Санкт-Петербургский государственный университет информационных технологий, механики и оптики

Кафедра «Компьютерные технологии»

Е.В. Калугин, П.В. Графов

Программа для обмена сообщениями в локальной сети

Проектная документация

Проект создан в рамках
«Движения за открытую проектную документацию»
http://is.ifmo.ru/

Санкт-Петербург
2004

Оглавление

Введение

Пусть имеется локальная *TCP/IP* – сеть. Требуется написать программу для системы обмена сообщениями между компьютерами этой сети.

Программа должна обладать функциональностью, близкой к известной системе *ICQ*, которая обеспечивает мгновенный обмен текстовой информацией между различными пользователями в сети.

Система не требует использования сервера доставки сообщений, так как компьютеры обмениваются информацией напрямую.

Анализ аналогичных проектов с открытыми кодами в интернете показал, что они имеют неудобный интерфейс. Отсутствие проектной документации делает практически невозможной их модификацию.

В локальных сетях используются компьютеры под управлением различных операционных систем. Поэтому программа разрабатывается в двух вариантах – для операционных систем (ОС) *Linux* и *Windows*.

При этом программа под ОС *Linux* проектируется из двух частей – сервера (демона, реализованного в виде конечного автомата) и клиента (графического интерфейса), написанного традиционным путем.

В отличие от традиционных клиент-серверных приложений в данной работе взаимодействие сервера и клиента происходит в рамках одной машины. При взаимодействии между программами по сети обе стороны являются равноправными.

Для реализации сервера используется *SWITCH*-технология [1] (http://is.ifmo.ru). Сервер реализован на языке *Cu* [2], а клиент – на языке *Cu++* [3].

Программа под ОС *Windows* реализована в виде одной компоненты без применения автоматов на языке *Cu++* [3].

Выполнено сравнение этих реализаций, в результате которого сделан вывод, что отладка программы при использовании автоматного подхода существенно упрощается.

1. Описание *Linux*-версии программы

Данная программа реализована из двух частей: сервера, являющегося *демоном* (*daemon*) – резидентной программой в *UNIX*-системах, не связанная ни с одним пользовательским сеансом [4], и клиента – графического интерфейса пользователя.

Демон выполняет следующие функции:

– прием и обработку сообщений из сети с последующей отправкой клиенту;

– прием и обработку сообщений от клиента с последующей отправкой в сеть;

– обеспечение работы приложений при сильном трафике.

Рассмотрим подробнее схему работы демона. В нем для связи с внешним миром используются два сокета – сетевой *Ipv4* сокет (*AF_INET*) и локальный сокет (*AF_UNIX*).

Сетевой сокет предназначен для связи с приложениями, работающими на других хостах. Через него осуществляется отправка сообщений с компьютера, на котором запущен демон, в сеть. Предполагается, что получатель и отправитель находятся в одной сети. Этот же сокет обеспечивает прием сообщений, предназначенных для клиента, запущенного на данном хосте. Пусть сокет, например, связан с портом *3538*.

Локальный сокет предназначен для связи с клиентским приложением. Через него осуществляется отправка и прем сообщений от клиента. Пусть сокет связан с файлом /tmp/schat3538.

При получении сообщения демон производит его преобразование во внешний/внутренний формат для последующей отправки через соответствующий сокет. Такое поведение демона весьма удобно для клиентского приложения, так как позволяет ему не заботиться о сетевой составляющей при взаимодействии с удаленным приложением.

Клиент является второй составной частью программы. Он обеспечивает непосредственное взаимодействие с пользователем. Клиент является связующим звеном между пользователем и серверной частью, реализующей сетевое взаимодействие.

2. Серверная часть

2.1. Словесное описание

Основой демона является конечный автомат. Его граф переходов приведен в разд. 2.2. В нем используются следующие обозначения.

Состояния

0. Ожидание.
1. Прием локального сообщения.
2. Преобразование в сетевой формат.
3. Отправка сообщения в сеть.
4. Прием сообщения из сети.
5. Преобразование в локальный формат.
6. Отправка локального сообщения.
7. Выход.

События

e0 – в локальном сокете появилось сообщение;
e1 – в сетевом сокете появилось сообщение;
e2 – в локальном сокете сообщений не осталось;
e3 – в сетевом сокете сообщений не осталось;
e4 – превышен лимит по количеству сообщений, полученных подряд из локального сокета;
e5 – превышен лимит по количеству сообщений, полученных подряд из сетевого сокета;
e6 – завершение работы клиента или аварийная ситуация;
e7 – успешный прием локального сообщения;
e8 – успешное преобразование в сетевой формат;
e9 – успешная отправка сообщения в сеть;
e10 – успешный прием сообщения из сети;
e11 – успешное преобразование в локальный формат;
e12 – успешная отправка локального сообщения.

Действия

z0 – ожидание прихода сообщения в один из сокетов. При завершении процедуры z0 в зависимости от того, в какой сокет сообщение пришло, генерируется событие e0 либо e1;

z1 – получение сообщения из локального сокета. Если сообщения в сокете не оказалось, то генерируется событие e2;

z2 – преобразование локального сообщения для последующего отправки его в сеть;

z3 – отправка сообщения в сеть. Если количество сообщений, отправленных подряд, превышает установленное число, то генерируется событие e4;

z4 – получение сообщения из сетевого сокета. Если сообщения в сокете не оказалось, то генерируется событие e3;

z5 – преобразование сетевого сообщения для последующей отправки его локальному клиенту;

z6 – отправка сообщения локальному клиенту. Если количество сообщений, отправленных подряд, превышает установленное число, то генерируется событие e5.

z7 – завершение работы демона.

Автомат ожидает прихода сообщения. После его прихода, автомат производит преобразование сообщения во внешний/внутренний формат и отправку его по соответствующему сокету. В зависимости от того, откуда пришло сообщение, генерируется одно из событий *e0* или *e1*. Отметим, что после отправки сообщения автомат не переходит в режим ожидания. Он вновь пытается получить сообщение из того же источника, и лишь убедившись в его отсутствии, переходит в режим ожидания.

Автомат также обеспечивает защиту демона от перегрузок при чрезмерном трафике. В случае получения из сети более N сообщений подряд, генерируется событие *e5*, по которому автомат принудительно переходит на прием сообщений из локального сокета. Аналогично, в случае получения подряд более M сообщений из локального сокета, генерируется событие *e4*. По этому событию автомат принудительно переходит на прием сообщений из сети. События *e2* и *e3* генерируются в случае отсутствия сообщений в локальном и сетевом сокете соответственно.

2.2. Граф переходов

На рис. 1 приведен граф переходов, описывающий работу демона.

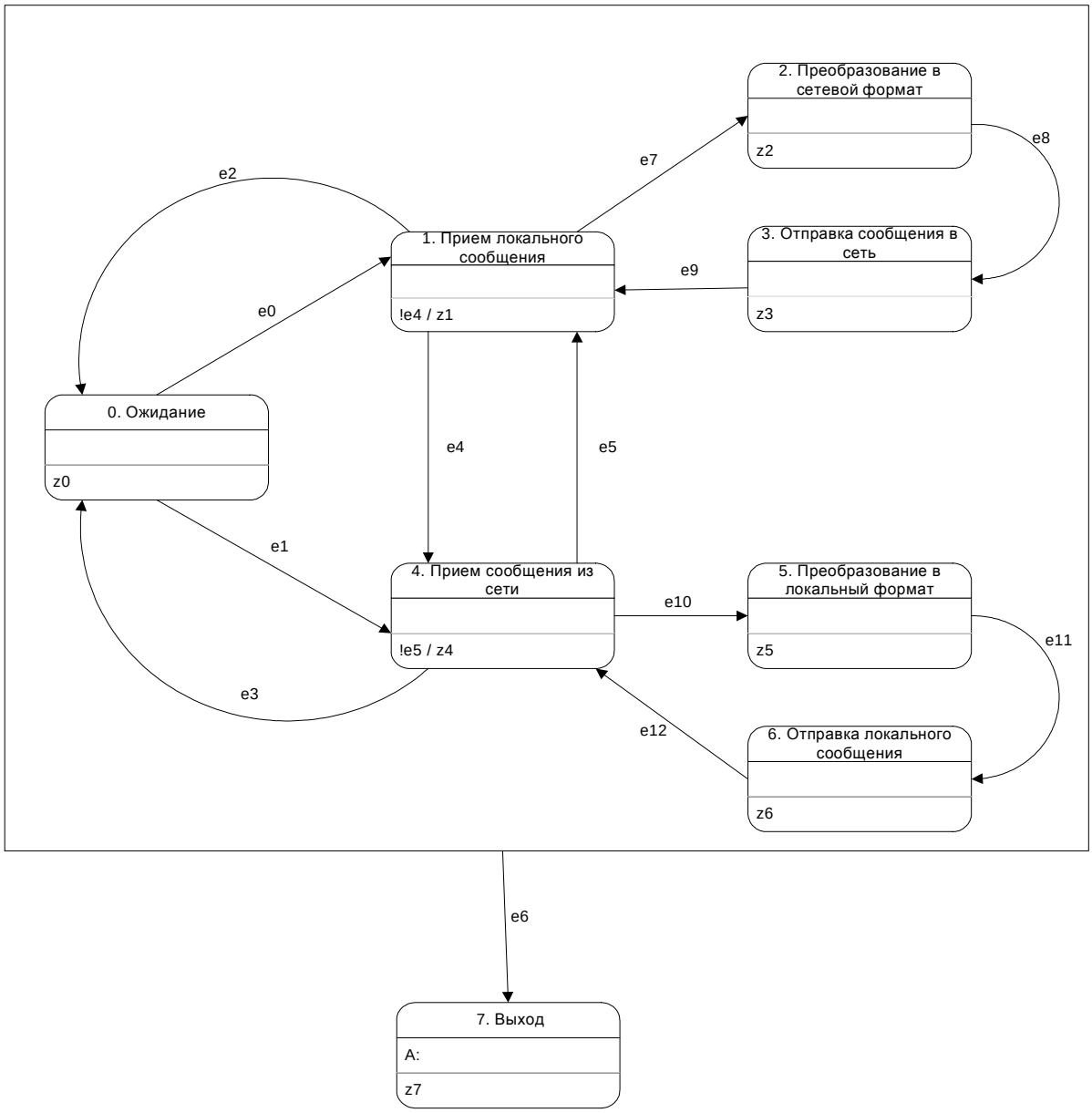

Рис. 1. Граф переходов, описывающий работу демона

3. Клиентская часть

На рис. 2 изображен пример работы *Linux* – клиента**.**

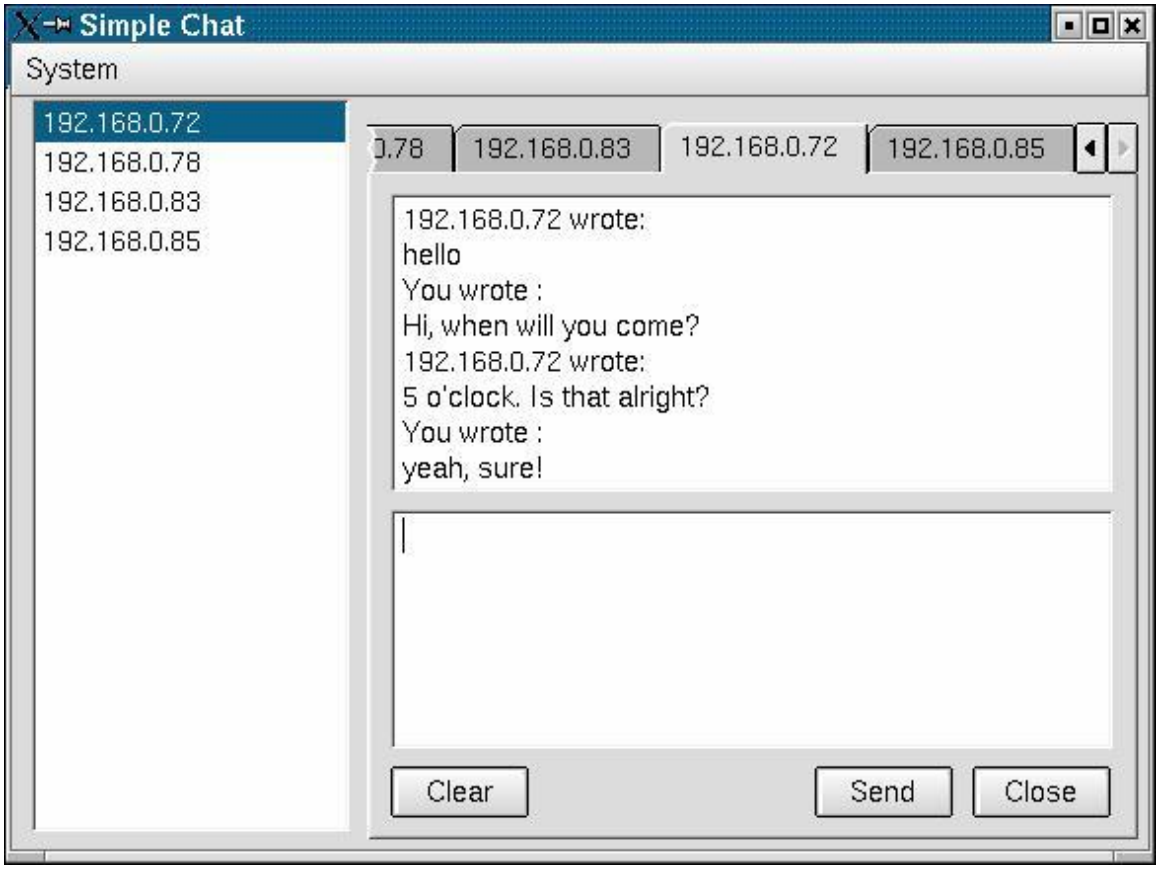

Рис. 2. Пример работы *Linux* – клиента

Клиентская часть *Linux*-приложения демонстрирует работу демона и является его логическим дополнением. Она позволяет пользователю посылать и принимать сообщения из сети, хранить список контактов, назначать имена пользователей.

Пользовательский интерфейс программы написан с применением библиотеки *Qt*. Он весьма схож с популярным *ICQ* клиентом под *Linux LICQ*. Поэтому освоение данной программы не должно составить затруднений для *Linux*-пользователей. Интерфейс программы интуитивно понятен и прост в использовании.

4. *Windows* – приложение

Windows-приложение написано традиционным путем – без применения конечных автоматов. Оно представляет собой обычное *MFC*-приложение, обеспечивающее те же функции, что и *Linux*-версия. Приложение, в частности, позволяет принимать и отсылать сообщения в локальной сети, хранить одновременно несколько контактов, назначать имена новым пользователям с целью упростить поиск нужного пользователя.

На рис. 3 изображен пример работы *Windows* – приложения.

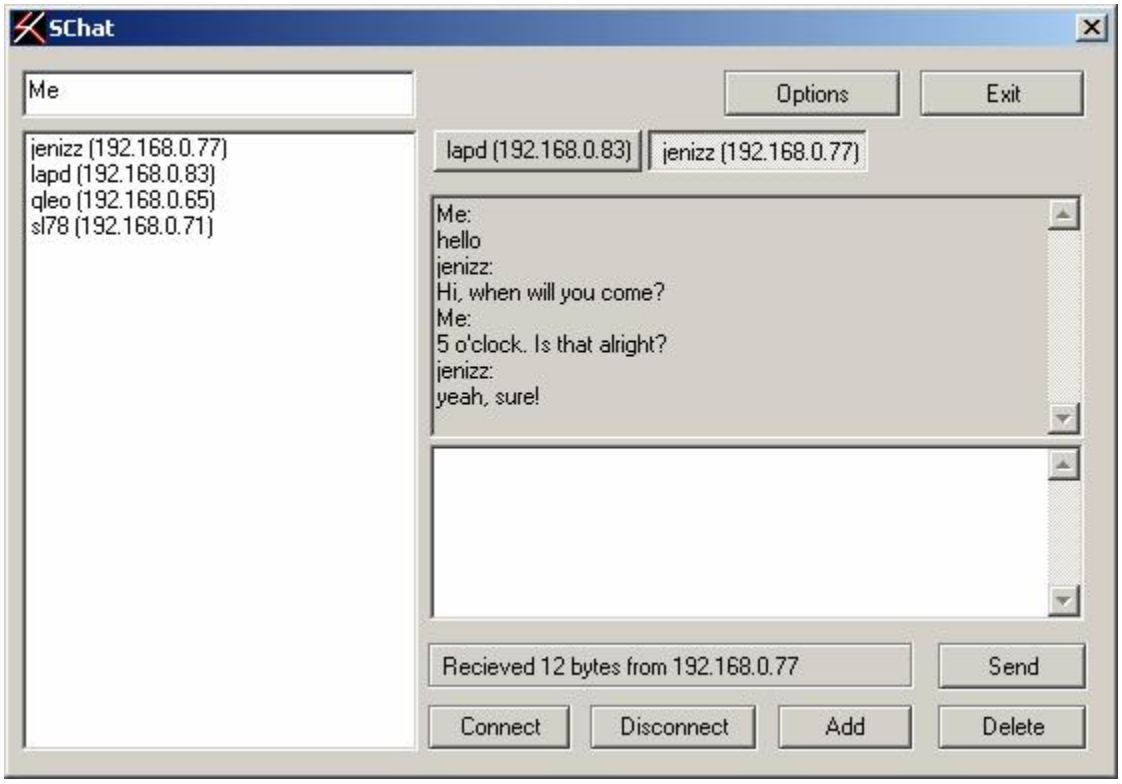

Рис. 3. Пример работы *Windows* – приложения

В левой части окна находится контакт-лист, каждая запись которого представляет собой пару “имя пользователя – его фактический IP-адрес”. Над контакт-листом находится поле, в котором можно задать собственное имя, отображаемое в диалоге, с целью придания ему читаемого вида. В правой части окна находится поле диалога и элементы управления соединениями и контакт-листом.

В процессе написания этой программы возникли сложности с отладкой сетевого соединения, характерные для традиционного подхода.

Заключение

В результате выполненой работы можно сделать следующие выводы.

1. Использование конечных автоматов при проектировании позволило описать поведение серверной части системы наглядно.
2. Существенно упростился процесс отладки программы.

Литература

1. *Шалыто А.А.* SWITCH-технология. Алгоритмизация и программирование задач логического управления. СПб.: Наука, 1998.
2. *Керниган Б., Ричи Д.* Язык Программирования Си. СПб.: Невский диалект, 2001.
3. *Страуструп Б.* Язык программирования C++. СПб.: Невский дилект, 2001
4. *Робачевский А.М.* Операционная система UNIX. СПб.: БХВ, 1999.

Приложение 1. Пример логов демона

```
Демон Запущен

Ждем соединения с клиентом... готово

A0 перешел в состояние 0   :   Ожидание сообщения

Событие e1                 :   В сетевом сокете появилось сообщение
A0 перешел в состояние 4   :   Получение сообщения из сетевого сокета

Событие e10                :   Успешный прием сообщения из сети
A0 перешел в состояние 5   :   Конверование сообщения во внутренний формат

Событие e11                :   Успешное преобразование в локальный формат
A0 перешел в состояние 6   :   Отправка сообщения через локальный сокет
    сообщение = 'hello'
    адрес отправителя  192.168.0.72

Событие e12                :   Успешная отправка локального сообщения
A0 перешел в состояние 4   :   Получение сообщения из сетевого сокета

Событие e3                 :   В сетевом сокете сообщений не осталось
A0 перешел в состояние 0   :   Ожидание сообщения

Событие e0                 :   В локальном сокете появилось сообщение
A0 перешел в состояние 1   :   Получение сообщения из локального сокета

Событие e7                 :   Успешный прием локального сообщения
A0 перешел в состояние 2   :   Конверование сообщения в сетевой формат

Событие e8                 :   Успешное преобразование в сетевой формат
A0 перешел в состояние 3   :   Отправка сообщения через сетевой сокет
    сообщение = 'Hi, when will you come?'
    адрес назначения = 192.168.0.72

Событие e9                 :   Успешная отправка сообщения в сеть
A0 перешел в состояние 1   :   Получение сообщения из локального сокета

Событие e2                 :   В локальном сокете сообщений не осталось
A0 перешел в состояние 0   :   Ожидание сообщения

Событие e1                 :   В сетевом сокете появилось сообщение
A0 перешел в состояние 4   :   Получение сообщения из сетевого сокета

Событие e10                :   Успешный прием сообщения из сети
A0 перешел в состояние 5   :   Конверование сообщения во внутренний формат

Событие e11                :   Успешное преобразование в локальный формат
A0 перешел в состояние 6   :   Отправка сообщения через локальный сокет
    сообщение = '5 o'clock. Is that alright?'
    адрес отправителя  192.168.0.72

Событие e12                :   Успешная отправка локального сообщения
A0 перешел в состояние 4   :   Получение сообщения из сетевого сокета

Событие e3                 :   В сетевом сокете сообщений не осталось
A0 перешел в состояние 0   :   Ожидание сообщения

Событие e0                 :   В локальном сокете появилось сообщение
```

```
A0 перешел в состояние 1   :   Получение сообщения из локального сокета

Событие e7                 :   Успешный прием локального сообщения
A0 перешел в состояние 2   :   Конверование сообщения в сетевой формат

Событие e8                 :   Успешное преобразование в сетевой формат
A0 перешел в состояние 3   :   Отправка сообщения через сетевой сокет
    сообщение = 'yeah, sure!'
    адрес назначения = 192.168.0.72

Событие e9                 :   Успешная отправка сообщения в сеть
A0 перешел в состояние 1   :   Получение сообщения из локального сокета

Событие e2                 :   В локальном сокете сообщений не осталось
A0 перешел в состояние 0   :   Ожидание сообщения

Событие e1                 :   В сетевом сокете появилось сообщение
A0 перешел в состояние 4   :   Получение сообщения из сетевого сокета

Событие e10                :   Успешный прием сообщения из сети
A0 перешел в состояние 5   :   Конверование сообщения во внутренний формат

Событие e11                :   Успешное преобразование в локальный формат
A0 перешел в состояние 6   :   Отправка сообщения через локальный сокет
    сообщение = 'ok. bye then...'
    адрес отправителя  192.168.0.72

Событие e12                :   Успешная отправка локального сообщения
A0 перешел в состояние 4   :   Получение сообщения из сетевого сокета

Событие e3                 :   В сетевом сокете сообщений не осталось
A0 перешел в состояние 0   :   Ожидание сообщения

Событие e0                 :   В локальном сокете появилось сообщение
A0 перешел в состояние 1   :   Получение сообщения из локального сокета

Событие e7                 :   Успешный прием локального сообщения
A0 перешел в состояние 2   :   Конверование сообщения в сетевой формат

Событие e8                 :   Успешное преобразование в сетевой формат
A0 перешел в состояние 3   :   Отправка сообщения через сетевой сокет
    сообщение = 'bye'
    адрес назначения = 192.168.0.72

Событие e9                 :   Успешная отправка сообщения в сеть
A0 перешел в состояние 1   :   Получение сообщения из локального сокета

Событие e2                 :   В локальном сокете сообщений не осталось
A0 перешел в состояние 0   :   Ожидание сообщения

Событие e6                 :   Завершение работы клиента
Демон закончил работу
```

Приложение 2. Исходный код *Windows* - приложения

StdAfx.h

```
// stdafx.h : include file for standard system include files,
// or project specific include files that are used frequently, but
// are changed infrequently
//

#if !defined(AFX_STDAFX_H__79D3467E_C0FC_4CEE_9E7B_78FDC23EDB3C__INCLUDED_)
#define AFX_STDAFX_H__79D3467E_C0FC_4CEE_9E7B_78FDC23EDB3C__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#define VC_EXTRALEAN            // Exclude rarely-used stuff from Windows
headers

#include <afxwin.h>         // MFC core and standard components
#include <afxext.h>         // MFC extensions
#include <afxdtctl.h>           // MFC support for Internet Explorer 4 Common
Controls
#ifndef _AFX_NO_AFXCMN_SUPPORT
#include <afxcmn.h>             // MFC support for Windows Common Controls
#endif // _AFX_NO_AFXCMN_SUPPORT

#include <afxsock.h>            // MFC socket extensions

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before
the previous line.

#endif //
!defined(AFX_STDAFX_H__79D3467E_C0FC_4CEE_9E7B_78FDC23EDB3C__INCLUDED_)
```

SChatDlg.h

```
#if !defined(AFX_SCHATDLG_H__0446EC25_3612_4A69_AE67_67D9F1C92F27__INCLUDED_)
#define AFX_SCHATDLG_H__0446EC25_3612_4A69_AE67_67D9F1C92F27__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

/////////////////////////////////////////////////////////////////////////////
// CSChatDlg dialog

#include "ChatTabCtrl.h"
#include "Contact.h"
#include "ContactArray.h"


class SCSocket : public CSocket
{
friend class CSChatDlg;
public:
	SCSocket(CSChatDlg *dlg);
	virtual BOOL OnMessagePending();
protected:
```

```
        CSChatDlg *chatDlg;
        virtual void OnReceive(int nErrorCode);

};

class CSChatDlg : public CDialog
{
// Construction
public:
        int cursel;
        int numTabs;
        int newnum;
        CSChatDlg(CWnd* pParent = NULL);  // standard constructor
         void A0( int event );
        ContactArray *contactarray;
        void OnCancel();
        void DeleteTab( int tn );
        void ChangeTab();
        void UpdateContactList();
        int Decompose
        ( CString str, BYTE *_ip1, BYTE *_ip2, BYTE *_ip3, BYTE *_ip4 );
    // Dialog Data
        //{{AFX_DATA(CSChatDlg)
        enum { IDD = IDD_SCHAT_DIALOG };
        CEdit  myName;
        CStatic      statusText;
        CButton      sendButton;
        CEdit  unsentEdit;
        CEdit  histEdit;
        CChatTabCtrl  TalkTabs;
        CListBox     ContactList;
        //}}AFX_DATA

        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CSChatDlg)
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);     // DDX/DDV support
        //}}AFX_VIRTUAL

        // Implementation
protected:
        HICON m_hIcon;
         SCSocket *sock;
        // Generated message map functions
        //{{AFX_MSG(CSChatDlg)
        virtual BOOL OnInitDialog();
        afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
        afx_msg void OnPaint();
        afx_msg HCURSOR OnQueryDragIcon();
        afx_msg void OnSelchangeTab1(NMHDR* pNMHDR, LRESULT* pResult);
        afx_msg void OnSelchangeList1();
        virtual void OnOK();
        afx_msg void OnDblclkContactList();
        afx_msg void OnButton2();
        afx_msg void OnButtonExit();
        afx_msg void OnButtonDisconnect();
        afx_msg void OnSelcancelContactList();
        afx_msg void OnButtonDelete();
        afx_msg void OnButtonSend();
        afx_msg void OnConnect();
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};
```

```
//CSChatDlg cdlg;

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before
the previous line.

#endif //
!defined(AFX_SCHATDLG_H__0446EC25_3612_4A69_AE67_67D9F1C92F27__INCLUDED_)
```

SChat.h

```
// SChat.h : main header file for the SCHAT application
//

#if !defined(AFX_SCHAT_H__D00D7BBB_1054_40C6_83DB_7A19FACB9AD7__INCLUDED_)
#define AFX_SCHAT_H__D00D7BBB_1054_40C6_83DB_7A19FACB9AD7__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#ifndef __AFXWIN_H__
	#error include 'stdafx.h' before including this file for PCH
#endif

#include "resource.h"		// main symbols
#include "afxsock.h"

/////////////////////////////////////////////////////////////////////////////
// CSChatApp:
// See SChat.cpp for the implementation of this class
//

class CSChatApp : public CWinApp
{
public:
	CSChatApp();

// Overrides
	// ClassWizard generated virtual function overrides
	//{{AFX_VIRTUAL(CSChatApp)
	public:
	virtual BOOL InitInstance();
	//}}AFX_VIRTUAL

// Implementation

	//{{AFX_MSG(CSChatApp)
	// NOTE - the ClassWizard will add and remove member functions here.
	// DO NOT EDIT what you see in these blocks of generated code !
	//}}AFX_MSG
	DECLARE_MESSAGE_MAP()
};


/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before
the previous line.
```

```
#endif //
!defined(AFX_SCHAT_H__D00D7BBB_1054_40C6_83DB_7A19FACB9AD7__INCLUDED_)
```

resource.h

```
//{{NO_DEPENDENCIES}}
// Microsoft Developer Studio generated include file.
// Used by SChat.rc
//
#define IDM_ABOUTBOX                    0x0010
#define IDD_ABOUTBOX                    100
#define IDS_ABOUTBOX                    101
#define IDD_SCHAT_DIALOG                102
#define IDP_SOCKETS_INIT_FAILED         103
#define IDR_MAINFRAME                   128
#define IDD_ADDDIALOG                   129
#define IDC_LIST1                       1000
#define IDC_TAB1                        1001
#define IDC_BUTTON1                     1002
#define IDC_BUTTON2                     1003
#define IDC_IPADDRESS1                  1004
#define IDC_BUTTON3                     1005
#define IDC_BUTTON4                     1006
#define IDC_EDIT1                       1013
#define IDC_EDIT2                       1014
#define IDC_EDIT3                       1015
#define IDC_BUTTON5                     1016
#define IDC_BUTTON6                     1017
#define IDC_BUTTON7                     1018
#define IDC_STATIC01                    1019
#define IDC_STATUS                      1020

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE        133
#define _APS_NEXT_COMMAND_VALUE         32771
#define _APS_NEXT_CONTROL_VALUE         1021
#define _APS_NEXT_SYMED_VALUE           101
#endif
#endif
```

ContactArray.h

```
// ContactArray.h: interface for the ContactArray class.
//
//////////////////////////////////////////////////////////////////////

#if
!defined(AFX_CONTACTARRAY_H__EE495AE6_57D7_41AE_B286_B7B06C174304__INCLUDED_)
#define AFX_CONTACTARRAY_H__EE495AE6_57D7_41AE_B286_B7B06C174304__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#include "Contact.h"

class ContactArray
```

```
{
public:

	Contact *contacts[100];
	int filled[100];

	int AddContact( BYTE _ip1, BYTE _ip2, BYTE _ip3, BYTE _ip4, CString name
);
	 int DeleteContact( int cNum );
	 int GetTabNumber( int cNum );
	void DeleteTab( int tNum );
	int FindByTab( int tNum );
	CString GetName( int cNum );
	CString GetIP( int cNum );
	int SetNewName( int cNum, CString name );
	int CheckIfPresents( BYTE _ip1, BYTE _ip2, BYTE _ip3, BYTE _ip4 );
	int CheckIfPresents( CString addr );
	int CheckIfPresents( int i );
	ContactArray();
	virtual ~ContactArray();

};

#endif //
!defined(AFX_CONTACTARRAY_H__EE495AE6_57D7_41AE_B286_B7B06C174304__INCLUDED_)
```

Contact.h

```
// Contact.h: interface for the Contact class.
//
//////////////////////////////////////////////////////////////////////

#if !defined(AFX_CONTACT_H__3963EBC3_ECE3_4606_A6EB_10CE92199D29__INCLUDED_)
#define AFX_CONTACT_H__3963EBC3_ECE3_4606_A6EB_10CE92199D29__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

class Contact
{
public:

	CString cName, cIP, cHistory, cInput, cUnsent;
	int tabNumber;
	BYTE ip1, ip2, ip3, ip4;

	Contact();
	 Contact(BYTE _ip1, BYTE _ip2, BYTE _ip3, BYTE _ip4, CString name);
	virtual ~Contact();

};

#endif //
!defined(AFX_CONTACT_H__3963EBC3_ECE3_4606_A6EB_10CE92199D29__INCLUDED_)
```

ChatTabCtrl.h

```
#if
!defined(AFX_CHATTABCTRL_H__7DA8A523_CFF4_4E78_8DBF_DC4810974CFE__INCLUDED_)
#define AFX_CHATTABCTRL_H__7DA8A523_CFF4_4E78_8DBF_DC4810974CFE__INCLUDED_
```

```
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// ChatTabCtrl.h : header file
//

/////////////////////////////////////////////////////////////////////////////
// CChatTabCtrl window

class CChatTabCtrl : public CTabCtrl
{
// Construction
public:

	int tabCurrent;
	int pagesNumber;

// Attributes
public:

// Operations
public:

	void Init();
	void SetRectangle();

// Overrides
	// ClassWizard generated virtual function overrides
	//{{AFX_VIRTUAL(CChatTabCtrl)
	//}}AFX_VIRTUAL

// Implementation
public:
	virtual ~CChatTabCtrl();

	// Generated message map functions
protected:
	//{{AFX_MSG(CChatTabCtrl)
	// NOTE - the ClassWizard will add and remove member functions here.
	//}}AFX_MSG

	DECLARE_MESSAGE_MAP()
};

/////////////////////////////////////////////////////////////////////////////

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before
the previous line.

#endif //
!defined(AFX_CHATTABCTRL_H__7DA8A523_CFF4_4E78_8DBF_DC4810974CFE__INCLUDED_)
```

AddDlg.h

```
#if !defined(AFX_ADDDLG_H__AB032604_5A15_4544_B458_1889595E4D01__INCLUDED_)
#define AFX_ADDDLG_H__AB032604_5A15_4544_B458_1889595E4D01__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
```

```
// AddDlg.h : header file
//
#include "SChatDlg.h"
/////////////////////////////////////////////////////////////////////////////
// CAddDlg dialog

class CAddDlg : public CDialog
{
// Construction
public:
	CAddDlg(CWnd* pParent = NULL);   // standard constructor

	void GetIp( BYTE& _ip1, BYTE& _ip2, BYTE& _ip3, BYTE& _ip4 );
	char* GetName();
	void SetOldIP( CString name );
	char nip[20];
	char nname[50];
	int validIP;
	int ipSet;
	BYTE ip1, ip2, ip3, ip4;
// Dialog Data
	//{{AFX_DATA(CAddDlg)
	enum { IDD = IDD_ADDDIALOG };
	CEdit  newName;
	CIPAddressCtrl	IpAddress;
	//}}AFX_DATA


// Overrides
	// ClassWizard generated virtual function overrides
	//{{AFX_VIRTUAL(CAddDlg)
	protected:
	virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
	//}}AFX_VIRTUAL

// Implementation
protected:

	// Generated message map functions
	//{{AFX_MSG(CAddDlg)
	virtual void OnOK();
	afx_msg void OnFieldchangedIpaddress1(NMHDR* pNMHDR, LRESULT* pResult);
	afx_msg void OnShowWindow(BOOL bShow, UINT nStatus);
	afx_msg void OnCancelMode();
	afx_msg void OnCaptureChanged(CWnd *pWnd);
	//}}AFX_MSG
	DECLARE_MESSAGE_MAP()
};

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before
the previous line.

#endif //
!defined(AFX_ADDDLG_H__AB032604_5A15_4544_B458_1889595E4D01__INCLUDED_)
```

StdAfx.cpp

```
// stdafx.cpp : source file that includes just the standard includes
//     SChat.pch will be the pre-compiled header
//     stdafx.obj will contain the pre-compiled type information
```

```
#include "stdafx.h"
```

SChatDlg.cpp

```
// SChatDlg.cpp : implementation file
//

#include "stdafx.h"
#include "SChat.h"
#include "SChatDlg.h"
#include "AddDlg.h"
#include "hatTab.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

SCSocket::SCSocket(CSChatDlg *dlg)
{
       CSocket::CSocket();
       chatDlg = dlg;
}

BOOL SCSocket::OnMessagePending(){
       MessageBoxEx(NULL, "true", "status", MB_OK, 0);
       return TRUE;
}

void SCSocket::OnReceive(int nErrorCode)       // CMyAsyncSocket is
                                               // derived from CAsyncSocket
{

    TCHAR buff[4096];
    BYTE a1, a2, a3, a4;
    int nRead;
    CString str;
    UINT port;
    nRead = ReceiveFrom( buff, 4096, str, port );

    switch (nRead)
    {
    case 0:
         break;
    case SOCKET_ERROR:
         break;
       default:
           buff[nRead] = 0;
        CString oldhist;
        char *string = ( char* ) malloc( 1000 );
             sprintf( string, "Recieved %d bytes from " + str, nRead );
             CString status( string );
             chatDlg->statusText.SetWindowText( status );
             chatDlg->Decompose( str, &a1, &a2, &a3, &a4 );
             char *naddr = ( char* )malloc( 1000 );
             int cn;
             int pres = chatDlg->contactarray->CheckIfPresents( str );
             if( pres != -1 ) {
```

```
                    if( chatDlg->contactarray->contacts[pres]->tabNumber != -1
) {
                        cn = pres;
                           int tn = chatDlg->contactarray->contacts[pres]-
>tabNumber;
                           chatDlg->TalkTabs.SetCurSel( tn );
                    } else {
                           cn = pres;
                           CString itemText = chatDlg->contactarray-
>contacts[cn]->cName +
                                  " (" + chatDlg->contactarray->contacts[cn]-
>cIP + ")";
                           chatDlg->TalkTabs.InsertItem( chatDlg->numTabs,
itemText );
                           chatDlg->contactarray->contacts[cn]->tabNumber =
chatDlg->numTabs;
                           chatDlg->TalkTabs.SetCurSel( chatDlg->numTabs );
                           chatDlg->numTabs++;
                    }
             } else {
                    cn = chatDlg->contactarray->AddContact( a1, a2, a3, a4,
"New contact" );
                    CString itemText = "New contact (" + str + ")";
                    chatDlg->TalkTabs.InsertItem( chatDlg->numTabs, itemText );
                chatDlg->contactarray->contacts[cn]->tabNumber = chatDlg-
>numTabs;
                    chatDlg->TalkTabs.SetCurSel( chatDlg->numTabs );
                    chatDlg->numTabs++;
             }
             CString newmess( buff );
             chatDlg->histEdit.GetWindowText( oldhist );
             if( oldhist != "" ) oldhist += "\r\n";
             oldhist += chatDlg->contactarray->contacts[cn]->cName + ":\r\n";
             oldhist += newmess;
             chatDlg->histEdit.SetWindowText( oldhist );
             chatDlg->UpdateContactList();
             chatDlg->ChangeTab();
       }
    chatDlg->histEdit.ScrollWindow( 0, chatDlg->histEdit.GetScrollLimit(
SB_VERT ) );
       CSocket::OnReceive(nErrorCode);
}

int CSChatDlg::Decompose( CString str, BYTE *_ip1, BYTE *_ip2, BYTE *_ip3, BYTE
*_ip4 ) {

       return sscanf( str, "%d.%d.%d.%d", _ip1, _ip2, _ip3, _ip4 );
}



class CAboutDlg : public CDialog
{
public:
       CAboutDlg();

    // Dialog Data
       //{{AFX_DATA(CAboutDlg)
       enum { IDD = IDD_ABOUTBOX };
       //}}AFX_DATA

       // ClassWizard generated virtual function overrides
       //{{AFX_VIRTUAL(CAboutDlg)
```

```
	protected:
	virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
	//}}AFX_VIRTUAL

// Implementation
protected:
	//{{AFX_MSG(CAboutDlg)
	//}}AFX_MSG
	DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
	//{{AFX_DATA_INIT(CAboutDlg)
	//}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CAboutDlg)
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
	//{{AFX_MSG_MAP(CAboutDlg)
		// No message handlers
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSChatDlg dialog

CSChatDlg::CSChatDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CSChatDlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CSChatDlg)
	//}}AFX_DATA_INIT
	// Note that LoadIcon does not require a subsequent DestroyIcon in Win32
	sock = new SCSocket(this);
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
    sock->Create(3538, SOCK_DGRAM );
	contactarray = new ContactArray();
	numTabs = 0;
	newnum = -1;
	cursel = -1;
}

void CSChatDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CSChatDlg)
	DDX_Control(pDX, IDC_EDIT1, myName);
	DDX_Control(pDX, IDC_STATUS, statusText);
	DDX_Control(pDX, IDC_BUTTON5, sendButton);
	DDX_Control(pDX, IDC_EDIT3, unsentEdit);
	DDX_Control(pDX, IDC_EDIT2, histEdit);
	DDX_Control(pDX, IDC_TAB1, TalkTabs);
	DDX_Control(pDX, IDC_LIST1, ContactList);
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CSChatDlg, CDialog)
```

```
	//{{AFX_MSG_MAP(CSChatDlg)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_NOTIFY(TCN_SELCHANGE, IDC_TAB1, OnSelchangeTab1)
	ON_LBN_SELCHANGE(IDC_LIST1, OnSelchangeList1)
	ON_LBN_DBLCLK(IDC_LIST1, OnDblclkContactList)
	ON_BN_CLICKED(IDC_BUTTON2, OnButton2)
	ON_BN_CLICKED(IDC_BUTTON6, OnButtonExit)
	ON_BN_CLICKED(IDC_BUTTON1, OnButtonDisconnect)
	ON_LBN_SELCANCEL(IDC_LIST1, OnSelcancelContactList)
	ON_BN_CLICKED(IDC_BUTTON4, OnButtonDelete)
	ON_BN_CLICKED(IDC_BUTTON5, OnButtonSend)
	ON_BN_CLICKED(IDC_BUTTON7, OnConnect)
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSChatDlg message handlers

BOOL CSChatDlg::OnInitDialog() {
	CDialog::OnInitDialog();

	// Add "About..." menu item to system menu.

	// IDM_ABOUTBOX must be in the system command range.
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);

	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL) {
		CString strAboutMenu;
		strAboutMenu.LoadString(IDS_ABOUTBOX);
		if (!strAboutMenu.IsEmpty()) {
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX,
strAboutMenu);
		}
	}

	// Set the icon for this dialog.  The framework does this automatically
	//  when the application's main window is not a dialog
	SetIcon(m_hIcon, TRUE);			// Set big icon
	SetIcon(m_hIcon, FALSE);		// Set small icon

	//	TalkTabs.InsertItem(0, _T("ta"));

	TalkTabs.Init();
	sendButton.BringWindowToTop();
	histEdit.BringWindowToTop();
	unsentEdit.BringWindowToTop();
	myName.SetWindowText( "Me" );
	return TRUE;  // return TRUE  unless you set the focus to a control
}


void CSChatDlg::OnSysCommand(UINT nID, LPARAM lParam) {
	if ((nID & 0xFFF0) == IDM_ABOUTBOX)	{
		CAboutDlg dlgAbout;
		dlgAbout.DoModal();
	} else {
		CDialog::OnSysCommand(nID, lParam);
	}
```

```
}

// If you add a minimize button to your dialog, you will need the code below
//  to draw the icon.  For MFC applications using the document/view model,
//  this is automatically done for you by the framework.

void CSChatDlg::OnPaint() {

	if (IsIconic())	{
		CPaintDC dc(this); // device context for painting

		SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);

		// Center icon in client rectangle
		int cxIcon = GetSystemMetrics(SM_CXICON);
		int cyIcon = GetSystemMetrics(SM_CYICON);
		CRect rect;
		GetClientRect(&rect);
		int x = (rect.Width() - cxIcon + 1) / 2;
		int y = (rect.Height() - cyIcon + 1) / 2;

		// Draw the icon
		dc.DrawIcon(x, y, m_hIcon);

	} else {
		CDialog::OnPaint();
	}
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CSChatDlg::OnQueryDragIcon() {

	return (HCURSOR) m_hIcon;
}

void CSChatDlg::ChangeTab() {

	int current = TalkTabs.GetCurSel();
	CString str;
	if( cursel != -1 ) {
		int cNum = contactarray->FindByTab( cursel );
		if( cNum != -1 ) {
			histEdit.GetWindowText( str );
			contactarray->contacts[cNum]->cHistory = str;
			unsentEdit.GetWindowText( str );
			contactarray->contacts[cNum]->cUnsent = str;
		}
	}
	if( current != -1 ) {
		int cNum = contactarray->FindByTab( current );
		if( cNum != -1 ) {
			str = contactarray->contacts[cNum]->cHistory;
			histEdit.SetWindowText( str );
			str = contactarray->contacts[cNum]->cUnsent;
			unsentEdit.SetWindowText( str );
			sendButton.ShowWindow( SW_SHOW );
			histEdit.ShowWindow( SW_SHOW );
			unsentEdit.ShowWindow( SW_SHOW );
		}
	} else {
		histEdit.SetWindowText( "" );
		unsentEdit.SetWindowText( "" );
```

```
		sendButton.ShowWindow( SW_HIDE );
		histEdit.ShowWindow( SW_HIDE );
		unsentEdit.ShowWindow( SW_HIDE );
	}
	cursel = current;
}

void CSChatDlg::OnSelchangeTab1(NMHDR* pNMHDR, LRESULT* pResult) {

	*pResult = 0;
	ChangeTab();
}

void CSChatDlg::OnSelchangeList1() {

}

void CSChatDlg::UpdateContactList() {

	ContactList.ResetContent();
	int k = 0;
	int l = 0;
	for( int i = 0; i < 100; i++ ) {
		if( contactarray->CheckIfPresents( i ) == 1 ) {
			CString ni = contactarray->GetIP( i );
			CString nn = contactarray->GetName( i );
			k = ContactList.AddString( nn + " (" + ni + ")" );
			ContactList.SetItemData( k, ( DWORD )i );
		}
	}
	if( newnum > 0 ) {
		for( i = 0; i < ContactList.GetCount(); i++ ) {
			int u = ( int )ContactList.GetItemData( i );
			if( u == newnum ) {
				ContactList.SetCurSel( i );
			}
		}
		newnum = -1;
	}
}

void CSChatDlg::OnDblclkContactList() {

	int selNum = ContactList.GetCurSel();
	if( selNum == LB_ERR ) return;
	CString itemText;
	ContactList.GetText( selNum, itemText );
	DWORD tn = ContactList.GetItemData( selNum );
	int tnum = contactarray->contacts[tn]->tabNumber;

	TCITEM item;
	BOOL res = TalkTabs.GetItem( tnum, &item );
	if( res == FALSE ) {
	    TalkTabs.InsertItem( numTabs, itemText );
		contactarray->contacts[tn]->tabNumber = numTabs;
		TalkTabs.SetCurSel( numTabs );
		numTabs++;
		ChangeTab();
	} else {
		TalkTabs.SetCurSel( tnum );
		ChangeTab();
	}
}
```

```
void CSChatDlg::OnOK() {

	OnDblclkContactList();
}

void CSChatDlg::OnCancel() {

	CDialog::ShowWindow( SW_MINIMIZE );
}


void CSChatDlg::OnButton2() { // "Add"

	CAddDlg adddlg;
	adddlg.validIP = 0;
	int i = ContactList.GetCurSel();
	BYTE ip1, ip2, ip3, ip4;
	if( i != LB_ERR ) {
		int k = ( int )ContactList.GetItemData( i );
		adddlg.ip1 = contactarray->contacts[k]->ip1;
		adddlg.ip2 = contactarray->contacts[k]->ip2;
		adddlg.ip3 = contactarray->contacts[k]->ip3;
		adddlg.ip4 = contactarray->contacts[k]->ip4;
		adddlg.ipSet = 1;
	} else adddlg.ipSet = 0;
	adddlg.DoModal();
	if( adddlg.validIP == 1 ) {
		adddlg.GetIp( ip1, ip2, ip3, ip4);
		CString newName = CString( adddlg.GetName() );
		if( newName == "" ) {
			newName = "Untitled";
		}
		int num = contactarray->CheckIfPresents( ip1, ip2, ip3, ip4 );
		if( num == -1 ) {
			newnum = contactarray->AddContact( ip1, ip2, ip3, ip4,
newName );
			UpdateContactList();
		} else {
			contactarray->SetNewName( num, newName );
			int cn = num;
			int tn = contactarray->contacts[num]->tabNumber;
			DeleteTab( tn );
			CString itemText = contactarray->contacts[cn]->cName +
				" (" + contactarray->contacts[cn]->cIP + ")";
			TalkTabs.InsertItem( numTabs, itemText );
			contactarray->contacts[cn]->tabNumber = numTabs;
			TalkTabs.SetCurSel( numTabs );
			numTabs++;
			ChangeTab();
			newnum = num;
			UpdateContactList();
		}
	}
}


void CSChatDlg::OnButtonExit() {

	CDialog::OnCancel();
}

void CSChatDlg::DeleteTab( int tn ) {
```

```
	if( TalkTabs.DeleteItem( tn ) == TRUE ) {
		contactarray->DeleteTab( tn );
		if( cursel == tn ) cursel = -1;
		if( cursel > tn ) cursel--;
		ChangeTab();
		numTabs--;
	}
}

void CSChatDlg::OnButtonDisconnect() {

	int tn = TalkTabs.GetCurSel();
	DeleteTab( tn );

}

void CSChatDlg::OnSelcancelContactList() {

	ContactList.SetCurSel( -1 );
}

void CSChatDlg::OnButtonDelete() {

	int i = ContactList.GetCurSel();
	if( i != LB_ERR ) {
		int j = ( int )ContactList.GetItemData( i );
		DeleteTab( contactarray->contacts[j]->tabNumber );
		contactarray->DeleteContact( j );
		UpdateContactList();
	}
}

void CSChatDlg::OnButtonSend() {

	CString str, str2;
	unsentEdit.GetWindowText( str );
	histEdit.GetWindowText( str2 );
	if( str2 != "" ) str2 += "\r\n";
	CString mn;
	myName.GetWindowText( mn );
    if( mn = "" ) mn = "Me";
	str2 += mn + ":\r\n";
	histEdit.SetWindowText( str2+str );
	int tn = TalkTabs.GetCurSel();
	int cn = contactarray->FindByTab( tn );
	CString rAddr = contactarray->contacts[cn]->cIP;
	int res = sock->SendTo( str, str.GetLength(), 3538, rAddr );

	char *_status = ( char* ) malloc( 1000 );
	sprintf( _status, "Sent %d bytes to " + rAddr , res );
	CString status( _status );
	statusText.SetWindowText( status );

	unsentEdit.SetWindowText( "" );
	histEdit.LineScroll( histEdit.GetLineCount()  - 8 -
histEdit.GetFirstVisibleLine() );
}

void CSChatDlg::OnConnect() {

	OnDblclkContactList();
}
```

SChat.cpp

```
// SChat.cpp : Defines the class behaviors for the application.
//

#include "stdafx.h"
#include "SChat.h"
#include "SChatDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif



/////////////////////////////////////////////////////////////////////////////
// CSChatApp

BEGIN_MESSAGE_MAP(CSChatApp, CWinApp)
	//{{AFX_MSG_MAP(CSChatApp)
		// NOTE - the ClassWizard will add and remove mapping macros here.
		//    DO NOT EDIT what you see in these blocks of generated code!
	//}}AFX_MSG
	ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CSChatApp construction

CSChatApp::CSChatApp()
{
	// TODO: add construction code here,
	// Place all significant initialization in InitInstance
}

/////////////////////////////////////////////////////////////////////////////
// The one and only CSChatApp object

CSChatApp theApp;

/////////////////////////////////////////////////////////////////////////////
// CSChatApp initialization

BOOL CSChatApp::InitInstance()
{
	if (!AfxSocketInit())
	{
		AfxMessageBox(IDP_SOCKETS_INIT_FAILED);
		return FALSE;
	}

	// Standard initialization
	// If you are not using these features and wish to reduce the size
	//  of your final executable, you should remove from the following
	//  the specific initialization routines you do not need.

#ifdef _AFXDLL
	Enable3dControls();			// Call this when using MFC in a shared
DLL
#else
	Enable3dControlsStatic();  // Call this when linking to MFC statically
```

```
#endif

	CSChatDlg dlg;

	m_pMainWnd = &dlg;
	int nResponse = dlg.DoModal();
	if (nResponse == IDOK)
	{
		// TODO: Place code here to handle when the dialog is
		//  dismissed with OK
	}
	else if (nResponse == IDCANCEL)
	{
		// TODO: Place code here to handle when the dialog is
		//  dismissed with Cancel
	}

	// Since the dialog has been closed, return FALSE so that we exit the
	//  application, rather than start the application's message pump.
	return FALSE;
}
```

ContactArray.cpp

```
// ContactArray.cpp: implementation of the ContactArray class.
//
//////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "SChat.h"
#include "ContactArray.h"
#include "Contact.h"

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////

ContactArray::ContactArray() {

	for( int i = 0; i < 100; i++ ) filled[i] = 0;
}

ContactArray::~ContactArray() {

	for( int i = 0; i < 100; i++ )
	{
		if( filled[i] == 1 ) {
			delete contacts[i];
		}
	}
}

int ContactArray::AddContact( BYTE _ip1, BYTE _ip2, BYTE _ip3, BYTE _ip4,
CString name ) {

	for( int i = 0; i < 100; i++ ) if( filled[i] != 1 ) break;
```

```
	if( i > 100 ) {
		return -1;
	}
	contacts[i] = new Contact( _ip1, _ip2, _ip3, _ip4, name );
	filled[i] = 1;
	return i;
}

int ContactArray::DeleteContact( int cNum ) {

	if( filled[cNum] != 1 )	{
		return -1;
	}
	delete contacts[cNum];
	filled[cNum] = 0;
	return 0;
}
CString ContactArray::GetName( int cNum ) {

	if( filled[cNum] == 1 )	{
		return contacts[cNum]->cName;
	} else {
		return "";
	}
}

CString ContactArray::GetIP( int cNum ) {

	if( filled[cNum] == 1 )	{
		return contacts[cNum]->cIP;
	} else {
		return "";
	}
}

int ContactArray::SetNewName( int cNum, CString name ) {

	if( filled[cNum] ) {
		contacts[cNum]->cName = name;
		return 1;
	} else {
		return -1;
	}
}

int ContactArray::CheckIfPresents( BYTE _ip1, BYTE _ip2, BYTE _ip3, BYTE _ip4 )
{

	for( int i = 0; i < 100; i++ ) {
		if( filled[i] == 1 ) {
			if( contacts[i]->ip1 == _ip1 && contacts[i]->ip2 == _ip2 &&
				contacts[i]->ip3 == _ip3 && contacts[i]->ip4 == _ip4
) return i;
		}
	}
	return -1;
}

int ContactArray::CheckIfPresents( CString addr ) {

	for( int i = 0; i < 100; i++ ) {
		if( filled[i] == 1 ) {
			if( contacts[i]->cIP == addr ) return i;
```

```
		}
	}
	return -1;
}

int ContactArray::CheckIfPresents( int i ){

	if( filled[i] ) return 1;
	else return 0;
}

int ContactArray::GetTabNumber( int cNum ){

	if( filled[cNum] ) return contacts[cNum]->tabNumber;
	else return -1;
}

void ContactArray::DeleteTab( int tNum ) {

	for( int i = 0; i < 100; i++ ) {
		if( filled[i] ) {
			if( contacts[i]->tabNumber > tNum ) contacts[i]->tabNumber-
-;
			else if( contacts[i]->tabNumber == tNum ) contacts[i]-
>tabNumber = -1;
		}
	}
}

int ContactArray::FindByTab( int tNum ) {

	for( int i = 0; i < 100; i++ ) {
		if( filled[i] ) {
			if( contacts[i]->tabNumber == tNum ) return i;
		}
	}
	return -1;
}
```

Contact.cpp

```
// Contact.cpp: implementation of the Contact class.
//
//////////////////////////////////////////////////////////////////////

#include "stdafx.h"
#include "SChat.h"
#include "Contact.h"

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

//////////////////////////////////////////////////////////////////////
// Construction/Destruction
//////////////////////////////////////////////////////////////////////

Contact::Contact()
{
	cHistory = "";
```

```
	cUnsent = "";
	cInput = "";
	cName = "";
	cIP = "";
	tabNumber = -1;

}
Contact::Contact(BYTE _ip1, BYTE _ip2, BYTE _ip3, BYTE _ip4, CString name)
{
	char *tmp = (char*) malloc(100);
	sprintf( tmp, "%d.%d.%d.%d", _ip1, _ip2, _ip3, _ip4 );
	cHistory = "";
	cUnsent = "";
	cInput = "";
	cName = name;
	cIP = tmp;
	tabNumber = -1;
	free( tmp );
	ip1 = _ip1;
	ip2 = _ip2;
	ip3 = _ip3;
	ip4 = _ip4;

}

Contact::~Contact()
{

}
```

ChatTabCtrl.cpp

```
// ChatTabCtrl.cpp : implementation file
//

#include "stdafx.h"
#include "SChat.h"
#include "ChatTabCtrl.h"
#include "hatTab.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CChatTabCtrl

CChatTabCtrl::~CChatTabCtrl()
{

}

void CChatTabCtrl::Init()
{

}

BEGIN_MESSAGE_MAP(CChatTabCtrl, CTabCtrl)
	//{{AFX_MSG_MAP(CChatTabCtrl)
		// NOTE - the ClassWizard will add and remove mapping macros here.
```

```
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CChatTabCtrl message handlers
```

AddDlg.cpp

```
// AddDlg.cpp : implementation file
//

#include "stdafx.h"
#include "SChat.h"
#include "AddDlg.h"
#include "SChatDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CAddDlg dialog


CAddDlg::CAddDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CAddDlg::IDD, pParent)
{
	ipSet = 0;
	validIP = 0;
	//{{AFX_DATA_INIT(CAddDlg)
		// NOTE: the ClassWizard will add member initialization here
	//}}AFX_DATA_INIT
}


void CAddDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CAddDlg)
	DDX_Control(pDX, IDC_EDIT1, newName);
	DDX_Control(pDX, IDC_IPADDRESS1, IpAddress);
	//}}AFX_DATA_MAP
}


BEGIN_MESSAGE_MAP(CAddDlg, CDialog)
	//{{AFX_MSG_MAP(CAddDlg)
	ON_NOTIFY(IPN_FIELDCHANGED, IDC_IPADDRESS1, OnFieldchangedIpaddress1)
	ON_WM_SHOWWINDOW()
	ON_WM_CANCELMODE()
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CAddDlg message handlers

void CAddDlg::GetIp( BYTE& _ip1, BYTE& _ip2, BYTE& _ip3, BYTE& _ip4 ) {
	_ip1 = ip1;
	_ip2 = ip2;
	_ip3 = ip3;
```

```
	_ip4 = ip4;
}

char* CAddDlg::GetName() {
	return nname;
}

void CAddDlg::OnOK() {

	if( IpAddress.GetAddress( ip1, ip2, ip3, ip4) != 4 ) {
		MessageBox("Address is not filled properly", "Error", MB_OK |
MB_ICONERROR );
	    return;
	}
	newName.GetLine( 0, nname, 45 );

	validIP = 1;
	CDialog::OnOK();
}

void CAddDlg::OnFieldchangedIpaddress1(NMHDR* pNMHDR, LRESULT* pResult) {

	*pResult = 0;
}

void CAddDlg::OnShowWindow(BOOL bShow, UINT nStatus)
{
	CDialog::OnShowWindow(bShow, nStatus);
	if( ipSet == 1 ) {
		IpAddress.SetAddress( ip1, ip2, ip3, ip4 );
	} else {
		IpAddress.ClearAddress();
	}

}

void CAddDlg::OnCancelMode()
{
	CDialog::OnCancelMode();

}
```

Приложение 3. Исходный код Linux - клиента

currentchat.cpp

```
/****************************************************************************
** Form implementation generated from reading ui file './currentchat.ui'
**
** Created: 17 07:37:14 2004
**      by:  The User Interface Compiler (uic)
**
** WARNING! All changes made in this file will be lost!
****************************************************************************/
#include "currentchat.h"
#include "unixsocket.h"

#include <qvariant.h>
#include <qpushbutton.h>
#include <qtextedit.h>
#include <qlayout.h>
#include <qtooltip.h>
#include <qwhatsthis.h>

/*
 *  Constructs a CurrentChat which is a child of 'parent', with the
 *  name 'name' and widget flags set to 'f'.
 */
CurrentChat::CurrentChat( UnixSocket *def_unix_socket, QWidget* parent,  const
char* name, WFlags fl,
            QHostAddress *def_ip_address, QString *def_alias )
    : QWidget( parent, name, fl )
{
    if( !name ) {
        setName( "CurrentChat" );
    }

    resize( 364, 344 );

    setCaption( trUtf8( "" ) );

    message = new QTextEdit( this, "message" );
    message->setGeometry( QRect( 10, 170, 366, 121 ) );
    message->setTextFormat( Qt::PlainText );

    chat = new QTextEdit( this, "chat" );
    chat->setGeometry( QRect( 10, 10, 366, 151 ) );
    chat->setReadOnly( TRUE );
    chat->setTextFormat( Qt::PlainText );


    button_send = new QPushButton( this, "button_send" );
    button_send->setGeometry( QRect( 225, 300, 70, 26 ) );
    button_send->setText( trUtf8( "Send" ) );

    button_close = new QPushButton( this, "button_close" );
    button_close->setGeometry( QRect( 305, 300, 70, 26 ) );
    button_close->setText( trUtf8( "Close" ) );

    button_clear = new QPushButton( this, "button_clear" );
    button_clear->setGeometry( QRect( 10, 300, 70, 26 ) );
    button_clear->setText( trUtf8( "Clear" ) );
```

```
    if( def_ip_address ) {
       ip_address = def_ip_address;
    }

    if( def_alias )
    {
       alias = def_alias;
    }

    unix_socket = def_unix_socket;

    connect( button_send, SIGNAL( released() ), this, SLOT( sendButtonReleased()
) );
    connect( button_clear, SIGNAL( released() ), this, SLOT(
clearButtonReleased() ) );
    connect( button_close, SIGNAL( released() ), this, SLOT(
closeButtonReleased() ) );
}

/*
 *  Destroys the object and frees any allocated resources
 */
CurrentChat::~CurrentChat()
{
    // no need to delete child widgets, Qt does it all for us
}

#include "currentchat.moc"

void CurrentChat::sendButtonReleased()
{

    chat->append( QString( "You wrote :" ) );
    chat->append( message->text() );

    unix_socket->Send( &message->text(), ip_address );
    message->clear();
}


void CurrentChat::clearButtonReleased()
{
    chat->clear();
}

void CurrentChat::closeButtonReleased()
{
    chat->clear();
}
```

currentchat.h

```
/****************************************************************************
** Form interface generated from reading ui file './currentchat.ui'
**

** Created: 17 07:36:06 2004
**      by:  The User Interface Compiler (uic)
**
** WARNING! All changes made in this file will be lost!
****************************************************************************/
#ifndef CURRENTCHAT_H
```

```
#define CURRENTCHAT_H

#include <qvariant.h>
#include <qwidget.h>
#include <qhostaddress.h>
#include <qstring.h>
class QVBoxLayout;
class QHBoxLayout;
class QGridLayout;
class QPushButton;
class QTextEdit;
class UnixSocket;

class CurrentChat : public QWidget
{
    Q_OBJECT

public:
    CurrentChat( UnixSocket *def_unix_socket, QWidget* parent = 0, const char*
name = 0, WFlags fl = 0,
         QHostAddress *def_ip_address = 0, QString *def_alias = 0 );
    ~CurrentChat();

    QPushButton* button_send;
    QPushButton* button_close;
    QPushButton* button_clear;
    QTextEdit* chat;
    QTextEdit* message;

    QHostAddress *ip_address;
    QString *alias;
protected:
    UnixSocket *unix_socket;

public slots: // Public slots

  void sendButtonReleased();
  void clearButtonReleased();
  void closeButtonReleased();
};

#endif // CURRENTCHAT_H


newcontactdialog.cpp

#include "newcontactdialog.h"
#include "schatcontactlist.h"

#include <qvariant.h>
#include <qlabel.h>
#include <qlineedit.h>
#include <qpushbutton.h>
#include <qmime.h>
#include <qdragobject.h>
#include <qlayout.h>
#include <qtooltip.h>
#include <qwhatsthis.h>
#include <qimage.h>
#include <qpixmap.h>
#include <qmessagebox.h>
```

```
static QPixmap uic_load_pixmap_NewContactDialog( const QString &name )
{
    const QMimeSource *m = QMimeSourceFactory::defaultFactory()->data( name );
    if ( !m )
       return QPixmap();
    QPixmap pix;
    QImageDrag::decode( m, pix );
    return pix;
}
/*
 *  Constructs a NewContactDialog which is a child of 'parent', with the
 *  name 'name' and widget flags set to 'f'.
 *
 *  The dialog will by default be modeless, unless you set 'modal' to
 *  TRUE to construct a modal dialog.
 */
NewContactDialog::NewContactDialog( QWidget* parent,  const char* name, bool
modal, WFlags fl )
    : QDialog( parent, name, modal, fl )
{
    if ( !name )
       setName( "NewContactDialog" );
    setEnabled( TRUE );
    resize( 250, 133 );
    setSizePolicy( QSizePolicy( (QSizePolicy::SizeType)5,
(QSizePolicy::SizeType)5, 0, 0, sizePolicy().hasHeightForWidth() ) );
    setCaption( trUtf8( "New Contact" ) );

    text_label = new QLabel( this, "text_label" );
    text_label->setGeometry( QRect( 70, 10, 120, 30 ) );
    text_label->setText( trUtf8( "Enter IP address" ) );

    ip_edit_field = new QLineEdit( this, "ip_edit_field" );
    ip_edit_field->setGeometry( QRect( 20, 40, 211, 31 ) );
    ip_edit_field->setFrameShape( QLineEdit::LineEditPanel );
    ip_edit_field->setFrameShadow( QLineEdit::Sunken );


    button_cancel = new QPushButton( this, "button_cancel" );
    button_cancel->setGeometry( QRect( 150, 80, 81, 31 ) );
    button_cancel->setText( trUtf8( "Cancel" ) );


    button_ok = new QPushButton( this, "button_ok" );
    button_ok->setGeometry( QRect( 20, 80, 81, 31 ) );
    button_ok->setText( trUtf8( "OK" ) );

    // signals and slots connections
    connect( button_cancel, SIGNAL( released() ), this, SLOT(
buttonCancelReleased() ) );
    connect( button_ok, SIGNAL( released() ), this, SLOT( buttonOkReleased() )
);
    connect( ip_edit_field, SIGNAL( returnPressed() ), this, SLOT(
buttonOkReleased() ) );

    my_parent=( SchatContactList* )parent;

    init();
}

/*

 *  Destroys the object and frees any allocated resources
```

```
 */
NewContactDialog::~NewContactDialog()
{
    // no need to delete child widgets, Qt does it all for us
}


void NewContactDialog::init()
{
//
}


void NewContactDialog::buttonCancelReleased()
{
    this->hide();
}


void NewContactDialog::buttonOkReleased()
{
    QString address;
    QHostAddress *ipAddress = new QHostAddress();
    address = ip_edit_field->text();

    if( !ipAddress->setAddress( address ) )
    {
        QMessageBox mb( "Simple Chat",
        "Wrong IP address",
        QMessageBox::Information,
        QMessageBox::Ok,
        QMessageBox::NoButton,
        QMessageBox::NoButton );
        mb.exec();
        return;
    }

    my_parent->newIPAddress( ipAddress );
    this->hide();
}
```

newcontactdialog.h

```
#ifndef NEWCONTACTDIALOG_H
#define NEWCONTACTDIALOG_H


#include <qvariant.h>
#include <qdialog.h>
#include <qhostaddress.h>
#include <qstring.h>
class QVBoxLayout;
class QHBoxLayout;
class QGridLayout;
class QLabel;
class QLineEdit;
class QPushButton;

class NewContactDialog : public QDialog
{
    Q_OBJECT
```

```
public:
    NewContactDialog( QWidget* parent = 0, const char* name = 0, bool modal =
FALSE, WFlags fl = 0 );
    ~NewContactDialog();

    QLabel* text_label;
    QPushButton* button_cancel;
    QLineEdit* ip_edit_field;
    QPushButton* button_ok;


public slots:
    virtual void init();
    virtual void buttonCancelReleased();
    virtual void buttonOkReleased();


protected:

    friend class SchatContactList;
    SchatContactList *my_parent;
};

#endif // NEWCONTACTDIALOG_H
```

schatcontactlist.cpp

```
#include "newcontactdialog.h"
#include "schatcontactlist.h"
#include "currentchat.h"
#include "unixsocket.h"
#include "list.h"

#include <qvariant.h>
#include <qlistbox.h>
#include <qtabwidget.h>
#include <qwidget.h>
#include <qmime.h>
#include <qdragobject.h>
#include <qlayout.h>
#include <qtooltip.h>
#include <qwhatsthis.h>

#include <qaction.h>
#include <qmenubar.h>
#include <qpopupmenu.h>
#include <qtoolbar.h>
#include <qimage.h>
#include <qpixmap.h>
#include <qtextedit.h>



/*
 *  Constructs a SchatContactList which is a child of 'parent', with the
 *  name 'name' and widget flags set to 'f'.
 *
 */
SchatContactList::SchatContactList( QWidget* parent,  const char* name, WFlags
fl )
    : QMainWindow( parent, name, fl )
```

```
{
    if ( !name )
        setName( "SchatContactList" );
    resize( 576, 408 );
    setCaption( trUtf8( "Simple Chat" ) );
    setCentralWidget( new QWidget( this, "qt_central_widget" ) );

    tab_widget = new QTabWidget( centralWidget(), "tab_widget" );
    tab_widget->setGeometry( QRect( 180, 12, 391, 365 ) );


    contact_list = new QListBox( centralWidget(), "contact_list" );
    contact_list->setGeometry( QRect( 10, 0, 161, 371 ) );

    // actions

    new_connection_dialog_action = new QAction( this,
"new_connection_dialog_action" );
    new_connection_dialog_action->setText( trUtf8( "New Connection" ) );
    new_connection_dialog_action->setMenuText( trUtf8( "&New Connection" ) );
    new_connection_dialog_action->setWhatsThis( trUtf8( "Create a new
connection" ) );
    new_connection_dialog_action->setAccel( 4194382 );

    delete_connection_action = new QAction( this, "delete_connection_action" );
    delete_connection_action->setText( trUtf8( "Delete Connection" ) );
    delete_connection_action->setMenuText( trUtf8( "&Delete Connection" ) );
    delete_connection_action->setWhatsThis( trUtf8( "Detele current connection"
) );
    delete_connection_action->setAccel( 4194372 );

    opions_action = new QAction( this, "opions_action" );
    opions_action->setText( trUtf8( "Options" ) );
    opions_action->setMenuText( trUtf8( "&Options" ) );
    opions_action->setWhatsThis( trUtf8( "Options menu" ) );
    opions_action->setAccel( 4194383 );

    exit_action = new QAction( this, "exit_action" );
    exit_action->setText( trUtf8( "Exit" ) );
    exit_action->setMenuText( trUtf8( "E&xit" ) );
    exit_action->setAccel( 4194385 );


    // menu_bar
    menu_bar = new QMenuBar( this, "menu_bar" );

    popup_menu = new QPopupMenu( this );
    new_connection_dialog_action->addTo( popup_menu );
    delete_connection_action->addTo( popup_menu );
    opions_action->addTo( popup_menu );
    exit_action->addTo( popup_menu );
    menu_bar->insertItem( trUtf8( "System" ), popup_menu );

    connect( exit_action, SIGNAL( activated() ), this, SLOT( close() ) );
    connect( new_connection_dialog_action, SIGNAL( activated() ), this, SLOT(
showDialog() ) );
    connect( contact_list, SIGNAL( doubleClicked(QListBoxItem*) ), this, SLOT(
contactListDoubleClicked(QListBoxItem*) ) );
    connect( contact_list, SIGNAL( returnPressed(QListBoxItem*) ), this, SLOT(
contactListDoubleClicked(QListBoxItem*) ) );
    connect( this, SIGNAL( messageReceived( QString*, QHostAddress*) ), this,
SLOT( messageHandle( QString*, QHostAddress*) ) );
    init();
```

```
    QThread::start();
}

/*
 *  Destroys the object and frees any allocated resources
 */
SchatContactList::~SchatContactList()
{
    // no need to delete child widgets, Qt does it all for us
}


void SchatContactList::init()
{
    contact_dialog = new NewContactDialog( this );
    list = new List();
    unix_socket = new UnixSocket();
}

void SchatContactList::run()
{
    while( TRUE )
    {
        QString *message;
        QHostAddress *from;

        unix_socket->Receive( &message, &from );

        emit messageReceived( message, from );
    }
}

void SchatContactList::helpIndex()
{

}

void SchatContactList::helpContents()
{

}

void SchatContactList::helpAbout()
{

}

void SchatContactList::showDialog()
{
    contact_dialog->show();
}

/** This function adds IP address to contact list */
void SchatContactList::newIPAddress( QHostAddress *new_address )
{
    int i;
    int index;


    if( ( i = list->getIndex( new_address ) ) >= 0 )
    {
       contact_list->setSelected( list->getListBoxItem( i ), TRUE );
       return;
```

```
    }


    index = list->add( new_address );

    QString *alias = new QString( new_address->toString() );


    list->add( alias, index );

    contact_list->insertItem( *alias );
    contact_list->setSelected( contact_list->count() - 1, TRUE );

    list->add( contact_list->item( contact_list->count() - 1 ), index );


    contact_list->sort();
}

/** No descriptions */
void SchatContactList::contactListDoubleClicked( QListBoxItem *list_box_item )
{
    int i;


    i = list->getIndex( list_box_item );

    if( list->getCurrentChat( i ) != NULL )
    {
        int index = list->getTabIndex( i );

        tab_widget->setTabEnabled( tab_widget->page( index ) , TRUE );
        tab_widget->showPage( tab_widget->page( index ) );
        return;
    }

    current_chat = new CurrentChat( unix_socket, this, NULL, 0,
                   list->getIP( i ), list->getAlias( i ) );
    tab_widget->addTab( current_chat, list_box_item->text() );
    tab_widget->setTabEnabled( current_chat, TRUE );
    tab_widget->showPage( current_chat );


    list->add( current_chat, i );
    list->setTabIndex( tab_widget->count() - 1, i );

    current_chat->show();
}


/** No descriptions */
void SchatContactList::closeContactList( QListBoxItem *list_box_item )
{
    int i;


    i = list->getIndex( list_box_item );

    if( list->getCurrentChat( i ) != NULL )
    {
        int index = list->getTabIndex( i );

        tab_widget->setTabEnabled( tab_widget->page( index ) , TRUE );
```

```
        tab_widget->showPage( tab_widget->page( index ) );
        return;
    }

    current_chat = new CurrentChat( unix_socket, this, NULL, 0,
                   list->getIP( i ), list->getAlias( i ) );
    tab_widget->addTab( current_chat, list_box_item->text() );
    tab_widget->setTabEnabled( current_chat, TRUE );
    tab_widget->showPage( current_chat );


    list->add( current_chat, i );
    list->setTabIndex( tab_widget->count() - 1, i );

    current_chat->show();
}


/** slot, connected to messageReceived. This function handles received message
*/
void SchatContactList::messageHandle( QString *message, QHostAddress *from )
{
    int index;

    index = list->getIndex( from );

    if( index >= 0 )
    {
        current_chat = list->getCurrentChat( index );
        current_chat->chat->append( from->toString() + QString( " wrote:" ) );
        current_chat->chat->append( *message );
        // TODO bolding list item
        return;
    }

    newIPAddress( from );

    index = list->getIndex( from );

    current_chat = new CurrentChat( unix_socket, this, "Current Chat" );
    current_chat = new CurrentChat( unix_socket, this, NULL, 0,
                   list->getIP( index ), list->getAlias( index ) );
    tab_widget->addTab( current_chat, *list->getAlias( index ) );

    tab_widget->setTabEnabled( current_chat, TRUE );
    tab_widget->showPage( current_chat );

    list->add( current_chat, index );
    list->setTabIndex( tab_widget->count() - 1, index );

    current_chat->show();


    list->getCurrentChat( index )->chat->append( from->toString() + QString( "
wrote:" ) );
    list->getCurrentChat( index )->chat->append( *message );
}
```

schatcontactlist.h

```
#ifndef SCHATCONTACTLIST_H
#define SCHATCONTACTLIST_H
```

```
#include <qvariant.h>
#include <qhostaddress.h>
#include <qmainwindow.h>

#include <qthread.h>

class QVBoxLayout;
class QHBoxLayout;
class QGridLayout;
class QAction;
class QActionGroup;
class QToolBar;
class QPopupMenu;
class QListBox;
class QListBoxItem;
class QTabWidget;
class QWidget;


class SchatContactList : public QMainWindow, public QThread
{
    Q_OBJECT

public:
    SchatContactList( QWidget* parent = 0, const char* name = 0, WFlags fl =
WType_TopLevel );
    ~SchatContactList();

    QTabWidget* tab_widget;
    QListBox* contact_list;
    QMenuBar *menu_bar;
    QPopupMenu *popup_menu;
    QPopupMenu *help_menu;
/*

    QAction* help_contents_action;
    QAction* help_index_action;
    QAction* help_about_action;
*/
    QAction* new_connection_dialog_action;
    QAction* delete_connection_action;
    QAction* opions_action;
    QAction* exit_action;



public slots:
    virtual void init();
    virtual void helpIndex();
    virtual void helpContents();
    virtual void helpAbout();



private slots:
    virtual void showDialog();

protected:

    void run();

    /** This function adds IP address to contact list */
```

```
    virtual void newIPAddress( QHostAddress *new_address );


    friend class NewContactDialog;
    NewContactDialog *contact_dialog;

    friend class CurrentChat;
    CurrentChat *current_chat;

    friend class List;
    List *list;

    friend class UnixSocket;
    UnixSocket *unix_socket;

public slots: // Public slots
  /** No descriptions */
  virtual void contactListDoubleClicked( QListBoxItem *list_box_item );
public slots: // Public slots
  /** slot, connected to messageReceived. This function handles received message
*/
  void messageHandle( QString *message, QHostAddress *from );
signals: // Signals
  /** signal, that message is received */
  void messageReceived( QString *message, QHostAddress *from );
};

#endif // SCHATCONTACTLIST_H
```

unixsocket.cpp

```
/***************************************************************************
                          unixsocket.cpp  -  description
                             -------------------
    begin                : 19 2004
    copyright            : (C) 2004 by Evgeny Kalugin
    email                : kalugin@ake-vg.spb.ru
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#include "unixsocket.h"

#include <unistd.h>
#include <stdio.h>

#include <qstring.h>
#include <qhostaddress.h>


UnixSocket::UnixSocket()
{
   struct sockaddr_un addr;

   sockfd = socket( AF_UNIX, SOCK_STREAM, 0 );
```

```
   if( !sockfd )
   {
       perror( "socket" );
   }

   addr.sun_family = AF_UNIX;
   strcpy( addr.sun_path, INT_SOCKET_PATH );


   if( connect( sockfd, ( struct sockaddr* )&addr, sizeof( addr ) ) != 0 )
   {
       perror( "connect" );
   }

}

UnixSocket::~UnixSocket()
{
    close( sockfd );
}

void UnixSocket::Receive( QString **message, QHostAddress **from )
{
    t_message_int msg;

    recv( sockfd, &msg, sizeof( msg ), 0 );


    *message = new QString( msg.message );
    *from = new QHostAddress( msg.ipaddr );

}
void UnixSocket::Send( QString *message, QHostAddress *to )
{

    t_message_int msg;


    sprintf( msg.message, "%s", message->latin1() );


    msg.ipaddr = to->ip4Addr();

    send( sockfd, &msg, strlen( msg.message ) + sizeof( msg.ipaddr ) + 1, 0 );

}
```

unixsocket.h

```
/***************************************************************************
                          unixsocket.h  -  description
                             -------------------
    begin                : 19 2004
    copyright            : (C) 2004 by Evgeny Kalugin
    email                : kalugin@ake-vg.spb.ru
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
```

```
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef UNIXSOCKET_H
#define UNIXSOCKET_H

#include <sys/socket.h>
#include <sys/un.h>


#define INT_SOCKET_PATH   "/tmp/schat3538"
#define MAX_MESSAGE_SIZE  64*1024


typedef struct {
   u_int32_t ipaddr;
   char message[MAX_MESSAGE_SIZE];
} t_message_int;



class QString;
class QHostAddress;


/**This class provides operating with AF_UNIX sockets
  */

class UnixSocket
{
private:
    int sockfd;

protected:
    friend class SchatContactList;
    SchatContactList *contact_list;

public:
    UnixSocket();
    ~UnixSocket();
    void Receive( QString **message, QHostAddress **from );
    void Send( QString *message, QHostAddress *to );

};

#endif
```

Приложение 4. Исходный код Linux - сервера

schat_deamon.c

```
/***************************************************************************
                          schat_deamon.c  -  the Simple chat deamon
                              source code
    begin                : Срд Янв 21 2004
    copyright            : (C) 2004 by Evgeny Kalugin
    email                : kalugin@ake-vg.spb.ru
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifdef HAVE_CONFIG_H
#include <config.h>
#endif


#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <errno.h>

#include <sys/poll.h>
#include <sys/socket.h>
#include <sys/un.h>
#include <netinet/in.h>


#include "schat_deamon.h"


t_message_int message_int;
t_message_ext message_ext;

int ext_socket, int_socket;

struct pollfd sockets[2];
int pollres;

int e;
int y0;

int n_int_msg, n_ext_msg;


extern int init_int_socket( void );
extern int init_ext_socket( void );



void z0( void );
void z1( void );
void z2( void );
```

```
void z3( void );
void z4( void );
void z5( void );
void z6( void );
void z7( void );



void log( void ) {


   if( e != -1 ) {
      printf( "Событие e%d                :   ", e );
   }

   switch( e ) {
      case 0:
         printf( "в локальном сокете появилось сообщение\n" );
         break;
      case 1:
         printf( "в сетевом сокете появилось сообщение\n" );
         break;
      case 2:
         printf( "в локальном сокете сообщений не осталось\n" );
         break;
      case 3:
         printf( "в сетевом сокете сообщений не осталось\n" );
         break;
      case 4:
         printf( "превышен лимит по количеству сообщений, полученых подряд из
локального сокета ( %d )\n", MAX_N_INT_MSG );
         break;
      case 5:
         printf( "превышен лимит по количеству сообщений, полученых подряд из
сетевого сокета ( %d )\n", MAX_N_EXT_MSG );
         break;
      case 6:
         printf( "завершение работы клиента\n" );
         break;
      case 7:
         printf( "успешный прием локального сообщения\n" );
         break;
      case 8:
         printf( "успешное преобразование в сетевой формат\n" );
         break;
      case 9:
         printf( "успешная отправка сообщения в сеть\n" );
         break;
      case 10:
         printf( "успешный прием сообщения из сети\n" );
         break;
      case 11:
         printf( "успешное преобразование в локальный формат\n" );
         break;
      case 12:
         printf( "успешная отправка локального сообщения\n" );
         break;
   }


   if( y0 != 7 ) {
      printf( "A0 перешел в состояние %d   :   ", y0 );
   }
```

```
   switch( y0 ) {
       case 0 :
          printf( "Ожидание сообщения\n\n" );
          break;
       case 1 :
          printf( "Получение сообщения из локального сокета\n\n" );
          break;
       case 2 :
          printf( "Конверование сообщения в сетевой формат\n\n" );
          break;
       case 6 :
          printf( "Отправка сообщения через локальный сокет\n" );
          printf("    сообщение = '%s'\n",
                          ( char* )message_int.message );
          printf("    адрес отправителя  %d.%d.%d.%d\n\n",
                          ( message_int.ipaddr >> 24 ) % 256,
                          ( message_int.ipaddr >> 16 ) % 256,
                          ( message_int.ipaddr >> 8 ) % 256,
                          ( message_int.ipaddr) % 256 );
          break;
       case 4 :
          printf( "Получение сообщения из сетевого сокета\n\n" );
          break;
       case 5 :
          printf( "Конверование сообщения во внутренний формат\n\n" );
          break;
       case 3 :
          printf( "Отправка сообщения через сетевой сокет\n" );
          printf("    сообщение = '%s'\n",
                          ( char* )message_ext.message );
          printf("    адрес назначения = %d.%d.%d.%d\n\n",
                          ( message_ext.ipaddr >> 24 ) % 256,
                          ( message_ext.ipaddr >> 16 ) % 256,
                          ( message_ext.ipaddr >> 8 ) % 256,
                          ( message_ext.ipaddr) % 256 );

          break;
       case 7 :
          printf( "Демон закончил работу\n" );
          break;
       default :
          printf( "Непредвиденная ситуация\n\n" );
          break;
    }
    fflush( stdout );
}


void A0( void )
{

   if( e == 6 ) { y0 = 7; };

   log();
   switch( y0 ) {
      case 0:
         z0();
         if( e == 0 ) { y0 = 1; break; };
         if( e == 1 ) { y0 = 4; break; };
         break;
      case 1:
```

```
         if( e == 4 ) { y0 = 4; break; };
         z1();
         if( e == 2 ) { y0 = 0; break; };
         if( e == 7 ) { y0 = 2; break; };
         break;
      case 2:
         z2();
         if( e == 8 ) { y0 = 3; break; };
         break;
      case 3:
         z3();
         if( e == 9 ) { y0 = 1; break; };
         break;
      case 4:
         if( e == 5 ) { y0 = 1; break; };
         z4();
         if( e == 3 ) { y0 = 0; break; };
         if( e == 10 ) { y0 = 5; break; };
         break;
      case 5:
         z5();
         if( e == 11 ) { y0 = 6; break; }
         break;
      case 6:
         z6();
         if( e == 12 ) { y0 = 4; break; }
         break;
      case 7:
         z7();
         break;
      default:
         y0 = 7;
         break;

   }
}

void z0( void ) {
   n_int_msg = 0;
   n_ext_msg = 0;

   pollres = poll( sockets, 2, -1 );

   if( sockets[0].revents & ( POLLERR | POLLHUP ) ) {
      perror( "Internal socket failed" );
      e = 6;
      return;
   }

   if( sockets[1].revents & ( POLLERR | POLLHUP ) ) {
      perror( "External socket failed" );
      e = 6;
      return;
   }


   if( sockets[0].revents & ( POLLIN | POLLPRI ) ) {
      e = 0;
      return;
   }

   if( sockets[1].revents & ( POLLIN | POLLPRI ) ) {
      e = 1;
```

```
      return;
   }

   e = 6;
}

void z1( void ) {
   int nbytes;

   errno = 0;

   if( n_int_msg >= MAX_N_INT_MSG ) {
      e = 4;
      n_int_msg = 0;
      return;
   }

   n_int_msg++;

   bzero( message_int.message, sizeof( message_int.message ) );

   nbytes = recv( int_socket, &message_int, sizeof( message_int ), MSG_DONTWAIT
);

   if( errno == EAGAIN ) {
      e = 2;
      return;
   }

   if(  nbytes < 0 ) {
      perror( "Error in receiving message from local socket" );
      e = 6;
      return;
   }

   e = 7;
   return;
}

void z2( void ) {
   message_ext.ipaddr = message_int.ipaddr;
   sprintf( message_ext.message, "%s", message_int.message );
   e = 8;
   return;
}

void z3( void ) {
   struct sockaddr_in clnt_addr;
   int caddrlen = sizeof( struct sockaddr_in );

   clnt_addr.sin_family = AF_INET;
   clnt_addr.sin_port = htons( ( u_short )EXT_SOCKET_PORT );
   clnt_addr.sin_addr.s_addr = htonl( message_ext.ipaddr );

   if( sendto( ext_socket, message_ext.message,
        strlen( message_ext.message ) + 1, 0,
        ( struct sockaddr* )&clnt_addr, caddrlen ) < 0 ) {
      perror( "Error in sending message to network" );
      e = 6;
      return;
   }
   e = 9;
   return;
```

```
}

void z4( void ) {

   struct sockaddr_in clnt_addr;
   int caddrlen = sizeof( struct sockaddr_in );
   int len;

   errno = 0;

   if( n_ext_msg >= MAX_N_EXT_MSG ) {
      e = 5;
      n_ext_msg = 0;
      return;
   }

   n_ext_msg++;

   bzero( message_ext.message, sizeof( message_ext.message ) );

   len = recvfrom( ext_socket, message_ext.message, sizeof( message_ext.message
),
           MSG_DONTWAIT, ( struct sockaddr* )&clnt_addr, &caddrlen );

   if( errno == EAGAIN ) {
      e = 3;
      return;
   }

   if( len < 0 ) {
      perror( "Error in receiving message from network" );
      e = 6;
      return;
   }

   message_ext.ipaddr = ntohl( clnt_addr.sin_addr.s_addr );

   e = 10;
   return;
}


void z5( void ) {
   message_int.ipaddr = message_ext.ipaddr;
   sprintf( message_int.message, "%s", message_ext.message );
   e = 11;
   return;
}

void z6( void ) {

   if( send( int_socket, &message_int,
       strlen( message_int.message ) + sizeof( message_int.ipaddr ) + 1, 0 ) < 0
) {
      perror( "Error in sending message through local socket" );
      e = 6;
      return;
   }

   e = 12;
   return;
}
```

```
void z7( void ) {
   close( int_socket );
   close( ext_socket );
   unlink( INT_SOCKET_PATH );
}


int main( void ) {
   printf( "\nДемон Запущен\n\n" );
   printf( "Ждем соединения с клиентом..." );

   int_socket = init_int_socket();
   ext_socket = init_ext_socket();

   printf( " готово\n\n" );

   e = -1;
   if( !int_socket || !ext_socket ) {
      e = 6;
   }

   sockets[0].fd = int_socket;
   sockets[1].fd = ext_socket;

   sockets[0].events = POLLIN | POLLPRI;
   sockets[1].events = POLLIN | POLLPRI;

   y0 = 0;

   while( y0 != 7 ) {
      A0();
   }

   return 0;
}
```

schat_deamon.h

```
/***************************************************************************
                          schat_deamon.h  -  description
                             -------------------
    begin                : Срд Янв 21 2004
    copyright            : (C) 2004 by Evgeny Kalugin
    email                : kalugin@ake-vg.spb.ru
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#ifndef  __SCHAT_DEAMON_H
#define __SCHAT_DEAMON_H
```

```
#define INT_SOCKET_PATH   "/tmp/schat3538"
#define EXT_SOCKET_PORT   3538
#define MAX_MESSAGE_SIZE  64*1024

#define MAX_N_INT_MSG 20
#define MAX_N_EXT_MSG 20

typedef struct {
   u_int32_t ipaddr;
   char message[MAX_MESSAGE_SIZE];
} t_message_int;


typedef struct {
   u_int32_t ipaddr;
   char message[MAX_MESSAGE_SIZE];
} t_message_ext;


#endif /* __SCHAT_DEAMON_H */
```

schat_deamon_utils.c

```
/***************************************************************************
                          schat_deamon_utils.c  -  description
                             -------------------
    begin                : Срд Янв 21 2004
    copyright            : (C) 2004 by Evgeny Kalugin
    email                : kalugin@ake-vg.spb.ru
 ***************************************************************************/

/***************************************************************************
 *                                                                         *
 *   This program is free software; you can redistribute it and/or modify  *
 *   it under the terms of the GNU General Public License as published by  *
 *   the Free Software Foundation; either version 2 of the License, or     *
 *   (at your option) any later version.                                   *
 *                                                                         *
 ***************************************************************************/

#include <unistd.h>
#include <stdio.h>


#include <sys/socket.h>
#include <sys/un.h>
#include <netinet/in.h>


#include "schat_deamon.h"



int init_int_socket( void ) {

   struct sockaddr_un addr;
   int sockfd, csockfd;

   unlink( INT_SOCKET_PATH );

   sockfd = socket( AF_UNIX, SOCK_STREAM, 0 );
```

```
   if( sockfd <= 0 ) {
      perror( "Error in creating local socket" );
      goto fail;
   }

   addr.sun_family = AF_UNIX;
   strcpy( addr.sun_path, INT_SOCKET_PATH );

   if( bind( sockfd, ( struct sockaddr* )&addr, sizeof( addr ) ) ) {
      perror( "Error in binding local socket" );
      goto fail;
   }

   if( listen( sockfd, 1 ) ) {
      perror( "Error while listening local socket" );
      goto fail;
   }

   csockfd = accept( sockfd, NULL, NULL );

   if( csockfd <= 0 ) {
      perror( "Error in accepting local connection" );
      goto fail;
   }

   if( close( sockfd ) != 0 ) {
      perror( "Error in closing parent local socket" );
   }

   return csockfd;

 fail:

   return 0;
}


int init_ext_socket( void ) {

   struct sockaddr_in addr;
   int sockfd;

   sockfd = socket( AF_INET, SOCK_DGRAM, IPPROTO_UDP );

   if( sockfd <= 0 ) {
      perror( "Error in creating network socket" );
      goto fail;
   }

   addr.sin_family = AF_INET;
   addr.sin_port = htons( ( u_short )EXT_SOCKET_PORT );
   addr.sin_addr.s_addr = INADDR_ANY;

   if( bind( sockfd, ( struct sockaddr* )&addr, sizeof( addr ) ) ) {
      perror( "Error in binding network socket" );
      goto fail;
   }

   return sockfd;

 fail:
   return 0;
}
```