Mutual Backup System

Av

André Håvak

Huy Nguyen

Masteroppgave i Informasjons- og Kommunikasjonsteknologi

Høgskolen i Agder
Fakultet for teknologi

Grimstad

mai 2007

Sammendrag

I de senere årene har det kommet nye teknologier for backup og behovet for sikkerhetskopiering er i økende vekst. Både privatbrukere og bedrifter har viktige data som må tas kopier av på en sikker måte og oppbevares på et sikret sted. De fleste bedrifter har gode nok ressurser til å lagre sine data via online backup eller andre løsninger som for eksempel SAN. Privatbrukere har ikke de samme lagringsbehovene, og derfor kan online backup være kostbart for lagring av små mengder data.
Vår oppgave går ut på å forske på eksisterende løsninger for sikkerhetskopiering og teknologier som kan benyttes for å utvikle et online backupsystem. Systemet vi skal utvikle er et 2peer backupsystem, hvor to brukere kan ta sikkerhetskopiering mot hverandre.
Backup er viktig for å hindre at data går tapt når man for eksempel får en systemkrasj, utsatt for tyveri eller brann. Backup av data blir i dag ofte lagret på CD-er, harddisker, minnepenn osv, noe som gjør at disse dataene ikke alltid er tilgjengelig ved slike hendelser. Med et 2peer backupsystem er data tilgjengelig for gjenoppretting hvor som helst, forutsatt at den som oppbevarer kopien er tilkoblet.
For å sette oss i stand til å utvikle et backupsystem må vi utforske metoder for backup ved å studere teori i fagbøker, artikler og online leksikon. Med den teoretiske kunnskapen vi har tilegnet oss etter litteraturstudie, kan arbeidet med design av backupsystemet begynne. Dette innebærer å lage en kravspesifikasjon og et systemdesigndokument.
De funnene vi har gjort i litteraturstudie og i designet av systemet, har vi utviklet en prototyp som viser prinsippet for online backup. Prototypen er blitt testet på lokalt nettverk(LAN), og tester viser at sikkerhetskopiering og gjenoppretting av store og små filer er vellykket.

Forord

Denne masteroppgaven er vår avsluttende oppgave ved Masterstudiet i Informasjons og kommunikasjonsteknologi ved Høgskolen i Agder (HiA), fakultetet for teknologi i Grimstad. Prosjektoppgaven ble gitt av Høgskolen i Agder uten involvering av et eksternt firma. Arbeidet startet med teori og oppgavebeskrivelse i kurset IKT 505 Metodeseminar høsten 2006. Selve arbeidet med masterprosjektet startet i januar 2007 og ble avsluttet i mai 2007.

Vi vil gjerne takke vår veileder Ulf Carlsen for å ha laget en spennende oppgave, samt god veiledning og de gode tilbakemeldingene vi har fått i løpet av prosjektperioden, noe som ga oss mye inspirasjon og motivasjon til å jobbe med dette prosjektet.

Huy Phat Nguyen

André Håvak

_____

_____

Grimstad, 29. mai 2007

Innholdsfortegnelse

Figurliste

Tabelliste

1 Innledning

1.1 Oppgavebeskrivelse

En enkel og billig måte å utføre sikkerhetskopiering på vil være å kopiere filene til en annen PC over internett [48]. For at dette skal være rettferdig må sluttbrukerne kunne utføre sikkerhetskopiering til hverandres PC-er. Denne oppgaven går ut på å utforske teknologier som gjør det mulig å utføre internettbasert backup gjennom utvikling av en prototyp eller proof of concept som demonstrerer gjensidig backup av filer mellom to peers.

1.2 Bakgrunn

Det finnes i dag flere løsninger for hvordan man kan foreta sikkerhetskopiering. Det brukes i dag for eksempel harddisk, minnepenn, optisk disk og minnekort for å utføre sikkerhetskopiering av filer. Disse er hardwarebaserte løsninger og data det er tatt kopi av er ikke alltid tilgjengelig. I de senere årene har det kommet flere systemer for online backup hvor man lagrer kopier av data hos en ekstern server. Online backup er ofte kostbart, og man må leie lagringsplass. En rimeligere måte å utføre sikkerhetskopiering på vil være å lagre filene på en annen PC, hvor man slipper kostnadene ved leie av lagringsplass. Med denne løsningen trenger man kun å installere en programvare, registrere seg mot en server og legge til den man ønsker å kopiere filer til. Filer man vil ta kopi av blir overført til en PC hvor man kjenner eieren.
Vi fordyper oss innen sikkerhet, og dette prosjektet gir oss en mulighet til å sette oss inn i forskjellige sikkerhetsløsninger ettersom dette systemet krever stor grad av sikkerhet. Dette prosjektet gir også en mulighet til å lære oss klient/tjener programmering. Et slikt system har også utviklingspotensialet utover prosjektperioden. Dette er en forholdsvis ny idé og denne formen for sikkerhetskopiering gir mange nye muligheter for både lagring av data og fildeling.

1.3 Målet med oppgaven

Målet med oppgaven er å sette oss inn i hvilke teknologier for backup som finnes i dag og utvikle et eget online backupsystem. Vi skal lage en kravspesifikasjon og en systemdesign som beskriver hvordan systemet skal se ut. Ut i fra designdokumentet skal vi utvikle en prototyp som beviser at et slikt system fungerer.

1.4 Problemområdet

For å kunne utvikle et backupsystem er det viktig å studere de backupteknologier som finnes i dag, slik at vi kan analysere dem og deretter finne en løsning som er best egnet i vårt system. I designet av systemet er det spesielt to ting som er viktig å ta hensyn til, nemlig brukervennlighet og sikkerhet. For at et slikt system skal bli tatt i bruk er det viktig å designe et oversiktlig GUI, slik at brukeren av systemet enkelt kan navigere seg fram. Sikkerhet er et annet viktig tema i et slikt system. All overføring foregår over et usikret nettverk, og det er viktig å sikre overføringen slik at ikke andre kan se data som overføres. Dette er spesielt viktig da det ofte er sensitive data man ønsker å foreta sikkerhetskopiering av. Prototypen skal inneholde de mest nødvendige funksjonene for å kunne utføre sikkerhetskopiering mellom to peers. For at det senere skal være enkelt å legge til nye moduler, er det viktig å gjøre kildekoden så adaptiv som mulig. Dette innebærer oversiktlig koding, og at metodene er strukturert og oversiktlig bygd opp.

1.4.1 Hovedproblem

Hovedproblemet i denne oppgaven er å klassifisere hvilke nødvendige funksjoner som må inkluderes for å gjøre det mulig å utføre sikkerhetskopiering og gjenoppretting. Det er viktig å kartlegge hvilke funksjoner vårt system skal inneholde ettersom dette avhenger av om brukerne vil bruke backupsystemet. Prototypen som skal utvikles skal innholde de mest nødvendige funksjonene for å vise prinsippet med systemet. De resterende funksjonene som ikke blir implementert i prototypen vil bli satt til fremtidig arbeid. Det er derfor viktig å lage en kildekode som lar seg lett videreutvikle.

1.4.2 Delproblemer

Det er fire delproblemer vi tar sikte på å løse i denne oppgaven. De to første delproblemene skal vi løse, og de to siste delproblemene vil vi løse om vi får tid. De delproblemene vi vil se på er følgende:

- Det første delproblemet er å lage en brukervennlig GUI i vår prototyp. Et brukergrensesnitt med mange knapper, bilder og funksjonalitet kan medføre at brukere blir forvirret av programmet. Her skal brukergrensesnittet være enkelt, leselig og oversiktlig slik at det skal være lett for enhver bruker å skjønne hvordan programmet skal brukes til å ta backup og gjenopprette filer.

- Det andre delproblemet vi tar sikte på å løse, er hvordan man håndterer statusen til den enkelte bruker og hvordan denne kan videresendes til de som har denne brukeren i sin brukerliste. Dette er nødvendig å løse slik at man sender data til riktig peer, og at man vet om denne brukeren er logget på eller ikke.

- Det tredje delproblemet er å sikre overføring av data og brukerinformasjon som er lagret hos klient og server. Dette er nødvendig å løse slik at integriteten til brukerne blir ivaretatt og data ikke blir tilgjengelig for andre enn eieren selv.

- Det siste delproblemet vi skal løse er hvordan vi skal håndtere rutere. Når en maskin er bak en ruter, vil den som oftest være konfigurert med en IP adresse som ikke er gyldig på internett. Dette gjør at når en pakke kommer inn til en ruter må ruteren vite hvem på innsiden denne pakken skal til. For at systemet skal være brukervennlig, er det viktig å ha et system som håndterer dette for brukerne.

1.5 Problemavgrensning

Vi begrenser oss til å se på en løsning hvor kun to peers kommuniserer med hverandre samtidig etter at de har logget seg på serveren. Overføring av filer foregår ikke via serveren. Sikkerhetskopiering av filer skal kun distribueres til en peer, og det vil ikke bli aktuelt å forske på hvordan man fordeler filer over flere peers slik som i et P2P nettverk.
Prototypen skal bli utviklet for Windows plattform og programmeringsspråket vi skal benytte er C# med .NET framework. Dette er et språk vi tidligere ikke har vært borti og vi ser på dette prosjektet som en mulighet til å lære oss et nytt språk. Vi har valgt en oppgave som er noe utenom vår fagkompetanse. Vi har fordypning innen sikkerhet, og vi ser på denne oppgaven som en mulighet til å kombinere sikkerhet og systemutvikling. Dette har vi valgt å gjøre for å få et bredere kompetanseområde.

2 Litteratur

2.1 Backup

Backup i informasjonsteknologi betyr å lagre kopier av data som i et senere tidspunkt kan gjenopprettes [1, 43, 44]. Et par eksempler hvor det er nyttig å lagre kopier av data er å kunne gjenopprette en datamaskin tilbake til driftsmessig tilstand etter en systemkrasj, eller gjenopprette filer etter at de har blitt slettet ved uhell. Andre årsaker som strømbrudd, brann og naturkatastrofer kan medføre til at systemet krasjer. For å sikre seg mot eventuelle tap av viktige og/eller kostbare data, er det derfor nødvendig å lagre kopier av disse dataene jevnlig på et annet sted enn på den PC-en man bruker daglig. Man kan for eksempel dele opp harddisken i flere partisjoner, ha to harddisker på en PC eller en ekstern harddisk, lagre kopier av data på forskjellige lagringsmedier (for eksempel CD, DVD, USB penn, osv) eller foreta online backup. På denne måten kan man gjenopprette alle eller deler av dataene hvis det skjer noe galt med de originale dataene på PC-en.

Å bestemme hva som skal tas kopier av er opptil den enkelte person. Vanligvis er det viktig å ta vare på data som ikke er lett å erstatte. Det kan være nødvendig å skrive opp en liste over filer som man selv syns er viktig og uerstattelig. Her er noen eksempler på hva som kan ha verdi for forskjellige brukergrupper [2]:

- Dokumenter og finansinformasjon
- Digitale bilder
- Filmer og musikk som er kjøpt og lastet ned fra internett
- Programvarer som er kjøpt og lastet ned fra internett
- Bokmerker lagret på nettleseren
- E-post adressebok
- Telefonbok
- Programvare/systemoppsett på maskinen

Man må også tenke på hvilken kostnad og/eller verdi hver enkelt type data har. For eksempel filmer og musikk koster mye for å laste ned, men de kan kjøpes igjen. Viktige dokumenter som prosjektarbeid, regnskapsdata, private digitale bilder, osv er uerstattelige. Disse filene er viktigere å lagre kopier av.

2.1.1 Backupmetoder

For å kunne lagre kopier av data må man ha et oppbevaringssted for data. Lagring av data trenger å bli lagret og organisert på en måte slik at de kan gjenopprettes ved en senere anledning. Forskjellige måter for oppbevaring av data har sine fordeler og ulemper [3, 45].

2.1.1.1 Full backup

Full backup vil si at man lagrer kopier av alle filer på systemet. Dette er en stor fordel hvor det sikrer at alle filer blir lagret. Typisk blir det kun tatt full backup ved en installasjon av for eksempel en tjener. Ved senere tid er det kun behov for full backup ved større endringer i systemet. Fordelen med en slik type backup er at man ved systemkrasj kan gjenopprette systemet til en tilstand før systemkrasj. Dette er en mer effektiv måte å gjenopprette på enn å installere alle applikasjoner på nytt. Generelt vil det bli lagret applikasjonsdata ved full backup av systemet. Ulempen med en slik type backup er at det tar mye plass på backupmediet etter hvert som mengden av data øker. Det er også tidkrevende å ta full backup, ettersom dette innebærer store mengder data.

2.1.1.2 Inkrementell backup

Inkrementell backup vil si å lagre kopier av data som er endret siden forrige backup. Denne type backup kan bli utført enten ved å overskrive filene som er endret, eller ved å tilføye endringer til eksisterende data. Fordelen med en slik type backup er at det tar kortere tid å utføre backup siden det er kun de siste filene som er blitt endret som blir kopiert. Ulempen er at det tar lenger tid for gjenoppretting slik som siste full backup, og all inkrementell backup fram til siste datatap trenger å gjenopprettes.

2.1.1.3 Differensial backup

Differensial backup vil si å lagre kopier av data som er endret siden forrige **fulle** backup. Dette innebærer full, inkrementell og differensial backup. En slik type backup gjør at man kan finne fram til en fil i en backup i hvilket som helst tidspunkt mellom nåværende backup og forrige fulle eller inkrementelle backup. Man kan da gjenopprette filer som er enten slettet, endret eller ødelagt på et bestemt tidspunkt. Fordelen med en slik type backup er at det bruker mindre ressurser og er mer effektiv enn andre typer backup. Ulempen er at det krever mer lagringsplass enn full eller inkrementell backup, noe som gjør at det er viktig å ha gode backuprutiner.
På figur 2.1 viser forskjellen mellom backupmetodene:

Figur 2.1: Forskjellen mellom backupmetodene

2.1.1.4 Backuprutiner

Når det skal lagres kopier av filer i et system, er det viktig med gode backuprutiner. Dette for at man skal kunne lagre kopier jevnlig, og man slipper å miste data som man har jobbet med over lengre tid. En god rutine for å foreta sikkerhetskopiering på er Grandfather-Father-Son prinsippet [4]. Det vil si at man lagrer kopier av data daglig (son), ukentlig (father) og månedlig (grandfather). Man har da et backupmedia for hver dag i uken (f.eks. 7 backuptaper, en for hver dag). Tapene kan merkes ”mandag”, ”tirsdag”, ”onsdag” osv. Hver dag tar man inkrementell backup slik at alle filer som er endret siden siste fulle backup blir tatt kopier av. Når en ny uke er kommet kan man gjenbruke tapene og ta ny inkrementell backup hver dag. En dag i uken tar man full backup av systemet, og man kan ha en tape per uke. Når måneden er ferdig blir disse tapene brukt om igjen og den ukentlige backupen blir skrevet over den gamle. Den månedlige backupen blir tatt den siste dagen i måneden. Her blir det tatt full backup av hele systemet. Her vil backupmediet bli tatt vare på, og en ny backuptape behøves for hver måned da disse ikke skal gjenbrukes. Man kan også ytterligere ta en full backup som er gjeldende for et helt år. Dette er en backupstrategi som er mye brukt, og egner seg godt for både private personer og bedrifter. Med denne strategien unngår man å tape data som er nyere enn 1 dag gammel. Om man ikke har lagret kopier av data på en uke og harddisken skulle krasje, vil det ta lang tid å ta igjen dette arbeid. Dette vil koste en bedrift både penger og kunder, noe som ikke er heldig.

2.1.2 RAID

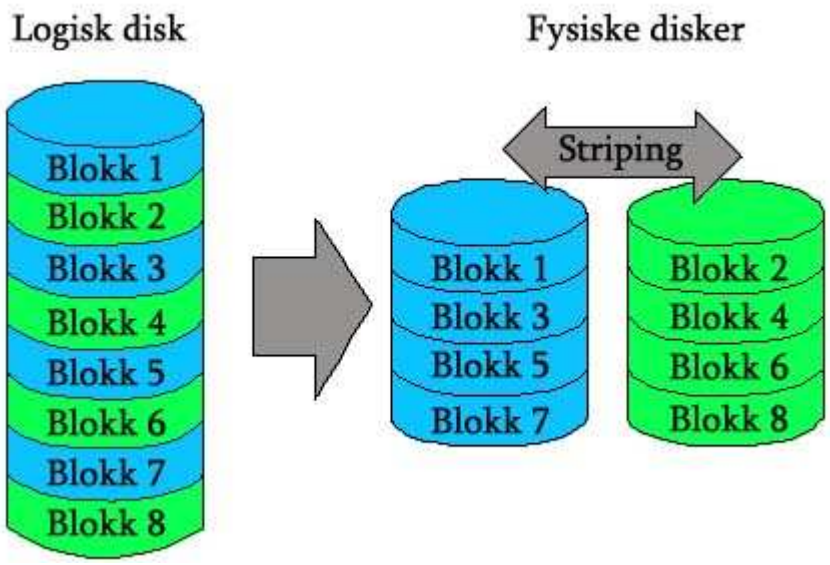
RAID står for Redundant Arrays of Inexpensive Disks [5, 6], og er en samling av harddisker for å gjøre lesing og skriving til harddisker mer effektivt. I 1978 ble Norman Ken Ouchi fra IBM tildelt et patent med tittel «System for recovering data stored in failed memory unit» som beskrev det som senere er blitt referert som RAID 5. Patentet nevnte også speiling av disker (RAID1) og beskyttelse med dedikert paritet (RAID4). Betegnelsen RAID ble først definert av David A. Patterson, Garth A. Gibson og Randy Katz ved universitetet i California. Hensikten med RAID er å gjøre lagring av data raskere og øke stabiliteten til diskene. RAID er en virtuell harddisk som består av to eller flere harddisker. Dette gjør det raskere å skrive data til harddisker, øker lagringskapasiteten og spare tid når en utfører vedlikehold av systemet. RAID er delt opp i nivåer hvor hvert nivå har sine fordeler og ulemper.

Nivå	Innehold	Minimum HD	Lagrings på HD	Feiltoleranse
RAID 0	Strip/Span	2	N	Ja
RAID 1	Speiling	2	N/2	Ja
RAID 0+1	Speiling + Stripe	4	N/2	Ja
RAID 3	Parallell med paritet	3	N-1	Ja
RAID 4	Parallell med paritet	3	N-1	Ja
RAID 5	Stripet med roteringsparitet	3	N-1	Ja

Tabell 2.1: Nivåer over RAID

2.1.2.1 RAID 0

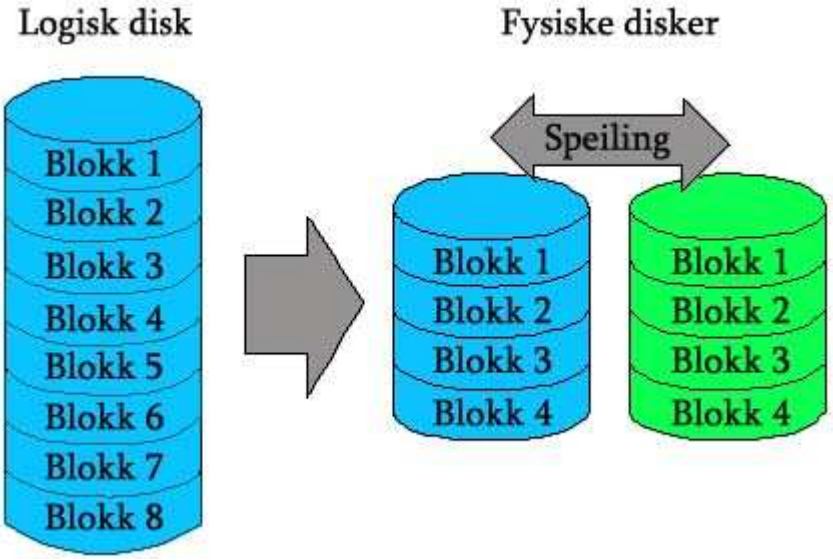
RAID 0 består av to eller flere harddisker som virker samtidig. Dette er en veldig effektiv måte for skriving av data til harddisk, og med flere harddisker installert blir skriving av data mer effektivt. RAID 0 fordeler dataene på de forskjellige diskene, og gjør flere små disker til en stor disk. Denne type RAID har ikke paritetssjekk, og kan derfor ikke gjenopprette data om en disk krasjer eller gjenopprette data ved å bytte til ny harddisk. Måten data skrives til diskene på er vist i figur 2.2.



Figur 2.2: RAID 0

2.1.2.2 RAID 1

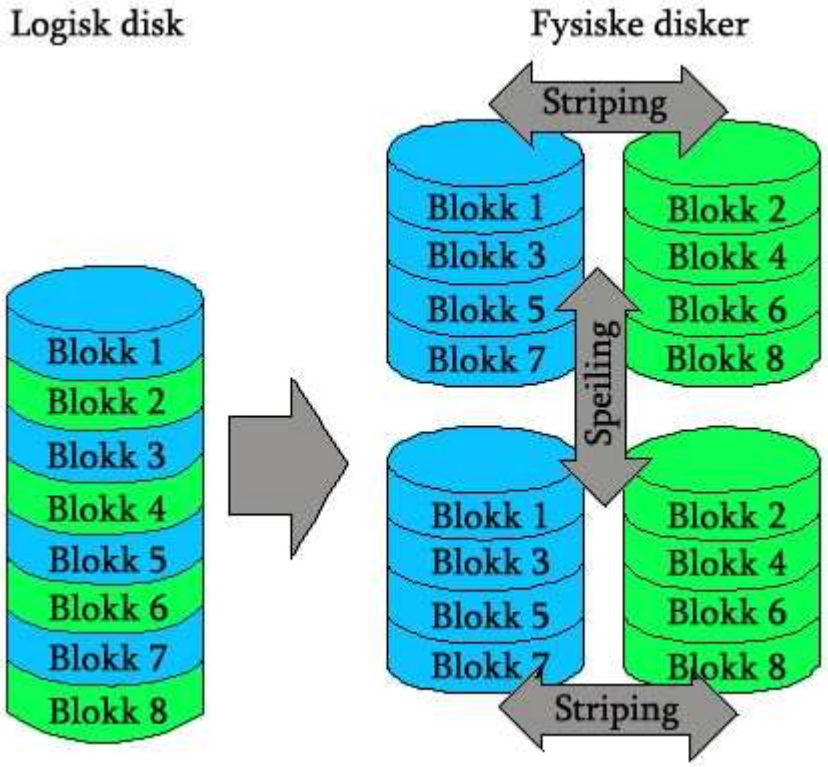
RAID 1 benytter to eller flere disker for lagring av data. Det må være et partall av disker, da to harddisker alltid har den samme data. Den ene disken brukes på vanlig måte, mens den andre disken brukes for backup. Ulempen med RAID 1 er at halvparten av diskplassen må bli brukt til backup, og er derfor en dyr måte å implementere RAID på.



Figur 2.3: RAID 1

2.1.2.3 RAID 0+1

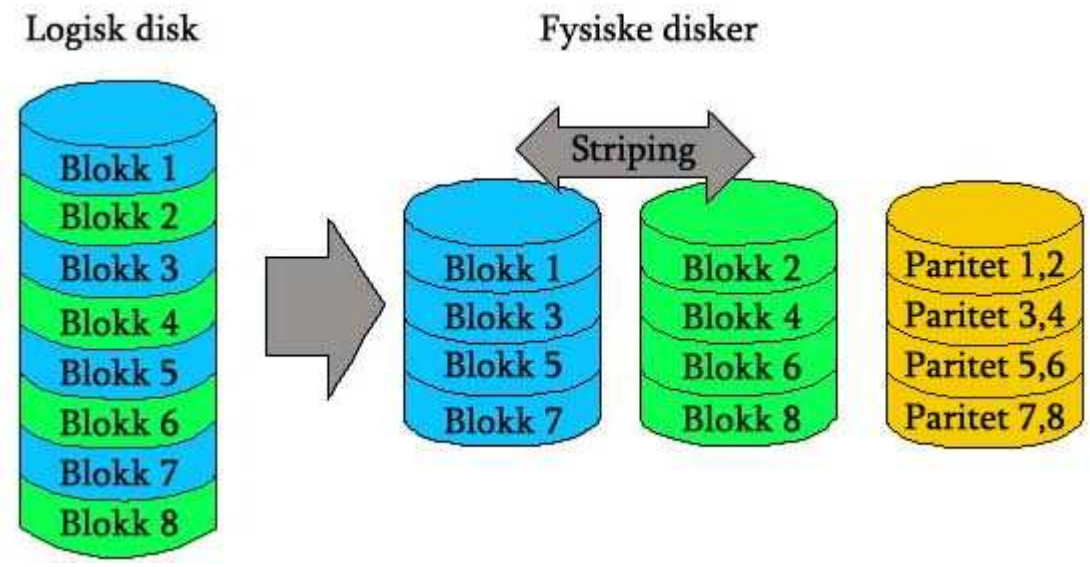
RAID 0+1 er en kombinasjon av RAID 0 og RAID 1. Her benyttes minimum 4 disker og antall disker må være partall. RAID 0 benyttes for å skrive til diskene raskere, samtidig som RAID 1 blir brukt for å ha en kopi av de andre diskene. Kun halvparten av den totale diskplassen går til lagring av data, noe som er ineffektivt. En fordel med RAID 0+1 er at den har feiltoleranse på diskene samtidig som skriving til diskene er effektivt.



Figur 2.4: RAID 0+1

2.1.2.4 RAID 3

RAID 3 består av tre eller flere harddisker og tolererer feil på harddisken. Den ene harddisken lagrer alle paritetssjekkene, som blir kalt paritetsdisk. Benytter man 3 harddisker i RAID 3 vil lagringskapasiteten bli på N-1 disker, som vil si to disker i dette tilfelle. Paritetsdata må skrives til paritetsdisken, så I/O effektiviteten vil være lavere enn i RAID 0. Lagringskapasiteten vil være større enn for RAID 1 og RAID 0+1.



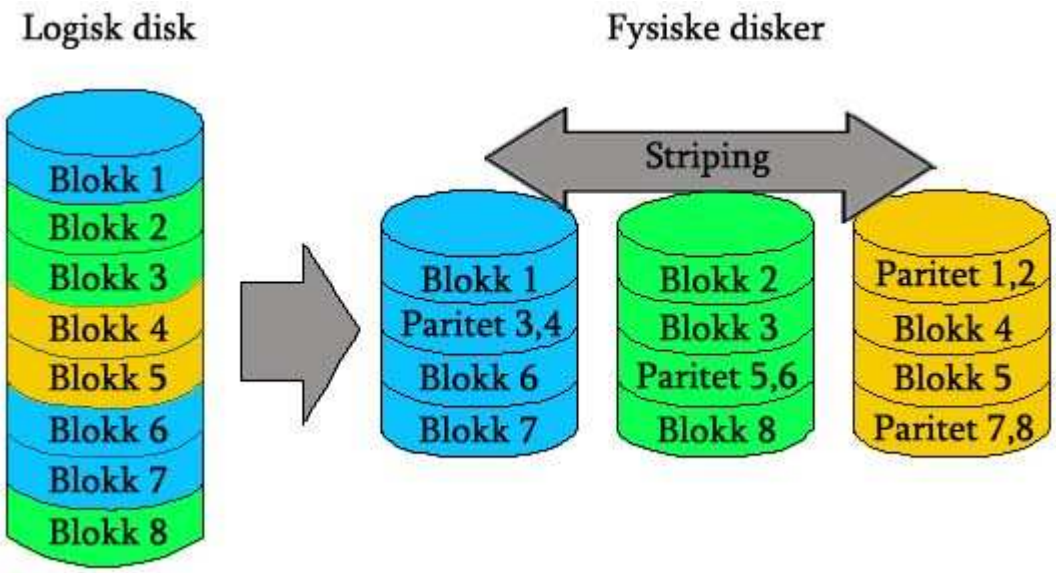
Figur 2.5: RAID 3

2.1.2.5 RAID 4

RAID 4 har samme virkemåte som RAID 3, men lagringsenheten i RAID 4 er blokker. Dette egner seg godt for systemer for PC tegninger, film prosessering og digitale bibliotek. Når data lagres på harddisken må paritetsdisken oppdateres, og gjør at dette systemet ikke egner seg hvor mange personer bruker databaser på samme tidspunkt.

2.1.2.6 RAID 5

Funksjonaliteten til RAID 5 er nokså lik RAID 3 og RAID 4. RAID 5 benytter seg av minimum 3 disker. Etter at paritetssjekk er kalkulert vil data bli skrevet til alle harddiskene. Dersom en RAID disk går i stykker vil RAID - kontrolleren stoppe alle I/O prosesser så lenge harddisken ikke er byttet ut med en ny og data er gjenopprettet. Reservedisk blir tatt i bruk slik at RAID - kontrolleren automatisk kan kjøre reservedisken for å gjenopprette data ved krasjer av RAID - harddisker. Det tar kun få minutter å gjenopprette data som er hovedfordelen med RAID 5.



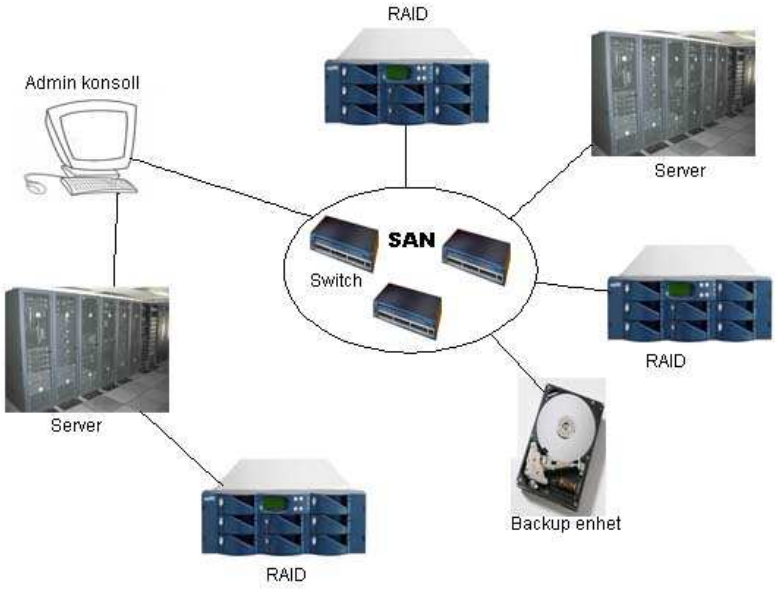
Figur 2.6: RAID 5

2.1.3 CAS

CAS står for Content addressable storage [7, 8], og er utviklet for å lagre data som sjeldent eller aldri forandrer seg. Dette kan typisk være digitale bilder, pasient røntgenbilder, rettsrapport osv. CAS systemer kan tilby lave kostnader for lagring av data ved å bruke for eksempel billige SATA eller SAS harddisker. CAS sikrer dataintegritet, fordi i slike systemer eksisterer data som kommenterte objekter som ikke kan bli duplikat eller endret. Data blir lagret på en fast plass på harddisken. Siden hvert objekt er unikt, vil det være umulig å lagre flere kopier av samme fil på harddisken. Dette gjør at man reduserer den totale lagringskapasiteten. Ved å koble til omfattende metadata til et objekt, kan data bli indeksert og søkt på uten å vite spesifikk filnavn, dato eller filbetegnelse. For hver fil som blir lagret, blir det generert en hashverdi til denne filen. Når en ny fil blir lagt til harddisken blir en hashverdi generert. Genererer den en hashverdi som allerede finnes, betyr det at samme data er blitt lagret på disken og kun metadata og peker til den eksisterende delen vil bli lagret. Er hashverdien ny, vil den delen av data og metadata bli lagt til harddisken.
En fordel med et slikt system er at man kan benytte seg av billigere SATA eller SAS harddisker istedet for fiberkanal disker. CAS gir også reduksjon av den totale mengden data som lagres, og gjør at man trenger færre harddisker.
CAS er utviklet for sekundærlagring, og er ikke anvendelig for høyytelses lagringsoppgaver. Ved generering av hashverdier kan kollisjon oppstå. Ulempen med dette er om et nytt objekt skal bli lagret på harddisken og CAS systemet generer en hashverdi som allerede eksisterer, vil ikke det nye objektet bli lagret og dette vil gå tapt ettersom CAS systemet tror at denne filen eksisterer fra før av. Et problem i dag er at hver leverandør av CAS systemer bruker egne APIer (application program interface) for å opprette og lagre metadata, slik at applikasjoner blir kodet til å tilpasse hver API. Dette gjør at man ikke kan flytte fra ett CAS system til et annet uavhengig av produkt og leverandør. SNIA (Storage Networking Industry Association) er en organisasjon som jobber med å standardisere grensesnittet mellom de forskjellige leverandørene, noe som gjør at man kan bytte system mellom de forskjellige produktene og leverandørene.

2.1.4 SAN

SAN står forkortet for *Storage Area Network* [9], er et separat høyhastighet lagringsnettverk designet for å koble sammen flere datalagringsenheter som disk- og tapestasjoner med tilknyttede servere. Et SAN er på lik linje med et lokalnett (LAN), men lagringssystemene er samlet i egne kabinetter som servere kobler seg til via fiberkanaler. Et SAN kan bestå av RAID lagringssystemer, servere, backupenheter og administrasjonskonsoll. SAN består av fiberkabler med switcher og huber som kobler alle enhetene sammen [10]. Se på figur 2.7.



Figur 2.7: SAN nettverk

2.1.4.1 Fordeler

Fordeler med SAN er at det gir økt fleksibilitet, høy ytelse, enklere administrasjon, bedre utnyttelse av diskkapasitet og mulighet for bedre funksjonalitet i lagringsløsningen. SAN gir en svært rask og pålitelig måte å lagre, dele og nyttegjøre data på. Dette er viktig på grunn av den økende mengden data hos bedrifter og det økende behovet for tilgang til data.
Med SAN er det mulig å sentralisere og dele datalagring mellom ulike servere i en bedrift. Dette innebærer at all datalagring er samlet sentralt på et sted og ansatte i bedriften har tilgang til disse data. Utnyttelsen av kapasiteten blir langt bedre enn DAS (*Direct Attached Storage*) ettersom denne lagringseneheten er direkte tilknyttet serveren. I tillegg er kostnaden med administrasjon av slike DAS-løsninger dyrere enn kostnaden med drift av SAN løsninger [11].
Med SAN kan man sette opp speiling i sanntid for å beskytte data, som er nyttig for å unngå kritiske tap av data; som i verste fall kan føre til at en bedrift går konkurs. Fiberkanal støtter

distanser opp til 10 km. Et datasenter kan speiles over til et annet område for maksimal tilgjengelighet og redundans. I tillegg er det vanlig at alle komponenter i SAN er redundante og dermed mer feiltoleranse enn et vanlig disksystem.
Virtualisering av datalagring i SAN gjør det enklere for SAN-administratorer å håndtere data og gjøre endringer med kun få enkle klikk. Virtualiseringsprogramvaren viser logiske volum til servere ettersom kapasitet trengs. Virtualisering av datalagring gjør det mulig å utnytte kapasitet minst 30 % mer enn tradisjonell lagring [11]. En annen fordel er at sikkerhetskopiering kan kjøres direkte over fiberkanal med høyhastighet.
Nyere SAN tillater også muligheten for dupliseringsfunksjoner som kloning og snapshotting av data som gir økt fleksibilitet for lagring av data, gjenoppretting av data ved katastrofe, eller duplisering av data. Snapshot vil si å ta en virtuell kopi av data i en gitt tid. Data blir ikke fysisk skrevet til disk, men lagret i minnet. Ved kloning vil det opprette en eksakt kopi av data i bakgrunnen mens ved snapshotting vil det kun lagre den siste blokken av den originale data som har blitt forandret ved forrige snapshot.

2.1.4.2 Ulemper

Det er ikke alle som kan ta i bruk SAN. Det kreves at man har god forståelse av teknologien for å kunne implementere en løsning som er tilpasset behovet for sin bedrift. Implementering av SAN er noe som krever tung kompetanse og detaljert planlegging. Det er begrensning på muligheten for å benytte lagringsenhetene sammen med ulike operativsystemer, databaser og teknologier. SAN kan være en utfordring å implementere innenfor en bedrifts eksisterende informasjonssystemarkitekttur. Dette krever vanligvis spesialisert personale for administrasjon av SAN. Det kreves kraftig og dyr programvare for å kunne administrere lagringsenhetene, ellers vil administrasjon og drift være komplisert.
En annen vurdering for en bedrift som ønsker å implementere SAN er kostnader. Dette kan koste flere millioner kroner å implementere, og er dermed ikke praktisk for mange små og mellomstore bedrifter [11]. Disse bør se på andre alternativer som NAS (Network Attached Storage), som er et kabinett koblet opp i nettverket for å gi utvidet lagringskapasitet. Kostnader kan være en mindre bekymring fremfor skalerbarhet, pålitelighet, ytelse og kapasitet man oppnår med SAN.

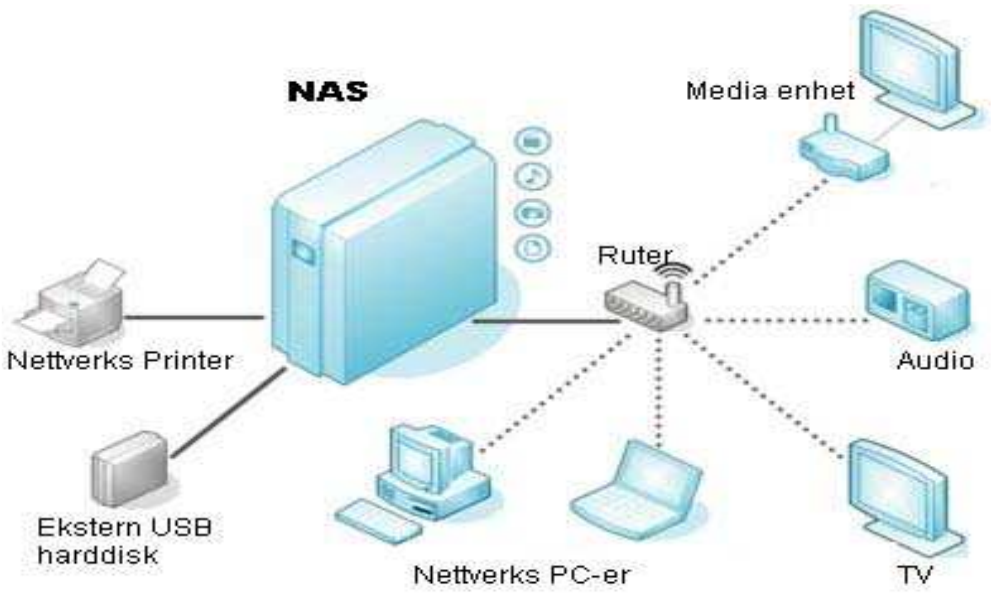
2.1.4.3 SAN komponenter

SAN består av større mengder komponenter som RAID systemer, switcher, huber, servere, lagringsenheter (backupenheter), fiberadaptere, fiberkabling og programvare.
RAID systemet er hovedkomponenten i et SAN. Den er satt opp med et titalls eller hundretalls harddisker med flere RAID5-sett. Et RAID - system består av et kabinett med mange harddisker og noen kontrollere hvor alle komponenter er redundante. Alle disker styres av en RAID kontroller, duplisert for redundans som kan dele ut all diskkapasitet som en stor disk eller som flere større og mindre virtuelle disker ut mot servere.
RAID - systemet er koblet til en fiberkanal svitsj via fiberkanal kabler. Servere får installert fiberkanal adaptere og nødvendig programvare. Både servere og eventuelle backupenheter kobles

direkte til fiberkanal svitsj.
I dag finnes det ingen fullverdig løsning for å kunne sikkerhetskopiere direkte av RAID systemet via fiberkanal. Man må ta veien om serverne for å kunne foreta sikkerhetskopiering da backupenhetene ikke er intelligente nok til å skjønne hvilke servere de virtuelle diskene på RAID systemet tilhører.

2.1.5 NAS

NAS står forkortet for Network Attached Storage [12], og er en betegnelse på maskinvare for lagring av data på et lokalt nettverk. I tillegg til å lagre data på en harddisk i en datamaskin, kan det foretas en sikkerhetskopiering av data på en sentralisert harddisk som er tilgjengelig for alle i nettverket. På denne måten kan flere brukere ha tilgang til de samme filene samtidig og filene trenger ikke å være lagret på enkelte maskiner. Lagring av data på tradisjonelle servere er for små for større bedrifter, og SAN er ofte for dyrt og komplisert [13]. NAS består av en eller flere harddisker, og kan tilby stor lagringskapasitet til en rimelig pris. Dette er en bra løsning for små og mellomstore bedrifter som ikke har råd til å implementere SAN. Privatpersoner kan også ta i bruk NAS. I 2004 begynte det å bli vanlig for privatpersoner å ta i bruk NAS på sine hjemmenettverk [14]. NAS var tidligere dyrt og unødvendig for privatpersoner, men prisen har sunket betraktelig de siste årene og i dag tilbys det enklere og billigere løsninger for privatpersoner som ønsker å utvide sin lagringskapasitet på sine nettverk.



Figur 2.8: NAS med flere enheter tilkoblet

2.1.5.1 Fordeler

NAS er en ”plug and play” enhet og er lett å ta i bruk. NAS krever lite installasjon og konfigurasjon for å koble til det lokale nettverket [15]. NAS erstatter flere servere med en enkel lagringsenhet på et sted, samtidig som NAS gir mulighet til å tilknytte flere operativsystemer og

dele data mellom flere brukere og servere. NAS har støtte for både NFS protokollen til UNIX og CIFS til Windows, slik at det er mulig med datadeling mellom plattformer.
Den største fordelen med NAS er tilgjengeligheten til data som ikke er avhengig av en server eller en datamaskin; det vil si selv om serveren er nede eller datamaskinen er slått av har brukerne fortsatt tilgang til data siden NAS kan kobles direkte til en ruter. Overføring av data til NAS går raskt da NAS ikke er avhengig er server, men overføringen er avhengig av nettverkshastighet og trafikk.

2.1.5.2 Ulemper

Er det mange brukere koblet opp mot NAS - enheten samtidig, vil det føre til at det lokale nettverket vil gå tregt; noe som vil føre til ineffektiv overføring av data. NAS kan ikke tilby garanti på lagring av data siden dette er en nettverksdelt lagring. Siden NAS kan brukes til deling av data, må systemadministrator innføre kvotering av data slik at noen brukere ikke kan legge beslag på mer data enn andre.

2.1.5.3 Sikkerhet

For å sikre at data lagret på NAS ikke havner hos uautoriserte brukere kan man ta i bruk krypteringsmekanismer. Når data lagret på NAS er kryptert kan man sikre seg mot at tyver eller uautoriserte brukere ikke får lest innholdet selv om de har klart å stjele dataene. Det kan lønne seg å foreta sikkerhetskopiering av dataene lagret på NAS enheten på et annet sted i tillegg [16].

2.1.5.4 Fremtiden til NAS

Eksperter er enige om at NAS har en lysende fremtid innenfor alle former for virksomheter i fremtiden [16]. I dag brukes NAS både i små og mellomstore bedrifter og i private hjem.
NAS har både fordeler og ulemper, men man kan dra nytte ut av fordelene med NAS, samt fordelene med SAN. I de siste årene har det kommet nye løsninger fra både IBM og HP som er integrering av SAN og NAS. Ved å kombinere en løsning med SAN og NAS kan de serverne som krever høy ytelse kobles direkte til SAN, mens de servene som kun krever fildeling kobles til via NAS.

Figur 2.9: NAS harddisk fra Western Digital

2.1.6 DAS

DAS står forkortet for *Direct Attached Storage* [17, 18], og er tilknyttet direkte en datamaskin eller server. DAS kan kobles opp mot flere enn en server, men lagring av data på hver server kan ikke deles med andre servere. DAS er en lokal lagringsenhet og ikke nettverksbasert slik som NAS og SAN.

Hovedprotokollene som brukes i DAS er SCSI, SAS og fiberkanal. DAS enheter brukes til å tilføye lagring på en eksisterende server. Hver enhet har en innebygd RAID kontroller som støtter flere RAID nivåer som er lette å konfigurere via programvare.
RAID tilbyr dataredundans; noe som er meget kritisk for servere. Hvis en disk krasjer, kan man erstatte den med en ny disk, og DAS - enheten vil gjenopprette data helt automatisk [19].

2.1.6.1 Fordeler og ulemper

Fordeler med DAS er at det er lettere å administrere siden det kun er koblet til en server. DAS brukes ikke bare til lagring og gjenoppretting av data, men kan også brukes til å håndtere applikasjoner.
Ulemper er at DAS ikke yter like høyt nivå som nettverksbaserte lagringsenheter. Man kan ikke dele lagrede data med andre servere slik som NAS og SAN [20].

2.1.7 Internett: Online backup

Online backup tilbyr en tjeneste som er mer effektiv enn andre backup medier som tape, optisk disk, harddisk og USB penn. Disse mediene er nyttige å bruke og er lett å ta med overalt, men de kan være problematiske å holde oversikt over [21].
Et annet problem er at folk flest vanligvis oppbevarer disse backupmediene på samme sted som man har PC-en sin; for eksempel oppbevare eksterne harddisker og CD-plater i samme rom i en bygning. Ved et uhell som brann eller ras vil man miste data som er lagret både på PC-en og harddiskene. En annen uheldig situasjon som kan oppstå er tyveri av PC-er og harddisker. Et annet alternativ er å oppbevare backupmediene et annet trygt sted enn på samme sted som PC-en befinner seg i. Problemet her er at det vil være tungvint å reise mye frem og tilbake for å ta backup og gjenopprette data [22].
Med online backup slipper man å tenke på disse problemene. Denne tjenesten tilbyr mange fordeler. Ved uhell eller tyveri kan man gjenopprette dataene, men da må man ha tilgang til dataene. For å få tilgang må man ha riktig brukernavn og passord.
En annen fordel er at man slipper å lete etter ønskede data blant flere CD-er og harddisker. Når man tar backup online, så ligger all data på et sted. I tillegg så kan man lagre kopier og gjenopprette dataene overalt så lenge det er internett tilgang; da slipper man å ta med backupmediene overalt.
En fordel til med online backup er å kunne dele dataene med andre brukere ved å gi dem tillatelse til å hente disse og/eller forandre på innholdet. Eieren av disse dataene bestemmer hva andre

brukere kan få tillatelse til [21].
Begrensninger i online backup er båndbredde og internett tilgang. Mange bedrifter har i dag internett tilgang med stor båndbredde, og det vil ikke være problemer å benytte online backupsystemer. Privatpersoner har ikke like stor båndbredde som bedrifter. I tillegg kan det være kostbart å lagre kopier av data online. Den største ulempen med online backup er utgiftene hver måned. I tabellen nedenfor er det listet opp priser som er hentet fra Nordic Backup [23] den 16.1.2007:

Priseksempler	Pris pr måned
1 GB Online Backup	19,00 kr
3 GB Online Backup	38,00 kr
5 GB Online Backup	56,00 kr
10 GB Online Backup	100,00 kr
15 GB Online Backup	150,00 kr
20 GB Online Backup	200,00 kr
25 GB Online Backup	250,00 kr
50 GB Online Backup	400,00 kr
100 GB Online Backup	650,00 kr
250 GB Online Backup	999,00 kr
500 GB Online Backup	1749,00 kr
1000 GB Online Backup	2250,00 kr
3000 GB Online Backup	4250,00 kr
5000 GB Online Backup	6250,00 kr

Tabell 2.2: Prisoversikt for Online backup

Prisene varierer fra forskjellige land og firmaer. I enkelte land er det billigere å bruke online backup enn i Norge.

2.2 Sikkerhetstjenester

Overføring av data over internett kan være risikabelt. Det kan føre til at andre brukere kan få tilgang til data og misbruke dem. Det er derfor nødvendig å ta i bruk sikkerhetstjenester som autentisering og kryptering.

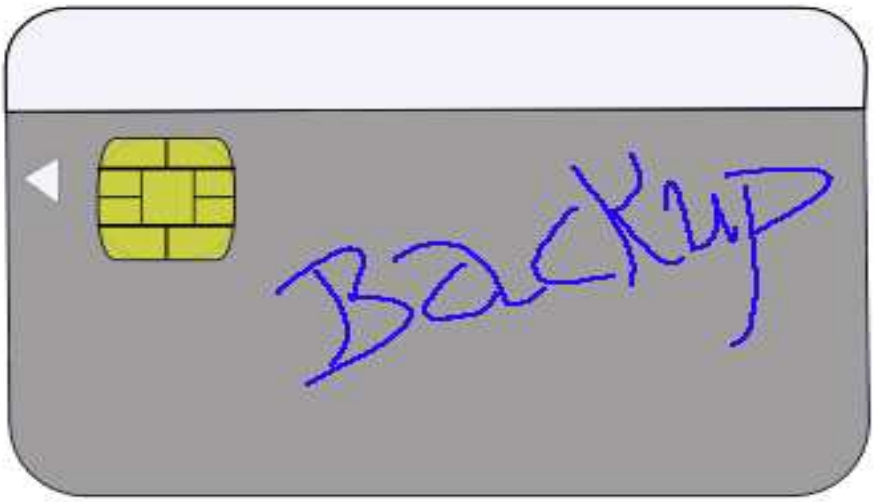
2.2.1 Autentisering

I informasjon og systemsikkerhet betyr autentisering at en bruker må identifisere seg selv for å verifisere at vedkommende virkelig er den man utgir seg for å være [24, 41]. Et datasystem som er ment for å bli brukt av autoriserte brukere må kunne detektere og ekskludere uautoriserte brukere. For å få tilgang til dette datasystemet må brukeren gå gjennom en autentiseringsfase. Denne autentiseringsfasen består for eksempel av brukernavn og passord. For eksempel når en bruker vil lese e-post fra sin konto, må vedkommende logge seg inn med å skrive inn riktig brukernavn og passord. Det finnes tre måter å autentisere seg på [41]:

- Noe man vet
- Noe man eier
- Noe man er

«Noe man vet» kan være enten passord eller en PIN kode. Dette er hemmelig informasjon som kun eier vet om. En PIN kode består normalt av en kode med 4 siffer som brukes f. eks ved betaling i bankterminaler, eller på mobiltelefonen for å autentisere seg for SIM-kortet. De fleste smartkort er konfigurert med en PIN kode slik at ikke andre enn eieren av kortet får tilgang. Passord er en annen måte å autentisere seg på. Et passord brukes ofte ved innlogging på PC, webshop, lesing av e-post eller ved å se på web-TV. Disse passordene genereres ved at man velger en kombinasjon av siffer, bokstaver eller tegn. Fordelen med en slik form for autentisering er at det er en billig måte å autentisere seg på, samt at eieren av passordet ofte selv kan velge hva dette skal være. En ulempe er at de fleste velger passord som er i tilknytning til for eksempel bursdager, navn på familiemedlemmer eller husdyr. Dette gjør det enkelt for andre å knekke passordet, og ofte er det satt kriterier av systemet for hvordan passord skal genereres for å redusere slikt. Dette kan være at et passord skal bestå av minst 6 tegn, og ett av disse tegnene må være enten siffer eller spesialtegn (f.eks. _?/+). Dette gjør det vanskeligere for andre å finne ut av passordet, samtidig som det er lett å huske. Passordfraser er en annen effektiv måte å generere passord på. Et eksempel på dette kan være Iy40WW2s, som betyr «In year 1940 World War 2 started». Her blander man både små og store bokstaver, samt tall. Et slikt passord er vanskelig å gjette, samtidig som det er lett for eieren av passordet og huske.

”Noe man eier” er for eksempel et adgangskort for å komme inn på et spesielt rom eller bygning. Det kan også være et bankkort eller et SIM kort. I noen tilfeller er det kun nødvendig å benytte kortet for å få tilgang. I andre tilfeller krever systemet at man må taste inn en PIN kode i tillegg. Dette kalles multifaktor autentisering, hvor det er en kombinasjon av to faktorer, som også kalles to-faktor-autentisering. Andre eksempler på multifaktor autentisering er bruk av bankkort på minibank, SIM kort på mobiltelefoner, osv. Dette gir en større grad av sikkerhet da man må autentisere seg med «noe man vet» og «noe man eier».



Figur 2.10: Et smartkort

”Noe man er” er biometrisk autentisering. Det vil si at en person autentiserer seg for en maskin ved å bruke sin egen kropp. Dette kan for eksempel være skanning av fingeravtrykk, håndflaten, ansikt, øyet (iris skanning), ganggjenkjenning, og stemmegjenkjenning. Dette er en annerledes form for autentisering der flere deler av kroppen er vanskelig å kopiere. Ulempen med et slikt system er at det er dyrt å implementere, samt at registrering av brukere kan i noen tilfeller ta lang tid. For biometrisk autentisering er det to typer feilrater, FAR(fail accept rate) og FRR(fail reject rate). FAR er når en bruker blir akseptert ved en feil under autentiseringsfasen. Dette oppstår når en person utgir seg for å være en man ikke er, og blir godkjent. FRR vil si at en person blir avvist fra å være den man egentlig er. Er dette tilfelle vil personen som rettmessig prøver å få tilgang ikke får aksess, noe som ikke er heldig. Disse to ratene kan veies opp mot hverandre. Har man en lav FAR, vil FRR være høy. Dette betyr at sannsynligheten for at uautoriserte brukere ikke får tilgang, men dette resulterer i at autoriserte brukere ikke får tilgang til steder de skal ha tilgang. Om man har en noe høyere FAR vil det være større sannsynlighet at uautoriserte brukere får tilgang, men da vil FRR være lavere, noe som resulterer i at det er mindre sannsynlighet for at autoriserte brukere ikke får tilgang til de stedene de skal.

2.2.2 Kryptering

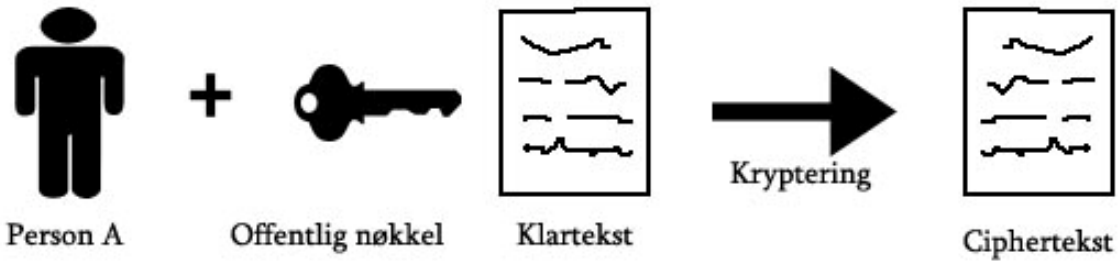
Kryptering vil si å gjøre informasjon uleselig [25, 41]. Kryptering kan bli brukt til å sikre hemmelighold av informasjon og kan bli brukt til å signere en melding. Cipher er en uleselig tekst som er opprettet etter at klartekst har blitt kryptert. For å utføre kryptering og dekryptering av informasjon må man ha en nøkkel for å lage en ciphertekst (kryptere) og for å gjenopprette til klartekst (dekryptere).

Det finnes i dag to typer algoritmer for kryptering og dekryptering av informasjon. Dette er symmetrisk og asymmetrisk nøkkelalgoritme. Symmetrisk nøkkelalgoritme vil si at man har den samme nøkkelen for kryptering og dekryptering (for eksempel DES og AES). Med symmetrisk nøkkel har man en hemmelig delt nøkkel som benyttes. Senderen bruker da denne nøkkelen for å kryptere data og mottaker bruker den samme nøkkelen for å dekryptere data.

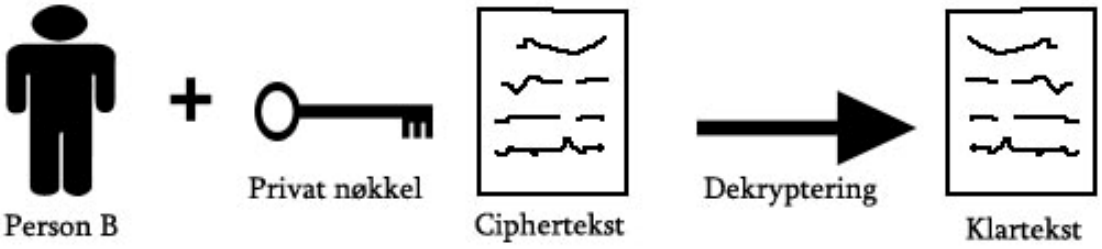
Asymmetrisk nøkkelalgoritme (for eksempel RSA) vil si at man har en nøkkel for å kryptere data og en annen nøkkel for å dekryptere data. Det benyttes da en offentlig nøkkel som brukes til å kryptere data. Denne operasjonen kan utføres av alle som er i besittelse av denne nøkkelen. Den andre nøkkelen som blir kalt privat nøkkel brukes for å dekryptere data. Denne nøkkelen er det kun mottaker som har, og det er derfor bare denne personen som kan utføre denne operasjonen.

En asymmetrisk nøkkel kan også benyttes til å signere en melding. Da benytter man den private nøkkelen til å signere en melding og man verifiserer signaturen med den offentlige nøkkelen. Dette benyttes ofte for å kunne bevise at man er den man utgir seg for å være som for eksempel i nettbankløsninger.

Krypteringsprosedyren avhenger av nøkkelen som benyttes for å generere ciphertekst. En nøkkellengde er en måling av mulige nøkler som kan bli brukt i en cipher. Denne nøkkellengden er spesifisert i antall bits. Mange krypteringsalgoritmer er basert på kjente algoritmer, og det er kun vanskeligheten i å fastsette nøkkelen som bestemmer sikkerheten i systemet; som også refereres til Kerchoffs prinsipp. En nøkkel betegnes som sikker om det ikke finnes noen snarvei-angrep (det vil si å kunne utnytte svakheter i algoritmen til å kunne gjette seg fram til hva nøkkelen er) eller at man ikke kan utføre noen brute-force angrep som vil si at det vil ta for lang tid å teste ut alle mulige nøkler for å finne den.



Person A bruker den offentlige nøkkelen til å kryptere klartekst, og deretter sender ciphertekst til Person B.



Person B mottar ciphertekst fra Person A, og bruker sin egen privat nøkkel til å dekryptere cipherteksten og får fram klarteksten.

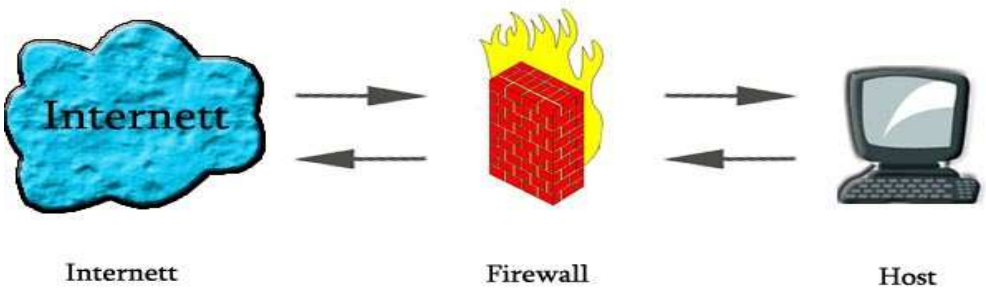
Figur 2.11: Eksempel på asymmetrisk kryptering

2.3 IP

IP (Internet Protocol) er en protokoll som brukes for kommunikasjon i et pakkesvitsjet nettverk. IP forsyner en unik adresse som tildeles den enkelte host som er koblet opp mot internett [26]. En host kan for eksempel være en PC eller en ruter. Denne adressen er i utgangspunktet unikt tildelt og er gyldig verden over. Adressen benyttes for å fortelle hvor i nettverket en host befinner seg, og en IP adresse består av en 32 bit adresse. Et eksempel på en IP adresse kan være 128.39.147.62 hvor hvert tall er representert med en 8 bit binær representasjon. Dette gir et adresseområde på 4.294.967.296 adresser, selv om mange ikke kan brukes; mange av disse adressene er reservert til spesielle formål som private nettverk eller multicast adresser. En IP adresse er delt opp i en nett – ID og en host – ID. Nett – ID bestemmer hvilket nettverk hosten befinner seg i, mens host – ID bestemmer ID til host i det nettverket den befinner seg i. Dette er gjort for å skille de forskjellige nettverkene fra hverandre, samt å skille de forskjellige hostene i ett nettverk fra hverandre. I adressen 128.39.147.62 er de første 16 bitene brukt til å identifisere nettveket og de siste 16 bitene identifiserer hosten. Tallet 128.39 blir da nett id og 147.62 blir da host id. Det er da plass til 65 536 maskiner i ett nettverk.

2.4 Brannmur

En brannmur har til hensikt å kontrollere data som blir sendt mellom to nettverk med forskjellig soner av tillitt [27]. Typisk eksempel er internett hvor man har en sone med ingen tillitt og det interne nettverket hvor man har en sone med tillitt som vist i figur 2.12.



Figur 2.12: Hvordan brannmuren kontrollerer trafikken inn og ut av internett.

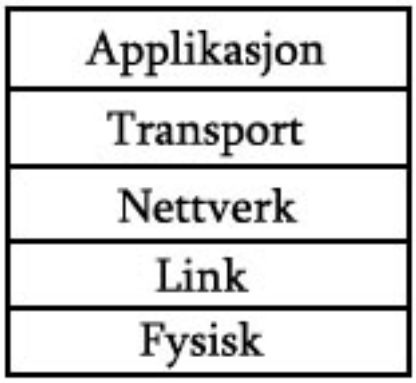
Det finnes tre typer brannmurer. Hver brannmur filtrerer pakker ved å analysere pakker opp til et visst nivå i nettverksprotokoll stakken. Disse tre typer brannmurer er:

- Pakkefilter brannmur
- Stateful pakkefilter
- Applikasjon fullmektig

En pakkefilter brannmur analyserer pakker opp til nettverkslaget som indikert i figur 2.13. Den kan filtrere pakker ut i fra avsenderadresse, destinasjonsadresse, avsenderport og destinasjonsport. Den kan på dette nivået ha forskjellig filtreringsregler på inngående og utgående trafikk. En ulempe med denne type brannmur er at den åpner for TCP angrep mot en host. Sender man en pakke med ACK bit satt, vil en brannmur tro at pakken er en del av en etablert forbindelse og slippe pakken igjennom. Når den kommer fram til hosten vil denne oppdage at pakken ikke er en del av en etablert forbindelse, og sende en RST (Reset the connection) pakke hvor mottaker av pakken forteller avsender at han skal terminere forbindelsen. På denne måten kan en angriper finne ut av hvilke porter som er åpne i en brannmur, og man kan benytte denne porten til å angripe et system.

En tilstandsbevisst pakkefilter har oversikt over alle TCP forbindelser og opererer på transportlaget i protokollstakken. Den husker også UDP forbindelser. På dette nivået tar det noe lengre tid å analysere pakker enn med pakkefilter brannmur. Denne type brannmur hindrer TCP angrep som nevnt over. Ulempe er at den ikke kan analysere applikasjonsdata.

Applikasjon proxy brannmur prosesserer data opp til applikasjonslaget i protokollstakken. Brannmuren er i stand til å verifisere at pakkene er legitime og at data inne i pakken er trygge. Ulempen med denne type brannmur er at det tar lengre tid å prosessere data. En fordel med en applikasjon proxy er at den hindrer firewalk angrep. Hensikten med denne type angrep er den samme som for TCP angrep, bortsett fra at her blir TTL(time to live) feltet i IP pakken satt. Vet en angriper IP adressen til brannmuren, antall hopp det er frem til brannmuren og en adresse til en host på innsiden av brannmuren kan man sende en pakke til denne hosten. Dette gjøres ved at TTL feltet settes til et hopp mer enn til brannmuren. Det kan deretter settes et portnummer i IP pakken. Om pakken ikke slipper igjennom brannmuren vil man ikke motta noe svar fra brannmur. Om pakken slipper igjennom brannmuren vil angriperen motta en «tid utløpt» melding fra den første ruteren innenfor brannmuren som mottar pakken. På denne måten kan man finne ut av hvilke porter som er åpne i brannmuren, for deretter å utføre angrep mot et nettverk.



Figur 2.13: En forenklet versjon av OSI modellen.

En brannmur kan enten være hardware basert eller software basert. En hardware basert brannmur er en boks hvor man plasserer for eksempel mellom internett og det lokale nettverket for å overvåke all trafikk mellom de to nettverkene. En softwarebasert brannmur er en applikasjon som installeres på en PC for å overvåke en enkelt host, eller at man plasserer denne PC-en mellom internett og det lokale nettverket slik at all trafikk går igjennom denne PC-en.

2.5 Ruter

En ruter er som et knutepunkt mellom to eller flere nettverk, og har som oppgave å videresende pakker mellom dem. For at man skal kunne koble flere maskiner mot internett i private hjem, er det nødvendig å ha en ruter. Denne ruteren vil koble det globale nettverket sammen med det lokale nettverket. Når en host på lokalnettet vil sende en pakke til en host på internett, vil ruteren oversette adressen til den lokale hosten til en global adresse som ruteren har fått tildelt av sin ISP (Internet Service Provider). For at ruteren skal kunne videresende pakken, må den være koblet til andre rutere som videresender pakken til andre nettverk om dette er nødvendig.
De fleste hjem har i dag en ruter tilkoblet internett, slik at flere maskiner kan kommunisere over internett samtidig. Dette er en fordel da flere maskiner kan dele den samme internettilkoblingen. En annen fordel med en ruter er at tildeling av IP – adresser kan gjøres automatisk, og man slipper konfigurering av dette.
En ulempe med en ruter er når man ønsker å sette opp en tjeneste på en maskin bak en ruter. Man må selv konfigurere ruteren til å videresende forespørsler på gitte tjenester, noe som krever at brukeren har kjennskap til konfigurering av rutere.

3 Metode

3.1 Arbeidsmetode og organisering

Vår gruppe består av to personer. Huy har fordypning innen bildebehandling, og André har fordypning innen nettverksdrift og sikkerhet på Bachelor grad. Vi har begge fordypning i sikkerhet på master grad med valgfag innen mobil kommunikasjon. Vi har nokså lik fordypning, noe som gjør at vi kan bidra likt på alle deler av prosjektet. Det vil ikke bli noen hovedfordeling av oppgaver i dette prosjektet, men vi vil dele inn i deloppgaver som vi gjør hver for oss for å få arbeidet raskere utført. Når vi har gjort vår del av oppgave vil vi sammen gå igjennom hverandres del for å komme med kommentarer til forbedringer. Arbeid som er utført vil kontinuerlig bli levert til vår oppdragsgiver for tilbakemelding slik at vi kan få en objektiv tilbakemelding på arbeidet som er gjort.

3.2 Software utviklingsmetoder

3.2.1 Scrum

Scrum er gjentakende, inkrementell prosess for utvikling av hvilken som helst produkt og håndtering av arbeid, og vil derfor passe godt for utvikling av programvare [28]. Scrum starter med en visjon for produktet som skal utvikles. Visjonen utarbeides en produktbacklogg som er en oversikt over krav og ønsker til produktet. Alt arbeidet utføres i sprinter, og på hver sprint er en iterasjon på 30 kalenderdager. Hver sprint starter med et planleggingsmøte hvor en gruppe velger ut i fra produktbackloggen det man tror man kan utføre og levere i løpet av neste sprint. Når gruppen har valgt ut funksjonaliteten man vil levere i løpet av sprinten, starter utviklingen av produktet og sprinten er i gang. Under sprinten møtes teamet daglig for å sikre fremdrift. Hver sprint avsluttes med et møte hvor teamet presenterer hva som er blitt gjort i løpet av sprinten. Etter et demonstrasjonsmøte, holdes et evalueringsmøte for sprinten.
Scrum egner seg godt for store prosjekter som ofte trenger endringer i kravene. Scrum er en teambasert arbeidsmetode (maks 6-7 personer) hvor man trenger jevnlig møte for å diskutere, demonstrere og evaluere, for å maksimere produktiviteten. På denne måten kan man hjelpe hverandre til å finne eventuelle feil i produktet og forbedre produktet før levering til kunder.

3.2.2 Agile

Agile består av flere programvareutviklingsmetodologier som gjør det mulig å utvikle et system som er ubestemt og kravspesifikasjonen ofte trenger endringer [29]. Hensikten til de fleste metodene er å gjøre det lettere å utvikle et produkt samtidig som det skal ta kortere tid å levere produktet slik at det kan tilfredsstille oppdragsgivere eller kunder. Det viktigste med denne metodologien er at man jobber i grupper slik at man kan fordele oppgaver, samarbeide og hjelpe hverandre. På denne måten kan man gi hverandre tilbakemeldinger på hva som er bra og dårlig. Agile består av følgende faser: planlegging, analyse av kravspesifikasjon, design, koding, testing og dokumentasjon. Ved hjelp av denne metodologien medfører dette til tidlige og kontinuerlige

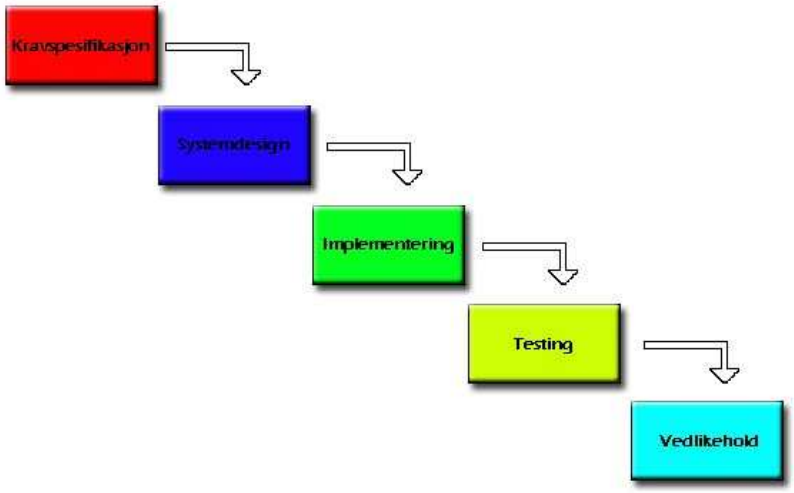
levering av et produkt, jevnlig kommunikasjon mellom systemutviklere og oppdragsgivere. Samarbeid og organisering i gruppen skaper motivasjon som vil føre til mer effektivisering i arbeidsprosessen.
I Agile har man muligheten til å gjøre endringer underveis og levere jevnlig et lite utdrag av produktet. På denne måten har man muligheten til å gjøre endringer og se på muligheten for å forbedre produktet.

3.2.3 Extreme programming (XP)

Denne metodologien er designet for å levere programvare som kunden trenger når det trengs [30]. Ved bruk av XP kan systemutviklere gjøre endringer i kunders kravspesifikasjon selv om det kan være mot slutten av prosjektperioden. Denne metodologien legger også vekt på gruppearbeid. Ledere, systemutviklere og kunder er alle som deltar i gruppen for å få levert et produkt. I et slikt prosjekt arbeides det effektiv gjennom kommunikasjon, enkelthet, tilbakemeldinger og mot. XP utviklere kommuniserer med kunder og andre medarbeidere. Man lager en enkel, strukturert og ryddig design underveis. Man får tilbakemeldinger ut i fra testing av programvaren allerede fra første dag. Man leverer produktet til kunden så tidlig som mulig, og gjør endringer i kildekoden hvis nødvendig. På denne måten har man mot til å gjøre endringer i kundenes kravspesifikasjon og teknologi ved senere behov. XP er som et puslespill med mange små brikker. Hver av brikkene har ingen betydning, men når de settes sammen vil de gi et godt oversiktsbilde over det komplette systemet.

3.2.4 Waterfall

Waterfall er en programvareutviklingsmodell som er nyttig å bruke til planlegging. Dette er en trinnvis metode å jobbe på hvor man jobber seg gjennom fasene fra kravspesifikasjon, design, implementering, integrasjon, testing, installasjon og vedlikehold [31]. Aller først blir det lagd et kravspesifikasjonsdokument hvor det blir beskrevet hvilke funksjonelle og ikke-funksjonelle krav som må være med for å få programvaren til fungere. Når kravspesifikasjonen er komplett, går man videre til designfasen. I systemdesignet skal man designe programvaren med brukervennligheten og funksjonaliteten som tilpasser kravene i kravspesifikasjonen. Når designet er komplett, begynner man med implementeringen. Etter at implementeringen er ferdig, er det nødvendig med testing av programvaren for å finne ut om det er eventuelt feil i kildekoden. Etter testingen begynner man med å rette opp eventuelle feil i kildekoden. Denne fremgangsmåten er illustrert i figur 3.1.



Figur 3.1: Waterfall modell

3.3 Valg av utviklingsmodell

Ved utviklingen av applikasjonen vil vi følge en av utviklingsmodell nevnt i avsnitt 3.2 slik at vi skal få en strukturert og oversiktlig måte å arbeide på. Vi har sett på de forskjellige metodologiene for å utvikle en programvare, og vi har kommet frem til at waterfallmodellen er den vi vil følge for å utvikle vår prototyp. Vi synes at denne modellen er ryddig og strukturert. Fordeling av arbeid i faser er en oversiktlig måte som gjør at vi kan fokusere på en fase av gangen før vi kan gå videre til neste. Denne måten gjør at vi slipper å gjøre flere ting på en gang, noe som kan føre til kaos i arbeidet med utviklingen av prototypen.

4 Systemdesign

Dette kapitlet beskriver utformingen av backupsystemet. Utviklingen av systemet vil bli beskrevet ved hjelp av use case, meldingssekvens og klassediagrammer for å vise hvordan strukturen i koden blir og hvilke meldinger som blir overført mellom klient – server og mellom klientene. Sikkerhet er en viktig del av et slikt system da all overføring av data foregår over internett. Dette kapitlet gir en beskrivelse av hvilke type sikkerhet som velges for overføring av data og ved lagring av data hos klientene. GUI vil bli beskrevet med bilder og en forklaring på rekkefølgen for å navigere seg fram or å utføre de forskjellige funksjonene.

4.1 Design metodologi

Metodologien som er benyttet for å designe dette backupsystemet er objekt orientert analyse og design. Modelleringsspråket vi vil benytte er UML (Unified Modeling Language). UML er et språk som benyttes for å lage en abstrakt modell av et system [32]. UML benytter en grafisk representasjon for å beskrive designet av en applikasjon. Den objektorienterte analysen skal modellere problemet, slik at vi lettere kan løse de problemene som er beskrevet. Designet skal beskrive hvordan problemene skal løses, og det skal gi en detaljert beskrivelse av hvordan systemet skal utvikles.

4.2 Systembeskrivelse

4.2.1 Systemoversikt

Dette er et 2peer system som består av en server og flere klienter. Serveren håndterer registrering av brukere, innlogging, utlogging, har oversikt over hvor den enkelte peer befinner seg og initialisere kontakt mellom to peers som ønsker kommunikasjon mellom seg. Serveren lytter kontinuerlig til innkommende forespørsler fra klientene, og utfører de handlinger det er forespurt om.

Sikkerhetskopiering og gjenoppretting vil foregå direkte mellom klientene. Når en klient ønsker å sikkerhetskopiere data, sender den en forespørsel til mottaker som svarer med en OK melding om den er klar til å motta data. Klienten begynner så overføring av data, og mottakeren lagrer data lokalt på sin maskin. Ved gjenoppretting sender klienten en forespørsel om hvilke data den ønsker å gjenopprette, og senderen av data gjør seg klar til å sende data som er forespurt.

Figuren 4.1 viser hvordan meldinger blir sendt mellom klient og server for å utføre de forskjellige funksjonene.

4.2.2 Hardware

Det stilles ingen krav til hardware for dette systemet. Man trenger en PC med Windows og .NET 2.0 installert, og man må være tilknyttet en bredbåndslinje mot internett.

4.2.3 Datakommunikasjon

Kommunikasjonen i dette systemet vil foregå over internett, og protokollen for overføring av data er TCP/IP. Nettverkskortet må følge IEEE 802 familien [34].

4.2.3.1 Protokoller

Protokollene som vil bli benyttet for å overføre meldinger og data vil være TCP/IP. TCP sikrer at data ikke går tapt, som er viktig ved overføring av viktige filer. Kommunikasjonsmedier som støtter disse protokollene skal kunne virke på dette systemet.

4.2.3.2 Porter

For at server skal kunne motta meldinger fra klienter skal denne lytte på port nummer 8080. Når data skal sendes mellom klientene, skal denne kommunikasjonen foregå på port nummer 1000.

4.2.4 Software

Klientapplikasjonen er designet for operativsystemene Windows 98 eller nyere, inkludert Windows Vista med .NET framework versjon 2.0 installert [35].
Serverapplikasjonen er designet også her for Windows 98 eller nyere, eller man kan benytte operativsystemene Windows 2000 eller 2003 Server med .NET framework versjon 2.0 installert. Det er ikke nødvendig med ytterligere programvare installert for dette systemet.

4.3 Systemarkitektur

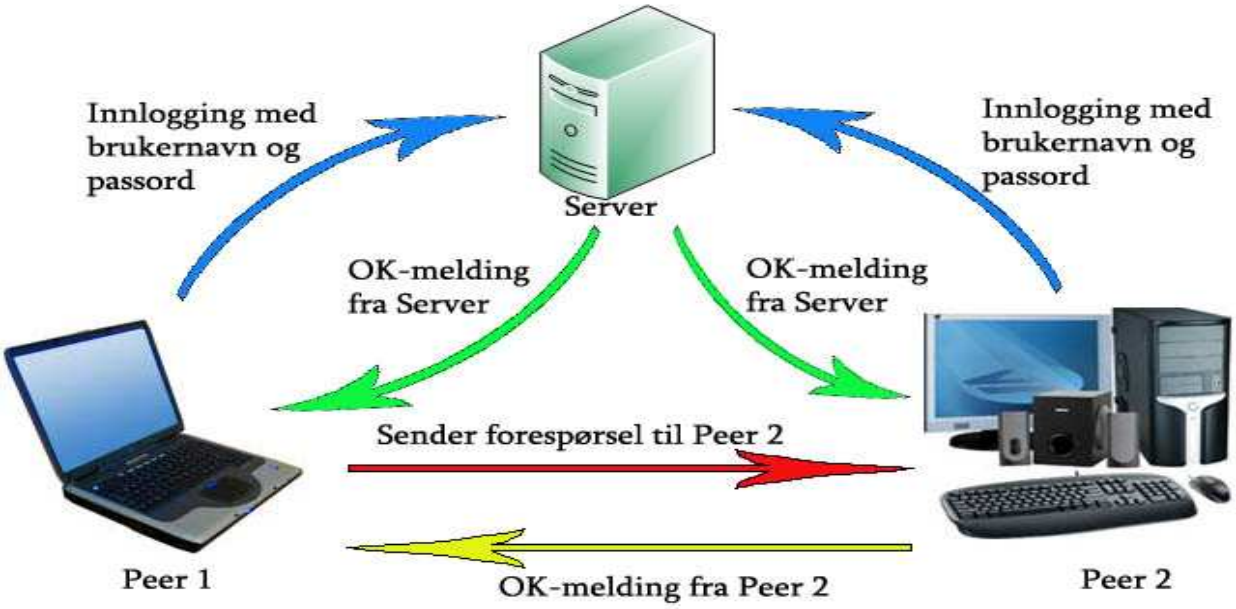
4.3.1 Utviklingsverktøy

Utviklingsverktøyet som skal benyttes for å utvikle backupsystemet er Visual Studio 2005 [36], og programmeringsspråket er C# [37, 42] med .NET framework versjon 2.0 som beskrevet i kravspesifikasjonen avsnitt C.2.5.1.

4.3.2 Skisser over arkitektur

4.3.2.1 Innlogging av klienter

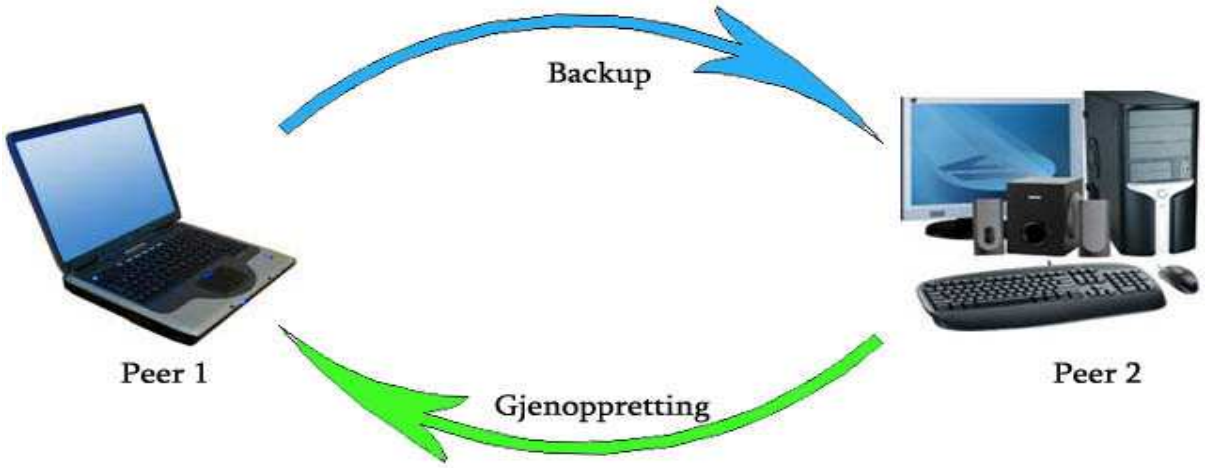
Før man kan foreta sikkerhetskopiering og gjenoppretting av data må man logge seg på serveren med brukernavn og passord. Dette gjøres for at man skal til enhver tid vite om man er pålogget eller ikke, og hvor den enkelte peer befinner seg. Slik informasjon vil ligge på serveren slik at klientene kun sender forespørsel om hvor den enkelte peer befinner seg og dens status. Når en klient logger seg på serveren vil den først sende en forespørsel om at den vil logge seg på. Den sender da innloggingsinformasjon til serveren, og verifiserer brukeren om dette er riktig. Klienten vil også fortelle hvilken IP adresse den er konfigurert med, slik at serveren kan ha oversikt over hvor den befinner seg. Etter at klienten har autentisert seg, kan den sende forespørsel om andre klienter for å finne ut hvor de befinner seg, samt sjekke statusen og deretter kan kommunikasjon mellom klientene starte. Dette er illustrert med en figur under, hvor vi har 1 server og 2 klienter.



Figur 4.1: Innlogging og kommunikasjon mellom 2 peers

4.3.2.2 Sikkerhetskopiering og gjenoppretting

Etter at autentiseringen er fullført, kan man starte sikkerhetskopiering og gjenoppretting av data. Når en peer ønsker å gjenopprette data sendes det en forespørsel til den peeren man ønsker å gjenopprette data fra, som vist i figuren over. Når ”peer1” har valgt hvilke filer den ønsker å gjenopprette, gjør ”peer2” seg klar til å sende de forespurte dataene til ”peer1”. Overføring av data vil foregå direkte mellom klientene, da det er lite hensiktsmessig å la data gå via server. Dette er vist i figur 4.2.



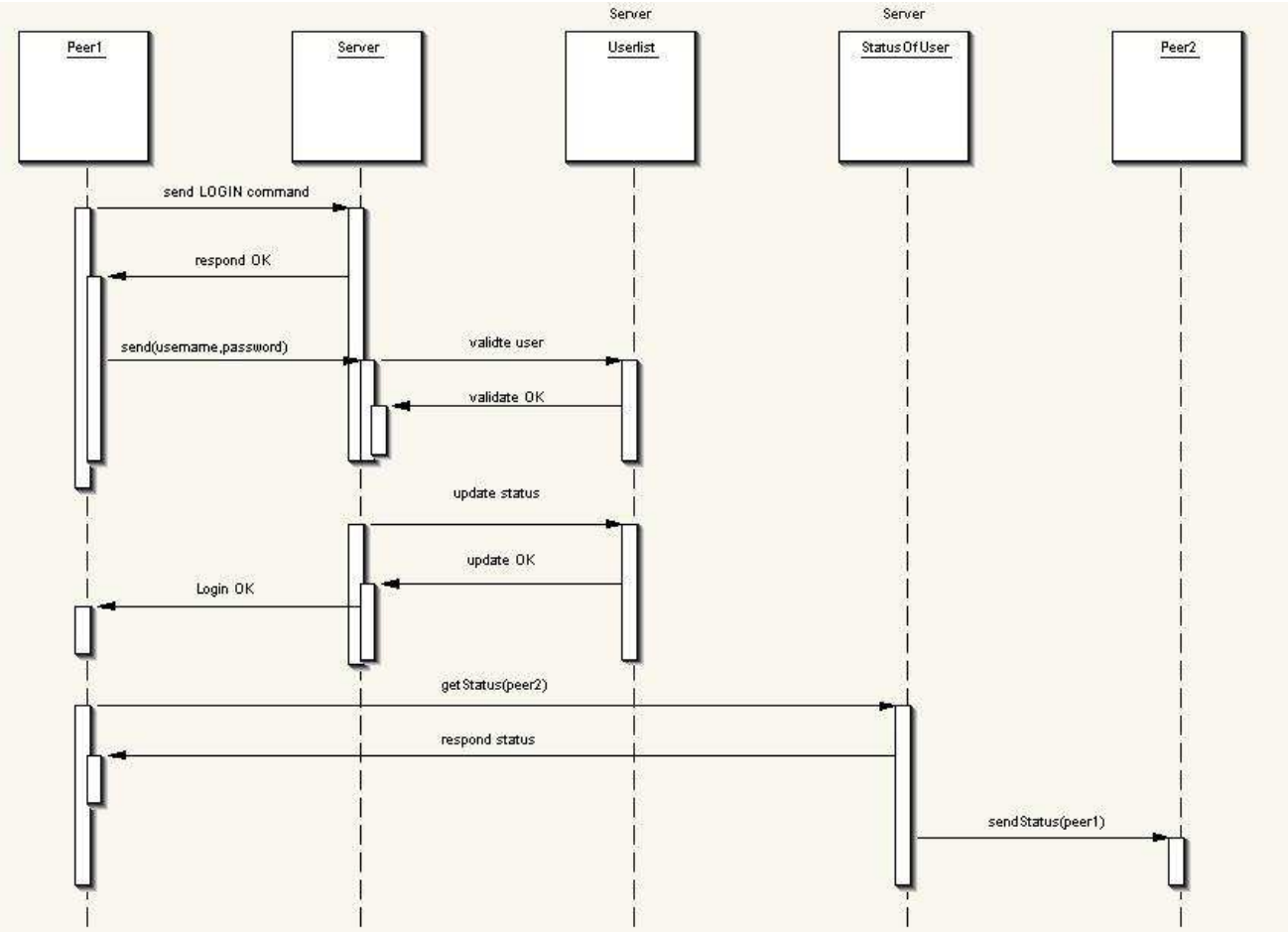
Figur 4.2: Sikkerhetskopiering og gjenoppretting

4.3.3 Sekvensdiagrammer

Dette avsnittet beskriver hvordan meldingssekvensene foregår for de viktigste funksjonene i vårt backupsystem. Sekvensdiagrammene vi skal se nærmere på er for innlogging, sikkerhetskopiering, gjenoppretting og legge til brukere. Sekvensdiagrammene for registrering av ny bruker og sletting av eksisterende konto har vi valgt å ikke se nærmere på, da de er av mindre betydning for hvordan klientene skal kunne kommunisere med hverandre. Sekvensdiagrammet for ulogging og sletting av brukere som er lagt til i klientens brukerliste da disse er nesten identiske for sekvensene for innlogging og å legge til brukere.

4.3.3.1 Innlogging

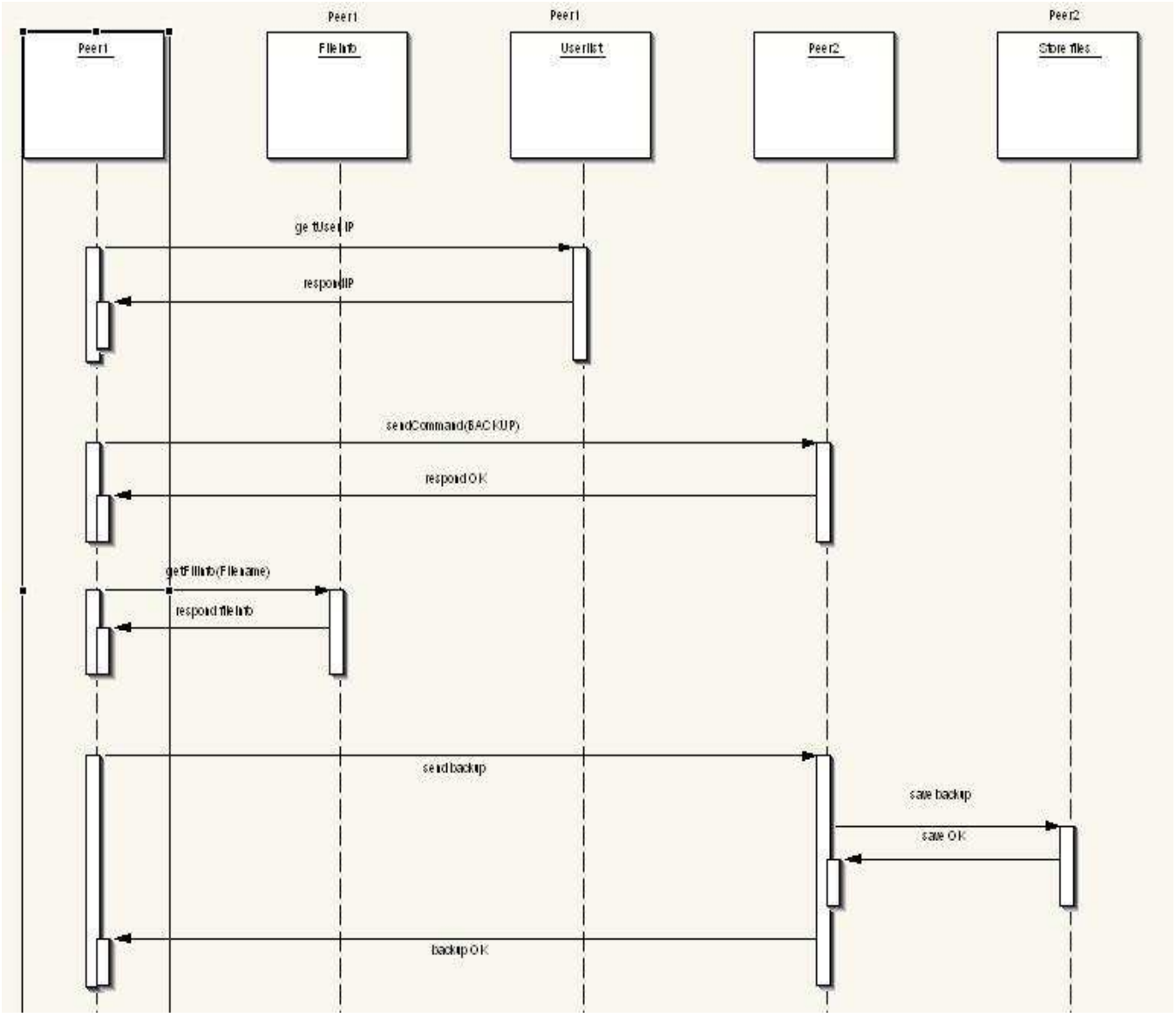
Dette diagrammet viser meldingsutvekslingen mellom serveren og 2 peers. ”Peer1” er brukeren som ønsker å logge seg på serveren, og serveren er den som lytter til forespørsler. ”Userlist” benyttes for å oppdatere brukerinformasjonen til ”peer1”. ”StatusOfUser” benyttes til å sende oppdatert status til brukere. ”Peer2” er den ”Peer1” forespør status etter.



Figur 4.3: Sekvensdiagram for innlogging

4.3.3.2 Sikkerhetskopiering

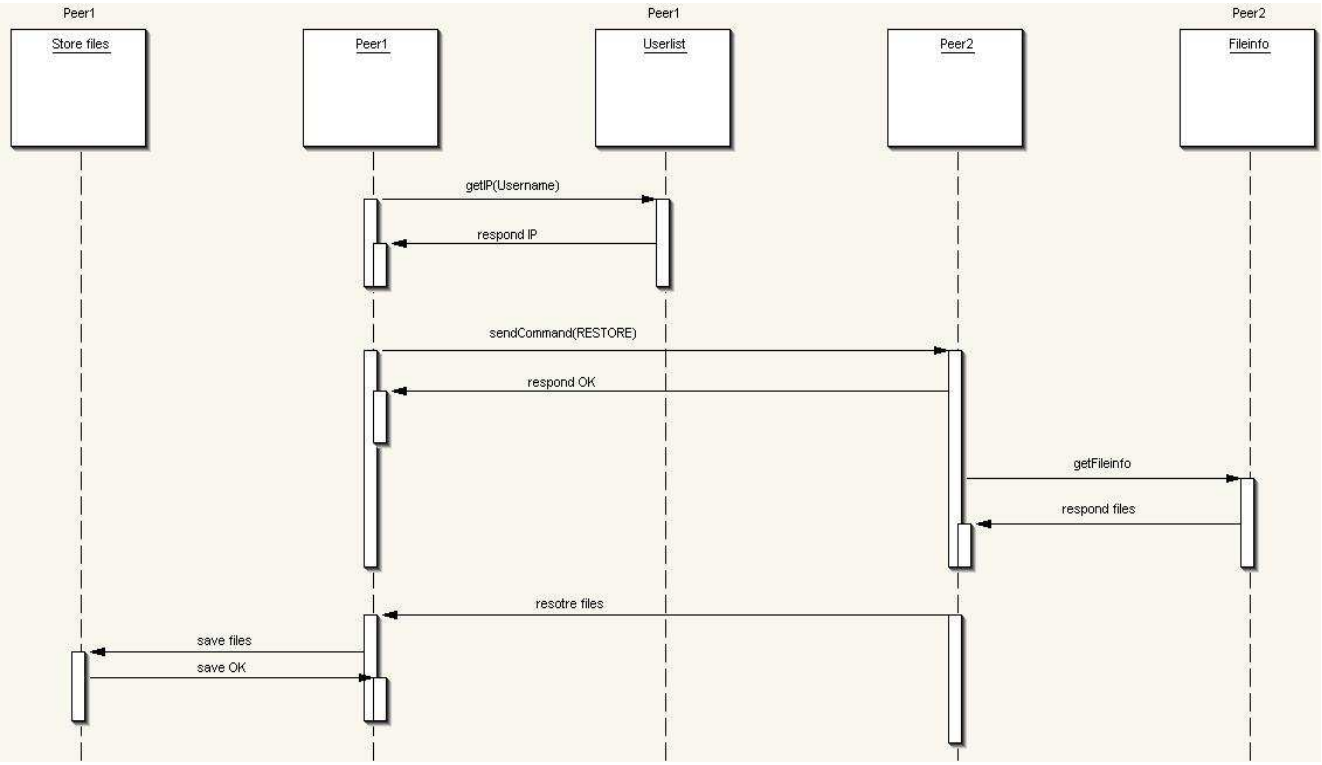
”Peer1” er den som skal utføre sikkerhetskopiering av filer, og ”Peer2” er den som skal motta filene og lagre de lokalt på sin maskin. ”Fileinfo” benyttes for å hente informasjon om de filene som skal overføres, som filnavn, filtype og fillengde. ”Store files” er metoden for å lagre filene som blir overført av ”Peer1”.



Figur 4.4: Sekvensdiagram for sikkerhetskopiering

4.3.3.3 Gjenoppretting

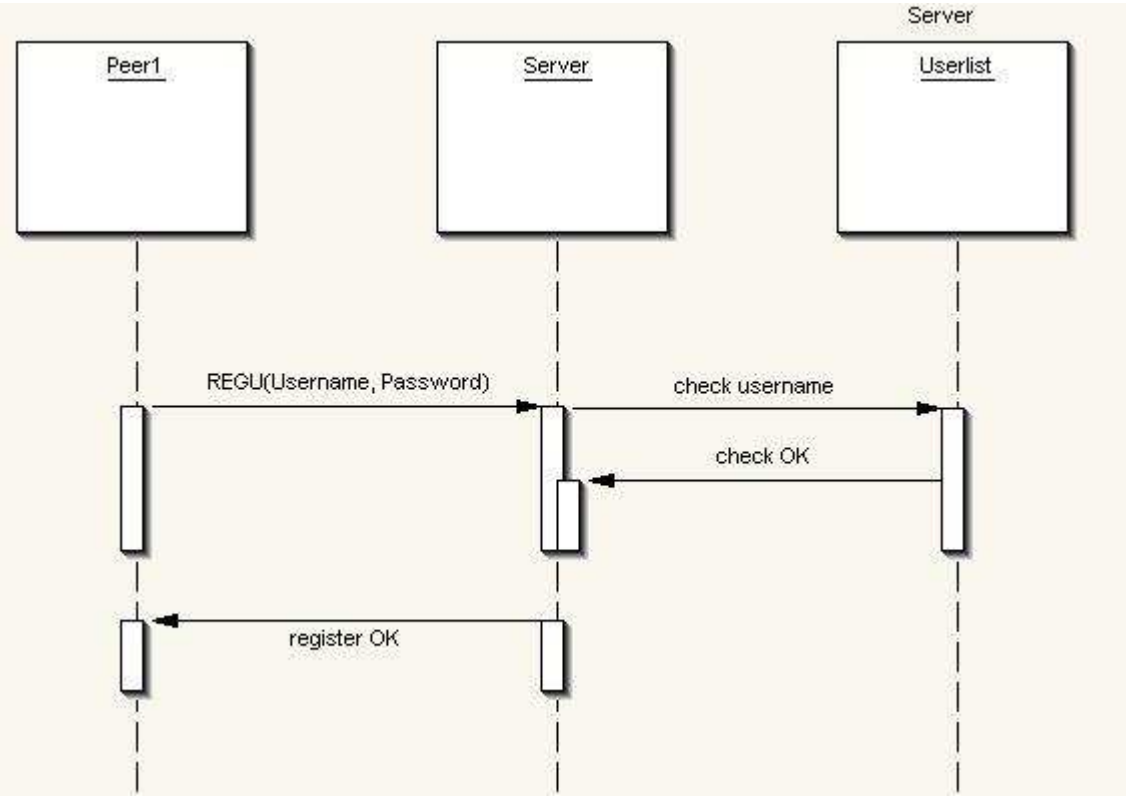
Objektene i sekvensdiagrammet nedenfor er det samme som i sekvensdiagrammet for sikkerhetskopiering.



Figur 4.5: Sekvensdiagram for gjenoppretting

4.3.3.4 Registrering av bruker

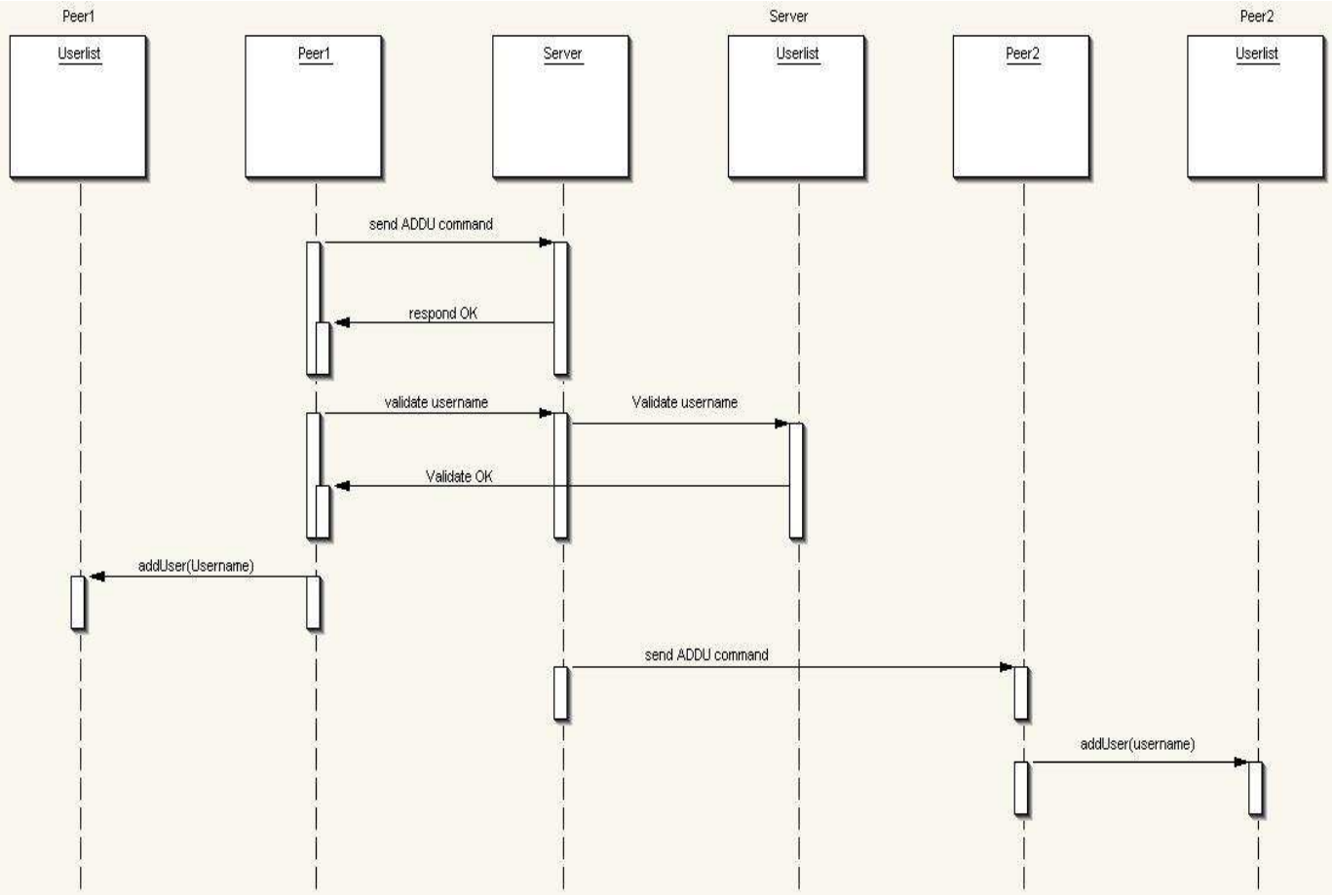
Dette sekvensdiagrammet viser meldingene mellom ”Peer1” og server når en bruker ønsker å registrere seg.



Figur 4.6: Registrering av bruker

4.3.3.5 Legg til bruker

Sekvensdiagrammet under viser meldingsutvekslingen mellom server og to peers når en ”Peer1” ønsker å legge til ”Peer2”. Objektene er som beskrevet i de tidligere sekvensdiagrammene.



Figur 4.7: Sekvensdiagram for legging til ny bruker

4.4 Detaljert klassedesign

Dette avsnittet inneholder en detaljert beskrivelse av de klassene som er nødvendig å ha med i vårt system. Hver klasse inneholder metoder som benyttes for å utføre bestemte funksjoner utført av brukeren av systemet, som også vil bli beskrevet i detalj.

4.4.1 Server

Server skal administrere brukere av backupsystemet. Den har en database som inneholder informasjon som brukernavn, passord, e-postadresse og IP adresse. Serveren har også som oppgave å kontrollere brukernavn og passord ved innlogging og holde styr på hvilke IP adresser den enkelte host har til enhver tid mens de er tilkoblet. Klassene som blir beskrevet for serverapplikasjonen er:
AdminUsers, Password, IpLookUp, Connect, CommandHandler og StatusOfUser.

4.4.1.1 AdminUsers

I denne klassen inneholder det metoder for legge og slette brukere i databasen, hente brukernavn og passord fra databasen.

Figur 4.8: Klassen AdminUsers på server

Forklaring på metodene i denne klassen:

addUser: Denne metoden legger til brukere i brukerdatabasen når en ny bruker registrerer seg. Den vil først sjekke om brukernavnet finnes fra før av, og gjør den ikke det vil brukeren bli lagt til.

deleteUser: Denne metoden sletter all informasjon om brukeren når en bruker sender en forespørsel om å deaktivere kontoen.

getIP: Denne metoden kalles når man forespør etter IP adressen til en gitt bruker.

setIP: Denne metoden kalles for å oppdatere IP adressen til en gitt bruker

getPassword: Denne metoden benyttes for å hente passordet til en gitt bruker når man skal sjekke om passordet som er sendt stemmer.

getStatus: Denne metoden henter status i brukerfilen til en gitt bruker

updateStatus: Metoden oppdaterer status til en gitt bruker i brukerlisten når den logger seg på eller av.

4.4.1.2 Password

Denne klassen tar for seg administrering av passord til brukere. Inneholder metoder for å editere passord hvis brukerne vil endre på sitt passord, og sende nytt passord hvis brukerne har glemt sitt passord.

Figur 4.9: Klassen Password på server

Forklaring på metodene:

editPassword: Når en bruker sender en forespørsel om å bytte passord, blir denne metoden kalt opp og sørger for at passordet til brukeren blir oppdatert i passordfilen på serveren.

sendPassword: Ved forespørsel om sending av glemt passord til e-post konto vil denne metoden bli kalt og sørger for at passord blir sendt på e-post.

4.4.1.3 Encryption

Denne klassen håndterer kryptering av meldinger som blir sendt mellom server og klient

Figur 4.10: Klassen Encryption

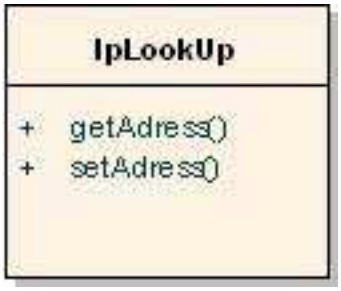
Forklaring på metodene:

encryptMessage: Denne metoden krypterer meldingen som blir sendt fra klient til server

decryptMessage: Når serveren mottar en melding fra en klient, blir denne metoden benyttett for å dekryptere meldingen.

4.4.1.4 IpLookUp

Denne klassen håndterer IP adresser til brukere til enhver tid. Den oppdaterer IP adressen til den enkelte bruker ved forandring, og sørger for at den enkelte bruker får vite dette.



Figur 4.11: Klassen IpLookUp på server

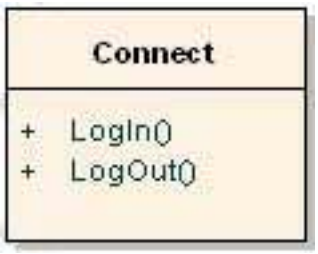
Forklaring på metodene:

getAdress: Henter IP adressen til den enkelte bruker ved forespørsler fra klienter.
Henter inn brukernes IP adresser til enhver.

setAdress: Denne metoden kalles når serveren må oppdatere en IP - adresse til en bruker i brukerdatabasen.

4.4.1.5 Connect

Denne klassen håndterer innlogging og utlogging av brukere når det sendes en forespørsel om dette.



Figur 4.12: Klassen Connect på server

Forklaring på metodene:

LogIn: Nå er bruker logger seg inn på systemet brukes denne metoden for å utføre login sekvensen. Ved feil inntasting av brukernavn/passord vil prosessen avbrytes. Ved riktig inntasting av brukernavn/passord vil denne metoden oppdatere status på brukeren, og sende tilbake svar om at innloggingen var vellykket.

LogOut: Denne metoden blir kalt når en bruker logger ut av systemet, og oppdaterer statusen til brukeren i brukerdatabasen.

4.4.1.6 CommandHandler

Denne klassen mottar og utfører kommandoer etter forespørsel fra klientsiden.

Figur 4.13: Klassen CommandHandler på server

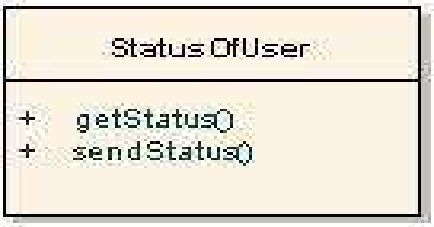
Forklaring på metodene:

receiveCommand: Denne metoden håndterer de forespørslene som kommer inn fra klienten, slik at serveren er klar over hvilke klasser og metoder i serverapplikasjonen som skal aktiveres.

sendCommand: Denne metoder sender ut forespørsler til klienter og responskommandoer til disse ved forespørsel fra en klient.

4.4.1.7 StatusOfUser

Ved innlogging og utlogging blir denne klassen kalt for å oppdatere den enkelte brukers status



Figur 4.14: Klassen StatusOfUser på server

Forklaring på metodene:

sendStatus: Når en bruker logger seg på eller logger seg av systemet, vil den metoden sørge for å gi beskjed om den nye statusen til de brukerne som denne har i sin brukerliste. Metoden vil også gi beskjed om de andre brukernes status.

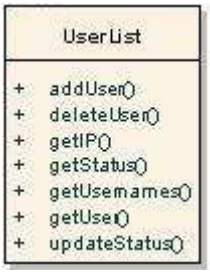
getStatus: Denne metoden blir kalt når en bruker logger seg på, for å oppdatere statusen til denne brukeren i brukerdatabasen.

4.4.2 Klient

Dette avsnittet tar for seg klassene og metodene som er definert i klient applikasjonen. Denne applikasjonen er installert på brukerens PC, og inneholder følgende klasser: Users, Password, Connect, Backup, Restore og Tasks.

4.4.2.1 Userlist

Denne klassen tar for seg administrering av buddies. Den inneholder metodene addUser() og deleteUser().



Figur 4.15: Klassen Userlist på klient

Forklaring på metodene:

addUser(): addUser inneholder alle funksjoner nødvendig for å legge til en bruker hos klienten. Dette går ut på å legge til brukere i brukerdatabasen, slik at man senere kan velge hvilke bruker man ønsker å utføre backup og restore mot.

deleteUser(): deleteUser inneholder funksjonen for å slette en bruker fra brukerdatabasen til klienten. Den inneholder de nødvendige operasjonene for å slette brukeren fra brukerdatabasen og informere denne brukeren om at han/hun er slettet.

4.4.2.2 Password

Denne klassen tar for seg administrering av passord til brukeren. Metoden som inneholder i denne klassen er editPassword() og getPassword.

Figur 4.16: Klassen Password på klient

Forklaring på metodene:

editPassword(): Denne metoden inneholder funksjonen for å kunne forandre innloggingspassordet til brukeren. Dette består i å oppdatere passordet på sin egen passordfil og informere serveren om at passordet er endret.

getPassword(): Denne metoden inneholder funksjoner for å få tilsendt passord om dette er glemt. Metoden inneholder funksjoner for å si i fra dette til serveren slik at den kan sende ut det glemte passordet på mail til den aktuelle brukeren.

4.4.2.3 Encryption

Denne klassen håndterer kryptering av meldinger som sendes til server og andre peers. Den håndterer også dekryptering når data blir sendt til en peer.

Figur 4.17: Klassen Encryption på klient

Forklaring på metodene:

encryptUserinfo: Metoden benyttes for å kryptere brukerinformasjon som skal bli skrevet til brukerfilen.

decryptUserinfo: Denne metoden benyttes for å dekryptere brukerinformasjon som blir lest i fra brukerfilen, slik at applikasjonen kan sjekke at brukerinformasjonen stemmer.

encryptMessage: Denne metoden benyttes for å kryptere meldinger som blir sendt til server og klient før de blir sendt over internett.

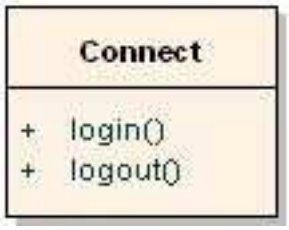
decryptMessage: Denne metoden dekrypterer data som blir sendt fra server og andre peers, slik at applikasjonen kan lese hva som blir sendt.

encryptFile: Benyttes for å kryptere filer før de blir sendt for sikkerhetskopiering. Dette sikrer at den som mottar dataene ikke kan se hva som blir sendt.

decryptFile: Denne metoden benyttes for å dekryptere data man gjenoppretter, slik at de blir leselig når de er gjenopprettet.

4.4.2.4 Connect

Connect - klassen inneholder de nødvendige funksjoner for å kunne logge seg på og logge seg av systemet. Klassen består av metodene login() og logout().



Figur 4.18: Klassen Connect på klient

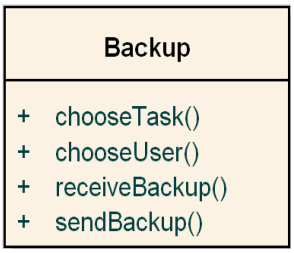
Forklaring på metodene:

Login(): Denne metoden inneholder nødvendige funksjoner for å kunne logge seg på et system. Metoden sender brukernavn og passord til server slik at serveren kan verifisere brukeren og få beskjed om at man ønsker å logge seg på. Når brukeren er godkjent vil statusen til brukeren bli endret til pålogget, og man kan da utføre backup og restore handlinger.

Logout(): Denne metoden inneholder nødvendige funksjoner for å logge ut av systemet. Den gir beskjed til server om at klienten har avsluttet applikasjonen, slik at server kan oppdatere status til brukeren.

4.4.2.5 Backup

Denne klassen inneholder funksjoner for det å kunne ta backup mot en annen peer, og motta backup ved forespørsel fra andre brukere. Denne klassen inneholder metodene chooseTask(), chooseUser() og sendBackup(), receiveBackup().



Figur 4.19: Klassen Backup på klient

Forklaring på metodene:

chooseTask(): Denne metoden inneholder funksjoner for å hente fram informasjon om den oppgaven man ønsker å utføre. Dette innebærer informasjon om filer og mapper det skal tas backup av.

chooseUser(): Metoden inneholder funksjoner for å velge den man skal utføre backup mot. Funksjoner henter fram IP - adressen til brukeren som er valgt slik at data blir sendt til riktig host.

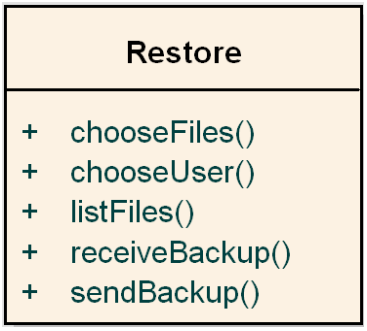
sendBackup(): Denne metoden sørger for at filer og mapper blir overført til hosten som skal

motta dataene.

receiveBackup(): Denne metoden sørger for at filer og mapper den mottar fra en host som utfører backup blir mottatt og lagret på den bestemte plasseringen på harddisken.

4.4.2.6 Restore

Klassen Restore inneholder nødvendige funksjoner for å utføre gjennoppretting av data, og sende data ved forespørsel fra andre brukere. Funksjonene i denne klassen er chooseFiles(), chooseUser(), listFiles(), receiveBackup() og sendBackup().



Figur 4.20: Klassen Restore på klient

Forklaring på metodene:

chooseFiles(): Denne metoden inneholder funksjoner for å hente fram informasjon om de filene som er valgt for å utføre gjenoppretting av data.

chooseUser(): Metoden henter fram informasjon som IP – adresse til brukeren man har valgt å gjenopprette data fra. Denne informasjonen blir benyttet i receiveBackup metoden slik at man gjenoppretter data fra riktig host.

listFiles(): Denne metoden henter informasjon om filer og mapper som det er tatt backup av hos den brukeren man ønsker å utføre gjenoppretting fra. Disse filene og mappene blir listet opp i en dialogboks i applikasjonen, slik at man kan velge hvilke filer og mapper man ønsker å gjenopprette.

receiveBackup(): Denne metoden inneholder funksjoner for å motta data som skal gjenopprettes, og lagrer de på en bestemt plassering på harddisken.

sendBackup(): Denne metoden utfører de operasjonene som skal til for å sende data til en bruker som sender en forespørsel om gjenoppretting av data.

4.4.2.7 CommandHandler

Denne klassen mottar og utfører kommandoer etter forespørsel fra klientsiden.

Figur 4.21: Klassen CommandHandler på klient

Forklaring på metodene:
receiveCommand: Tar imot en kommando om hva som skal utføres.

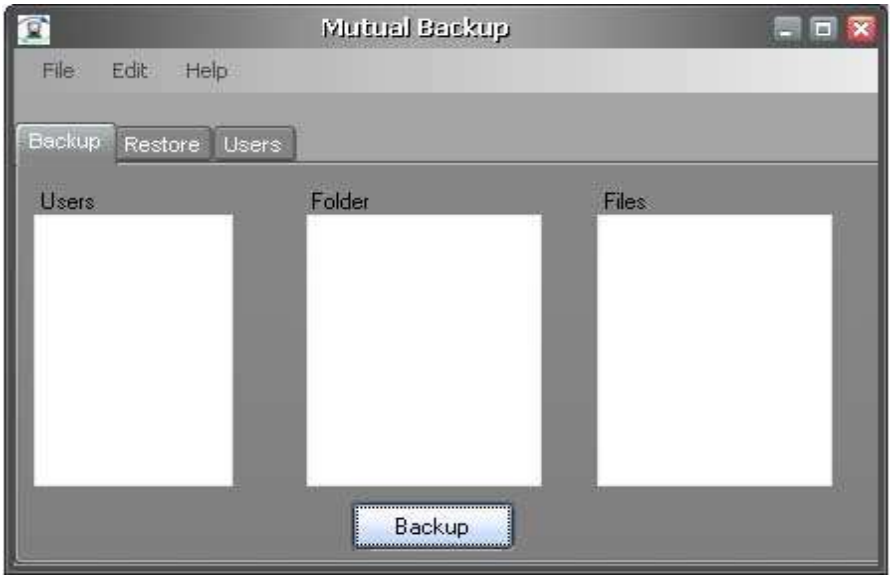
sendCommand: Utfører en kommando etter forespørsel fra en annen klient.

4.5 GUI design

Dette avsnittet beskriver og viser visuelle bilder av hvordan det fullstendige backupsystemet vil se ut. GUI designet består av et vindu med tre tabulatorer: Backup, Restore og Users. I tillegg finnes det en filmeny.

4.5.1 Backup

I denne tabulatoren er det tre listebokser. Den første listeboksen fra venstre viser en liste av brukere som er lagt til av eieren av applikasjonen. Listeboksen i midten viser en liste av mapper som er på maskinen det skal foretas sikkerhetskopiering fra. Den siste listeboksen lister opp filene som er i den mappen du befinner deg i. Knappen ”Backup” aktiverer sending av de filene man ønsker å sikkerhetskopiere til den brukeren man har valgt. Figur 4.22 viser hvordan dette vil se ut.

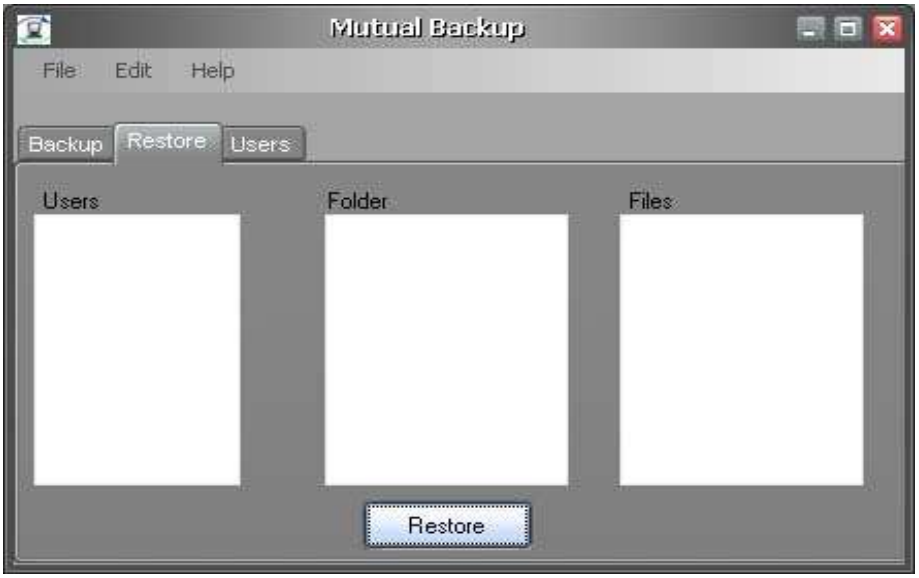


Figur 4.22: Mutual Backup - Backup tabulator

For å utføre sikkerhetskopiering må man først velge den brukeren man ønske å ta sikkerhetskopiering til og deretter merke av mappe og/eller fil før man kan trykke på knappen ”Backup”. Da vil programmet sende den valgte mappen/filen til den utvalgte brukeren. Det kan bare sendes til en utvalgt bruker av gangen.

4.5.2 Restore

Denne tabulatoren inneholder de samme listboksene som i Backup - tabulatoren. Folder listboks og file listboks viser filene som er lagret hos den brukeren man ønsker å forta gjenoppretting av. Dette er vist i figur 4.23.



Figur 4.23: Mutual Backup - Restore tabulator

For å utføre gjenoppretting av data må man først velge brukernavnet til den man ønsker å ta gjenoppretting fra. Deretter velger man filer og mapper man ønsker å gjenopprette, og deretter trykker man på ”Restore” knappen. Deretter må man velge hvilke mappe man ønsker å lagre de gjenopprettede data i.

4.5.3 Users

Denne tabulatoren inneholder en listboks over eksisterende brukernavn og to knapper: ”Add User” for å legge til brukere og ”Delete User” for å slette brukere fra brukernavn listboksen. Det siste feltet er en tekstboks hvor man kan skrive inn brukernavnet man ønsker å legge til i listen over brukere.

Figur 4.24: Mutual Backup - Users tabulator

For å legge til en bruker må man først skrive inn brukernavnet i tekstboksen til høyre. Deretter trykker man på ”Add User” knappen og applikasjonen sender en forespørsel om man får tillatelse til å legge til denne brukeren. Får man dette vil denne brukeren bli lagt til i brukernavnlisten. For å slette en bruker velger man den man skal slette i listboksen til venstre og trykker deretter på ”Delete User” knappen.

4.5.4 Menylinje

Menylinjen består av følgende funksjoner: File, Edit og Help.

4.5.4.1 File

File - menyen består av Exit - element som avslutter programmet.

Figur 4.25: File meny

4.5.4.2 Edit

Edit menyen består av elementene ”Add User”, ”Delete User”, ”Change password” og ”Option. I figur 4.26 viser hvordan rekkefølgen på disse er.

Figur 4.26: Edit meny

Trykker man på ”Add User” knappen får man opp en dialogboks hvor man kan skrive opp et brukernavn som man kan legge til. Med å trykke på ”Delete User” knappen får man opp en liste over brukere som er i brukernavnlisten og man kan da velge den man vil slette fra listen. Ved å trykke på ”Change password” knappen kommer det opp et pop-up vindu hvor man kan velge å enten endre innloggingspassordet eller få tilsendt nytt passord om dette er glemt. ”Options” menyen inneholder informasjon om brukeren som er registrert, og man kan her endre brukeropplysninger.

4.5.4.3 Help

I Help menyen kan man velge menyelementene ”About” og ”Help”. Under vises figur 4.27 Help menyen.

Figur 4.27: Help meny

Figur 4.28: About vindu

Når man klikker på ”About” vil det åpne et pop-up vindu som viser informasjon om programmet. Når man klikker på ”Help” vil det dukke opp en hjelpemeny som skal vise brukerne hvordan programmet skal brukes på en riktig måte.

4.6 Brukerinformasjon

All brukerinformasjon til hver enkelt bruker vil bli lagret i en XML fil på både klienten og serveren.

4.6.1 Brukerinformasjon server

På serveren vil all brukerinformasjon om hver enkelt bruker ligge i XML filen. Denne informasjonen innebærer brukernavn, passord, status til bruker, IP – adresse og e-post adresse til den enkelte bruker. All informasjon som ligger i filen vil bli kryptert i henhold til avsnitt C.4.3.

4.6.2 Brukerinformasjon klient

Brukerinformasjon om den enkelte bruker klienten ønsker å overføre sikkerhetskopiering mot blir også her lagret i en XML fil. Brukerinformasjon som blir lagret i denne filen er brukernavn, IP adresse og status til den enkelte bruker. Som i avsnitt 4.4.1 blir også denne filen kryptert som beskrevet i avsnitt C.4.3.

4.7 Sikkerhet

4.7.1 Autentisering

Autentiseringsmetoden som skal bli benyttet i dette systemet er brukernavn og passord. Dette er en enkel form for autentisering som er tilgjengelig for alle samtidig som den gir en tilstrekkelig sikkerhet til et slikt system. Brukernavn og passord vil bli opprettet av brukeren når denne personen registrerer seg. Brukernavnet vil være valgfritt etter brukerens ønske. Det vil bli satt restriksjoner på passordet, noe som er gjort for at brukerne ikke oppretter passord som er for enkle for andre å finne ut av. Kriteriet er at passordet skal være på minimum 8 tegn, bestående av store og små bokstaver, og inneholde minst ett spesialtegn eller et tall. Et slikt passord kan for eksempel være «London/99». Dette gjør det vanskelig for andre å knekke passordet, samtidig som det gir en trygghet for brukeren som benytter systemet.

4.7.2 Kryptering

4.7.2.1 Kryptering av autentiseringen

For at andre ikke skal kunne se hva slags brukernavn og passord man benytter vil all informasjon som blir overført mellom server og klient bli kryptert. Krypteringsalgoritmen vi vil benytte er 128 bits AES [38], som regnes som en sikker algoritme. Informasjonen som utveksles mellom klient og server er små mengder data, og en nøkkellengde på 128 bit vil i dette tilfelle være tilstrekkelig.

4.7.2.2 Kryptering av overføringen

Ettersom overføringen av data foregår over internett vil det bli opprettet en VPN [39] tunnel mellom hostene som skal foreta backup. IPSec [40] vil bli benyttet for å sikre IP

kommunikasjonen mellom hostene. Krypteringsalgoritmen som vil bli brukt er AES, og nøkkelstørrelsen er 256 bit. Dette er en symmetrisk nøkkelalgoritme hvor partene blir enige om en felles nøkkel for kryptering og dekryptering.

4.7.2.3 Kryptering av filer

Når det blir tatt sikkerhetskopi av filer lagret på host, må disse krypteres slik at de blir uleselig for de som mottar filene. Kryptering er derfor nødvendig, og AES blir brukt til dette. Krypteringsnøkkelen vil også her være på 256 bit. Krypteringsnøkkelen vil kun være kjent for eieren av data, og data vil ikke bli dekryptert før man gjenoppretter dataene man har tatt sikkerhetskopi av. Dette gjør at data er uleselig for den som oppbevarer data, slik at ingen andre enn eieren av data skal kunne lese dem.

4.7.2.4 Kryptering av brukerinformasjonsfiler

Brukerinformasjonen som blir lagret i XML filen hos både klienten og serveren vil bli kryptert med en AES 128 bit krypteringsnøkkel. Dette gjøres for å sikre integriteten til den enkelte bruker.

5 Prototypdesign

5.1 Introduksjon

Dette kapitlet inneholder en beskrivelse av hvordan GUI er designet og hvordan funksjonene som er tatt med i prototypen er implementert. Dette innebærer funksjoner som sikkerhetskopiering, gjenoppretting, registrering brukere, legge til brukere, slette brukere og innlogging og utlogging. Til slutt vil vi gi en demonstrasjon som skal vise hvordan man tar i bruk backupsystemet. For at det skal være mulig å demonstrere hvordan filer blir overført og gjenopprettet mellom to maskiner, må begge maskinene legge inn prototypen. Hensikten med prototypen er å vise ved demonstrasjon at de nødvendige funksjonene fungerer som beskrevet i systemdesignet, og at de andre tilleggsfunksjonene kan implementeres senere når det er behov for dette.

5.2 GUI forklaring

5.2.1 Server

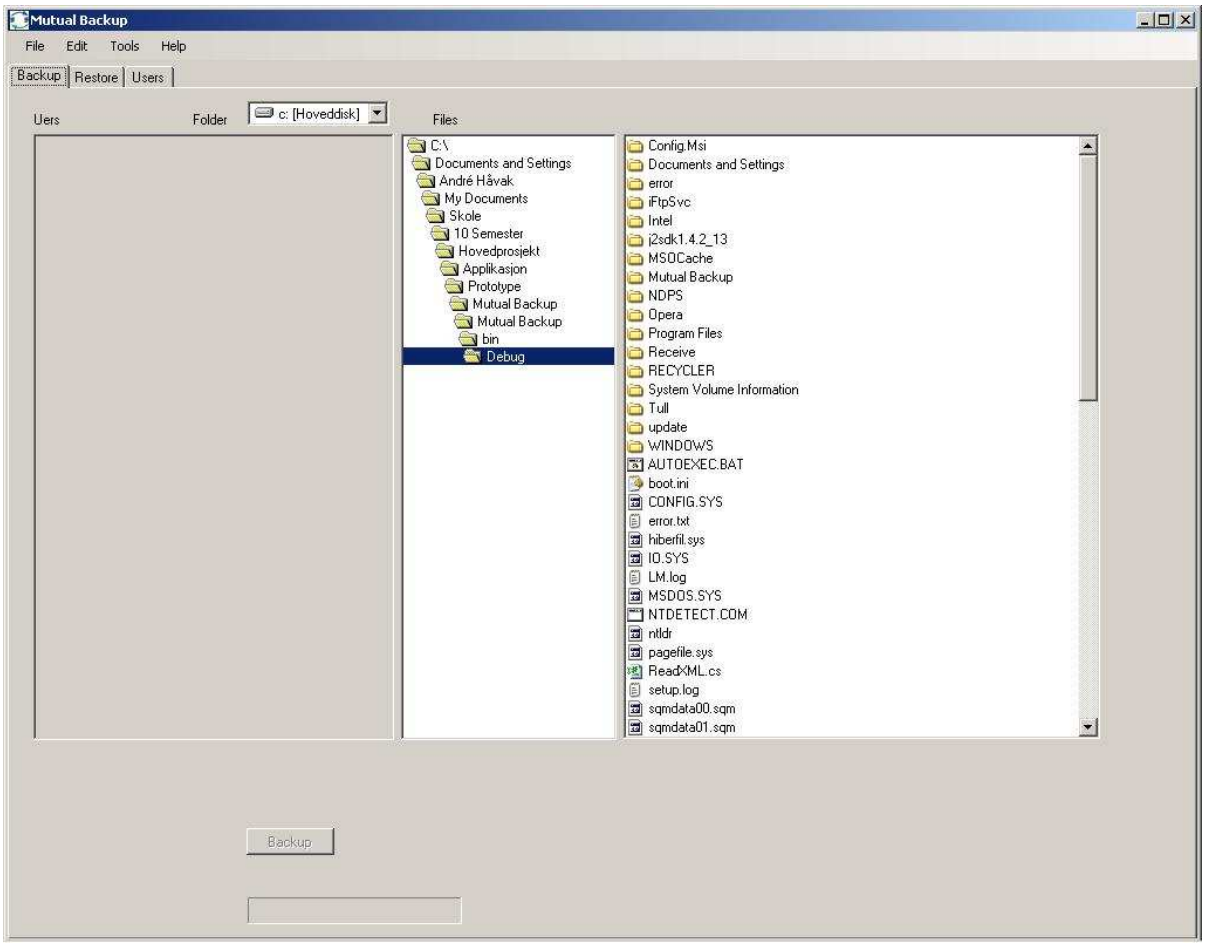
Serverens funksjon er å håndtere forespørsler fra en klient. Serveren har ingen GUI komponenter ettersom alle funksjonskallene foregår via klienten.

Figur 5.1: GUI server

5.2.2 Klient

Klientapplikasjonen inneholder alle funksjoner for å utføre sikkerhetskopiering og gjenoppretting av data. Klienten sender forespørsler til server om innlogging, utlogging, legge til brukere i brukerlisten og sjekk av status til andre brukere. Funksjoner som sikkerhetskopiering og gjenoppretting av filer utføres direkte mellom klientene. Prototypen består av en menylinje og tre

tabulatorer som vist i figuren under. Menylinjen består av menyknappene ”File”, ”Edit”, ”Tools” og ”Help”. Tabulatorene som er implementert er ”Backup”, ”Restore” og ”Users”.I ”Backup” tabulatoren utføres sikkerhetskopiering av filer og i ”Restore” tabulatoren utføres gjenoppretting av filer. I ”Users” – tabulatoren kan man legge til brukere og slette brukere man ønsker å ha i sin brukerliste. Funksjonene i prototypen vil bli beskrevet mer detaljert under avsnittet 5.3.



Figur 5.2: Klient

5.3 Implementering

5.3.1 Protokoller

Meldinger som blir overført mellom server og klient foregår over protokollene TCP/IP som spesifisert i systemdesign avsnitt 4.2.3. Serveren lytter på port 8080 etter innkommende meldinger, og klienten lytter på port 8080 for innkommende meldinger i fra serveren. Overføring av data mellom to peers foregår på port 1000.

5.3.2 Kommandomeldinger

Funksjonskall gjøres via kommandoer som sendes mellom server og klient og to klienter. Når en melding ankommer serveren vil den se på hvilken kommando som er sendt og kalle opp funksjonene i koden som er nødvendig for å utføre forespørselen fra klienten.

5.3.2.1 Meldingsformat

Hver melding starter med en kommando etterfulgt av de data som er nødvendig for å utføre handlingen det er forespurt om. Et eksempel kan være ved innlogging, hvor brukeren først sender en melding til serveren som inneholder følgende:

”LOGIN;IP;”

LOGIN er kommandoen som forteller at klienten ønsker å logge seg på og IP er IP – adressen til den som ønsker å logge seg på. Semikolon benyttes for å skille mellom feltene i meldingen. Meldingsformatene vil bli nøyere beskrevet der det er aktuelt.

5.3.2.2 Sending av melding

Det er tre scenarioer for hvor meldinger blir sendt; som er fra klient til server, server til klient og mellom to klienter. For å sende en melding må en først konvertere meldingen fra en tekststreng til en byte tabell som gjøres med metoden

```
Encoding.ASCII.GetBytes(command.ToCharArray());
```

Når konverteringen er fullført benyttes metoden `netSend.Write(bytesend, 0, lengde)`, hvor bytesend er data som skal sendes, tallet 0 er startindeks for hvor i bytetabellen en skal starte å sende fra og lengde er antall byte som skal sendes.

5.3.2.3 Mottak av melding

Når en server eller klient mottar en melding, må den først dele opp slik at hvert felt i meldingen blir en egen tekststreng. Dette gjøres som vist under

```
for (int i = 0; i < commandLength; i++)
{
      if (commandReceived.Substring(i, 1).Equals(";"))
      {
            numberArgs++;
      }

      else
      {
            commandInfo[numberArgs]+= commandReceived.Substring(i,1);
      }
}
```

Denne metoden går igjennom hvert tegn i meldingen og ser etter tegnet ";". Hver gang et tegn er forskjellig fra ";" legger den til tegnet i commandInfo som er en tabell av tekststrenger. Om tegnet den sjekker lik ";", øker den numberArgs med 1 slik at den begynner å skrive til neste plass i tabellen. Denne prosedyren utføres til det ikke er mer tegn å sjekke.
Når meldingen er analysert vil applikasjonen gå inn i receiveCommand() metoden i klassen CommandHandler for å se hvilken kommando som skal utføres. Under vises hvordan dette gjøres:

```
switch (messageInfo[0])
{
      case "LOGIN":
      sendCommand("READY;", messageInfo[1]);
      break;
```

I dette tilfelle sjekker den om kommandoen som er mottatt er "LOGIN". Stemmer dette utføres `sendCommand("READY;", messageInfo[1])` metoden, og den avslutter testen med hjelp av en `break;` melding.

5.3.3 Brukerinformasjon

Hver bruker som er registrert på serveren blir lagret i en XML – fil som heter "userlist.xml". All brukerinformasjon som blir lagret i filen vil være kryptert so vist i figur 5.3. Som vist i figuren under består XML filen av en rot – node Users. Users består av en eller flere Userinfo noder. En slik node blir opprettet for hver bruker som registrerer seg. Userinfo består av nodene Username, Password, IP og Status. Username og password vil normalt ikke forandre seg ikke, men status kan enten være Offline eller Online. Er status til en bruker online vil node - IP bli satt til den IP – adressen hosten er konfigurert med i det den logger seg på serveren. Er statusen til en bruker Offline vil IP – noden bli satt til en standard verdi. Denne filen vil kontinuerlig bli oppdatert etter hver gang en bruker logger seg på eller av.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Users>
  - <Userinfo>
      <Username>xPuCixjWTS9IzgZNj1JCQQ==</Username>
      <Password>bP0oLs3u6AwJxHluHRWJVw==</Password>
      <IP>xI3oeM5uZNoAGdygD9K0VQ==</IP>
      <Status>pOhAzNBtY0yuh484a9cfiQ==</Status>
    </Userinfo>
  </Users>
```

Figur 5.3: XML fil på server

5.3.4 Sikkerhet

5.3.4.1 Meldinger

Meldinger som blir overført mellom klient og server og mellom klientene vil bli overført kryptert. Krypteringsalgoritmen vi har valgt å implementere er som beskrevet i avsnitt C.4.3 om sikkerhet. I meldinger som blir overført er det sensitiv data som overføres, og det er derfor nødvendig å sikre at ikke andre får tilgang til disse.

5.3.4.2 Overføring av filer

Overføringen av filene mellom to peers vil foregå kryptert som beskrevet i avsnitt C.4.3. Dette er gjort da det er kun de som deltar i overføringen som skal se informasjonen som blir overført.

5.3.4.3 Lagring av filer

Filene som blir overført ved sikkerhetskopiering vil være synlig for den som mottar filene, noe som innebærer at filene ikke vil være kryptert.

5.3.5 Registrering av bruker

Når en bruker installerer klientapplikasjonen på sin PC vil brukeren få beskjed om å registrere et brukernavn og passord. For at brukeren skal bli registrert, må brukeren sende brukerinformasjon til server. Serveren sjekker da om brukeren eksisterer eller ikke. Eksisterer ikke brukeren vil brukerinformasjonen bli lagt til i brukerlisten til serveren og klienten vil få melding om dette. Eksisterer brukeren vil serveren sjekke at passordet som sendes er riktig. Er passordet riktig vil brukeren få beskjed om at registreringen var vellykket, eller klienten vil få en melding om at registreringen var mislykket.

5.3.6 Innlogging og utlogging

5.3.6.1 Innlogging

For å logge inn og ut av system må man trykke på ”Login/Logout” – knappen på ”File” menyen. Innlogging og utlogging vil skje som beskrevet i avsnitt 4.2.1. Ved innlogging sender klienten en forespørsel til server om at den ønsker å logge inn, og deretter sender den brukernavn, passord og IP – adresse. Serveren verifiserer brukeren og sender oppdatert status til de brukerne klienten forespør status til om disse er pålogget. Foreløpig fungerer prototypen kun innenfor ett nettverk, og klienten vil sende med den lokale IP adressen den er tildelt, og ikke den globale som den gjør i et ferdig system.

Meldinger som blir vekslet mellom server og klient er som følger:

Først sender klient følgende melding til server:

LOGIN	IP

Kommandoen ”LOGIN” sier til server at klienten ønsker å logge seg på serveren og IP er adressen til denne klienten.

Serveren svarer så med følgende melding:

READY

Denne meldingen forteller klienten om at serveren er klar til å motta login informajon, og klienten sender da denne meldingen:

LOGINI	Brukernavn	Passord	IP

Informasjon etter kommandoen er brukernavn, passord og IP – adressen til denne brukeren. Serveren sjekker med sin brukerfil om denne brukeren eksisterer og om passordet er korrekt. Stemmer dette vil serveren oppdatere statusen til denne brukeren med den IP – adressen den logget på med. Når statusen er oppdatert vil serveren sende en melding til klienten om at innlogging var vellykket med følgende melding:

LOGINOK

5.3.6.2 Utlogging

Når en bruker ønsker å logge seg av serveren må brukeren trykke på ”Logout” – knappen på ”File” – menyen. Brukeren sender da en melding til server om at den ønsker å logge seg ut med følgende melding:

LOGOUT	IP

Når servere mottar denne meldingen svarer den med en READY melding. Brukeren svarer da med en logg ut melding til server med følgende format:

LOGOUTI	Brukernavn	IP	Brukerliste

Brukernavn og passord er informasjon til brukeren som logger seg av ,og Brukerliste er alle brukernavn som er lagt til i brukerfilen til brukeren som logger seg av. Når serveren mottar denne meldingen sender den melding til alle brukerne i Brukerlisten som er sendt med om at brukeren er logget av. Denne meldingen er vist i avsnitt 5.3.7.

5.3.7 Oppdatering av status

Når innloggingsprosedyren er gjennomført, må klienten sende en forespørsel om statusen til brukerne som er lagt til i brukerfilen. Måten dette gjøres på er at klienten sender en melding for hver bruker i filen til serveren og forespør statusen med følgende melding:

UPST	Brukernavn	IP

Brukernavn er brukernavnet til den brukeren klienten forespør etter og IP er adressen til klienten. Når serveren for denne meldingen sjekker den med brukerfilen om brukeren eksisterer, og om dette er tilfelle svarer den klienten med status og IP med følgende melding:

UPST	Brukernavn	IP	Status

Informasjonen som blir sendt tilbake er IP adresse og status til denne brukeren. Om denne brukeren er pålogget sender server også en melding til denne brukeren om at den brukeren som forespør er logget på med sammen melding som over.

5.3.8 Legge til/slette brukere

For å legge til en bruker i sin brukerliste må man gå inn i ”Users” tabulator, og deretter skrive inn et brukernavn i tekstfeltet. Når brukernavn er skrevet inn trykker man på knappen ”Add user”, og klienten sender dette brukernavnet til server. Verifiseringen av brukeren skjer i henhold til avsnitt 4.3.4 med unntak av at den som blir lagt til ikke får noen melding om dette. Ved sletting av bruker merker man av brukernavnet i brukerlisten og trykker på ”Delete user” knappen. Server eller den som blir slettet vil ikke få melding om dette i prototypen.

5.3.8.1 Meldingsutveksling

Når en bruker ønsker å legge til et brukernavn i sin brukerliste, må hosten sende en melding til serveren om denne brukeren eksisterer. Klienten sender da meldingen vist under:

ADDU	Brukernavn	IP

Klienten sender kommandoen ”ADDU” med Brukernavn til den brukeren ønsker å legge til og IP – adressen til klienten. Serveren sjekker så om dette brukernavnet eksisterer i brukerfilen sin, og sender en melding tilbake til klienten om brukeren eksisterer eller ikke. Om brukeren eksisterer svarer serveren med følgende melding:

ADDU	T	IP	Status

Det andre feltet er bokstaven T som betyr at brukeren eksisterer, og de to resterende feltene er henholdsvis IP – adressen til brukeren og status.

5.3.8.2 Skrive til fil

Når statusen er mottatt hos klienten må dette skrives til brukerfilen lokalt på maskinen. Et utdrag av kildekoden for hvordan man skriver brukeriformasjon til en fil er vist under. Koden viser kun ”Username” elementet som blir lagt til ”Userinfo” – elementet.. ”IP” – elementet og ”Status” – elementet blir lagt til på samme måte som vist under og er derfor ikke tatt med.

```
XmlDocument xmlDoc = new XmlDocument();
xmlDoc.Load(“Users.xml”);

XmlNode root = xmlDoc.DocumentElement;
XmlElement childNode1 = xmlDoc.CreateElement("Userinfo");
XmlElement childNode2 = xmlDoc.CreateElement("Username");
XmlText textNode = xmlDoc.CreateTextNode(user);

root.AppendChild(childNode1);
childNode1.AppendChild(childNode2);
childNode2.AppendChild(textNode);

xmlDoc.Save(filename);
```

Først må man laste opp filen man ønsker å skrive til ved benytte metoden

xmlDoc.Load(”Users.xml”);

Metoden root.AppendChild(childNode1) legger til undernoden ”Userinfo” til rot – noden, metoden childNode1.AppendChild(childNode2) legger til ”Username” – noden under ”Userinfo” – noden. Metoden childNode2.AppendChild(textNode) legger til nodeteksten til elementet

”Username”, og til slutt skrives metoden xmlDoc.Save(”Users.xml”) for å lagre informasjonen til filen. Når skrivingen til filen er fullført vil xml – filen se ut som på figur 5.3 i avsnitt 5.3.4.

5.3.9 Sikkerhetskopiering

For å ta sikkerhetskopiering av en fil må man først dobbeltklikke på den brukeren man ønsker å utføre sikkerhetskopiering mot. Deretter velger man de filene man ønsker å kopiere og trykker deretter på knappen ”Backup”. Filene vil bli lagret på mottakene host og klienten vil få beskjed om overføringen var vellykket. Sekvensen for sikkerhetskopiering vil foregå som i avsnitt 4.3.2. I prototypen er det kun mulig å utføre sikkerhetskopiering av filer i en mappe. Filene vil bli overført kryptert, men de vil ikke være kryptert når de lagres på den mottaende host.

5.3.9.1 Meldingsutveksling

Meldingsutvekslingen mellom klientene som er involvert i sikkerhetskopieringsprosessen forgår som beskrevet i sekvensdiagrammet i avsnitt 4.3.3.2.

For å sende backup til en host, er det nødvendig å gjøre den som skal motta filene oppmerksom på dette. Dette gjøres ved å sende en melding til hosten med følgene format:

BACKUP	IP

Hosten svarer med en ”READY” - melding og begynner å lytte til innkommende data.

5.3.9.2 Sending av data

Under er et utdrag av metoden for å sende en fil til en peer. Metoden

```
while ((bytesize = filestream.Read(sendbyte, 0, sendbyte.Length)) > 0)
      {
            strSend.Write(sendbyte, 0, bytesize);
            transfer += bytesize;
            int prosent = Convert.ToInt32((transfer * 100) / totallength);
            Form1.progressBackup.Value = prosent;
      }
```

For å sende over en større fil, er det nødvendig å dele opp filen. Dette gjøres ved at man leser av x antall byte fra filen som skal overføres. Hver gang denne while – løkken har kjørt igjennom vil variabelen bytesize, som er et heltall, bli satt til antall byte som blir lest fra filen. Denne while – løkken vil repetere seg selv så lenge det er mer byte å lese.
For å lese av byte fra en fil må man opprette en instans av klassen `Stream` som vi kaller filestream, som benyttes for lage en strøm av bytes fra filen som skal overføres. Metoden

`filestream.Read(sendbyte, 0, sendbyte.Length)` brukes for å lese av et gitt antall byte fra strømmen, hvor sendbyte er en array av byte som det skriver data til. Tallet 0 er start indeks til hvor i bytestrømmen man skal starte og lese fra og sendbyte. Lenght er antall byte av strømmen som skal leses fra strømmen.
For å skrive data til nettverket benyttes `strSend.Write(sendbyte, 0, bytesize),` hvor strSend er en instans av klassen `NetworkStream` som lager en strøm av data for nettverksaksess. Metoden `strSend.Write(sendbyte, 0, bytesize);` benyttes for å skrive data til nettverket, hvor ”sendbyte” er data som skal skrives til nettverket, tallet 0 er start indeks fra hvor i bufferet en skal starte og skrive i fra og bytesize er antall byte som skal skrives til nettverksstrømmen.

5.3.9.3 Mottak av data

Mottak av data skjer på samme måte som ved sending av data. I klassen `NetworkStream` benyttes metoden strRemote.Read(downBuffer, 0, blockSize); som vist i kildekoden under. Her leser den data fra nettverket og lagrer data i downBuffer, som er en array av byte, hvor data fra nettverkstrømmen blir lagret. Tallet 0 er startindeks fra hvor i bufferet enn skal starte og lagre data fra, og blockSize er antall byte av nettverksstrømmen som skal lagres.
`fileStream.Write(downBuffer, 0, byteSize)` er metoden for å skrive data til filen. Argumentene downBuffer er data som skal skrives til filen, tallet 0 er startindeks fra hvor data skal skrives fra downBuffer og byteSize er antall byte av nettverksstrømmen som skal skrives til filen.

```
while (true)
{
      try
      {
            byteSize = strRemote.Read(downBuffer, 0, blockSize);
      }
      catch (Exception exMessage)
      {
            MessageBox.Show(exMessage.Message);
      }

      fileStream.Write(downBuffer, 0, byteSize);

      if (byteSize == 0) break;
}
```

5.3.9.4 Lagring av filer

Alle filer som det tas backup av blir lagret i mappen til den som utfører backup. Stien til denne mappen er ”C:\Mututal Backup\brukernavn\”, hvor brukernavn er katalognavnet til brukeren som opprettes når en bruker legger til en bruker i sin brukerfil.

5.3.10 Gjenoppretting

For å gjenopprette data må man dobbeltklikke på den brukeren man ønsker å gjenopprette filene

fra. Alle filer man har sikkerhetskopiert mot denne brukeren vil bli overført.
Meldingsutvekslingen vil foregå som i avsnitt 4.3.3.3.
Funksjonen for å velge hvilke filer man ønsker å gjenopprette er ikke valgt å implementere i prototypen.

5.3.10.1 Meldingsutveksling

Gjenopprettingsprosessen foregår i henhold til sekvensdiagrammet i avsnitt 4.3.3. Den første meldingen som blir sendt fra ”peer1” er:

RESTORE	Brukernavn	IP

Brukernavn og IP er informasjon om brukeren som forespør gjenoppretting av data. ”peer2” svarer med en ”FILE” – melding som er som følger:

FILE	Brukernavn	IP

Brukernavn er brukernavnet til ”peer1” og IP er adressen til ”peer2”. Når ”peer1” mottar denne meldingen sender ”Host” en ”FILER” – melding til ”Host2” og gjør seg klar til å motta data. Melding er som følger:

FILER	Brukernavn	IP

Her er Brukernavn brukernavnet til ”peer1” og IP er adressen til ”peer1”. Når ”peer2” mottar FILER – meldingen begynner ”peer2” og overføre filer til ”peer1”.

5.3.10.2 Sending/mottak av data

Sending og mottak av data skjer i henhold til beskrivelsen i avsnitt 5.3.8.2 og 5.3.8.3, og vil ikke bli beskrevet ytterligere i dette avsnittet.

5.3.10.3 Lagring av filer

Når data er gjenopprettet blir de lagret lokalt på maskinen. Stien filene blir lagret på bestemmes av den som utfører gjenopprettingen, og skjer på samme måte som beskrevet i avsnitt 5.3.8.4

5.3.11 Avslutt

For å avslutte programmet må man trykke på ”Exit” knappen på ”File” – menyen. Denne funksjonen avslutter de trådene som kjører i applikasjonen for å lytte til innkommende data fra server og andre klienter. Klienten sender en logout melding til server om at den logger seg ut når applikasjonen avsluttes. Denne prosedyren utføres som nevnt i avsnitt 5.2.2.

5.4 Demonstrasjon

I dette avsnittet vil vi gi en demonstrasjon av hva som må gjøres for å utføre sikkerhetskopiering og gjenoppretting av data mellom to brukere. Scenarioet tar for seg brukeren ”Huy” og ”André”, hvor Huy utfører sikkerhetskopiering av en fil mot André, og utfører gjenoppretting av denne filen.

5.4.1 Installering av applikasjonen

Når man starter applikasjonen for første gang får man opp et vindu hvor man må registrere et brukernavn og passord. Dette vinduet er vist i figuren under. Både Huy og André må gjøre dette på hver sin datamaskin. Når dette er skrevet inn trykker de på ”Register” knappen. Brukerinformasjonen blir da sendt til serveren som verifiserer om brukernavnet eksisterer fra før av. Serveren legger til brukerinformasjonen i sin brukerliste, og sender en melding til klienten at registreringen var vellykket.

Figur 5.4: Registreringsvindu

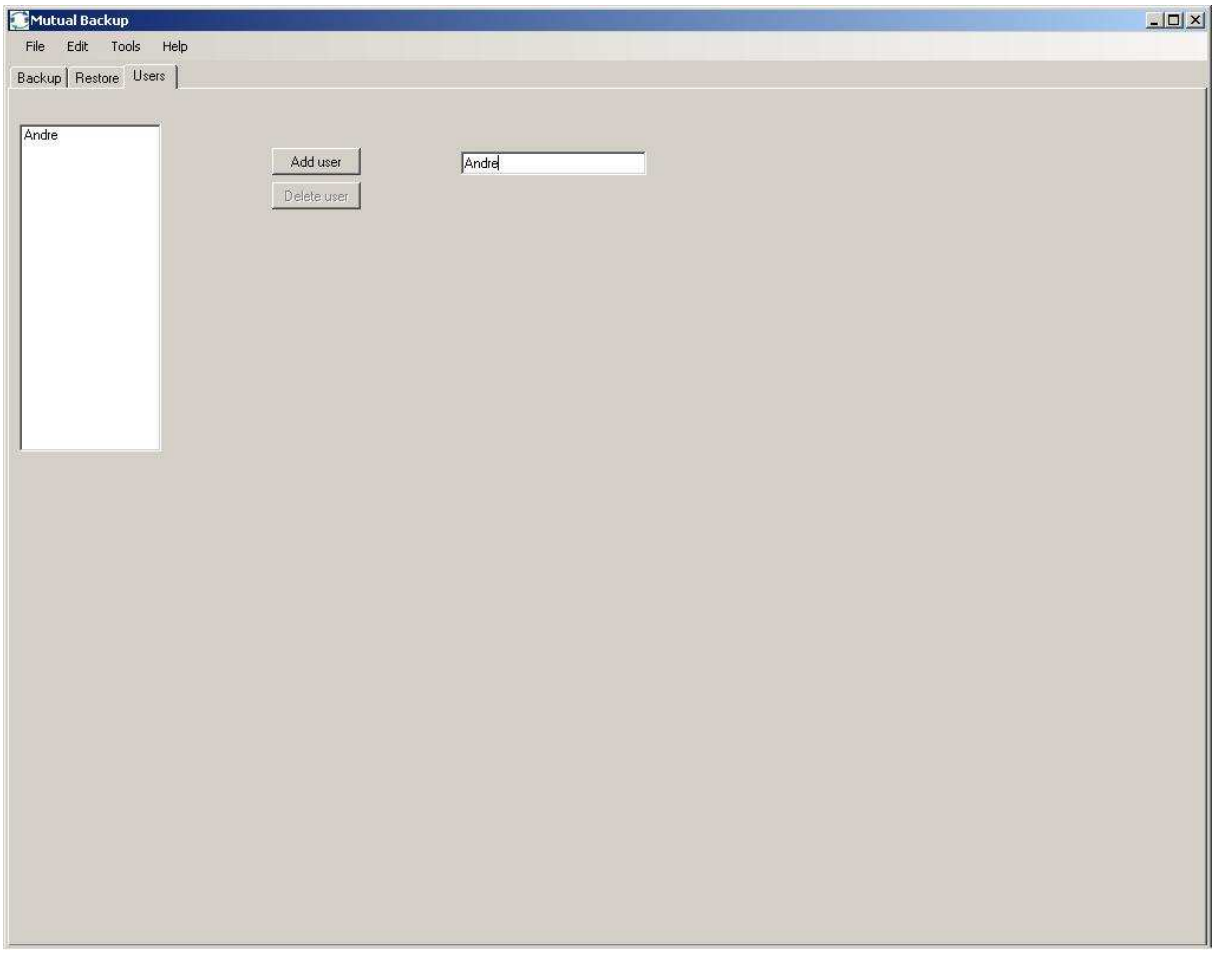
5.4.2 Innlogging

For å logge inn på systemet må Huy og Andre gå på ”File” – menyen og trykke på knappen ”Login”. Når logg inn prosedyren er fullført vil brukerlisten og knapper være aktive i applikasjonen som vist i figur 5.5.

Figur 5.5: File- meny

5.4.3 Legge til bruker

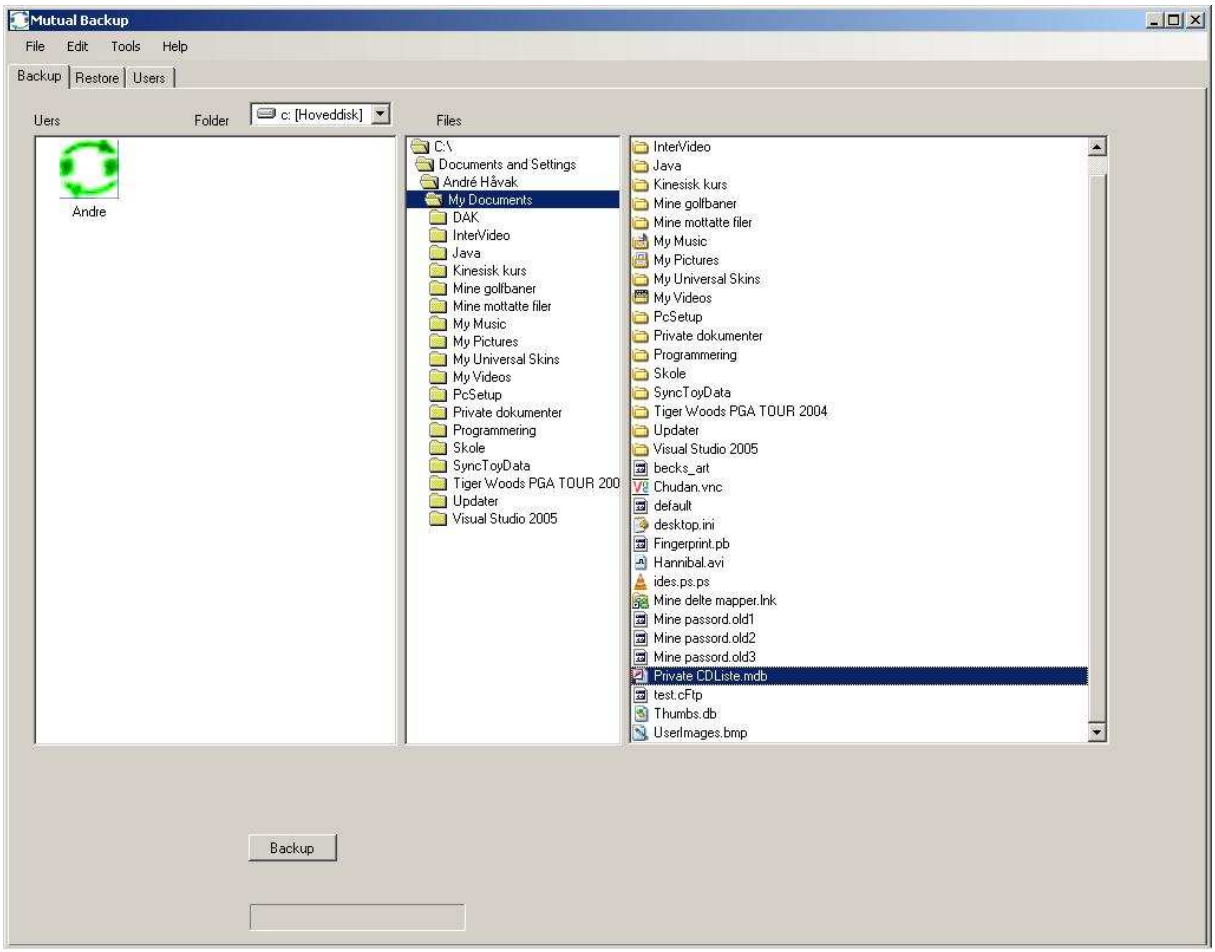
Når innloggingen er fullført må man legge til brukere man ønsker å ha i sin brukerliste. Dette gjøres ved å gå til tabulatoren ”Users”, og i tekstfeltet skriver man inn ønsket brukernavn. I dette tilfelle legger Huy til brukeren Andre og trykker på knappen ”Add user”. Huy sender da en melding med brukernavnet til serveren for å verifisere at brukeren eksisterer. Huy får en melding tilbake fra server om at brukeren eksisterer og brukeren blir lagt til i brukerlisten som vist på venstre side i bildet under.



Figur 5.6: Legge til bruker Andre

5.4.4 Sikkerhetskopiering

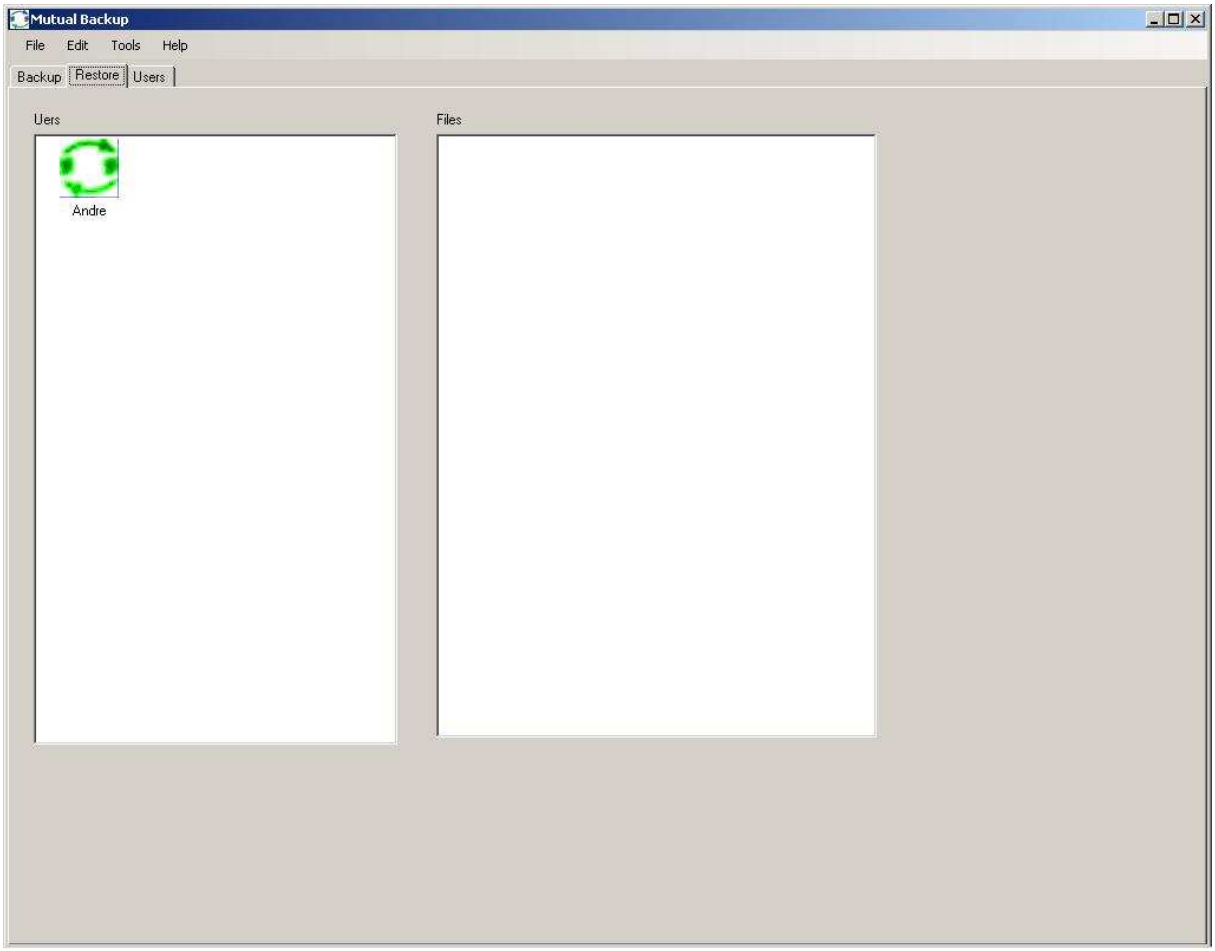
Etter at både Huy og Andre er innlogget er Andre klar til å motta fil fra Huy. For å utføre sikkerhetskopiering må Huy først dobbeltklikke på brukeren Andre for å gi melding om at man ønsker å ta sikkerhetskopiering. Deretter velger Huy filen ”Private CDListe.mdb” og trykker deretter på knappen ”Backup”. Filen blir deretter overført og Huy får opp en melding om at overføringen er vellykket. Filene blir lagret under katalogen ”C:\Mutual Backup\Huy\” på maskinen til Andre.



Figur 5.7: Backup tabulator

5.4.5 Gjenoppretting

For at ”Huy” skal få utført gjenoppretting av data må han dobbeltklikke på brukeren ”Andre”. Alle filer som er sikkerhetskopiert hos ”Andre” blir da automatisk overført og lagret lokalt på maskinen til Huy under valgt katalog.



Figur 5.8: Gjenopprettingsfanen

6 Testing og resultater

6.1 Introduksjon

Prototypen er utviklet i .NET framework som passer meget bra for å utvikle applikasjoner for Windows plattformer. Dette kapitlet beskriver testing av forskjellige moduler, samt resultater fra testingene. Modulene som skal testes ut er brukergrensesnitt, funksjonalitet, nettverk og overføring av filer i forskjellige størrelser.

6.2 Brukertest

Denne brukertesten går ut på å la to brukere teste og evaluere brukervennligheten til prototypen. Testbrukerne består av en erfaren og en uerfaren PC-bruker. De skal registrere seg, legge til hverandre og foreta sikkerhetskopiering som beskrevet i avsnitt 5.4.

6.2.1 Testbruker 1

Denne brukeren er en erfaren PC-bruker og er kjent med Windows baserte applikasjoner.

Resultat:

Spørsmål	Svar
Er GUI oversiktlig?	Ja, GUI er enkelt og består av få knapper og andre komponenter. Det er strukturert og oversiktlig, det er lett å navigere seg fram.
Er det lett å komme i gang med å bruke applikasjonen?	Det er lett å komme i gang. Registrering er på samme måte som de fleste applikasjoner hvor man skriver inn brukernavn og passord. Etter det kan man logge seg inn for å bruke applikasjonen.
Hvordan er det å overføre en fil til den andre brukeren?	Det er bra at applikasjonen sier fra at man må velge en bruker før man kan velge en fil å overføre. Selve overføringen av filen fungerer bra, og filen blir overført til riktig bruker.
Hvordan er det å gjenopprette data?	Man får gjenopprette data som man har overført
Er det noe uklart i applikasjonen?	Nei, ingenting er uklart. Denne applikasjonen er lett å skjønne og lett å bruke.
Hva er bra med applikasjonen?	Den er strukturert og oversiktlig. Enkel å bruke. Den tar ikke mye diskplass.
Er det noe som kan forbedres?	Jeg synes det hadde vært bedre om jeg kunne velge hvilke filer som skal gjenopprettes, og bestemme hvor jeg kan lagre de gjenopprettede filer.

6.2.2 Testbruker 2

Denne brukeren er en uerfaren bruker, som bruker PC-en kun til å surfe på internett, skrive enkle dokumenter og bruke chatte-programmer.

Spørsmål	Svar
Er GUI oversiktlig?	Ja, jeg klarte å navigere meg fram i menyene til de funksjonene som var nødvendig å utføre testen.
Er det lett å komme i gang med å bruke applikasjonen?	Det var enkelt å komme i gang. Fikk beskjed om å registrere et brukernavn og passord, og deretter kom applikasjonen opp.
Hvordan er det å overføre en fil til den andre brukeren?	Når jeg først skjønte rekkefølgen for hva jeg skulle trykke på gikk det bra. Prøvde å velge fil og trykke på knappen ”Backup”, og da fikk jeg opp en melding som sa jeg måtte velge bruker først.
Hvordan er det å gjenopprette data?	Jeg skjønte ikke når overføringen var ferdig, ettersom jeg ikke fikk noen meling om dette. Programmet fortalte ingenting om hvor ting ble lagret, så jeg fant ikke disse dataene med en gang.
Er det noe uklart i applikasjonen?	Skjønte ikke helt det med rekkefølgen når jeg skulle utføre sikkerhetskopiering og gjenoppretting.
Hva er bra med applikasjonen?	Det var ikke mulig å trykke på knapper og slikt før man var logget inn. Dette gjorde at jeg skjønte at jeg måtte logge meg inn før jeg kunne begynne å overføre filer.
Er det noe som kan forbedres?	Syntes man burde kunne velge hvor man lagret filer, så jeg kan lettere finne de etter gjenopprettingen. Brukte litt tid på og finne de tilbake. Kunne vært en innloggingsknapp også i startvinduet.

6.3 Funksjonalitetstesting

I dette avsnittet skal vi teste ut funksjonaliteten bak de forskjellige knappene som finnes i grensesnittet, samt teste forskjellige scenarioer ved bruk av prototypen. Det skal testes ut på innlogging og utlogging, overføring og gjenoppretting av data, og til slutt feilhåndteringer.

6.3.1 Testingsoppsett

Testing av funksjonaliteten foregår mellom to PC-er. Den ene overfører filer til den andre som venter på å ta imot filer. En tredje PC fungerer som server kun for å oppdatere statusene til begge PC-ene. Begge PC-ene er tilknyttet lokalnettverk med en ruter.

6.3.2 Testresultater

Testresultatene i tabellene nedenfor kommer fra forskjellige testscenarioer. På hvert scenario er det en beskrivelse av et ønsket resultat. Hvis resultatet er oppnådd, vil det stå ”JA” i tabellen; hvis resultatet ikke er oppnådd, vil det stå ”NEI” i tabellen samt en begrunnelse for hvorfor resultatet ikke er oppnådd.

Tabell 6.1: Start av applikasjon

Nr.	Testscenario	Ønsket resultat	Resultat oppnådd	Begrunnelse
1-1	Starter applikasjon første gang.	Nytt vindu for registrering for første gang.	JA	
1-2	Starter applikasjon etter registrering	Åpner applikasjonen.	JA	

Tabell 6.2: Registrering

Nr.	Testscenario	Ønsket resultat	Resultat oppnådd	Begrunnelse
2-1	Brukernavn ikke eksisterer	Nytt brukernavn og passord registrert.	JA	
2-2	Brukernavn eksisterer	Melding om brukernavn allerede eksisterer.	JA	
2-3	Bekrefte passord: riktig	Registrering fullført.	JA	
2-4	Bekrefte passord: feil	Melding om passord ikke er like.	JA	

Tabell 6.3: Avslutt applikasjon

Nr.	Testscenario	Ønsket resultat	Resultat oppnådd	Begrunnelse
3-1	Avslutte uten å logge av	Brukeren blir automatisk logget av og applikasjon avsluttet. Server og andre brukere online mottar nye status.	JA	
3-2	Logger av før avslutning	Applikasjon avsluttes.	JA	

Tabell 6.4: Innlogging

Nr.	Testscenario	Ønsket resultat	Resultat oppnådd	Begrunnelse
4-1	Innlogging uten forbindelse	Melding om innlogging mislykkes.	JA	
4-2	Innlogging med forbindelse	Innlogging vellykket.	JA	
4-3	Innlogging med samme brukernavn og passord	Et brukernavn kan være innlogget på en host av gangen.	JA	

Tabell 6.5: Utlogging

Nr.	Testscenario	Ønsket resultat	Resultat oppnådd	Begrunnelse
5-1	Logger ut	Oppdaterer status til server og brukere. Brukerlisten blir inaktiv.	JA	

Tabell 6.6: Legg til bruker

Nr.	Testscenario	Ønsket resultat	Resultat oppnådd	Begrunnelse
6-1	Legge til en registrert bruker	Brukeren blir lagt til på brukerlisten.	JA	
6-2	Legge til en ikke-eksisterende bruker	Melding om at brukernavnet ikke eksisterer. Ingen bruker blir lagt til.	JA	
6-3	Legge til en bruker som allerede eksisterer på brukerlisten	Melding om at brukernavnet allerede eksisterer. Ingen bruker blir lagt til.	JA	

Tabell 6.7: Slett bruker

Nr.	Testscenario	Ønsket resultat	Resultat oppnådd	Begrunnelse
7-1	Velger et brukernavn for å slette brukeren fra listen	Brukernavn og informasjon, samt mapper og filer blir slettet.	JA	
7-2	Sletter en bruker uten å velge brukernavnet	Melding om å velge et brukernavn for å slette brukeren fra listen.	JA	

Tabell 6.8: Nettverksfeil

Nr.	Testscenario	Ønsket resultat	Resultat oppnådd	Begrunnelse
8-1	Mister forbindelse mens brukernavn er pålogget	Applikasjon vises som frakoblet. Server og andre brukere får oppdatert status.	NEI	Server sender ingen ping melding til brukere for å undersøke om de kan nås.

Tabell 6.9: Sikkerhetskopiering

Nr.	Testscenario	Ønsket resultat	Resultat oppnådd	Begrunnelse
9-1	Kobler til en frakoblet bruker	Kan ikke opprette forbindelse med brukeren. Får melding om at brukeren er frakoblet.	JA	
9-2	Kobler til en tilkoblet bruker	Tilkobling vellykket.	JA	
9-3	Velger overføring av fil før valg av bruker	Melding om å velge en bruker først.	JA	
9-4	Overføring fil til en valgt bruker	Fil blir overført til valgt bruker. Fil blir lagret i riktig mappestruktur.	JA	
9-5	Mister forbindelse under overføring av fil	Melding om forbindelsen er brutt.	JA	
9-6	Slette filer som blir avbrutt	Filer som blir avbrutt underoverføring blir automatisk slett.	JA	

Tabell 6.10: Gjenoppretting

Nr.	Testscenario	Ønsket resultat	Resultat oppnådd	Begrunnelse
10-1	Kobler til en frakoblet bruker	Samme som i pkt 7-1	JA	
10-2	Kobler til en tilkoblet bruker	Alle filer lagret i mappen hos brukeren blir gjenopprettet.	JA	
10-3	Mister forbindelse under gjenoppretting av fil	Samme som i pkt 7-5	JA	

6.4 Testing av filer i forskjellige størrelser

I denne testdelen skal vi teste overføring av forskjellige filtyper og filstørrelser for å undersøke om det kan være en sammenheng med overføringstiden og nettverkshastigheten.

6.4.1 Testingsoppsett

Det vil først bli overført en filtype i liten størrelse (for eksempel en .jpg fil på noen få kilobyte) fra en PC til en annen, og vi vil beregne tiden for å undersøke hvor lang tid det tar å overføre denne filen. Deretter overfører vi en annen filtype som er litt større (eks. et dokument på noen megabyte) og tar tiden. Slik skal denne testdelen fortsette til vi finner ut når vi har nådd maksimumsgrensen for filstørrelsen.

6.4.2 Testresultater

Tabellene nedenfor viser testresultater for sikkerhetskopiering og gjenoppretting av filer. I tabellene er det beskrevet hvilke filtype og størrelse som overføres, hvor lang tid det bruker, om filen blir vellykket overført og til slutt en måling av overføringshastigheten.

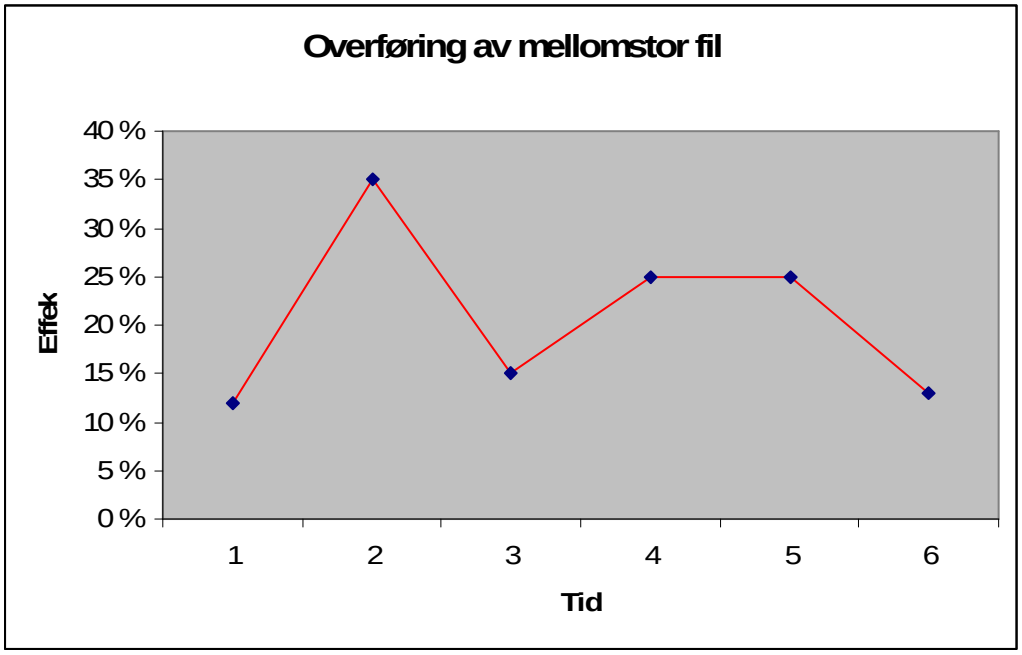
Tabell 6.11: Tid for sikkerhetskopiering

Nr.	Filtype	Størrelse	Tid	Resultat	Overføringshastighet
1-1	Bilde: jpg	134 KB	0,1 sek	Fil overført	-
1-2	Dokument: PDF	8 089 KB	3 sek	Fil overført	21 570 kb/ sek
1-3	Videoklipp: AVI	683 420 KB	265 sek	Fil overført	20 632 kb/sek

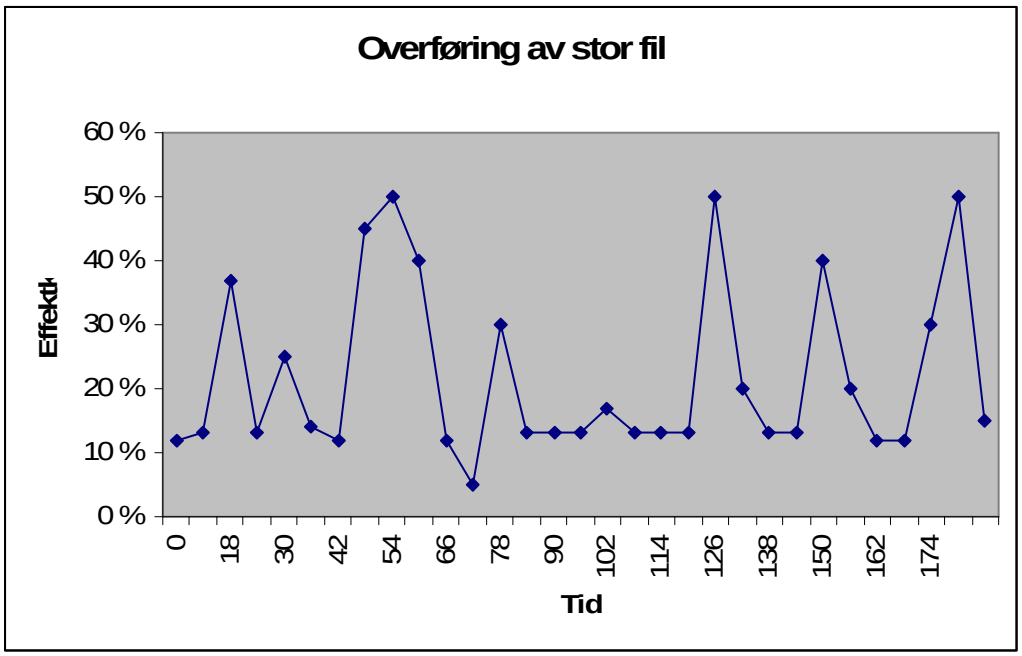
Tabell 6.12: Tid for gjenoppretting

Nr.	Filtype	Størrelse	Tid	Resultat	Overføringshastighet
2-1	Bilde: jpg	134 KB	0,1 sek	Fil overført	-
2-2	Dokument: PDF	8 089 KB	3 sek	Fil overført	21 570 kb/ sek
2-3	Videoklipp: AVI	683 420 KB	180 sek	Fil overført	30 374 kb/sek

Under vises to grafer som illustrerer effektiviteten av overføringen av en stor og en mellomstor fil i et lokalt nettverk. Maksimal overføringshastighet er 100 mbps, og tallene vises i prosent.



Figur 6.1: Overføring av mellomstor fil



Figur 6.2: Overføring av stor fil

7 Diskusjon

7.1 Introduksjon

I dette kapitlet skal vi analysere og diskutere resultatene og begrensninger vi har erfart ved å utvikle dette systemet. Hovedpunktene vi skal diskutere er som følger:

- Forskningsarbeid
- Systemdesign
- Prototyp
- Testing
- Andre erfaringer

7.2 Forskningsarbeid

Litteraturstudiet som ble utført i starten av prosjektperioden var helt nødvendig for oss slik at vi fikk kunnskap som gjorde oss i stand til å utvikle vårt eget backupsystem. Gjennom litteraturstudie har vi sett på forskjellige backupmetoder og forskjellige eksisterende backupløsninger og viktigheten av disse.

Det var tre backupmetoder som vi har vurdert. Disse metodene er full backup, inkrementell backup og differensial backup. Full backup er nyttig til å ta kopi av all data første gangen. Denne metoden er tidkrevende ved sikkerhetskopiering av store mengder data, og i et online backupsystem vil dette ofte ta for lang tid. Ved jevnlig sikkerhetskopiering av data vil denne metoden ikke være egnet, ettersom denne metoden overskriver all data uansett om de er forandret siden forrige backup eller ikke.

Differensial backup er en mer effektiv metode enn full backup, men har sin begrensning i at den tar sikkerhetskopiering av det som er endret siden forrige fulle backup. Med vårt backupsystem kan man ikke velge backup metode og med differensial backup vil man derfor ta sikkerhetskopiering av all data siden man begynte å overvåke en mappe. Etter hvert som mange filer blir endret og lagt til blir det store mengder data å kopiere, og systemet vil til slutt bli inneffektivt.

Inkrementell backup er den mest effektive backup metoden av de tre som er evaluert. Denne metoden tar sikkerhetskopiering av kun data som er endret siden forrige backup metode. Dette gjør at man reduserer tiden det tar å kopiere over filer noe som er vesentlig i et online backupsystem. Utfører man sikkerhetskopiering jevnlig vil det være små mengder data som overføres, og man kan redusere båndbredden slik at ikke systemet tar for mye ressurser.

Autentisering er viktig når man skal ha tilgang til ressurser over internett. På internett er det enkelt å forfalske en identitet, og gode autentiseringsmetoder er derfor viktig. Med de metodene vi har forsket på er det kombinasjonen av brukernavn og passord som er den beste løsningen for vårt system. Dette gir en tilstrekkelig sikkerhet, og har sine fordeler med at den er enkel å ta i bruk og tilgjengelig for alle. Andre løsninger som for eksempel biometriske løsninger og smartkortautentisering krever ytterligere hardware og softwareinstallasjon, og egner seg best i større systemer som hos bedrifter.

Internett er et usikret nettverk, og det er derfor nødvendig å kryptere informasjon slik at ikke andre enn eieren av data kan lese disse. Når data lagres hos en annen peer, er det viktig at data forblir kryptert. Dette er nødvendig ettersom det ikke alltid er ønskelig at andre skal se hva du tar sikkerhetskopi av.

7.3 Systemdesign

Dette designkapitlet er basert på litteraturstudie og eksperimentering med andre backupløsninger. Litteraturstudie er nødvendig for å kunne designe et backupsystem, da det er viktig å forstå hvordan systemene er bygd opp og hvilke funksjonalitet som er viktig å inkludere.

Systemdesignet representerer en måte å utvikle et online backupsystem på, og er en detaljert systembeskrivelse. Designet kan utvides til å benytte til for eksempel fildeling og synkronisering av filer. Disse bruksområdene er nyttige om to brukere arbeider med samme fil.

Vi har valgt å utvikle applikasjonen i .NET versjon 2.0. I .NET 2.0 er ikke alle komponentene i de tidligere versjonene tatt med, noe som gjør at applikasjonen ikke er bakover kompatibel med tidligere versjoner. .NET 2.0 er inkludert i versjon 3.0 og kan derfor benyttes i blant annet Microsoft Windows Vista.

7.4 Utvikling av prototypen

Etter at vi fullførte kravspesifikasjonen og systemdesignet startet arbeidet med implementeringen av prototypen. Dette gjorde at vi fikk god oversikt over hvilke funksjoner vi skal implementere i prototypen. Denne prototypen er ment for å vise prinsippene i dette systemet. I kapittel 5 er det beskrevet og vist hvordan man utfører sikkerhetskopiering av data mellom to peers.

Dette kapitlet er ment å vise hvordan man kan implementere funksjoner som er nødvendig for å utføre filoverføring. .NET framework 2.0 har støtte for mange funksjoner, og var til stor hjelp i utviklingen av prototypen. Dette innebar at det var mange fagbøker og eksempler som kunne benyttes.

Underveis i implementeringen har vi måttet ta hensyn til stabilitet i applikasjonen, slik at applikasjonen ikke henger ved brukerfeil. Et eksempel kan være at en bruker prøver å overføre en fil før man har valgt en bruker å overføre mot. Vi har kontinuerlig testet hver funksjon som er implementert, og vi mener at dette har hjulpet oss til å få en stabil og brukervennlig prototyp. Dette bekreftes i brukertesten utført i kapittel 6.

Et problem som oppstod underveis var hvordan data skulle bli sendt til en host som er plassert bak en ruter. Måten vi implementerte prototypen på, gjorde at vi kunne sende data fra klient til server, men man ikke sende data fra server til klient uten å konfigurere ruteren. Dette er en stor begrensning i forhold til de som er plassert bak en ruter. De må selv konfigurere ruteren til å videresende pakker til den hosten bak ruteren som skal motta pakken, og det er kun en peer som kan motta pakker fra serveren.
Ut i fra de positive tilbakemeldingene vi har fått i fra testbrukerne og veilederen vår, kan vi si at implementeringen av et brukervennlig backupsystem er vellykket. Dette er til stor nytte for det videre arbeidet av prototypen, slik at den kan tilfredsstille brukerne.

Prototypen slik den er i dag fungerer som et fildelingssystem uten kryptering av filer. Brukerne kan sende data over til hverandre, lese og slette hverandres data. For å gjøre dette om til et komplett backupsystem er det nødvendig å implementere funksjonene diskutert i avsnitt 7.7.

7.5 Testing

Testene vi har utført gir en indikasjon på funksjonaliteten og ytelsen i applikasjonen. Testen viser at funksjonene virker i flere situasjoner, men testen sier ingenting om stabiliteten. Siden dette er en prototyp er ikke alle feilhåndteringer rettet opp, noe som gjør at applikasjonen kan virke noe ustabil. Hvis denne applikasjonen skal videreutvikles, vil ikke-funksjonelle krav ha større betydning for implementeringen.

Tester viser at man ikke får utnyttet båndbredden maksimalt i lokale nettverk hvor hastigheten er opp mot 100 Mbps. Dette er noe svakt i forhold til hva vi forventet i forkant av testene, men årsaken til dette vil bli rettet opp i det fremtidige arbeidet.

For å teste hvordan hastigheten er ved overføring av flere typer filer, har vi testet overføring av små og store filer. Hastigheten er den samme for små og store filer.

Tester som er utført av prototypen sier ingenting om systemet virker utenfor et lokalt nettverk, da testoppsettet fokuserte på testing i LAN. Overføringshastigheter over internett er derfor ikke en del av testingen.

Det er utført test av brukervennligheten i vår applikasjon. Dette har vi gjort med å teste applikasjonen med en erfaren databruker og en uerfaren bruker. Testene viser at applikasjonen var enkel å ta i bruk, noe som er i forhold til de intensjonene vi hadde før implementering av prototypen.

Noen av resultatene i testen var som forventet, men vi kan se at det gjenstår noe arbeid for å gjøre systemet mer effektivt. De funksjonene som er implementert virker i henhold til kravspesifikasjonen og systemdesignet, og dette forteller oss at det vi har implementert er i riktig retning mot et ferdig produkt. Med dette konkluderer vi med at vi har oppnådd målet med testingen.

7.6 Andre erfaringer

7.6.1 Waterfall model

Vi så på de forskjellige utviklingsmodellene, og valgte å følge Waterfallmodellen ettersom denne modellen var tilpasset vår metode å arbeide på. Denne modellen har vært nyttig for oss i arbeidet med å utvikle prototypen. Vi fant ut at det ikke var enkelt å følge denne modellen fullstendig, da det dukket stadig opp endringer som måtte gjøres i tidligere faser. Under selve implementeringen dukket det stadig opp nye ideer og andre løsninger; noe som førte til at vi måtte gå tilbake for å gjøre endringer kravspesifikasjonen og systemdesignet. Under testing fant vi ut at vi måtte gå tilbake å endre punkter i systemdesignet som for eksempel klassediagrammene.

Med Waterfall som arbeidsmodell har det hjulpet oss med å planlegge, strukturere arbeidet og samtidig ha en god oversikt over arbeidsprosessen. En slik modell mener vi er vanskelig å følge fullstendig, ettersom vi fikk erfare at det vil alltid bli endringer under store deler av utviklingsprosessen. Om vi skulle startet en slik utviklingsprosess på nytt, ville vi ha valgt å følge Agile utviklingsmodell. Denne er mer effektiv da vi kan gi hyppigere leveringer til veileder, og vi kan da få tilbakemeldinger underveis. Denne gir også muligheten til å gjøre endringer i kravene ved senere tidspunkt, noe som var tilfelle i vår utvikling

7.6.2 Utviklingsverktøy

Utviklingsverktøyet vi benyttet i dette prosjektet var Visual Studio 2005. Dette verktøyet gir mulighet til valg av flere programmeringsspråk, som gjør at det når flere brukere. Design av GUI var til stor hjelp for oss i både designdokumentet og i prototypdesignet, da dette kan gjøres grafisk. Kildekoden for GUI og metodene til komponentene blir automatisk generert, noe som gjør arbeidet enklere og raskere. Den største ulempen med Visual Studio er at man må vite nøyaktig klassenavn om man ikke har inkludert konteksten til denne klassen fra før av. Dette gjør at man ikke får opp alternativer til klassenavn man kan benytte.

7.7 Fremtidig arbeid

Det gjenstår en del arbeid med systemet før det kan bli til et ferdig produkt. Prototypen er designet for å vise prinsippet for sikkerhetskopiering over internett. I fremtidig arbeid vil det være nødvendig å implementere funksjoner som kryptering av brukerinformasjon og filer, og implementering av inkrementell backup. Kryptering er viktig for å beholde integriteten og begrense tilgjengeligheten til data. Inkrementell backup er viktig for å gjøre overføringen effektiv, slik at man sparer ressurser ved overføring.
I tillegg må vi også implementere funksjoner for å overføre mapper og undermapper, slik at overføring av mange filer og mapper blir enklere.
For at systemet skal bli mer brukervennlig, er det nødvendig å implementere funksjoner for å håndtere sending av pakker til maskiner som ligger bak en ruter. Dette er nødvendig for at brukerne av systemet ikke skal måtte konfigurere ruteren til å videresende pakker til riktig peer bak ruteren.

8 Konklusjon

I denne masteroppgaven har vi evaluert de forskjellige backupmetoder og backupløsninger, og sett på deres fordeler og ulemper. I denne oppgaven har vi presentert en måte å utføre sikkerhetskopiering basert på P2P teknologi. En slik løsning er et forholdsvis nytt forskningsområde, og gir nye muligheter for sikkerhetskopiering av filer. Et slikt system er egnet for personer som har små mengder data man ønsker å ta kopier av. I forhold til andre backupløsninger er dette en rimelig måte å utføre sikkerhetskopiering på, da man ikke har behov for å kjøpe dyr hardware eller leie diskplass hos en online backup tilbyder.

Et slikt backupsystem kan utvides til å brukes til fildeling, hvor eieren av data gir rettigheter til filer og mapper man har tatt sikkerhetskopiering av. Man kan også bruke systemet til å synkronisere data mellom to brukere slik at man kan arbeide med de samme filene. Slike bruksområder kan være nyttig i for eksempel prosjektarbeid i skole og jobbsammenheng. Med en slik løsning som vi presenterer her vil sikkerhetskopien være tilgjengelig for brukeren hvor som helst, forutsatt at den som oppbevarer kopien er pålogget.

I denne oppgaven har vi utviklet en prototyp som demonstrerer hvordan vi kan sikkerhetskopiere og gjenopprette data mellom to PC-er via internett. Prototypen er utviklet i .NET som resulterer i at vi får en Windows-basert applikasjon. Ved å utvikle en Windows-basert applikasjon vil det kunne bli tatt i bruk av flere brukere siden de fleste PC-brukere har Microsoft Windows installert.

.NET 2.0 inneholder all API som er nødvendig for å kunne utvikle et slikt system. Dokumentasjonen og veiledning for rammeverket var lett tilgjengelig på internett, og var enkelt å finne. Dette var til stor hjelp for oss da dette var en teknologi vi tidligere ikke har benyttet.
En ulempe med dette rammeverket er at den ikke tillater to samtidige tråder som lytter til samme nettverksport. Dette fant vi ingen løsning på, og det var derfor nødvendig å benytte flere porter.

Ettersom overføringen av data foregår over internett egner dette systemet seg til å ta kopi av små mengder data. Dette begrunnes med at internettforbindelsen i dag setter en begrensning for overføringshastigheten (typisk 4Mbps for private hjem), og overføring av store mengder data vil ta for lang tid til at det vil være hensiktsmessig å benytte seg av denne løsningen.

Denne masteroppgaven har vært til stor nytte for oss da vi har lært teorier og viktigheten rundt backup, og hvordan man kan utvikle et slikt system fra design til implementering. De studiene som er gjort mener vi også kan bidra til å gi leserne ideer om hvordan man kan utvikle lignende systemer.

Referanser

Internett:

[1] Wikipedia, *Backup*
http://en.wikipedia.org/wiki/Backup
(10.01.07)

[2] *How to decide what data to backup*
http://www.microsoft.com/athome/security/update/backup.mspx
(10.01.07)

[3] *Backup basics*
http://www.informit.com/articles/article.asp?p=478529&rl=1
(13.01.07)

[4] Amarillo Datasafe, *Backup rotation methods*
http://www.amarillodatasafe.com/Backup_rotation.htm
(15.01.07)

[5] Wikipedia, *RAID*
http://en.wikipedia.org/wiki/**RAID**
(23.01.07)

[6] Andreas Grimsby, *RAID*
http://www.hardware.no/guider/harddisk/raid/32088
(29.01.07)

[7] Wikipedia, *CAS*
http://en.wikipedia.org/wiki/Content-addressable_storage
(25.01.07)

[8] Stephen J. Bigelow, *Tech Close up: Content-addressed storage*
http://searchstorage.techtarget.com/originalContent/0,289142,sid5_gci1158705,00.html
(27.01.07)

[9] Wikipedia, *SAN*
http://en.wikipedia.org/wiki/Storage_Area_Network
(02.02.07)

[10] Laura Lee Brennan and Marc Olanie, *Storage Area Network*
http://searchstorage.techtarget.com/sDefinition/0,290660,sid5_gci212937,00.html
(05.02.07)

[11] Frank Aune, *SAN*
http://itpro.no/art/1455.html
(16.02.07)

[12] Wikipedia, *NAS*
http://en.wikipedia.org/wiki/Network-attached_storage
(05.02.07)

[13] Whatis.com, *NAS*
http://searchstorage.techtarget.com/sDefinition/0,,sid5_gci214410,00.html
(05.02.07)

[14] Wikipedia, NAS
http://no.wikipedia.org/wiki/Network-attached_storage
(06.02.07)

[15] Randy Kerns, *NAS in the small to midsized business: Selecting a NAS system*
http://searchstorage.techtarget.com/tip/0,289483,sid5_gci1214425,00.html
(06.02.07)

[16] Randy Kerns, *NAS security*
http://searchstorage.techtarget.com/expert/KnowledgebaseAnswer/0,289625,sid5_gci1233552,00.html
(07.02.07)

[17] DAS
http://www.webopedia.com/TERM/D/direct_attached_storage.html
(19.02.07)

[18] DAS
http://searchstorage.techtarget.com/sDefinition/0,290660,sid5_gci1108786,00.html
(19.02.07)

[19] DAS
http://www.pc-pitstop.com/das/
(20.02.07)

[20] DAS
http://www.onestopclick.com/Directory/Storage---Backup-DAS/29-192-201-201.html
(21.02.07)

[21] *What is an online backup?*
http://free-backup.info/what-is-an-online-backup.html
(01.02.07)

[22] James A. Martin, *Mobile Computing: Online backup services*
http://www.pcworld.com/article/id,118454-page,1/article.html

(01.02.07)

[23] Nordic Backup
www.nordic-backup.com
(16.01.07)

[24] Ementor, *Autentisering*
http://www.ementor.no/templates/Page.aspx?id=9198
(20.02.07)

[25] Wikipedia, *Kryptografi*
http://no.wikipedia.org/wiki/Kryptografi
(20.02.07)

[26] SearchVoIP.com, *IP*
http://searchvoip.techtarget.com/sDefinition/0,,sid66_gci214031,00.html
(20.01.07)

[27] Wikipedia, *Firewall (networking)*
http://en.wikipedia.org/wiki/Firewall_(networking)
(21.02.07)

[28] Controlchaos, *What is Scrum?*
http://www.controlchaos.com/
(19.05.07)

[29] Object mentor, *What is Agile?*
http://www.objectmentor.com/omSolutions/agile_what.html
(19.05.07)

[30] XP, *What is Extreme programming?*
http://www.extremeprogramming.org/what.html
(19.05.07)

[31] Select Business solutions, *What is the waterfall model?*
http://www.selectbs.com/glossary/what-is-the-waterfall-model.htm
(19.05.07)

[32] UML
http://atlas.kennesaw.edu/~dbraun/csis4650/A&D/UML_tutorial/what_is_uml.htm
(16.05.07)

[33] Alex Homer, *What's new in .NET framework 2.0*
http://dnjonline.com/article.aspx?ID=dotNET2_intro
(05.03.07)

[34] Devicesacpe, *Glossary*
http://www.devicescape.com/docs/conductor/UserGuide/Glossary.php
(06.03.07)

[35] Thomson course technology, *Operating systems*
http://www.course.com/careers/glossary/operating.cfm#windowsxp
(06.03.07)

[36] MSDN, *Visual Studio 2005*
http://msdn2.microsoft.com/en-us/vstudio/default.aspx
(07.03.07)

[37] Jesse Liberty, *What is C#*
http://www.ondotnet.com/pub/a/dotnet/2005/10/03/what-is-csharp.html
(07.03.07)

[38] Orionsec, *Glossary of Security terminology*
http://www.orionsec.com/Security_Glossary.html
(23.01.07)

[39] VPN
http://www.cisco.com/univercd/cc/td/doc/product/software/ios123/123newft/123t/123t_7/gtaimvpn.htm
(19.02.07)

[40] Wikipedia, *IPsec*
http://en.wikipedia.org/wiki/IPsec
(19.01.07)

Fagbøker:

[41] Mark Stamp, 2006, *Information Security – Principles and practice*, John Wiley & Sons Ltd

[42] Fiach Reid, 2004, *Network programming in .NET: C# and Visual Basic.NET*, Elsevier Science & Technology

[43] W.Curtis Preston, 2007, *Backup and recovery*, O'Reilly Media, Inc, USA

[44] D.B. Little, 2003, *Implementing backup and recovery: The readiness guide for the enterprise*, John Wiley & Sons Ltd

Artikler:

[45] Christophe Bertrand, 2005, *Backup Technology: Overview and Perspectives*

[46] Christos A. Polyzois, *Evaluation of Remote Backup Algorithms for Transaction-Processing Systems*

[47] Four Leaf Fokus: *Nettverksbasert storage*

[48] Landon P. Cox, Christopher D. Murray og Brian D. Noble, *Making backup cheap and easy*

Vedlegg

Denne delen inneholder følgende vedlegg:

- Ordliste
- Backupmedia
- Kravspesifikasjon
- Tidsplan

Vedlegg A: Ordliste

.NET	Rammeverk brukes til å utvikle programvare.
2Peer	Kun to hoster som kommuniserer med hverandre.
AES	**Advance Encryption Standard** – krypteringsalgoritme bestående av blokk cipher.
Backup	Lagrer kopi(er) av viktige data i et annet sted slik at det blir mulig å gjenopprette ved senere tidspunkt.
Brannmur	En enhet som brukes for å kontrollere trafikken mellom internett og lokal nett.
Brute-force	En metode for å knekke kryptografi. Typisk er dette ved å teste ut alle mulige kombinasjoner for å finne den hemmelige nøkkelen i krypteringsalgoritmen.
CAS	**Content adressable storage** – metode for å lagre informasjon.
CD/DVD	**Compact Disc** og **Digital Video Disc** – er optisk disklagring. Brukes til å lagre datamengder.
CIFS	**(common Internet file system) -** En samling av protokoller som brukes i Windows- og UNIX-miljøer for deling av fil- og utskriftstjenester. Kalles også SMB-protokoll.
CIFS, NFS, NCP, MAC, HTTP og FTP	Protokoller som gjør at maskiner kan sende informasjon til hverandre i et nettverk.
Cipher	En melding som er blitt uleselig etter kryptering.
Clustring	Evnen til å gruppere flere NAS-systemer slik at de vises som én logisk NAS-filserver for sluttbrukere. En server i et cluster kalles en node, dvs. at fire Storage-servere utgjør et firenoders cluster.
C#	Et objekt-orientert programmeringsspråk utviklet av Microsoft som en del av .NET
DAS (Direct Attached Storage)	Utplassering av dedikerte lagringsenheter for hver server. Blant ulempene er ineffektiv bruk og tildeling av lagringsplass, og at det er grensesnitt for lagring og administrasjon fra flere leverandører.
DES	**Data Encryption Standard –** en metode for

	kryptering.
DHCP	**Dynamic Host Configuration Protocol** – automatisk tildeling av IP adresser til hoster enten via nettverksadapter eller ruter.
FAR	**False accept rate** – uautorisert person blir feilaktig akseptert.
Fiberkanal	Topologien og transportprotokollen som brukes for å sende datainformasjon på blokknivå mellom server og lagring.
FRR	**False rejection rate** – autorisert person blir feilaktig nektet adgang.
GUI	**Grapical User Interface** – Grafisk grensesnitt for personer til å utføre arbeid på PC.
Hardware	Engelsk ord for maskinvare. Eks. på maskinvarer er PC, tastatur, mus.
Host	En datamaskin som er koblet til internett.
ID	**Identification**
IEEE 802	**Institute of Electrical and Electronics Engineers** – standard som håndterer LAN.
Internett	Et felles offentlig nettverk hvor alle kan overføre data ved hjelp pakkesvitsjet nettverk ved å bruke internett protokoll (IP).
I/O	**Input/Output** – en samling av enheter for å motta og sende ut signaler til en enhet.
IP	**Internet Protocol** – brukes for datakommunikasjon gjennom pakkesvitsjet nettverk.
IPSec	**IP security** – protokoll for å autentisere og kryptere hver IP datapakke.
ISP	**Internet Service Provider** – er et firma eller en organisasjon som tilbyr internett tilgang til forbrukere.
LAN (Local Area Network)	Et lokalt nettverk innen en bygning. Vanligvis basert på Ethernet.
Microsoft Windows	Et operativsystem utviklet av Microsoft. Det finnes mange versjoner av Windows: Windows 95, 98, 2000 og XP. I løpet av 2007 kommer Vista.
NAT	**Network Address Translation** – oversetter avsenders og destinasjons adresser til IP pakker når de går gjennom en ruter eller en brannmur.
P2P	**Peer-to-peer** – et nettverk hvor alle hoster i et nettverk er likeverdige.
PC	**Personal computer** – personlig datamaskin.

PIN	**Personal Identification Number** – Personlig kode for autentisering.
Prototyp	En grunnleggende standard av et originalt fullstendig produkt.
Proxy	Tilbyr nettverksservice til å tillate klienter å foreta indirekte nettverkskoblinger mot andre nettverksservicer.
RAID	RAID (Redundant Array of Inexpensive Disks) er evnen til å gruppere disker som om de var én fysisk enhet, og også gi redundans innen diskgruppen, dvs. speiling av data.
RSA	En krypteringsalgoritme som bruker offentlig nøkkel til å kryptere melding, og privat nøkkel for å dekryptere. RSA er utviklet av Ron Rivest, Adi Shamir og Len Adleman. Bokstavene RSA kommer fra disse tre navnene.
RTS	Reset the connection – brukes for terminere en forbindelse.
Ruter	En nettverksenhet som videresender datapakker gjennom nettverk til destinasjoner via en prosess som kalles ruting (routing).
SAN (Storage Area Network)	Et høyhastighetsnettverk som har som formål å koble sammen ulike lagringsenheter med tilknyttede servere. Nettverket kan gi sikkerhetskopiering og arkivering for flere steder, også fjerntilkoblede steder.
SAS	**Serial attached SCSI** – seriekommunikasjonsprotokoll for DAS enheter.
SATA	**Serial ATA** – teknologi designet for å overføre data til og fra DAS enheter.
SIM	**Subscriber Identity Module** – et smartkort fort mobiltelefoner.
Snapshot	Evnen til å kunne duplisere data innen en server, NAS-enhet eller RAID-disksystem og lagre datakopien med minst mulig bruk av diskplass.
SNIA	**Storage networking Industry Association** – en organisasjon for datalagring i nettverk.
TCP	**Transfer Control Protocol** – Protokoll som garanterer levering av datapakker over internett.
TTL	**Time to live** – et felt i IP headeren som bestemmer levetiden til en datapakke.
UDP	**User datagram protocol** – brukes for å

	sende korte meldinger.
UML	**Unified modeling language** – modelleringsspråk for å lage en abstrakt modell av et system.
USB	**Universal Serial Bus** – en serial bus standard for grensesnitt enheter, dvs. overføring mellom datamaskiner. Var designet for datamaskiner, men ble etter hvert populært. I dag finnes det USB på mobiltelefoner, MP3 spillere, digitale kameraer, osv. som brukes for å overføre filer til datamaskiner.
VPN	**Virtual private network** – Brukes for å kommunisere sikkert over internett.
WAN (Wide Area Network)	Et nettverk som kan omfatte flere bygninger over store avstander - til og med i flere land.
XML	**Extensible Markup Language** – markup språk som brukes til å dele data over forskjellige systemer.
XP	**Extreme programming** – software utviklingsmetodologi.

Vedlegg B: Backupmedia

B.1.1 Harddisk

En annen lagringsenhet for data er harddisk, også kalt HDD (hard disk drive) eller hard drive. Harddisker ble oppfunnet i 1956 av IBM, og de aller første harddiskene var store tungtvinte, og de ble kun brukt på beskyttede datasentre, bedrifter eller store kontorer. Før 1980 tallet fantes det harddisker som var flyttbare og relativt store, 8" (20 cm) og 14" (35 cm), og de trengte mye plass for å oppbevare dem. Disse harddiskene fikk kallenavnet "vaskemaskiner". På grunn av størrelsene til disse harddiskene ble ikke harddisker tatt i bruk for hjemme PC-er før etter 1980 årene. Da ble det utviklet nye mindre harddisker på 5,25" med en kapasitet på 5 MB. Etter hvert har kapasitetene på harddisker økt betraktelig. I midten av 1990 tallet har det økt til 1 GB; utover 90 tallet har det økt til 40-60 GB. I skrivende stund, år 2007, finnes det harddisker opptil 750 GB. I IT miljøet har det vært snakk om å utvikle harddisker fra 1 TB i løpet av 2007, og i nærmeste fremtid vil det komme enda større harddisker. Samtidig som kapasiteten til harddisker øker, vil prisen bli gunstigere etter hvert. Det gjør at harddisker blir hovedkonkurrenten til magnetisk tape som brukes til lagring av store mengder data. Den største fordelen med harddisker er at de kan lagre store mengder data og at det tar kort tid å overføre filer til og fra disken.
Siden midten av 90 tallet har det vært vanlig med å ha en ekstern harddisk i tillegg til en (eller flere) interne harddisk på PC. Med en ekstern harddisk kan man koble til PC via USB eller Firewire. Denne løsningen egner seg bra for folk som vil flytte på backup data mellom flere PC-er.

Første harddisk fra 1956 – IBM 305 RAMAC

Harddisker gjennom tidene.

Årstall	Hendelser
1956	IBM hadde utviklet den aller første harddisk kalt IBM 305 RAMAC den 13. september 1956. Harddisken hadde kapasitet på 5 MB, og kostet 50 000 $; det vil si 10 000 $ pr MB. Denne harddisken var omtrent like stor som to kjøleskaper til sammen og hadde plass til 50 stk 24 tommers diskplater.
1961	IBM oppfant skrivehoder for disker som ”flyr” på en pute av luft eller ”luftbærer”.
1963	IBM oppfant den første flyttbare harddisk, kalt for 1311. Denne harddisken brukte 6 stk. 14 tommers diskplater og hadde en kapasitet på 2,6 MB.
1966	IBM introduserte den første harddisk som brukte en spole skrivehode.
1970	General Digital Corporation (omdøpt til Western Digital I 1971) ble grunnlagt i California.
1973	IBM annonserte den første moderne harddisk kalt 3340 som hadde en forseglet innpakning, smøret stenger og low-mass hoder.
1978	Første RAID (Redundant Arrays of Independent Disks) teknologi ble tatt i bruk.
1979	• En gruppe ledet av Al Shugart grunnla en harddisk fabrikk kalt Seagate Technology. • IBMs 3370 brukte 7 stk 14 tommers diskplater for å lagre 571 MB. Denne var den første disken som brukte tynn-film hoder. • IBMs 62 PC kalt Piccolo brukte 6 stk 8 tommers diskplater til å lagre 64 MB. • Seagate introduserte ST-506 disk og grensesnitt som ble brukt i mikrodatamaskin implementering.
1980	• IBM introduserte den første gigabyte harddisk. Den var like stor som et kjøleskap og veide over 200 kg, og den kostet

	40 000 $. • Seagate lanserte den første 5,25 tommers harddisk.
1982	Western Digital annonserte den første singel brikke Winchester harddisk kontroller (WD1010).
1983	Rodime lanserte den første 3,5 tommers harddisk; RO352 som hadde to diskplater, og med en kapasitet på 10 MB.
1984	Western Digital lagde første Winchester harddisk kontroller kort for IBM PC/AT.
1985	• Control Data, Compaq Computer og Western Digital samarbeidet for å utvikle 40-pin IDE grensesnitt. IDE står for Intelligent Drive Electronics. • Imprimis integrerte første harddisk kontroller på en driver. • Quantum introduserte Plus Hardcard som gjorde mulig å • Western Digital produserte første ESDI (Enhanced Small Device Interface) kontrollpanel som gjorde det mulig å bruke raskere harddisk med større kapasitet på PC-er.
1986	Den offisielle SCSI spesifikasjon ble lansert. Apples datamaskiner var en av de første som brukte denne.
1988	• Prairie Tek lanserte den første 2,5 tommers harddisk kalt 220. Den var designet for bærbare PC-er. Den brukte to diskplater for å lagre 20 MB. • Connor introduserte den første 1 tommers høy, 3,5 tommers harddisk som fremdeles er i bruk den dag i dag.
1990	Western Digital introduserte den første 3,5 tommers Caviar IDE harddisk.
1991	IBM introduserte 0633 Corsair, den første platedrev med tynn film magnetresistive hoder. Den hadde 8 stk 3,5 diskplater og lagret 1 GB.
1992	• Seagate kom ut med første støtføling 2,5 tommers harddisk. • Seagate lanserte den første 7200 rpm (revolutions per minute) harddisk, nemlig 2,1 GB Barracuda. • Hewlett Packards C3013A Kitty Hawk disk brukte 2 stk 1,3 tommers diskplater for å lagre 2,1 GB.
1994	Western Digital utviklet Enhanced IDE, et forbedret harddiskgrensesnitt. EIDE tillater også forbindelse av optiske og tapedrivere.
1996	Seagate introduserte sin Cheetah familie, nemlig de første 10 000 rpm harddisker.
1997	IBM introduserte den første harddisksom bruker GMR (giant magneto resistive), nemlig 16,8 GB Deskstar 16GP Titan.
1998	IBM annonserte sin Microdrive, den minste harddisken. Den får plass til 340 MB på en 1 tommers diskplate.
2000	• Maxtor kjøpte opp konkurrenten Quantum. Med denne overtagelsen gjorde Maxtor til verdens største harddisk fabrikant.

	• Seagate produserte den første 15 000 rpm harddisk kalt Cheetah X15.
2002	Seagate var først igjen med ny type harddisk: Barracuda ATA V, den første Serial ATA harddisk.
2003	• IBM solgte sin Datalagringsavdeling til Hitachi, og dermed avsluttet de produksjonen og markedsføring av harddisk. • Western Digital introduserte den første 10 000 rpm SATA harddisk, 37 GB Raptor.
2004	Toshiba introduserte sin første 0,85 tommers harddisk, MK2001MTN. Den kan lagre 2 GB på en singel diskplate
2005	Toshiba introduserte sin MK4007 GAL, som kan lagre 40 GB på en 1,8 tommers diskplate.
2006	• Seagate sin Momentus 5400.3, en harddisk for bærbar PC var den første 2,5 tommers modell som brukte magnetisk skriving. Denne harddisken har en kapasitet opptil 160 GB. • Seagate lanserte Barracuda 7200.10, den største harddisken som finnes hittil med en kapasitet på 750 GB. • Western Digital lanserte 10 000 rpm Raptor X SATA harddisk. • Cornice og Seagate annonserte hver sin1 tommers harddisk som kan oppbevare 12 GB.

Historikk om harddisker

B.1.2 Tape

Tape drive er en datalagringsenhet som leser og skriver data lagret på magnetisk tape. Tape ble tatt i bruk på 1980 tallet da det var behov for flyttbare backup media. Med tape kan man ta med seg mengder av data til andre PC-er. Magnetisk tape har lenge vært brukt til lagring av store mengder data og kopier av filer.
Tape er ikke så mye brukt i dag ettersom det har kommet nye backupmedier på markedet som er både billigere og lettere å bruke. Det finnes i dag et fåtall nettbutikker som fortsatt selger tape drive og magnetisk tape.
Tape har som regel hatt større kapasitet enn harddisk, men i det siste har tape og harddisk blitt omtrent like. Prisene på magnetisk tape varierer fra ca 300 kr til 800 kr. Det kommer litt an på kapasiteten til disse tapene. Vanligvis er kapasiteten på 20 – 40 GB.

B.1.3 Floppy disk

En annen flyttbar lagringsenhet er Floppy (fleksible) disketter. Disketter ble brukt mye mellom 1980 og 1990 tallet for å lagre kopier av data og overføre dokumenter. Før harddisker ble populære, ble disketter brukt til lagring av operativsystem, applikasjon software og andre data. På 1990 tallet ble størrelsen på applikasjoner økt, og da var ikke disketter veldig nyttig lenger. I dag produseres ikke PC-er med diskettstasjoner lenger. Disketter er på god vei ut av markedet. Det selges fremdeles diskettstasjoner og disketter på noen få nettbutikker og databutikker.

Floppy disk.

B.1.4 Optisk disk

Optisk disk er en ny flyttbar lagringsenhet som erstattet både tape backup og diskett backup. Optisk disk ble oppfunnet på 1960 tallet, men backup på optisk disk kom ikke før på begynnelsen av 1990 tallet da det ble utviklet CD-R og CD-RW plater. I tillegg må man ha en CD-brenner for å lagre data på en optisk disk. Fordelen med optiske disker er at de er flyttbare og de har mye mer kapasitet enn disketter. Ulempen er hvis man skal lagre store mengder data så er man ta nødt til å lagre på flere disker. Da kan det medføre til at viktige data blir rotet bort, og man ikke har god oversikt over alt som er lagret.
I dag finnes det flere optiske media typer på markedet som brukes til lagring av data: CD-R, CD-RW, DVD-R, DVD+R, DVD-RW og Blue-ray.
En CD-R plate kan brukes som en backup enhet. En fordel med CD plater er at de kan lagre mye data på en plate. Vanlige kapasiteter på en CD-R plate i dag er på 650 – 800 MB. Det tilsvarer lagring av ca 12000 bilder eller ca 500 MP3 filer på en plate. Dataene som blir lagret på en CD-R plate kan gjenopprettes på hvilken som helst PC med en CD-ROM.
Et annet tilsvarende format er DVD+R. Dette formatet har en kapasitet på 4.7 GB. Dataene som lagres i DVD+R eller CD-R kan ikke forandres eller slettes etter at de er blitt lagret på platen; de kan kun leses av platen. Med DVD+RW og CD-RW kan hver enkelte fil eller hele innholdet på platen slettes og overskrives.
Det kommer etter hvert et nytt format som kalles Blu-ray, og denne platen har en kapasitet på ca. 50 GB. Det finnes både singel og dual layer plater (dobbeltsidige plater).
Pris: Blu-ray 25 GB, ca 140 kr pr stk. CD-R 700 MB ca 4,90 kr pr stk. DVD 4.7 GB ca 6,90 kr pr stk.

Optisk disk. Fra venstre: CD-R, DVD-R, HD-DVD og Blue-ray BD.

B.1.5 Andre lagringsenheter

Andre typer små lagringsenheter er USB minnepenn og diverse type minnekort som Compact flash, Smart media, Memory stick, Secure Digital (SD kort), mini Security Digital (miniSD) osv. Disse enhetene er relativt kostbare og har lave kapasitet. I skrivende stund har en USB minnepenn har minnestørrelser fra 512 MB til 16 GB på markedet, og minnekortene har minnestørrelser fra 512 MB opp til 4 GB. Fordelen med disse enhetene er at de er ganske nyttige siden de er små, lett å ta med overalt, og de er lette å bruke. En USB minnepenn settes inn på USB porten på en PC, og den er klar til bruk. Minnekortene settes inn i kortleseren hvis den støtter kortet. Kortleseren er enten integrert på PC eller ekstern tilkoblet. En ulempe med disse enhetene er at de kan lagre kun små mengder data pr i dag. En annen ulempe kan være hvis en PC ikke har USB port eller minnekortleser; da blir disse enhetene nyttesløs.
USB penn har priser fra 150 kr til 3000 kr. Priser på minnekortene varierer fra 100 kr til 2000 kr. Prisene synker stadig ettersom det blir utvidet med større kapasiteter. Disse lagringsenhetene er dyrere enn harddisker. De koster fra 100 til 300 kr pr GB, mens harddisker koster fra 2 til 4 kr pr GB.

B.1.5.1 USB flash drive

USB flash drive (også kalles for USB penn) er en moderne form for flyttbar lagringsenhet. USB penn ble oppfunnet i 1998. I begynnelsen var kapasitetene relativt små. De første USB pennene var på 16 MB, 32 MB og 64 MB, og det var snakk om at disse enhetene skulle erstatte disketter for godt siden de er mindre og lettere å bruke samtidig som de har langt større kapasitet. Kapasiteten til USB penn har på lik linje med harddisk økt betraktelig i de siste årene. I dag kan man kjøpe USB penn med minste kapasitet på 512 MB og største på 16 GB. Samtidig har også formen og utseendet blitt forbedret; liten og fin, lett å ta med overalt.

USB minnepenn i forskjellige fasonger.

B.1.5.2 Minnekort

Minnekort er en moderne form for flyttbar lagringsenhet som ble tatt i bruk mot slutten av 1990 tallet og i begynnelsen av år 2000. Minnekort brukes mest på digitalt kamera, MP3 spillere, mobiltelefoner, spillkonsoller og alle elektroniske utstyr. Brukes også på bærbare og stasjonære PC-er, men det er betinget at PC må enten ha innebygd kortleser eller ekstern kortleser. Minnekort har blitt foreslått som en mulig erstatter for disketter, og i dag selges det PC-er med innebygd kortleser istedenfor diskettstasjon. Minnekort typene som finnes på markedet i dag er nevnt i følgende tabell:

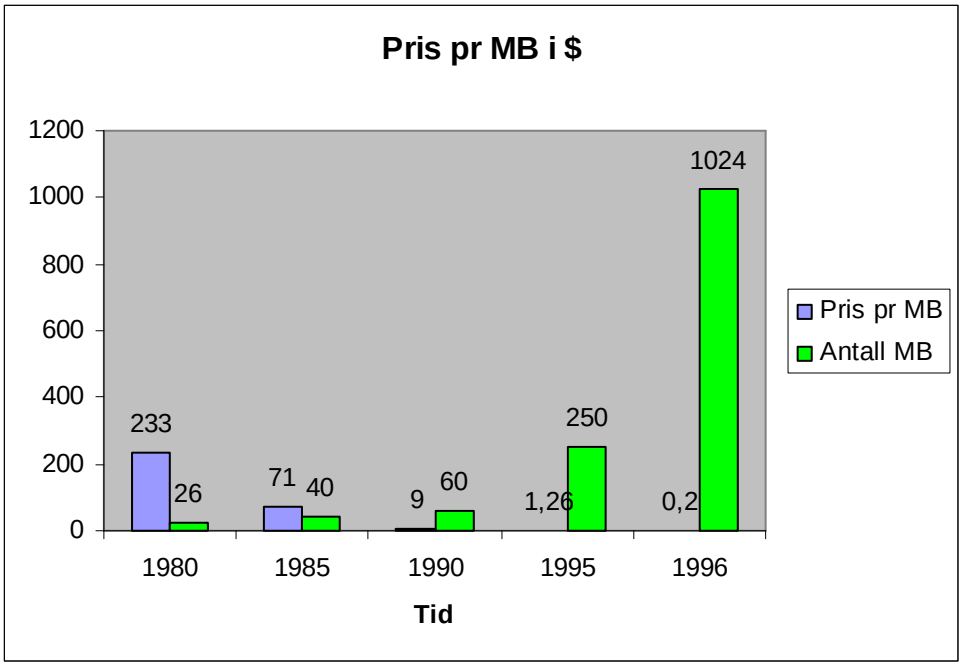
Navn	Akronym	Form faktor
PC-Card	PCMCIA	85.6 × 54 × 3.3 med mer
CompactFlash I	CF-I	43 × 36 × 3.3 med mer
CompactFlash II	CF-II	43 × 36 × 5.5 med mer
SmartMedia	SM / SMC	45 × 37 × 0.76 med mer
Memory Stick	MS	50.0 × 21.5 × 2.8 mm
Memory Stick Duo	MSD	31.0 × 20.0 × 1.6 mm
Memory Stick Micro M2	M2	15.0 × 12.5 × 1.2 mm
Multimedia Card	MMC	32 × 24 × 1.5 mm
Reduced Size Multimedia Card	RS-MMC	16 × 24 × 1.5 mm
MMCmicro Card	MMCmicro	12 × 14 × 1.1 mm
Secure Digital Card	SD	32 × 24 × 2.1 mm
miniSD Card	miniSD	21.5 × 20 × 1.4 mm
microSD Card	microSD	11 × 15 × 1 mm
xD-Picture Card	xD	20 × 25 × 1.7 mm
Intelligent Stick	iStick	24 x 18 x 2.8 mm

Minnekort tabell

Minnekort

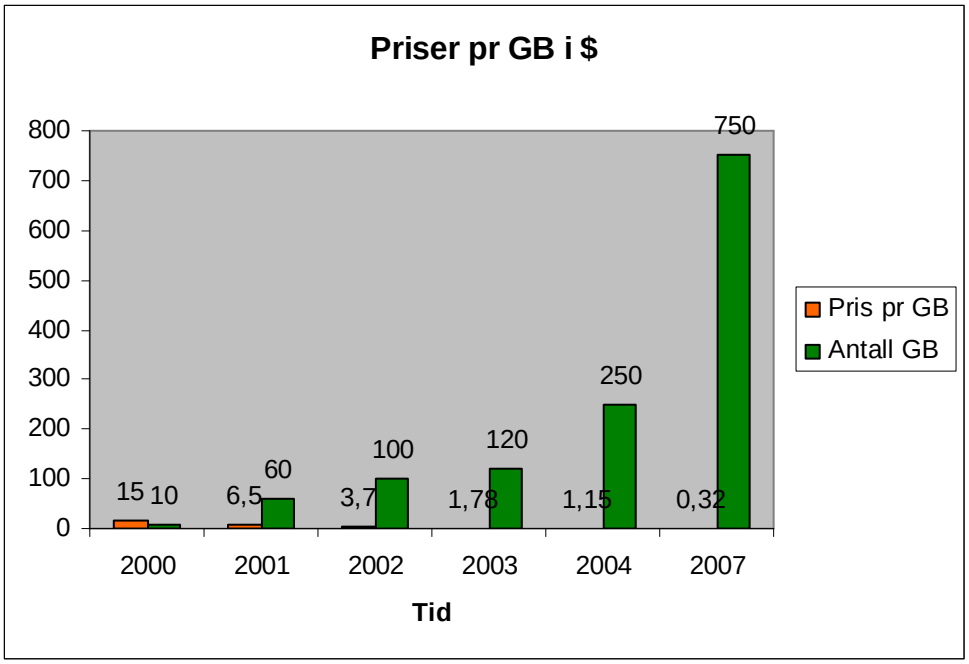
B.2 Statistikk

I dette avsnittet viser statistikk for utvikling og prisendringer på backup mediene grafer. Prisene i grafene er oppgitt i amerikanske dollar $. Prisen på den aller første harddisken fra 1956 koster 50 000 $ for 5 MB, det vil si 10 000 $ pr MB. I grafene under har vi tatt med tall fra 1980 siden første harddisk til hjemme PC kom ikke på markedet før på 1980 tallet.



Pris pr Megabyte

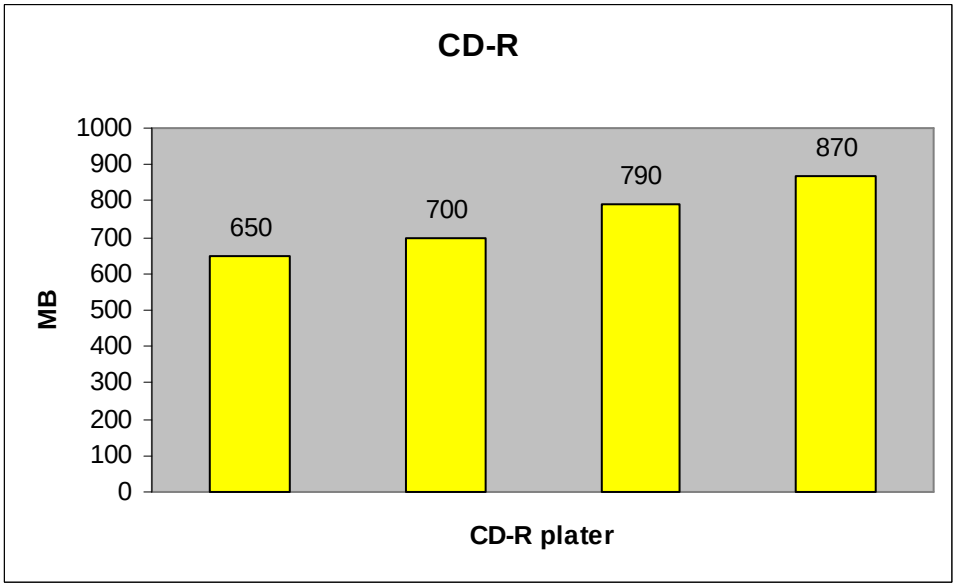
Fra og med år 2000 er det ikke lenger snakk om pris pr MB, men kun om pris pr GB. På tabellen under kan man se utviklingen av kapasitet og prisendringer.



Pris pr Gigabyte

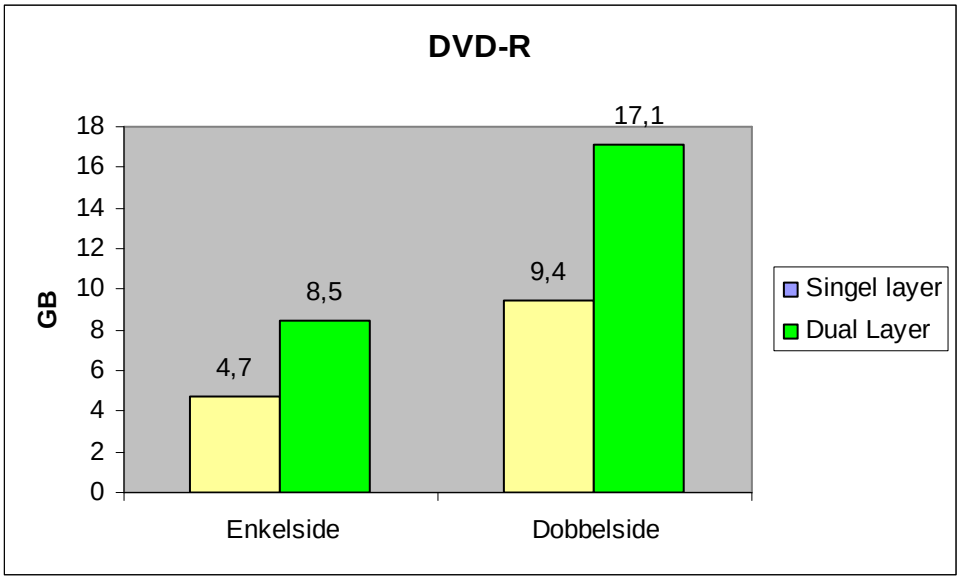
Ut i fra de to grafene ovenfor kan vi se at prisen på harddisker har sunket drastisk samtidig som kapasiteten fortsetter å øke.

Optisk disk har ikke hatt så stor økning på kapasitet slik som harddisker, men det har blitt utviklet nye formater som DVD, Blu-ray og HD-DVD. De to sistnevnte er nye på markedet.

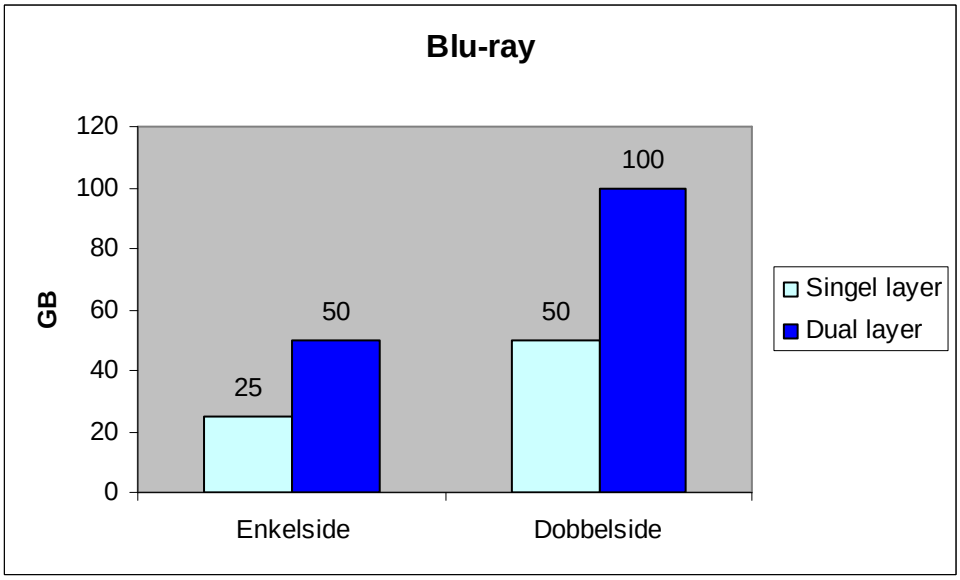


Lagringskapasitet for CD-R

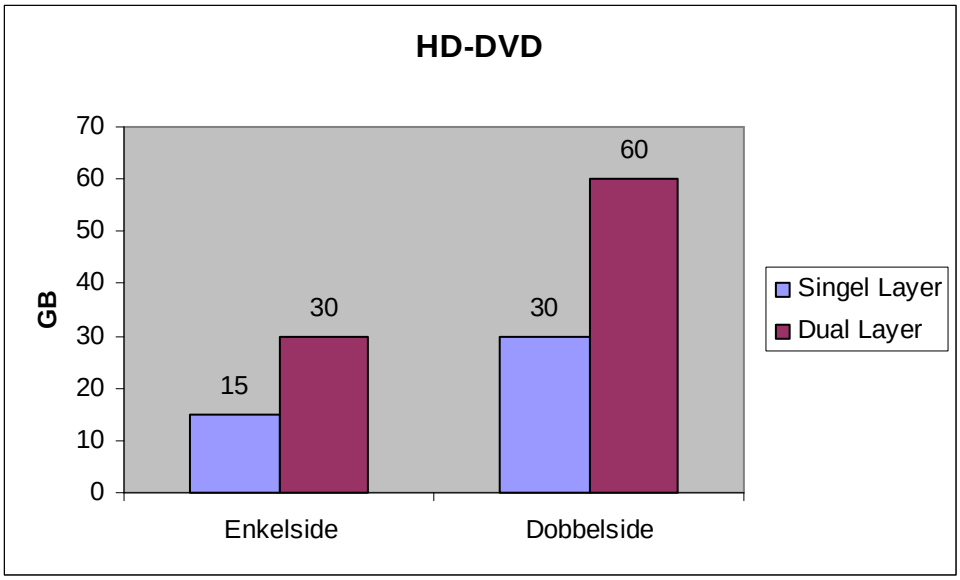
Grafene under viser kapasiteten til både singel og dual layer til enkelsidet og dobbelsidet DVD, Blu-ray og HD-DVD.



Lagringskapasitet for DVD-R



Lagringskapasitet for Blu-ray



Lagringskapasitet for HD-DVD

Vedlegg C: Kravspesifikasjon

C.1.1 Innledning

Denne kravspesifikasjonen beskriver hvilke funksjoner som skal utvikles i backup systemet; ikke hvordan funksjonene skal implementeres. Funksjonene som beskrives i denne kravspesifikasjonen består av funksjoner systemet må ha for å virke og funksjoner som kan implementeres ved behov. Funksjonsbeskrivelsene av funksjonene som må være med i systemet er delt opp i *funksjonelle krav* og *ikke-funksjonelle krav*.
Funksjonelle krav beskriver hvilke funksjoner backup systemet har å tilby, og hva hver enkelte funksjon gjør i forskjellige hendelser.
Ikke-funksjonelle krav beskriver hva som er nødvendig å ha med for å kunne bruke dette backup systemet. Nødvendige krav til systemet er brukergrensesnitt, hardware, software og kommunikasjon.

C.1.2 Leserveiledning

Dette kapitlet beskriver de funksjonelle kravene vårt system skal bestå av ut i fra de funksjonene vi mener bør være med i et slikt system. Kravene vil bli beskrevet i en tabell, samt en mer utfyllende tekst der det er behov. En figur vil vise en use case diagram for de kravene det er nødvendig. Tabellen vil inneholde et kravspesifikasjonsnummer, hva den gjør, samt hvilken prioritet denne funksjonen har. Det vil bli delt inn i tre prioriteter i dette dokumentet. Lav prioritet er funksjoner som kan implementeres om ønskelig, men som ikke er nødvendig. Middels prioritet er funksjoner som kan implementeres om ønskelig, men er funksjoner vi anbefaler blir implementert for å gjøre systemet mer robust. Høy prioritet er funksjoner som må implementeres for at systemet skal virke slik det skal.
Kravspesifikasjonsnummeret vil bli skrevet som KR####, hvor #### betyr kravnummer. Det første tallet står for om kravet er et funksjonelt eller ikke-funksjonelt krav. Tall nummer 2 står for hvilke type kravspesifikasjonsgruppe kravet tilhører. Tall nummer tre står for eventuelt hvilke krav nummer den har innen denne gruppen, eller hvilke undergruppe kravet tilhører. Om det er en undergruppe til kravet blir et fjerde tall benyttet for å gi kravet et nummer. Et eksempel på et krav kan være KR 1.2.1.2, som er en GUI funksjon for å legge til en bruker til systemet. Det første tallet forteller at dette er et funksjonelt krav. Tall nummer to forteller at kravet tilhører gruppen for GUI funksjoner. Tall tre viser til at dette er en funksjon som tilhører menylinjen i applikasjonen, og det siste tallet hvilket funksjonsnummer den har i denne gruppen.

C.2 Generell beskrivelse

C.2.1 Systemperspektiv

Uttrykk helheten i systemet her.

C.2.2 Systemkjennetegn

Backup skal foregå mellom to maskiner av gangen. Brukere av systemet skal kunne overføre filer til en annen PC og gjenopprette disse filene.

C.2.3 Brukerklasser og egenskaper

Systemet er beregnet for private brukere siden dette systemet begrenser seg til å ta backup av små mengder data, og at overføringen av filer foregår mellom to hoster over internett. Systemet skal være brukervennlig slik at alle brukere med generell datakunnskap skal kunne benytte seg av dette systemet.

C.2.4 Omgivelser

Backup systemet skal kunne brukes på en standard PC, og den vil ikke fungere på Mac. Operativsystemet som støttes er kun Windows 98 og nyere versjoner. I tillegg må PC-ene ha Ethernet kort som følger 802 standarden, samt tilgang til internett. Det stilles ingen ytterligere krav til hardware i dette systemet.

C.2.5 Design og implementeringsbegrensning

C.2.5.1 Plattform og utviklingsverktøy

Dette systemet skal utvikles til å kjøres på en 32 bit Windows operativsystem med .NET versjon 2.0 installert. Utviklingsverktøyet som skal benyttes for å utvikle systemet er Visual Studio 2005, og programmeringsspråket skal være C#.

C.2.5.2 Lagring av brukerinformasjon

På serveren skal brukerinformasjonen til alle brukerne lagres i en XML fil. Denne filen skal inneholde informasjon om brukernavn, passord, e-post adresse, IP - adresse og status.

På klienten skal brukere i brukerlisten lagres i en XML fil, som inneholder informasjon om brukernavn, IP – adresse og status til den enkelte bruker. Den personlige brukerinformasjonen skal lagres i en vanlig tekstfil som inneholder brukernavn, passord og e-post adresse.

C.2.5.3 Kryptering

All informasjon om brukerinformasjon på server og klient skal være kryptert. Data som sendes mellom klient og server, og mellom klientene skal være kryptert. Dette inneholder meldinger og filer som blir overført.

C.2.6 Brukerdokumentasjon

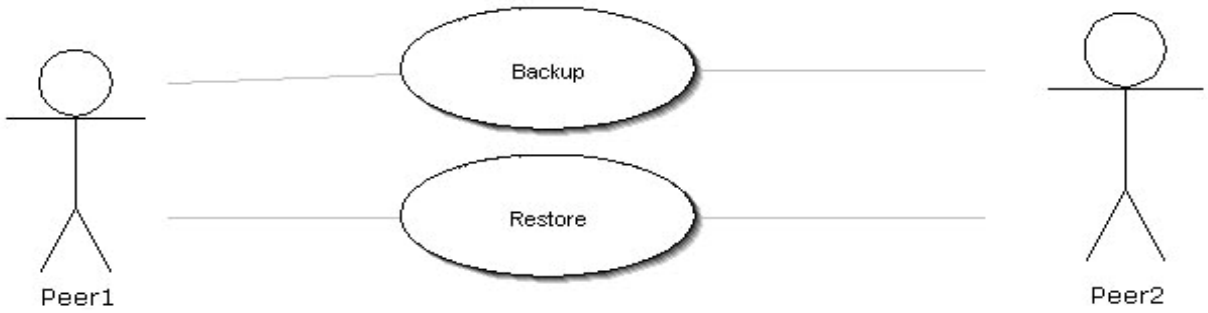
Det skal bli levert en hjelpemeny i applikasjonen. Denne menyen skal gi en beskrivelse på hvordan man skal komme i gang, samt en detaljert beskrivelse av hvordan hver enkelt funksjon i programmet virker. Det skal ikke følge med noen brukermanual med dette systemet, og det skal ikke være noen online hjelp.

C.3 Funksjonelle krav

C.3.1 Sluttbrukerfunksjoner

C.3.1.1 Backup/restore av data

Disse to funksjonene er hovedfunksjonene i vårt system. Disse funksjonene inneholder alle funksjonelle krav som skal til for å utføre backup og gjenoppretting av filer. Disse funksjonene har høy prioritet, og vil være de funksjonene vi vil legge mest vekt på i designet av prototypen.



Use case diagram for backup/restore

KR 1.1.1	**Backup av filer**
Handling	Brukeren tar backup av filer over på en annen host
Prioritet	Høy prioritet
Deltagere	Peer1 og Peer2
Betingelse	Både sender og mottaker av data må være pålogget systemet
Funksjonelle krav	FK - 1.1.1.1.: Velge den man skal overføre til FK - 1.1.1.2: Velge filer og mapper det skal tas backup av FK - 1.1.1.3: Sjekker filer som er endret siden forrige backup FK - 1.1.1.4: Kryptere filer som skal overføres FK - 1.1.1.5: Finne IP til den man skal overføre til FK - 1.1.1.6: Sende filer til venn FK - 1.1.1.7: Lagring av data

Funksjonsoversikt av backup av data

KR 1.1.2	**Gjenoppretting av filer**
Handling	Brukeren gjenoppretter filene han/hun har tatt backup av
Prioritet	Høy prioritet
Deltagere	Peer1 og Peer2
Betingelse	Både sender og mottaker av data må være pålogget systemet
Funksjonelle krav	FK - 1.1.2.1: Velge person man skal ta gjenoppretting av FK - 1.1.2.2: Velge filer det skal tas gjenoppretting av FK - 1.1.2.3: Velge lagringssted på maskin

Tabell for funksjoner i gjenoppretting av data

KR 1.1.3	**IM (Chatting)**
Handling	Brukere kan sende hverandre korte meldinger
Prioritet	Middels prioritet
Deltagere	Peer1 og Peer2
Betingelse	Applikasjon startet og koblet opp mot systemet
Funksjonelle krav	

IM chatting

KR 1.1.4	**Manuell pålogging/automatisk pålogging**
Handling	Velge automatisk/manuell pålogging
Prioritet	Middels prioritet
Deltagere	Peer1
Betingelse	
Funksjonelle krav	

Manuell/automatisk pålogging

KR 1.1.5	**Fildeling**
Handling	Dele filer mellom peers
Prioritet	Lav prioritet
Deltagere	Peer1 og Peer2
Betingelse	Mapper og filer på maskinen er delt
Funksjonelle krav	

Fildeling

C.3.2 GUI funksjoner

C.3.2.1 Meny-linje

KR 1.2.1.1	**Avslutt-knapp**
Handling	Aktiverer krav KR 1.3.4
Prioritet	Høy prioritet
Funksjonelle krav	

Avslutt knapp

KR 1.2.1.2	Login/Logout-knapp
Handling	Aktiverer krav KR 1.3.5 og KR 1.3.6
Prioritet	Høy prioritet
Funksjonelle krav	

Login/Logout knapp

KR 1.2.1.2	**Legge til flere brukere man ønsker backup mot**
Handling	Aktiverer krav KR1.3.7
Prioritet	Lav prioritet

Legg til brukere

KR 1.2.1.3	**Slette brukere man har lagt til**
Handling	Aktiverer krav KR 1.3.8
Prioritet	Lav prioritet

Slette brukere

KR 1.2.1.5	**Hjelpemeny**
Handling	Viser en hjelpemeny som viser applikasjonens funksjoner
Prioritet	Høy prioritet

Hjelpemeny

KR 1.2.1.5	**Redigere e-post adresse til bruker**
Handling	Aktiverer KR 1.2.7
Prioritet	Lav prioritet

Redigere adresser

KR 1.2.1.6	**Endre passord**
Handling	Aktiverer KR 1.2.7
Prioritet	Lav prioritet

Endre passord

KR 1.2.1.7	**Hente passord**
Handling	Få tilsendt glemt passord
Prioritet	Lav prioritet

Hente passord

C.3.2.2 Vindu ved oppstart

KR 1.2.2.1	**Velge bruker**
Handling	Velge den man skal overføre data til
Prioritet	Høy prioritet
Type GUI komponent	Ikoner som viser hver bruker

Velge bruker

KR 1.2.2.2	**Foreta backup**
Handling	Aktivere krav KR 1.1.1 for backup av data
Prioritet	Høy prioritet
Type GUI komponent	Knapp merket ”Backup”

Backup

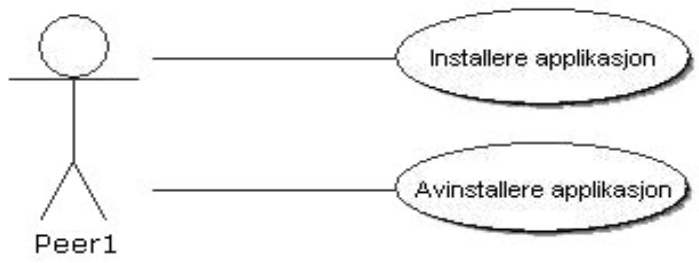
KR 1.2.2.3	**Foreta gjenoppretting**
Handling	Aktiverer krav KR 1.1.2 for gjenoppretting av data
Prioritet	Høy prioritet
Type GUI komponent	Knapp merket ”Restore”

Restore

C.3.3 Administrative funksjoner

C.3.3.1 Installasjon/avinstallasjon av program

Installasjon er en nødvendig funksjon for å kopiere de nødvendige filene som behøves for at dette systemet skal virke som det skal. Under installasjon må det være noe konfigurasjon brukeren selv må gjøre. Disse er listet i tabell for installasjon.
Avinstallasjon av applikasjonen vil sørge for at alle nødvendige filer som er lagt inn vil bli slettet. Dette gjelder installasjonsfiler og brukerinnstillinger. Hendelsesforløpet for denne funksjonen blir vist i tabell for avinstallasjon.



Use case diagram for installering og avinstallering av applikasjon

KR 1.3.1	**Installasjon**
Handling	Installering av applikasjonen
Prioritet	Høy prioritet
Deltagere	Peer1
Betingelse	PC-en må ha Windows XP
Funksjonelle krav	

Installasjon av applikasjon

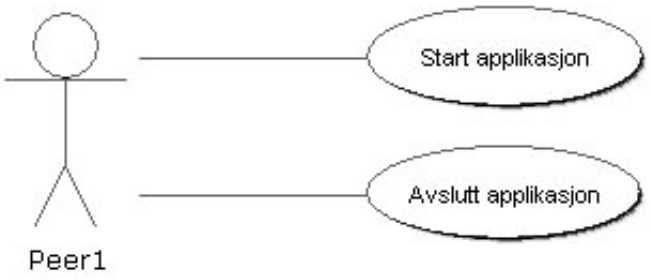
KR 1.3.2	**Avinstallasjon**
Deltagere	Peer1
Handling	Avinstallasjon av applikasjonen
Betingelse	Applikasjonen er allerede installert på PC
Funksjonelle krav	

Avinstallasjon av applikasjon

C.3.3.2 Start/Avslutt

Starter opp applikasjonen for å begynne å ta backup eller gjenopprette filer.

Use Case diagram:



Use case diagram over start og avslutt funksjon

KR 1.3.3	**Start applikasjon**
Handling	Brukeren starter opp applikasjonen
Prioritet	Høy prioritet
Deltagere	Peer1
Betingelse	Applikasjonen er installert og klar til å bruke
Funksjonelle krav	- Innlogging - IP status

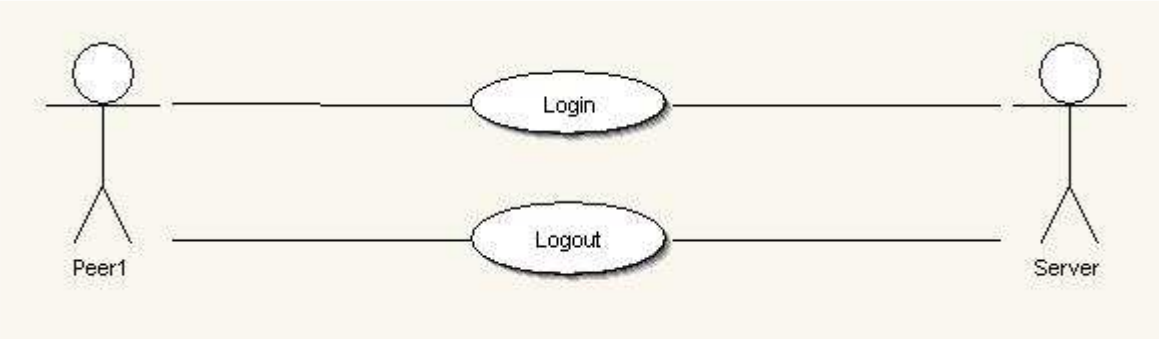
Start av applikasjon

KR 1.3.4	**Avslutt applikasjon**
Handling	Brukeren avslutter applikasjonen
Prioritet	Høy prioritet
Deltagere	Peer1
Betingelse	
Funksjonelle krav	

Avslutt av applikasjon

C.3.3.4 Login/Logout

Brukeren logger seg av og på serveren for å fortelle sin status.



KR 1.3.5	**Login**
Handling	Brukeren logger seg på serveren
Prioritet	Høy prioritet
Deltagere	Peer1
Betingelse	Applikasjonen må være startet
Funksjonelle krav	

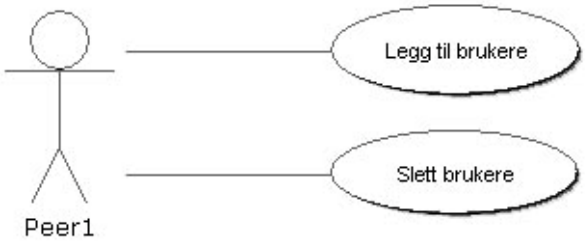
Logge inn på server

KR 1.3.6	**Logout**
Handling	Brukeren logger seg av serveren
Prioritet	Høy prioritet
Deltagere	Peer1
Betingelse	Brukeren må være logget på
Funksjonelle krav	

Logge ut av server

C.3.3.3 Legge til/slette brukere

Brukeren legger til andre brukere på sin backup liste.



Use case diagram for å legge til og slette brukere

KR 1.3.7	**Legge til brukere**
Handling	Brukeren legger til andre brukere
Prioritet	Høy prioritet
Deltagere	Peer1
Betingelse	• Applikasjon må være startet • Applikasjon må være tilkoblet server • Bruker en skal legge til må være registrert
Funksjonelle krav	FK – 1: Søke etter eksisterende brukere FK – 2: Godkjenne bruker

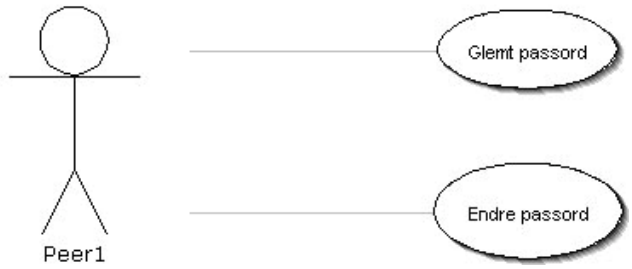
Legge til brukere

KR 1.3.8	**Slette brukere**
Handling	Brukeren sletter andre brukere
Prioritet	Høy prioritet
Deltagere	Peer1
Betingelse	- Applikasjonen er startet - Applikasjonen er tilkoblet server - Brukere må finnes på backup lista til ”Peer1”
Funksjonelle krav	

Slette brukere

C.3.3.4 Administrering av passord

Funksjonene som innebærer i dette er å kunne forandre passord og få tildelt passord om dette blir glemt. Det må være mulig for en bruker å kunne endre et passord, ettersom man ofte ønsker å legge til passord som kan være enten enklere å huske, eller at man ønsker et sikrere passord. Oppretting og sletting av brukere vil komme under avsnittet installering/avinstallering av applikasjon, ettersom dette tilhører disse funksjonene. Administrering av brukere er av mindre prioritet for utviklingen av systemet, men er en nyttig funksjon for brukere av systemet da dette gir de valgfrihet til å kunne endre passord når det er ønskelig.



Administrering av passord

KR 1.3.9	**Endring av passord**
Handling	Brukeren endrer passord for innlogging til server
Prioritet	Lav prioritet
Deltagere	Peer1
Betingelse	Brukeren må være pålogget systemet
Funksjonelle krav	FK-1: Endring av passord

Endring av passord

KR 1.3.10	**Tilsending av nytt passord**
Handling	Bruker ønsker passord sendt til seg
Prioritet	Lav prioritet
Deltagere	Peer1
Betingelse	Brukeren må være pålogget systemet
Funksjonelle krav	FK-1: Sending av passord

Sending av nytt passord

C.3.3.5 Valg av kryptering

For å gjøre valgmuligheten større for brukere av systemet, kan det være fint å kunne velge type kryptering man ønsker på sine filer. Det vil derfor være mulig å legge til flere typer krypteringsalgoritmer, for å gjøre overføring av data mer sikret.

KR 1.3.11	**Valg av kryptering**
Handling	Velge kryptering brukt ved overføring av data
Prioritet	Lav prioritet
Deltagere	Peer1
Betingelse	
Funksjonelle krav	

Valg av kryptering

C.3.3.6 Smartkort autentisering

For å gjøre autentiseringen mer robust, kan man benytte seg av en 2 faktor autentisering. For å gjøre dette mulig kan vi implementere autentisering via smartkort. På denne måten er man nødt til å eie et gyldig smartkort og i tillegg vite passordet for innlogging. Dette gir en ekstra grad av sikkerhet noe som er til fordel for brukeren av systemet.

KR 1.3.12	**Smartkort autentisering**
Handling	Autentisere seg ved hjelp av smartkort
Prioritet	Lav prioritet
Deltagere	Peer1
Betingelse	Smartkortleser installert
Funksjonelle krav	

Smartkort autentisering

C.3.4 Automatiske funksjoner

KR 1.4.1	**Backup av endrede filer**
Handling	Foretar backup
Prioritet	Lav prioritet
Deltagere	Peer1, Peer2
Betingelse	- Begge er logget på systemet

Backup av endrede filer

KR 1.4.2	**Lagring av data**
Handling	Automatisk lagring av data på bestemt plassering
Prioritet	Høy prioritet
Deltagere	Peer1
Betingelse	

Lagring av data

KR 1.4.3	**Kryptering av data**
Handling	Krypterer data før overføring
Prioritet	Høy prioritet
Deltagere	Peer1
Betingelse	KR 1.2.2.2 er aktivert

Kryptering av data

C.4 Ikke-funksjonelle krav

C.4.1 Krav til ytelse

KR 2.1.1	**Hardware**
Krav	Hardware ytelse
Prioritet	Høy prioritet
Ytelseskrav	PC som kan kjøre Windows 2000 eller nyere

Krav til hardware

KR 2.1.2	**Nettverkskort**
Krav	Nettverkskort
Prioritet	Middels prioritet
Ytelseskrav	Følge 802.3 Ethernet standarden Følge 802.11 WLAN standard

Krav til nettverkskort

KR 2.1.3	**Overføring**
Type krav	Overføringshastighet
Prioritet	Middels prioritet
Ytelseskrav	Minimum 385 Kbps bredbåndslinje

Krav til overføringshastighet

KR 2.1.4	**Innlogging**
Type krav	Autentisering av bruker
Prioritet	Høy prioritet
Ytelseskrav	Innloggingsfasen må skje innen 10 sekunder

Krav til autentiseringsfasen

C.4.2 Sikringskrav

KR 2.2.1	**Brudd på overføring**
Type krav	Sikring av data som overføres
Prioritet	Høy prioritet
Ytelseskrav	Ved brudd på nettverksforbindelsen må det være funksjoner for å kunne fortsette overføringen der den slapp

Brudd på overføring

C.4.3 Sikkerhetskrav

KR 2.3.1	**Registrering**
Type krav	Generering av passord
Prioritet	Middels prioritet
Ytelseskrav	Brukernavn skal være valgfritt. Passord skal bestå av 6 tegn og minst 1 av de må inneholde et siffer.

Tabell for innlogging

KR 2.3.2	**Lagring av data**
Type krav	Kryptering av lagret data
Prioritering	Høy prioritering
Ytelseskrav	Krypteringsnøkkel må minimum 256 bit, og kun eier av data har kryptering/dekrypteringsnøkkel

Tabell for lagring av data

KR 2.3.3	**Kryptering av melding**
Type krav	Kryptere melding som blir utvekslet
Prioritering	Høy prioritering
Ytelseskrav	Krypteringsnøkkelen skal være på 128

C.5 Software kvalitetsegenskaper

C.5.1 Vises i systemtray

Når backupsystemet kjører er det unødvendig å vise at vinduet er oppe å kjøre hele tiden, og det tar også unødvendig plass på oppgavelinjen. Det kan derfor bli implementert en funksjon slik at programmet vises i systemtray når det lukkes. Da vises den ikke plass på oppgavelinjen, men vil allikevel gjøre at backup prosessen vil foregå normalt.

KR 2.4.1	**Systemtray**
Type krav	Vise ikon i systemtray
Prioritet	Middels prioritet
Funksjonelle krav	Vise et ikon i systemtray hvor man har en hurtigmeny for de mest brukte funksjoner

Tabell over systemtray

KR 2.4.2	**Nøyaktighet**
Type krav	Nøyaktighet i overføringen av data
Prioritet	Høy prioritet
Funksjonelle krav	Feil må oppdages og re-transmisjon må være mulig

Krav til re-transmisjon

C.6 Eksterne grensesnittskrav

C.6.1 Brukergrensesnitt

Brukergrensesnittet skal være veldig enkelt; man trenger ikke mye datakunnskaper for å kunne bruke applikasjonen. Grensesnittet vil bestå av et enkelt vindu med få enkle funksjoner. Meny og funksjonsvalg i grensesnittet skal være strukturert og oversiktlig. Selve backup og gjenoppretting av filer foregår i maksimum 4 steg:

Backup:

- Velg en bruker
- Velg fil(er)
- Utfør Overføring

Gjenoppretting:

- Velg en bruker
- Velg fil(er)
- Bestem et lagringssted
- Utfør gjenoppretting

C.6.2 Hardwaregrensesnitt

Systemet skal designes for å støtte Ethernet kort som følger 802 standard. Det skal ikke stilles krav til ytelsen av hardware som er installert.

C.6.3 Softwaregrensesnitt

Operativsystemet som skal benyttes skal være Microsoft Windows 98 eller nyere, og .NET 2.0 må være installert. Andre type operativsystemer støttes ikke. Det skal ikke kreves ytterlige software for at systemet skal fungere.

C.6.4 Kommunikasjonsgrensesnitt

Kommunikasjonsprotokollene som skal benyttes for overføring av data er TCP/IP. Disse protokollene skal benyttes ved overføringer av meldinger mellom server og klient, og den skal benyttes ved overføring av backup og restore. Det stilles ingen krav til filoverføringsprotokoller som for eksempel FTP.

Vedlegg D: Tidsplan