

PDA상에서 동작하는 친구 찾기 메신저의 구현

김목련*, 박영호**, 송혜주***

요약

Hellow Fellow Messenger는 친구추가, 채팅, 쪽지 교환 등의 기본 메신저의 기능 아래 친구 찾기 기능에 중점을 둔 PDA와 PC에서 사용 가능한 메신저이다. 기존에 등록 된 친구 외에도 영화를 같이 볼 수 있는 친구, 편안한 대화를 할 수 있는 친구, 같이 운동을 할 친구 등 사용자들은 자신이 찾고 싶은 조건에 맞는 친구 찾기가 가능하며, 찾은 친구와 쪽지 교환, 대화가 가능하다. 매일 달라지는 기분이나 만나기 원하는 친구의 조건에 부합하는 새로운 친구를 찾는 것이 가능해 지는 것이다. 이를 위하여 서버는 데이터베이스로부터 필요한 사용자의 데이터를 불러들여 조건에 맞는 데이터를 클라이언트에 전송하고, 클라이언트는 이 데이터를 받아 원하는 결과를 출력해준다. 클라이언트-서버 통신은 MFC[1]를 이용하여 구현하였다. 제안하는 메신저는 PC 뿐만 아니라 PDA를 통해서도 사용 가능하여 이동성이 있고 무료 채팅기능을 통하여 문자메시지 대응으로 사용할 수 있다는 장점을 가진다.

An Implementation of a Messenger for Finding Well-Matched Friends on PDAs

Mok-Ryun Kim*, Young-Ho Park**, Hye-Ju Song***

Abstract

Recently, As the increase in interaction between the person and the person, the people plentifully use a messenger. Thus, we propose a new messenger program, we call Hellow Fellow Messenger. The hellow fellow messenger has functions of general messenger such as chatting, note exchange, friend addition. Also, hellow fellow messenger provide with special function, which find friends satisfied conditions in PDA and PC environment. So, Client can find not only friend who is register already but also friend who is the possibility of seeing a movie and can comfortably talk together and so on. For this, server sends information for users satisfied in condition from databases to client and client selects a user among received information for users. Client-server communication is implemented using MFC[1].

Keywords : PDA, Messenger, Match Making

1. 서론

최근 PC를 사용자들은 누구나 웹상에서 친구와의 대화를 위해 메신저를 사용하고 있다. 메신저란 인터넷상에서 시간 지연 없이 메시지와 데이터를 주고받을 수 있는 소프트웨어이다. 인터넷에 연결되어 있다면 언제, 어디서나 친구와 대화가 가능하다는 편의에 의해 PC에서 사용 가능

한 메신저의 종류와 사용자수가 늘어나고 있다. 뿐만 아니라 이동성을 갖춘 PDA에서도 친구와의 대화가 가능한 기본 기능을 제공하는 메신저가 등장하였다. 그러나 현재 PDA용 메신저에는 친구 찾기 기능이 제공되고 있지 않다. 본 논문에서는 PC뿐 아니라 PDA에서도 사용자가 원하는 조건에 맞는 친구를 찾아 쪽지 교환과 대화가 가능하고, 공개하고 싶지 않은 정보는 비공개로 설정하여 개인정보를 보호 해 줄 수 있도록 하는 것에 초점을 맞춘 메신저를 제안한다.

친구 찾기를 원하는 사용자들은 비공개, 공개 설정을 통하여 우선 자신의 부가정보를 저장한다. 공개, 비공개 설정은 데이터베이스 user 테이블에 비공개 attribute를 추가하여 구현하였다.

※ 제일저자(First Author) : 김목련

접수일자:2006년12월15일, 심사완료:2007년01월24일

* 숙명여자대학교 멀티미디어과학 전공

ch3090431@nate.com

** 숙명여자대학교 멀티미디어과학 전공

*** 숙명여자대학교 멀티미디어과학 전공

서버는 각 사용자 정보를 데이터베이스에 저장한다. 친구 찾기 기능은 클라이언트가 원하는 조건을 선택하여 전송하면 서버는 데이터베이스와 연동하여 송부된 조건에 맞는 사용자를 검색하여 매치하는 클라이언트에 반송한다. 보다 자세한 내용은 3장에서 소개한다.

본 논문은 메신저 사용에 대해 다음과 같은 공헌을 제시한다.

- 이동성을 갖춘 친구 찾기 : PDA라는 이동성이 있는 무선 단말기에서 기존에 제공되지 않았던 친구 찾기 기능을 제안하고 구현하였다.
- 조건에 맞는 친구 찾기 : 사용자별 원하는 조건에 맞게 친구를 찾을 수 있는 기능을 제안하고 구현하였다.
- 사용자의 프라이버시 보장 : 공개하고 싶지 않은 항목은 비공개로 설정해 다른 사용자들에게 공개 되지 않도록 하는 프라이버시 보호 정책을 구현하였다.

본 논문의 구성은 다음과 같다. 2장에서는 기존에 제공되고 있는 관련 서비스와 관련 기술을 소개한다. 3장에서는 친구 찾기 기능의 방안과 알고리즘을 제시한다. 4장에서는 실험환경을 소개하고, 5장에서는 3장에서 제시한 구조를 기반으로 하는 실제 코드와 실행결과를 제시한다. 마지막으로 6장에서는 본 논문의 결론을 내린다.

2. 관련 연구와 관련 기술

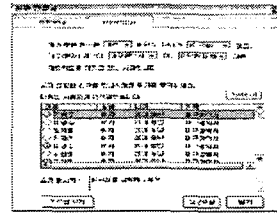
본 장에서는 현재 제공되고 있는 PC용, PDA용 메신저 서비스를 알아보고, 다음으로 본 구현에 사용한 기술에 대해 소개한다.

2.1 관련 연구

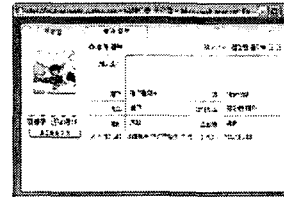
PC용 메신저로는 가장 널리 사용되고 있는 네이트온 메신저에서의 친구 찾기 기능과 PDA용으로 제공되고 있는 MSN 메신저를 살펴본다.

먼저 최근 들어 많은 사용자를 확보한 네이트온 메신저는 기본 메신저 기능에 친구 찾기 기능을 포함하고 있다. (그림 1)(a)에서 보는 것과 같이 조건에 맞는 친구 찾기가 가능하며 (그림

1)(b)의 프로필 보기, 상태 보기, 친구 요청 등이 가능하다. 하지만 우리가 제안하는 메신저는 PC에서 뿐만 아니라 PDA에서도 친구 찾기를 제공함으로써 시간과 공간에 제약받지 않고 원하는 순가네 친구를 바로 찾을 수 있다는 장점이 있다.



(a) 조건에 맞는 친구 찾기

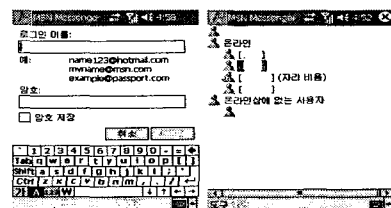


(b) 찾은 친구의 프로필 확인

(그림 1) PC용 네이트온 메신저의 친구 찾기

PDA용 MSN메신저는 (그림 2)(a)에서 보는 것과 같이 로그인 화면이 나타나고 로그인을 실시하면, (그림 2)(b)와 같이 등록되어 있는 친구가 화면에 나타나며 등록되어 있는 친구를 더블 클릭하면 채팅이 가능하다.

이와 같이, 기존의 PDA에서 제공해주는 기능들은 친구 추가, 대화, 쪽지 교환 등만 가능하다. PC용과 같은 새로운 친구 찾기 기능은 제공되고 있지 않다. 따라서, 본 논문에서는 PDA상에서 메신저의 기본기능의 제공 뿐만 아니라 나의 현재 기분 또는 요구에 따라 조건에 맞는 친구 찾기 기능을 추가한 PDA용 소프트웨어를 제안한다.



(a) 로그인 (b)대화상대 정렬
(그림 2) PDA용 MSN 메신저의 기본기능

2.2 관련 기술

2.2.1 MFC

Microsoft Foundation Class의 약자로 마이크로소프트에서 개발한 기본 클래스 라이브러리이다. 모든 운영체제는 API(어플리케이션 프로그래밍 인터페이스)함수를 제공한다. API란 응용 프로그램을 개발하기 위해 운영체제가 제공하는 함수이다. 이때, 윈도우즈 API란 윈도우즈 기반 프로그래밍을 쉽게 할 수 있도록 운영체제가 제공하는 여러 함수들의 모임이라 할 수 있다. 윈도우창을 C언어로 직접 구현 한다면 창하나 만드는 것도 매우 오랜 시간을 사용할 것을 하나의 API함수로 제공하기 때문에 함수 하나만 호출해도 윈도우창을 띄울 수가 있는 것이 장점이다. VC++6.0으로 코딩할 때 ::치면 여러 가지 함수들이 나오는데 거기에 있는 것 중 함수에 해당 하는 것이 바로 API함수이다. 윈도우계열에서 제공되는 API함수들은 모두 C언어로 작성되어 있다. MFC는 API함수를 사용하여 좀 더 쉽게 윈도우즈 프로그래밍 하기 위해 만들어졌다. 기반언어는 C++이다.

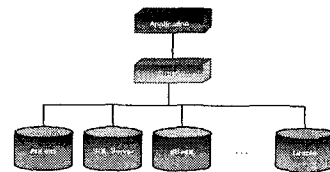
2.2.2 MySQL[2]

MySQL은 멀티유저, 멀티스레드SQL(Structured Query Language) 데이터베이스 서버이다. SQL은 가장 널리 쓰이는 표준 질의어이다. MySQL은 DB서버 데몬인 mysqld와 여러 가지 사용자 프로그램, 그리고 라이브러리들로 구성되어 있다. MySQL의 주목표는 속도, 뛰어난 처리능력, 그리고 사용의 편리함이다. 처음에 MySQL은 TcX[3]내부에서 자체적으로 사용할 목적으로 만들어 졌다. 그 당시 어떤 제조업체가 제공하는 SQL 서버 보다 강력한 DB서버가 필요했기 때문이다. TcX에서는 MySQL을 1996년부터 사용하여 왔는데 현재 700만 레코드 이상 되는 500개 이상의 테이블들이 주요 업무에 사용되고 있다고 한다. MySQL은 높은 수행능력을 발휘하는 것을 목표로 수년 전 개발이 시작 되었다. 여전히 개발이 진행 중이지만 매우 풍부하고 강력한 함수들을 제공한다.

2.2.3 ODBC[4]

ODBC(Open DataBase Connectivity)는 다양

한 종류의 데이터베이스를 액세스하기 위해 마이크로소프트가 정의한 표준 개방형 데이터베이스 인터페이스 규약을 말한다. 이 ODBC API를 이용하면 Microsoft Access, dBase, DB2, Text, SQL Server, Oracle등 여러 가지 종류의 데이터베이스를 같은 방법으로 액세스할 수 있다. 이를 위해 ODBC 소프트웨어 이외의 액세스할 각 데이터베이스마다 별도의 모듈이나 드라이버가 필요하다. 드라이버는 데이터베이스 벤더가 지원하며, 데이터베이스 제품별 혹은 버전별로 제공된다. ODBC는 오픈 그룹의 표준 SQL Call-Level 인터페이스에 근간을 두고 만들어졌다. ODBC는 프로그램들이 데이터베이스의 고유한 인터페이스에 대해 알지 못하더라도, 데이터베이스 액세스를 위한 쿼리를 사용할 수 있게 한다. ODBC는 SQL의 요청을 받아서, 그것을 개개의 데이터베이스 시스템들이 이해할 수 있는 요청으로 변환한다. ODBC는 SQL 액세스 그룹에 의해 만들어졌으며, 1992년 9월에 처음 나왔다. 처음에는 마이크로소프트가 윈도우용 ODBC 제품을 공급했지만, 지금은 유닉스, OS/2및 매킨토시 등을 위한 버전들도 출시되었다.



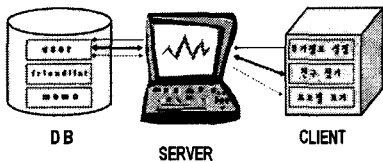
(그림 3) ODBC

3. 본 론

모든 메신저 유저는 친구 찾기 기능을 통해 나와 같이 메신저를 사용하는 모든 유저를 검색할 수 있으며, 찾은 즉시 관심 상대의 프로필 보기와 그 상대에게 쪽지를 보내는 기능을 사용할 수 있다. 친구 찾기 창에서는 유저의 아이디, 닉네임, 지역, 나이, 로그인/아웃 상태를 확인할 수 있으며 나이와 지역, 채팅 의도와 직업, 이성친구 등 조건에 맞는 유저를 검색할 수 있도록 구현하였다. 내 정보 설정 시 공개/비공개 옵션을 두어 친구 찾기 기능을 이용할 때엔 비공개로 설정한 사람은 찾을 수 없도록 하였다.

3.1 시스템구조

친구 찾기 창을 통하여 유저를 검색을 하기 위해서는 회원가입 시 데이터베이스의 user테이블에 저장한다. 이름, 성별, 아이디와 내 정보설정 창을 통해 이를 출력하기 위해서는 데이터베이스의 user테이블에 저장한 부가정보; 지역, 직업, 채팅의도를 가져와 화면에 출력하는 방식으로 구현하였다. 원하는 조건을 선택해 패킷에 담아 클라이언트가 서버에게 쿼리를 보내면 서버는 데이터베이스에서 클라이언트가 원하는 정보에 맞는 사용자를 찾아 다시 패킷형태로 클라이언트에게 보내주게 된다. 그렇게 받은 정보를 클라이언트는 화면에 보여 주는 것이다. (그림 4)는 전체적인 시스템 구조를 보여준다.



(그림 4) 클라이언트-서버, 데이터베이스 구조

3.2 데이터베이스 설계

처음 메시지를 사용하기 위해선 회원가입을 해야 한다. 회원 가입 시 기본적으로 저장해야 하는 사항은 이름, 아이디, 패스워드, 생년월일, 성별이다. 유저가 가입한 사항은 서버를 통해 데이터베이스의 user테이블에 저장되게 된다. 친구 찾기 기능을 사용하기 위해서는 부가정보도 설정해야 하는데, 부가정보인 지역, 직업, 채팅의도, 각각의 비공개/공개 여부도 user테이블에 저장된다. 비공개/공개 여부는 user테이블에 fage, flocation, fjob, faim attribute를 추가하여 나이를 비공개로 설정했을 시 fage에 0값이 저장되도록 공개로 설정했을 시에는 1값이 저장되도록 구현하였다. 유저의 친구 정보를 갖고 있는 friendlist 테이블에는 각각의 유저의 아이디, 친구 아이디, 친구의 상태(로그인/로그아웃)를 저장하고 있다. memo 테이블에는 쪽지를 보낸 사람 아이디, 받는 사람 아이디, 쪽지 내용이 저장되어 있어, 로그 아웃 시에도 쪽지 저장이 가능하다. (그림 5)에서 친구 찾기에 사용되는 테이블 구성을 볼 수 있다.

user		friendlist	memo
id	loc	id	id
pass	job	friendid	rid
name	plan	friendstate	msg
sex	fname		
year	fid		
month	fage		
day	fphone		
state	floc		
nick	fjob		
phone	fplan		
age			

(그림 5) 데이터베이스 각 테이블 구성

3.3 주요 기능

친구 메뉴의 친구 찾기를 클릭하면 친구 찾기 다이얼로그가 뜨게 된다. 조건별 검색이 가능하게 하는 나이, 직업, 채팅의도, 직업, 이성 친구 항목 등이 창 윗부분에 있다. 그 항목들을 원하는 조건으로 설정해서 검색버튼을 누르면 다이얼로그 창 아래 부분에 조건에 맞는 유저의 아이디, 닉네임, 지역, 나이, 로그인/아웃 상태가 나타나게 되며, 이 때 내 정보 설정에서 비공개로 처리한 유저는 뜨지 않게 된다.

검색된 유저 중 관심 상대에 대해 더 자세히 알고 싶은 경우, 다이얼로그 하단의 프로필보기 버튼을 클릭하면 프로필 창이 뜨게 된다. 이 때 프로필 창에서 이름, 성별, 아이디, 나이, 닉네임, 직업, 휴대폰, 지역, 채팅의도를 확인할 수 있다. 프로필 창 하단의 앞 뒤 버튼으로 프로필간 이동이 가능하며, 다시 새로운 창을 띄우는 수고를 덜 수 있도록 구현하였다. 프로필 보기 기능은 원하는 상대의 닉네임을 더블클릭 했을 때에도 확인 가능하다.

원하는 상대의 프로필을 확인한 후 쪽지를 보내고 싶으면 상대의 닉네임을 클릭하고 친구 찾기 창 하단의 프로필 보기 버튼 옆의 쪽지 보내기 버튼을 누르면 쪽지를 쓸 수 있는 다이얼로그가 뜬다. 내용을 입력한 후 확인을 누르면 상대방에게 쪽지가 전달된다. 이 때 상대방이 로그아웃 중이어도 쪽지가 성공적으로 전달된다. 로그아웃 중이었던 상대방은 로그인 했을 때 받은 쪽지를 확인할 수 있다.

3.4 개발환경

서버는 Microsoft Windows XP기반에 메인메모리 512MB, 하드디스크는 55GB인 PC와 데이터베이스서버, PDA(HP RW-6100)로 구성된다. 데이터베이스는 MySQL 5.0을 사용했고 MFC를 기반으로 하는 서버와의 연동은 ODBC 3.51을 설치하여 사용했다. PC용 소스는 Visual Studio C++ 6.0에서 작성하였으며, PDA용 소스는 Embedded Visual C++ 4.0, Microsoft Pocket PC 2003 SDK를 기반으로 작성하였다. 클라이언트-서버 구조[5]는 MFC에서 제공되는 CAsyncSocket(서버)과 CSocket(클라이언트)를 상속받아 필요한 함수만 재정의 하여 구현하였다.

4. 실행 결과

4.1 친구 찾기

(그림 6)(a)는 클라이언트 측 소스로 사용자가 친구를 찾을 때 나이, 성별, 지역, 채팅의도에서 각각 자신이 원하는 조건을 선택하여 패킷에 담아 서버로 보낸다. (그림 6)(b)은 클라이언트가 보내온 패킷 내용을 분석해 클라이언트가 원하는 조건에 맞는 사용자를 찾기 위해 데이터베이스에 조건별 쿼리를 보내는 서버 측 소스이다. 데이터베이스로부터 얻은 데이터를 다시 클라이언트에 보내준다. 서버가 보내 준 데이터를 받은 클라이언트는 친구 검색 창외의 리스트 컨트롤 안에 검색 내용을 출력해 준다. 여기서 확인 가능한 사항은 닉네임, 나이, 지역, 상태이다. 친구의 상태 확인이 가능해 로그인 중인 친구와는 바로 쪽지 교환이 가능하다.

```
void CSearchFriend::OnSearchFriend()
{
    int cnt=m_CtrList.GetItemCount()-1;
    for(int i=cnt;i>-1;i--)
        m_CtrList.DeleteItem(i);
    UpdateData(TRUE);

    //내 성별 얻어오기
    pack.type='s';
    lstrcpy(pack.msg.info.l_strID, myID);
    g_sock.Send(&pack, sizeof(Packet));
}
```

```
pack.type='k';
CString temp ;

lstrcpy(pack.msg.myinfo.l_strID, myID);
pack.msg.myinfo.age=m_maxage;
pack.msg.myinfo.minage=m_minage;

pack.msg.myinfo.loc=m_loca.GetCurSel();
pack.msg.myinfo.job=m_occu.GetCurSel();
pack.msg.myinfo.plan=m_aim.GetCurSel();

g_sock.Send(&pack, sizeof(Packet));
UpdateData(FALSE);
}
```

(a) 조건을 선택해 서버로 보내는 소스

```
// 친구 찾기
if(pack.msg.myinfo.loc&&pack.msg.myinfo.job
&& pack.msg.myinfo.plan)
{
    wsprintf(szQuery,"select count(*) from user
where nick!='NULL' and id!='%s' and
age>=%d and age<=%d and loc='%d' and
job='%d' and plan='%d' ",pack.msg.myinfo.
l_strID, pack.msg.myinfo.minage,
pack.msg.myinfo.age,pack.msg.myinfo.
loc,pack.msg.myinfo.job, pack.msg.myinfo.plan);
}
else if(!pack.msg.myinfo.loc&&pack.msg.myinfo.
job&& pack.msg.myinfo.plan)
{
    wsprintf(szQuery,"select,count(*) from user
where nick!='NULL' and id!='%s' and
age>=%d
and age<=%d and job='%d' and plan='%d'
", pack.msg.myinfo.
l_strID, pack.msg.myinfo.minage, pack.msg.
myinfo.age, pack.msg.myinfo.job,
pack.msg.myinfo.plan);
}
else if(pack.msg.myinfo.loc && !pack.msg.
myinfo.job && pack.msg.myinfo.plan)
{
    wsprintf(szQuery, "select count(*) from user
where
:
:
:"
```

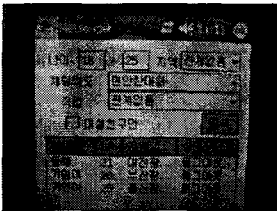
```

if(returnMsgType == Query_Success)

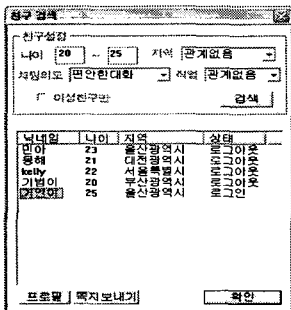
    send(rssock, (char *)&pack, sizeof(Packet),0);
    }
}
    
```

(b) 조건에 맞는 사용자 찾는 소스
(그림 6) 친구 찾기 소스

(그림 7)는 조건에 맞는 친구를 찾는 화면으로 (그림 7)(a)는 PDA용 화면, (그림 7)(b)는 PC용 화면이다. 이와 같이 PDA와 PC에서 같은 서비스 제공이 가능하다.



(a) PDA용 조건별 친구 찾기



(b) PC용 조건별 친구 찾기
(그림 7) 친구 찾기

4.2 프로필 보기

(그림 8)(a)는 서버 측 소스로 유저가 컨트롤리스트에 나타난 친구 중 한 사람의 닉네임을 선택하면 그 사람의 기본정보 및 부가정보를 데이터베이스로부터 가져와 클라이언트에 보내준다. (그림 8)(b)는 서버로부터 선택한 친구의 데이터를 받아 새로운 다이얼로그 창을 생성하고 그 친구의 내용을 출력해하도록 하는 소스이다. 여기서 각 사용자가 비공개로 선택한 사항은 '비공개'라는 메시지로 출력되도록 하였다.

```

void GetProfile(Packet pack)
{
    char szQuery[1024];
    ReturnMsgType_IncoptsDB returnMsgType;
    MULTIFIELDVALUELIST fieldMValueList;

    wsprintf(szQuery, "select * from user where
    nick='%s' ", pack.msg.myinfo.nick);
    returnMsgType =
    g_db.GetRecordList(szQuery,
    &fieldMValueList);

    for(int i=0 ; i<g_nUseCnt ; i++){
    if(!strcmp(pack.msg.myinfo.l_strID,
    cliInfo[i].strID)){
        rssock = cliInfo[i].sock;
    }
    }

    if(returnMsgType == Query_Success)
    {
        char id[16], job[3], plan[3];
        char phone[16], name[16], nick[16], age[16],
        loc[16],sex[16];
        char fname[2], fid[2], fage[2], fphone[2],
        flocc[2],fjob[2], fplan[2];

        sprintf(id, "%s", fieldMValueList[0][0]);
        sprintf(name, "%s", fieldMValueList[0][2]);
        :
        pack.type='p';
        strcpy(pack.msg.myinfo.nick, nick);
        strcpy(pack.msg.myinfo.l_strID, id);
        :
        pack.msg.myinfo.age=atoi(age);
        pack.msg.myinfo.sex = sex[0];
        :
        send(rssock, (char *)&pack, sizeof(Packet),0);
    }
    else if(returnMsgType == Query_Fail
    || Query_exception)
    {
        AfxMessageBox("해당 아이디 정보 없음");
    }
    }
}
    
```

(a) 프로필 전송하는 소스

```

BOOL CProfile::OnInitDialog()
{
    CDialog::OnInitDialog();
}
    
```

```

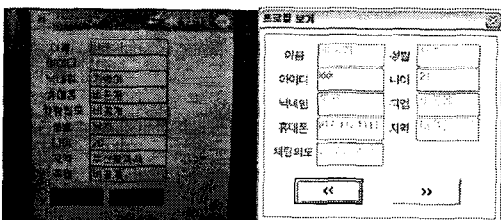
UpdateData(TRUE);
m_pnick=pack.msg.myinfo.nick;
if(pack.msg.myinfo.fname==0)
m_pname="비공개";
else m_pname=pack.msg.myinfo.l_strName;
:
if(pack.msg.myinfo.sex==='1') m_psex="남자";
else m_psex="여자";

if(pack.msg.myinfo.floc==0) m_ploc="비공개";
else {
switch(pack.msg.myinfo.loc){
case 0: m_ploc="설정안됨";break;
case 1: m_ploc="서울특별시";break;
:
}}
if(pack.msg.myinfo.fjob==0) m_job="비공개";
else {
switch(pack.msg.myinfo.job){
case 0: m_job="설정안됨";break;
case 1: m_job="공무원";break;
:
}}
UpdateData(FALSE);
return TRUE;
}
    
```

(b)서버로부터 전송받은 정보를 화면에 출력하는 소스

(그림 8) 프로필 보기 소스

(그림 9)(a),(b)에서 PDA용, PC용 프로필 보기 결과를 확인 할 수 있다.



(a)PDA용 프로필보기 (b)PC용 프로필 보기

(그림 9) 프로필 보기

4.3 쪽지 교환

(그림 10)(a)는 클라이언트가 검색된 친구 중 마음에 드는 상대의 닉네임을 선택하여, 상대의 닉네임, 나의 아이디, 쪽지 내용을 패킷에 담아

서버에 보내는 소스이다. (그림 10)(b)는 서버가 클라이언트에서 보내온 패킷에 포함된 닉네임으로 유저의 아이디를 알아내어, 해당 아이디를 가진 유저에게 쪽지 내용을 전달하는 소스이다.

```

void SendMemoID(Packet pack)
{
char szQuery[1024];
ReturnMsgType_IncopsDB returnMsgType;
MULTIFIELDVALUELIST fieldMValueList;

wsprintf(szQuery, "select * from user
where nick='%s' ",pack.msg.memo.szMsg);
returnMsgType =
g_db.GetRecordList(szQuery,
&fieldMValueList);

if(returnMsgType == Query_Success)
{ sprintf(pack.msg.memo.strRID, "%s",
fieldMValueList[0][0]);

for(int i=0 ; i<g_nUseCnt ; i++){
if(!strcmp(pack.msg.memo.strID, cliInfo[i].strID)){
rsock = cliInfo[i].sock;
} }
send(rsock, (char *)&pack, sizeof(Packet),0);
} }
    
```

(a) 원하는 상대에게 쪽지 보내는 소스

```

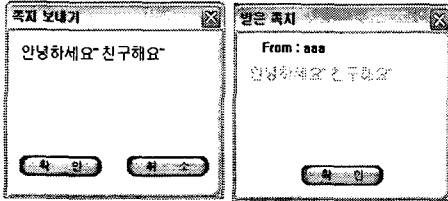
void CSearchFriend::OnMemo()
{
max=m_CtrList.GetItemCount();
for(int iltme = 0; iltme < max; iltme++)
{
if(m_CtrList.GetItemState(iltme,LVIS_SELECTED) != 0)
{
item=iltme;
CString nick=m_CtrList.GetItemText(iltme, 0);

pack.type='o';
lstrcpy(pack.msg.memo.szMsg, nick);
lstrcpy(pack.msg.memo.strID, myID);
g_sock.Send(&pack, sizeof(Packet));
} }
    
```

(b)서버에서 쪽지 전달하는 소스

(그림 10) 쪽지 교환 소스

(그림 11)(a),(b)를 통해 유저가 쪽지를 보낼 때 사용하는 화면, 쪽지를 받았을 때의 결과 화면을 확인할 수 있다.



(a) 쪽지 보내기 (b) 받은쪽지
(그림 11) 쪽지 교환

5. 결 론

본 논문에서는 PC에서는 제공되고 있으나 이동성을 갖춘 PDA 단말기에서 제공되고 있지 않은 다양한 패턴에 따른 친구 찾기, 프라이버시 보장이 가능한 친구 찾기 기능 추가에 초점을 두어 제시 하였다. HELLO FELLOW Messenger에서 사용하는 주요 기술은 서버-클라이언트 간의 통신, 데이터베이스 연동에 의한 데이터 저장, 색출, 갱신이다. 이 모든 기능이 PC뿐만 아니라 PDA에서도 사용 가능하도록 구현하여 사용자가 언제, 어디서든지 메신저 서비스 이용이 가능하도록 하였다. 기존에는 제공되고 있지 않은 서비스 제공이 가능하게 된 것이다. PC-PDA간의 파일 전송, 간혹 일어나는 패킷 유실 문제가 해결된다면 더욱 유용한 메신저 서비스가 될 것이라고 생각한다.

참고문헌

- [1] 전병선, Microsoft visual C++ 6.0 MFC programming second edition, p.93~135, p.163~218, p.859~933, p.105 7~1079, 삼양 출판사, 1999년
- [2] mysql <http://www.mysql.com/why-mysql/>
- [3] T.c.X <http://www.tcx.se/>
- [4] Open link software <http://uda.openlinksw.com/odbc/>
- [5] 임강진, 신재호, 권병희 공저, Network Bible 3rd edition, p.684~732, 729~739, 영진닷컴, 2001년
- [6] 고재관, Mobile PDA programming, p.143~292, 삼각형 익스프레스, 2001년
- [7] 전인호, 권태호, 서민환, 조재용, 김하영 공저, C Progra

mming Bible, p.884~917, 영진닷컴, 2004년

- [8] 최재규, Visual C#.NET 2005 2nd edition, p.363~393, 영진닷컴, 2005년



김 목 련

2007년 : 숙명여자대학교
멀티미디어과학(학사)
2007 - 현재 : 숙명여자대학교
멀티미디어과학(석사)
관심분야 : Information Retrieval (IR), index



박 영 호

1990년 : 동국대학교 공과대학
컴퓨터공학과(학사)
1992년 : 동국대학교 일반대학원
컴퓨터공학(공학석사)
2005년 : 한국과학기술원
전산학과(공학박사)
1993년 - 1999년 : 한국전자통신연구원(ETRI)
교환전송연구단 선임연구원
1999년 - 2006년 : (주)코스모 책임연구원
(데이터베이스팀)
2006년 - 현재: 모두스타 (주) 기술상무이사 (CTO)
2006년 - 현재: 숙명여자대학교 이과대학
멀티미디어과학과 조교수
관심분야 : Database Management System(DBMS)
Information Retrieval(IR), XML
Telecommunication System.



송 혜 주

2007년 : 숙명여자대학교
멀티미디어과학(학사)
2007 - 현재: 숙명여자대학교
멀티미디어과학(석사)
관심분야 : Database Management System(DBMS)