

293

TGPSI

Technological Project Project leChat

Report (25-06-2012)

Nº 103629 – David Samuel Pereira Silva

Nº 104289 – Diogo Alexandre Flores Simões de Jesus



Student Identification

David Samuel Pereira Silva

Age: 18

Class: 293

E-mail: zgrav@null.net

Phone number: +351918586295

Diogo Alexandre Jesus

Age: 16

Class: 293

E-mail: evilmonstah@innocent.com

Phone number: +351915345721

Index

Content

[Introduction 4](#_Toc329069976)

[Project Goals 5](#_Toc329069977)

[Project proposal 6](#_Toc329069978)

[Functionalities 7](#_Toc329069979)

[Multithreading 7](#_Toc329069980)

[Sockets 7](#_Toc329069981)

[Code 9](#_Toc329069982)

[Server 9](#_Toc329069983)

[Client 15](#_Toc329069984)

[*Stresstester* 18](#_Toc329069985)

[Program history 20](#_Toc329069986)

[Technologies used 23](#_Toc329069987)

[Conclusion 24](#_Toc329069988)

[Gratitude 25](#_Toc329069989)

# Introduction

Within the 2nd year of TGPSI (Technical Management and Systems Programming), we were asked to develop a Technology Project and proceed with the presentation of the same at the end of their school year.

This design consists of two field-based applications some acquired during the course itself and also by self-exploration. It aims to demonstrate these skills and expose the development of a little more complex project using Threads and Sockets.

The time given for completion and documentation of the project was roughly around 2 months. Having started with the proposals on 18 April 2012, the date of delivery of the project was set for June 26, 2012.

The theme and features of the project are left to the students. We chose to create a kind of chat over LAN.

Throughout this introduction, we will address the following aspects:

* Project Goals;
* Project Proposal;
* Technologies and resources used;
* Contribution of the project;
* Organization of the report;

# Project Goals

The topic (Chat over LAN) presents several characteristics that may be well developed if approached properly, if not several complications may arise and may not be as user friendly as it should.  
After having seen some programs and their functions, it was noticed that some did not contain some features crucial for its development to be at 100% and that's what we try to make in our project.

The main objectives of this project are:

* Server & Client
* A list of users
* Message system and the date/time of message
* History
* Notifications
* File transfers

Future implementations:

* Image display for each user
* Emoticons
* Font formatting
* Themes

# Project proposal

The proposal defined the main objectives (p. 4) and taking into account the objectives were defined start dates and end of each stage of development and schedule.

The proposal was delivered May 8th, 2012.

|  |  |  |  |
| --- | --- | --- | --- |
| Phase | Start date | End date | Hours/days |
| Planning | 18-04-2012 | 20-04-2012 | 48/2 |
| Proposal | 08-05-2012 | 08-05-2012 | 24/1 |
| Analysis | 19-04-2012 | 20-04-2012 | 48/2 |
| Implementation | 19-04-2012 | 25-06-2012 | 1632/68 |
| Documentation | 25-06-2012 | 25-06-2012 | 24/1 |
| Apresentation | ? | ? | ? |

Table 1 – Dates relative to the development phase

Table 1 shows predictions for the start and end of each development phase and the number of hours.

Although the table is useful, you need a detailed schedule to be able to visually interpret the time set for each stage.

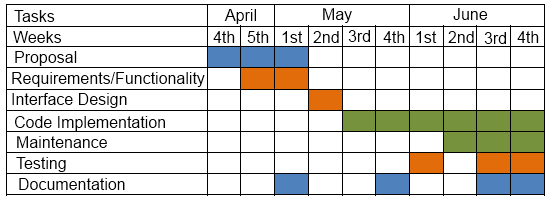


Tabela 2 – Chronogram

Legend:

Blue – Documentation

Orange – Analysis

Green – Implementation

# Functionalities

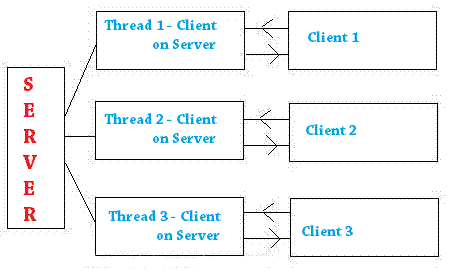
Allows the exchange of messages and / or files among users connected to the server, it also facilitates communication between co-workers who are scattered around the building and need to communicate with each other, however the program can also be used in a living space.

## Multithreading

Socket Programming with multithreading means that a server can communicate with more than one client at the same time using communication via socket.

The basic idea behind multithreaded programming with sockets is that whenever the server receives a connection request from client, the server creates a thread (independent) separately for each client request.

This means that for every client, there is a thread of the client separately. Thus, the client can communicate independently with their own client on the server thread.

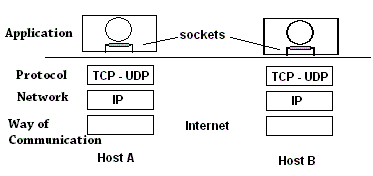
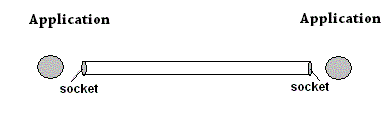


## Sockets

A socket can be understood as a port of a communication that allows a process running on a computer to send / receive messages to / from another process that may be running on the same computer or a remote computer.

The sockets then allow the process to process communication as follows:

* Local communication: local processes using local sockets
* Remote Communication: remote processes using network sockets (TCP/IP)



# Code

## Server

The server accepts connections from clients and deals with information and messages of the same, we will show you the three most important functions of the server.

Main Function – This function sets the port that "listens" to clients and starts the server on that port. Receives connections and redirects the client to the hashtable where he is assigned a position where the client and stores all the information corresponding to it. It also starts the function of the console commands and timer that updates the list of clients automatically. It also deals with the server restart automatically after three seconds if a value in the port is not adequate.

|  |
| --- |
| #Region "TCP/IP protocol and Client handling"  Sub Main()  Console.Title = "leChat Server"  Dim port As String  Console.WriteLine("Please insert a port value.")  Console.WriteLine("If no port value is entered, the default will be 8888.")  port = ReadLine()  If String.IsNullOrEmpty(port) Then  port = "8888"  End If  If IsNumeric(port) = True Then  '---timer---  listtimer.AutoReset = True  listtimer.Interval = 60000  AddHandler listtimer.Elapsed, AddressOf tick  listtimer.Enabled = True  listtimer.Start()  '---timer end---  Try  Dim server As New TcpListener(port)  Dim client As TcpClient  Dim IP = GetIPv4Address()  Console.Clear()  Dim conShellThread As Threading.Thread = New Threading.Thread(AddressOf conShell)  server.Start()  message("Server up @ " + IP)  message("Port: " + port)  conShellThread.Start()  count = 0  While (True)  count += 1  client = server.AcceptTcpClient()  Dim bytes(10024) As Byte  Dim netwrkStream As NetworkStream = client.GetStream() ' Network and Client data Stream  netwrkStream.Read(bytes, 0, CInt(client.ReceiveBufferSize)) ' Reads data received from client.  dataClnt = System.Text.Encoding.ASCII.GetString(bytes)  If dataClnt.IndexOf("$") < 0 Then  clntDisc = True  Exit Sub  Else  dataClnt = dataClnt.Substring(0, dataClnt.IndexOf("$"))  Dim orgnick As String  Dim countuser As Integer  Dim orgnickbackup As String  orgnick = dataClnt  countuser = countuser + 1  orgnickbackup = orgnick  If countuser > 1 AndAlso orgnick = dataClnt Then  If clList.Contains(orgnickbackup) Then  dataClnt = dataClnt & "\_" & countuser  orgnick = ""  message("Multiple nickname found: " & orgnickbackup)  clList(dataClnt) = client  bcast(dataClnt + " " + "(" + Convert.ToString(count) + ")" + " " + "joined " + DateAndTime.Now, dataClnt, False)  message(dataClnt + " " + "(" + Convert.ToString(count) + ")" + " " + "joined the lobby. " + DateTime.Now)  bcast("This nickname is in use by another client.", Nothing, False)  Dim clienth2 As New handleClnt  clienth2.startClient(client, dataClnt, clList)  countuser = 0  Continue While  End If  End If  If userlist.Contains(orgnickbackup) Then  For i = 0 To userlist.Count - 1  If userlist.Item(i).ToString.Contains(orgnickbackup) Then  countuser = countuser + 1  End If  Next  dataClnt = dataClnt & "\_" & countuser  orgnick = ""  message("Multiple nickname found: " & orgnickbackup)  clList(dataClnt) = client  bcast(dataClnt + " " + "(" + Convert.ToString(count) + ")" + " " + "joined " + DateAndTime.Now, dataClnt, False)  message(dataClnt + " " + "(" + Convert.ToString(count) + ")" + " " + "joined the lobby. " + DateTime.Now)  bcast("This nickname is in use by another client.", Nothing, False)  Dim clienth3 As New handleClnt  clienth3.startClient(client, dataClnt, clList)  countuser = 0  Continue While  Else  clList(dataClnt) = client  bcast(dataClnt + " " + "(" + Convert.ToString(count) + ")" + " " + "joined " + DateAndTime.Now, dataClnt, False)  message(dataClnt + " " + "(" + Convert.ToString(count) + ")" + " " + "joined the lobby. " + DateTime.Now)  Dim clienth As New handleClnt  clienth.startClient(client, dataClnt, clList) ' Client handler  End If  End If  End While  Catch ex As IOException  ' message("Socket not disposed properly.")  clntDisc = True  End Try  Else  Console.WriteLine()  Console.WriteLine("An incorrect port value has been set. Program restarting in 3 seconds.")  If Directory.Exists(Application.StartupPath & "\Log") = False Then  Call Directory.CreateDirectory(Application.StartupPath & "\Log")  End If  My.Computer.FileSystem.WriteAllText("Log\Server Log " + FormatDateTime(DateAndTime.Today, DateFormat.LongDate) + ".log", DateTime.UtcNow + " >> " + "Server restarted!" + Environment.NewLine, True)  Thread.Sleep(3000)  Application.Restart()  End If  End Sub  #End Region |

Broadcast function – This feature receives messages from the client and spreads them to all connected clients.

|  |
| --- |
| #Region "Broadcasting"  Private Sub bcast(ByVal msg As String, ByVal userName As String, ByVal msgflag As Boolean)  Dim Item As DictionaryEntry  Try  For Each Item In clList  Dim bcastSocket As TcpClient 'Broadcast Socket  bcastSocket = CType(Item.Value, TcpClient)  Dim bcastStream As NetworkStream = bcastSocket.GetStream()  Dim bcastBytes As [Byte]() 'Broadcast Length received on bcastStream.Write  If msgflag = True Then  bcastBytes = Encoding.ASCII.GetBytes(DateTime.Now + " " + userName + " says : " + msg)  msg2 = msg  Else  bcastBytes = Encoding.ASCII.GetBytes(msg)  End If  bcastStream.Write(bcastBytes, 0, bcastBytes.Length)  bcastStream.Flush()  Next  Catch ex As IOException  ' message("Socket not disposed properly.")  clntDisc = True  Catch ex2 As ObjectDisposedException  ' message("Object not disposed properly.")  clntDisc = True  Catch ex3 As InvalidOperationException  ' message("Socket not disposed properly.")  clntDisc = True  End Try  End Sub  #End Region |

**Client handling function** – This function saves the connection data of the Client (Name, IP) in a hashtable and also receives and redirect commands to the right place to perform their respective function. It also performs the function of saving the new user name if it is changed. Also handles client disconnection.

|  |
| --- |
| #Region "Client/Chat handling"  Public Class handleClnt  Dim cSocket As TcpClient ' Client Socket  Dim clName As String  Dim clList As Hashtable  Dim cnum As Integer  Dim cip As String  Dim nickcount As Integer  Public Sub startClient(ByVal inClntSocket As TcpClient, ByVal clntName As String, ByVal cList As Hashtable)  Me.cSocket = inClntSocket ' Client Socket  Me.clName = clntName ' Client Name  Me.clList = cList ' Client List  Me.cip = inClntSocket.Client.RemoteEndPoint.ToString  cip = cip.Substring(0, cip.IndexOf(":"))  Me.cnum = count ' Client Number  userlist.Add(Me.clName)  iplist.Add(Me.cip)  Dim ctThread As Threading.Thread = New Threading.Thread(AddressOf initiateChat) ' Client Threading  ctThread.Start()  End Sub  Private Sub initiateChat() ' Message from Client  Dim bytesFrom(10024) As Byte  Dim dataClnt As String ' Data from Client to Server  While (True)  Try  Dim Stream\_network As NetworkStream = cSocket.GetStream() ' Network Stream  Stream\_network.Read(bytesFrom, 0, CInt(cSocket.ReceiveBufferSize))  dataClnt = System.Text.Encoding.ASCII.GetString(bytesFrom)  If dataClnt.IndexOf("$") < 0 Then  clntDisc = True  cdisc()  Exit Sub  Else  dataClnt = dataClnt.Substring(0, dataClnt.IndexOf("$"))  message("Client - " + DateTime.Now + " " + Me.clName + " " + Me.cip + " " + "(" + Convert.ToString(cnum) + ")" + " : " + dataClnt)  Dim namelength As Integer  namelength = Me.clName.Length  If dataClnt.Contains("is sending a file") = True Then  fileflag = True  bcast(dataClnt, clName, False)  Else  fileflag = False  'msg2 = dataClnt  cmds()  If dataClnt = "/getusers" Then  msg2 = dataClnt  cmds()  bcast("Command typed: " + dataClnt, Nothing, False)  ElseIf dataClnt.Contains("/whois") Then  msg2 = dataClnt  cmds()  bcast("Command typed: " + dataClnt, Nothing, False)  ElseIf dataClnt.Contains("/myinfo") Then  msg2 = dataClnt  cmds()  bcast("Command typed: " + dataClnt, Nothing, False)  Else  bcast(dataClnt, clName, True)  End If  End If  If dataClnt.Contains("has changed his name to ") = True Then  Dim length As Integer  Dim copystring As String  Dim stringv2 As String = " has changed his name to"  copystring = dataClnt.Remove(0, namelength)  length = stringv2.Length  copystring = copystring.Remove(0, length - 1)  copystring = copystring.Replace(" ", "")  For i = 0 To userlist.Count - 1  If userlist.Item(i).ToString.Contains(Me.clName) Then  userlist.RemoveAt(i)  End If  Next  For i = 0 To Me.clList.Count - 1 'tofix  If clList.Contains(Me.clName) Then  clList.Remove(Me.clName)  clList.Add(copystring, cSocket)  End If  Next  Dim ctThread As Threading.Thread = New Threading.Thread(AddressOf initiateChat)  Me.clName = copystring  userlist.Add(copystring)  dataClnt = copystring  startClient(cSocket, copystring, clList)  ctThread.Start()  End If  End If  Catch ex2 As Exception  If cSocket.Connected = False Then  clntDisc = True  cSocket.Close()  cdisc()  Exit Sub  End If  End Try  End While  End Sub  Sub cdisc()  If clntDisc = True Then  message(clName + " " + "(" + Convert.ToString(cnum) + ")" + " " + "has disconnected. " + DateTime.Now)  'bcast(clName + " " + "(" + Convert.ToString(cnum) + ")" + " " + "has disconnected " + DateAndTime.Now, Nothing, False)  bcast(clName + " has disconnected. $", Nothing, False)  If Me.cnum = cnum Then  clList.Remove(Me.clName)  clList.Remove(Me.cip)  clList.Remove(Me.cnum)  clList.Remove(Me.cSocket)  userlist.Remove(Me.clName)  iplist.Remove(Me.cip)  End If  cSocket.Close()  clntDisc = False  Exit Sub  End If  End Sub  Sub cmds()  Dim whoisstr As String  If msg2 Is Nothing Then  Exit Sub  Else  Select Case msg2  Case "/getusers"  If msg2 = "/getusers" Then  Dim countclient As Integer = count - 1  Dim i As Integer = 0  While i <> countclient  Try  If i <> countclient Then  bcast("$Username: " & userlist.Item(i).ToString, Nothing, False)  Thread.Sleep(500) 'temporary fix?  i = i + 1  End If  Catch ex As ArgumentOutOfRangeException  Exit While  End Try  End While  End If  Case "/myinfo"  Dim getip As String  If msg2 = "/myinfo" Then  msg2 = ""  getip = Me.cip  bcast("IP is: " + getip + " ", Me.clName, False)  End If  End Select  If msg2.Contains("/whois") Then  whoisstr = "/whoisstr"  End If  Select Case whoisstr  Case "/whoisstr"  Dim copy As String  copy = msg2  msg2 = ""  copy = copy.Remove(0, 7)  Dim i As Integer  Dim test As Integer = count - 1  While i <> test  Try  If i <> test Then  If userlist.Item(i).Equals(copy) Then  Dim ip As String  ip = iplist.Item(i).ToString  bcast("Whois" + Environment.NewLine + "Username: " + copy + Environment.NewLine + "IP: " + ip, Nothing, False)  End If  i = i + 1  End If  Catch ex As ArgumentOutOfRangeException  End Try  End While  End Select  End If  End Sub  End Class  #End Region |

## Client

The client makes the connection to the server and allows chatting with other connected to the server (if they exist, of course.), also allows the sending and receiving files.

**Connect/Disconnect function** – This function handles the connection/disconnect to the user specified IP/Port

|  |
| --- |
| #Region "Connect/Dc button"  Private Sub cntbutton\_Click(sender As System.Object, e As System.EventArgs) Handles cntbutton.Click  If connB = True Then  Try  If info.Nickname = "" Or info.Nickname Is Nothing Or \_  info.IP = "" Or info.IP Is Nothing Or Not IsNumeric(info.IP) Then  MsgBox("Please review needed fields on preferences.")  Exit Sub  Else  cSocket = New System.Net.Sockets.TcpClient  msgthread()  cSocket.Connect(info.IP, info.Port) ' Connects to IP and port determined by svip and port.  servStream = cSocket.GetStream() ' Client to Server handling  Dim Stream\_out As Byte() = System.Text.Encoding.ASCII.GetBytes(info.Nickname + "$")  servStream.Write(Stream\_out, 0, Stream\_out.Length)  servStream.Flush()  Dim ctThread As Threading.Thread = New Threading.Thread(AddressOf getMsg) ' Client Threading  ctThread.Start()  cntbutton.Text = "Disconnect"  sndbtn.Enabled = True  connB = False  End If  Catch ex As SocketException  chattxt.Text = chattxt.Text + Environment.NewLine + " >> " + "Connection timed out! Check IP/port"  cntbutton.Enabled = True  sndbtn.Enabled = False  End Try  ElseIf connB = False Then  CheckForIllegalCrossThreadCalls = False  sndbtn.Enabled = False  cntbutton.Text = "Connect"  servStream.Flush()  cSocket.Close()  ListBox1.Items.Clear()  connB = True  CheckForIllegalCrossThreadCalls = True  End If  End Sub  #End Region |

**Message handling** – This function receives messages from other clients that are sent to the server, or messages from the server itself.

|  |
| --- |
| #Region "Get Message handle"  Private Sub getMsg()  Try  Do While True  If cSocket.Connected = False Then  CheckForIllegalCrossThreadCalls = False  chattxt.Text = chattxt.Text + Environment.NewLine + " >> " + "Disconnected from server!"  ListBox1.Items.Clear()  CheckForIllegalCrossThreadCalls = True  Exit Sub  Else  servStream = cSocket.GetStream() ' Server to Client Handling  Dim buffSize As Integer ' Buffer size  Dim Stream\_in(10024) As Byte ' Buffer size for BuffSize (also max bytes allowed by server.)  buffSize = cSocket.ReceiveBufferSize  servStream.Read(Stream\_in, 0, buffSize)  Dim returnd As String = System.Text.Encoding.ASCII.GetString(Stream\_in)  read = "" + returnd ' Data received from server.  msgthread()  End If  Loop  Catch ex As IOException  CheckForIllegalCrossThreadCalls = False  chattxt.Text = chattxt.Text + Environment.NewLine + " >> " + "Connection terminated!"  cntbutton.Enabled = True  cntbutton.Text = "Connect"  ListBox1.Items.Clear()  sndbtn.Enabled = False  CheckForIllegalCrossThreadCalls = True  Catch ex2 As ObjectDisposedException  CheckForIllegalCrossThreadCalls = False  'chattxt.Text = chattxt.Text + Environment.NewLine + " >> " + "Object not disposed properly!"  cntbutton.Enabled = True  sndbtn.Enabled = False  CheckForIllegalCrossThreadCalls = True  Catch ex3 As InvalidOperationException  CheckForIllegalCrossThreadCalls = False  'chattxt.Text = chattxt.Text + Environment.NewLine + " >> " + "Invalid Operation exception!"  cntbutton.Enabled = True  sndbtn.Enabled = False  CheckForIllegalCrossThreadCalls = True  Catch ex4 As Win32Exception  End  End Try  End Sub  #End Region |

**Message Handling & Multithreading** – This function deals with what is received in the messages and if any command to redirect to the right place, if not it’s treated as a common message.

|  |
| --- |
| #Region "Multithreading + Message Handling"  Private Sub msgthread()  If Me.InvokeRequired Then ' Thank you André for helping me out on this part.  Me.Invoke(New MethodInvoker(AddressOf msgthread)) ' Multithread  Else  Try  Dim Copystr As String  Dim copystrlength As Integer  'If read.Contains(info.Nickname) AndAlso read.Contains("joined") Then  ' ListBox1.Items.Add(info.Nickname)  'End If  If read.Contains("$Username: ") Then  read = read.Substring(11)  Copystr = read  copystrlength = Copystr.LastIndexOf("/getusers")  If Copystr.Contains("/getusers") Then  Copystr = Copystr.Substring(0, copystrlength)  End If  If ListBox1.Items.Contains(Copystr) Then  read = ""  Copystr = ""  Exit Sub  Else  ListBox1.Items.Add(Copystr)  read = ""  Copystr = ""  Exit Sub  End If  End If  If read.Contains("Server Operator has terminated your connection. $") Then  read = read.Substring(0, 47)  cntbutton.PerformClick()  ListBox1.Items.Clear()  End If  If read.Contains(" has disconnected. $") Then  Dim copyread As String = read  Dim copyreadlength As Integer = copyread.LastIndexOf("$")  Dim readlength As Integer = read.LastIndexOf(" has disconnected. $")  copyread = copyread.Substring(0, copyreadlength)  read = read.Substring(0, readlength)  Dim readcopy As String = read  For i = 0 To ListBox1.Items.Count - 1  If ListBox1.Items(i).ToString.Contains(readcopy) Then  ListBox1.Items.RemoveAt(i)  ListBox1.Refresh()  End If  Next  read = ""  chattxt.Text = chattxt.Text + Environment.NewLine + " >> " + copyread  Exit Sub  End If  If read.Contains("has changed his name to ") = True Then  ListBox1.Items.Clear()  msgtxt.Text = "/getusers"  sendBtn.PerformClick()  msgtxt.Text = ""  End If  If read.Contains("joined ") Then  ListBox1.Items.Clear()  msgtxt.Text = "/getusers"  sendBtn.PerformClick()  msgtxt.Text = ""  End If  Catch ex As NullReferenceException  End Try  chattxt.Text = chattxt.Text + Environment.NewLine + " >> " + read  End If  End Sub  #End Region |

## *Stresstester*

This program was developed using parts of the code of the Client and is used only for testing.

**Stresstest function** – This function performs the tests when the values ​​that are ordered are entered.

|  |
| --- |
| #Region "Stresstest Button"  Private Sub Button1\_Click(sender As System.Object, e As System.EventArgs) Handles Button1.Click  Dim usercount As Integer  If Not IsNumeric(TextBox1.Text) Then  MsgBox("Please insert a numeric value into the textbox.")  Else  Try  Do  Try  Dim nickname As String = "yo" & usercount  cSocket = New System.Net.Sockets.TcpClient  msgthread()  If String.IsNullOrEmpty(TextBox2.Text) Then  MsgBox("Please insert a value into the Port Textbox.")  Exit Sub  End If  cSocket.Connect("127.0.0.1", TextBox2.Text) ' Connects to IP and port  servStream = cSocket.GetStream() ' Client to Server handling  Dim Stream\_out As Byte() = System.Text.Encoding.ASCII.GetBytes(nickname + "$")  servStream.Write(Stream\_out, 0, Stream\_out.Length)  servStream.Flush()  usercount = usercount + 1  count = count + 1  Dim ctThread As Threading.Thread = New Threading.Thread(AddressOf getMsg) ' Client Threading  ctThread.Start()  Catch ex As SocketException  If MsgBox("Error performing Stresstest.") = DialogResult.OK Then  My.Computer.FileSystem.WriteAllText("StressTest " + FormatDateTime(DateAndTime.Today, DateFormat.LongDate) + ".log", "Stress test FAILED at: " + DateTime.UtcNow + Environment.NewLine, True)  Exit Sub  End If  End Try  Loop Until count = TextBox1.Text  MsgBox("Stress test completed." + Environment.NewLine + "Amount of users tested: " & usercount)  Catch ex As IOException  End Try  End If  End Sub  #End Region |

# Program history

0.1a:

* Idea organization, revised objectives and project start;
* Interface design;
* Server development started.

0.1b:

* Server at 50% completion;
* Client development started.

0.1c:

* Server now accepts more than one client;
* Fixed some exceptions;
* Client now connects to other places besides localhost;
* Multithread development started.

0.1d:

* File transfer development started;
* Some breakthrough on Multithread development.

0.1e:

* File transfer complete;
* Prototype for file transfer notification started;

0.2:

* Structure implemented to organize each user’s information;
* Preference window implemented;
* User command development started;
* Client list optimized;
* Whois command development started;

0.3:

* Server list manipulated by user through a XML file implemented;
* Whois system complete;
* Planning for PM system started;

0.4:

* PM system on hold due to full recode required;
* RSS feed reader added;
* Server optimization;
* Client list optimization;

0.5:

* Client list now removes disconnected clients;
* Stresstester added (WIP);
* Commands via Server added;

0.5a:

* Console input forced to startup;
* All “if” statements for server commands changed to a “Select” statement.

0.6:

* “Say” function implemented on Server;
* Nickname checked, partially implemented and deactivated at the moment. (if existant, automatically changes it.).

0.7:

* Automatic client list updater (Timer) implementation started;

0.8/0.9:

* Client list optimized;
* Nickname checker implemented;
* Client list updater implemented;
* Function to turn on/off client list timer on Server implemented;
* User status (Online, Away, Busy) function implemented;
* Client structure modification now allows user to remember old settings used;
* User-defined port option re-implemented;
* Stresstester updated;
* New function implemented on Server (locate)

Pre-1.0/a:

* Splash and about box added on Client

Pre-1.0b:

* Visual content added to Server and Client

Pre-1.0c:

* Verification of installed network cards function added to Server

Pre-1.0d:

* Improved exception handling.

Pre-1.0e:

* Improved nickname verification function.

Pre-1.0f:

* Added more visual material
* Updated Stresstester
* Info and about functions added to Server

Pre-1.0g:

* Implemented a ban/unban system on Server

# Technologies used

This project was developed using the framework VB.NET (Visual Basic.NET) present in Microsoft Visual Studio.

We used as a resource for testing the project, virtual machines at school using the Oracle VirtualBox and at home using VMWare.

We also used a Stresstester developed specifically to test the CPU load, RAM usage and maximum clients the server could support.

# Conclusion

This project was developed for evaluation as a Technological Project.

In our opinion, a project was a little difficult in some parts but the difficulties were quickly overtaken and rapid implementation of new revisions with new features and those that were not 100% complete.

We were introduced to the Threading system and Sockets over TCP / IP, things which were not given in the classroom and that were discussed and studied by ourselves, which gave us motivation to continue the development of this project.

It was a project that at the end of it, after so many hours nosing around, testing the code and finding bugs gave us a solid basis to continue with such motivation in the future.

# Gratitude

Basically, we would like to thank Prof. Manuel Tróia and Prof. Sandra Rodrigues for all the help and time that they spent on this project.

And also to our fellow classmates, who accompanied the development of it and gave some ideas to implement.