T.C.

HARRAN ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

PROJE ÖDEVİ

CLİENT-SERVER PROJESİ (SOCKET PROGRAMMİNG)

Önder Fatih BUHURCU

Mehmet Fatih TÜYSÜZ

ŞANLIURFA

2020

T.C.

HARRAN ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

PROJE ÖDEVİ

CLİENT-SERVER PROJESİ (SOCKET PROGRAMMİNG)

Önder Fatih BUHURCU

Mehmet Fatih TÜYSÜZ

ŞANLIURFA

2020

İÇİNDEKİLER

Sayfa No

ŞEKİLLER DİZİNİ

Sayfa No

TABLOLAR DİZİNİ

Sayfa No

SİMGELER DİZİNİ

TCP	Transmission Control Protocol
IP	Internet Protocol
TXT	Text

1. GİRİŞ

C# dili kapsamında bir ağ üzerindeki bilgisayarların birbiri ile iletişim kurarak bilgi alışverişi sağlanmaktadır. Socket yapısında haberleşmede Client-Server modelini kullanır.

Bu projede, Client-Server socket bağlantısı ve Multithereading konuları TCP yapısına göre bir socket uygulaması üzerinde kullanıcıların istediği işlem türüne göre toplama veya faktöriyel hesaplama ile ve yapılmasını istediği değerin işlem sonucunu tekrardan kullanıcıya iletilmesini, bilgi alışverişi yapmaları sağlanacaktır. Bu sayede Client işlem yaparak yorulmasını önlemek bunun için sadece Server gerekli yorulma işlemini gösterecektir.

Bu çalışmanın aşamaları Server tasarlanarak socket yapısı ve buna bağlanabilecek Client için ortam sağlanmıştır. Çoklu Client bağlantısını sağlanarak haberleşme bağımsızlığı için Multithreading yapısı kullanıldı. Birbirleri ile iletişim yapabilme ortamını kuran Server ve bu ortama bağlanan bir veya daha fazla Client şeklinde tasarlanmıştır. Client mesajlarını kavraması ve gereken şekilde cevap mesajını iletilmesi sağlandı.

2. YAPILAN ÇALIŞMA

2.1. Server

C# console uygulaması üzerinden Server'ın yapması gereken işlemler kodlanmıştır.

2.1.1. Socket bağlantı oluşumu

Bir Server ve bir veya birden fazla Client tasarımı yapıldı. Server tasarımının C# dili ile kodlanarak ilk socket tasarımı, socket üzerinden bağlantıların kabul edilmesi sağlandı. Socket sınıfı ile socket oluşturulmaktadır. Socket sınıfı oluşturulurken ilk IP adresi sonrasında socket tipini en sonda bağlantı protokol tipini belirlenmiştir.

```
Socket Serverlistener = new Socket(IPAddress.Parse("0.0.0.0").AddressFamily,
                    SocketType.Stream, ProtocolType.Tcp);
```

Socket yapısı oluşturulan Server'in socket sınıfının Bind() methodu ile socket yapısını çalıştırılmaktadır. Listen() methodu ile Server'in dinleme işlemi yani veri alabilme imkanı verilmiştir. Accept() methodu ile Client bağlanılması beklenmektedir.

```
Serverlistener.Bind(ip,port)
Serverlistener.Listen(1)
Serverlistener.Accept()
```

2.1.2. Multithread

Bağlanan Client sayısı bir veya birden fazla olacağı için Multithread yapısı oluşturulmaya başlandı. C# dilinin vermiş olduğu thread sınıfı oluşturulup, Start() methodu ile bağlanan Client için thread tasarlanıp başladı.

```
userThread.Start();
```

2.1.3. Socket mesaj alma

Client'in gönderdiği mesajı beklemeye başlayan ve mesaj geldiğinde bunu okumasını sağlayan socket sınıfının methodu olan Receive() sayesinde okundu.

```
byte[] islemclient = new byte[1024];

client.Receive(islemclient);
```

2.1.4. İşlem

Byte değerinde gelen mesajı string değere çevirisi yapıldı. Gönderilen mesajın ilk byte değeri işlem türünü belirtmektedir. Bu sayede gelen mesajın toplama veya faktöriyel hesaplama işleminin hangisinin yapılacağını bilen Server işlem fonksiyonlarından hangisi ise onu çağırmaktadır.

```
var deger = Encoding.ASCII.GetString(islemclient,0,size);

if (deger.Substring(0,1) == "1")

else if (deger.Substring(0, 1) == "2")
```

Toplama işleminde gelen mesajdaki ilk byte değeri çıkarılıp kalan byte değerlerini işleme alıp tek basamak sonucu elde edesiye kadar tekrar içinde çalışmaktadır. Tek basamak değerine ulaştıktan sonra "ok" mesajı göndermektedir.

```
Int64 sonuc = 0;
while (sayi > 0)
{
sonuc += (sayi % 10);
sayi = sayi / 10;
}
return sonuc;
```

Faktöriyel işleminde ise faktöriyeli alınacak değeri alan Server işlemi uygulayıp sonucu geri mesaj olarak Client'a iletilip sonrasında "ok" mesajı göndermektedir.

```
Int64 sonuc = 1;
for (int i = 1; i <= sayi; i++)
{
sonuc = i * sonuc;
}
return sonuc;
```

2.1.5. Socket üzerinden mesaj iletimi

Serverda hesaplanan değeri Client'a göndermek için Socket sınıfının Send() methodu kullanılmaktadır. Hesaplanan değer ilk olarak byte türüne çevrilir ve değerin uzunluğuda belirtilerek mesaj şeklinde iletilir.

```
client.Send(Encoding.ASCII.GetBytes(sonuc.ToString()), 0,
sonuc.ToString().Length, SocketFlags.None);
```

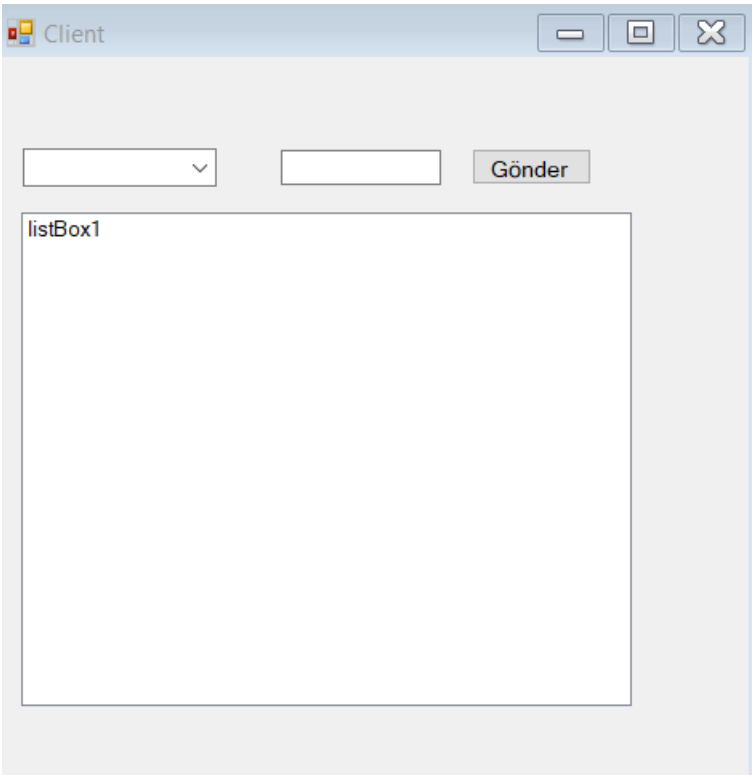
2.1.6. TXT dosyasına kayıt

Gereken bilgiler toparlanarak TXT dosya türüne kayıt yeri masaüstü olacak şekilde kayıt yapılması sağlanmıştır.

```
string baglantikayit=
System.Environment.GetFolderPath(Environment.SpecialFolder.Desktop).ToString()
+ "\\baglantikayit.txt";

FileStream fs = new FileStream(surekayit, FileMode.Append,
FileAccess.Write);

StreamWriter sw = new StreamWriter(fs);

sw.WriteLine("Host Adı=" + hostEntry.HostName + "\tIpv4 Adres=" +
hostEntry.AddressList[1] + "\tIpv6 Adres=" + hostEntry.AddressList[0] + "\tSistem
Saati=" + DateTime.Now.ToString() +"\tKullanıcının Yapmak İstediği işlem türü=
Faktöriyel Hesaplama" +"\tKullanıcı Gönderdiği değer=" + deger.Substring(1));

sw.Close();
```

2.2. Client

C# dili ile Windows form uygulaması yapıldı. Resim 2.1. ile tasarım gösterilmiştir.



Şekil 2.1. Form Tasarım

2.2.1. IP adresimizi belirleme

Servera bağlanabilmemiz için uygulamamızı çalıştırdığımız bilgisayarda IP adres bilgilerini buluyoruz. Kendi IP adresimize ek olarak Server'ın çalıştığı port numaramızıda giriyoruz.

```
IPHostEntry ipHostInfo;
IPAddress ipAddress;
IPEndPoint localEndPoint;
ipHostInfo = Dns.GetHostEntry(Dns.GetHostName());
ipAddress = ipHostInfo.AddressList[1];
localEndPoint = new IPEndPoint(ipAddress, 13000);
```

2.2.2. Socket üzerinden bağlantı kurma

Socket bağlantısı oluşturularak Server IP adresini de belirterek bağlantı oluşturulmaktadır. Socket sınıfının Connect() methodu ile IP adresi ve port bilgileri gönderilerek Server ile bağlantı başlamış olmaktadır.

```
Socket Serverlistener;

Serverlistener.Connect(localEndPoint);
```

2.2.3. Servera veri gönderme

Kullanıcıdan textboxa değer girilmesi ve yapılması istenen işlem türünün combobox sayesinde belirlenmesi ile gönder butonuna tıklanması sonucu değerler Servera iletilmektedir. Socket sınıfının Send() methodu ile veri gönderilmektedir. Gönderi zamanı alınarak cevap dönüş zamanının hesaplamasında kullanılacaktır. Listboxa gönderdiğimiz değeri ekleyerek işlem ilerlemesini rahatça görülebilmesi sağlanmaştır.

```
Serverlistener.Send(Encoding.ASCII.GetBytes(gonderi), 0, gonderi.Length,
SocketFlags.None);

listBox1.Items.Add("Client>" + textBox1.Text);

gonderizamani = DateTime.Now;
```

2.2.4. Backgroundworker ile server dinlemesi

Arka planda sürekli olarak Server dinlemesi sağlanarak gelen mesajlar listboxa eklenmektedir.

```
byte[] MessageFromServer = new byte[1024];

int size = Serverlistener.Receive(MessageFromServer);

listBox1.Items.Add("Server>" +
Encoding.ASCII.GetString(MessageFromServer, 0, size));
```

2.2.5. Backgroundworker ile gelen sürenin hesaplanması

Gelen mesaj içerinde ok yazısı araması yapılarak eğer ok mesajı var olduğunda yeni bir zaman tanımına eşitlenerek gelen zamandan gönderilen zaman farkı alınmıştır.

```
if (Encoding.ASCII.GetString(MessageFromServer).IndexOf("ok") >= 0)

donuszamani = DateTime.Now;
```

2.2.6. TXT dosyasına bilgi kayıt

Gereken bilgiler toparlanarak TXT dosya türüne kayıt yeri masaüstü olacak şekilde kayıt yapılması sağlanmıştır.

```
string surekayit =
System.Environment.GetFolderPath(Environment.SpecialFolder.Desktop).ToString()
+ "\\sürebilgi.txt";

FileStream fs = new FileStream(surekayit, FileMode.Append,
FileAccess.Write);

StreamWriter sw = new StreamWriter(fs);

sw.WriteLine("Gönderilen Değer="+textBox1.Text+"\tGönderilme zamanı="
+ gonderizamani.ToString("yyyy-MM-dd HH:mm:ss.fff") + "\tCevabın Verilme
Süresi=" + donuszamani.ToString("yyyy-MM-dd HH:mm:ss.fff")+"\tSüre
Farkı="+Convert.ToString(donuszamani - gonderizamani));

sw.Close();
```

3. BULGULAR

Yapılan testlerde elde edilen bulgular socket programlama ile Server -Client haberleşmesinde gecikme olmadığı ama mesaj okuma kısmında kod çalıştırmada oluşan milisaniyelik gecikmeler oluştuğu sonucuna varıldı. Server okuma zamanı ve Client gönderme zamanları Tablo 3.1.'de verilmiştir.

Tablo 3.1. Mesaj gönderim zaman ve okuma zaman farkı

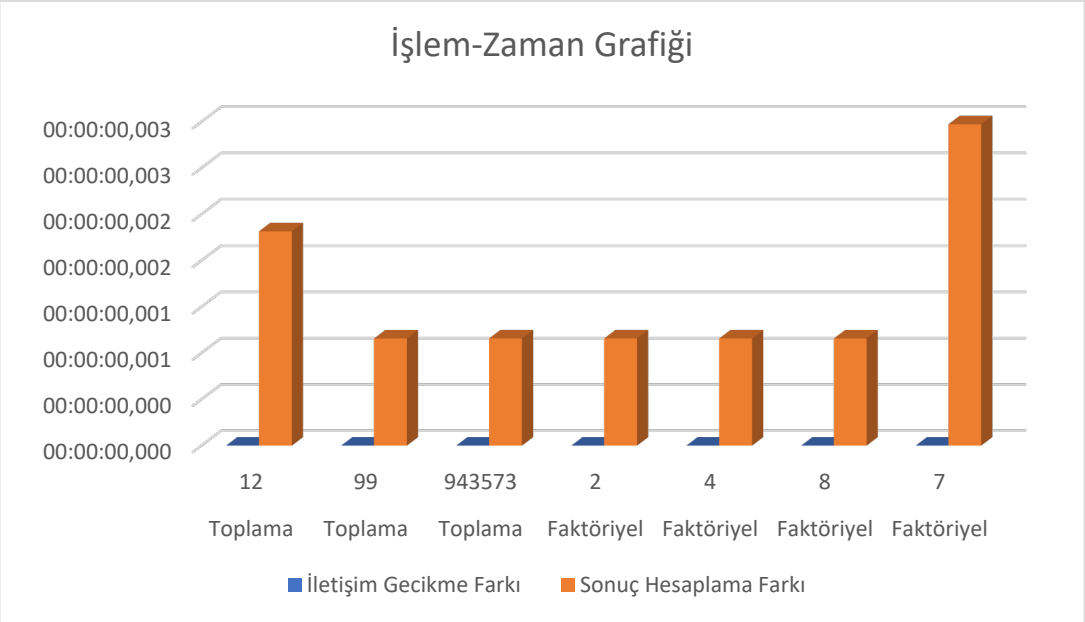
Değer	İşlem Türü	Client Gönderi Zamanı	Server Okuma Zamanı	Zaman Farkı
12	Toplama	2020-05-14 23:46:56.465	2020-05-14 23:46:56.465	0
943573	Toplama	2020-05-14 23:47:13.531	2020-05-14 23:47:13.531	0
8	Faktöriyel	2020-05-14 23:47:26.187	2020-05-14 23:47:26.187	0

Client mesajı gönderme ve sonucun gelmesi zaman farkı ise iletim hızında yavaşlama olmadığı için anlık iletişim sağlandığı, hesaplama ve kodların çalıştırılması süresi sonucu, mesajın gittiği zaman ve mesajın sonuncunun geldiği zaman farkı tablo 3.2.'de verilmiştir.

Tablo 3.2. Mesaj gönderilme ve hesaplanan sonuç dönüş zaman farkı

Değer	İşlem Türü	Client Mesaj Gönderi Zamanı	Client Sonuç Okuma Zamanı	Zaman Farkı
12	Toplama	2020-05-14 23:46:56.465	2020-05-14 23:46:56.466	00:00:00.0010012
943573	Toplama	2020-05-14 23:47:13.531	2020-05-14 23:47:13.532	00:00:00.0009951
8	Faktöriyel	2020-05-14 23:47:26.187	2020-05-14 23:47:26.188	00:00:00.0010188

Zaman farkının görüldüğü üzere iletişim süresinden olmadığı sadece hesaplama kodlarının sonucu bulmak için çalışma süresinin olduğu görülmüştür. Şekil 3.1.'de detaylı grafik gösterilmiştir.



Şekil 3.1. İletişim gecikmesi ve sonuç hesaplaması zaman farkı sonuç

4. SONUÇLAR VE ÖNERİLER

Yapılan proje sonucunda iki bilgisayar üzerinden C# dili ile TCP haberleşme sağlandı. Bu sayede Client bilgisayarın gönderdiği değerler Server bilgisayar üzerinden okunarak gereken işlemleri yapabilmektedir. İşlem süresinin sonucu ile aynı bilgisayar üzerinde haberleşme yapıldığından kodların işlenmesi süresi hesaplanabilmektedir. İletişim sürecinde gecikme yaşanmamaktadır. İletişim süresi uzun olduğu zamanlarda oluşabilecek 2 işlem gönderildiğinde sonuçları farklı farklı nasıl döndürmesidir. Server iletişimi kodlardan daha hızlı çalıştığı için gelen mesajları arka arkaya eklenmiş şekilde Client okumaktadır. Kodların bu iletişim hızına yetişememesi, iletişimden daha yavaş çalışması, ortalama tüm hesaplama işlemlerinin 1 milisaniye içerisinde gerçekleştiği görülmektedir.

KAYNAKLAR

Arat, 26.12.2018: https://berkarat.com/c-socket-programlama/

Bob, 10.04.2012: https://www.codeproject.com/Articles/12286/Simple-Client-server-Interactions-using-C

C# Network Programming Sockets (Asynchronous Client), 29.05.2018: https://www.youtube.com/watch?v=LV45r8BH98Y

C# simple socket tutorial, 04.09.2016: https://www.youtube.com/watch?v=KxdOOk6d_I0

C# Socket Programming - Multiple Clients, 09.03.2013: https://www.youtube.com/watch?v=xgLRe7QV6QI

C# Tutorial - TCP/IP Client Server | FoxLearn, 27.01.2017: https://www.youtube.com/watch?v=ve2LX1tOwIM

Client (and Server) Sockets Communication, 12.05.2020: https://www.winsocketdotnetworkprogramming.com/clientserversocketnetworkcommunication8b.html

Client Server programming in C# (Chat application), 11.03.2017: https://www.youtube.com/watch?v=X16IyNbcAr0

Dezio, 10.05.2020: https://www.geeksforgeeks.org/socket-programming-in-c-sharp/

Gökalp, 04.06.2015: https://www.gokhan-gokalp.com/c-ile-asenkron-socket-programlama/

How to C# Socket programming, 10.05.2020: http://csharp.net-informations.com/communications/csharp-socket-programming.htm

Network stream and multiple connections, 08.06.2012: https://stackoverflow.com/questions/10954383/network-stream-and-multiple-connections

NetworkStream Class, 11.05.2020: https://docs.microsoft.com/en-us/dotnet/api/system.net.sockets.networkstream?view=netcore-3.1

Server Client send/receive simple text, 16.04.2012: https://stackoverflow.com/questions/10182751/server-client-send-receive-simple-text

Socket Programming In C#, 07.06.2019: https://www.c-sharpcorner.com/article/socket-programming-in-C-Sharp/

Synchronous Client Socket Example, 30.03.2017: https://docs.microsoft.com/tr-tr/dotnet/framework/network-programming/synchronous-client-socket-example