분석하였다.
그럼, NATEON을 예로 들어 Ethereal을 사용하여 프로토콜을 분석하는 과정을 살펴보도록 하겠다.

NATEON의 Protocol을 분석하기 위해서 먼저 NATEON이 주고 받는 프로토콜을 Ethereal이라는 프로그램을 사용하여 캡쳐 하였다. 다음은 Ethereal 프로그램의 초기화면을 보여주는 것이다.



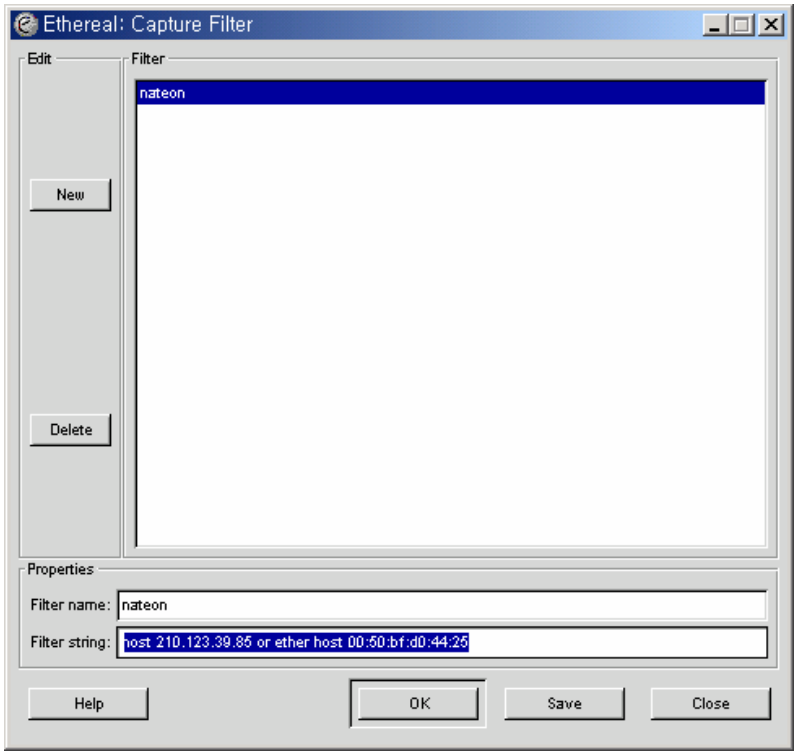
<그림 2-1>



<그림 2-2>

다음의 화면은 Ethereal 프로그램을 사용해 프로토콜을 뽑아내기 위한 Option을 설정하는

화면이다. NATEON이 TCP/IP를 통하여 프로토콜을 주고 받으므로 PC상의 TCP/IP의 패킷을 잡도록 설정을 해준다. 설정은 다음과 같이 해 준다. Caption Filter를 누르면 다음과 같은 창이 뜬다.

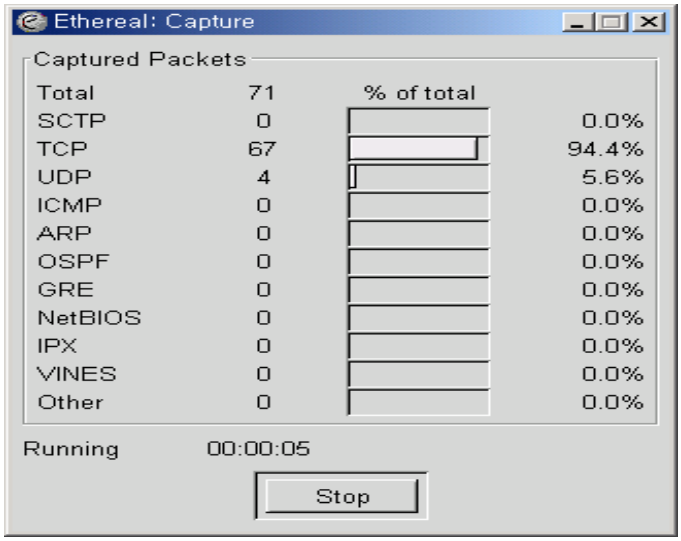


<그림 2-3>

Option을 설정하기 위해 우선 자신의 PC의 IP Address와 Physical Address를 알아야 한다. 그 정보를 통하여 Filter string을 다음과 같이 기입한다.

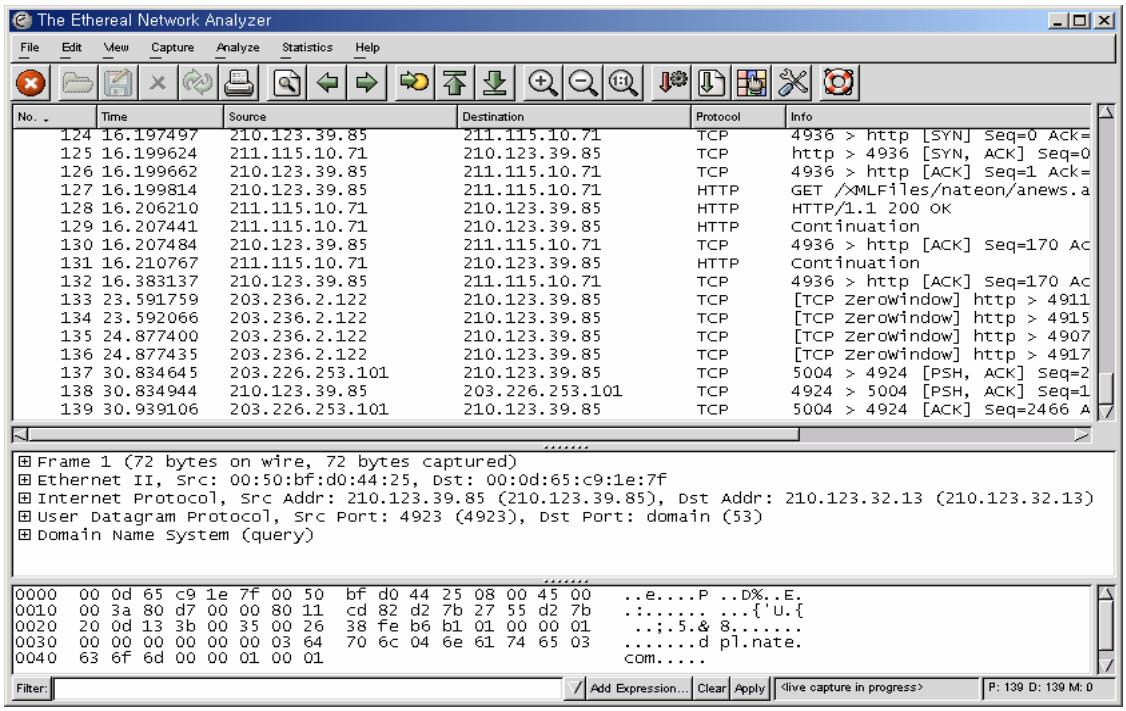
Filter string : host (IP Address) or ether host (Physical Address)

만약 패킷을 주고 받는 과정을 실시간으로 보고 싶다면 <그림 2-2>의 Display Option에서 Update list of packets in real time을 체크해주고 자동 스크롤의 지원을 원한다면 Automatic scrolling in live capture를 체크해준다.



<그림 2-4>

설정이 끝난 뒤에 ok버튼을 누르면 다음과 같이 실시간으로 PC상의 패킷을 잡는 과정을 보여주는 화면이 뜬다.

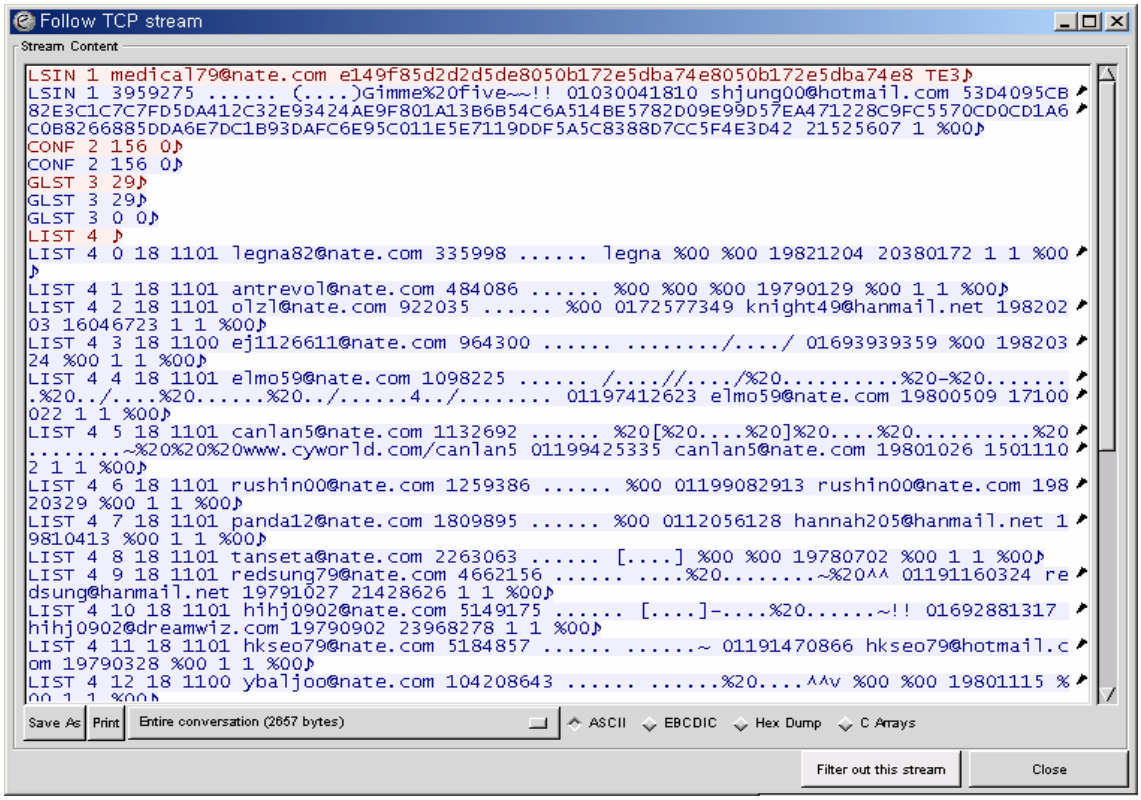


<그림 2-5>

위의 것은 Ethereal 프로그램을 통해서 잡은 전체의 패킷 리스트를 보여주는 것이다. 이중에 NATEON의 프로토콜을 주고 받는 방식인 TCP/IP 방식만을 뽑아서 보면 된다. 각 패킷의 자세한 내용을 보는 방법은 해당 패킷 위에서 오른쪽 마우스를 눌러 Flow TCP

stream을 선택하면 <그림 2-6>과 같은 자세한 내용을 볼 수 있다.

다음은 잡은 패킷 중에 하나의 내용을 보기 위해서 자세히 나타낸 것으로 서로 관련된 패킷들이 서로 묶여서 나타나고, 이것을 가지고 NATEON이 서로 주고 받는 패킷을 분석하여 프로토콜을 알아내었다. 화면상의 글자의 깨짐은 내용을 텍스트 파일로 저장하여 보면 볼 수 있다.



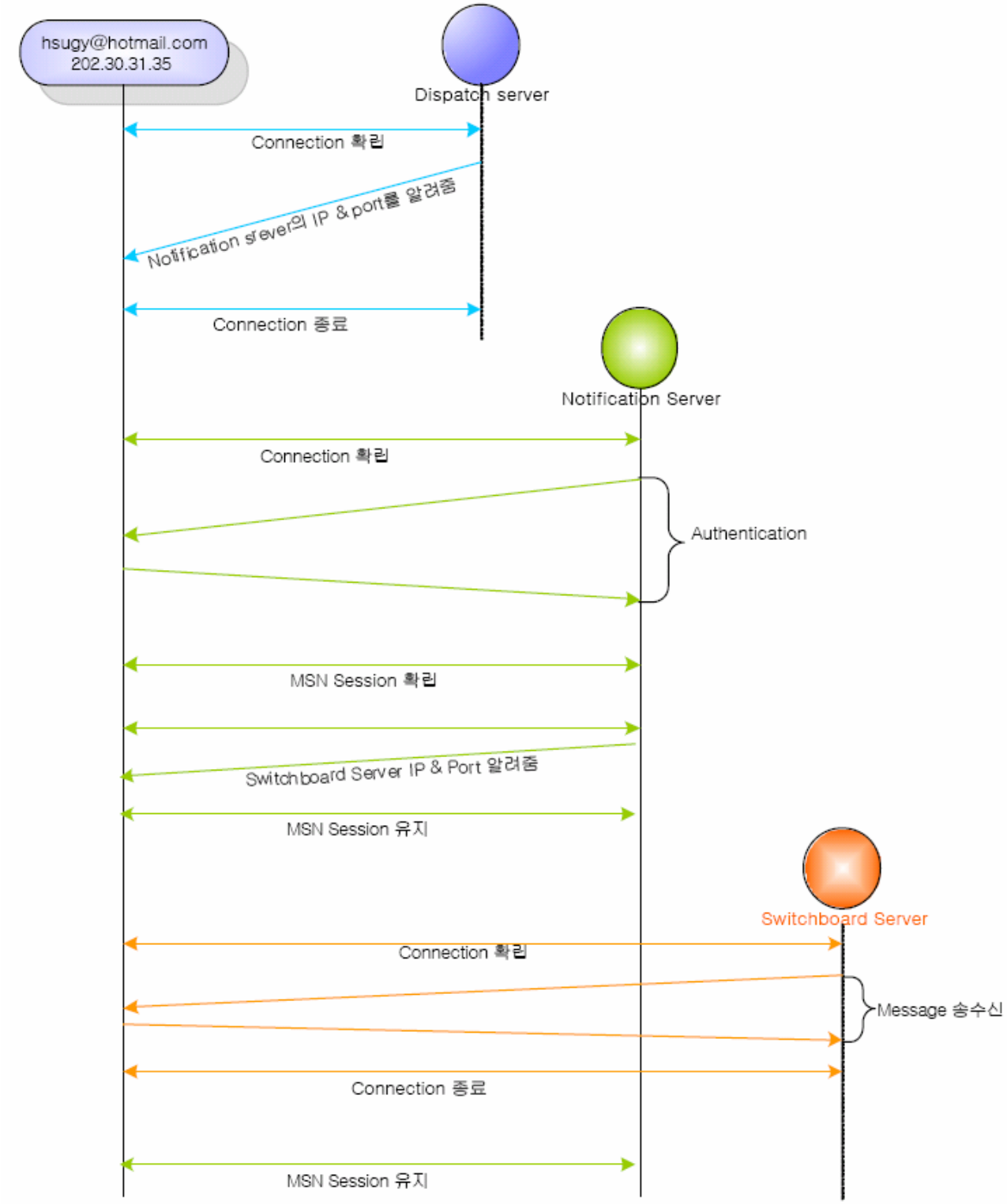
<그림 2-6>

이상과 같이 Ethereal 프로그램을 통해 서버와 클라이언트 사이에 주고 받는 패킷을 분석하여 프로토콜을 예상하여 갔다.
MSN Messenger Protocol과 NATEON Messenger Protocol에 관한 자세한 내용은 [부록1. MSN Messenger Protocol]과 [부록2. NATEON Messenger Protocol]을 참고하기 바란다.

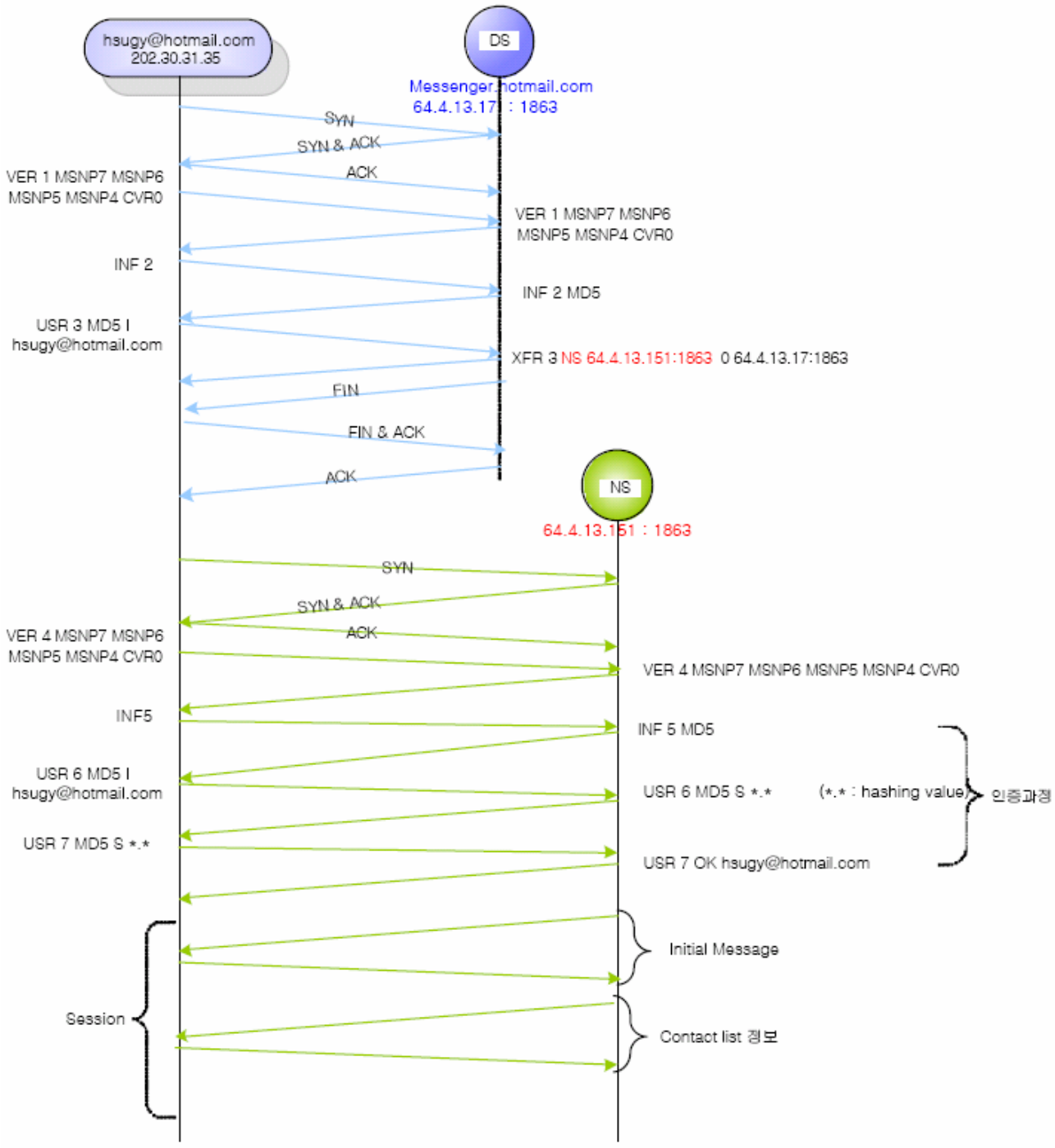
앞에서 언급했던 것처럼 메신저 자사의 프로토콜은 현재 공개되어 있지 않다. 따라서 수많은 테스트와 패킷 분석에 의해서도 완벽한 프로토콜 분석은 불가능하다. 하지만 우리 팀은 현재까지 분석한 프로토콜을 통해 MSN과 NATEON 라이브러리를 구현하였으며 그 라이브러리를 통해 PC용과 Mobile용 통합 메신저를 만드는데 성공하였다. 그럼 이제부터 패킷 분석을 통해 알아낸 MSN, NATEON 프로토콜을 Flow 그림을 통해 살펴본 후 분석한 프로토콜을 라이브러리로 모듈화한 구성도를 보도록 하겠다.

2.2 MSN Instant Messenger Protocol

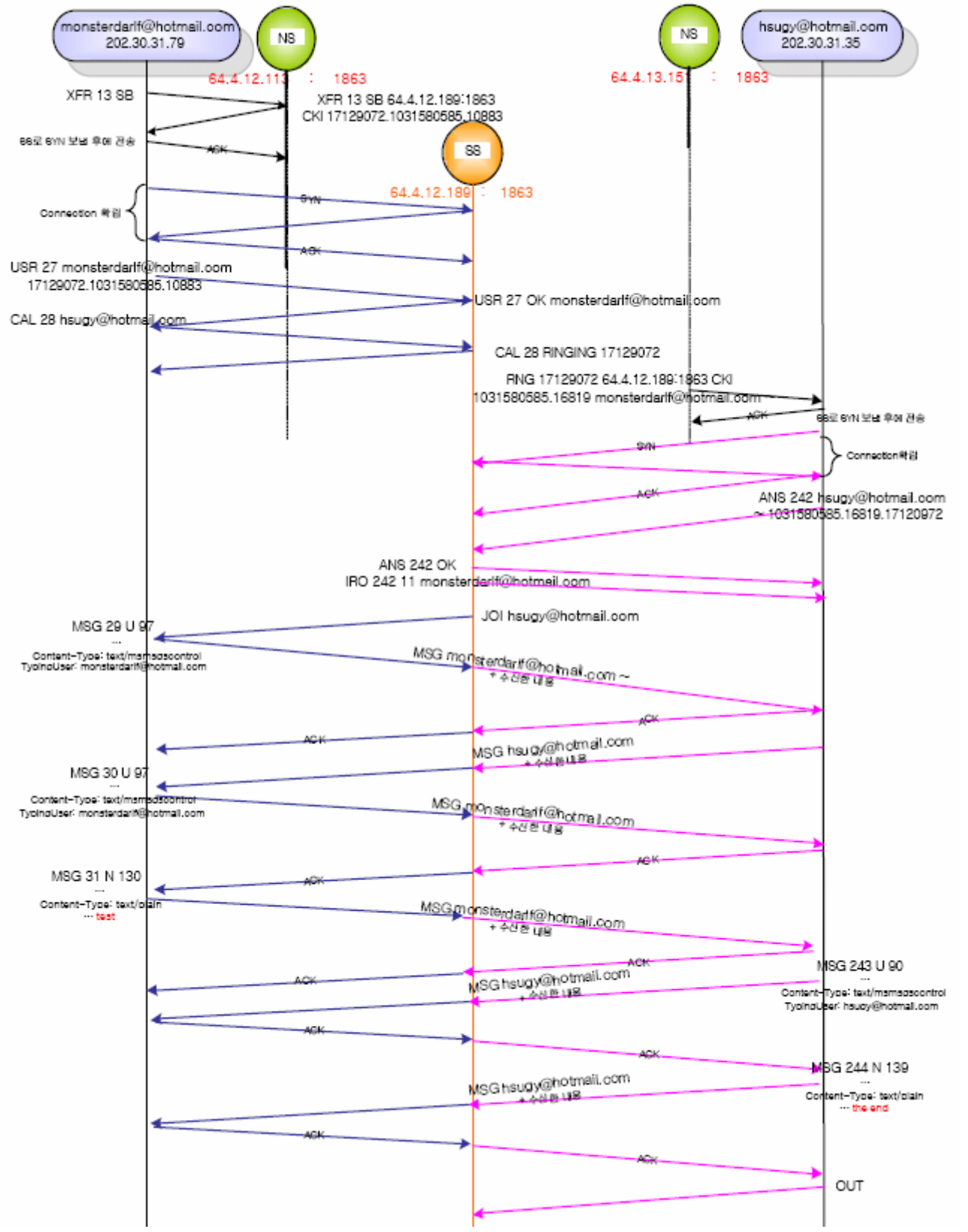
1) Overview Flow



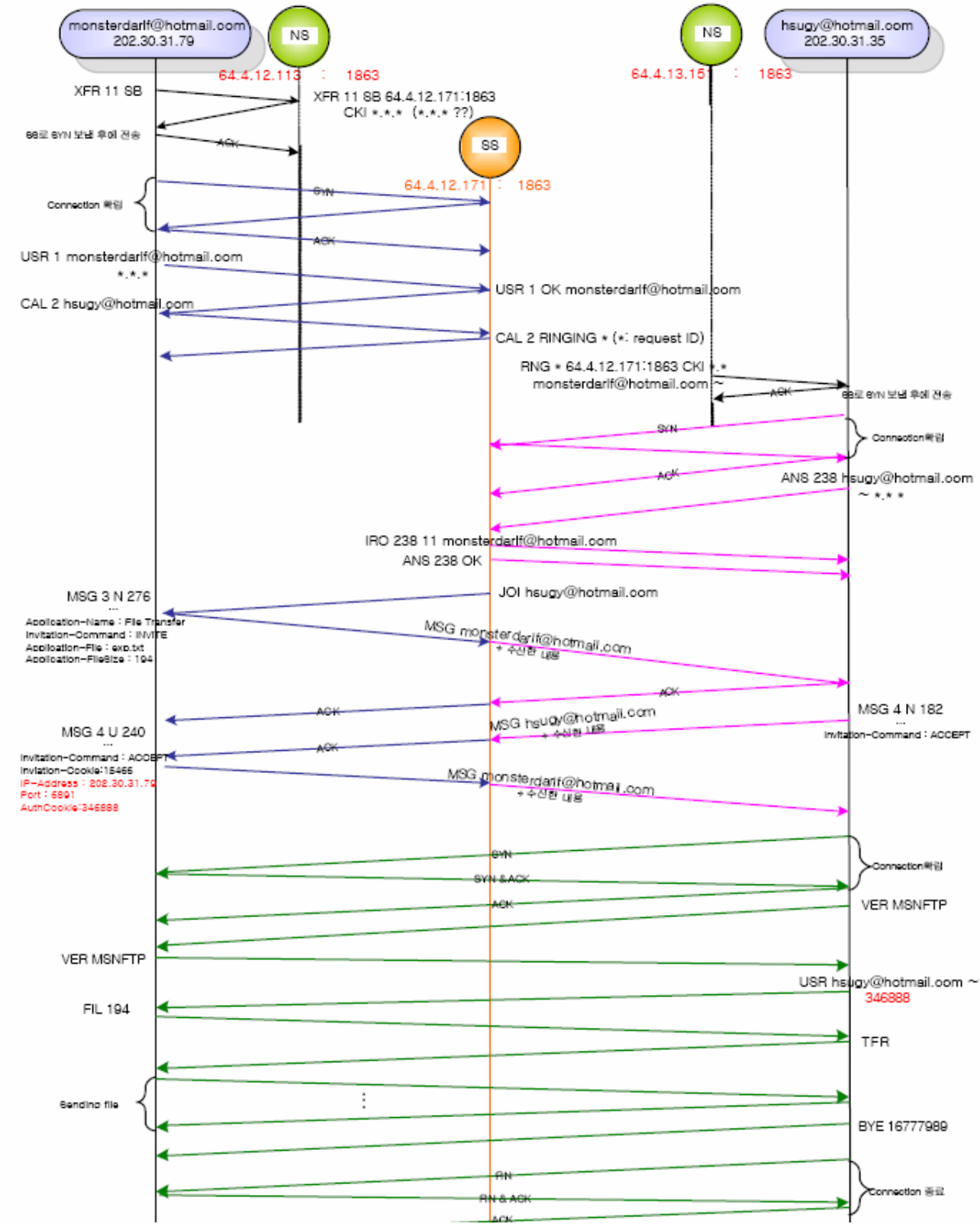
2) Connecting Flow



3) Messaging Flow

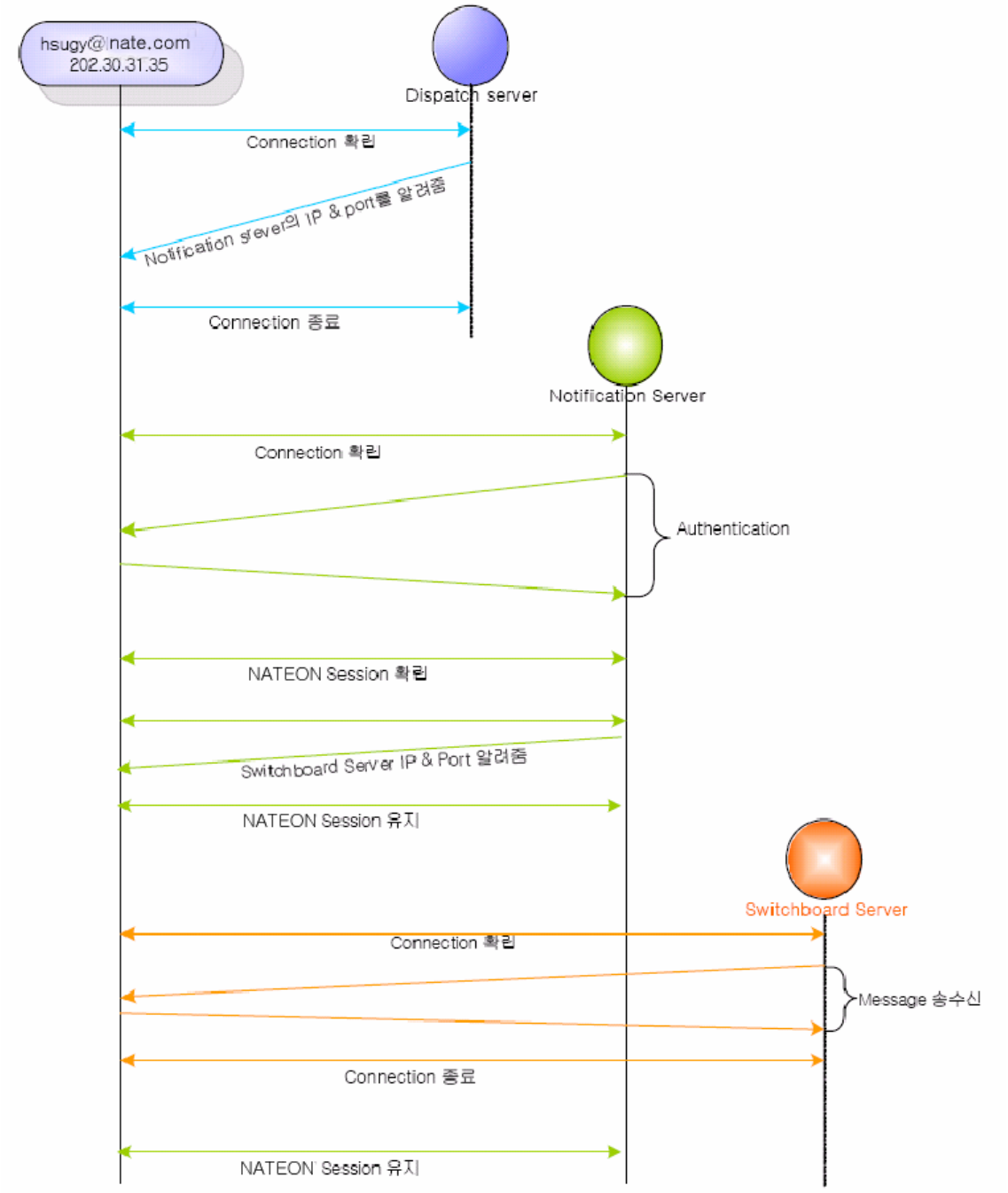


4) File Transfer Flow

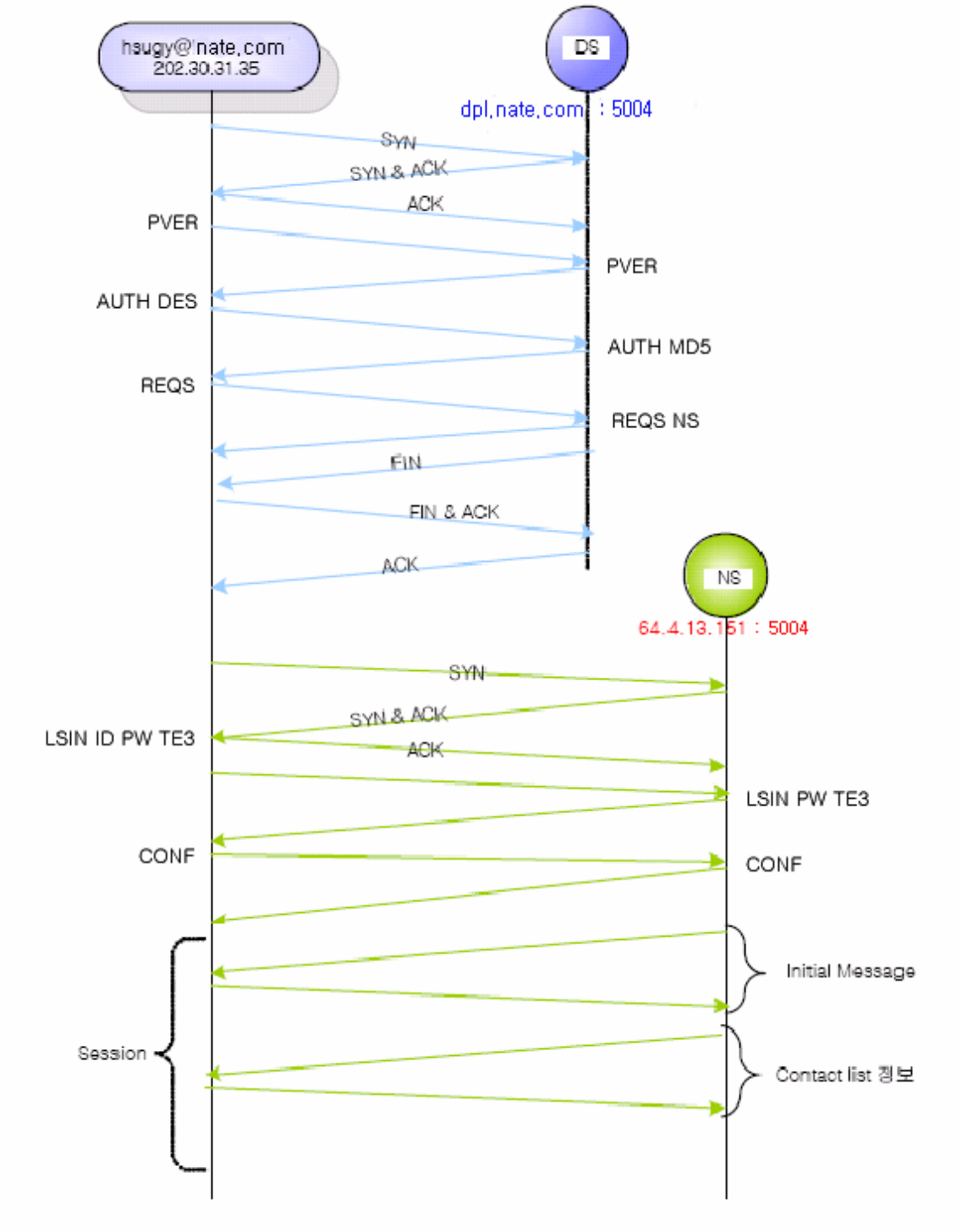


2.3 NATEON Instant Messenger Protocol

1) Overview Flow

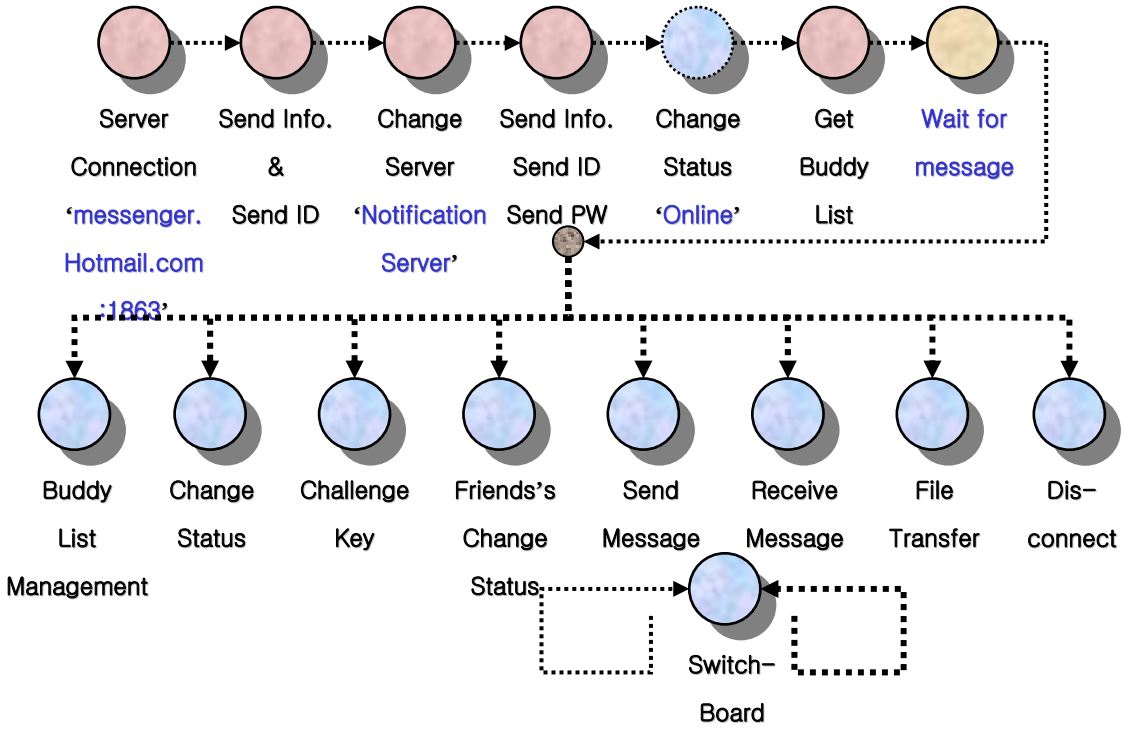


2) Connecting Flow



3. MSN 라이브러리 구현

< MSN 모듈 구성도 >

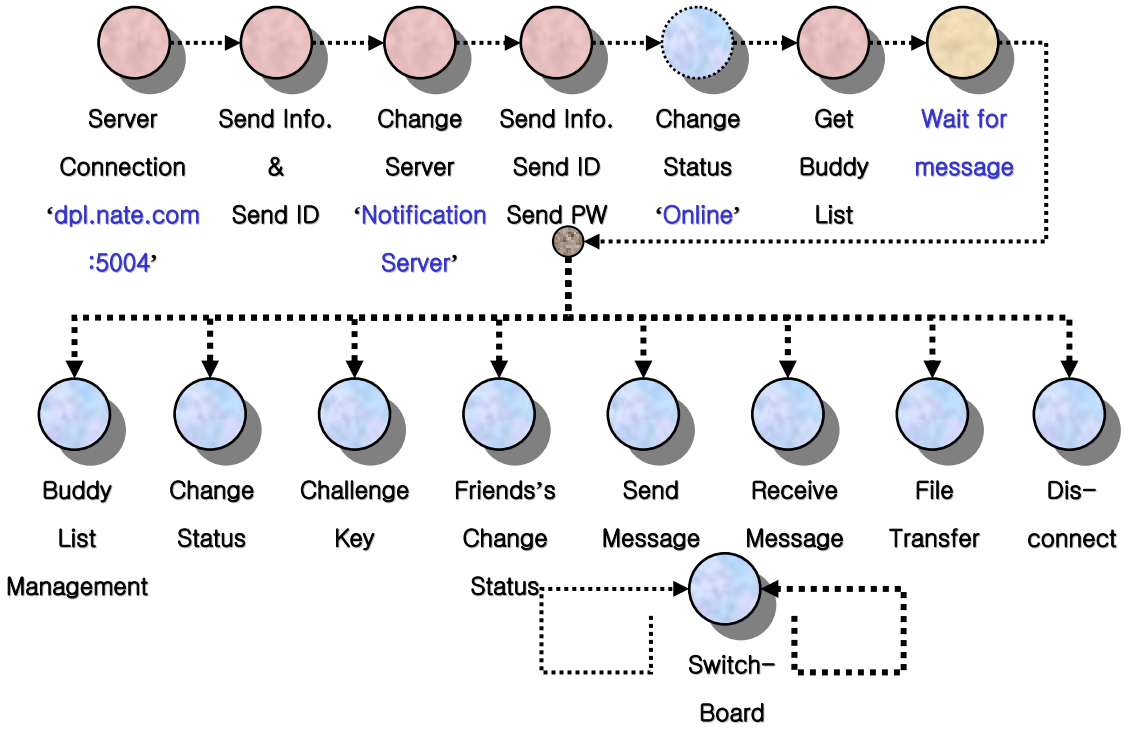


위에서 설명한 MSN 프로토콜을 라이브러리로 구현하면 위에 그림과 같은 모듈 구성도로 나타낼 수 있다.

각각의 모듈은 여러 가지 패키지, 클래스, 메소드들로 구현되었다.

4. NATEON 라이브러리 구현

< NATE ON 모듈 구성도 >



NATEON 프로토콜을 위와 같은 모듈로 구성할 수 있다.

MSN과 비슷한 모듈 구성을 위해 초기접속 후 기본서버를 NS, 대화를 위한 서버를 SS로 명명하였다.

dpl.nate.com 5004 포트를 통해 초기 접속 후 NS로 재접속하여 로그인 과정을 완료한다.

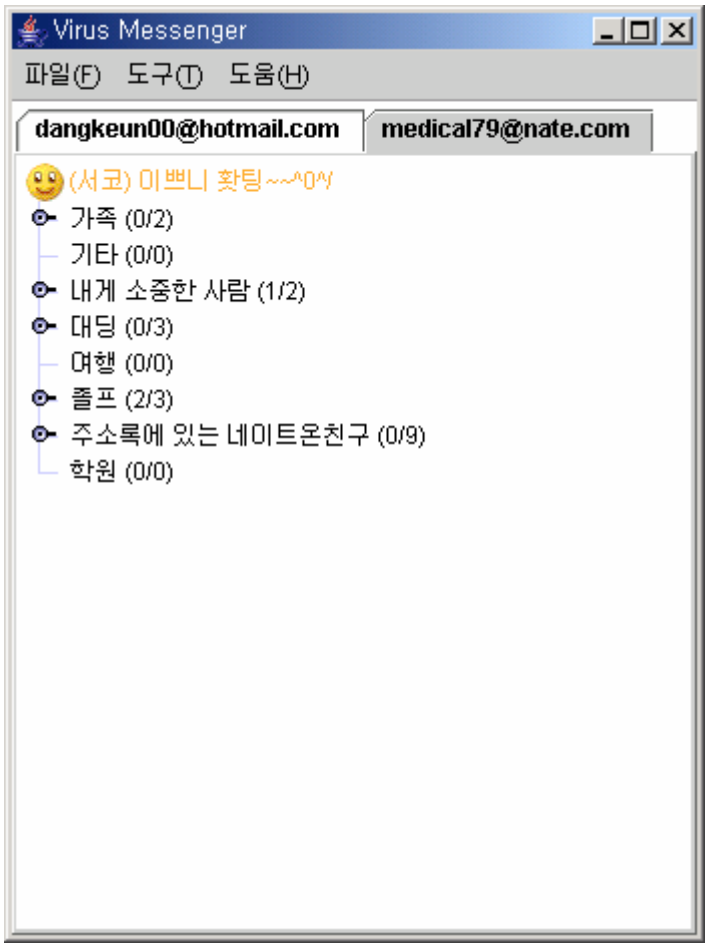
NS연결은 유지하여 기본적인 명령은 서버로부터 전달받고, 적합한 응답을 해주거나, 클라이언트 스스로 적합한 반응을 보인다.

대화를 위해서는 새로운 서버 SS를 통해 상대방과 대화를 할 수 있다.

이렇게 기본적인 과정이 MSN과 비슷하기 때문에 모듈 구성도 MSN과 거의 동일하다.

5. PC 통합 메신저 구현

MSN 프로토콜 라이브러리에 UI를 제작하여 먼저 MSN PC용 메신저를 구현하였다. 다음으로 NATEON 프로토콜 라이브러리를 제작하여 NATEON PC용 메신저를 구현하였다. 이렇게 구현된 두 개의 메신저를 하나로 통합하는 과정을 설명하겠다.



< 5-1 >

Identity 클래스를 통해 구현된 각각의 메신저 객체를 구별한다. 하나의 계정으로 로그인 할 때마다 Identity 객체가 하나씩 생성되면서 각각의 계정을 관리한다. 이러한 방법으로 MSN, NATEON에 각각 다중 로그인이 가능하며, 최종 목적인 MSN, NATEON 멀티 로그인도 가능하다.

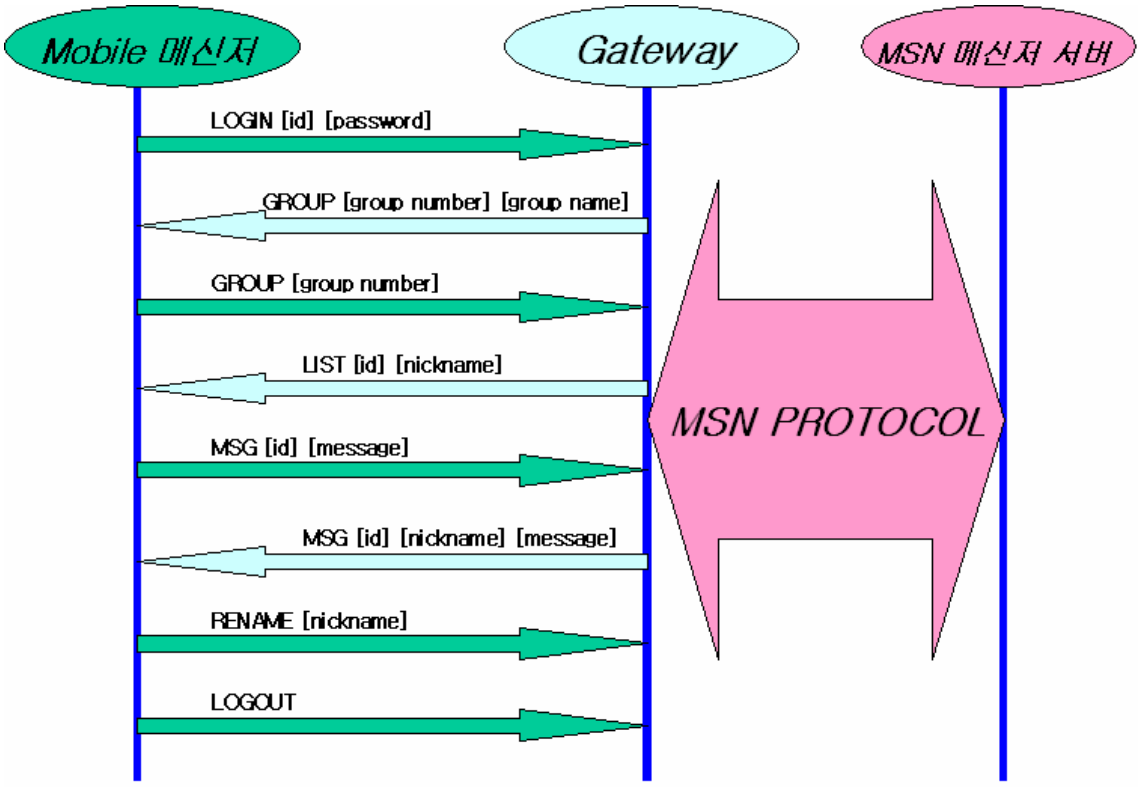
그림 <5-1> 에서는 MSN과 NATEON 계정으로 멀티 로그인 한 후의 메인 윈도우를 보여주고 있다. 로그인 된 두 명의 사용자를 탭을 통하여 구분하고 있다.

하나의 탭은 한 명의 사용자를 의미하고, 탭을 클릭하면 해당 사용자의 버디리스트를 화면에 보여준다.

탭으로 선택된 사용자는 자신만의 메뉴바를 통한 다양한 기능을 자신에게 적용할 수 있다.

6. Gateway Server

- Mobile Client 프로토콜 제작 및 라이브러리 설명



PC상에서 메신저를 구현하는 것과 달리 Mobile상에서 메신저를 구현하는 것에는 많은 제약이 따른다.

따라서 모바일 단말기 상에서 모든 일을 처리할 수 없기 때문에 MSN 서버와 모바일 메신저 사이에 중계 역할을 해주는 Gateway 서버가 필요하다. 따라서 우리 팀은 위에 그림처럼 Mobile 메신저와 Gateway 서버 사이에 주고 받는 프로토콜을 정의 및 구현하여 Mobile 상에서도 MSN 메신저를 구현하는데 성공하였다. Gateway Server와 Mobile Client 프로토콜에 대한 자세한 내용은 [부록3. Gateway Server-Mobile Client 프로토콜]을 참고하기 바란다.

그럼, 이제부터 Gateway Server와 MSN Mobile용 메신저에 대해서 자세히 살펴보도록 하겠다.

7. Gateway Server 구현

7.1 Gateway Server의 필요성

메신저에서는 대화창을 하나 열 때마다 하나의 포트가 새로 열린다. 따라서 메신저 서버와 통신을 하기 위해서는 한 개 이상의 소켓을 사용해야 한다. 하지만 모바일에서는 한 개의 소켓만을 사용할 수 있기 때문에 게이트웨이를 사용함으로써 한 개 이상의 소켓을 사용하는 것과 동일한 효과를 낼 수 있다. 또 유선 메신저와 메신저 서버는 ASCII기반의 프로토콜 메시지를 이용하여 통신을 하고 있다. 모바일 메신저 역시 메신저 서버를 이용하기 위해서는 같은 프로토콜을 사용해야 한다. 하지만 모바일로 전송되는 모든 메시지들은 패킷 단위로 사용자들에게 요금을 부과하게 된다. 따라서 메신저 서버에서 전송하는 프로토콜 메시지를 모두 모바일로 전송하는 것은 사용자에게 패킷 비용의 부담을 안기게 된다. 또 모바일에서 모든 프로토콜을 처리하는 것은 어려움이 따른다. 따라서 모바일 메신저와 메신저 서버 사이에 게이트웨이를 사용함으로써 패킷 비용의 부담을 줄이고, 유선 메신저를 사용하는 것과 동일하게 프로토콜을 처리 할 수 있다.

7.2 Gateway Server의 기본 사항

게이트웨이는 모바일 클라이언트와 메신저 서버 사이의 중개자 역할을 담당하고 있다. 따라서 게이트웨이는 메신저 서버와 통신이 가능한 것은 물론이고 모바일 클라이언트와의 통신도 가능해야 한다. 메신저 서버와는 기존의 사용되는 프로토콜을 사용하고, 모바일 클라이언트와 통신 하기 위해서는 자체적인 프로토콜을 제작하여 사용해야 한다.
또 게이트웨이는 메신저 서버에서 보낸 메시지 중 클라이언트가 저장하기 어려운 정보를 저장하고 있다. 그리고 모바일 클라이언트가 필요한 정보를 요청할 경우 가지고 있던 정보를 모바일 클라이언트로 전달해주는 역할을 한다.

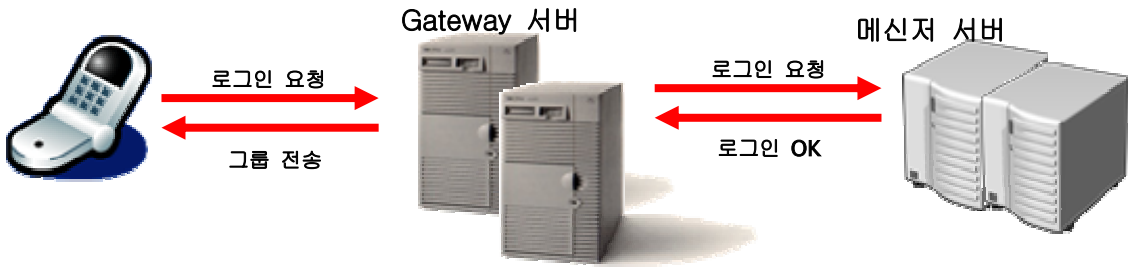
7.3 모바일 클라이언트와 게이트웨이 간의 프로토콜 제작

모바일 클라이언트가 게이트웨이와 통신하기 위한 프로토콜은 다음과 같다. 모바일이라는 상황을 고려하여 패킷비용을 절감하기 위해서 헤더는 최소한의 문자를 사용하도록 하였다.

모바일 클라이언트에서 게이트웨이로 보내는 프로토콜		
로그인	i (or ii) [ID] [PASSWORD]	
로그아웃	o(or oo)	
대화명 변경	r (or rr) [NICKNAME]	
해당 그룹의 리스트	g(or gg) [GROUP NUMBER]	그룹에 해당하는 숫자를 기억하고 있다가 그룹의 목록을 요청
메시지 전송	m(or mm) [ID] [MESSAGE]	
게이트웨이에서 모바일 클라이언트로 보내는 프로토콜		
그룹 리스트 전송	g(or gg) [GROUP NUMBER] [GROUP NAME]	그룹 번호와 그룹 이름을 매칭시켜 보냄
친구 목록 전송	l (or ii) [ID] [NICKNAME]	로그인 되어 있는 사람의 리스트만을 전송
메시지 전송	m(or mm) [ID] [NICKNAME] [MESSAGE]	닉네임은 공백을 제거하여 8글자만 전송

7.4 Gateway Server의 동작

1) 로그인 과정



클라이언트는 i 명령어를 사용해 게이트웨이에게 로그인을 요청한다. 로그인을 요청 받은 게이트웨이는 메신저 서버로 로그인을 위한 명령어를 전송하기 시작한다. 메신저 서버에서의 로그인 과정은 사용자 인증 과정을 거쳐 사용자가 접속해야 할 서버를 알려주는 과정으로 이루어져 있다. 때문에 메신저 서버에서 응답이 돌아올 경우에만 다음 명령어를 보내는 순차적인 과정을 보인다.

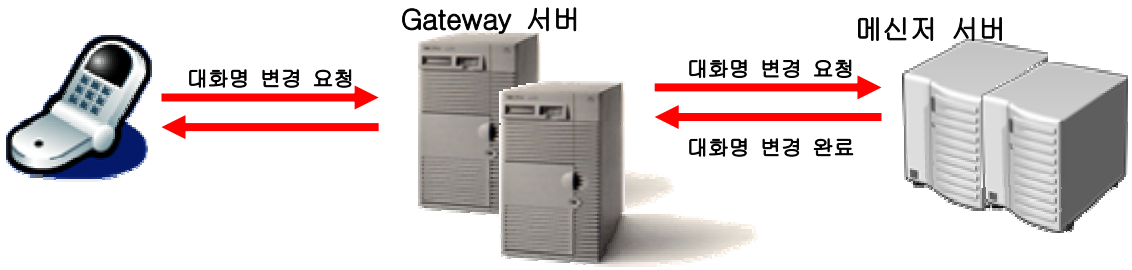
우선 게이트웨이는 VER 명령어를 통해 메신저의 프로토콜 버전을 메신저 서버로 전달한다. 메신저 서버는 마찬가지로 VER 명령어로 게이트웨이에 응답을 보낸다. 다음 게이트웨이는 CVR 명령어를 통해 클라이언트와 사용중인 운영체제와 그 밖의 다른 정보를 메신

저 서버로 전송한다. 메신저 서버는 마찬가지로 CVR을 통해 응답을 한다. 응답을 받은 게이트웨이는 첫번째 USR 명령어를 통해 클라이언트와 서버간의 인증을 요청한다. 요청을 받은 서버는 XFR 명령어를 전송해 로그인을 요청한 사용자의 정보가 있는 서버로 다시 연결하도록 새로운 아이피를 전송한다. 게이트웨이는 메신저 서버의 응답을 받은 뒤 다시 VER과 CVR 명령어를 전달한 위의 과정을 반복한다. 그 다음 다시 USR 명령어를 전달해 로그인 요청 사용자의 패스워드가 맞는지 인증을 하는 과정을 거친다. 사용자 인증 과정이 끝나면 서버는 USR 명령어와 OK를 전송해 주고 인증을 완료한다.

메신저 서버와의 인증이 완료되면 게이트웨이는 메신저 서버에 SYN 명령어를 전송해 게이트웨이가 저장하고 있는 친구목록의 동기화를 요청한다. 메신저 서버는 친구 목록의 버전이 서버의 버전과 같지 않으면 그룹 목록과 업데이트된 친구 목록을 LSG와 LST 명령어를 통해 게이트웨이에 전송한다. 게이트웨이는 전달 받은 그룹 목록과 친구 목록을 저장한다. 게이트웨이는 CHG 명령어를 전달해 클라이언트의 상태를 설정한다. 메신저 서버에서 CHG 명령어를 받으면 로그인이 모두 완료된다.

게이트웨이는 위와 같은 메신저 서버와의 로그인 과정을 마친 뒤 클라이언트에 저장된 그룹 목록을 g 명령어를 통해 그룹넘버와 함께 전달하면서, 최종적으로 모바일 클라이언트의 로그인 과정도 끝난다.

2) 대화명 변경

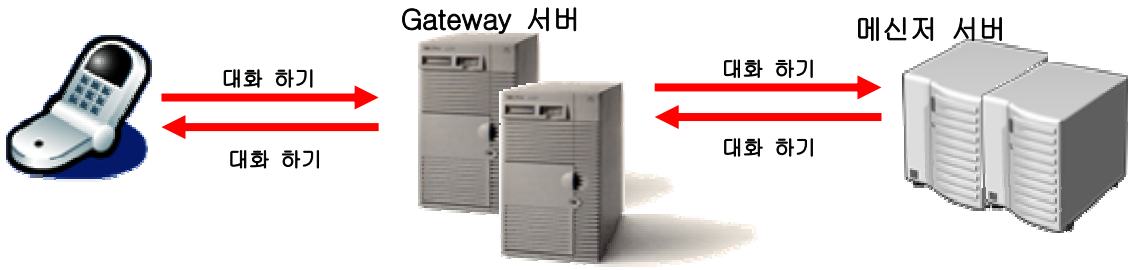


클라이언트에서 대화명을 변경하기 위해서는 r 명령어를 게이트웨이로 전달한다. 대화명 변경을 요청 받은 게이트웨이는 다시 메신저 서버로 REA 명령어를 전달한다. 명령어를 전달 받은 메신저 서버는 대화명을 변경한 뒤 게이트웨이로 REA를 전달해 대화명이 변경됐음을 알려준다. 이때 게이트웨이는 클라이언트에 변경이 완료 된 것을 전달하지 않는데 변경을 시도하면서 클라이언트의 대화명을 변경하기 때문이다.

3) 친구 목록 요청

로그인이 완료 된 뒤에 게이트웨이에서 클라이언트로 전달 된 것은 그룹 목록만 전달되고, 친구 목록은 전달되지 않는다. 이것은 친구목록을 필요할 경우에만 전달함으로써 패킷 비용을 절감하기 위한 방법이다. 따라서 클라이언트는 친구에게 대화를 요청하기 위해서는 먼저 친구가 속한 그룹의 친구 리스트를 요청하는 과정이 필요하다. 클라이언트는 떠 있는 그룹 목록 중에서 대화를 요청할 친구가 있는 그룹을 선택함으로써 게이트웨이에 친구 목록을 요청할 수 있다. g 명령어를 통해 그룹의 친구 목록 요청을 전달 받은 게이트웨이는 메신저 서버와의 통신 없이 로그인시 저장한 친구 목록을 클라이언트에게 전달한다.

4) 대화 하기



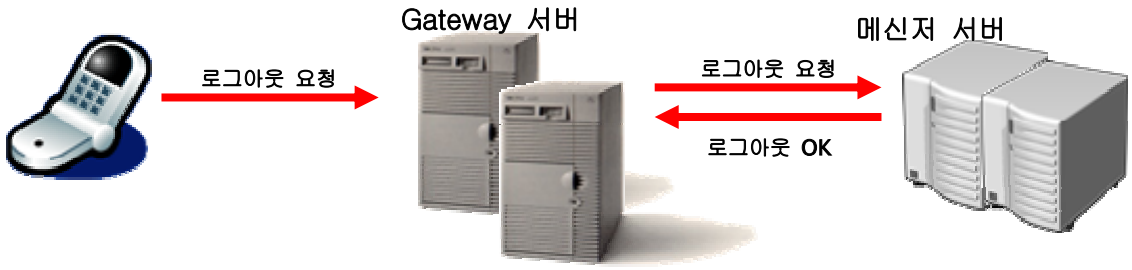
대화를 하는 과정은 두 가지로 나누어져 있다. 대화를 요청해 대화요청을 받은 사용자와 세션연결을 하는 것과 세션에 연결된 사용자에게 메시지를 보내는 것이다. 이 두 과정은 나누어져 있지만 대화를 보내면 자동으로 대화요청을 받은 사용자와 세션연결이 되고 메시지를 보내게 된다. 따라서 클라이언트는 m 메시지를 게이트웨이에 전달하는 것만으로 세션연결과 메시지 전달을 동시에 할 수 있다. m 메시지를 통해 대화를 전달 받은 게이트웨이는 먼저 세션이 연결됐는지 확인을 한다. 세션이 연결되어 있지 않은 사용자에게 대화를 요청했을 경우 먼저 메신저 서버에 XFR 명령어를 통해 세션 연결을 요청한다. 메신저 서버는 XFR 명령어로 응답을 한다. 응답을 받은 게이트웨이는 다시 USR 명령어로 세션을 연결한 뒤 인증을 요청한다. 메신저 서버는 인증을 끝낸 뒤 USR 명령어를 게이트웨이에 보내면서 인증을 마친다. 세션연결이 끝나면 게이트웨이는 CAL 명령어를 메신저

서버로 전달해 채팅서버에 친구추가를 요청한다. 메신저 서버는 CAL 명령어를 보내 친구 추가를 요청 받았음을 확인한다. 메신저 서버는 JOI 명령어를 게이트웨이에 보내 초청된 친구가 들어왔음을 알려준다. 게이트웨이는 초청한 친구가 들어왔다는 명령을 받은 뒤에 전송하려는 메시지를 MSG 명령어를 통해 메신저 서버에 전달한다.

만약 클라이언트가 대화를 요청 했을 때 이미 게이트웨이와 메신저 서버 사이의 세션이 연결 되어 있으면, 게이트웨이는 세션 연결 요청 과정을 거치지 않고, MSG 명령어를 이용해 바로 메시지를 전달한다.

이와 반대로 상대방이 대화를 요청하는 경우 메신저 서버는 게이트웨이에 RNG 명령어를 전달해 다른 사용자로부터 대화요청을 받았음을 알려준다. 이 요청에 대해 게이트웨이는 ANS 명령어를 메신저 서버에 전달해 채팅서버에 접속할 것을 알려준다. 대화 요청에 응한다는 메시지를 받으면 메신저 서버는 게이트웨이에 IRO 명령어를 전달해 채팅창에 접속해 있는 사용자들의 정보를 전달한다. 그리고 MSG 명령어를 통해 게이트웨이에 메시지를 전달한다. 메시지를 받은 게이트웨이는 m 명령어를 이용해 클라이언트에 전달 받은 메시지를 전달한다.

5) 로그아웃



모든 작업을 마친 클라이언트는 로그아웃을 하기 위해 o 명령어를 게이트웨이에 전달한다. 로그아웃을 요청 받은 게이트웨이는 OUT 명령어를 메신저 서버에 전달하면 로그아웃 되면서 메신저 서버와의 접속이 끊어진다. 로그아웃이 완료되면 게이트웨이는 클라이언트와의 접속도 종료시킨다.

8. Mobile 통합 메신저 구현

8.1 UI 디자인 및 구현

에뮬레이터를 처음 실행시키면 다음 그림과 같은 초기화면이 뜬다. 커맨드바에는 MSN 메신저에 접속하는 메뉴와 네이트온에 접속하는 메뉴가 있다.

초기화면에서 msn 메뉴를 선택하면 로그인 창으로 전환된다. 메일주소와 패스워드를 집어넣고 확인버튼을 누르면 로그인 중인 창으로 전환된다. 이때 모바일 메신저 클라이언트 프로그램에서는 *LOGIN [redsung@msn.com] [password]* 라는 프로토콜을 게이트웨이에게 전송하여 로그인을 요청하게 된다. 로그인 중인 창은 게이트웨이로부터 친구목록을 받아올 때까지 보여진다.

MSN 메신저 서버에 로그인이 완료되면 MSN 메신저 서버에서 친구목록을 보내주고 그 친구목록을 게이트웨이가 저장한다. 게이트웨이는 우선 그룹명을 모바일 메신저 클라이언트에게 보낸다. 이 때 게이트웨이는 *GROUP [1] [개인], GROUP [2] [후배], GROUP [3] [친구]* …… 을 연속하여 모바일 메신저 클라이언트에게 보낸다. MSN 메신저에는 핸드폰으로 로그인 했음을 알려준다.

그룹을 하나 선택하면 모바일 메신저 클라이언트는 게이트웨이에게 선택된 그룹에 속하는 친구 리스트를 요청하기 위해 *GROUP [1]* 을 보낸다. 여기에서 1은 선택한 그룹의 그룹 넘버이다. 게이트웨이는 *LIST [aegisfight@hotmail.com] [희진]*과 같은 프로토콜을 계속 보내어 온라인 상태의 친구 리스트를 모바일 메신저 클라이언트에게 전송한다. 커맨드바에는 그룹리스트로 다시 돌아가는 메뉴가 있다.

대화하고 싶은 친구를 선택하면 다음과 같은 대화창으로 전환되어 서로 메시지를 주고 받을 수 있다. 이때 모바일 메신저 클라이언트에서는 *MSG [redsung@msn.com] [희진아 안녕~]* 프로토콜을 게이트웨이로 전송하고 게이트웨이에서는 *MSG [aegisfight@hotmail.com] [희진] [성민이도 안녕~]* 프로토콜을 모바일 메신저 클라이언트에게 전송한다. 커맨드 바에 대화창 닫기를 선택하면 친구목록을 보여주는 창으로 전환된다.

로그인이 완료되고 대기하고 있는 상태에서 상대방으로부터 대화요청이 왔을 경우 메시지를 보낸 상대방과 바로 대화를 할 수 있도록 대화창으로 화면이 전화되게 된다.

그리고 이때 게이트웨이에서는 *MSG [aegisfight@hotmail.com] [희진] [안녕~]* 이라는 프로토콜을 모바일 메신저 클라이언트에게 전송하게 된다.

커맨드 바에서 메뉴를 선택하면 다음과 같은 메뉴화면이 나온다. 친구목록을 선택하면 친구목록을 보여주는 창으로 전환되고, 대화명변경을 선택하면 대화명을 변경하는 창으로 전환된다. 밑에 보는 것처럼 메뉴는 MSN이 로그인 되어 있는지 NATEON이 로그인 되어 있는지 둘 다 로그인 되어 있는지에 따라 유동적으로 목록이 변하게 된다.

메뉴창에서 대화명 변경을 선택하면 다음과 같이 대화명 변경창으로 전환된다. 변경할 대화명을 입력하고 확인버튼을 누르면 대화명이 변경되어 MSN 메신저에 변경된 대화명이 보여진다. 이때 모바일 메신저 클라이언트에서는 *RENAME [홍길동]* 프로토콜을 게이트웨이로 전송하여 대화명 변경 요청을 하게 된다.

9. 결과 및 진행 방향

우리 팀은 최종 목표한 대로 국내에서 가장 많이 사용하고 있는 메신저인 MSN과 NATEON을 통합하여 유무선 연동이 가능한 유무선 통합 메신저를 구현하였다.

PC상에서는 MSN과 NATEON 멀티 로그인 뿐만 아니라 MSN 다중 로그인과 NATEON 다중 로그인도 지원하여 메신저를 사용하는 사용자의 편의를 최대한 보장하려고 노력하였다.

또한 뛰어난 휴대성으로 누구나 가지고 다니는 모바일 상에서도 MSN과 NATEON를 통합한 메신저를 구현함으로써 유선 사용자뿐 아니라 무선 사용자의 편의성도 보장하려고 노력하였다.

하지만 아직까지 보완해야 할 점과 추가해야 할 기능들이 많이 있다.

지금은 국내에서 가장 많이 사용되는 MSN과 NATEON 메신저만 통합한 상태이다. 하지만 전세계적으로 봤을 때 가장 많이 사용되고 있는 AIM과 ICQ와 같은 국외 IM(Instance Messenger)도 통합이 된다면 더 편리한 통합 메신저가 될 수 있을 것이다.

또한 P2P 파일 전송이나 게임, 화상, 음성채팅과 같은 멀티미디어 요소를 가미하고 개인 블로그의 연동도 추가한다면 더 많은 사람들의 기호를 충족시켜 줄 수 있을 것이다.

10. 참고 문헌 및 싸이트

1) [도서] 이클립스 기반 프로젝트 필수 유틸리티: CVS, Ant, JUnit
민진우, 이인선 공저 | 한빛미디어 | 2004년 01월

2) [도서] Java 세상을 덮치는 Eclipse
윤성준, 조상민, 송정일 지음 | Insight | 2003년 05월

3) [도서] 위피 모바일 프로그래밍
박수원, 안은석, 이경철 | 한빛소프트디지털캠퍼스㈜ | 2003년 11월

4) [도서] 프로그래머를 위한 JAVA2
최재영, 최종명, 유재우 공저 | 홍릉과학출판사 | 2003년 2월

5) [도서] 자바 2 JDK 1.4 - 클릭하세요
정의현,김성진 공저 | 대림 | 2002년 08월

6) [웹페이지] http://jabook.org - 소설 같은 자바

7) [웹페이지] http://www.hypothetic.org/docs/msn/index.php
- MSN Messenger Protocol

8) [웹페이지] http://mobilejava.co.kr - Mobile JAVA Developer community

9) [웹페이지] http://789.bz - 모바일 통합 메신저 (MSN, AOL, ICQ)

10) [웹페이지] http://itpia.net - Mobile & PDA Programming

11) [웹페이지] http://ethereal.com - A Network Protocol Analyzer

12) [웹페이지] http://eclipse.org - Eclipse.org Main Page

13) [웹페이지] http://www.cvsnt.org - The CVSNT Wiki

14) [웹페이지] http:// developer.wipi.or.kr - WIPI 개발자 커뮤니티

< 부록 1 - MSN Messenger Protocol >

목 차

1. MSN Messenger Protocol

MSN Messenger protocol은 client 와 server 사이의 command 집합으로 구성되어 있다.
다음에서는 Notification Server (NS) 와 Switchboard (SB) 에서 사용되는 command를 각 상황 별로 설명한다.
먼저 command 의 기본 구성을 살펴 본 후, MSN protocol 에서 사용되고 있는 command 이름과 세부 설명, 예제를 통해 전체 구성을 알아본다.

2. command 기본 구성

command 는 3개의 문자, 다양한 파라미터, ₩r₩n으로 구성된다.
보통 3개의 문자 이후에 따라오는 파라미터의 시작은 TrID를 나타내는 경우가 대부분이다.
client가 server에게 보내는 TrID 와 그에 대한 응답으로 server가 client에게 보내는 TrID 는 동일하다. TrID는 1부터 시작해서 1씩 증가하며 나타난다.
그러나, 모든 command가 파라미터의 시작으로 TrID를 가지는 것은 아니며, TrID 가 없는 command도 볼 수 있다.
각각의 파라미터 사이에는 공백을 하나씩 두며, 마지막 파라미터는 공백 없이 바로 ₩r₩n 이 위치한다.
아래 세부 설명에서 >>> 기호는 client 가 server 에게 보내는 명령이고, <<< 기호는 server 가 client 에게 보내는 명령을 의미한다.
3개의 문자 이후에는 파라미터를 나타내는데, [] 기호를 통해 표현한다.
마지막으로는 실제 client 와 server 사이에서 주고받는 예제를 통해 어떻게 사용되는지를 나타낸다.

3. Notification Server

3.1 Authentication

VER

MSN Messenger protocol version 을 나타낸다.
만약 ver8 이라면 "MSNP8" 과 "CVR0" 지원 해야 한다.

>>> VER [1][2]₩r₩n
<<< VER [1][2]₩r₩n
[1] TrID
[2] 지원하는 프로토콜 버전리스트

(example)

>>> VER 1 MSNP8 CVR0₩r₩n

<<< VER 1 MSNP8 CVR0₩r₩n

CVR

client 와 OS 에 관한 버전정보를 알려준다.

>>> CVR [1][2][3][4][5][6][7][8][9]₩r₩n

[1] TrID

[2] locale ID 라 하며 해당언어마다 특정 번호가 주어져 있다. 예를 들어 Korean 은 1042, English – U.S 는 1033 이다. 이 번호를 16 진수로 나타낸 수가 들어간다.

[3] OS type ("win" 일 경우 windows 를 의미)

[4] OS version ("4.10" 은 windows 98 을 의미)

[5] 해당 컴퓨터의 archithcture

[6] client name

[7] client messenger version

[8] always MSNSGR (이 공식적인 MSN Messenger client 를 나타낸다.)

[8] client passport

<<< CVR [1][2][3][4][5][6]₩r₩n

[1] TrID

[2] 추천하는 Messenger version

[3] [2]의 확인

[4] 사용할 수 있는 최소 version

[5] 추천하는 version 을 다운로드 받을 수 있는 URL

[6] 더 많은 정보를 얻을 수 있는 URL

(example)

>>> CVR 2 0x0409 win 4.10 i386 MSNMSGR 5.0.0544 MSMSGS example@passport.com₩r₩n

<<< CVR 2 6.0.0602 6.0.0602 1.0.0000 http://download.microsoft.com/download/8/a/4/8a42bcae-f533-4468-b871-d2bc8dd32e9e/SETUP9x.EXE http://messenger.msn.com₩r₩n

USR (Initial)

Client 와 server 간의 인증

>>> USR [1][2][3]₩r₩n

[1] TrID
[2] always TWN : authentication system
[3] always I : initiating authentication
[4] account name
(example)
>>> USR 3 TWN I example@passport.com₩r₩n

XFR

USR(initial) 에 대한 서버 응답이다.
새로운 서버 연결 (NS 로의 연결)을 의미한다.
messenger.hotmail.com 로 연결했을 때는 항상 XFR 보낸다.
gateway.messenger.hotmail.com 로 연결했을 때는 XFR 을 보내지 않는다.
XFR 이후, 이전의 서버는 연결을 닫고, 특정 NS 에 접속한 후, 처음부터 다시 로그인 과정을 시작한다.

<<< XFR [1][2][3][4][5]₩r₩n
[1] TrID
[2] NS
[3] NS ip:port
[4] 0
[5] 현재의 ip:port

(example)
>>> USR 3 TWN I example@passport.com₩r₩n
<<< XFR 3 NS 207.46.106.118:1863 0 207.46.104.20:1863₩r₩n
// 특정 NS 로 다시 연결, 초기 연결 과정 반복
>>> VER 4 MSNP8 CVR0₩r₩n
<<< VER 4 MSNP8 CVR0₩r₩n
>>> CVR 5 0x0409 win 4.10 i386 MSNMSGR 5.0.0544 MSMSGS
example@passport.com₩r₩n
<<< CVR 5 6.0.0602 6.0.0602 1.0.0000
http://download.microsoft.com/download/8/a/4/8a42bcae-f533-4468-b871-d2bc8dd32e9e/SETUP9x.EXE http://messenger.msn.com₩r₩n
// USR 명령까지 동일
>>> USR 6 TWN I example@passport.com₩r₩n

USR (subsequent USR response)

USR 에 대한 응답이다.

<<< USR [1][2][3][4]₩r₩n

[1] TrID

[2] TWN

[3] S : subsequent

[4] long string : TWN authentication

이 string 을 통해 "ticket" 을 얻게 되고, 올바를 ticket 을 서버에 보내야 인증이 완료된다.

(example)

>>> USR 6 TWN I example@passport.com₩r₩n

<<< USR 6 TWN S

lc=1033,id=507,tw=40,fs=1,ru=http%3A%2F%2Fmessenger%2Emsn%2Ecom,ct=1062764229,kpp=1,kv=5,ver=2.1.0173.1,tpf=43f8a4c8ed940c04e3740be46c4d1619₩r₩n

USR (final)

위에서 얻은 string 으로 ticket 을 뽑아내고, 이 ticket 을 다시 서버에 보낸다.

서버는 올바른 ticket 을 받았을 때 OK 응답을 보낸다.

>>> USR [1][2][3][4]₩r₩n

[1] TrID

[2] TWN

[3] S

[4] ticket

<<< USR [1][2][3][4][5][6]₩r₩n

[1] TrID

[2] OK

[3] account name

[4] display name

[5] 0 or 1 : passport 가 증명되었는지 true/false 로 나타낸다.

[6] 0

(example)

>>> USR 7 TWN S t=53*1hAu8ADuD3TEwdXoOMi08sD*2!cMrntTwVMTjoB3p6t

<<< USR 7 OK test@passport.com TEST 1 0 ₩r₩n

OUT

NS 에서 로그오프 하는 방법은 모두에게 자신의 오프라인 상태를 알리는 것이다.
TrID 와 파라미터가 없는 OUT command 를 서버에게 보내면 서버는 즉시 연결을 끊는다.
스스로 로그오프 하는 경우 이외에도 서버 연결이 닫히는 경우가 있다.
첫번째 서버가 다운이 되는 경우이다.
이럴 경우에는 MSN 에서는 몇 분 전에 경고메시지를 보낸 후, 서버를 다운시킨다.
Server Shuttin Down 을 의미하는 OUT SSD 를 보낸다.
두번째 경우, 로그인 되어 있는 상태에서 다른 곳에서 다시 로그인 할 때이다.
이때는 original 연결은 닫히고, NS 는 client 에게 OUT command 를 보낸다.

3.2 Presence information

CHG

자신의 상태를 변경하거나 세팅할때 사용한다.
>>> CHG [1][2][3]₩r₩n
<<< CHG [1][2][3]₩r₩n
[1] TrID
[2] state
Statuses는 크게 on-line과 offline으로 나뉜다.
on-line 에 남아있으면서 상태를 닫았을때가 있는데 이런 경우를 “hiding" 또는 "appearing offline" 이라고 부른다.
sub-state로는 다음과 같은 경우가 있다.
NLN - Available
BSY - Busy
IDL - Idle
BRB - Be Right Back
AWY - Away
PHN - On the Phone
LUN - Out to Lunch
[3] client ID
사용할 수 있는 기능과 지원하는 프로토콜버전의 합
1, 2, 4, 7, 16, 32,64, 128, 268435456(MSNC1), 536870912(MSNC2)
각 상황에 따라 이 숫자들을 더해서 client ID를 구성한다.
(example)
>>> CHG 12 NLN 6₩r₩n
<<< CHG 12 NLN 6₩r₩n

ILN

등록된 사람들 중에서 현재 접속중인 사람들의 상태정보를 받는다.
접속 당시에만 이 명령어가 사용된다.
내가 아닌 다른 사람의 정보를 의미하므로 ILN 명령 이후의 파라미터는 모두 buddy 정보가 된다.

<<< ILN [1][2][3][4][5]₩r₩n
[1] TrID
[2] buddy state
[3] buddy account name
[4] buddy display name
[5] buddy client ID

(example)
<<< ILN 12 NLN test@hotmail.com test 536870948₩r₩n

NLN / FLN

초기에 친구들의 상태정보를 받는데 ILN 명령어가 사용되었다면, 그 이후에는 NLN 명령을 통해서 변경되는 상태정보를 받을 수 있다.
<< NLN [1][2][3][4]₩r₩n
[1] buddy state
[2] buddy account name
[3] buddy display name
[4] buddy client ID

접속중 이었던 친구가 로그오프 한 경우에는 FLN 명령어를 받는다.
<<< FLN [1]₩r₩n
[1] buddy account name

(example)
<<< FLN example@passport.com₩r₩n
<<< NLN NLN example@passport.com TEST 12₩r₩n
<<< NLN AWY example@passport.com TEST 0₩r₩n

3.3 Pings / Challenges

PNG / QNG

ping : 서버나 클라이언트에 의해 보내지는 command 이다.

ping 은 client pings와 server pings로 나누어 진다.

먼저 client pings 에는 PNG / QNG 이 있는데 모든 command 중에서 가장 간단하다.

TrID와 파라미터가 없이 사용된다.

>>> PNG₩r₩n

<<< QNG₩r₩n

CHL

서버에서 클라이언트가 살아있는지 확인하기 위해 보내는 신호로 server pings (challenges) 이며, 정기적이지 않다.

<<< CHL [1][2]₩r₩n

[1] always 0

[2] challenge string : 20-digit number

QRY

서버로부터 CHL 을 받으면 클라이언트는 서버에게 QRY를 보내 주어야 한다.

>>> QRY [1][2][3]₩r₩n[4]

[1] TrID

[2] client ID

[3] payload length : 이후 오는 byte 수를 나타낸다.

[4] client ID code : CHL 다음에 오는 해쉬를 'Q1P7W2E4J9R8U3S5'와 합친후에 16진수 처리하여 QRY 와 함께 답변한다. clieng ID code 이후에는 new line 이 없다.

<<< QRY [1]₩r₩n

[1] TrID

(example)

<<< CHL 0 15570131571988941333₩r₩n

>>> QRY 11 msmsgs@msnmsgr.com 32₩r₩n

8f2f5a91b72102cd28355e9fc9000d6e (no newline)

<<< QRY 11₩r₩n

3.4 Getting details

SYN

접속 후 동기화 ("synchronisation")

>>> SYN [1][2]₩r₩n

[1] TrID

[2] version num : 리스트가 바뀔 때마다 번호가 증가된다. 0이면 아무것도 저장된 것이 없는 상태를 의미한다. client와 server가 같은 버전을 가지고 있으면 리스트를 다운받을 필요가 없다. version num의 범위는 '0 ~ 2^32-1' 이다.

버전번호가 같을 때의 서버측 응답

<<< SYN [1][2]₩r₩n

[1] TrID

[2] version num : client가 보낸 버전 번호

버전번호가 다를때의 서버측 응답

<<< SYN [1][2][3][4]₩r₩n

[1] TrID

[2] version num

[3] 현재 접속해있는 사람수

[4] 그룹 수

(example)

>>> SYN 1 0₩r₩n

<<< SYN 1 139 5 4₩r₩n

앞으로 몇 개의 command는 SYN command 이후 뒤따라오는 command 이다.

GTC

누군가가 나를 추가하려고 할 때 무엇을 해야 하는지, RL변화에 대한 설정을 나타낸다.

GTC command 는 TrID를 포함하지 않는다.

List 종류에는 다음의 4가지가 있다.

Forward List(FL) : 나의 buddy list에 보여지는 사람이다. 150명 이상은 추가 할 수 없다.

Reverse List(RL) : 나를 FL에 등록한 사람

Allow List(AL) : 내 상태 보여주기를 허락한 리스트

Block List(BL) : 차단된 리스트

<<< GTC [1]₩r₩n

[1] A / N : default 는 A로 설정되어있다.

BLP

AL / BL에 있는 사람이 보낸 메시지 처리를 어떻게 할 것인지를 나타낸다.

<<< BLP [1]₩r₩n

1) AL/ BL : AL이라면 switchboard 세션에 초대하고, BL이라면 offline으로 알려준다.

PRP

폰 번호 세팅을 나타낸다.

<<< PRP [1][2]₩r₩n

[1] 번호타입 : 다음과 같은 5가지의 타입이 있다.

PHH - home phone number

PHW - work phone number

PHM - mobile phone number

MOB - are other people authorised to contact me on my MSN Mobile device?

MBE - do I have a mobile device enabled on MSN Mobile?

[2] 폰 번호

LSG

그룹 리스트를 나타낸다.

<<< LSG [1][2][3]₩r₩n

[1] 그룹 ID

[2] 그룹이름

[3] 0

LST

server 는 client의 buddy list중 현재 접속해 있는 리스트를 전송한다.

<<< LST [1][2][3]₩r₩n

[1] account name

[2] nickname

[3] list number

FL : 1, AL : 2, BL : 4, RL : 8 이며, 해당 buddy가 자신의 어떤 리스트에 속해있는지, 이 숫자들의 합으로 리스트 번호가 구성된다.

여기까지가 SYN command 이후 뒤따라 오는 세부사항이다.

(example)

>>> SYN 1 0₩r₩n

<<< SYN 1 139 5 4₩r₩n

<<< GTC A₩r₩n

<<< BLP AL₩r₩n

<<< PRP PHH 01%20234₩r₩n

<<< PRP PHM 56%20789₩r₩n

<<< LSG 0 Other%20Contacts 0₩r₩n

<<< LSG 1 Coworkers 0₩r₩n

<<< LSG 2 Friends 0₩r₩n

<<< LSG 3 Family 0₩r₩n

<<< LST principal1@passport.com principal1 4₩r₩n

<<< LST principal2@passport.com principal2 10₩r₩n

3.5 Setting details

GTC / BLP / PRP

GTC를 세팅할 때 사용한다.

GTC 값은 A/N 이다. 이 값은 서버에서 바꾸어 주지 않고 사용자가 바꾸어 줄 수 있다.

TrID 와 새로운값을 보내주면 된다.

>>> GTC [1][2]₩r₩n

[1] TrID

[2] A / N

<<< GTC [1][2][3]₩r₩n

[1] TrID

[2] new version

[3] 사용자가 세팅한 값 (A/N)

BLP 은 GTC 세팅과 동일하다.

PRP는 폰 번호는 세팅할 때 사용한다.

>>> PRP [1][2][3]₩r₩n

[1] TrID

[2] 번호타입

[3] 세팅할 폰 번호

<<< PRP [1][2][3][4]₩r₩n

[1] TrID

[2] new version

[3] 세팅된 번호타입

[4] 세팅된 폰 번호

(example)

>>> GTC 20 A₩r₩n

<<< GTC 20 200 A₩r₩n

>>> GTC 21 N₩r₩n

<<< GTC 21 201 N₩r₩n

>>> BLP 24 AL₩r₩n

<<< BLP 24 202 AL₩r₩n

>>> BLP 25 BL₩r₩n

<<< BLP 25 203 BL₩r₩n

>>> PRP 55 PHH 555-1234₩r₩n

<<< PRP 55 12183 PHH 555-1234₩r₩n

>>> PRP 56 PHW₩r₩n

<<< PRP 56 12184 PHW₩r₩n

ADD

친구등록하기(principals)

>>> ADD [1][2][3][4][5]₩r₩n

[1] TrID

[2] 추가하려는 리스트 위치

[3] account name

[4] nick name

[5] FL 에 등록하려고 할 때 그룹 아이디를 결정할 수 있다.

* 친구 등록 시 제한사항

1) RL로 친구 등록 할 수 없다.

2) AL과 BL로 동시 등록 할 수 없다.

3) FL 에서 150명 이상 등록 할 수 없다.

4) nickname 이 387바이트 이상이면 연결이 끊긴다.

ADD가 성공한 경우 서버 응답

<<< [1][2]₩r₩n

[1] new version

[2] TrID

FL로 등록했을 경우 서버로부터 BRP command를 full set으로 응답 받는다.

FL로 등록한 친구가 온라인상태라면 ILN 과 함께 초기상태를 응답 받는다.

(example)

>>> ADD 23 FL myname_123@hotmail.com myname_123@hotmail.com 1₩r₩n

<<< ADD 23 FL 1201 myname_123@hotmail.com myname_123@hotmail.com 1₩r₩n

<<< BPR 1201 myname_123@hotmail.com PHH₩r₩n

<<< BPR 1201 myname_123@hotmail.com PHW₩r₩n

<<< BPR 1201 myname_123@hotmail.com PHM₩r₩n

<<< BPR 1201 myname_123@hotmail.com MOB N₩r₩n

REM

친구 삭제하기

>>> REM [1][2][3][4]₩r₩n

[1] TrID

[2] 삭제하려는 리스트

[3] account name

[4] 그룹ID

<<< REM [1][2][3][4][5]₩r₩n

[1] TrID

[2] 삭제하려는 리스트

[3] new version

[4] account name

[5] 그룹ID

(example)

>>> REM 31 FL myname_123@hotmail.com₩r₩n

<<< REM 31 FL 1202 myname_123@hotmail.com₩r₩n

REA

사용자명 변경하기

>>> REA [1][2]₩r₩n

[1] account name

[2] URL 인코드된 새로운 닉네임

>>> REA [1][2][3]₩r₩n

[1] account name

[2] new version

[3] URL 인코드된 새로운 닉네임

(example)

>>> REA 103 mypassport@passport.com new%20name₩r₩n

<<< REA 103 3055 mypassport@passport.com new%20name₩r₩n

ADG

그룹 추가하기

>>> ADG [1][2][3]₩r₩n

[1] TrID

[2] 새로운 그룹명

[3] 0

<<< ADG 18 4029 MY%20NEW%20Group 4 0 ₩r₩n

[1] TrID

[2] new version

[3] 새로운 그룹명

[4] 그룹 ID

[5] 0

(example)

>>> ADG 38 My%20New%20Group 0₩r₩n

<<< ADG 38 4029 My%20New%20Group 4 0₩r₩n

RMG

그룹 삭제하기

>>> RMG [1][2]₩r₩n

[1] TrID

[2] 그룹ID

<<< RMG [1][2][3]40 4030 4 ₩r₩n

[1] TrID

[2] 새로운 버전번호

[3] 그룹ID : 0그룹을 삭제하려고 하면 에러가 발생한다.

(example)

>>> RMG 40 4₩r₩n

<<< RMG 40 4030 4₩r₩n

REG

그룹 이름 변경하기

>>> REG [1][2][3][4]₩r₩n

[1] TrID

[2] 그룹ID

[3] 새로운 그룹명

[4] 0

<<< REG [1][2][3][4][5]₩r₩n

[1] TrID

[2] new version

[3] 그룹ID

[4] 새로운 그룹명

[5] 0

(example)

>>> REG 43 3 My%20New%20Name 0₩r₩n

<<< REG 43 4031 3 My%20New%20Name 0₩r₩n

4. Switchboard

4.1 Requesting a Switchboard Session

XFR

Switchboard로의 세션 요청

>>> XFR [1][2]₩r₩n

[1] TrID

[2] SB

<<< XFR [1][2][3][4][5]₩r₩n

[1] TrID

[2] SB

[3]switchboard ip : port

[4] CKI

[5] 클라이언트가 스위치보드에 연결할 때 필요한 인증 코드

(example)

>>> XFR 15 SB₩r₩n

<<< XFR 10 SB 207.46.108.37:1863 CKI 17262740.1050826919.32308₩r₩n

---

USR

---

switchboard의 연결 이후 인증과정

>>> USR [1][2][3]₩r₩n

[1] TrID

[2] account name

[3] 인증코드

<<< USR [1][2][3][4]₩r₩n

[1] TrID

[2] 올바로 인증이 되었다면 OK

[3] account name

[4] display name

(example)

>>> USR 1 example@passport.com 17262740.1050826919.32308₩r₩n

<<< USR 1 OK example@passport.com Example%20Name₩r₩n

4.2 Getting Invited to a Switchboard Session

RNG

다른 사용자로부터 대화 요청을 받았을 때 서버로부터 오는 명령 (no TrID)

<<< RNG [1][2][3][4][5][6]₩r₩n

[1] 새로운 스위치보드 세션에서의 세션 ID

[2] ip : port

[3] CKI

[4] 인증 코드

[5] account name

[6] display name

ANS

RNG 에 대한 클라이언트의 응답으로 채팅서버에 접속하겠다는 의미이다.

>>> ANS [1][2][3][4]₩r₩n

[1] TrID

[2] account name

[3] 인증 코드

[4] 세션 ID

아래의 IRO command를 다 받은 이후에 서버로부터 마지막 확인 메시지로 ANS를 다시 받는다.

<<< ANS [1][2]₩r₩n

[1] TrID : IRO와 같은 TrID를 가진다

[2] OK

IRO

채팅 서버에 접속 시 이미 접속해 있는 친구의 정보를 얻는다.

ANS와 같은 TrID를 가지며, 이미 접속해 있는 사용자수만큼 받는다.

<<< IRO [1][2][3][4][5]1 1 2 bob@passport.com Bob₩r₩n

[1] TrID

[2] command 수만큼 하나씩 증가한다.

[3] IRO command 전체 수

[4] buddy account name

[5] buddy display name

예제를 통해 Getting Invited to a Switchboard Session 과정을 다시 정리한다.

(example)

<<< RNG 11752013 207.46.108.38:1863 CKI 849102291.520491113

example@passport.com Example%20Name₩r₩n

.... Client Connects to 207.46.108.38 1863 (Switchboard)

>>> ANS 1 test@hotmail.com 849102291.520491113 11752013₩r₩n

<<< IRO 1 1 2 example@passport.com Mike₩r₩n

<<< IRO 1 2 2 myname@msn.com My%20Name₩r₩n

<<< ANS 1 OK₩r₩n

4.3 Inviting Principals

CAL

채팅 서버에 친구추가를 서버에게 요청할 때

>>> CAL [1][2]₩r₩n

[1] TrID

[2] 초대하려는 친구의 account name

<<< CAL [1][2][3]₩r₩n

[1] TrID

[2] RINGING

[3] 이 switchboard의 session ID

JOI

초대된 친구가 채팅서버에 참여한다는 것을 알려준다. (no TrID)

<<< JOI [1][2]₩r₩n

1) 이 세션에 참가하려는 account name

2) display name

예제를 통해 Inviting Principals 과정을 정리한다.

(example)

>>> CAL 2 test@hotmail.com₩r₩n

<<< CAL 2 RINGING 11752013₩r₩n

<<< JOI test@hotmail.com test₩r₩n

BYE

친구가 세션에서 나갔을 때 나머지 사용자들에게 보내지는 명령 (no TrID)

<<< BYE [1]₩r₩n

[1] 세션에서 제외된 account name

(example)

>>> OUT₩r₩n // test@passport.com 사용자가 서버에게 보내는 command

<<< BYE example@passport.com₩r₩n // 세션에 있던 나머지 사용자에게 보내지는 명령

4.4 Message

MSG

실제 메시지를 보낼 때 사용되는 보내기

>>> MSG [1][2][3]₩r₩n

[1] TrID

[2] 승인타입 U / N / A

U : acknowledgement를 받지 않는다.

N : 메시지가 실패했을 경우 NAK를 받고 성공했을 경우는 아무것도 받지 않는다.

A : 메시지가 성공했을 경우 ACK를 받고 실패했을 경우 NAK를 받는다.

[3] payload 길이

이후로 MSG의 payload 가 온다.

(example)

>>> MSG 4 N 133₩r₩n

MIME-Version: 1.0₩r₩n

Content-Type: text/plain; charset=UTF-8₩r₩n

X-MMS-IM-Format: FN=Arial; EF=I; CO=0; CS=0; PF=22₩r₩n

₩r₩n

Hello! How are you?

<부록 2 - NATE On Messenger Protocol >

목 차

1. NATE ON Protocol

client는 NATE ON 접속을 위하여 dpl.nate.com(203.226.253.91)에 TCP/IP의 형태로 접속한다. port는 5004이고 version 은 현재 client 2.0 version이다.

MSN 과는 다르게 NATE ON에 접속할 수 있는 계정은 nate.com 또는 lycos.co.kr 뿐이다.

다음의 command 들은 로그인 과정부터, 메시지 전송과정까지 순차적으로 나열한 것이지만, 실제로 command 들을 이 순서대로 주고 받는 것은 아니다.

Command 들은 비동기적으로 처리되고 있다. Client가 보낸 하나의 command에 대해 반드시 server 의 응답이 있어야 다음 과정이 진행되는 것은 아니라는 의미이다.

2. Command 기본 구성

NATE ON command 는 4개의 문자와 파라미터, ₩r₩n으로 구성된다.

보통 첫번째 파라미터는 TrID를 나타내며, 1부터 시작해서 1씩 증가한다.

그 외의 command 기본구성과 설명은, 부록1의 MSN과 동일하다.

아직 밝혀지지 않은 일부 파라미터는 '…'으로 표시하였다.

3. 초기 접속 서버

3.1 Authentication

PVER

버전 정보를 알려준다.

>>> PVER [1][2][3]₩r₩n

[1] TrID

[2] …

[3] client version

<<< PVER [1][2]₩r₩n

[1] TrID

[2] client version

(example)

>>> PVER 1 2.25 2.0₩r₩n

<<< PVER 1 2.0₩r₩n

AUTH

인증 정보를 알려준다.

>>> AUTH [1][2]₩r₩n

[1] TrID

[2] DES

DES (Data Encryption Standard) ; 데이터 암호화 표준

DES는 개인 키를 사용하여 데이터를 암호화하는 방법으로서 널리 사용되며, DES에는 72,000,000,000,000,000 (72천조)개 이상의 암호 키가 사용되는 것이 가능하다.

주어진 각 메시지를 위한 키는, 이렇게 막대한 량의 키 중에서 무작위로 선택된다.

다른 개인키 암호화 방법과 마찬가지로, 송신자와 수신자 둘 모두는 동일한 개인 키를 알고, 사용 해야한다.

<<< AUTH [1][2]₩r₩n

[1] TrID

[2] md5

(example)

>>> AUTH 2 DES₩r₩n

<<< AUTH 2 md5₩r₩n

REQS

Server 에게 client 접속 요청하기

>>> REQS [1][2][3]₩r₩n

[1] TrID

[2] DES

[3] client ID

<<< REQS [1][2][3][4]₩r₩n

[1] TrID

[2] dp …:…

[3] 접속할 ip

[4] 접속할 port

(example)

>>> REQS 3 DES test@nate.com₩r₩n

<<< REQS 3 dp12:43799 203.226.253.171 5004₩r₩n

LSIN

새로운 연결하기

서버로 받은 IP와 PORT를 가지고 새로운 접속(Socket 연결)을 한다.

(MSN의 표현을 빌리면 DS의 연결에서 NS의 접속을 하는 것)

새로운 IP와 PORT의 접속을 한 뒤에 자신의 ID와 인코드된 Password를 보낸다.

>>> LIST [1][2][3][4]₩r₩n

[1] TrID

[2] client ID

[3] encoded password

[4] TE3

<<< LIST [1][2][3][4][5][6][7][8]₩r₩n

[1] TrID

[2] 개인 고유 번호 : 각 사용자마다 자신의 유일한 개인번호가 주어진다.

[3] 실제 등록자 이름

[4] nickname : 대화명 중 띄워쓰기는 %20 으로 표시된다.

[5] 핸드폰 번호

[6] 계정 등록 시 기입한 다른 E-mail 주소

[7]…

[8]…

(example)

>>> LIST 1 test@nate.com e21420f243b8050dba324ldo83e TE3₩r₩n

<<< LIST 1 1098225 홍길동 test%20test 01234567890 test@hotmail.com
169B84B70A9969F4452019A383CEF8D00326CA8741A1FBBC28C76A17E11603E6D5EF33
8D465F82E982FDB2EA0251BF0ED281EE17E3B086B6C825452D0850F1E3B408BCC1C38
EAB8E 17100022₩r₩n

3.2 Getting details

CONF

환경 설정에 대한 정보

>>> CONF [1][2][3]₩r₩n

[1] TrID

[2] …

[3] …

<<< CONF [1][2][3]₩r₩n

[1] TrID

[2] …

[3] …

(example)

>>> CONF 2 127 0₩r₩n

<<< CONF 2 127 1₩r₩n

GLST

자신에게 등록 되어있는 GROUP LIST에 대한 정보를 보내준다.

>>> GLST [1][2]₩r₩n

[1] TrID

[2] 그룹리스트 변화를 나타내는 숫자

이 숫자는 1부터 시작하며, 자신의 그룹에 대한 변화가 있을 때마다 변화한다.

그룹 추가 시에는 숫자가 2씩 증가하며, 그룹 삭제 시에는 숫자가 1씩 증가한다.

그러므로 이 숫자가 감소하는 경우는 없다.

Client 와 server의 그룹리스트 번호가 같을 경우에는 다시 보내줄 필요가 없다.

이럴 경우의 서버 응답은 다음과 같다.

<<< GLST [1][2]₩r₩n

[1] TrID

[2] 서버측에서 가지고 있는 그룹 리스트에 대한 숫자

<<< GLST [1][2][3]₩r₩n

[1] TrID

[2] …

[3] …

server로부터 오는 두 개의 연속되는 명령은 client 가 GLST를 보낸 직후에 바로 응답이 오지 않는다. NATE ON 의 기타 다양한 정보가 온 뒤에 GLST 에 대한 정보가 들어온다.

만약 client 와 server 의 그룹리스트 번호가 다를 경우에는 server 는 자신이 가지고 있는 최신 그룹리스트를 client 에게 다시 보내준다.

<<< GLST [1][2]₩r₩n

[1] TrID

[2] 서버가 가지고 있는 새로운 그룹리스트 번호

<<< GLST [1][2][3][4][5][6]₩r₩n

[1] TrID

[2] 전체 그룹 수 중 현재 위치

[3] 전체 그룹 수

[4] Y / N : 그룹이 현재 보이는지, 보이지 않는지를 나타낸다.

그룹을 만들었다가 삭제 했다고 하여도, 서버는 삭제된 그룹을 N 옵션으로 가지고 있다.

[5] 그룹 번호 : 이후 LIST 명령의 12번째 파라미터의 그룹 번호를 통해 해당 사용자가 어떤 그룹에 위치하는지 알 수 있다.

[6] 그룹 이름

(example)

Server 와 client 의 그룹리스트 번호가 같은 경우

>>> GLST 3 10₩r₩n

<<< GLST 3 10₩r₩n

<<< GLST 3 0 0₩r₩n

Server 와 client 의 그룹리스트 번호가 다른 경우

>>> GLST 3 13₩r₩n

<<< GLST 3 331₩r₩n

<<< GLST 3 0 3 Y 0 GROUP1

<<< GLST 3 1 3 Y 104 GROUP2

<<< GLST 3 2 3 N 264634 0

LIST

client가 server에게 자신의 buddy list를 요청한다.

>>> LIST [1]₩r₩n

[1] TrID

<<< LIST [1][2][3][4][5][6][7][8][9][10][11][12][13]₩r₩n

[1] TrID

[2] list 중 현재 위치 : [3]번 파라미터는 전체 buddy 수를 나타내는데, 0부터 1씩 증가하며 나타난다.

[3] list 전체 수 : 나의 buddy list 전체 수를 나타낸다.

[4] 사용자의 리스트 위치

리스트는 MSN 과 마찬가지로 4개로 구성되어 있다.

4개의 숫자로 이루어져 있는데, 각각의 숫자는 리스트 하나씩을 의미한다.

1이면 해당 리스트에 포함되어 있다는 의미이고, 0 이면 해당리스트에 포함되어 있지 않다는 의미이다. 리스트는 FL(Forward List), AL(Allow List), BL(Block List), RL(Reverse List) 순으로 위치한다.

예를 들어 0011 이라면 FL과 AL 에는 위치하지 않고, BL과 RL 에만 위치하고 있다는

의미이다. 이것이 의미하는 바는 0011 인 상대가 나의 리스트에는 없으며, 나의 온라인 상황을 볼 수 없는 상태이고, 나에겐 차단된 상태이며, 상대방에게는 내가 등록되어 있다는 것이다. 이것은 내가 상대방을 차단한 후에 삭제하였음을 의미한다.

[5] buddy ID

[6] 개인 고유 번호

[7] 실제 등록자 이름

[8] nick name

[9] 등록한 핸드폰 번호 : 등록하지 않았을 경우 %00으로 표시됨.

[10] 등록한 메일 주소 : 등록하지 않았을 경우 %00으로 표시됨.

[11] 실제 등록자의 생년월일

[12] 그룹 번호

[13] 1

(example)

>>> LIST 4₩r₩n /* LIST TrID */

<<< LIST 4 0 4 1101 test1@nate.com 335998 …

<<< LIST 4 1 4 1101 test2@nate.com 1098225 …

<<< LIST 4 2 4 1101 test3@nate.com 4662156 …

<<< LIST 4 3 4 1101 test4@nate.com 1928402…

3.3 Setting details

CNIK

자신의 대화명을 변경할 때 서버에게 보내는 명령어이다.

>>> CNIK [1][2]₩r₩n

[1] TrID

[2] new nick name

<<< CNKI [1]₩r₩n

[1] TrID

RENG

자신의 버디리스트에 있는 그룹명을 변경할때 서버에게 보내는 명령어이다.

>>>RENG [TrID] [serial_num] [group_index] [group_name]

[1]TrID

[2] serial number

[3] group index

[4] group name

<<< RENG [1][2]₩r₩n

[1] TrID

[2] serial number

ADDB

친구를 추가할 때 서버에게 보내는 명령어이다.

또는 친구를 차단하고, 차단 해제 할 때에도 사용된다.

>>> ADDB [1][2][3][4][5]₩r₩n

[1] TrID

[2] List (AL/ FL/ BL/ RL)

차단하는 경우는 [BL], 차단 해제하는 경우는 [AL]로 설정하면 된다.

[3] 파일방 이름

[4] 추가하는 상대방 ID

[5] 0

<<< ADDB [1][2][3][4]₩r₩n

[1] TrID

[2] group number

[3] buddy name

[4] serial number

RMVB

친구를 삭제할 때 서버에게 보내는 명령어이다.

>>> RMVB [1][2][3][4][5]₩r₩n

[1] TrID

[2] List (AL/ FL/ BL/ RL)

[3] group number

[4] buddy id

[5] 0

<<< RMVB [1][2]₩r₩n

[1] TrID

[2] serial number

RMVG

그룹을 삭제할 때 서버에게 보내는 명령어이다.

그룹을 삭제할 때는 해당 그룹에 속하는 사용자가 한명도 없어야 한다.

>>> RMVG [1][2][3]₩r₩n

[1] TrID

[2] serial number

[3] group number

<<< RMVG [1][2]₩r₩n

[1] TrID

[2] serial number

3.4 Presence Information

ONST

자신의 현재 상태를 서버에게 알려준다.

>>> ONST [1][2][3]₩r₩n

[1] TrID

[2] 상태 표시

O : 온라인

F : 오프라인

A : 자리 비움

B : 다른 용무 중

M : 회의 중

P : 통화 중

[3] 0

INFY

현재 온라인 상태에 있는 buddy의 list를 받는다.

<<< INFY [1][2][3][4][5][6]₩r₩n

[1] TrID

[2] buddy ID

[3] 상대방의 현재 상태

[4] 0

[5] 파일방 설정 이름 : 설정하지 않았을 경우 %00으로 표시된다.

[6] 1

(example)

>>> ONST 5 O 0 %00 1

<<< INFY 0 test1@nate.com A 0 %00 1

<<< INFY 0 test2@nate.com O 0 %00 1

<<< INFY 0 test3@nate.com O 1 FILE 1

NTFY

새로 로그인 되는 buddy가 있다면 NTFY 명령을 통해서 알려준다.

또는 상대방의 상태정보가 변경될때도 NTFY 명령이 온다.

<<< NTFY [1][2][3][4][5][6]₩r₩n

[1] TrID

[2] buddy ID

[3] 상대방의 현재 상태

[4] 0

[5] 파일방 설정 이름 : 설정하지 않았을 경우 %00으로 표시된다.

[6] 1

(example)

<<< NTFY 0 you@nate.com O 0 %00 1

NNIK

상대방의 대화명이 변경되었을 때 서버로부터 오는 명령이다.

<<< NNIK [1][2][3]₩r₩n

[1] 0

[2] buddy ID

[3] new buddy nick

3.5 Ping

PING

일정 시간 동안 작업 진행이 없을 경우, server에서 ping 을 보낸다.

Client도 ping으로 응답을 한다.

<<< PING [1]₩r₩n

>>> PING [1]₩r₩n

[1] TrID

4. 대화 연결 서버

4.1 Requesting a conversation Session

RESS

Client는 대화를 하기 위해 서버에게 대화 세션을 요청한다.

>>> RESS [1]₩r₩n

[1] TrID

<<< RESS [1][2][3][4]₩r₩n

[1] TrID

[2] 대화를 위한 IP

[3] PORT

[4] Key : 이 key를 통해서 대화 세션에 참가할 수 있다.

(example)

>>>RESS 6

<<< RESS 6 203.226.253.143 5004 31407D17204A017651BF004F38D0

4.2 Inviting Principals

CTOC

Client TO Client

Client 가 client 에게 보내는 명령이다.

>>> CTOC [1][2][3][4]₩r₩n

[1] TrID

[2] buddy ID : 이 명령을 받는 쪽의 ID 이다.

[3] N

[4] byte length : 이 숫자 이후에 나오는 명령의 byte 수이다.

보통 CTOC 다음에 byte length 만큼 다른 명령이 뒤따른다.

상대방으로부터의 CTOC 응답도 같은 구성이며, [2]위치의 ID 는 항상 같다.

만약 client A 가 client B 에게 먼저 CTOC 명령을 보내면 A와 B사이의 CTOC 명령 사이에서의 ID 는 B의 ID 가 된다.

INVT

상대방을 자신의 대화 세션으로 초대하는 명령이다.

앞에서 RESS를 통해서 서버로부터 전달받은 IP와 PORT, Key를 사용한다.

CTOC 명령 뒤에 위치한다.

>>> INVT [1][2][3][4]₩r₩n

[1] 자신의 ID : 초대하려는 buddy의 ID 가 아니다.

[2] IP : RESS를 통해 전달받은 IP

[3] PORT

[4] KEY

REQC

상대방을 자신의 대화 세션으로 초대하고, 자신의 정보를 상대방에게 전달한다.

CTOC 명령 뒤에 위치한다.

>>> REQC [1][2][3]₩r₩n

[1] NEW

[2] 자신의 IP : PORT

[3] 자신의 ID : …

CTOC 명령 뒤에 위치한다.

<<< REQC [1][2][3]₩r₩n

[1] RES

[2] 상대방의 IP : PORT

[3] 상대방의 ID : …

ENTR

대화 세션에 연결되었음을 서버에게 알려준다.

>>> ENTR [1][2][3][4][5]₩r₩n

[1] TrID

[2] 자신의 ID

[3] nick name

[4] 이름

[5] key

<<< ENTR [1]₩r₩n

[1] TrID

JOIN

대화 요청한 상대방이 해당 대화 세션에 연결되었음을 알려준다.

<<< JOIN [1][2][3][4]₩r₩n

[1] TrID

[2] 상대방의 ID

[3] 상대방의 nick name

[4] 상대방의 이름

예제를 통해 대화 세션을 요청하여 연결하고, 상대방이 수락하여 실제 대화가 가능한 상태까지의 상황을 살펴본다.

(example)

>>> RESS 6

<<< RESS 6 203.226.253.143 5004 31407D17204A017651BF004F38D0

>>> CTOC 7 you@nate.com N 73

>>> INVT me@nate.com 203.226.253.143 5004 31407D17204A017651BF004F38D0

>>> CTOC 8 you@nate.com N 53

>>> REQC NEW 210.123.39.84:5004 me@nate.com:5560

<<< CTOC 9 you@nate.com N 53

<<< REQC RES 210.123.39.86:5004 me@nate.com:5560

>>> ENTR 10 me@nate.com ME 홍길동 31407D17204A017651BF004F38D0

<<< ENTR 10

<<< JOIN 11 you@nate.com YOU 솜이

4.3 Getting Invited to a conversation session

USER

상대방으로부터 대화세션으로 초대받는 경우도, 내가 초대하는 경우와 거의 비슷하다. ENTR 명령을 통해 내가 대화 세션에 참여 했음을 서버에게 알리고 나면, USER 명령을 통해 현재 대화세션에 있는 buddy 정보를 나에게 알려준다.

<<< USER [1][2][3][4][5][6]₩r₩n

[1] TrID

[2] 대화 세션 참여자 번호 : 전체 참여자

[3] 대화 세션의 전체 참여자

[4] 참여한 buddy ID

[5] 참여한 buddy nick name

[6] 참여한 buddy 이름

예제를 통해 대화 세션에 요청 받았을 경우, 실제 대화가 이루어 지기까지의 과정을 살펴본다.

(example)

<<< CTOC 1 me@nate.com 72

<<< INVT me@nate.com 226.253.143 5004 283B74E116F80B7C523F004F2E47

<<< CTOC 1 me@nate.com 53

<<< REQC NEW 210.123.39.86:5004 me@nate.com:13901

>>> CTOC 2 me@nate.com 53

>>> REQC RES 210.123.39.85:5004 me@nate.com:13901

>>> ENTR 3 me@nate.com test 홍길동 283B74E116F80B7C523F004F2E47

<<< USER 4 1 1 you@nate.com YOU 솜이

<<< ENTR 1

QUIT

상대방이 참여하고 있던 대화 세션에서 나갈 때 알려주는 명령이다.

<<< QUIT [1][2]₩r₩n

[1] TrID

[2] buddy ID

4.4 Message

MESS

Incoming message와 outgoing message가 있는데, 둘의 구성은 동일하다.

1) Typing 시작할 때

>>> MESG [1][2][3][4]₩r₩n

[1] TrID

[2] client ID

[3] TYPING

[4] 1 / 2

2) 실제 메시지를 전송 할 때

>>> MESG [1][2][3][4]₩r₩n

[1] TrID

[2] client ID

[3] MSG

[4] Message 정보구성

%09를 구분자로 하여, 메시지의 정보를 나타낸다.

[글꼴정보]%09[글씨색]%09[글씨상태]%09[실제 메시지]

3) 이모티콘 전송 할 때

메시지 전송 때와 구성이 같고, 실제 메시지의 위치에 '/이모티콘 이름/' 이 위치한다.

(example)

MESG 6

MESG 7 MSG 굴림%090%09%09안녕하세요%20테스트중입니다.

MESG 7

MESG 8 TYPING 2

MESG 8

MESG 9 TYPING 1

MESG 9

MESG 10 MSG 굴림%090%09%09/엉엉/

MESG 10

< 부록 3 – Gateway Server – Mobile Client Protocol >

목 차

1. Gateway Server – Mobile Client Protocol

Gateway Server – Mobile Client Protocol은 Client와 Gateway Server 사이의 command의 집합으로 구성되어있다. 다음에서는 Mobile Client 가 Gateway Server로 보내는 command를 상황 별로 설명한다. 그 다음 Gateway Server가 Mobile Client로 보내는 command를 상황 별로 설명한다.

먼저 command의 기본 구성을 살펴 본 후, 자체 제작한 Gateway Server – Mobile Client Protocol의 command의 이름과 세부 설명, 예제를 통해 구성을 알아본다.

2. command 기본 구성

command는 한 개의 문자와 몇 개의 파라미터로 구성되어있다.

command가 한 개의 문자인 것은 모바일에서 패킷 비용을 절감하기 위해서이다.

모든 command가 다른 파라미터의 개수를 가지고 있으며, 파라미터가 없는 경우도 있다. 또한 게이트웨이로 전송한 command와 게이트웨이가 응답한 command가 같지 않은 경우도 있다. 각 파라미터는 공백으로 구분된다.

한 개의 command 이후에는 파라미터를 나타내는데, [] 기호를 통해 표현한다. 마지막으로는 실제로 주고 받은 예제를 통해 어떻게 사용되는지를 나타낸다.

3. Mobile Client 에서 Gateway로 보내는 프로토콜

3.1 로그인

---

i

Mobile Client에서 Gateway Server로 로그인을 요청한다. Gateway에서 보내 주는 응답은 없다.

>> i [1] [2]
[1] ID
[2] Password
(Example)
>> i redsung@msn.com red1212

3.2 로그아웃

---

o

Mobile Client에서 Gateway로 로그아웃을 요청한다. 로그인과 마찬가지로 서버에서의 응답은 없고, 파라미터는 없다.

>> o

(Example)
>>o

3.3 대화명 변경

---

r

Mobile Client에서 Gateway에 Client가 사용하는 닉네임의 변경을 요청한다. 서버에서의 응답은 없다.

>> r [1]
[1] 변경할 닉네임
(Example)
>> r 홍성민

3.4 해당 그룹의 리스트

---

g

Mobile Client가 Gateway에게 선택한 그룹의 리스트를 요청한다. 파라미터는 이미 알고 있는 그룹 넘버로, 그룹에 해당하는 숫자를 기억하고 있다가 그룹의 목록을 요청한다. 요청을 받은 Gateway는 뒤에 설명하는 l 명령을 통해 응답을 해준다.

>> g [1]
[1] 요청하는 그룹의 넘버
(Example)
>> g 1
<< l legna1004@hotmail.com (f)시스터즈(f) 미영
<< l shjung@hotmail.com (서코==청각바이러스)
<< l elmo8059@hotmail.com 카에루레아

3.5 메시지 전송

---

m

Mobile Client가 Gateway에 원하는 친구에게 메시지를 전달 할 것을 요청한다. 파라미터는 대화를 원하는 사용자와 전하고자 하는 메시지이다.
>> m [1] [2]

[1] 대화를 원하는 사용자의 ID
[2] 전달하고자 하는 메시지
(example)
>> m legna1004@hotmail.com 안녕!

4. Gateway에서 Mobile Client로 보내는 Protocol

4.1 그룹 리스트 전송

---

g

Mobile Client가 로그인에 성공하면 Gateway가 Client로 전송한다. 그룹 안의 리스트는 Cleint가 요청하기 전까지는 전송하지 않는다. 그룹 번호와 그룹 이름을 매칭 시켜 전달한다.

<< g [1] [2]
[1] 그룹 넘버
[2] 그룸 이름
(Example)
<< g 5 졸프조

4.2 친구 목록 전송

---

l

Mobile Client가 그룹 안의 리스트를 요청할 때 Gateway가 응답으로 보낸다. 요청한 그룹의 리스트를 Client에게 전송한다. 로그인 되어있는 사람의 리스트만을 전송해준다.

<< l [1] [2]
[1] 로그인 되어있는 사람의 아이디
[2] 로그인 되어있는 사람의 닉네임
(Example)
>> g 1
<< l legna1004@hotmail.com (f)시스터즈(f) 미영
<< l shjung@hotmail.com (서코==청각바이러스)
<< l elmo8059@hotmail.com 카에루레아

4.3 메시지 전송

m

Mobile Client가 대화를 요청한 상대의 응답이나, 다른 사용자가 Client에게 먼저 대화를 요청했을 경우 Gateway가 Client에게 보내는 응답 메시지이다. Client가 메시지를 전송할 때와 달리 닉네임을 같이 전송해준다. Client는 닉네임을 저장하고 있지 않기 때문이다. 전송하는 닉네임은 공백을 제거하여 8글자만 전송한다.

<< m [1] [2] [3]
(example)
>> m elmo8059@hotmail.com 안녕!
<< m elmo8059@hotmail.com 카에루레아 오랜만이야^^