Journal for øvelserne 8 & 9

soæidjgæo isdjoi jdsfæogij fdoi

Bjørn Nørgaard Sørensen
stud.nr: 201370248
bjornnorgaard@post.au.dk

Joachim Dam Andersen
stud.nr: 201370031
mr.anderson@post.au.dk

Arni Thor Thorsteinsson
stud.nr: 201370318
201370318@post.au.dk

October 26, 2015

Indholdsfortegnelse

1. Øvelse 8 - TCP/IP socket programming

1.1. Introduktion

I denne øvelse i socket programmering, laves en TCP client og TCP server. Vi har valgt at programmere i det objektorienterede sprog C#. Clienten skal kunne forbinde til serveren, og downloade en fil herfra. Clienten og serveren køres på hver sin virtuelle linux maskine. I dette dokument beskrives udviklingsforløbet med tilhørende digrammer og kodeforklaringer.

1.2. Udviklingsforløb

Vi har designet koden således at server og client er opdelt i to hoveddele, hhv. constructor og overføringsfunktionalitet. For serveren betyder dette afsendingsfunktionalitet, og for clienten modtagelsesfunktionalitet. På kodeudsnit: 1 Hoveddesign for server og 2 Hoveddesign for client, kan psudokoden for metoderne ses. Ønsker nærmere gennemgang af kode, kan source koden findes i medfølgende src.zip fil.

Code 1: Hoveddesign for server

```
1  public FileServer()
2  {
3      //Setting up server and connecting client
4      //Getting filename an calculating lenght
5      //Sending file
6      //Closing connection
7  }
8  public void SendFile(string filename, long filesize, NetworkStream stream)
9  {
10     //Local variables
11     //Assigning variables
12     //Sending file
13     //Closing connection
14 }
```

Code 2: Hoveddesign for client

```
1  public FileClient()
2  {
3      //Declaring variables
4      //Assigning variables
5      //Requesting and receiving file
6      //Closing connection
7  }
8  public void ReceiveFile(string filename, NetworkStream stream)
9  {
10     //Declaring variables
11     //Assigning variables and setting up fileStream
12     //Receiving data
13     //Closing connection
14 }
```

1.3. Funktionalitet

Vores design gør det muligt for brugeren at selv indtaste en ønsket TCP servers IP-addresse. Dette giver mulighed for yderligere udvidelse af programmets funktionalitet. Der oprettes nu forbindelse til serveren med den indtastede addresse. Herefter venter clienten på at brugeren indtaster et navn på den fil der ønskes downloadet fra serveren. Clienten anmoder da serveren om den specifikke fil, hvorefter serveren melder tilbage. Hvis filen eksisterer påbegyndes overførslen. Der er udarbejdet en sekvensdiagram der illustrer det overordnede system. Se dette på figur 3 på side 4.

1.4. Resultater

Vi har testet vores system og vedlagt screenshots heraf. På figur 1 ses test af serveren, hvor der sendes to filer. På billedet ses filer til afsendelse, som ligger i fileserverens Debug folder (markeret med lyserød). Billedet illusterer desuden programflowet med konsoludskrifter.

På figur 2 ses test af clienten fra samme testsekvens som på figur 1. Her modtages to filer, hhv. Herp.jpg og Derp.jpg. Igen illustreres programflowet med konsoludskrifter.

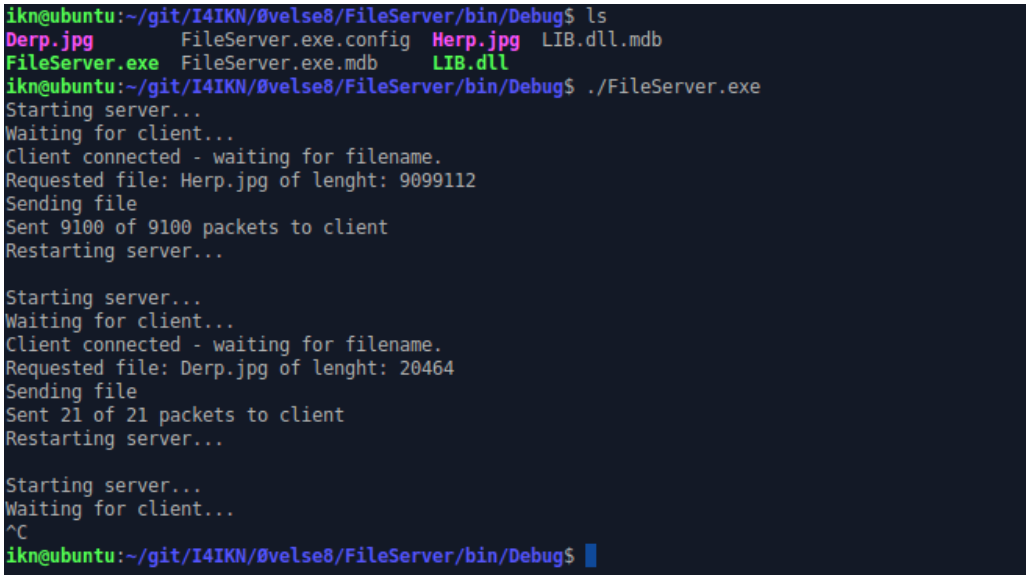

Figure 1: Test af TCP server/client - billede fra server.

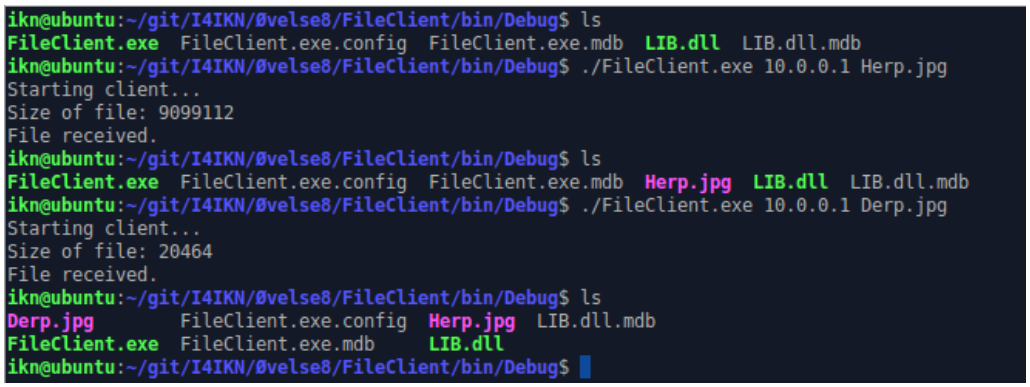

Figure 2: Test af TCP server/client - billede fra client.

1.5. Konklusion

I arbejdet med TCP socket programmering er vi kommet frem til en læsning der opfylder kravene givet i opgaven. Det kan derfor konstateres at teorien stemmer overens med praksis.

2. Øvelse 9 - UDP/IP socket programming

2.1. Introduktion

2.2. Udviklingsforløb

2.3. Funktionalitet

2.4. Resultater

2.5. Konklusion

A. Appendix: Sekvens diagrammer

Figure 3: Sekvensdiagram for TCP server/client - uden Send og Receive methoder