Objekt Orienteret Programmering
Eksamens projekt
Gruppe 3

Martin Gunneskov
Jonas Jermiin Ravn Moll

Generelt system til håndtering af distribueret kommunikation mellem maskiner, med instant messaging som særtilfælde.

Formål

At udvikle et system, i Java, til at håndtere kommunikation mellem distribuerede maskiner. Vi vil lægge vægt på en opbygning med generiske komponenter, således at systemet uden kodemodifikation kan løse flere forskellige opgaver indenfor distribueret digital kommunikation.
Specifikt vil vi designe systemet med baggrund i objekt orienterede designprincipper, som f.eks. nedarvning, delegering, abstrakte klasser samt grænseflader, hvilket vil gøre de enkelte dele af systemet lette at udvide, ændre eller helt at udskifte. Systemet kan altså håndtere kommunikation mellem maskiner, hvad enten det er lyd-, grafik- eller tekstbaseret.
For praktisk at dokumentere systemets anvendelighed, vil vi som et særtilfælde udvikle en applikation der benytter systemet til IM (instant messaging).

System definition

Med system mener vi en samlet pakke, der umiddelbart følger en generel server/klient struktur, med en hoved server der udgør kernen af systemet og med klienter der dels kan kommunikere med serveren, men også direkte med hinanden.
Serveren skal kun varetage informative opgaver, såsom at lagre adresserne på de forskellige tilsluttede klienter, klienternes status, information om antallet af online klienter og lignende.
Ideen er at systemets klienter forbindes direkte med hinanden, som i en P2P (peer to peer) løsning.
Ønsker en klient at udveksle data med en anden klient, forespørges serveren om klientens status og eventuelle adresse. Derefter etablerer den forespørgende klient selv en direkte forbindelse til den ønskede anden klient.
Herved varetager klienterne selv netværkstrafikken, og serveren fritages for unødig trafik og arbejde, der ellers skulle distribueres til de enkelte klienter.

Frameworks

Vi vil bl.a. benytte følgende frameworks i udviklingen:

RMI/Sockets

Det vil komme an på en vurdering fra vores side, om vi vælger at implementere systemet med Javas RMI teknologi eller om det bliver en socketbaseret løsning der håndterer kommunikationen mellem klienter og server.

Threads
I håndteringen af flere samtidige klienter, samt forespørgsler og processer på serveren.

I/O inkl. generelle datastrømme
Ind- og udlæsning til og fra de forskellige sockets. Samt ved eventuel logføring.

Swing
Til opbygning af grafiske brugergrænseflader, bl.a.ved implementation af IM klienter.

Krav

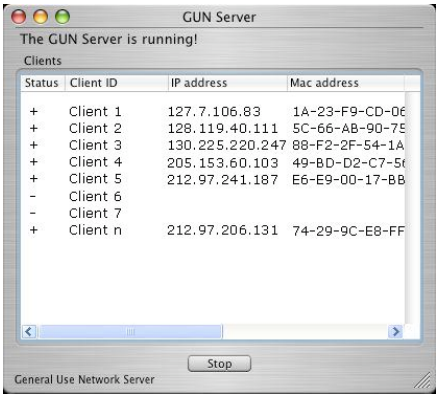
Generelt
Al kommunikation mellem server og klienter håndteres med socket-objekter.
Alle komponenter implementeres som applikationer.

Server delen

- Serveren starter sine services, når server applikationen initialiseres.
- Serveren skal konstant være i stand til at håndtere multiple klienters anmodning om en forbindelse, samtidig med løbende at opdatere status på online/offline klienter.
- Serveren kan højst håndtere en backlog på 50 klienter.
- Lagre information om tilknyttede klienter, ved klient ID. Også imellem at serveren startes og stoppes.
- Holde information om tilknyttede klienter (v. MAC/IP-adresse samt klient ID).
- Holde referencer mellem klient ID og IP-adresse.
- Mulighed for at oprette og slette klienter, ud fra deres ID.

Server GUI

- Der skal være mulighed for, at starte og stoppe den service serveren udbyder, ved tryk på en enkelt knap (Toggle button).
- Server status vises øverst i applikations vinduet.
- Se en liste over oprettede klienter, deres online status, klient ID, samt eventuelle IP- og MAC-adresse. Listen udstyres med scrollbar, så vinduets størrelse ikke direkte afhænger af online status.
- Mulighed for at slette en given klient, ved højreklik på denne i listen.

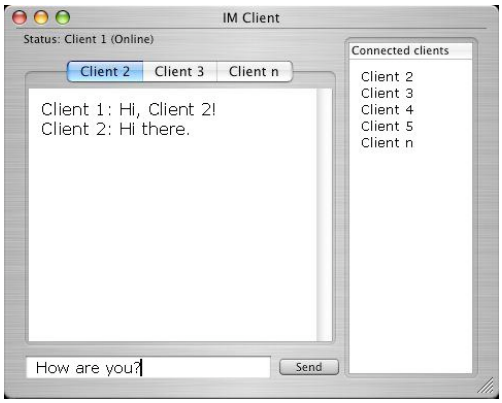


Klient delen

- Man registreres som online, når applikationen initialiseres.
- Mulighed for at afsende og modtage data.
- Mulighed for at angive kommunikations/datatype.
- Oplysning om antal online klienter.
- Registrere dato og tidspunkt for afsendelse/modtagelse af data.
- Hvem sender/afsender.
- Anmode om at blive oprettet eller slettet som klient.

- Klient skal samtidig med at kunne omtræde som en reel klient, også implementere en server struktur.
- Skal kunne håndtere flere samtidige forbindelser.

IM klientdel (GUI)



- Brugerens status vises øverst i vinduet.
- Se en liste over online klienter repræsenteret med deres klient ID. Listen er udstyret med scrollbar.
- Mulighed for med dobbeltklik på klient ID at anmode forbindelse til vedkommende.
- De klienter der er skabt forbindelse til, inddeles med klient ID i faneblade med hver deres ikke editerbar tekstfelt. Her vises samtalen.
- Nederst et tekstfelt. Afsending af tekststreng sker ved tryk på returtasten eller på knappen ”Send”.