NTNU
Det skapende universitet

Bruk av mellomvare i interaktive produktkataloger

Anders H. Beite
Camilla Jacobsen
Christian Jensen

Master i datateknikk
Oppgaven levert: Juni 2006
Hovedveileder: Trond Aalberg, IDI

Norges teknisk-naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

Oppgavetekst

Teknologi er i dag i rask vekst; man kan skape løsninger som man knapt hadde drømt om tidligere. Behovet eller ønsket om å nyttiggjøre seg av de mulighetene som finnes, gjør at man kan tenke nytt og innovativt innenfor områder som marketing og eksponeringstjenester. Publisering av produktkataloger innebærer ofte mye distribusjon og printing av dokumenter. For mange organisasjoner medfører dette store kostnader knyttet til dokumenthåndtering og at man ikke når sine egentlige mål. I eksisterende løsninger på papir eller CD vil produktutvalget ofte være begrenset og kundetilpasningen dårlig.

Ved å implementere en distribuert løsning ved bruk av mellomvareteknologi, vil man kunne få en løsning som administreres sentralt. Dette vil bidra med fordeler som alltid oppdatert produktportefølje, mindre krav til klientplattform, lite eller ingen administrasjon på klientsiden, kunde-/forhandlertilpasset innhold og bedre kostnadseffektivitet. Både ved hyppige oppdateringer av porteføljen og høy grad av markedstilpasning, vil behovet for en distribuert løsning øke.

I dagens teknologijungel finnes det mange former for mellomvare til ulike formål. Noe av problemet er å vite hvilken mellomvareteknologi som egner seg til hvilket formål. Ved å velge rett vil man kunne høste frukter som økt effektivitet og forbedret brukeropplevelse.

Ved hjelp av effektivitetstester mot reelle implementasjoner og testdata skal det avgjøres hvilken mellomvareteknologi som er mest effektiv for bruk i en distribuert interaktiv produktkatalog.

Det er ønskelig å finne den teknologien som gir en best mulig løsning med tanke på effektivitet, oppdatering, dynamiskhet og brukeropplevelse.

Sentrale aspekter vil være:
- Hvilken innvirkning har båndbredden?
- Hvilken innvirkning har hardware?
- Hvor effektivt utføres oppslag og opprettelse av distribuerte objekter?
- Overføring av store vs små datamengder
- Brukergrensesnitt

Oppgaven gitt: 20. januar 2006
Hovedveileder: Trond Aalberg, IDI

Abstract

Traditional solutions for publishing product catalogues on CD or paper often involves high costs associated with distribution and printing of documents. This can cause actors to primarily focus on the handling of documents, at the sacrifice of own goals that involves creating closer affiliation between customer and product.

Today's growth within technology creates opportunities for solutions that we in the past only could dream of. The growth of technology creates a need or a wish to make use of the opportunities found within areas such as marketing and exposure services. By implementing a distributed solution with use of middleware technology you will get a solution that can be centrally administrated. This will lead to advantages such as constant updated product portfolio, less client platform requirements, little or none administration on the client side, customer-/distributor customized content and more cost efficiency. If the product portfolio is frequently updated or there are differences in the distributors market, the need for a distributed solution will increase.

There are many different types of middleware for different kinds of objectives. One problem is knowing which technology to use for an implementation. By choosing the appropriate one, you can take benefit of advantages such as higher efficiency and improved user experience. Our main goal has been to look into different middleware technologies and determine the most efficient to be used in a distributed interactive product catalogue. We have chosen .NET Remoting, Java RMI and Web Service for our implementations, and we will use unit testing tools to test the functionalities of the implementations.

Through a general evaluation of our test results and the graphical user interface, our conclusion is that .NET Remoting over TCP/BIN is the best choice for this task. Its time consumption is superior when creating distributed objects and handling XML. It also transfers a single picture faster than all the other technologies on an ADSL bandwidth. Our evaluation also shows that the .NET GUIs are faster than the ones in Java.

Sammendrag

Tradisjonelle løsninger for publisering av produktkataloger på CD eller papir innebærer ofte store kostnader knyttet til distribusjon og printing av dokumenter. Dette kan medføre at aktørene fokuserer mye på dokumenthåndteringen, på bekostning av egne målsetninger om skape nærmere tilknytning mellom kunde og produkt.

Dagens vekst innenfor teknologi skaper muligheter for løsninger man tidligere anså som *drømmetenking*. Teknologiveksten i seg selv er også med på å skape et behov eller et ønske om å dra nytte av de mulighetene som finnes innenfor områder som markedsføring og eksponeringstjenester. Ved å implementere en distribuert løsning ved bruk av mellomvareteknologi, vil man kunne få en løsning som administreres sentralt. Dette vil bidra med fordeler som alltid oppdatert produktportefølje, mindre krav til klientplattform, lite eller ingen administrasjon på klientsiden, kunde-/forhandlertilpasset innhold og bedre kostnadseffektivitet. Både ved hyppige oppdateringer av porteføljen og høy grad av markedstilpasning, vil behovet for en distribuert løsning øke.

I dagens teknologijungel finnes det mange former for mellomvare til ulike formål. Noe av problemet er å vite hvilken mellomvareteknologi som egner seg til et bestemt formål. Ved å velge rett vil man kunne høste frukter som økt effektivitet og forbedret brukeropplevelse. Vårt hovedfokus har derfor vært å se på aktuelle mellomvareteknologier som kan benyttes i en distribuert interaktiv produktkatalog, i den hensikt å avgjøre gjennom effektivitetstester hvilken teknologi som vil være mest effektiv. Vi har valgt ut .NET Remoting, Java RMI og Web Service for å benytte i implementasjoner med funksjonalitet som vi anser som sentral i en slik løsning. Vi har deretter testet funksjonaliteten i de ulike implementasjonene ved hjelp av verktøy for enhetstesting.

Gjennom en helhetlig vurdering av effektivitet basert på våre testresultater og grafisk grensesnitt konkluderer vi med at .NET Remoting over TCP/BIN er det beste valget. Den er overlegen i tidsbruk på oppretting av et objekt med XML håndtering, og henter ett bilde raskest av samtlige mellomvarer på en ADSL linje som er reelt for en distribuert produktkatalog. Samtidig viser vår evaluering av de grafiske grensesnittene at .NET brukergrensesnittene er raskere enn Java grensesnittene.

Forord

Denne oppgaven er resultatet av vår hovedoppgave, som den avsluttende delen av sivilingeniørstudiet ved Institutt for datateknikk og informasjonsvitenskap, Norges Tekniske Naturvitenskapelige Universitet, våren 2006. Oppgaven er gjennomført i samarbeid med Scarp AS, og er utført ved NTNU i Trondheim.

Vi vil rette en stor takk til veileder Trond Aalberg, som gjennom hele prosessen har kommet med gode innspill og konstruktive tilbakemeldinger.

Vi vil også takke Scarp AS for at De har gitt oss anledning til å ta del i denne spennende problemstillingen. De har også vært behjelpelig med å skaffe til veie nødvendig informasjon for å gjennomføre oppgaven.

Trondheim, 16.juni 2006

Anders H. Beite

Camilla Jacobsen

Christian Jensen

Innhold

Figurer

Kapittel 1

Innledning

Teknologi er i dag i rask vekst; man kan skape løsninger som man knapt hadde drømt om tidligere. Behovet eller ønsket om å nyttiggjøre seg av de mulighetene som finnes, gjør at man kan tenke nytt og innovativt innenfor områder som markedsføring og eksponeringstjenester.

Publisering av produktkataloger innebærer ofte mye distribusjon og printing av dokumenter. For mange organisasjoner medfører dette store kostnader knyttet til dokumenthåndtering og at man ikke når sine egentlige mål. I eksisterende løsninger på papir eller CD vil produktutvalget ofte være begrenset og kundetilpasningen dårlig.

Løsninger der informasjonsutvalget både kan styres sentralt og av sluttbrukerne selv, øyner muligheten for å tilpasse informasjon til bestemte målgrupper. Mange av dagens produktkataloger krever kundetilpassede utgaver med tilpasning av priser og betingelser. Dette må gjøres manuelt samtidig som det er ressurskrevende og medfører mulighet for feil og mangler. Vi er av den oppfatning at distribuert teknologi ved hjelp av mellomvare kan ta begrepet produktkatalog et steg videre.

Vi har i denne oppgaven implementert en reell løsning i tre ulike mellomvareteknologier, men i fem ulike oppsett. Vi har lagt ned betydelig arbeid i implementasjonene for å kunne teste med realistisk funksjonalitet og data i den hensikt å skulle kunne avgjøre hvilken mellomvareteknologi som er best egnet i en interaktiv produktkatalog.

1.1 Motivasjon

Ved å implementere en distribuert løsning ved bruk av mellomvareteknologi, vil man kunne få en løsning som administreres sentralt. Dette vil bidra med fordeler som alltid oppdatert produktportefølje, mindre krav til klientplattform, lite eller ingen administrasjon på klientsiden, kunde-/forhandlertilpasset innhold og bedre kostnadseffektivitet. Både

ved hyppige oppdateringer av porteføljen og høy grad av markedstilpasning, vil behovet for en distribuert løsning øke.

I dagens teknologijungel finnes det mange former for mellomvare til ulike formål. Noe av problemet er å vite hvilken mellomvareteknologi som egner seg til et bestemt formål. Ved å velge rett vil man kunne høste frukter som økt effektivitet og forbedret brukeropplevelse.

1.2 Hovedmål

Vårt mål er å ved hjelp av effektivitetstester mot reelle implementasjoner og testdata avgjøre hvilken mellomvareteknologi som er mest effektiv for bruk i en distribuert interaktiv produktkatalog.

1.3 Problemstilling

For sluttbruker er brukeropplevelsen alfa omega. Det nytter ikke å implementere en løsning der den gode brukeropplevelsen er fraværende, når målet er å skape nærmere tilknytning mellom kunde og produkt. Det er ønskelig å benytte den teknologien som gir en best mulig løsning med tanke på effektivitet, oppdatering, dynamiskhet og brukeropplevelse. Noen av hovedutfordringene ved distribuerte løsninger er begrensninger i båndbredde og kostnader ved hardware.

Sentrale aspekter vil være:

- Hvilken innvirkning har båndbredden?
- Hvilken innvirkning har hardware?
- Hvor effektivt utføres oppslag og opprettelse av distribuerte objekter?
- Overføring av store vs små datamengder
- Brukergrensesnitt

1.4 Rapportens oppbygning

Denne oppgaven er delt inn i tre hoveddeler, del I Bakgrunn og teori, del II Design og implementasjon og del III Resultat og konklusjon.

Del I Bakgrunn og teori I denne delen inngår kapitlene 2-4. Kapittel 2 inneholder en casebeskrivelse og en introduksjon til interaktive produktkataloger, samt utfordringer knyttet til dem. I kapittel 3 beskriver vi fremgangsmåten for oppgaven samt verktøy som vi har benyttet. Kapittel 4 baserer seg på teori funnet i litteraturen som er sentrale emner for gjennomføring av oppgaven. Dette kapittelet er også knyttet til en tidligere prosjektoppgave.

Del II Design og implementasjon Del II omhandler design og implementasjon og dekker kapitlene 5-7. Kapitlene 5-6 omhandler direkte implementasjonen som gjøres hvor kapittel 5 beskriver designen og kapittel 6 beskriver selve implementasjonen. Kapittel 7 gir en beskrivelse av testene vi skal gjennomføre, testoppsett og hvorfor vi skal teste.

Del III Resultat og konklusjon Denne delen omhandler kapitlene 8-9. Kapittel 8 gjengir resultatene som vi har oppnådd i gjennomføring av testene, og en drøfting innenfor hvert enkelt testscenario. I det siste kapittelet gir vi en konklusjon ut fra de resultatene vi har oppnådd, samt at vi beskriver videreføringsalternativene for oppgaven. Vi evaluerer også vårt arbeid i forhold til arbeid som er gjort tidligere.

Del I

Bakgrunn og teori

Kapittel 2

Interaktive produktkataloger

En typisk produktkatalog vil inneholde hele produktporteføljen til leverandøren. Produktene er ofte illustrert ved hjelp av bilder og er sammen med produktbeskrivelsen avgjørende for om kunden går til anskaffelse av produktet. Hensikten med en produktkatalog er å formidle overfor sluttbruker hvilke produkter og variasjoner som eksisterer. I mange tilfeller distribueres produkt- og priskataloger separat, der priskatalogene er unike for hver forhandler.

Dagens produktkataloger preges ofte av lav oppdateringsfrekvens og statisk innhold. Tradisjonelt sett eksisterer produktkataloger på papirformat, men har i senere tid også ekspandert til CD og DVD. Etter hvert ser vi også at det har vokst frem interaktive produktkataloger, gjerne flashbaserte , der kunden selv kan konfigurere produktet etter sine ønsker.

Interaktive produktkataloger karakteriseres som programmer eller applikasjoner som tar imot instruksjoner og gir *feedback* . Disse kjennetegnes ved at interaksjon med brukeren er i fokus, og brukergrensesnittet spiller her en sentral rolle. Variantene av interaktive produktkataloger er mange og de benytter ulike uttrykksformer i interaksjonen; tekst, tale, foto, video, musikk, lydeffekter, grafikk og animasjon. Informasjonsteknologi er den tekniske forutsetningen for slike systemer, og gjør det mulig å lagre, organisere, finne og fremvise de forskjellige elementene som utgjør systemets innhold [27].

I design av slike systemer må man ta hensyn til kommunikasjonsmessige aspekter og vi vil i avsnitt 2.2 ta for oss utfordringer som finnes innenfor eksisterende løsninger for interaktive produktkataloger.

2.1 Casebeskrivelse

Oppgaven går ut på å danne infrastrukturen for konseptet Shop in Shop for en større møbelprodusent der det endelige resultatet vil ha planlagt lansering i 2008. Som ledende i verdensmarkedet innenfor møbelindustrien, vil man søke å opprettholde markedsposisjon. Visjonen om å bringe møbelet til kunden, er et scenario som blir mer og mer utbredt blant aktører som selger denne typen produkter. Aktørene ønsker at produktet skal være oppdatert og at det formidles raskt og effektivt ut til sluttbruker for å få produktet til å virke mer attraktivt.

Vår oppgave vil ta utgangspunkt i møbelkolleksjonen til Ekornes®, som i dag distribueres på en CD som sendes ut til alle forhandlere. CD-en består av en flash- applikasjon hvor man kan velge mellom modeller for så å velge forskjellige farger og treverk for å oppnå den modellen som brukeren ønsker. CD-en er lik for alle forhandlere og inneholder derfor ikke informasjon som pris, språk etc. Dette gir begrensning i den grad at modellene er fastsatt når CD-en lages. I det øyeblikket det blir produsert en ny modell vil CD-en være utdatert og det er kostbart å produsere en ny CD for å kunne innlemme nye modeller. Enkelte modeller selges bare i et gitt marked, og forhandlerne vil da ha en løsning som inneholder modeller som ikke selges i deres spesifikke marked.

Ideen for en tenkt løsning er at en potensiell møbelkunde skal kunne sitte i et møbel fra den aktuelle produsenten og bla gjennom produktkatalogen som da vil finnes på ett eller annet medium, eksempelvis konsoll eller pc med tilhørende flatskjerm. Tanken er at man skal kunne prøvesitte et av møblene og samtidig kunne sette sammen en ønsket konfigurasjon av f.eks. farger, stofftyper og liknende, for å kunne få et visuelt inntrykk av hvordan dette vil se ut. Kunden vil dermed få god oversikt over mulige kombinasjoner som den aktuelle modellen tilbyr. Her vil man også kunne tilby ekstraprodukter eller tilbehør som kan bidra til mersalg. Produktkatalogen skal etter hvert utvides på en slik måte at man også skal kunne møblere et rom ved å velge farge og tekstil både på møbel, vegger, gulv og annet inventar.

2.1.1 Ekstern samarbeidspartner

Scarp AS i Ålesund har vært med å utforme denne oppgaven, og gitt oss tilgang til de dataene som har vært nødvendig for å realisere innholdet i produktkatalogen. Vi har fått tilgang til produktbilder gjennom Scene7 som leverer løsninger for rendring av bilder. For de bildene som benyttes i vår implementasjon vil bilder av trekk og treverk ligge på Image Server. [38]. Bilder av modellene vil ligge på en Render Server [39].

I tillegg til produktbildene har vi fått informasjon om relasjonene mellom modeller og møbler, samt tilhørende metadata som pris, beskrivelse, konfigurasjoner etc.

2.2 Utfordringer

Interaktive produktkataloger som publiseres på CD eller DVD medfører store kostnader knyttet til distribusjon av mediet, ikke bare første gang den distribueres, men også fordi den må distribueres på nytt ved endringer i produktinformasjon. Dette skaper en statisk løsning som er sårbar for feil og mangler, uten oppdateringsmuligheter. At innholdet er oppdatert og at alle tilgjengelige produkter finnes der er essensielt. Dersom oppdateringsfrekvensen på innholdet i produktkatalogen er høy vil behovet for en dynamisk løsning være stort.

En interaktiv produktkatalog baserer seg på et samspill mellom applikasjon og bruker, slik at kunden kan konfigurere produktet etter eget ønske. Slike behov krever brukervennlighet og effektivitet. I tilfeller hvor brukeren må bruke unødvendig lang tid på å få utført sine interaksjoner, vil katalogen fort virke mot sin hensikt. Dette krever et velfungerende GUI for å kunne tilfredsstille krav hvor sluttbrukerne blir stadig mer utålmodige. De vil ikke vente mange sekunder på et svar. De forlanger lynrask og mest mulig presis respons fra systemet uansett hvor stor datamengden er. Dette er en stor utfordring som applikasjonsutviklere står overfor.

Innenfor gitte segmenter som forskjellige markeder eller ulike forhandlere kan variasjonene av innholdet være stort. Ett marked kan ha en helt annen produktporteføljе enn et annet, og er dette tilfelle på forhandlernivå kan variasjonene bli mange. Behovet for å kunne tilpasse innholdet i produktkatalogen dynamisk vil etter hvert bli stort når variasjonene mellom segmentene er store.

Ved en distribuert interaktiv produktkatalog kommer det i tillegg utfordringer med begrenset båndbredde og kostnader i forhold til hardware.

Kapittel 3

Fremgangsmåte og verktøy

For å oppnå våre mål og løse problemstillingene gitt i vår oppgave, vil vi gå gjennom ulike faser som kjennetegner generell systemutvikling:

- Analyse, der man avdekker generelle funksjonalitetsbehov.
- Design, hvor man bestemmer hvordan systemet skal løse oppgavene som er spesifisert. Her vil også arkitektur–og GUI design inngå .
- Implementasjon, som vil bestå av mye koding.
- Testing av systemet.
- Resultat som innebærer evaluering av testresultater

Vi har tidligere vært gjennom en fase hvor vi så på ulike teknologier som kunne være aktuelle i en interaktiv produktkatalog. Vi har definert klare mål og problemstillinger i innledningen til denne oppgaven og vil nå ta for oss hvordan vi har tenkt å gå fram for å løse de gjenstående fasene.

Vi vil designe en basis for applikasjonene som skal utvikles. Deretter vil vi utføre selve implementasjonen og kjøre tester mot de utviklede applikasjonene. Som en avslutning vil vi sammenligne og evaluere resultatene fra testene.

3.1 Analyse

Denne fasen går ut på å avdekke generell funksjonalitetsbehov for de mulige teknologiene. Funksjonaliteten dekkes gjennom casebeskrivelsen i avsnitt 2.1. Som teknologivalg har vi valgt å ta for oss Java RMI og .NET Remoting som sies å inneha 40% av markedet hver innenfor distribuerte teknologier[35]. I tillegg har vi valgt å implementere en løsning basert på Web Services, som er en teknologi i sterk vekst. Bakgrunnen for disse valgene baserer vi på vårt forstudium høsten 2005 [4].

3.2 Design

Vi var ut fra vårt samarbeid med Scarp ikke låst til noen eksisterende løsning eller plattform for eksisterende data, men vi ser at behovet for å ha en god struktur på data og metadata er viktig. Dette letter arbeidet med implementasjon, og de forskjellige implementasjonene blir tilnærmet identiske. På den måten vil grunnlaget for å kunne sammenligne resultatene bli så likt som mulig. Vi vil legge vekt på at data og metadata kan hentes ut, eller omformes slik at de kan hentes ut, uavhengig av utviklingsplattformen til implementasjonen.

I designet av klasser som skal brukes i implementasjon vil vi legge vekt på at dette gjøres slik at hovedfunksjonaliteten for en produktkatalog er dekket.

3.3 Implementasjon

Vi vil utvikle en tjenerapplikasjon og en klient med grafisk grensesnitt for hver av de forskjellige mellomvareteknologiene vi har valgt ut.

Innenfor .NET Remoting har vi valgt å skrive programkoden i C# og benytte .NET Framework 2.0 . Implementasjonen vil kunne bruke to måter å overføre data på. Den ene er binær formatering over TCP protokollen, og den andre er SOAP formatering over HTTP protokollen. Implementasjonen av .NET applikasjoner finnes i kapittel 6.7

For Java implementasjonen bruker vi Java RMI som har sin egen protokoll for overføring av data. Vi har valgt å implementere støtte både for Windows og Linux plattform. Implementasjonen av javaapplikasjonen finnes i kapittel 6.8

Web Service implementasjonen har vi også valgt å skrive i C# for .NET 2.0. Servicen skal kjøres på Microsofts *Internet Information Services* (IIS) webtjener. Implementasjonen av applikasjonen ved hjelp av Web Services teknologi finnes i kapittel 6.9

3.3.1 Utviklingsmetodikk

Med begrepet metodikk mener vi i denne sammenhengen hva som ligger til grunn når vi foretar våre valg under selve utvikling av applikasjonene. Det er et behov for å ha retningslinjer som kan guide oss gjennom utviklingsprosessen, som ofte vil være kompleks.

Vi har kjennskap til XP (eXtreme Programming) som utviklingsmetodikk og har benyttet de prinsippene i XP som vil være fordelaktig for å kunne lykkes i vårt utviklingsprosjekt. Som basis er der 12 verdier/prinsipper[21], vi har valgt følgende:

- Små utgivelser
  Vi vil starte i det små og implementere den enkleste funksjonaliteten først. Publisere tidlig og ofte, og legge til ny funksjonalitet hver gang.

- Enkel design
  Vi vil alltid benytte enklest mulig design som kreves får å få jobben gjort. Kravene vil hele tiden endres, så vi vil implementere det som trengs for å tilfredsstille kravene til ethvert tidspunkt.

- Fortløpende testing

  - Enhetstesting og funksjonelle tester

- Parprogrammering
  Intensjonen er at tilnærmet all kode skal bli skrevet av alle samtidig. På denne måten vil koden bli gjennomgått mens den blir skrevet. Vi vil i tillegg benytte oss av Visual SourceSafe for versjonshåndtering innenfor .NET. Se avsnitt 3.6.2 på side 14

- Standarder
  Det vil bli viktig å få lagt på plass standarder for blant annet lagring av data og metadata. Se avsnitt 5.3.1 på side 38

3.4 Tester

Vi vil bruke rammeverk for enhetstesting. Dette vil gi oss mulighet til å lage automatiserte tester slik at de blir tilnærmet identiske for alle implementasjoner, og vi kan bruke testdata til å sammenligne implementasjonene. Vi vil hovedsaklig bruke enhetstestene til å teste på effektivitet og tidsbruk; hvor lang tid systemet bruker på å utføre viktig funksjonalitet, og hvor mye data må overføres for å få det gjort. Til dette har vi valgt oss ut to testrammeverk NUnit (kapittel 7.2.1) og JUnit (kapittel 7.2.2).

Selve brukeropplevelsen og det grafiske grensesnittet finnes det ikke noe fullgodt enhetstestingsverktøy for. Vi vil derfor gjøre en egen vurdering på hvor godt de forskjellige mellomvareløsningene kan støttes av grafiske grensesnitt. Du finner mer om testene i kapittel 7 på side 55

3.5 Resultat

Resultatene fra testene vi gjennomfører vil vi bruke til å sammenligne mellomvareteknologiene opp mot hverandre. Vi vil lage en grafisk fremstilling ved hjelp av diagrammer fra testresultatene. Vi håper da å kunne finne svar på hvilken av teknologiene som passer best til en produktkatalog basert på distribuert teknologi ved hjelp av mellomvare.

3.6 Utviklingsverktøy

3.6.1 Visual Studio 2005

Utviklingsmiljø med blant annet database- og applikasjonsutvikling. .NET Framework 2.0. Støtter programmeringsspråkene Visual C#, Visual Basic, Visual J# og Visual C++, i tillegg til Visual Web Developer. Vi har valgt å benytte oss av språket C# og har utviklet .NET Remoting og Web Service applikasjonene i dette språket.

3.6.2 Visual SourceSafe 2005

Versjonskontroll for Visual Studio .NET som gjør det enkelt å ta vare på viktige filer. Man kan dele filer mellom flere brukere uavhengig av hva slags filtype det er snakk om. Alle filene blir lagt i en database for å gjøre det enklere å finne tilbake til tidligere versjoner av filen som er lagret, og når en fil trengs å deles mellom flere brukere. Det er også mulig å gjenopprette en gammel fil.

3.6.3 Eclipse versjon 3.1.1

Eclipse er et program som gir et IDE (Integrated Development Environment.) til å utvikle f. eks javaprogrammer. Tanken bak IDE er å gi utviklere et program som gir full oversikt over programmeringsprosessen. Andre IDE programmer er f. eks NetBeans, JavaBuilder, etc. (Eclipse kan også brukes til å utvikle programmer i andre språk enn Java). [44] Vi har benyttet Eclipse til å utvikle RMI applikasjonen, og har også hatt god nytte av plugins som RMIPlugin og EclipseUMLPlugin.

3.6.4 Altova XMLSpy 2005

XML- editor for å redigere og manipulere XML- dokumenter. [2]

3.6.5 MS SQL Server 2005

Databaseserver med støtte for XML- håndtering som vi har valgt å benytte oss av i denne oppgaven.

3.6.6 Testrammeverk

Vi har valgt å benytte oss av NUnit og JUnit for enhetstesting. Se kapittel 7.2 på side 57

3.6.7 Commview

CommView er blitt benyttet til å registrere hvor mange pakker og bytes som blir overført mellom tjener og klient.

Kapittel 4

Teori

Dette kapittelet vil vi bruke til å danne et teoretisk grunnlag for det videre arbeidet i oppgaven. Her tar vi for oss temaer som omhandler distribuerte systemer, mellomvare, teknologispesifikke emner som .NET Remoting, Java RMI, Web Service og XML.

4.1 Om distribuerte systemer

4.1.1 Historie

Distribuert programmering begynte rundt 1970 med sammenslåing av to teknologier:

1. Minidatamaskiner, som blir arbeidsstasjoner, deretter PC-er [Figur 4.1]
2. Nettverk (senere Ethernet og internett)

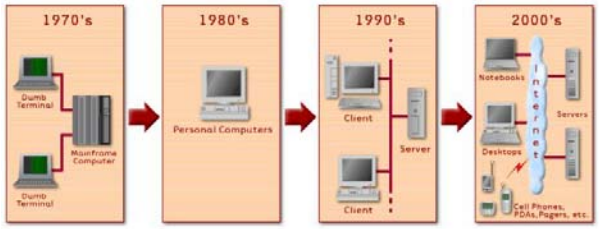
Problemet med minidatamaskiner var at de ikke var like skalerbar og kunne håndtere like mange brukere som en mainframe maskin kunne. Måten de løste dette problemet på var å kjøpe flere minidatamaskiner.

Arbeidet med den første arbeidsstasjonen begynte i 1970 hos Xerox PARC ved Palo Alto Research Center (PARC) og ble kalt Alto. Over de 10 neste årene ville PARC være ledende innenfor utvikling av teknologi til arbeidsstasjoner og PC-er som vi bruker i dag.

En type distribuerte programmering som blir mer og mer viktig i dag er stor skala distribusjonen som er gjort mulig ved hjelp av internett.

Omtrent samtidig som arbeidsstasjoner og lokalt nettverk ble oppfunnet, brukte forsvaret i USA store mengder tid og penger for å sette opp et kommunikasjonssystem for å støtte distribuert vitenskap og forskning. De var svært opptatt av at systemene ikke skulle være sentralisert ettersom de da var svært sårbar for angrep, og ville derfor ha et

distribuert system. Som et resultat av dette kom ARPAnet til verden, som senere ble til internett som vi kjenner det i dag. [6]



Figur 4.1: Utviklingen av datamaskiner [3]

4.1.2 Hva er et distribuert system

Et distribuert system består av maskinvare- og programvarekomponenter lokalisert i et nettverk av datamaskiner som kommuniserer sine handlinger ved å sende meldinger [13] . Systemet jobber sammen slik at det for klienten virker som ett system.

For å få en bedre fremstilling av hvordan et distribuert system er sammensatt kan vi illustrere dette ved følgende lagdeling [Tabell 4.1], hvor de to siste lagene utgjør systemets plattform.

Applikasjoner, tjenester
Mellomvare
Operativsystem
Maskinvare, nettverk og datamaskiner

Tabell 4.1: Lagdeling i distribuerte systemer

Et distribuert system kan bygges opp på mange ulike måter og består av flere separate sammenkoblede systemer. Systemkonfigurasjonene kan avvike mye fra hverandre og være meget komplekse [45].

- Klient-tjener arkitektur
  En klient oppretter kontakt med en tjener(som oftest på en annen maskin) for å få utført en bestemt funksjon.

- 3-lags arkitektur
  Klientlogikken flyttes til et mellomlag slik at tilstandsløse klienter kan benyttes.

- N-lags arkitektur
  Typiske webapplikasjoner som videresender til andre lag.
- Tett koblet arkitektur
  Maskinene kjører samme prosess parallelt, med deler oppgavene i biter for så å sette de sammen til det endelige resultatet.
- Peer-to-peer
  Arkitektur der ikke én gitt maskin(er) tilbyr en tjeneste, men ansvarsforholdet deles likt over alle frittstående maskiner.

4.1.3 Hvorfor bruke distribuerte systemer?

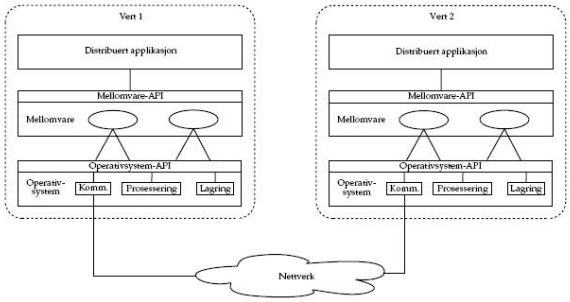
Krav som ressursdeling, åpenhet, skalerbarhet, feiltoleranse og heterogenitet kan tilfredsstilles av distribuerte systemer [13].

- *Ressursdeling*
  Gir mulighet for å benytte tilgjengelige ressurser hvor som helst.
- *Åpenhet*
  Et åpent system kan utvides og forbedres inkrementelt.
- *Skalerbarhet*
  Det er vanskelig å forutsi hvordan et system vil utvikle seg over tid. Dette gjelder antall brukere, datamengde, maskinvare og funksjonalitet. Derfor er det viktig å bruke en plattform og arkitektur som lar seg skalere uten å endre for mye av koden. Viktige faktorer ved skalerbarhet er pålitelighet, tilgjengelighet og hvor lett det er å administrere systemet.
- *Feiltoleranse*
  Det må vurderes hvor tolerant systemet skal være ovenfor feil, men det er viktig å kunne opprettholde tilgjengelighet selv i tilfeller der komponenter har liten pålitelighet.
- *Heterogenitet*
  Man kan implementere ulike maskinvare, operativsystem og programmeringsspråk.

Hovedårsaken til å konstruere et distribuert system er gjenbruk og deling av ressurser samt for å begrense virkningene om en del av systemet skulle gå ned. Ressursene blir håndtert av tjeneren og er tilgjengelig for brukerne eller de kan innkapsles som objekt som andre brukerobjekt kan aksessere.

4.2 Mellomvare

ObjectWeb consortium gir følgende definisjon på mellomvare: *"I et distribuert system, er mellomvare definert som programvarelaget som ligger mellom operativsystemet og applikasjonene på hver side av et system"*.[46]



Figur 4.2: Mellomvare [33]

Mellomvare er en programvareteknologi som leverer data mellom forskjellige systemer.[Figur 4.2] Kompleksiteten i mellomvare er som regel relatert til kompleksiteten mellom system. I et enkelt system, én enkel applikasjon programmert i ett språk, kan mellomvaren være ubetydelig fra systemfunksjonene. I et slikt enkelt system er mellomvaren bare et abstrakt konsept som binder deler av systemet som datastrukturer eller objekter som er referert til flere punkter innenfor systemet. I et komplekst system, blir mellomvaren en nødvendig programvarekomponent som ofte er uavhengig av applikasjonskomponentene som sammen utgjør systemet. I et slikt scenario har mellomvaren fremdeles fokus på å gjøre dataobjekter tilgjengelig gjennom systemet slik at alle delene av et system kan jobbe sammen.[34]

Mellomvare blir i dag brukt til å beskrive webtjenere, applikasjonstjenere, CMS og andre verktøy som støtter applikasjonsutvikling. Mellomvare i dag er spesielt tilpasset informasjon basert på XML, SOAP, Web Services og tjenesteorientert arkitektur. Mellomvare er en teknologi som muliggjør applikasjonsintegrasjon.

Det finnes ulike former for mellomvare [4]

- Database mellomvare som Oracle SQLNet.
- Distribuerte TP monitor system som Tuxedo.
- Meldingsorientert mellomvare som IBM MQ Series for å utføre asynkron kommunikasjon med garantert meldingslevering.
- RPC systemer som DCE for å utføre synkron kommunikasjon mellom prosedurale (ikke objektorienterte) systemer.
- ORB systemer som CORBA, JAVA og .NET for å utføre synkron kommunikasjon mellom objekt (eller komponent) baserte systemer.

- Web mellomvare, dvs. webtjenere, ASP og servlet teknologi for å presentere dynamisk genererte HTML sider.
- OTM Applikasjonstjenere som tilbyr infrastruktur for kjøring"av komponenter. Eksempler er MS MTS, IBM WebSphere og GemStone.

Det er mellomvare av typen ORB som skal evalueres videre i oppgaven.

Objektorientert mellomvare er bygd på konseptet å kalle på en operasjon i et objekt som befinner seg på et annet system. Istedenfor et klient-tjener forhold, er det et klient-objekt forhold. For å kunne nå et objekt på et annet system, må et program ha en referanse som peker på objektet. Programmerere er vant til å kalle lokale objekt, og kompleksiteten ved å kalle et objekt på et annet system blir skjult i mellomvaren for å gjøre kallene så enkle som mulig for programmereren. Ulikt fra tidligere variasjoner av mellomvare, kreves det kun to steg for å kalle en operasjon på et objekt på et annet system: Få referansen til et objekt og så kalle en operasjon på objektet. Når du først har referansen kan du kalle objektet når det er nødvendig.[5]

For å kunne hanskes med denne kompleksiteten og heterogeniteten er det essensielt å benytte seg av mellomvare når man utvikler en arkitektur for et distribuert system. Mellomvare isolerer applikasjonen fra lavnivå protokoller og infrastruktur, og gjør systemet mer fleksibelt og tilpasningsdyktig.[33]

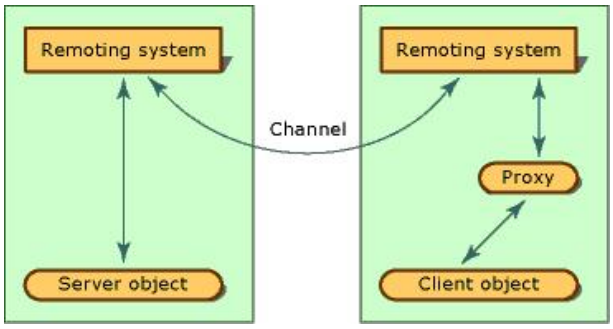
4.3 .NET Remoting

.NET Remoting er en teknologi for å utvikle distribuerte applikasjoner. .NET Remoting muliggjør kommunikasjon mellom applikasjoner og er et felles system som applikasjoner kan bruke for å kommunisere med hverandre.

.NET objekter blir tilgjengeliggjort for fjerntliggende prosesser og tillater dermed kommunikasjon med fjerntliggende applikasjoner. Applikasjonen kan være plassert på samme maskinen, andre maskiner i samme nettverk eller andre maskiner i andre nettverk. Objekter kan bli overført som en kopi av det originale objektet, eller at klienten mottar en referanse til objektet som befinner seg hos tjeneren. Ved å benytte en referanse kan klienten utnytte seg av deler av objektet uten at det er nødvendig å overføre hele objektet over til klienten.

For å utveksle informasjon mellom klienten og tjeneren benytter .NET Remoting seg av kanaler[Figur 4.3]. Kanalen tar streamen av data som skal utveksles, oppretter en pakke i henhold til nettverksprotokollen som er valgt og sender pakken til mottakeren. Noen kanaler kan bare sende data og noen kan bare motta, men enkelte kanaler som TcpChannel og HttpChannel kan både motta og sende data og brukes derfor som standard i .NET Remoting. Som kanalnavnene i seg selv sier, TcpChannel bruker Tcp protokollen til overføring og HttpChannel bruker Http protokollen til overføring. Innenfor hver av kanalene

har også .NET Remoting fordelen av å kunne velge hvilken formateringstype som skal brukes på pakkene som sendes. Standard i .NET Remoting er XML-SOAP og binær formatering. XML-SOAP har et menneskelesbart format samtidig som det er en åpen standard, men til gjengjeld krever XML-SOAP mye overhead i pakkene som sendes og krever mer ut av tjeneren til selve formateringen av pakkene. Binær formatering derimot krever mindre ut av tjeneren på selve formateringen og har mindre overhead enn XML-SOAP.



Figur 4.3: .NET Remoting prosessen [25]

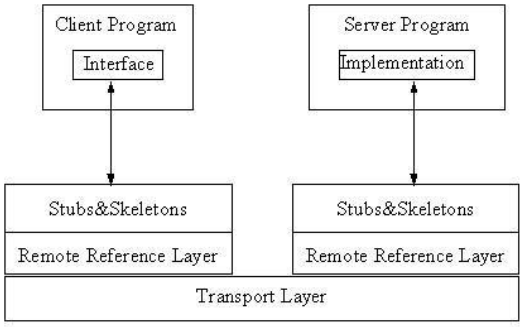
.NET Remoting kan utvikles i flere forskjellige språk men ettersom det er en Microsoft teknologi kan applikasjonene kun kjøres på en Windows plattform [32].

4.4 Java RMI

Java Remote Method Invocation (Java RMI) gjør det mulig for en prosess å kalle metoder på et objekt som er opprettet av en annen prosess. Denne prosessen kan være på samme maskin, andre maskiner i samme nettverk eller andre maskiner i andre nettverk. Java RMI kan kun utvikles i ett språk, men er en multiplattform teknologi. [29]

Java RMI består av tre lag[Figur 4.4]:

Det første laget er Stub/Skeleton-laget. Dette laget håndterer grensesnittet mellom klient og tjener. Med klient menes her prosessen som ønsker å kalle en metode, og tjeneren er prosessen som opprettet objektet metoden skal kalles på. En stub er den lokale representanten til et fjerntliggende objekt. Når det kalles på den lokale stuben har den ansvaret for å videreformidle kallet til det fjerntliggende objektet og motta svar. Et hvert fjerntliggende objekt har et korresponderende Skeleton som videre formidler kall til riktig metode. Når et skeleton mottar et innkommende metodekall leser den parametrene, kaller på metoden for den reelle objekt implementasjonen, og sender svar tilbake.



Figur 4.4: Java RMI arkitektur [48]

Det andre laget kalles Remote Reference Layer (RRL). Dette laget holder rede på hvor forskjellige Remote-objekter befinner seg, og tar seg av en del annen kommunikasjon som har med multithreading og garbage-collection for Remote-objekter å gjøre.

Det tredje laget er transportlaget. Det er i dette laget at informasjon mellom klient og tjener faktisk overføres. Dette gjøres ved hjelp av TCP/IP-forbindelser. RMI transportlaget kommuniserer gjennom åpne og direkte sockets til verter over Internett. Mange brannmurer på intranett har få åpne porter, og for at en RMI-applikasjon skal kunne kommunisere over nettverk, tilbyr RMI to alternative HTTP-baserte mekanismer som tillater en klient på baksiden av en brannmur til å kalle et fjerntliggende objekt utenfor brannmuren.

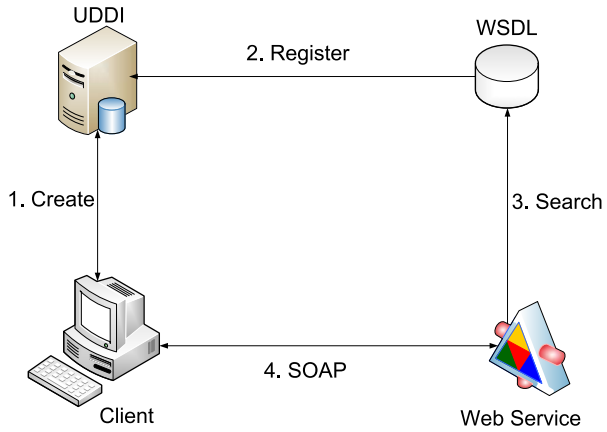
RMI applikasjoner innebefatter ofte to separate program, en tjener og en klient. En typisk tjenerapplikasjon oppretter et antall fjerntliggende objekter, gjør referansene til disse objektene tilgjengelig og venter på klienter til å påkalle disse metodene på de fjerntliggende objektene. En typisk klientapplikasjon får en fjerntliggende referanse til en eller flere fjerntliggende objekter på tjeneren og påkaller metodene på dem. RMI tilbyr mekanismen som tjeneren og klienten kommuniserer mellom.

Et argument til, eller en returverdi fra, et fjerntliggende objekt kan være hvilket som helst objekt som er *serializable (se 4.8)*. Dette inkluderer primitive typer, fjerntliggende objekter, og lokale objekter som implementerer serializable grensesnittet.

Når et lokalt objekt er sendt som et argument eller returverdi i en remote method invocation, blir innholdet av det lokale objektet kopiert før gjennomføringen av kallet til det fjerntliggende objektet. Når et lokalt objekt blir returnert fra en remote method invocation, blir et nytt objekt opprettet i den kallende virtual machine [4].

4.5 Web Services

Web Services kan betraktes som *internett mellomvare*. Det kan beskrives som en applikasjonsservice som kan aksesseres ved bruk av standard webprotokoller som er uavhengig av utvikler, plattform og utviklingsspråk. Innkapslede, løst koblede og forkortede funksjoner som tilbys over standard protokoller [4].



Figur 4.5: Web Service [42]

Kjernekonsepter [Figur 4.5]:

- Aksesserbart over internett ved hjelp av plattformuavhengige webprotokoller
- En definert kommunikasjonsprotokoll og interface som kan påkalles fra en klient eller tilbys av en tjener
- Web Services Definition Language (WSDL) som ligger som et lag mellom implementasjon og interface

Den underliggende transportprotokollen bak Web Services er bygget rundt de åpne standardene HTTP og XML.

De tre største bidragsyterne er W3C (World Wide Web Consortium), OASIS og WS-I (Web Services Interoperability Organization). Web Services er stateless og connectionless . Det vil si at man ikke holder styring på tidligere handlinger slik at hver handling baseres bare på den informasjonen den har tilgjengelig for øyeblikket. Man må heller ikke opprette noen form for kontakt med en gitt prosess. Mottakerapplikasjonen handler ut i fra en request og svarer på denne om nødvendig. Kommunikasjonen er avhengig av Remote Procedure Call (RPC) der forespørsler og svar utveksles ved hjelp av XML over HTTP [4].

Web Service bruker SOAP formatering på utveksling av beskjeder. SOAP er en utvidbar XML messaging protokoll som tilbyr en applikasjon å sende XML beskjeder til en annen applikasjon. En hvilken som helst applikasjon kan delta i kommunikasjonen som enten SOAP sender eller SOAP mottaker [41]. SOAP ble først utviklet av DevelopMentor, Microsoft og UserLand som en Microsoftspesifikk XML- basert RPC protokoll i 1999. Tidlig i 2000 deltok Lotus og IBM for å lage en åpen og utvidbar versjon, som var både språk- og plattformuavhengig. Denne ble kalt SOAP 1.1 og ble senere standardisert av W3C.

WSDL er et XML vokabular for å beskrive en Web Service. Et WSDL-dokument beskriver den funksjonaliteten som en Web Service tilbyr, hvordan den kommuniserer og hvor den er tilgjengelig. WSDL har en strukturert mekanisme for å beskrive operasjonene en Web Service kan tilby, formatet på meldingene den kan prosessere, protokollene den støtter og hvor den er tilgjengelig.

Et WSDL- dokument bruker følgende elementer for beskrivelse av nettverksservicer:

- Types
- Messages
- Operation
- Port Type
- Binding
- Port
- Service

4.6 Mobile klienter

Mobile klienter blir mer og mer aktuelt i dag samtidig som vi er avhengig av høybåndbredde. Mellomvare tar ikke hensyn til lav båndbredde men det forskes på å komprimere data og gjøre uoppnåelighet normalt istedenfor eksepsjonelt. [1] Ved caching av produktkataloger kan mobile klienter dra nytten av å koble seg opp mot tjeneren for deretter å oppdatere produktkatalogen for så å koble seg av igjen. Den mobile klienten kan deretter brukes når som helst med den informasjonen som den allerede har hentet fra tjeneren. Ved bruk av mellomvare kan man også utnytte at lokasjonen til klienten er ubetydelig. Den mobile klienten kan dermed koble seg til tjeneren hvor som helst den har internett tilgjengelig for så å oppdatere produktkatalogen sin fra tjeneren. Dette gir nye muligheter hvor en salgsperson kan ta med seg en mobil klient til en kunde, uten å være avhengig av internett. [32]

4.7 XML

XML står for eXtensible Markup Language. XML ble utviklet av W3C i 1996 og anbefalt fra 1998. Man tok utgangspunkt i SGML og brukte samtidig de erfaringene man hadde gjort seg med HTML. XML ble utviklet for å beskrive data og for å fokusere på hva data er. XML skal lagre, bære og utveksle informasjon, og skal bedre funksjonaliteten av web gjennom å tilby mer fleksibel og anvendelig identifisering av informasjon. XML er betegnet som et metaspråk,- et språk som beskriver andre språk. Dette kan XML gjøre fordi det er skrevet i SGML. Mens HTML er en dokumenttype laget etter regler fra SGML, er XML er en forenkelt utgave av SGML med mulighet til å lage egne dokumenttyper.

XML gjør ingen operasjon, det er kun ren informasjon pakket i XML-tagger. Og her ligger XMLs styrke; man kan utvikle sitt eget Markup Language som en så kan bruke til å formattere egne dokumenter. Markup Language vil inneholde tagger som beskriver de data det inneholder. Ved å bruke XML kan man forstå eller kode informasjonen i dokumenter på en langt mer presis måte enn dokumenter skrevet i HTML. Informasjon skrevet i XML kan være rikere og lettere å benytte fordi de beskrivende egenskaper og evnen til å lenke hypertekst er bedre. At XML er et metaspråk, som kan brukes til å lage mange språk, gjør XML nøytralt. [12]

4.8 Serialisering

Serialisering er prosessen å lagre et objekt til et lagringsmedium eller å sende det over en nettverksforbindelse som en serie av bytes eller i et mer menneskeliglesbart format som XML. Bytestrømmen, eller den typen format som brukes til overføringen, brukes til å gjenopprette objektet hos mottakeren. Denne typen serialisering er mest brukt til distribusjon av identiske objekter til flere applikasjoner. Prosessen å serialisere objekter er også kjent som *marshalling*.

Java tilbyr automatisk serialisering og krever kun at objektet merkes som *Serializable*, serialiseringen håndteres da internt i Java.

.NET tilbyr tre formater for serialisering av objekter; *binary*, *SOAP* og *XML*.[43] Hvor av de to første er godt egnet når man skal transportere objekter på tvers av nettverk. SOAP er ideelt for å sende via HTTP, mens binary er optimalisert til TCP protokollen.

Marshalling i .NET kan gjøres på to måter; *marshal–by–ref* (MBR) og *marshal–by–value* (MBV). [28] Ved MBR har klienten en referanse til objektet som befinner seg på tjeneren. Dersom tjeneren går ned vil ikke klienten lenger kunne benytte seg av objektet. Ved MBV opprettes objektet på tjeneren og kopieres over til klienten. På denne måten kan klienten benytte seg av objektet selv om tjeneren skulle gå ned.

4.9 Caching

Caching brukes for å oppnå forskjellige mål. Caching benyttes enten for å øke effektiviteten ved lesing fra minnet, eller brukes for å holde på data som er ønskelige å gjenbruke.

I en interaktiv produktkatalog er det ønskelig å benytte seg av begge egenskapene til caching. For at det ikke skal være nødvendig for klienten å hente objekter hver gang de blir forespurt, blir objektene lagret i cachen etter hvert som de kommer inn til klienten. Ved denne måten oppnår klienten også den andre egenskapen til caching som er effektivitet. Ettersom objektene ligger i cachen vil klienten hente objektene raskere og dermed vise dem raskere for brukeren.

Et scenario som kan oppstå er at nettilgangen er nede og det ikke oppnås kontakt med tjener. Caching- og oppdateringsmekanismer må opprettes med hensyn på hvor viktig det er at applikasjonen alltid er operatibel i forhold til hvor enkel klienten skal være og effektivitet ved bruk av applikasjonen.

Om det skal gjøres endringer i henhold til selve applikasjonen må det også taes hensyn til hvordan selve applikasjon kan/skal oppdateres samt hva som skal gjøres dersom URI til tjener endres eller at tjeneren endrer plassering.

Det som må taes hensyn til både ved caching- og oppdateringsmekanismene er responstid fra tjener. Båndbredden vil ha en liten betydning om det er en uakseptabel lang responstid fra tjeneren, også kjent som pingtid. Mekanismene må også ta hensyn til overføringstid.

4.9.1 Klienthåndtering

Ettersom cachingemnet omhandler en stor mengde informasjon vil vi he ta for oss de caching typene som vil være aktuell for oppgaven.

Filer lokalt vs filer hos tjener

Ressurs filene som brukes i en distribuert interaktiv produkt katalog kan håndteres på forskjellige måter.

Filene befinner seg hos tjeneren og klienten forespør hele filen ved et kall og lagrer så filen lokalt for bruk.

I dette tilfellet er en oppdateringsmekanisme enkel. Klienten forespør tjeneren hvilken dato han har på den aktuelle filen, hvis det er en nyere dato enn den som klienten har selv må klienten oppdatere.

Fordel:

- Applikasjonen vil være fullstendig operatibel med de filene den har lagret selv om de ikke er av nyeste versjon. Dette er med en bemerkelse på at klienten har alle filene som er nødvendig for systemet, men klienten vil være delvis operatibel.
- Om det ikke eksisterer en ny versjon av forespurt fil unngår klienten overføring og dermed liten ventetid før applikasjonen er oppdatert.

Ulemper:

- Data som overføres er store mengder ettersom hele filen blir etterspurt og ikke bare ønsket informasjon. Noe som ofte vil føre til unødvendig lang overføringstid. Selv om bare deler av filen er oppdatert vil hele den nye filen overføres
- Klienter med applikasjonen kan ha begrenset tilgang til filsystem som kan føre til problemer ved lagring av nye filer.
- Utenforstående har enklere tilgang til lokale filer for manipulering av filene.

Filene kan kun befinne seg hos tjeneren og klienten forspør hele filen ved et kall men beholder filen i minnet under kjøring

Klienten forespør tjeneren hvilken dato han har på den aktuelle filen, klienten må selv ha lagret en dato for når han mottok sin egen informasjon, hvis tjeneren har en nyere dato enn den som klienten har selv, må klienten oppdatere.

Fordel:

- Unngår problemer med lagring til disk
- Mer effektiv bruk ved lesing fra minnet
- Ingen utenforstående kan endre informasjonen like enkelt som når filene ligger lagret på disk.

Ulemper:

- Etter flere forespørsler vil minnebruken bli meget høy og man vil oppnå problemer når minnet blir fullt. Det kan da opprettes en funksjon for å sette en maks begrensing på minne bruken. Filene kan bli tømt fra minnet men må da hentes på nytt neste gang de blir forespurt.
- Selv om bare deler av filen er oppdatert vil hele den nye filen overføres
- Hvis nettverket er nede vil det ikke bli mulig å oppdatere informasjonen i minnet. Hvis ønsket informasjon ikke finnes i minnet, vil forspurt funksjonalitet ikke være tilgjengelig. Det er større sannsynlighet at noe ikke eksisterer i minnet enn i forhold til lokal disk under de samme forutsetningene.

- Ved en omstart av applikasjonen vil filene bli tømt fra minnet og alle forespørsler må gjøres på nytt.

Filen kan kun befinne seg hos tjeneren og klienten forespør kun spesifikke deler av filen for så å holde kun ønsket informasjon i minnet

Klienten forespør tjeneren hvilken dato han har på den aktuelle filen, klienten må selv ha lagret en dato for når han mottok sin egen informasjon, hvis tjeneren har en nyere dato enn den som klienten har selv, må klienten oppdatere selv om det ikke er nødvendig at akkurat den informasjonen som klienten har fått er den informasjonen som er oppdatert.

Fordeler:

- Mer effektiv bruk ved lesing fra minnet
- Unngår problemer med lagring til disk
- Ingen utenforstående kan endre informasjonen like enkelt som når filene ligger lagret på disk
- Minnet vil ikke bli like fort fullt som når hele filene blir lest.

Ulemper:

- Hyppige forespørsler.
- Ved lang responstid fra tjener vil applikasjonen oppnå unødvendig lang tid for oppdatering.
- Minnet vil bli fullt og det vil være nødvendig å fjerne informasjon fra minnet. Det vil da være nødvendig med et nytt kall til tjeneren.
- Hvis nettverket er nede vil det ikke bli mulig å oppdatere informasjonen i minnet. Hvis ønsket informasjon ikke finnes i minnet, vil forespurt funksjonalitet ikke være tilgjengelig. Det er større sannsynlighet at noe ikke eksisterer i minnet enn i forhold til lokal disk under de samme forutsetningene.
- Ved en omstart av applikasjonen vil informasjon bli tømt fra minnet og alle forespørsler må gjøres på nytt.

4.9.2 Tjenerhåndtering

Filen(e) leses fra disk hver gang de blir forespurt.

Fordeler:

- Lite minnebruk

- Alltid siste versjon av filen

Ulemper:

- Mye lesing fra lokal disk som gir ineffektivitet

Filen(e) leses fra disk første gang og blir deretter beholdt i cache etter visse kriterier som hvor ofte den skal være forespurt før den blir tømt fra cache, eller en tidsbegrenset levetid.

Fordeler:

- Bedre effektivitet ved lesing fra minnet

Ulemper:

- Kan ikke være sikker på at det er siste versjon av filen som er i bruk. Kan opprette mekanisme for å sjekke datoen på den lokale filen mot når filen ble lastet til minnet for å så oppdatere minnet om det er kommet en ny versjon, men vil igjen kreve ekstra last på tjeneren.
- Høyt minnebruk

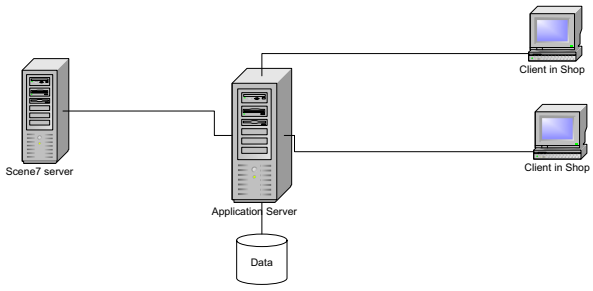
Del II

Design og implementasjon

Kapittel 5

Design

5.1 Systemarkitektur



Figur 5.1: Overordnet systemarkitektur

Vi vil lage en tjenerapplikasjon som vil holde på all informasjon om de forskjellige produktene. Når en klient kobler seg til tjenerapplikasjonen vil den få oppdatert informasjon om hvilke produkter som finnes i den aktuelle forhandlers portefølje, og opprette nødvendige objekter til klientapplikasjonen. All denne informasjonen hentes fra en XML-fil som er spesifikk for hver enkelt forhandler. Etterhvert som klientapplikasjonen brukes vil den hente informasjon den ikke er i besittelse av fra tjenerapplikasjonen. Bilder av møblene rendres av Scene7-tjeneren fra forespørsler ut fra hvilke treverk og trekk som er ønsket på møbelet [Figur 5.1].

5.2 Lagring og strukturering av metadata

Vi har valgt å lage et fundament som skal holde på og strukturere data og metadata. Valget falt på å lage en basis ut fra XML filer. XML er en åpen standard og har den fordelen at det ikke er plattform- eller applikasjonsavhengig, og det finnes gode muligheter for å

portere informasjon fra mange medium til nettopp XML. Strukturen i XML passet oss godt i forhold til hvordan vi ville strukturere informasjonen, og tilbyr den rette miksen av fleksibilitet og struktur. Med XML kan man definere egne elementer, som både kan inneholde tekst og andre elementer. Dette gjør at man kan definere elementer med en hierarkisk oppbygning. I tillegg finnes det XML-støtte i alle mellomvareteknologiene vi skal benytte oss av. Det vil si at alle applikasjoner bruker samme utgangspunkt for å hente data.

For å få en bedre oversikt over nødvendige data og relasjonene mellom dem, la vi først inn testdataene våre i en relasjonsdatabase , for så å portere de videre til XML-filer. Vi fikk da også testet hvordan dette lar seg gjøre fra f.eks. en database, ettersom dette kan være relevant for tilsvarende løsninger som allerede er i besittelse av dataene på et annet format en XML.

5.2.1 Databasen med testdata

Våre applikasjoner er ikke på noen måte avhengig av databasen vi har designet, ettersom vi har valgt å bruke XML som lagringsformat for våre metadata, men den har gitt oss stor nytteverdi og forenkelt arbeidet med å opprette XML- filer. Vi velger derfor å gi en kort fremstilling av databasen og dens viktigste tabeller og relasjoner [Figur 5.2] med tanke på relevans som tidligere nevnt.

- Møbel:
  Dette er hvert enkelt møbel (sofa,stol,stressless etc.) liknende. Hvert enkelt møbel har bredde–, høyde–og dybdemål, samt et unikt modellnummer.

- Modell:
  Hvert møbel hører inn under en modellserie som kan være en samling av forskjellige møbler. Modellen angir også størrelsen på møblene i serien; om de er tilpasset store eller små personer.

- Kategori:
  Modellene kan leveres i forskjellige kategorier.
  Hovedsakelig to typer som skiller mellom gitte egenskaper som bevegelig rygg eller ikke.

- Treverk:
  Hvert møbel kan produseres i et utvalg av treverk.

- Trekk/Farge:
  De ulike modellene kan leveres i et utvalg av forskjellige trekk (hud, microfiber, tekstil) som leveres i forskjellige farger.

- Tilhørende modell/størrelse:
  Enkelte modellserier kan linkes til hverandre som sammenfallende pga utseende og annet. Det må også komme frem funksjonalitet som fanger opp om modellene finnes i andre størrelser.

- Forhandler:
  Hver forhandler opererer bare med et gitt utvalg modeller.

- Region:
  De ulike regionene er geografiske avgrensninger som hver enkelt forhandler opererer innenfor.

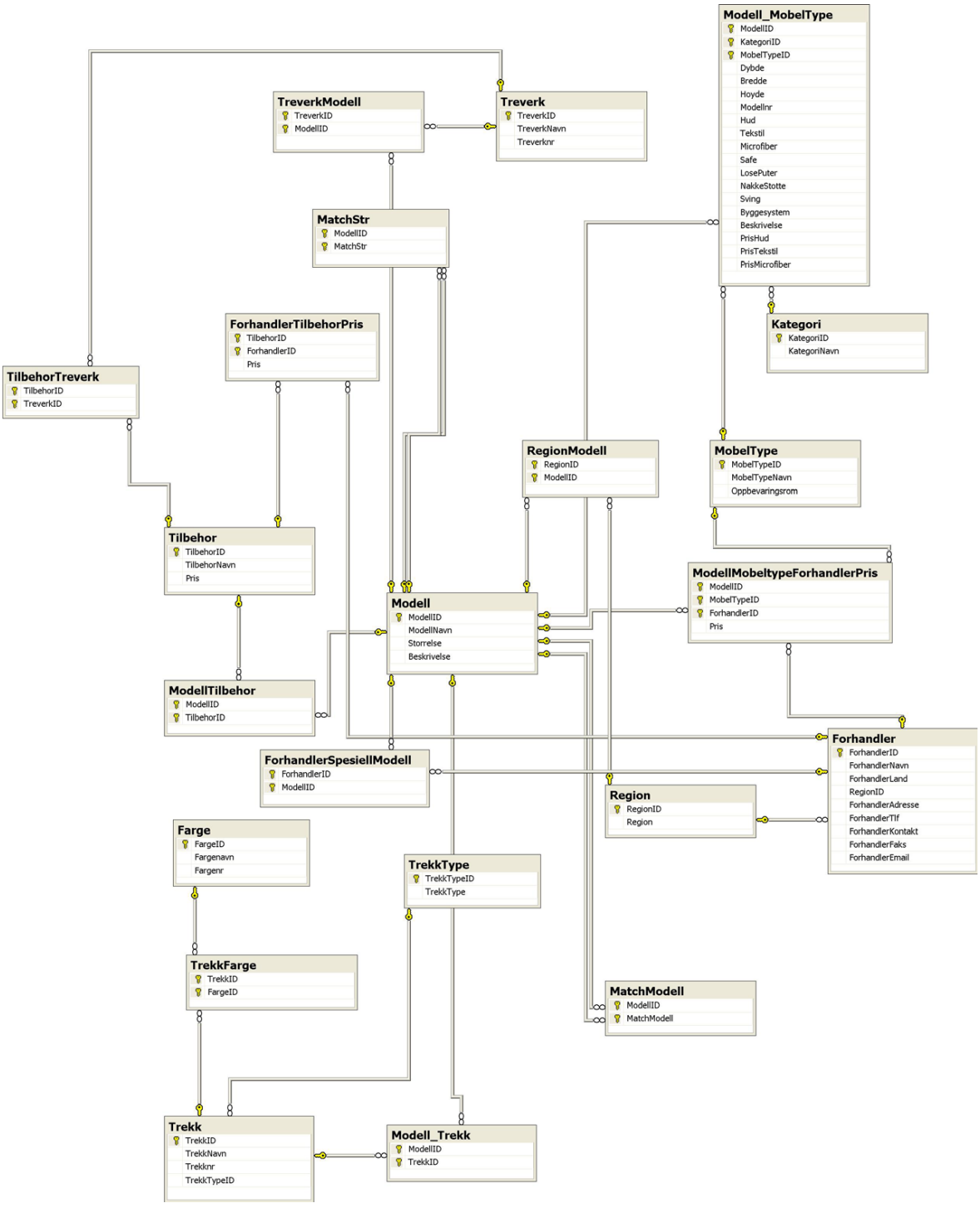
5.2.2 Struktur på XML- filer

Vi har delt opp informasjonen til produktkatalogen i to kategorier:

- Forhandler
- Møbel

Hver forhandler skal ha hver sin XML fil hos tjeneren som inneholder informasjon om forhandleren samt de modellene som forhandleren har i sitt sortiment. Alle modellene inneholder informasjon som hvilke møbler som finnes innenfor hver modell, navn, matchende møbler, tilbehør, størrelse, pris, m.m. [Figur 5.3]

Hver enkelt møbel består av tilgjengelige kombinasjoner, og skal ha en egen XML fil som beskriver hvilke kombinasjoner av treverk, trekk og farge som er lovlig. [Figur 5.4]



Figur 5.2: Databasediagram

```
- <Forhandler ID="1002" Navn="Skeidar Trondheim" Land="Norge" Adresse="Ivar
  Lykkesvei 3, 7075 Tiller" Tlf="72894800" Kontakt="Nils Arne" Faks="72894801"
  Email="nei">
  - <Modell Modellnavn="diplomat" Size="S">
    - <Kategori Kategorinavn="Stressless">
      - <Mobel Modellnr="005" navn="Stressless" Oppbevaringsrom="0" Dybde="70"
          Bredde="72" Hoyde="94" Hud="1" Tekstil="1" Microfiber="1" Safe="1"
          LosePuter="1" NakkeStotte="1" Sving="1" Byggesystem="1"
          Beskrivelse="tulle" PrisHud="30000" PrisTekstil="100"
          PrisMicrofiber="100">
        - <MatchModell>
            <MatchModell MatchID="1012" ModellNavn="windsor" />
            <MatchModell MatchID="1013" ModellNavn="tampa" />
          </MatchModell>
        - <MatchStr>
            <MatchStr MatchStr="1009" ModellNavn="consul" />
            <MatchStr MatchStr="1010" ModellNavn="ambassador" />
          </MatchStr>
        - <Tilbehor>
            <Tilbehor TilbehorID="1000" TilbehorNavn="Stressless PC-bord" />
          </Tilbehor>
        </Mobel>
      </Kategori>
    </Modell>
  + <Modell Modellnavn="consul" Size="M">
  + <Modell Modellnavn="ambassador" Size="L">
  + <Modell Modellnavn="eldorado" Size="M">
  - <Modell Modellnavn="windsor" Size="M">
    - <Kategori Kategorinavn="Stressless">
      - <Mobel Modellnr="012" navn="Stol" Oppbevaringsrom="0" Dybde="79"
          Bredde="94" Hoyde="103" Hud="1" Tekstil="1" Safe="0" LosePuter="0"
          NakkeStotte="0" Sving="0" Byggesystem="0" Beskrivelse="Leveres med
          synlig treverk" PrisHud="11100" PrisTekstil="9370">
        - <MatchModell>
            <MatchModell MatchID="1013" ModellNavn="tampa" />
          </MatchModell>
        </Mobel>
      - <Mobel Modellnr="012" navn="2-Seter" Oppbevaringsrom="0" Dybde="79"
          Bredde="149" Hoyde="103" Hud="1" Tekstil="1" Safe="0" LosePuter="0"
          NakkeStotte="0" Sving="0" Byggesystem="0" Beskrivelse="Leveres med
          synlig treverk" PrisHud="15860" PrisTekstil="12930">
        - <MatchModell>
            <MatchModell MatchID="1013" ModellNavn="tampa" />
          </MatchModell>
        </Mobel>
      - <Mobel Modellnr="012" navn="3-Seter" Oppbevaringsrom="0" Dybde="79"
          Bredde="204" Hoyde="103" Hud="1" Tekstil="1" Safe="0" LosePuter="0"
          NakkeStotte="0" Sving="0" Byggesystem="0" Beskrivelse="Leveres med
          synlig treverk" PrisHud="20560" PrisTekstil="17340">
        - <MatchModell>
            <MatchModell MatchID="1013" ModellNavn="tampa" />
          </MatchModell>
        </Mobel>
```

Figur 5.3: Utdrag fra xml

```
- <Modell_MobelType Modellnr="050">
  - <Treverk Treverknr="0">
    - <Trekk Trekknr="092">
        <Farge Fargenr="02" />
        <Farge Fargenr="14" />
        <Farge Fargenr="16" />
        <Farge Fargenr="19" />
        <Farge Fargenr="24" />
        <Farge Fargenr="28" />
        <Farge Fargenr="30" />
        <Farge Fargenr="44" />
        <Farge Fargenr="47" />
        <Farge Fargenr="57" />
        <Farge Fargenr="64" />
        <Farge Fargenr="72" />
        <Farge Fargenr="81" />
        <Farge Fargenr="91" />
        <Farge Fargenr="92" />
      </Trekk>
    + <Trekk Trekknr="093">
    + <Trekk Trekknr="094">
    + <Trekk Trekknr="095">
    </Treverk>
  + <Treverk Treverknr="1">
  + <Treverk Treverknr="2">
  + <Treverk Treverknr="3">
  + <Treverk Treverknr="5">
  + <Treverk Treverknr="9">
  </Modell_MobelType>
```

Figur 5.4: XML struktur for treverk-, trekk- og fargekombinasjoner

5.3 Bildebehandling

Hvert møbel har et tilhørende bilde, og kommer i forskjellige trekk og treverk. Bilder av møblene rendres av Scene7-tjeneren fra forespørsler ut fra hvilke treverk og trekk som er ønsket på møbelet. Bildene hentes ved hjelp av URL-forespørsler på følgende måte:

```
http://s7ondemand4.scene7.com/is/image/Ekornes/09425?crop=100,100,100,100
```

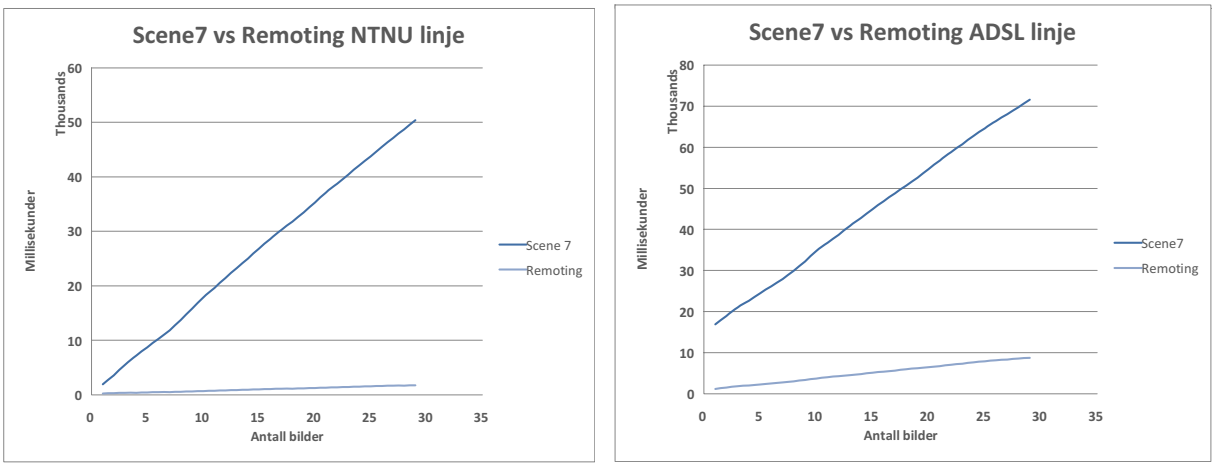
Bilder av modellene ligger på en Render Server. [39]
URL-en ser slik ut:

```
http://s7irondemand4.scene7.com/EkornesRender/<Vignette>?<Scene7-parameter>&
obj=body&src=<trekk>&obj=wood&src=<treverk>&gloss=0
```

<Vignette> er modellnummer bindestrek tusen, f.eks. 050-1000 for Stressless Senator.
Eksempel på URL:

```
http://s7irondemand4.scene7.com/EkornesRender/050-1000?wid=364
&hei=260\obj=body&src=09425&obj=wood&src=00&gloss=0
```

Etter en effektivitetstest [Figur 5.5] angående hvor raskt Scene7 kunne levere bilder i forhold til .NET Remoting TCP-BIN, ser vi at vi vil oppnå en større effektivitet ved å lagre bildene på applikasjonstjeneren som klientene kaller for å få de bildene som de trenger. Med andre ord er det kun applikasjonstjeneren som henvender seg til Scene7-tjeneren for å hente ned bilder. Applikasjonstjeneren vil dermed ha en lokal kopi av alle mulige lovlige kombinasjoner. Ved forandringer i produktporteføljen vil applikasjonstjeneren sørge for å hente kombinasjonene den mangler fra Scene7-tjeneren.



Figur 5.5: Sammenlikning ved nedlasting av bilder fra Scene7-tjener og .NET Remoting applikasjonstjener

5.3.1 Standard for lagring av filer

Når man opererer med mange filer er det hensiktsmessig å navngi de etter en standard. Dette gjør det enklere å finne igjen spesifikke filer manuelt.

Vi har valgt å navngi bildene etter følgende standarder:

- Bilder som viser modeller
  <modellNavn>.jpg
  Eksempel:
  diplomat.jpg

- Bilder som viser tilgjengelige møbler innenfor en modell
  modell<modellNr>_ <møbelNavn>.jpg
  Eksempel:
  modell050_Stressless.jpg, som gir Diplomat stressless
  modell012_3–Seter.jpg, som gir Windsor 3–seter

- Bilder som viser treverk
  tre_ <treNr>.jpg
  Eksempel:
  tre_2.jpg, som gir teak

- Bilder som viser trekkfarger
  <trekktypeNr><fargeNr>.jpg
  Eksempel:
  09202.jpg, som gir hudtypen classic i fargen vanilla

- Bilder som viser kombinasjoner
  modell<modellNr>_ <trekktype>_ <trekktypeNr><fargeNr>tre_ <treNr>.jpg
  Eksempel:
  modell012_Hud_09202tre_2.jpg, som gir Windsor 3–seter i hudtypen classic i fargen vanilla, og med teak treverk
  modell050_Hud_09202tre_2.jpg, som gir Diplomat stressless i hudtypen classic i fargen vanilla, og med teak treverk

XML- filene har også logiske navn der *ForhanderID* og *ModellNr* benyttes.

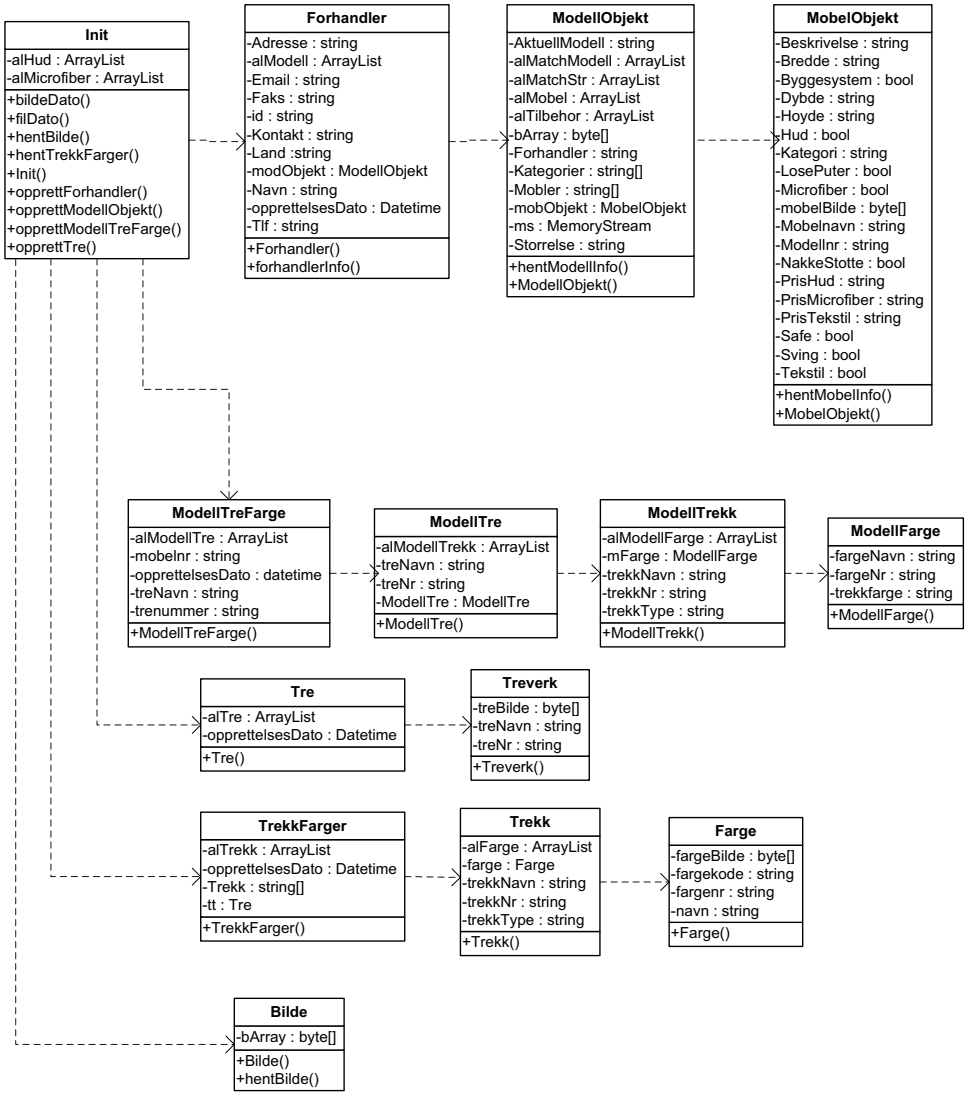
5.4 Design av applikasjon

Klassene som skal brukes til utveksling av data skal finnes i et eget klassebibliotek og må følge med både klienten og tjeneren. Ved hjelp av klassebiblioteket vet klienten hvilke

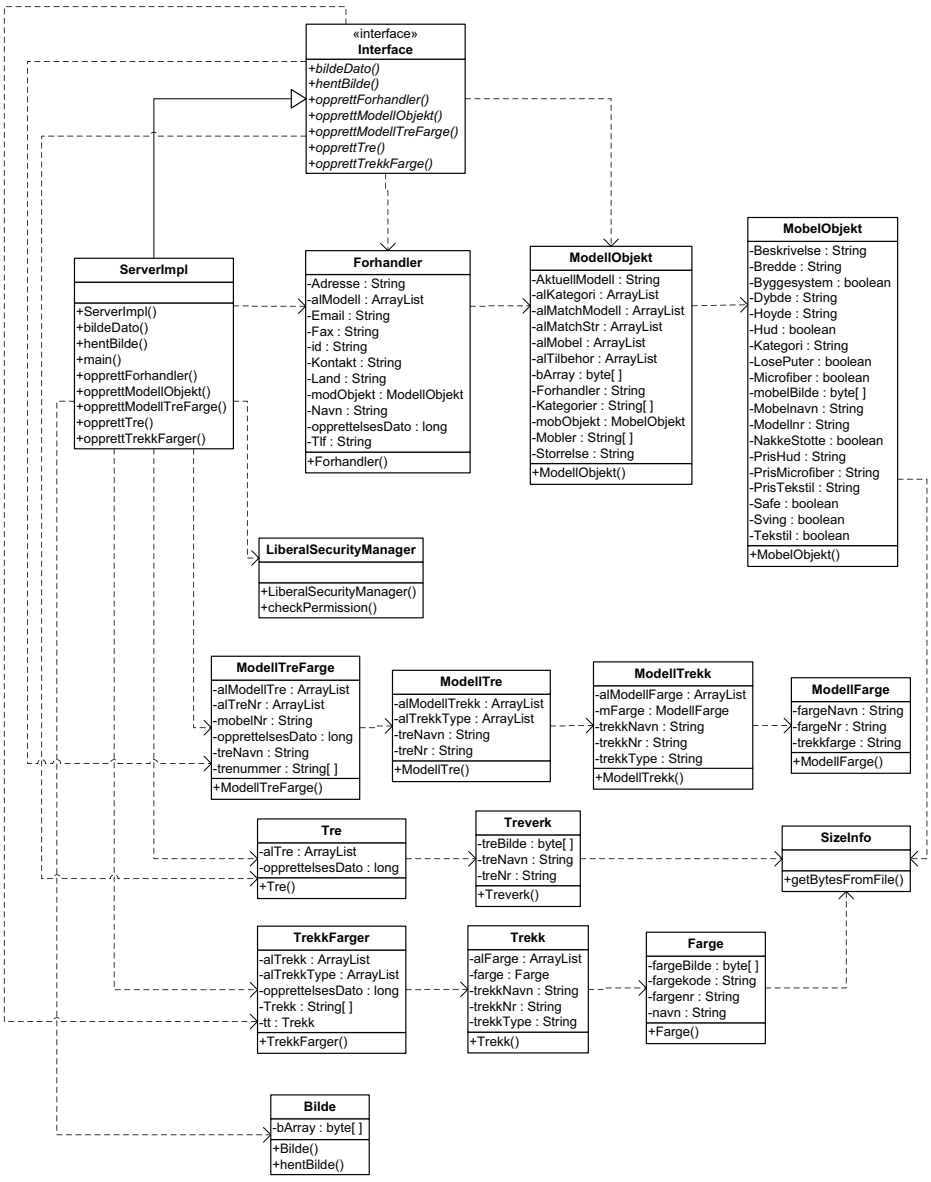
metoder og verdier tjener har tilgjengelig og som klienten kan benytte seg av.

Generelt for alle teknologiene er at tjeneren skal ha alle metodene som er tilgjengelig for klienten. Klienten forespør tjeneren etter informasjon som han ønsker. Hver klient skal identifiseres ved sitt forhandlernummer. Tjeneren skal ha XML filer som inneholder informasjon om hver forhandler og hvilke modeller den har tilgjengelig i sitt sortiment, og skal også ha bilder og informasjon om alle modellene som er tilgjengelig.

5.4.1 Klasser



Figur 5.6: .NET klassediagram



Figur 5.7: JAVA klassediagram

5.4.2 Caching og oppdateringsmekanismer

I teoridelen har vi beskrevet ulike typer caching som kan benyttes, se kapittel 4.9 på side 25. Vi ønsker å øke effektiviteten ved klienten ved å holde på bildene i cache for raskest mulig fremvisning av bildet ved klienten. Etter en viss størrelse vil de minst sette bildene som klienten holder på bli tømt ut av minnet for å frigjøre tilgjengelig minne.

Til vårt valg har vi lagt til grunn at det er en distribuert applikasjon som utvikles og direkte kontakt til tjeneren er dermed viktig og bør anses å ha en høy oppetid. Som vårt valg til oppdaterings- og caching mekanismer har vi valgt:

Klienten:

- Klienten skal cache bildene lokalt opptil 500 bilder.
- Klienten skal lagre bildene i tillegg til dato og tidspunkt for når den hentet bildet. Ved neste forespørsel, skal klienten spørre tjeneren om den har et nyere bilde.
- Klienten skal bruke de lokalt cachede bildene om klienten ikke skulle ha tilgang til tjeneren.
- Klienten skal oppdatere produktkatalogen automatisk når den kobler opp mot tjeneren og de nye modellene skal bli øyeblikkelig tilgjengelig.

Tjeneren:

- Bildene må være oppdatert i forhold til de som befinner seg hos Scene7 tjeneren
- Skal ikke cache bildene, men har de lagret på disk ettersom det er en liten effektivitets inntjening i å ha bildene i cache i forhold til tiden som brukes på å overføre et bilde.

Denne cachemekanismen vil ikke påvirke testene teknisk sett, siden den bare vil ha innvirkning på klienten og dens grafiske grensesnitt. Ettersom det ikke eksisterer et testverktøy for klientgrensesnittet vil vi diskutere effekten av den under evaluering av det grafiske grensesnittet i kapittel 8.2.1 på side 72.

5.4.3 Klient og grafisk grensesnitt

Vi vil opprette et brukergrensesnitt som vil oppføre seg som en reell applikasjon. Det skal være mulig å benytte seg av alle de elementære funksjonene som bør være tilgjengelig i en interaktiv produktkatalog. I vårt tilfelle vil det være å endre modell, farge o.l. for så å få opp informasjon om de forskjellige produktene. Det skal designes slik at all interaksjon skal kunne foregå ved å trykke på knapper eller bilder. Det skal med andre ord ikke være nødvendig å ha tilgang til tastatur for å kunne konfigurere ønsket møbel.

Det grafiske grensesnittet skal inneholde en liste over tilgjengelige modeller gjerne ved hjelp av bilder. Man skal kunne trykke på disse og få frem informasjon om den aktuelle modellen og hvilke møbler som tilhører.

Møblene skal fremvises ved hjelp av bilder, og man skal få frem alle tilgjengelige trekk og treverk i alle mulige farger. Kunden skal deretter kunne trykke seg frem til ønsket konfigurasjon på møbelet, og få vist bilde av det.

Hensikten er ikke å ferdigutvikle et grafisk grensesnitt som er klart til bruk, men å designe det slik at vi kan evaluere *"look and feel"*. Derfor er det viktig at det innenfor alle teknologier designes forholdsvis likt.

Kapittel 6

Implementasjon

Som nevnt tidligere i oppgaven har vi valgt å implementere en løsning i tre forskjellige mellomvarestandarder for å kunne sette dem opp mot hverandre. Vi vil her kort ta frem de viktigste elementene i implementasjonen av løsningene og vi vil først ta for oss emner som er generelle for alle tre implementasjonene og deretter ta frem emner som er spesifikk for de forskjellige teknologiene.

6.1 XML

Under utformingen av strukturen for XML-filer, oppdaget vi at mange metadata hadde kompliserte relasjoner. Derfor bestemte vi oss for å lage en relasjonsdatabase i MSSQL 2005, hvor vi la inn all informasjonen for deretter å opprette de nødvendige XML filene. Vi støtte på et problem hvor den innebygde funksjonen i MSSQL 2005 kun skrev 2033 tegn omgangen hvor den så laget et mellomrom for så å skrive 2033 tegn igjen. Dette resulterte i at XML filene som ble opprettet var ugyldige. For å løse dette problemet laget vi vår egen Stored Procedure som skrev XML filene. Vi opprettet så Triggere i databasen slik at XML filene ble oppdatert når informasjonen i databasen ble endret. Databasen ble altså kun brukt for å opprette XML filene som inneholder informasjonen og er ikke knyttet direkte til applikasjonene.

Vi har opprettet to typer XML filer, én XML fil for hver forhandler som inneholder forhandler og modell informasjon, og én XML for hvert møbel.

6.2 Optimalisering ved henting av bilder

Under utvikling av applikasjonen så vi at det tok lang tid å opprette bildeobjektene. Vi undersøkte da ulike metoder for å opprette bilder, og kom fram til at det ville være mer effektivt å returnere et bytearray av streamen enn det ville være å returnere en MemoryStream som først antatt. Ved å gjøre denne endringen oppnådde vi en effektivisering

i tidsbruk på 91%.

Minnebruk har også vært et viktig aspekt å ta hensyn til og var noe av det første vi la merke til ved applikasjonen. Vi så også her at vi kunne optimalisere for å senke minnebruken. Vi startet med å lagre bildet som en MemoryStream i en arraylist, noe som medførte en reell minnebruk på ca 368MB på 300 bilder (start av applikasjonen krever ca 32MB). Bruk av Stream førte i tillegg til uakseptabel minnebruk på godt over 500MB, men dette problemet løste seg også når vi gikk over til å bruke byteArray. Vi opprettet og en egen klasse hos klienten for å holde på mer informasjon enn selve bildet som antall ganger bildet er blitt vist og datoen bildet ble hentet fra tjeneren. Dette senket minnebruken betraktelig; 12MB på 300 bilder. Det første alternativ brukte da ca 800% mer minne.

6.3 Caching av bilder

For å spare tid på overføring av bilder, samt ha raskere tilgang til bildene lokalt, cacher vi bildene lokalt i et byteArray. Ved hjelp av denne funksjonen vil et bilde som blir hentet en gang bli lagret i minnet slik at den kan brukes senere uten at det er nødvendig å hente det samme bildet fra tjeneren på nytt. For å holde orden på alle bildene bruker vi en ArrayList som har fordelen av å være av dynamisk størrelse.

```
1  if (alBilde.Count > 500) {
2      //skal flushe 100 minst sette bilder
3
4      for (int i = 0; i < alBilde.Count; i++)
5      {
6          for (int j =0; j < alBilde.Count; j++)
7          {
8              if (((BildeInfo)alBilde[j]).Klikk <
9              ((BildeInfo)alBilde[i]).Klikk)
10             {
11                 BildeInfo temp =((BildeInfo)alBilde[i]);
12                 alBilde[i] = alBilde[j];
13                 alBilde[j] = temp;
14             }
15         }
16     }
17     alBilde.RemoveRange(400,alBilde.Count-400);
```

For å holde klienten under en viss mengde i minnebruk opprettet vi en metode for å kaste ut en viss mengde bilder. Når det er kommet over 500 bilder inn i listen som holder på alle bildene, blir det kjørt en funksjon for å dumpe de minst sette bildene. Først sorterer vi bildelisten etter hvor mange ganger bildet er blitt vist, slik at vi tar vare på de bildene som er vist flest ganger. Deretter kaster vi ut bildene fra nummer 401 og til slutten av listen for å frigjøre minne.

Klientene må innenfor et bestemt tidspunkt sjekke tjeneren om produktproteføljen er

oppdatert. Denne syklusen bestemmes av variabler som online/offline eller en bestemt tid på døgnet hvor forbruket av internett forbindelsen er lav. Klienter med en konstant høyhastighets forbindelse kan oppdatere oftere og på andre tider av døgnet i forhold til en klient med oppringt forbindelse og lav overføringskapasitet.

For å holde på enkel informasjon om bildene har vi opprettet en BildeInfo klasse som inneholder datoen når bildet ble hentet, antall ganger det bildet er blitt klikket siden det ble tømt fra minnet og selve bildet. BildeInfo objektet blir deretter lagret i cachen. Det er altså BildeInfo objektet som blir oppbevart i cachen og ikke kun selve bildet. Bildedatoen som blir lagret blir brukt for oppdatering av bildet. Når klienten velger et bilde som ligger i cachen blir datoen som er lagret i BildeInfo klassen sammenlignet med datoen som tjeneren har på sitt bilde. Hvis tjeneren har en nyere dato på sitt bilde er klientbildet gammelt og klienten vil oppdatere bildet sitt. Hver gang et bilde som ligger i cachen vises vil klikk telleren til BildeInfo objektet inkrementeres og det er ut fra dette kriteriet det bestemmes hvilke bilder som beholdes i minnet når cachen flushes ned til 400 bilder. De bildene som er vist flest ganger blir beholdt i minnet, ettersom det er teoretisk størst sannsynlighet for at det er de bildene som vil velges igjen og minsker dermed unødvendig overføring fra tjeneren.

6.4 Kopiering av tjenerobjekt

For å kunne bruke objektene som blir opprettet hos tjeneren lokalt må klienten kopiere over objektene. For at klienten automatisk skal kopiere over objektene merkes tjenerklassene i JAVA RMI og .NET som Serializable, og i WebService som WebMethod for at klienten skal kopiere og opprette et lokalt objekt av tjenerklassene. Hvis klassen ikke er Serializable vil klienten bare få en referanse til objektet til tjeneren og det vil da ikke være mulig å bruke klienten delvis om nettforbindelsen skulle være nede.

6.5 Tilgang til tjeneren

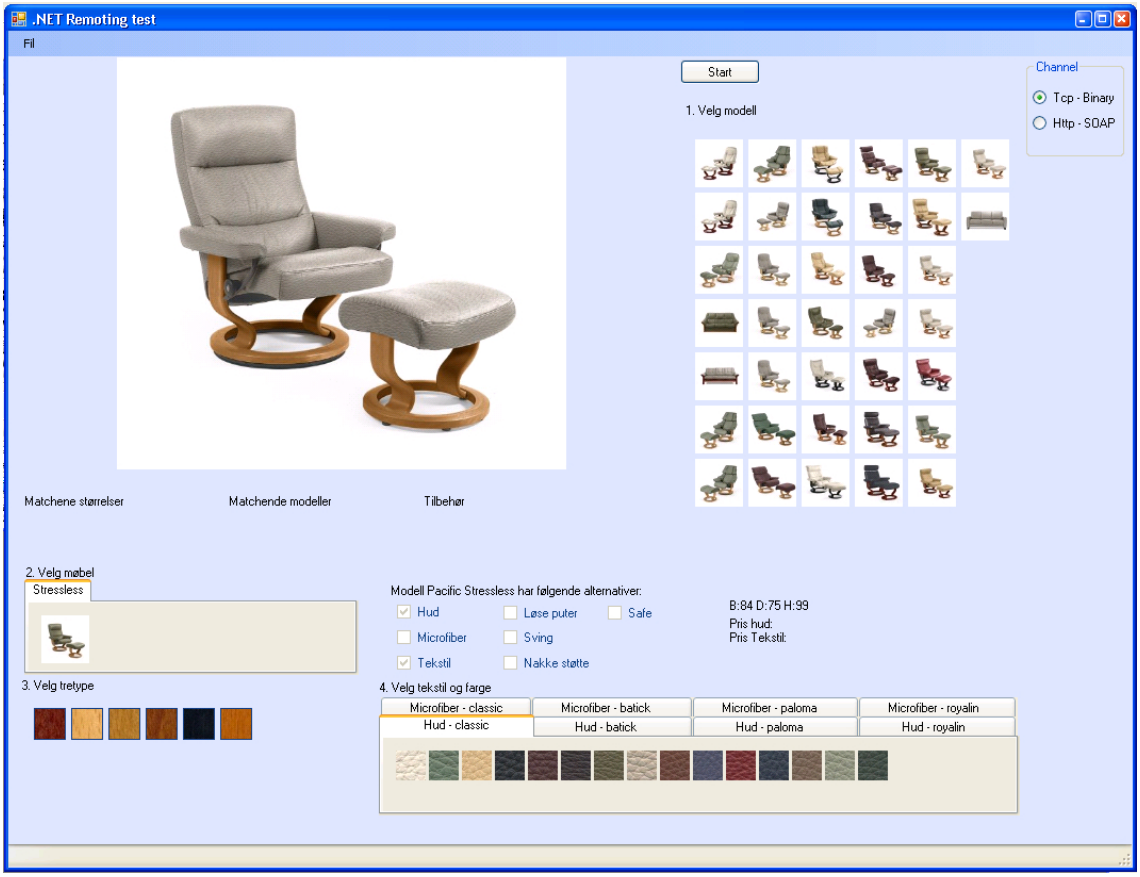
```
if (internett == true) { ...
```

For å kontrollere om du har kontakt med tjeneren har vi laget en funksjon for å sjekke om den er tilgjengelig. Hvis tjeneren ikke er tilgjengelig vil klienten bruke de bildene som den allerede har lokalt. Hvis tjeneren er tilgjengelig vil klienten hente de bildene som den ikke har, samt sjekke datoen på eksisterende bilder for så å oppdatere dersom tjeneren har et bilde av nyere dato.

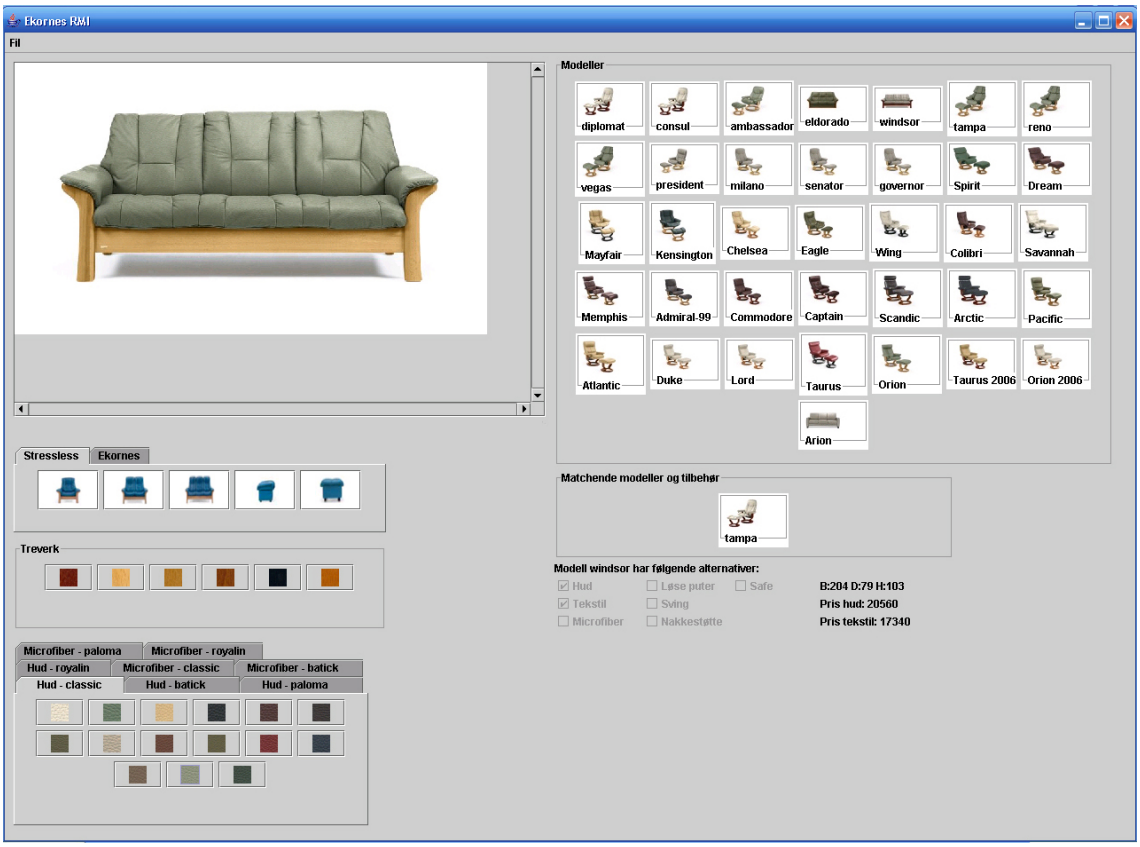
6.6 GUI

Bilde [Figur 6.1, 6.2, 6.3] viser GUIen til den respektive applikasjonen. Alle implementasjonene har tilsvarende lik funksjon, men i .NET Remoting applikasjonen er det mulig

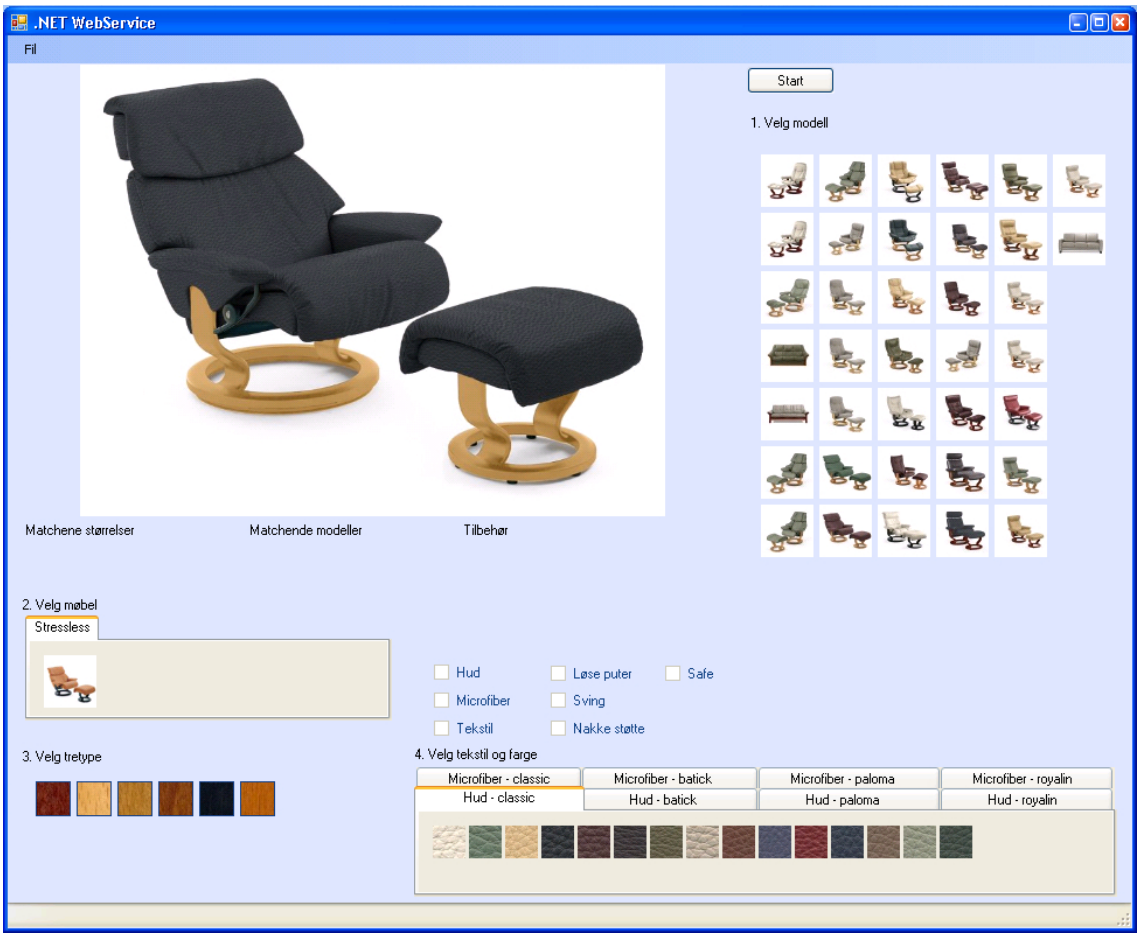
å velge mellom TCP-BIN eller HTTP-SOAP protokoll og formatering. Under punkt 1 velges det en modell. Den aktuelle modellen vises i det store bildet midt på skjermen, alle modellene vises på høyre siden. Under vises matchende størrelser, som er lik stol men i annen størrelse, matchende modeller og tilbehør til den aktuelle modellen. Under punkt 2 velges den møbel typen som ønskes (stol, sofa, puff, og lignende). Deretter kan man velge tretype og farge på det møbelet som man ser på. Fargene er delt inn i typer og farge. Over fargene vises hvilke alternativer som er tilgjengelig for det aktuelle møbelet samt bredde, høyde, dybde og pris for de ulike trekk alternativene.



Figur 6.1: .NET GUI



Figur 6.2: RMI GUI



Figur 6.3: Web Service GUI

6.7 .NET applikasjon

.NET Remoting er en teknologi for å utvikle Etter at all datastruktur var på plass begynte vi med å utforme applikasjonen. Vi startet med å utvikle applikasjonen i Microsoft sitt .NET rammeverk.

6.7.1 Tjener

Ved oppsettet av tjeneren har vi valgt hvilke porter, formattere og overføringskanaler som skal benyttes. Under oppsettet kan man definere forskjellige kanaler som kan benyttes for å koble til tjeneren. For å kunne evaluere de forskjellige scenarioene har vi satt opp en TCP kanal med binær formatering og en HTTP kanal med SOAP formatering.

```
1  BinaryServerFormatterSinkProvider bProvider = new
2  BinaryServerFormatterSinkProvider(); SoapServerFormatterSinkProvider
3  sProvider = new SoapServerFormatterSinkProvider();
4  bProvider.TypeFilterLevel = TypeFilterLevel.Full;
5  sProvider.TypeFilterLevel = TypeFilterLevel.Full;
6
7  IDictionary propshttpsoap = new Hashtable();
8
9  IDictionary propstcpbin = new Hashtable();
10
11 propshttpsoap["port"] = 8088; propshttpsoap["name"] = "httpsoap";
12 propstcpbin["port"] = 8085; propstcpbin["name"] = "tcpbin";
13
14 propstcpbin["authenticationMode"] = "IdentifyCallers";
15
16 chantcpbin = new TcpChannel(propstcpbin, null, bProvider);
17 chanhttpsoap = new HttpChannel(propshttpsoap, null, sProvider);
18
19 ChannelServices.RegisterChannel(chantcpbin, false);
20 ChannelServices.RegisterChannel(chanhttpsoap, false);
21
22 RemotingConfiguration.RegisterWellKnownServiceType(typeof(Share.Init),
23 "Modell", WellKnownObjectMode.SingleCall);
```

Kanalene blir så registrert før tjenesten blir registret og gjort tilgjengelig for klientene. Tjeneren blir så bare en applikasjon som ligger og lytter på klienter.

6.7.2 Klient

For å koble til tjeneren må klienten selv opprette en kanal og velge hvilken type formatering den skal bruke, om klienten prøver en kanal- og formateringskombinasjon som ikke støttes av tjeneren vil klienten ikke klare å koble til. Det er derfor viktig at klienten vet hvilke kommunikasjonstyper tjeneren støtter. I kodeeksempelet under ser vi hvordan klienten kobler til tjeneren og oppretter et objekt *obj* av tjener klassen hvor *adr* er IPen til tjeneren.

```
1  TcpChannel tcpChan = new tcpChannel(); tcpChan = new
```

```
2 TcpChannel(null, new BinaryClientFormatterSinkProvider(), null);
3
4 Share.HelloServer obj =
5 (Share.HelloServer)Activator.GetObject(typeof(Share. HelloServer),
6 adr);
```

For å kontrollere at vi får noe tilbake fra tjeneren er der en kontroll på at objektet ikke er null.

```
1 if(obj == null)
2         MessageBox.Show("Fant ikke server");
```

Alle feltene blir opprettet dynamisk avhengig av hvordan modellene er satt opp i XML filene hos tjeneren.

```
1  if (lokalt == false) {
2      mtf = init.opprettModellTreFarge(((MobelObjekt)
3      ((ModellObjekt)fo.alModell[indeks]).alMobel
4      [indeks2]).Modellnr);
5
6      alMTF.Add(mtf);
7
8      mtfIndeks = alMTF.Count - 1;
9      lovligeKnapper(mtfIndeks);
10     visBilde();
11 }
```

6.8 RMI applikasjon

6.8.1 Definering av grensesnittet

Remote- grensesnittet(interfacet) definerer hvilke metoder Remote- objektet skal eksportere (gjøre tilgjengelig for andre prosesser). Grensesnittet inneholder vanligvis bare metoder.

Grensesnittet må arve fra grensesnittet java.rmi.Remote, og alle metodene må kaste en java.rmi.RemoteException for å håndtere eventuelle feil under metodekallet.

```
1  import java.rmi.Remote; import java.rmi.RemoteException;
2
3  public interface Interface extends Remote {
4
5      public Tre opprettTre()throws RemoteException;
6      public TrekkFarge opprettTrekkFarge()throws RemoteException;
7      public Forhandler opprettForhandler(String forhandler)throws
8      RemoteException;
9
10     public byte[] hentBilde(String bilde)throws RemoteException;
11     public ModellTreFarge opprettModellTreFarge(String modell)throws
12     RemoteException;
13
```

```
14      public ModellObjekt opprettModellObjekt(String forhandler,String
15      modellnavn)throws RemoteException;
16
17      public long bildeDato(String fil)throws RemoteException;
18
19  }
```

6.8.2 Implementasjon av grensesnittet

Vi opprettet en klasse; ServerImpl.java, som implementerer grensesnittet. Klassen arver fra klassen UnicastRemoteObject i pakken java.rmi.server. Denne klassen implementerer grensesnittet Remote og inneholder funksjonalitet som gjør objekter tilgjengelige for andre prosesser via TCP/IP- tilkoblinger.

6.8.3 Serialisering av parametere og returverdier

```
1       public Forhandler opprettForhandler(String forhandler) {
2           Forhandler fo = new Forhandler(forhandler);
3           return fo;
4       }
5
6   }
```

Metoden opprettForhandler() oppretter et nytt lokalt Forhandler- objekt som så returneres til klienten. Dette objektet blir serialisert for å kunne sendes tilbake til klienten.

6.8.4 Generering av Stub og Skeleton

Etter å ha laget grensesnittet og implementasjonen av grensesnittet, måtte vi generere kode for Stub og Skeleton. Dette gjøres automatisk av kompilatoren rmic som følger med JDK. Kommandoen under genererer class- filer til Stub og Skeleton.

```
C:\j2sdk1.4.2_08\bin\rmic ServerImpl
```

6.8.5 SecurityManager

Registrering av SecurityManager gjøres på tjenersiden. En SecurityManager er et objekt som er ansvarlig for å overvåke alle sikkerhetssensitive operasjoner et javaprogram prøver å utføre, og bestemme hva slags operasjoner som er tillatt. Den standard SecurityManager'en er for restriktiv i forhold til behovene til et program som bruker RMI, fordi den ikke tillater nettverkstilkoblinger og nedlasting av kode over nettverket. Derfor registrerer vi vår egen SecurityManager. Klassen LiberalSecurityManager tillater alt av operasjoner.

```
System.setSecurityManager(new LiberalSecurityManager());
```

```
/**
 * policy.all
 */
grant{
    permission java.security.AllPermission;
};
```

6.8.6 Klientsiden

Vi oppretter en URL som identifiserer objektet vi ønsker å bruke. Denne brukes av klienten til å slå opp objektet i RMI-registry. For å starte RMI-registry:

```
start C:\j2sdk1.4.2_08\bin\rmiregistry
```

URL'en gis inn til den statiske metoden Naming.lookup(), som kontakter registry og returnerer en referanse til det forespurte objektet.

```
    public void opprettForhandler() {

        Interface server = null;
        try {
            server = (Interface) Naming.lookup("rmi://" + sServerIp
                    + ":1099/Server");
        } catch (MalformedURLException e2) {
            // TODO Auto-generated catch block
            e2.printStackTrace();
        } catch (RemoteException e2) {
            // TODO Auto-generated catch block
            e2.printStackTrace();
        } catch (NotBoundException e2) {
            // TODO Auto-generated catch block
            e2.printStackTrace();
        }
}
```

Etter at klienten har kalt metoden *Naming.lookUp()*, kan den bruke den returnerte referansen til å kalle de metodene den måtte ønske på objektet (så lenge metodene er definert i grensesnittet).

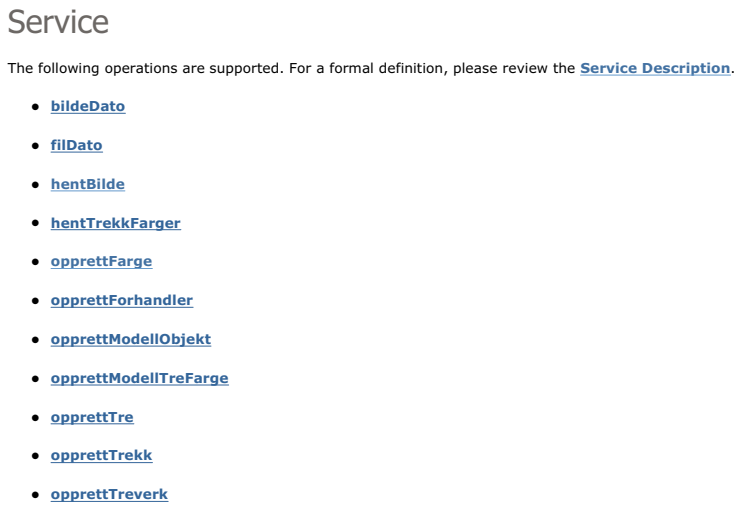
```
fo = null; try {
    fo = server.opprettForhandler("1003");
    } catch (RemoteException e){
        // TODO Auto-generated catch block
        e.printStackTrace();
        }
    int vert = 0;
    int hori = 0;

    for (int i = 0; i < fo.alModell.size(); i++) {
    ImageIcon ii = null;

    try {
```

```
14          ii = new ImageIcon(((ModellObjekt) fo.alModell.get(i)).bArray);
15          } catch (ImageFormatException e1) {
16              // TODO Auto-generated catch block
17              e1.printStackTrace();
18              } catch (Exception e1) {
19                  // TODO Auto-generated catch block
20                  e1.printStackTrace();
21              }
22
23          btMod = new JButton(ii);
```

6.9 Web Service

For implementasjon av Web Service løsningen har vi valgt å bruke .NET Framework og skrive koden i C#, og vi bruker Visual Studio 2005 som verktøy til å implementere. Den vil kjøre på en Internet Information Service (IIS) webtjener.

Når klienten skal bruke noen av metodene i Web Servicen henvender den seg til IIS-webtjeneren som igjen henvender seg til WSDL (se Tillegg B) som inneholder beskrivelsen for servicen. Ved å henvende seg til servicen ved hjelp av URLen i en nettleser vil man kunne se hvilke metoder som klienten kan benytte seg av[Figur 6.4].



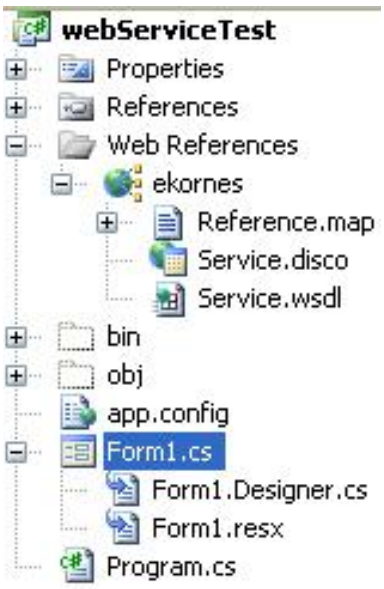
Figur 6.4: Beskrivelse av Web Service

Web Servicen skal implementeres etter samme design som .NET Remoting, og inneholde de samme klassene. Det vil si at selve koden for behandling av metadata vil være tilnærmet identisk med Remoting, bare funksjonene for oppkobling til Web Servicen vil

variere. Hovedforskjellen mellom klassene i Remoting og Web Service er at metodene som skal kunne kalles fra klienten merkes med attributten WebMethod istedenfor Serializable.

```
1        [WebMethod]
2        public Forhandler opprettForhandler(string forhandler)
3        {
4            Forhandler fo = new Forhandler(forhandler);
5            return fo;
6        }
```

For å bruke de tilgjengelige metodene hos Web Servicen må klientapplikasjonen ha en referanse til URLen der den befinner seg. Dette gjøres ved å legge til en *web-reference* i Visual Studio [Figur 6.5]. Deretter kan klientapplikasjonen opprette en forekomst av webreferansen og benytte alle tilgjengelige metoder eller klasser.



Figur 6.5: Web Service referanse

```
1 ekornes.Service init = new ekornes.Service();
```

Kapittel 7

Tester

Hvorfor skal vi teste?
Vi vil utføre en del tester mot applikasjonene vi har implementert, for å kunne avgjøre hvilke mellomvareteknologi som vil være mest effektiv til bruk i en distribuert produktkatalog. Vi ser på det som essensielt å bruke fungerende applikasjoner og relevante testdata og teste mot, for å få et så realistisk resultat som mulig.

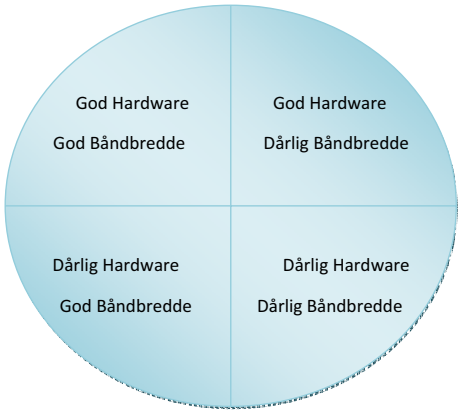
Hva skal vi teste?
Vi vil hovedsaklig bruke enhetstestene til å teste på effektivitet og tidsbruk. Altså hvor lang tid bruker tjenerapplikasjonen på å utføre viktig funksjonalitet, og hvor mye data må overføres for å få det gjort. Selve brukeropplevelsen og det grafiske grensesnittet finnes det ikke noe fullgodt enhetstestingsverktøy for. Derfor vil vi gjøre en egen vurdering på hvor godt de forskjellige mellomvare løsningene kan støttes av grafiske grensesnitt.

Hvordan skal vi teste?
Vi vil bruke rammeverk for enhetstesting, og vi har valgt oss ut to testrammeverk NUnit [7.2.1] og JUnit [7.2.2]. Dette vil gi oss mulighet til å lage automatiserte tester slik at de blir tilnærmet identiske for alle implementasjoner, og vi kan bruke testdata til å sammenligne implementasjonene.

7.1 Testmiljø

Vi har opprettet testmiljøer for å kunne evaluere de forskjellige mellomvareteknologiene under flere forutsetninger. For å evalurere med hensyn på båndbredde og hardware har vi valgt to tjenere, en med god HW og én med dårlig, og to båndbredder, én med LAN forbindelse og én ADSL forbindelse. Med disse kombinasjonene vil vi oppnå fire testmiljøer [Figur 7.1].

Begge tjenerene vil kjøre både Windows XP og Mandriva Linux, men siden .NET er en Microsoft teknologi som ikke kan kjøres under Linux, vil .NET Remoting og Web Service testene bare kjøres på windows plattform. Derfor er det bare Java RMI testene som vil bli utført på begge plattformer.



Figur 7.1: Testmiljøer

7.1.1 Hardware og operativsystem

	CPU	RAM	Operativsystem
Tjener 1	Intel Pentium 4, 3.0GHz	1GB	Windows XP, SP2 Mandriva Linux 10.01
Tjener 2	AMD Athlon, 1.53GHz	512MB	Windows XP, SP2 Mandriva Linux 10.01
Klient 1	AMD Sempron 3100+, 1.81GHz	1GB	Windows XP, SP2
Klient 2	Intel Pentium 3, 1.0GHz	512MB	Windows 2003 Server, RC2

7.1.2 Båndbredde

Vi har kjørt tjenerene på NTNUs nett som vi anser som høy båndbredde. Klientene har vi kjørt fra to forskjellige lokasjoner. *Klient 1* har kjørt tester innen innenfor NTNU

nettet, mens *Klient 2* har kjørt sine tester mot en ADSL-linje på 1,5 Mb/s, noe vi pr. i dag anser som moderat båndbredde.

7.2 Testrammeverk

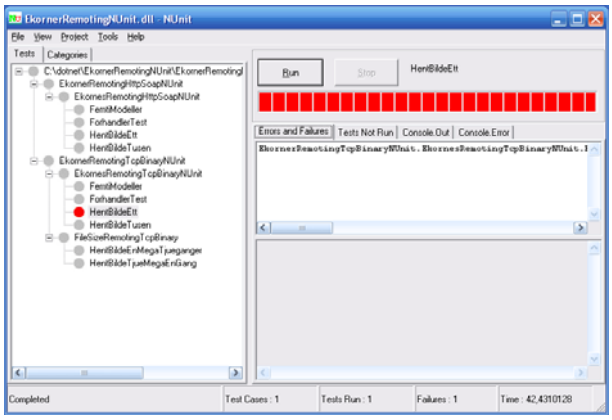
Vi har valgt oss ut to testrammeverk, *NUnit* [30] og *JUnit* [22]. Vi har ikke brukt rammeverkene til fullskala testing på alle klasser, da det ikke har vært vårt hovedmål å teste på all funksjonalitet i applikasjonen vår. Men vi har i hovedsak brukt dem til å sjekke hovedfunksjonalitet og tidsbruk for de viktigste klasser og funksjoner.

7.2.1 NUnit

For enhetstesting av de viktigste klassene og funksjonaliteten i applikasjonene skrevet i .NET har vi valgt NUnit v.2.2.7. NUnit er et rammeverk for enhetstesting for alle .NET språk. Det har sin opprinnelse i JUnit, men er fullstendig omskrevet og redesignet i C# for å kunne utnytte .NET sine muligheter fullt ut. [30]

NUnit baserer seg på rammeverket NUnit.Framework, som må implementeres i testklassen. Den krever en attributt [TestFixture], som indikerer at klassen inneholder testkode. Hver testmetode må merkes med attributten [Test]. [31] I tillegg kan man bruke attributten [SetUp] for å kunne sette opp en initialiserings-klasse for kode som skal kjøres før testene starter.

NUnit 2.2.7 kommer med et eget GUI-verktøy[Figur 7.2] for å kjøre testene. Den gir en indikasjon på om testene har kjørt uten feil, eller om de har feilet. Her kan man lagre resultatene til XML etter at de er ferdig. Man kan da i ettertid gå til XML-filen for den aktuelle testen for å finne ut hvor lang tid testen tok, hva som feilet og eventuell consol-utskrift. I tillegg til NUnit GUI, brukte vi også TestDriven 2.0. Dette er en plugin



Figur 7.2: NUnit GUI verktøy

til Visual Studio som gjør at man kan kjøre testkode direkte. Utskriften vil da komme i *debug* vinduet til Visual Studio.

7.2.2 JUnit og Ant

Til testing av java applikasjonen brukte vi JUnit [22] og Ant [37]. JUnit test klassene har ganske lik syntaks som NUnit, men krever ikke attributtene som C# koden krever. Men testklassen må ha "extends TestCase". Metoden SetUp brukes til å implementere kode som kjøres før selve testen.

Ant er et javabasert build verktøy som brukes til å kunne automatisere bygging og kjøring av javakode. Vi benyttet det hovedsaklig til å kjøre JUnit koden vår og generere rapporter fra testkjøringene.

Siden vi benyttet Eclipse til utvikling av java applikasjoner brukte vi den innebygde JUnit og Ant støtten som finnes der. Vi skrev Ant script for å kjøre JUnit klassene og fikk ut HTML rapporter fra testkjøringene slik at vi kunne sammenligne alle testene mot hverandre.

7.3 Test scenarioer

Vi har satt opp fire tester som vil teste på hovedfunksjonaliteten i applikasjonen vår. To av dem er tester som omhandler endel serverside operasjoner, som oppretting og overføring av distribuerte objekter og oppslag i XML filer. De to andre vil være rene overføringstester der vi overfører reelle testbilder.

Vi vil hovedsakelig se etter to ting, om vi får fornuftige svar fra tjeneren og tiden det tar å utføre testene. I tillegg har vi laget to tester som vil teste forskjellen mellom overføring av ett stort bilde, eller mange små.

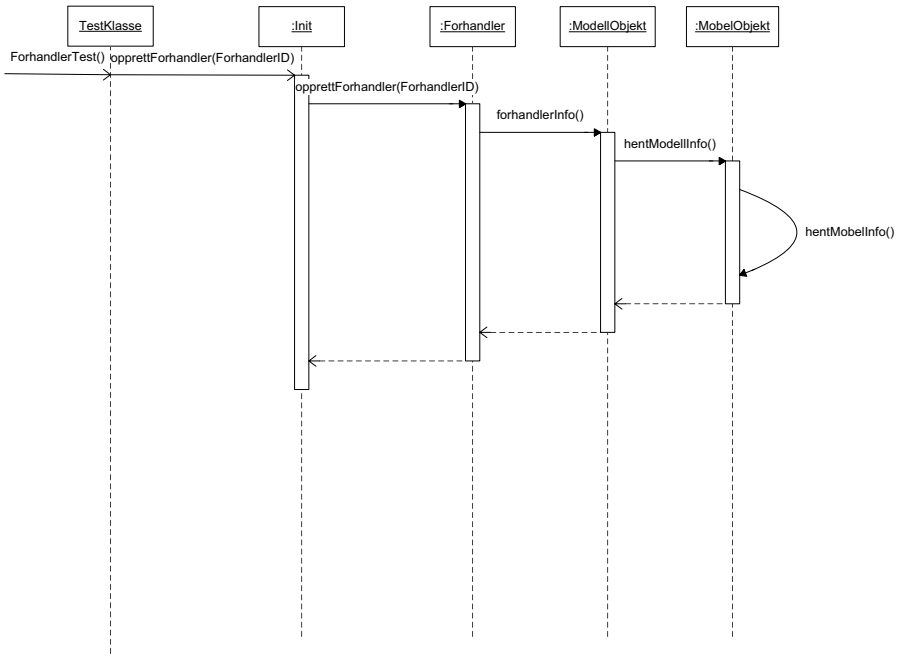
Når det gjelder overføringstestene der vi overfører bilder har vi valgt å laste ned bildene som et byte-array, for så å konvertere det til et bildeobjekt etter nedlasting. Vi testet også en annen metode der vi opprettet en memorystream som vi overførte til klienten, for så å overføre bildeobjektet direkte, men dette viste seg å ta mye lengre tid. Altså vesentlig mye mindre effektivt.

7.3.1 Oppsett

Vi initialiserte testene ved å koble opp mot tjenerene før testene ble kjørt. Dette gjøres ved at koden som benyttes for tilkobling til tjenerene plasseres i SetUp-metoden til Testklassen.

7.3.2 Forhandler test

I forhandlertesten vil vi simulere *oppstarten* av applikasjonen. Her sender vi en forespørsel til tjeneren om å opprette en forhandler. Dette er en test som krever forholdsvis mye på tjenersiden ettersom det er mange objekter som opprettes og mange oppslag i XML-filene [Figur 7.3]. Vi vil ved hjelp av denne testen kunne se hvordan de forskjellige applikasjonene takler opprettelse og overføring av større distribuerte objekter.



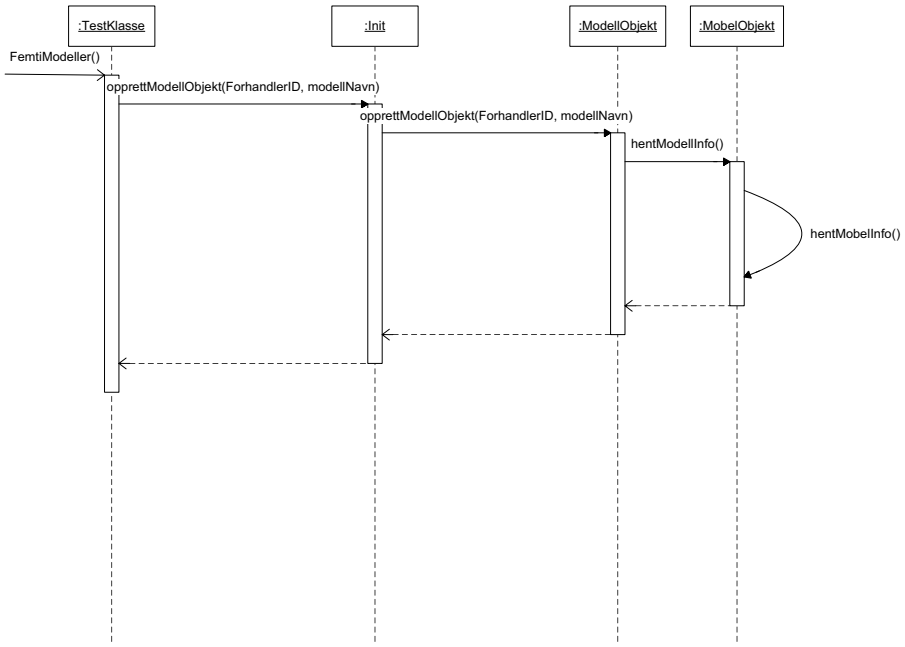
Figur 7.3: Sekvensdiagram for oppretting av forhandler

7.3.3 50 modeller

Denne testen vil sende forespørsel til tjeneren for å motta 50 objekter av typen modell. Dette er også en test som krever en del på tjenersiden [Figur 7.4]. Hensikten med denne testen er å se hvordan applikasjonene oppfører seg når det skal opprettes mindre objekter enn i den første testen, men et større antall. Modellobjektet er ikke så komplekst og stort som forhandler objektet, men til gjengjeld kjører vi 50 kall. Vi henter inn 5 forskjellige modeller og kjører dette i en løkke 10 ganger.

7.3.4 Ett bilde

I denne testen er vi på jakt etter å se hvor raskt og effektivt tjeneren kan levere ett bilde til klienten. Her sender vi en forepørsel til tjeneren om å returnere ett spesifikt bilde [Figur 7.5]. Det opprettes et bytearray som holder på alle bildedataene. Dette vil være en



Figur 7.4: Sekvensdiagram for oppretting av modell

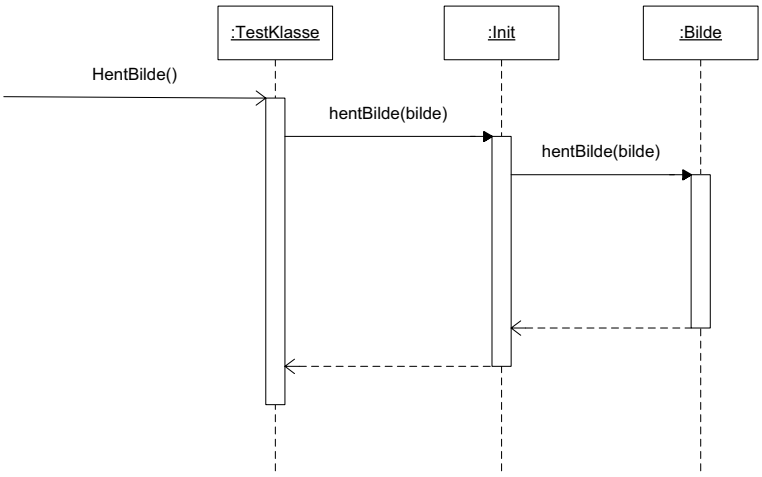
av hovedfunksjonene til en distribuert produktkatalog, der man spør etter ett produkt og overfører ett bilde av produktet. Så ser man på det før man kaller et nytt bilde. Derfor kan det være interressant å se hvordan mellomvarene takler å hente bare ett bilde.

7.3.5 Tusen bilder

Vi vil også se på hvordan tjeneren takler å sende ut et større antall bilder [Figur 7.5], og hvor raskt og effektivt dette gjøres. En mulig situasjon vil være at det lastes ned et større antall bilder eller filer av en viss størrelse som skal lagres/caches på klienten. Derfor vil vi teste hvordan mellomvarene takler å overføre et større antall bilder. Vi har valgt oss ut 10 bilder som brukes av den faktiske applikasjonen, og kjører disse i en løkke for å hentes 100 ganger hver. Tilsammen lastes det da ned 1000 bilder.

7.3.6 Filstørrelsestester

På samme måte som vi hentet bilder i de foregående testene lastet vi ned bilder til klienten. Vi vil prøve å overføre ett bilde på 20 MB for å se hvor lang tid det tar. Deretter vil vi overføre 20 bilder på 1 MB for å kunne sammenligne tidene. Målet med testen vil være å se om der er forskjell på å overføre like stor mengde data, i en stor eller flere små filer.



Figur 7.5: Sekvensdiagram for henting av bilde

7.4 Testresultater

Vi har satt opp en grafisk fremstilling av testresultatene. Fremstillingen viser resultatene fra hovedtesten vår mot de to tjenerene, og filstørrelse-testresultatene mot de samme to tjenerene.

Vi har kjørt alle testene to ganger, dette var for å dobbelsjekke resultatene våre. Fullstendig grafisk oversikt over alle tester finnes i TILLEGG A. Drøfting av testresultatene vil du finne i Kapittel 8.1 på side 65

Del III

Resultat og konklusjon

Kapittel 8

Resultat

Vi vil her gå nærmere innpå testresultatene og tyde de resultatene vi har fått. Fullstendige testresultater og grafiske fremstillinger finnes i Tillegg A.

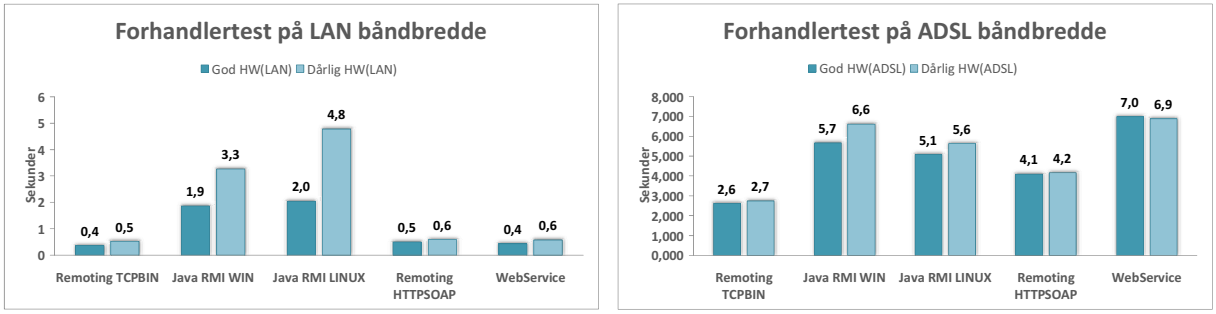
8.1 Evaluering av testresultater

Vi kan dele opp testscenarioene i kapittel 7.3 i følgende tre kategorier:

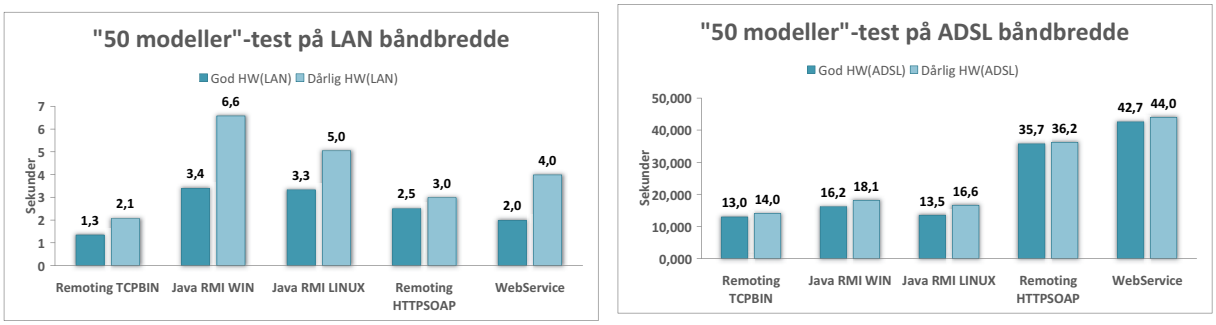
- *Tester med overføring av objekter og XML håndtering på tjener*
- *Bildeoverføringstester*
- *Filstørrelsetester*

8.1.1 Tester med overføring av objekter og XML håndtering på tjener

Her har vi to tester som inneholder mye serverside interaksjon, *forhandlertesten* og *50 modellertesten*. Forhandlertesten vil gi et riktig inntrykk på hvor lang tid det vil ta å få over nødvendig informasjon før applikasjonen vil være tilgjengelig. Dette er en test som krever mye arbeid på tjenersiden med oppretting av et eller flere objekter og håndtering av XML filer. Oppretting av 50 modeller er en mindre utgave av forhandler testen som oppretter 50 modeller med de møblene som tilhører hver modell.
Ut fra de grafiske fremstillingene i figur 8.1 og figur 8.2 kan vi se at .NET Remoting TCP-BIN serveren utfører testene raskest uavhengig av båndbredde og hardware. Vi ser at ved lokalnett er .NET Remoting TCP-BIN opptil 800% raskere ved kalling av et objekt som har mye XML håndtering på tjenersiden, enn for eksempel Java RMI Linux. I samme scenario ved bruk av en ADSL linje er .NET Remoting 100% raskere enn Java RMI Linux. Men hvilke innvirkning har hardware og båndbredde på de forskjellige typene mellomvare for denne testen?



Figur 8.1: Sammenligning av god og dårlig HW på tjenersiden ved oppretting av forhandler objekt



Figur 8.2: Sammenligning av god og dårlig HW på tjenersiden ved oppretting av 50 modeller

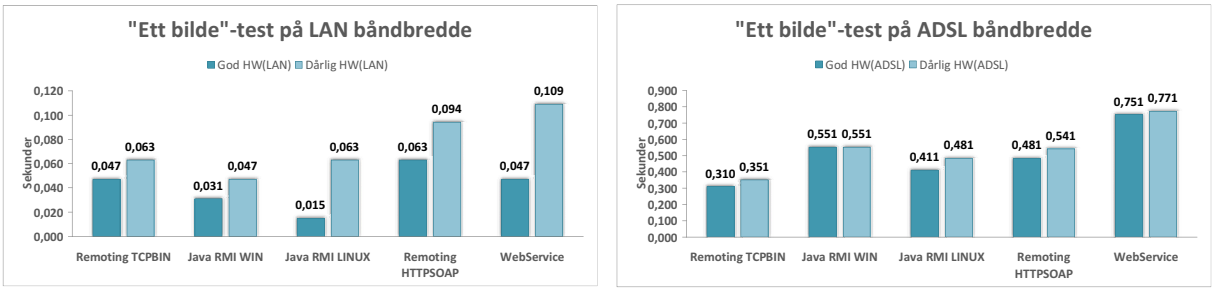
Hvilken innvirkning har båndbredden for disse testene?
Vi kan se at mellomvarene som bruker HTTP/SOAP til å overføre informasjonen, .NET Remoting HTTP-SOAP og WebService, er mye mer sårbar når man går fra god til dårlig båndbredde enn de andre. Dette kommer av at de har mye mer overhead 8.1.4 ved sending av data. Altså overføres det rett og slett mer data her for å utføre de samme operasjonene.

Hvilken innvirkning har hardwaren?
De største utslagene ser vi her når vi har ubegrenset båndbredde, siden selve overføringen ikke er noen reel flaskehals. Vi kan se at for testene med mye XML-oppslag er de .NET baserte mellomvarene betydelig raskere enn de javabaserte. De javabaserte mellomvarene er også mye mer sårbare ved en overgang fra god til dårligere hardware. Dette kan nok tyde på at XML-behandlingen i .NET er vesentlig bedre og mindre krevende for tjeneren.

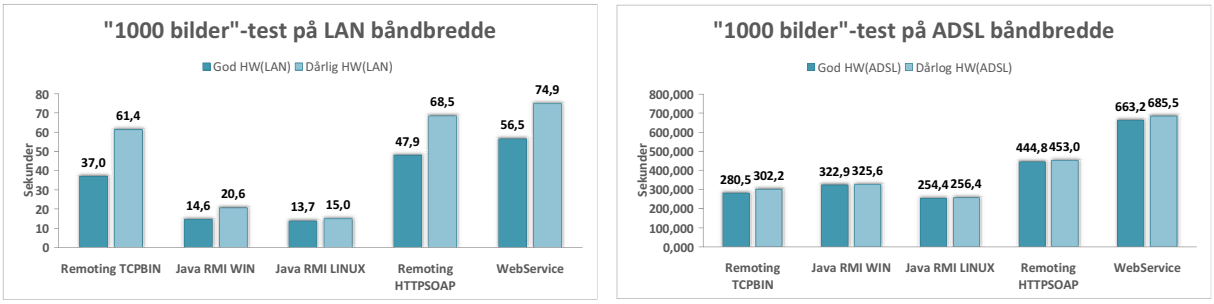
Vi konkluderer med at .NET Remoting har en mer effektiv behandling av XML filer samt oppretting av objekter. Og at de HTTP/SOAP baserte mellomvarene er mer sårbar for dårligere båndbredde ettersom de sender mer data enn de andre teknologiene.(se avsnitt 8.1.4

8.1.2 Bildeoverføringstester

Her har vi også to tester, overføring av ett bilde og overføring av 1000 bilder [Figur 8.3 og 8.4]. Scenarioet med ett bilde er det som er mest aktuelt for en produktkatalog hvor det ofte blir overført bare ett eller få antall bilder per forespørsel. Vi har også testet overføring av 1000 bilder for å se hvordan de forskjellige mellomvarene takler overføring av et stort antall filer. Dette kan være aktuelt for produktkataloger hvor klienten oppdaterer hele produktkatalogen når den kjører første gang eller ved større oppdateringer som inneholder mange filer.



Figur 8.3: Sammenligning av god og dårlig HW på tjenersiden ved overføring av ett bilde



Figur 8.4: Sammenligning av god og dårlig HW på tjenersiden ved overføring av 1000 bilder

Hvilken innvirkning har båndbredden?
Ved ubegrenset båndbredde ser vi at tidene ved overføring av ett bilde er så lik at det vil ha liten effekt på bruker av en distribuert produktkatalog, vi kan likevel legge merke til at RMI applikasjonen som kjører på Linux plattform gjør dette eksepsjonelt raskt. Når vi får begrenset båndbredde ser vi at noen av applikasjonene gjør dette bedre enn andre. Vi kan f. eks. legge merke til at .NET Remoting overfører bildet på under halve tiden av det Web Servicen bruker, selv om de gjorde de like raskt på god hardware under ubegrenset båndbredde. Vi legger også merke til at .NET Remoting takler tap i båndbredden vesentlig bedre enn de andre mellomvarene. Ved overføring av 1000 bil-

der ser vi ut fra resultatene vi har fått, at Java er overlegen ved lokalnett og er godt over 100% raskere ved overføringen enn .NET Remoting TCP-BIN. Ved bruk av ADSL linje er derimot forskjellene mindre og .NET Remoting TCP-BIN oppnår et bedre resultat enn Java RMI Windows og Java RMI Linux er 15,4% raskere enn .NET Remoting.

Hvilken innvirkning har hardwaren?
Ved en ADSL tilknytning blir resultatene lik for både god og dårlig HW, siden det er båndbredden som blir den egentlige flaskehalsen her. Men ved å studere grafene for LAN båndbredde kan vi se hvordan applikasjonene takler å gå fra god til en dårligere hardware. På overføring av ett bilde ser vi at alle teknologier får tap i overføringstid ved å gå over på dårligere hardware. Det er også vanskelig å lese noen trend ut av dette da det opereres med så korte tidsintervaller. Ved å se på testen som overfører 1000 bilder ser vi at de .NET baserte teknologiene ser ut til å takle overgang til dårligere hardware dårligst. Mens Java RMI ser ut til å ha ganske lik overføringstid uavhengig av hardwaren som ble brukt i testene.

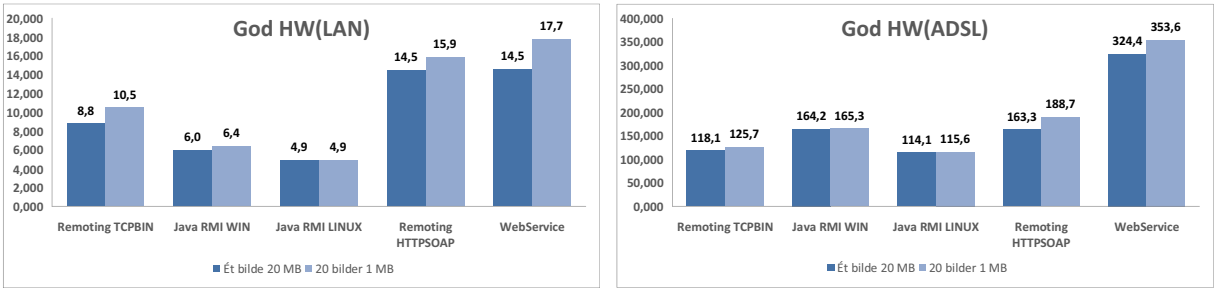
Vi konkluderer med at Java RMI Linux er teknologien som er best egnet til overføring av store mengder med bilder, mens .NET Remoting TCP-BIN er teknologien som er best egnet til overføring av ett enkelt bilde. Årsaken til at Java RMI Linux har en effektiv overføring av store mengder bilder blir diskutert i 8.1.3

8.1.3 Filstørrelsetester

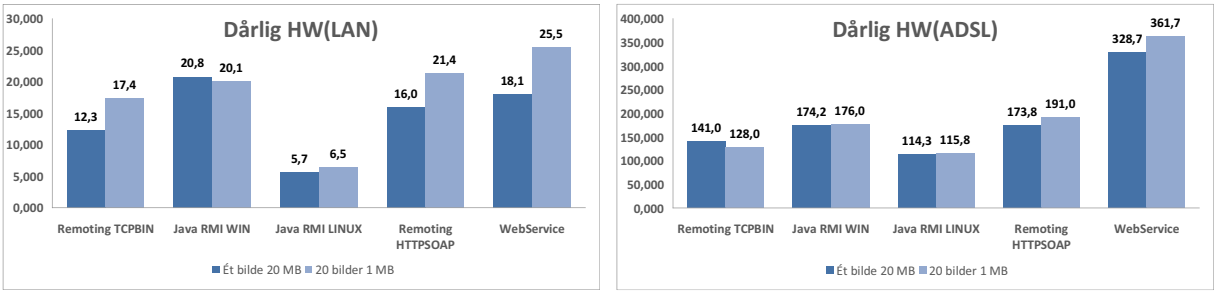
Denne testen ble først og fremst gjennomført for å se om det innenfor mellomvarene var store forskjeller hvis det som ble overført var av relativ liten størrelse flere ganger, eller en relativt stor størrelse en gang.

Vi vil også kunne bruke denne testen til å se på rene overføringstider og hvor mye data som overføres hver gang. Vi kan da sammenligne hvilke innvirkning hardware og båndbredde har på samme måte som de foregående testene. Vi kan også bruke testen til å gjøre en overhead analyse, siden vi vet størrelsen på det som blir overført.

Hva er forskjellen på overføring av ett stort og mange små bilder?
En sammenligning av de testene innefor hver teknologi [Figur 8.5 og 8.6] gir oss en oversikt over hvordan de oppfører seg i disse to tilfellene. Java RMI viser ingen nevnbar forskjell på overføring av 20 bilder á én MB eller ett bilde á 20MB verken ved ADSL eller LAN, men alle de andre teknologiene bruker lenger tid på 20 bilder enn ett bilde. Java RMI har dermed en mer effektiv *handshake* enn de andre. Vi ser også at .NET teknologiene gjennomgående bruker lengre tid på overføring av flere små enn en stor fil, med ett unntak. Ved dårlig HW på ADSL ser vi en underlig faktor der .NET Remoting TCP-BIN bruker mindre tid på overføring av 20 bilder á én MB enn omvendt. Vi ser det kun ved bruk av dårlig hardware på en begrenset linje og vi kan konkludere med at dette må ha med *marshaling* og *unmarshaling* å gjøre.



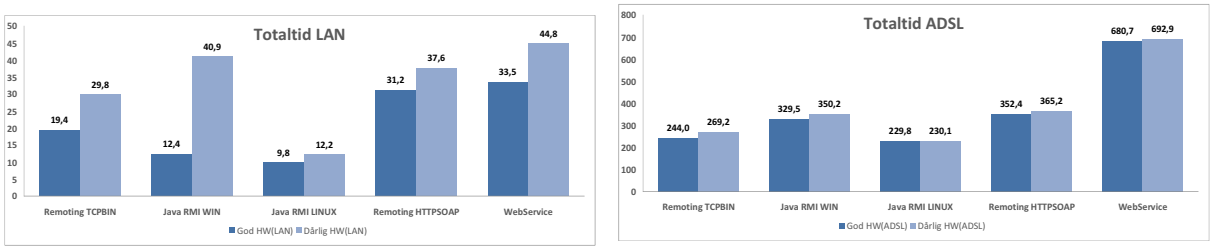
Figur 8.5: Sammenligning av bildeoverføringer ved god hardware



Figur 8.6: Sammenligning av bildeoverføringer ved dårlig hardware

Hvilken innvirkning har hardwaren?
På totaltiden [Figur 8.7] ser vi hvordan de forskjellige teknologiene oppfører seg ved dårlig mot god HW i tester som kun overfører data uten å opprette objektet eller håndtere XML. Vi kan som før se på LAN forbindelsen for å se på hvilken innvirkning hardwaren har. Java RMI Linux overfører klart raskeste av de forskjellige teknologiene og har liten forskjell mellom god og dårlig HW, noe som står i stor kontrast til Java RMI Win. Java RMI Win bruker 230% lengre tid til å utføre de samme operasjonene ved dårlig HW enn ved god HW, som er langt mer enn noen av de andre teknologiene. Resten av teknologiene, .NET Remoting TCP-BIN, .NET Remoting HTTP-SOAP og Web Services, blir også sterkt påvirket av HW valget og er langt mer effektiv ved god HW enn ved dårlig HW.

Hvilken innvirkning har båndbredden?
Ved ADSL ser vi ingen nevneverdig forskjell i tidsbruken mellom dårlig og god HW, og vi kan da se hvordan båndbredden innvirker. Testen kan settes i sammenheng med 1000 bilder testen og skal ideelt sett ha tilnærmet samme utfall siden det der også dreier seg om overføring uten noen form for objekt- eller XML-håndtering. Når vi sammenligner dem ser vi at vi oppnår tilsvarende resultater. Vi ser på samme måte at .NET Remoting takler tap i båndbredden bedre enn de andre mellomvarene, og at Java RMI på Linux



Figur 8.7: Sammenligning av totaltider ved bildeoverføring

plattform utfører overføringen raskest. Under gjennomføringen av testene monitorerte vi båndbredden som ble brukt og så på LAN testene at Java RMI klarte å utnytte båndbredde som var tilgjengelig mer effektivt. .NET hadde en topp på 0,9MB/sec mens Java RMI hadde en topp på 1,9MB/sec. Dette gir store utslag i tiden som ble brukt på å overføre bilder hvor Java RMI har en vesentlig bedre tid enn .NET. Samtidig er Java RMI langt mindre avhengig av HW når den kjører Linux. Dette gjør at Java RMI kan kjøres under Linux ved dårlig HW og fremdeles oppnå høy effektivitet.

Vi kan summere opp resultatene fra denne testen med å si at RMI ikke skiller nevneverdig med hensyn på tidsbruk om man overfører ett stort bilde eller flere mindre bilder, mens det er større utslag i .NET Remoting og Web Service. Også her ser vi at .NET Remoting takler overgangen fra god til dårlig båndbredde godt, og at Web Services bruker lengst tid p.g.a. stor overføringsmengde. Vi ser også at RMI er avhengig av god hardware på Windows plattform, mens på Linux plattform utfører Java RMI overføringstestene raskest i alle våre 4 testmiljøer.

8.1.4 Overheadanalyse

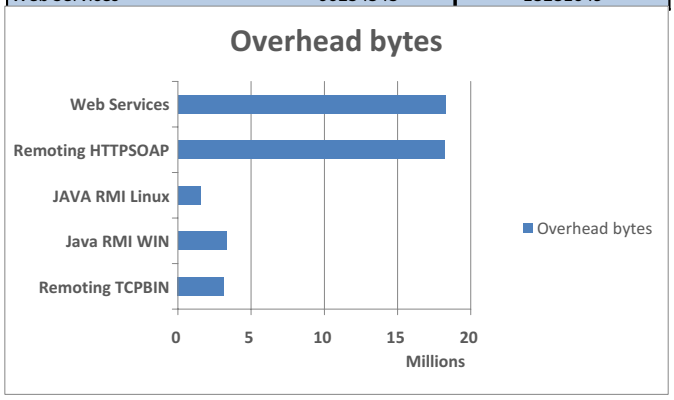
I filstørrelse-testene visste vi nøyaktig hvor mye data som skulle overføres, og ved å monitorere hvor mye data som virkelig ble overført kunne vi se hvor mye av dataene som var overhead for de forskjellige mellomvareteknologiene. Vi tok utgangspunkt i testen som overførte data via ADSL og på serveren med god hardware og laget en grafisk fremstilling av resultatene [Figur 8.8]. Samlet overføringsstørrelse på bildene var 41953494 bytes.

Svarene vi fikk var ikke på noen måte sjokkerende. Som vi tidligere har antatt ser vi den virkelig store forskjellen ved formatering til SOAP/HTTP. Vi ser klart at de tre mellomvarene som ikke bruker SOAP formatering ikke har på langt nær så mye overhead som .NET Remoting SOAP/HTTP og WebService. Vi kan konkludere med at dette overheadet skyldes formateringsmekanismene som brukes ved disse teknologiene.

Overhead Analyse ADSL-linje, god HW

Filoverføringsstørrelse: 41953494 bytes

	Overført bytes	Overhead bytes
Remoting TCPBIN	45100552	3147058
Java RMI WIN	45283822	3330328
JAVA RMI Linux	43510826	1557332
Remoting HTTPSOAP	60149466	18195972
Web Services	60234543	18281049



Figur 8.8: Overhead

8.2 Evaluering av grafisk grensesnitt

Ettersom det ikke finnes et evalueringsverktøy for GUI gjør vi kun en kort evaluering utifra egne opplevelser.

GUI'en er utformet så likt som mulig i de ulike teknologiene, og består for det meste av dynamiske knapper med bilder. Ved en tilfeldig valgt modell, oppnådde vi 175 komponenter i GUI'en. Tegning- og oppdateringstiden kan bli påvirket av en så kompleks GUI, og dette kan ha ulik virkning i de forskjellige teknologiene.

Vi har testet hvor lang tid det tar å starte applikasjonen, endre modell, endre møbel og skifte trekk/treverk. Ved oppstart av kllientapplikasjonen er det merkbar stor forskjell mellom .NET Remoting og Java RMI. Web Service brukte lengst tid av alle. De samme forholdene ble oppdaget ved endring av møbel og modell, men ved endring av trekk/treverk var der ingen merkbar forskjell mellom noen av teknologiene.

Java RMI bruker mye lengre tid på oppstarten av klienten enn noen av konkurrentene. Som sett ut fra testresultatene bruker Java RMI vesentlig lengre tid på å opprette et forhandlerobjekt enn .NET, noe som er det første som skjer når man kjører applikasjonen. Dette kan være årsaken til at Java RMI bruker lengre tid på oppstarten av klienten enn noen av de andre. Ved endring av produktbilde brukte fremdeles Java RMI

lengre tid selv om Java RMI er raskere til å overføre bilder. Dette kan skyldes at .NET har tilgang til lavere nivås grafikkontrollere i Windows enn hva Java RMI har, men det forsvarer ikke hvorfor Java RMI bruker like lang tid under Linux. Vi har sett flere artikler som omhandler Java GUI og det diskuteres hvorfor det grafiske grensesnittet har lavere responstid enn andre teknologier. Java hadde fra starten av et stort fokus på Applets som er beregnet for nettlesere, noe som medførte et lavt fokus på utvikling av GUI til klientapplikasjoner. Mange er av den oppfatning at dette har ført til at Java applikasjoner tar lang tid å laste og at oppdateringsfrekvens på brukergrensesnittet er lav. [17]

Utifra egne evalueringer konkluderer vi med at det grafiske grensesnittet i .NET oppnår en bedre brukeropplevelse enn i Java.

8.2.1 Hvilken innvirkning gav implementasjonen av caching?

Vi implementerte funksjonalitet for caching av bilder på klienten, ved å holde på bildene i cache for raskest mulig fremvisning. Klienten cacher opptil 500 bilder og beholder de mest fremviste bildene. Bildefilene har en størrelse på ca 800 kB.

På alle klienter ser vi merkbart at det går fortere å vise frem bilder som ligger i cache enn bilder som må overføres fra tjeneren. Siden klienten bare henter bilder lokalt er det naturlig at det går fortere, men det som er interessant er at effekten av at bildevisningen er så mye raskere gjør at man får en bedre brukeropplevelse.

Vi kan konkludere med at det er hensiktsmessig å implementere en løsning for caching av de mest benyttede bildene, da dette gir en merkbar positiv effekt på brukeropplevelsen.

Kapittel 9

Oppsummering og konklusjon

9.1 Konklusjon

En del av hovedmålet for denne oppgaven var å kunne utføre enhetstester mot fungerende applikasjoner og reelle testdata. Dette for å kunne gi en helhetlig vurdering av hvilken mellomvareteknologi som egner seg best til utvikling av en distribuert interaktiv produktkatalog. Med det som grunnlag utførte vi tester mot applikasjoner som benytter seg av mellomvareteknologiene .NET Remoting, Java RMI og Web Services og kan konkludere med følgende:

Gjennom en helhetlig vurdering av effektivitet basert på våre testresultater og grafisk grensesnitt konkluderer vi med at .NET Remoting over TCP/BIN er det beste valget. Den er overlegen med tanke på tidsbruk ved oppretting av et objekt med XML håndtering, og henter ett bilde raskest av samtlige mellomvarer på en ADSL linje som er reellt for en distribuert produktkatalog. Samtidig viser vår evaluering av de grafiske grensesnittene at .NET brukergrensesnittene er raskere enn Java grensesnittene.

9.2 Evaluering av eget arbeid

I denne oppgaven har vi dekt aspektene ved de problemstillingene som ble presentert i innledningen, og har på denne måten nådd vårt hovedmål om å avdekke hvilken mellomvareteknologi som er mest effektiv for bruk i en distribuert interaktiv produktkatalog.

9.2.1 Design og implementasjon

Utgangspunktet for oppgaven var å teste mot applikasjoner med fungerende hovedfunksjonalitet og reelle testdata, derfor har vi lagt ned en betydelig mengde arbeid i selve designet og implementasjonen. Vi har også forsøkt å holde designet av applikasjonene så

identiske som mulig, blant annet ved og la samtlige applikasjoner benytte XML som lagringsformat for metadata, siden det er en åpen standard som er plattformuavhengig. Ved å benytte XML som lagringsformat, er evalueringer i stor grad basert på den respektive mellomvarens håndtering av XML som format. Vi er av den oppfatning at håndtering av andre datalager kan gi andre utfall, men ut fra vår konkrete casebeskrivelse og gitte rammer har det ikke vært hensiktsmessig å utprøve andre alternativer.

9.2.2 Tester og resultater

Ved å skreddersy våre egne tester og skape egne testmiljøer som er likt for alle teknologiene, har testene ikke blitt påvirket av noen eksterne faktorer. Testresultatene har dermed kun blitt basert på teknologiene i seg selv.

Testeresultatene fra enhetstestene har gitt oss mulighet til å dekke emnene i problemstillingen. De gir et godt bilde av hvilken innvirkning båndbredde og hardware har å si for effektiviteten av viktig funksjonalitet i en distribuert produktkatalog.

Implementasjonene av grafisk grensesnitt mot hver av teknologiene har vært til stor hjelp for å kunne trekke en helhetlig konklusjon. En slik type applikasjon krever et velfungerende GUI for å kunne tilfredsstille krav hvor sluttbrukerne blir stadig mer utålmodige. De vil ikke vente mange sekunder på et svar. De forlanger lynrask og mest mulig presis respons fra systemet uansett hvor stor datamengden er. Dette er en stor utfordring som applikasjonsutviklere står overfor.

9.3 Tidligere arbeid

Til forprosjektoppgaven som ble utført høsten 2005 [4], baserte vi våre resultater på undersøkelser utført av andre. Et gjennomgående problem som vi oppdaget ved de utførte testene var at de på urettmessig vis favoriserte den ene teknologien foran den andre, enten som følge av egne interesser og/eller at de ikke dekte alle teknologiene som har vært relevant i denne oppgaven.

Under arbeidet med denne oppgaven har vi tatt opp igjen tidligere evaluerte tester samt sett på noen nye. Nedenfor nevner vi noen av de og hvilken relevans de har hatt for vårt arbeid.

Comparing Microsoft .NET Pet Shop Performance and Scalability to the Sun Java Pet Store
Denne testen er på samme måte som vår, en sammenligning av to applikasjoner som er implementert i henholdsvis .NET og Java [15]. Testen er utført i flere versjoner og er mye omtalt blant utviklere av distribuerte applikasjoner, men har et annet fokus enn hva vi har hatt i denne oppgaven.

Comparison of Web Services, Java–RMI, and CORBA service implementations
Denne testen tar for seg Web Services, Java RMI og CORBA, og viser blant annet hvor mange pakker og bytes som overføres i de respektive teknologiene. Den tar også for seg ytelse med hensyn på CPU-bruk. CORBA har ikke vært aktuell i denne oppgaven, men vi kan også se av denne testen at Web Service overfører mye mer pakker og data enn Java RMI. [16]

Performance Evaluation of Distributed Object Platforms for Public Information System Implementation
Denne testen legger stor vekt på måling av ytelse, overhead og skalerbarhet innenfor Java RMI, .NET Remoting og Web Services [23]. Overhead er noe vi har sett på i denne oppgaven og sammenlignet med denne testen har vi oppnådd forskjellige resultat. Denne testen hadde kommet frem til at .NET Remoting TCP-BIN hadde minst overhead og Java RMI og Web Service hadde omtrent likt men mye mer enn .NET. Som sett ut fra testresultatene i denne oppgaven har Java RMI minst overhead og .NET Remoting TCP-BIN hadde litt mer, men Web Service hadde klart mest overhead sammen med .NET Remoting HTTP-SOAP. Dette skyldes formateringsmekanismene som lager mye overhead ved SOAP formatering.

Java RMI and .Net Remoting Performance Comparison
Denne testen sammenligner ytelse innenfor Java RMI og .NET Remoting i tre scenarioer, hvor to er overføringstester med henholdsvis lite (*int* parameter) og mye data (1MB). Teknologiene er basert på HTTP protokollen [47]. Problemstillingene i denne testen har mange likheter med vårt formål i denne oppgaven; ytelse ved overføring. Testresultatene er også lik, da vi i denne oppgaven også har konkludert med at Java RMI er overlegen på ren overføring.

9.4 Videreføring av oppgaven

Effektivitet i mellomvareteknologier som skal benyttes i en distribuert produktkatalog er bare en av faktorene for en endelig løsning. For å unngå misbruk av tjeneren er det viktig at brukerne blir identifisert og autorisert for den informasjonen de skal ha tak i. Det er viktig at andre salgssteder/kjeder ikke får tilgang til sensitivt materiale som blir oppbevart ved tjeneren, som f.eks. innpris hos en annen forhandler. Informasjonen som skal nåes tak i må derfor både autoriseres og krypteres når den sendes over internett. Ettersom IT-kompetansen ved salgsstedene ofte er lav, er det viktig at brukeridentifiseringsprosessen er enkel. Vi vil peke på noen vesentlige aspekter som bør tas i betraktning for å kunne videreutvikle en fullgod løsning.

Emner som må taes hensyn til

- Brukeridentifisering
- Kryptering

- Flere samtidige brukere
- Forbedring av grafisk grensesnitt
- Data blir sendt til riktig mottaker

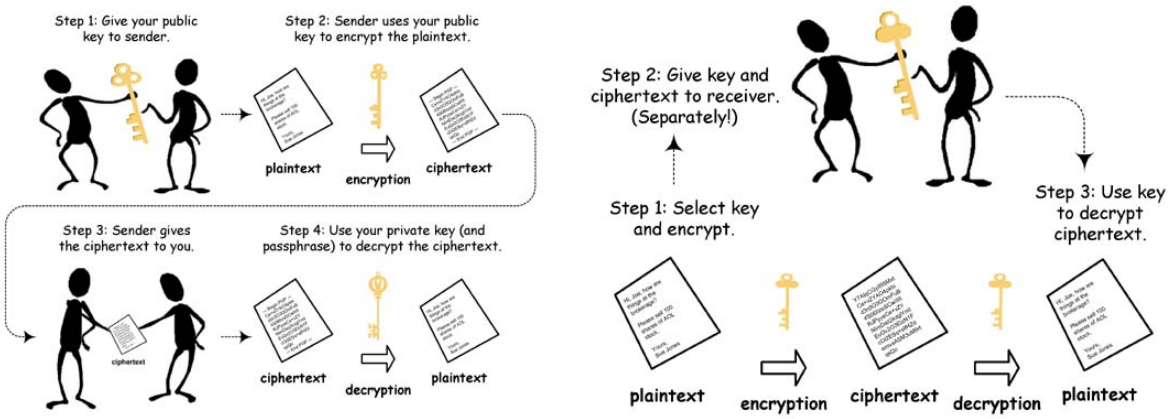
9.4.1 Brukeridentifisering

For å sikre at en bruker har tilgang til tjenestene som tjeneren tilbyr samt at brukeren får den informasjonen som er beregnet for han, må brukeren identifisere seg selv. Dette kan designes og implementeres på en rekke måter, og er et såpass stort tema at det ikke ble innbefattet i denne oppgaven. Vi vil bare peke på at det er funksjonalitet som bør tas stilling til for den videre utviklingen av en distribuert produktkatalog.

9.4.2 Kryptering

For å sikre at uønskede personer ikke får tilgang til informasjon er det viktig at informasjonen som sendes er kryptert. Kryptering er prosessen med å kode en beskjed slik at innholdet blir skjult. Det finnes mange måter å løse dette problemet på, men selve prossessen skjer oftes på én av følgende to måter:

Det er to hovedmåter å kryptere på [Figur 9.1]. I begge tilfeller brukes en nøkkel, som er parametrene for krypteringen.Uten kjennskap til denne nøkkelen er det tilnærmet umulig å fremskaffe den opprinnelige meldingen. Den ene måten å kryptere på består av en delt nøkkel. Både klienten og tjeneren bruker den samme nøkkelen for å kode og dekode meldingen. Denne nøkkelen er det kun de har kjennskap til. Den andre måten er at den som skal motta meldingen har en private key og en public key. De som skal sende en melding bruker public key til å kryptere meldingen. Meldingen kan kun dekrypteres ved hjelp av private key som det bare er mottaker som har kjennskap til.



Figur 9.1: Public- og secret nøkkel kryptering (symmetric og asymmentric kryptering [7])

Problemet med delt nøkkel er selve utvekslingen av nøkkelen. For at uønskede personer ikke skal få tak i nøkkelen må utvekslingen gjøres sikkert, noe som er vanskelig over et nettverk. Med en private/public key kreves to sett med nøkler ettersom det ikke er mulig å dekryptere en melding med public key som er kryptert med en private key.[40]

9.4.3 Skalerbarhet

Vi har ikke utført noen tester på hvordan mellomvarene vil takle en skalering i antall brukere, noe som er en viktig egenskap for et distribuert system. Når tjeneren har flere samtidige brukere, er det er viktig at de ikke oppstår problemer med tilgang til ressurser, samtidig som tjeneren må behandle de forskjellige brukerne på en effektiv måte.

For å unngå en deadlock må det holdes kontroll over hvem som holder en ressurs til enhver tid. Når tjeneren skal oppdatere biblioteket, er det viktig at ingen klienter bruker ressursen på det gitte tidspunkt, ettersom oppdateringen da vil bli ugyldig. Hvis noen skal ha skrivetilgang til en fil, må ingen andre ha tilgang til filen på samme tidspunkt. Hvis en prosess krever to eller flere ressurser, må det sikres mot at to uavhengige operasjoner ikke holder fast på én ressurs og venter på en annen, samtidig som en annen prosses gjør det motsatte. Dette kalles en deadlock.

9.4.4 Forbedring av grafisk grensesnitt

Å designe et lettfattelig og brukervennlig grensesnitt er uhyre viktig der bruk av bilder, video og animasjoner vil være nødvendigheter for å tilfredsstille dagens krevende brukermasse av PC-brukere.

Figur 9.2: Test av WinFX for grafisk grensesnitt

Vi har sett på neste generasjons utviklingsmuligheter av grafisk grensesnitt for applikasjoner utviklet under .NET rammeverk. Microsoft har sluppet en beta av sitt nye .NET API kalt WinFX eller .NET Framework 3.0. Der har vi sett på API for GUI kalt

Windows Presentation Foundation. Dette er basert på vektorgrafikk og et Markup Language kalt XAML. Etter som dette bare er en beta har vi ikke valgt å vurdere det som et alternativ i oppgaven, men vil se på det som en mulighet for videreutvikling.

Vi laget et enkelt grensesnitt [Figur 9.2] mot vår applikasjon og er meget imponert av muligheten som ligger her for bruk av animasjon, video og vektorgrafikk. Vi ville helt klart vurdert sterkt å teste ut bruk av WinFX så snart stabile APIer er klar.

9.4.5 Data blir sendt til riktig mottaker

Når en bruker spør etter informasjon må tjeneren levere informasjonen til den brukeren som spurte, og ingen andre. Kryptert informasjon som sendes kan ikke utnyttes av andre, men for å opprettholde essensen i et distribuert system er det viktig at det er riktig bruker som mottar ønsket informasjon.

Bibliografi

[1] *www.idi.ntnu.no/emner/dif8914/ kompendium-2004/ppt-2004/m1-wolfgang.ppt*, Sist aksessert: 29.05.2006.

[2] Inc. Altova. *http://www.altova.com/*, Sist aksessert 21.05.2006.

[3] AutomatedBuildings.com. *http://automatedbuildings.com/news/jan02/art/alc/alc.htm*, Sist aksessert: 26.05.2006.

[4] Anders Beite, Christian Jensen, and Camilla Jacobsen. Infrastruktur for møbelkatalog. 2005.

[5] Peter Bye Chris Britton. Middleware: A history of objects, components, and the web. *http://www.awprofessional.com/articles/article.asp?p=345781&rl=1*, Sist aksessert: 27.05.2006.

[6] The University Of British Columbia. A brief history of distributed computing. *http://www.ugrad.cs.ubc.ca/ cs416/X/Notes/01%20-%20Sep%2010y/02.history.html*, Sist aksessert 24.05.2006.

[7] The A3C Connection. Pretty good personal privacy. *http://www.uic.edu/depts/accc/newsletter/adn26/pgpp.html#3*, Sist aksessert: 01.06.2006.

[8] Microsoft Corporation. .net remoting architecture. *http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconnetremotingarchitecture.asp*, Sist aksessert: 01.06.2006.

[9] Microsoft Corporation. Web services specifications. *http://msdn.microsoft.com/webservices/webservices/understanding/specs/default.aspx*, Sist aksessert: 01.06.2006.

[10] Chris Dix Roger Kovack Jonathan Pinnock Jeff Rafter David Hunter, Kurt Cagle. *Beginning XML*. Wrox, second edition, 2001.

[11] Mark Egan. *The Executive guide to Information Security.* Addison textendash Wesley Professional, 2004.

[12] Hege Fjeld. Html og xml. *http://snipsnap.nr.no/projectlink/space/HTML+og+XML*, Sist aksessert: 29.05.2006.

[13] Tim Kindberg George Coulouris, Jean Dollimore. *Distributed Systems: Concepts and Design.* 2005.

[14] gotdotnet. Comparing microsoft .net pet shop performance and scalability to the sun java pet store. *http://www.gotdotnet.com/team/compare/petshopperf.aspx*, Sist aksessert: 29.05.2006.

[15] gotdotnet. Microsoft® .net vs.sun® microsystem's j2eetm. *http://www.gotdotnet.com/team/compare/Nile%20Benchmark%20Results.doc*, Sist aksessert: 29.05.2006.

[16] N.A.B Gray. Comparison of web services, java-rmi, and corba service implementations. *http://mercury.it.swin.edu.au/ctg/AWSA04/Papers/gray.pdf*, Sist aksessert: 29.05.2006.

[17] Anil Hemrajani. Does sun understand gui design? *http://www.javaworld.com/javaworld/jw-05-2001/jw-0504-soapbox.html*, Sist aksessert: 11.06.2006.

[18] Imran Hussain. J2ee vs dot–net –a technical comparison. *http://www.cjjug.com/CJJUG–J2EE_ Vs_ DotNet.pdf*, Sist aksessert: 01.06.2006.

[19] Sun Microsystems Inc. Java remote method invocation (rmi). *http://java.sun.com/j2se/1.4.2/docs/guide/rmi/index.html*, Sist aksessert: 01.06.2006.

[20] Christian Damsgaard Jensen. Secure software architectures. *http://www.ifi.uib.no/konf/nwper98/papers/jensen.ps*, Sist aksessert: 01.06.2006.

[21] Jera Design John Brewer. Extreme programming faq. *http://www.jera.com/techinfo/xpfaq.html*, Sist aksessert 23.05.2006.

[22] JUnit.org. *http://www.junit.org/index.htm*, Sist aksessert: 24.05.2006.

[23] Ivan Skuliber Mario Stefanec, Sinisa Srbljic. Performance evaluation of distributed object platforms for public information system implementation. *http://ris.zemris.fer.hr/mediator/documents/Paper_ PerformanceEvaluationOf DistributedObjectPlatformsForPublicInformationSystemImplementation.pdf*, Sist aksessert: 29.05.2006.

[24] Marjan Hericko Ivan Rozman Ivan Vezocnik Matjaz B. Juric, Bostjan Kezmah. Java rmi, rmi tunneling and web services comparison and performance analysis. *http://portal.acm.org/citation.cfm?id=997146*,
Sist aksessert: 01.06.2006.

[25] Microsoft. .net remoting architecture. *http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconnetremotingarchitecture.asp*,
Sist aksessert: 08.06.2006.

[26] Judith M.Myerson. *The Complete Book Of Middleware*. 2002.

[27] Kenneth Hansen Morten Hertzum. Toningsuddannelsen i interaktive medier. *http://www.ruc.dk/upload/application/pdf/4cb5f381/Toningbeskr_2003E.pdf*,
Sist aksessert: 09.06.2006.

[28] Shawn Van Ness. Copying, cloning, and marshalling in .net. *http://www.ondotnet.com/pub/a/dotnet/2002/11/25/copying.html*,
Sist aksessert: 30.05.2006.

[29] IDI NTNU. Java rmi. *http://www.idi.ntnu.no/emner/tdt4190/programvare/javaRMI.html*,
Sist aksessert: 24.05.2006.

[30] NUnit.org. *http://www.nunit.org/*,
Sist aksessert: 24.05.2006.

[31] NUnit.org. Nunit quick start. *http://www.nunit.org/index.php?p=quickStart&r=2.2.7*,
Sist aksessert: 24.05.2006.

[32] Piet Obermeyer and Jonathan Hawkins. Microsoft .net remoting: A technical overview. *http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dndotnet/html/hawkremoting.asp*,
Sist aksessert: 01.06.2006.

[33] ObjectWare. Mellomvare introduksjon. *http://www.objectware.no/templates/Page.aspx?id=2109*,
Sist aksessert: 27.05.2006.

[34] University of Alabama at Birmingham. *http://webapp.lab.ac.uab.edu/wiki/mlist/index.php/MlistGlossary*,
Sist aksessert: 27.05.2006.

[35] Elise Peterson. Time for a truce. *http://www.ftponline.com/wss/2003_02/magazine/departments/ednote/*,
Sist aksessert: 01.06.2006.

[36] Charles Petzold. *Programming Windows With C#*. Microsoft Press, 2001.

[37] Apache Ant Project. *http://ant.apache.org/*,
Sist aksessert: 24.05.2006.

[38] Scene7. *http://www.scene7.com/solutions/platform/image_server.asp*, Sist aksessert: 24.05.2006.

[39] Scene7. *http://www.scene7.com/solutions/platform/render_server.asp*, Sist aksessert: 24.05.2006.

[40] RSA Security. What are some of the more popular techniques in cryptography? *http://www.rsasecurity.com/rsalabs/node.asp?id=2158*, Sist aksessert: 01.06.2006.

[41] Rouge Wave Software. An introduction to web services. *http://www.roguewave.com/news/whitepapers/LEIFWP3.pdf*, Sist aksessert: 09.06.2006.

[42] Spot-Media. Was ist ein webservice? *http://www.spot-media.de/index.php/article_product/detail/445*, Sist aksessert: 08.06.2006.

[43] Mark Strawmyer. Serialization/deserialization in .net. *http://www.developer.com/net/csharp/article.php/10918_3110371_2*, Sist aksessert: 30.05.2006.

[44] UIO. *http://termvakt.uio.no/wiki/index.php?n=Windows.Eclipse*, Sist aksessert 21.05.2006.

[45] Wikipedia. Distributed computing. *http://en.wikipedia.org/wiki/Distributed_system*, Sist aksessert: 09.06.2006.

[46] Wikipedia. *http://en.wikipedia.org/wiki/Middleware*, Sist aksessert: 27.05.2006.

[47] Frank Koopmans Willem Elbers and Ken Madlener. Java rmi and .net remoting performance comparison. *http://www.niii.ru.nl/ marko/onderwijs/oss/RMI_and_Remoting.pdf*, Sist aksessert: 12.06.2006.

[48] Yong Zhu. Building interactive distributed applications. *http://www-static.cc.gatech.edu/ yongzhu/Cs7001/cs7001_2_report.html*, Sist aksessert: 08.06.2006.

Ordliste

.NET Framework et rammeverk for å utvikle og anvende applikasjoner, 12

.NET Remoting teknologi for å utvikle distribuerte applikasjoner, 19, 49

ADSL Asymmetric Digital Subscriber Line, 56

Arkitektur struktur og organisering av et system, 11, 31

Båndbredde mål for kapasiteten til en kommunikasjonskanal, 2, 56

C# programmeringsspråk fra Microsoft, 12

Caching , 25, 41, 44

CMS Customer Management System, 18

Connectionless kommunikasjon mellom to endepunkter i et nettverk der en beskjed kan bli sendt uten tidligere oppsett , 22

Distribuert objekt softwaremoduler som er designet til å jobbe sammen men som befinner seg på ulike datasystemer, 2

Distribuert system maskinvare- og programvarekomponenter lokalisert i et nettverk av datamaskiner som kommuniserer sine handlinger ved å sende meldinger, 16

Feedback tilbakemelding, 7

Flash teknologi utviklet for å produsere applikasjoner med vektorgrafikk og animasjoner, 7

Formatering organisering av informasjon etter gitte regler, 12

GUI graphical user interface grafisk brukergrensesnitt, 9, 11, 41, 45

hardware mekaniske magnetiske elektroniske og elektriske komponenter som tilsammen utgjør en datamaskin, 2

HTTP HyperteText Transfer Protocol - protokoll for å overføre filer over internett, 12

IIS Internet Information Services - webtjener utviklet av Microsoft, 12

Java RMI Java Remote Method Invocation - gjør det mulig for en prosess å kalle metoder på et objekt som er opprettet av en annen prosess, 20, 50

JUnit rammeverk for enhetstesting, 58

Klassebibliotek en samling av klasser for gjenbruk i objektorientert programmering, 38

Kryptering prosessen med å kode en beskjed slik at innholdet blir skjult, 76

LAN Local Area Network, 56

Marshalling prosessen å serialisere objekter, 24

Metadata data om data, 8, 31

Mobile klienter , 23

NUnit rammeverk for enhetstesting, 57

ORB Object Request Broker, 18

OTM Object Transaction Monitor, 19

Pakke blokk med data som blir sendt over nettverket, 14

Relasjonsdatabase databasesystem som er organisert i henhold til relasjoner mellom dataelementer, 32

Rendring prosessen for å produsere pixlene i et bilde fra en beskrivelse av komponenten, 8

RPC Remote Procedure Call, 18

Serialisering prosessen å lagre et objekt til et lagringsmedium eller å sende det over en nettverksforbindelse som en serie av bytes, 24

Serialisering prosessen å lagre et objekt til et lagringsmedium, 45

Skalerbarhet egenskap til å takle en økning i antall brukere uten at det påvirker ytelsen, 77

SOAP Simple Object Access Protocol - en XML basert protokoll for å utveksle struktrert data, 12

Stateless beskriver om en applikasjon er designet for å huske en eller flere etterfølgende hendelser i en gitt sekvens av interaksjoner, 22

Stored Procedure et sett av SQL statements som er lagret i databasen, 43

TCP Transport Control Protocol - en transportprotokoll som sender pakker mellom applikasjoner, 12

TP Transaction Processing, 18

Trigger en hendelse som blir utført når den detekterer en spesifikk operasjon, 43

URI Uniform Resource Identifier - unik identifikator til en ressurs, 25

Web Services applikasjonsservice som kan aksesseres ved bruk av standard webprotokoller, 22, 53

XML eXtensible Markup Language, 24, 33, 43

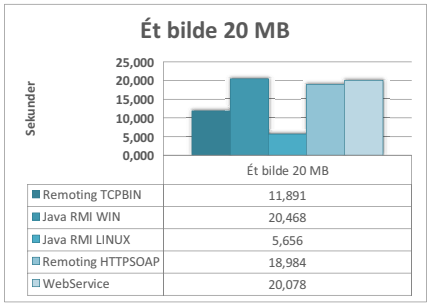
Tillegg A

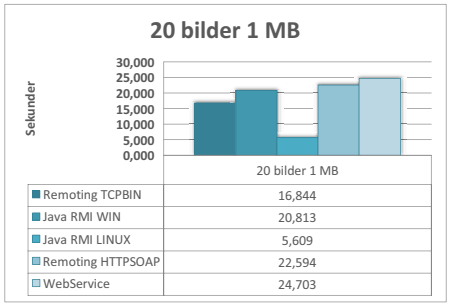
Grafisk fremstilling av testresultater

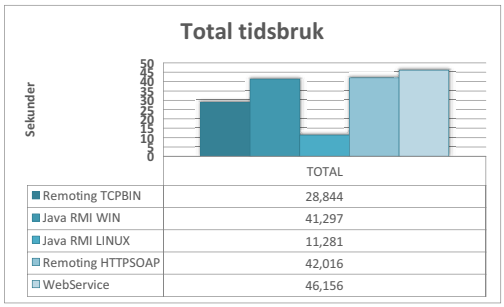
A.1 Filstørrelsetest

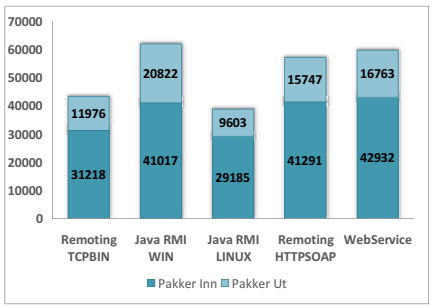
TESTRESULTATER OVERFØRING AV ÉN STOR OG TJUE SMÅ BILDEFILER
SERVER: ***Athlon 1,53GHZ 512 MB RAM, Windows XP/Mandriva Linux***
CLIENT: ***Sempron 3100+, 1,81 GHZ, 1GB RAM, Windows XP***

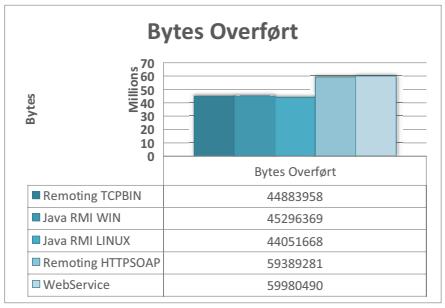
RUN 1	*Ét bilde 20 MB*	*20 bilder 1 MB*	*TOTAL*	*Pakker Inn*	*Pakker Ut*	*Bytes Overført*
Remoting TCPBIN	11,891	16,844	28,844	31218	11976	44883958
Java RMI WIN	20,468	20,813	41,297	41017	20822	45296369
Java RMI LINUX	5,656	5,609	11,281	29185	9603	44051668
Remoting HTTPSOAP	18,984	22,594	42,016	41291	15747	59389281
WebService	20,078	24,703	46,156	42932	16763	59980490



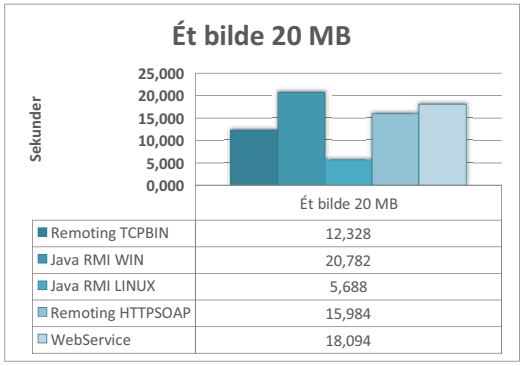


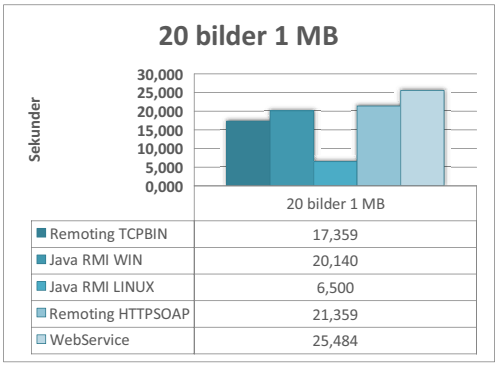


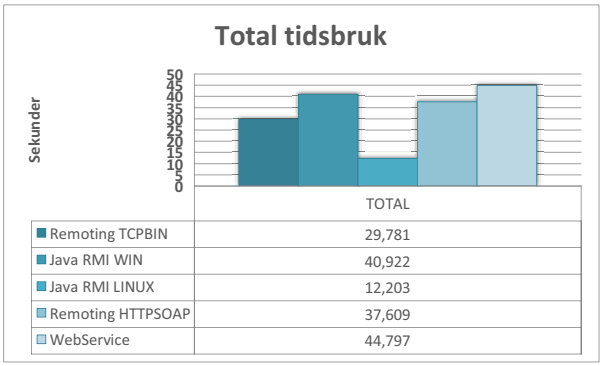


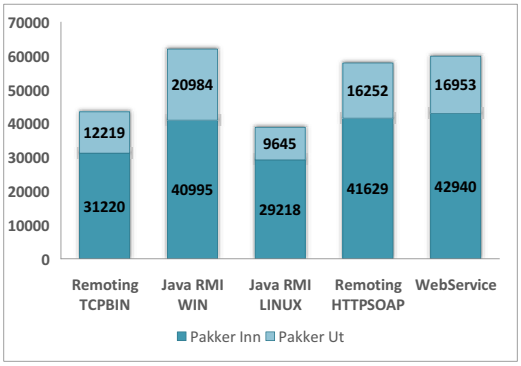


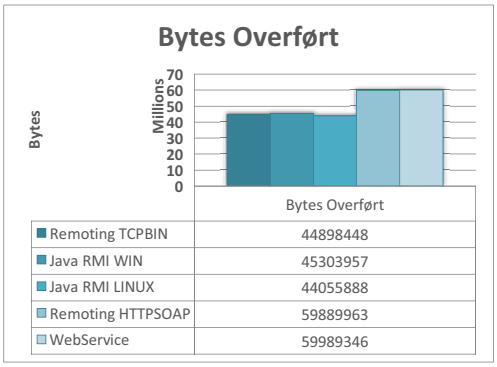
RUN 2	Ét bilde 20 MB	20 bilder 1 MB	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	12,328	17,359	29,781	31220	12219	44898448
Java RMI WIN	20,782	20,140	40,922	40995	20984	45303957
Java RMI LINUX	5,688	6,500	12,203	29218	9645	44055888
Remoting HTTPSOAP	15,984	21,359	37,609	41629	16252	59889963
WebService	18,094	25,484	44,797	42940	16953	59989346





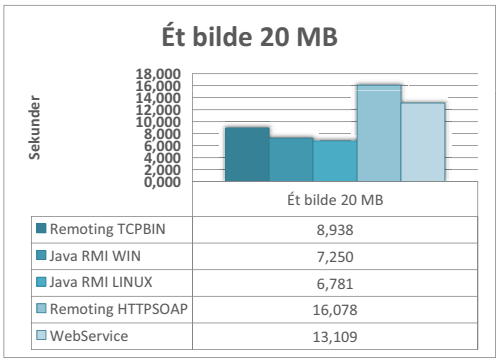


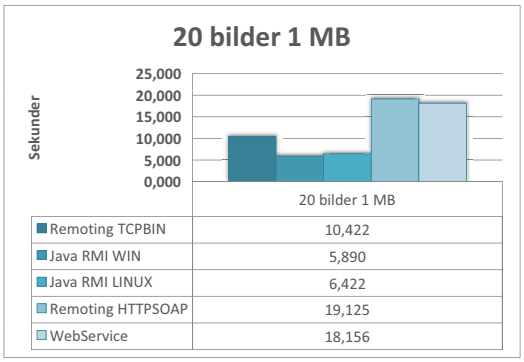


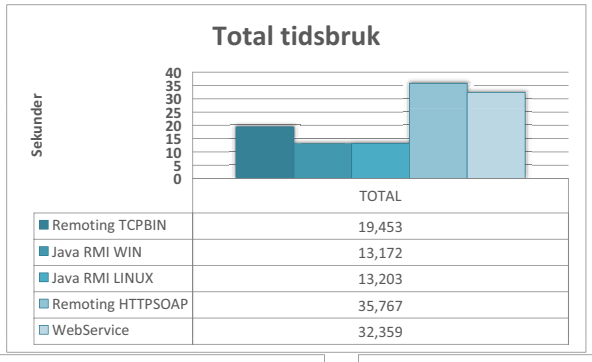


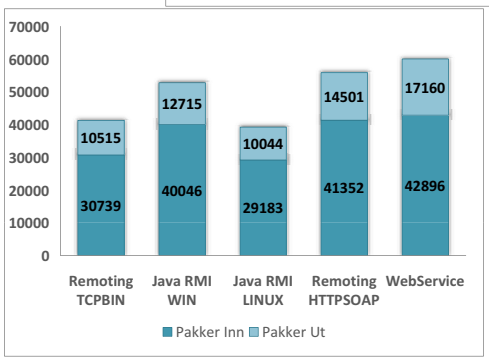
TESTRESULTATER OVERFØRING AV ÉN STOR OG TJUE SMÅ BILDEFILER
SERVER: ***Pentium 4 3GHZ, 1GB RAM, Windows XP/Mandriva Linux***
CLIENT: ***Sempron 3100+ 1,8 GHZ, 1GB RAM, Windows XP***

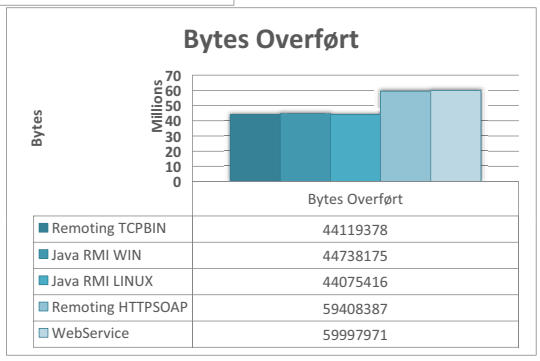
RUN 1	Ét bilde 20 MB	20 bilder 1 MB	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	8,938	10,422	19,453	30739	10515	44119378
Java RMI WIN	7,250	5,890	13,172	40046	12715	44738175
Java RMI LINUX	6,781	6,422	13,203	29183	10044	44075416
Remoting HTTPSOAP	16,078	19,125	35,767	41352	14501	59408387
WebService	13,109	18,156	32,359	42896	17160	59997971



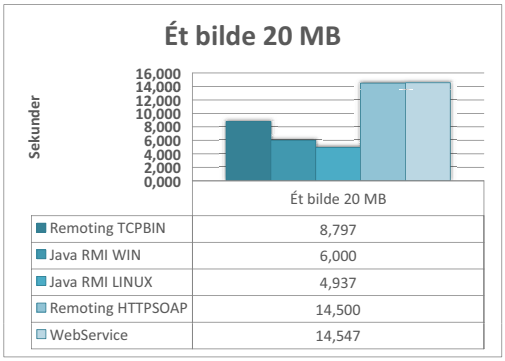


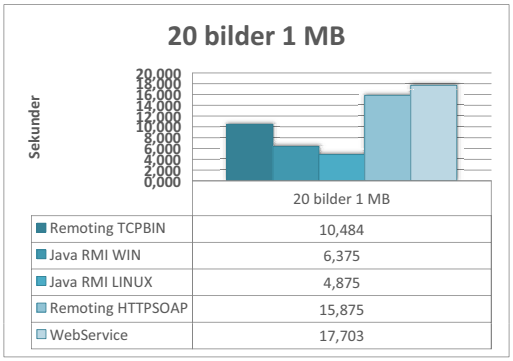


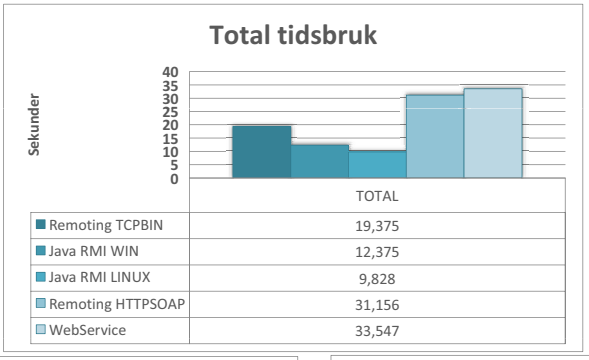


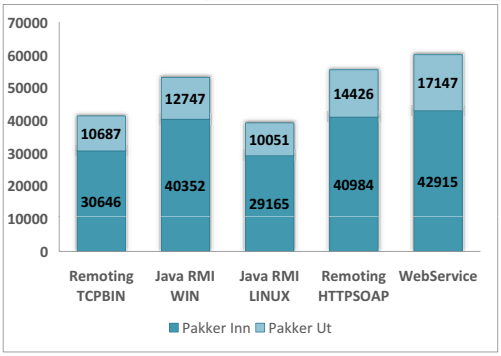


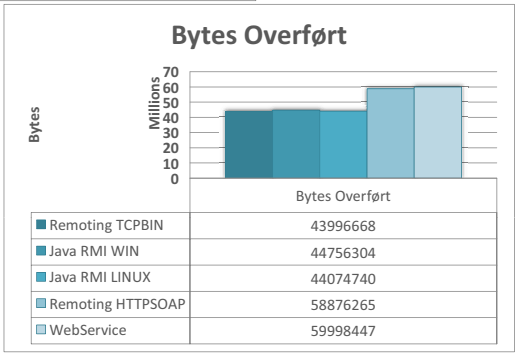
RUN 2	Et bilde 20 MB	20 bilder 1 MB	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	8,797	10,484	19,375	30646	10687	43996668
Java RMI WIN	6,000	6,375	12,375	40352	12747	44756304
Java RMI LINUX	4,937	4,875	9,828	29165	10051	44074740
Remoting HTTPSOAP	14,500	15,875	31,156	40984	14426	58876265
WebService	14,547	17,703	33,547	42915	17147	59998447





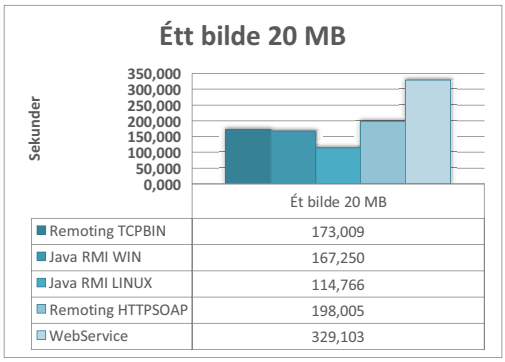


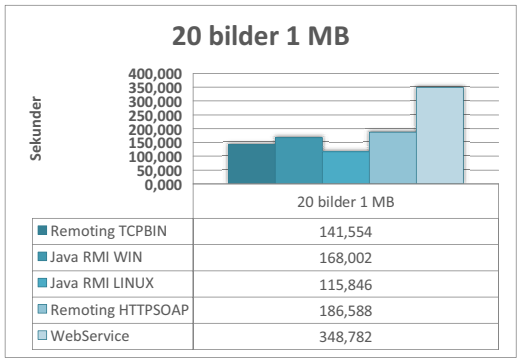


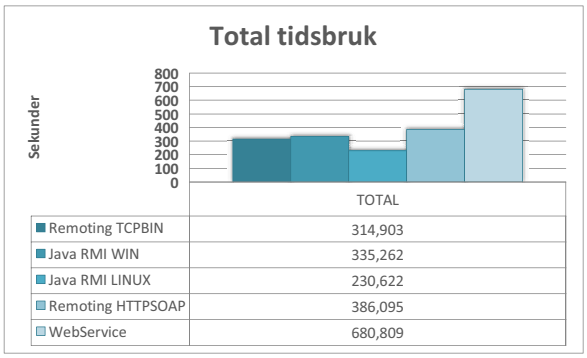


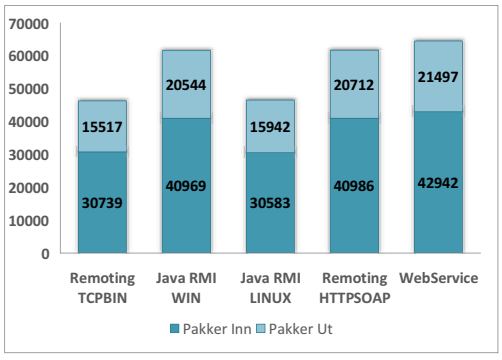
TESTRESULTATER OVERFØRING AV ÉN STOR OG TJUE SMÅ BILDEFILER
SERVER: ***Athlon 1,53GHZ 512 MB RAM, Windows XP/Mandriva Linux***
CLIENT: ***Pentium 3, 1GHZ, 512 MB RAM, Windows 2003 Server***

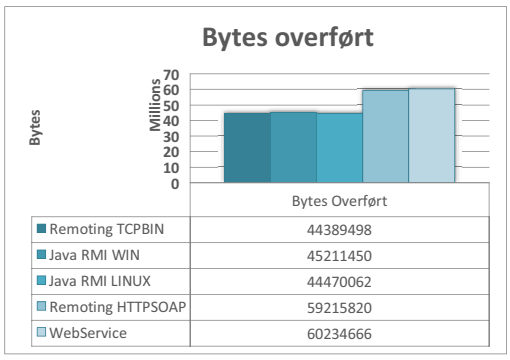
RUN 1	*Ét bilde 20 MB*	*20 bilder 1 MB*	*TOTAL*	*Pakker Inn*	*Pakker Ut*	*Bytes Overført*
Remoting TCPBIN	173,009	141,554	314,903	30739	15517	44389498
Java RMI WIN	167,250	168,002	335,262	40969	20544	45211450
Java RMI LINUX	114,766	115,846	230,622	30583	15942	44470062
Remoting HTTPSOAP	198,005	186,588	386,095	40986	20712	59215820
WebService	329,103	348,782	680,809	42942	21497	60234666



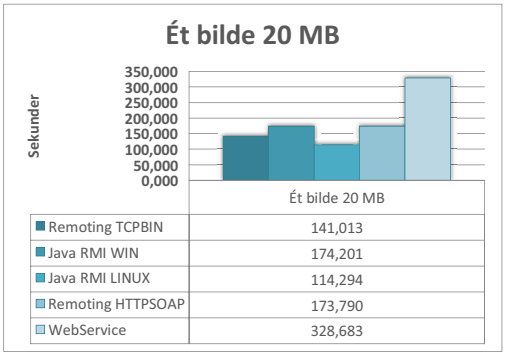


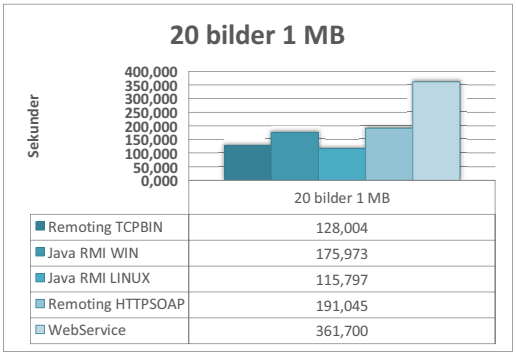


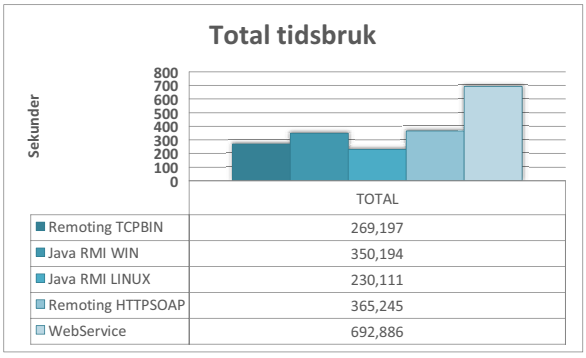


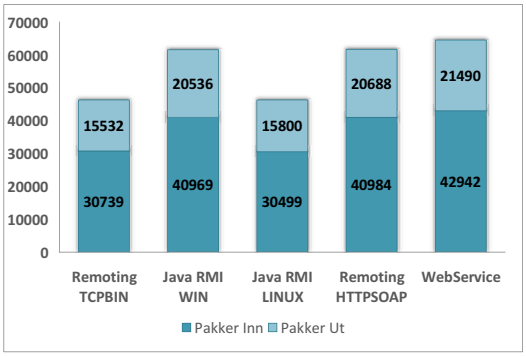


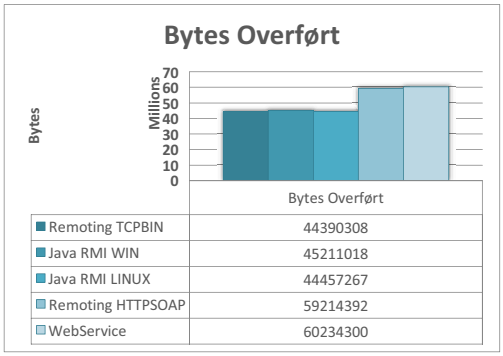
RUN 2	Et bilde 20 MB	20 bilder 1 MB	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	141,013	128,004	269,197	30739	15532	44390308
Java RMI WIN	174,201	175,973	350,194	40969	20536	45211018
Java RMI LINUX	114,294	115,797	230,111	30499	15800	44457267
Remoting HTTPSOAP	173,790	191,045	365,245	40984	20688	59214392
WebService	328,683	361,700	692,886	42942	21490	60234300





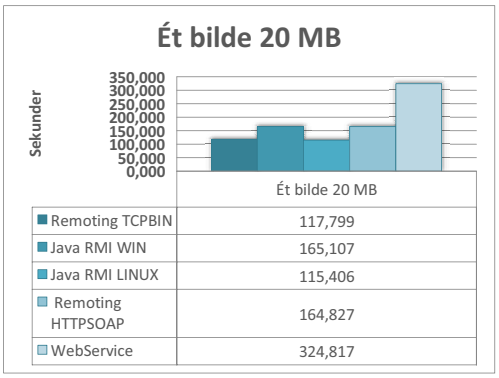


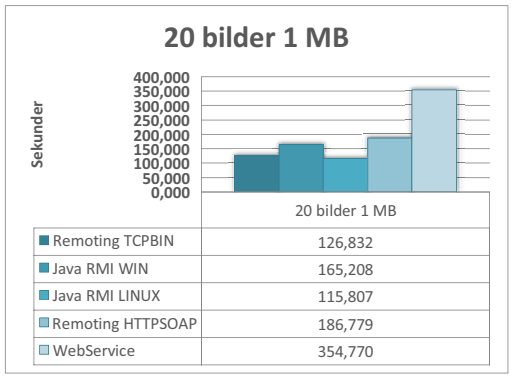


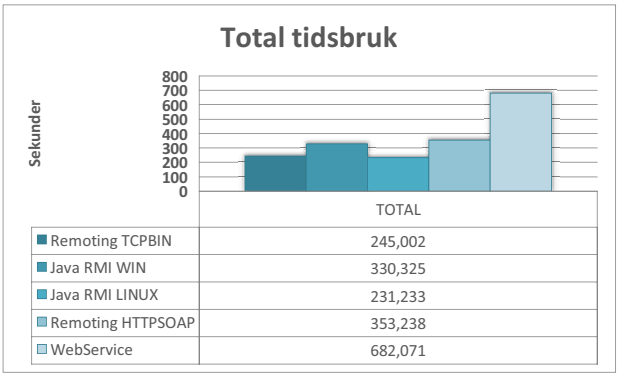


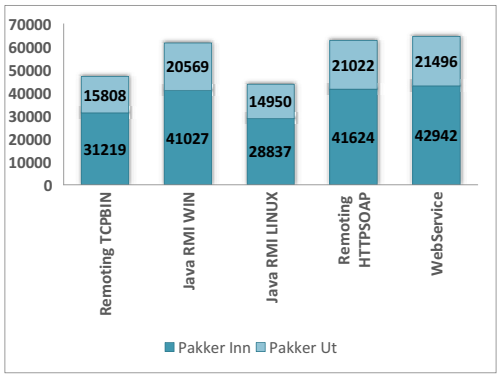
TESTRESULTATER OVERFØRING AV ÉN STOR OG TJUE SMÅ BILDEFILER
SERVER: ***Pentium 4, 3GHZ, 1GB RAM, Windows XP/Mandriva Linux***
CLIENT: ***Pentium 3, 1GHZ, 512 MB RAM, Windows 2003 Server***

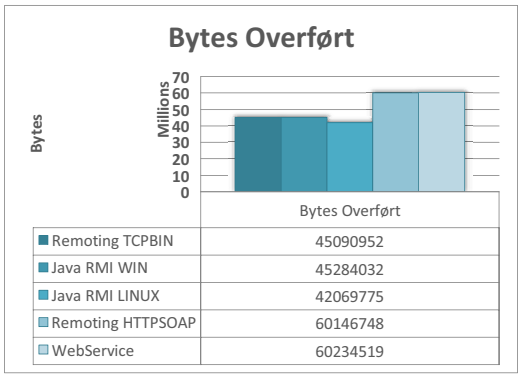
RUN 1	*Ét bilde 20 MB*	*20 bilder 1 MB*	*TOTAL*	*Pakker Inn*	*Pakker Ut*	*Bytes Overført*
Remoting TCPBIN	117,799	126,832	245,002	31219	15808	45090952
Java RMI WIN	165,107	165,208	330,325	41027	20569	45284032
Java RMI LINUX	115,406	115,807	231,233	28837	14950	42069775
Remoting HTTPSOAP	164,827	186,779	353,238	41624	21022	60146748
WebService	324,817	354,770	682,071	42942	21496	60234519



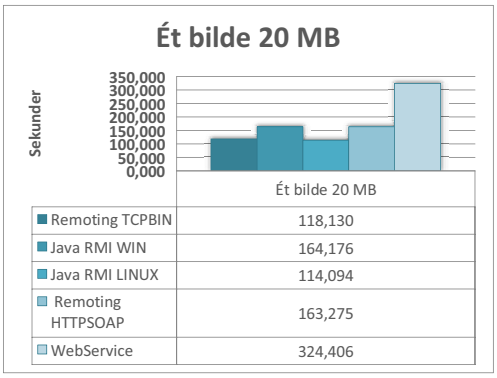


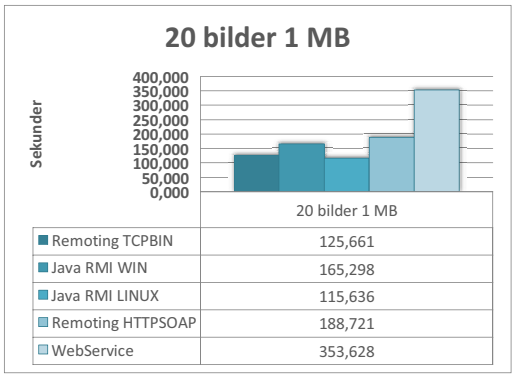


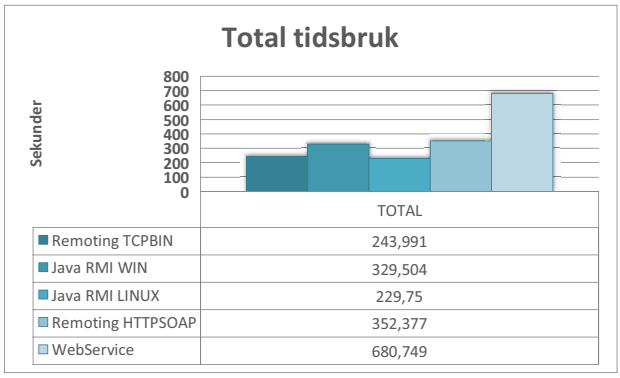


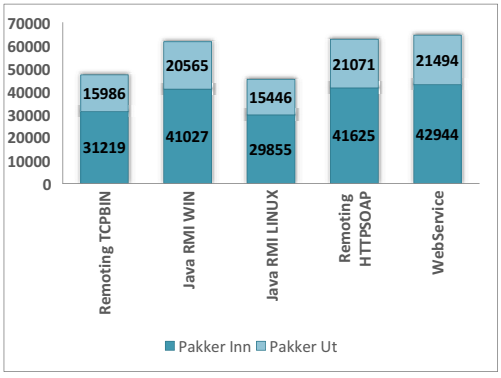


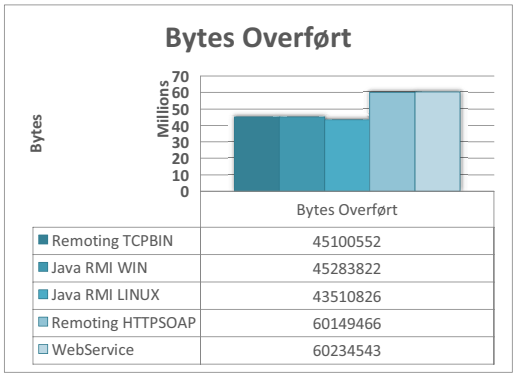
RUN 2	Ébilde 20 MB	20 bilder 1 MB	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	118,130	125,661	243,991	31219	15986	45100552
Java RMI WIN	164,176	165,298	329,504	41027	20565	45283822
Java RMI LINUX	114,094	115,636	229,75	29855	15446	43510826
Remoting HTTPSOAP	163,275	188,721	352,377	41625	21071	60149466
WebService	324,406	353,628	680,749	42944	21494	60234543







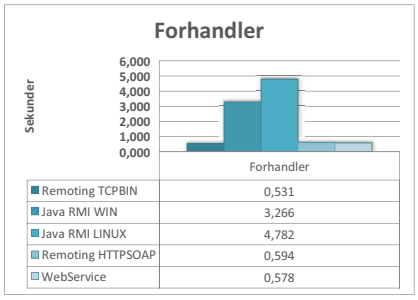


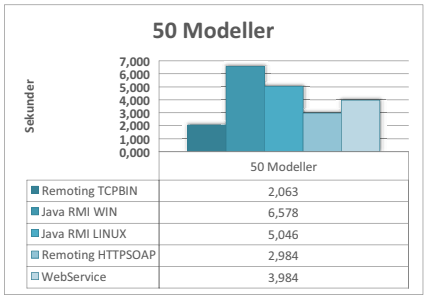


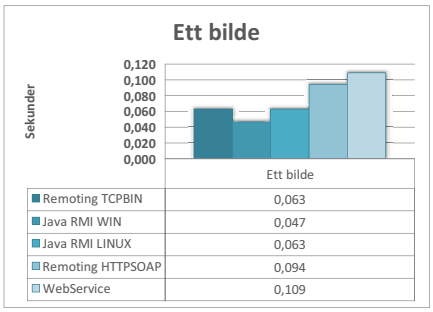
A.2 Hovedtester

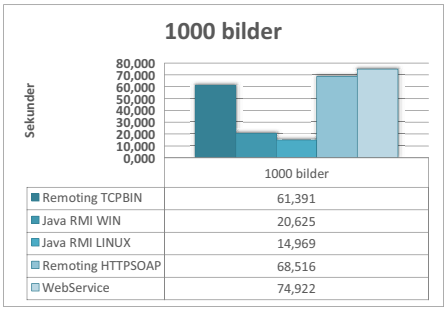
SERVER: ***Athlon 1,53GHZ 512 MB RAM, Windows XP/Mandriva Linux***
CLIENT: ***Sempron 3100+, 1,81 GHZ, 1GB RAM, Windows XP***

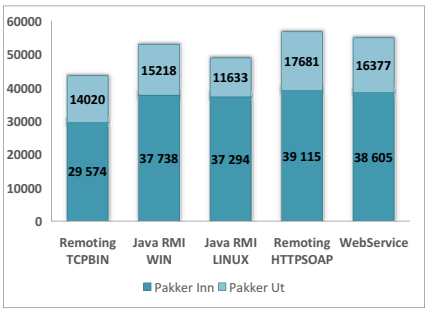
RUN 1	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	0,531	2,063	0,063	61,391	64,156	29574	14020	39550533
Java RMI WIN	3,266	6,578	0,047	20,625	30,516	37738	15218	39759880
Java RMI LINUX	4,782	5,046	0,063	14,969	24,86	37294	11633	39539758
Remoting HTTPSOAP	0,594	2,984	0,094	68,516	72,266	39115	17681	54408236
WebService	0,578	3,984	0,109	74,922	80,609	38605	16377	53440080

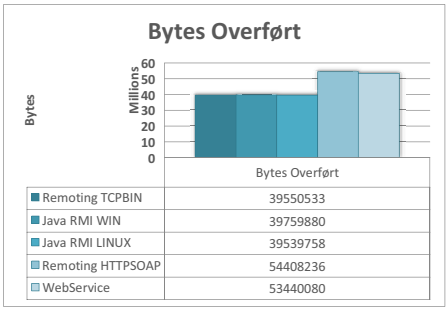




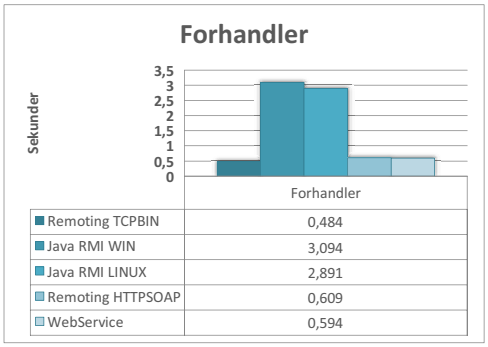


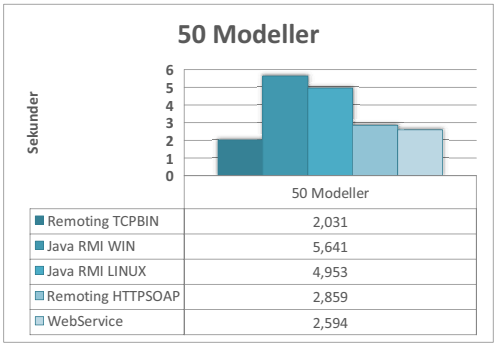


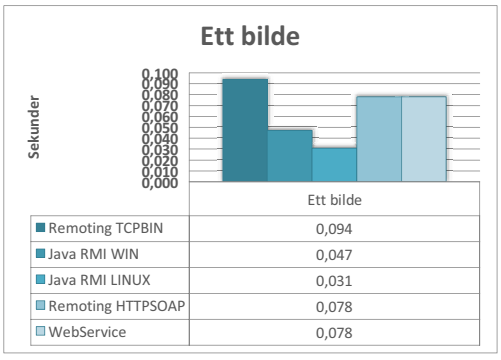


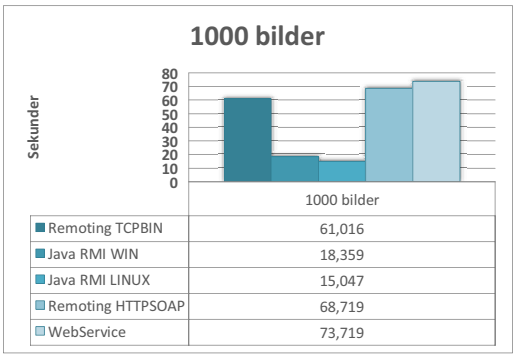


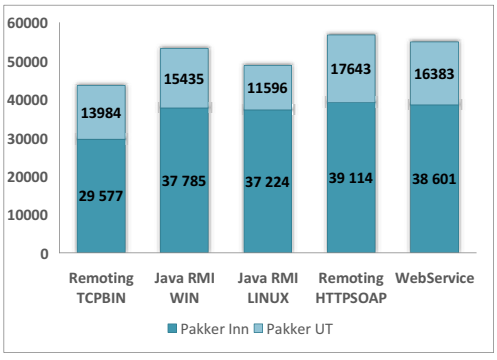
RUN 2	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker UT	Bytes Overført
Remoting TCPBIN	0,484	2,031	0,094	61,016	63,75	29577	13984	39549231
Java RMI WIN	3,094	5,641	0,047	18,359	27,156	37785	15435	39774720
Java RMI LINUX	2,891	4,953	0,031	15,047	22,922	37224	11596	39532897
Remoting HTTPSOAP	0,609	2,859	0,078	68,719	72,328	39114	17643	54405662
WebService	0,594	2,594	0,078	73,719	78,031	38601	16383	53439738

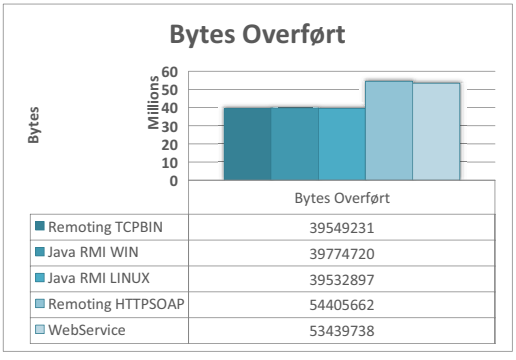






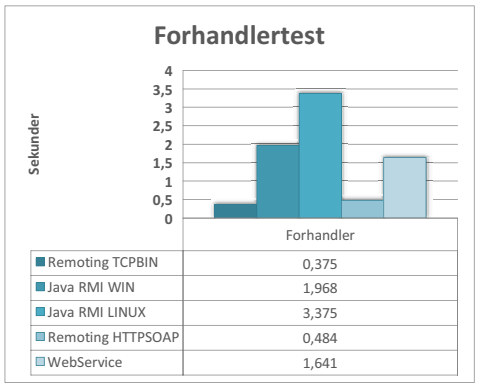


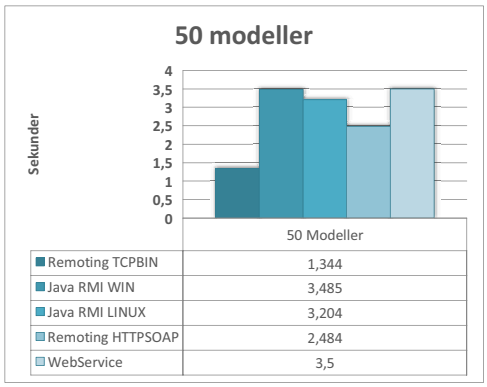


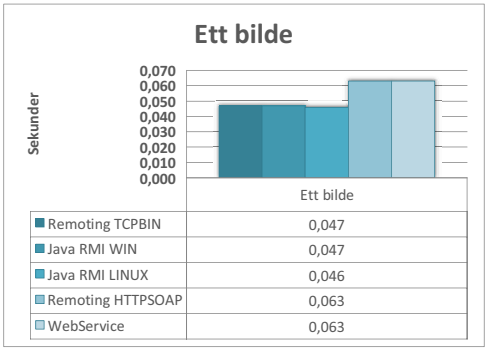


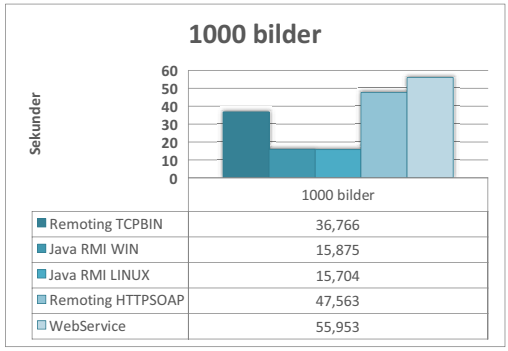
SERVER: ***Pentium 4, 3GHZ, 1GB RAM, Windows XP/Mandriva Linux***
CLIENT: ***Sempron 3100+, 1,81 GHZ, 1GB RAM, Windows XP***

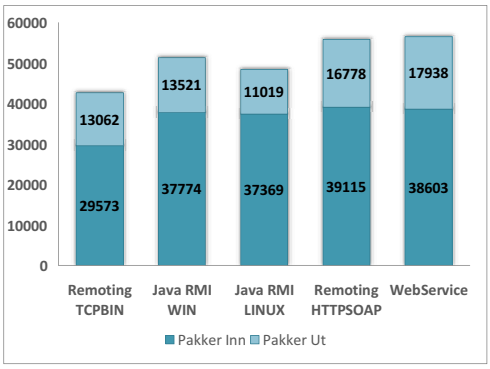
RUN 1	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	0,375	1,344	0,047	36,766	38,75	29573	13062	39504147
Java RMI WIN	1,968	3,485	0,047	15,875	21,391	37774	13521	39669652
Java RMI LINUX	3,375	3,204	0,046	15,704	22,344	37369	11019	39511181
Remoting HTTPSOAP	0,484	2,484	0,063	47,563	50,641	39115	16778	54358919
WebService	1,641	3,5	0,063	55,953	62,578	38603	17938	53518234

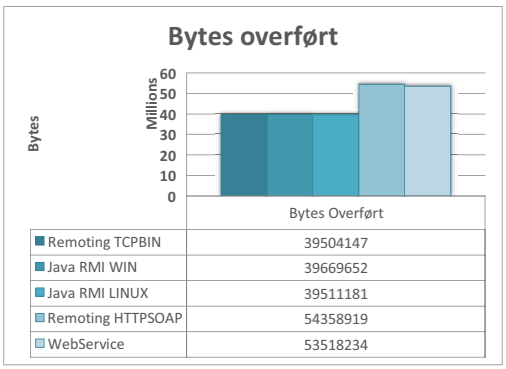




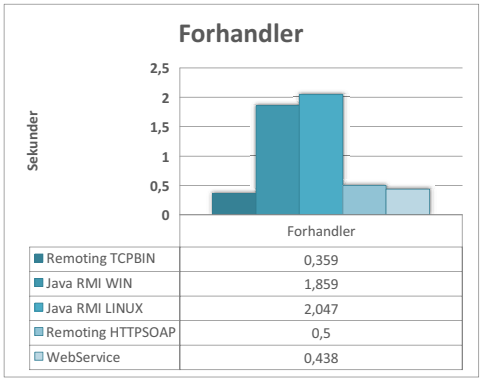


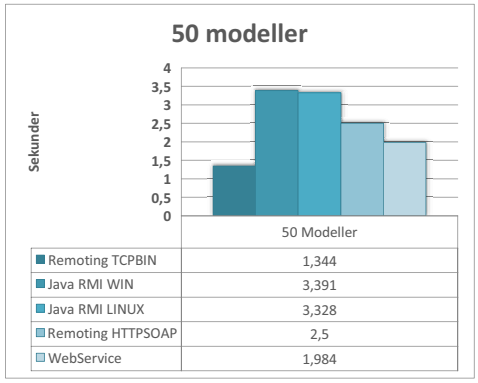


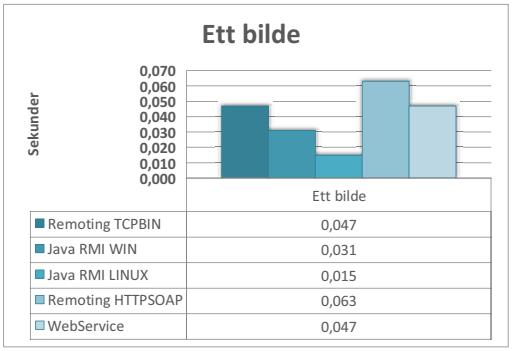


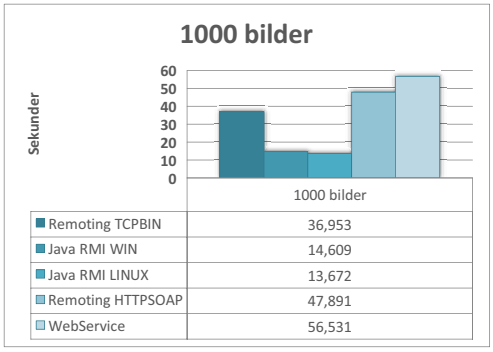


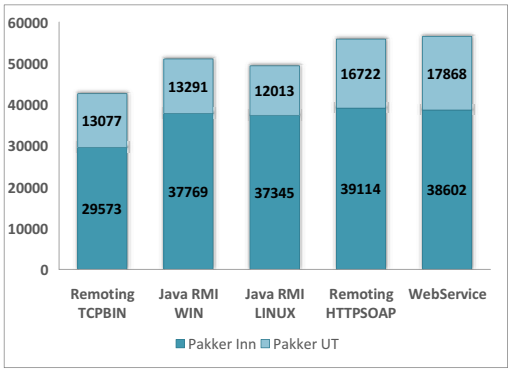
RUN 2	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker UT	Bytes Overført
Remoting TCPBIN	0,359	1,344	0,047	36,953	38,781	29573	13077	39499551
Java RMI WIN	1,859	3,391	0,031	14,609	19,89	37769	13291	39656524
Java RMI LINUX	2,047	3,328	0,015	13,672	19,602	37345	12013	39563587
Remoting HTTPSOAP	0,5	2,5	0,063	47,891	51,016	39114	16722	54355927
WebService	0,438	1,984	0,047	56,531	60,266	38602	17868	53514312

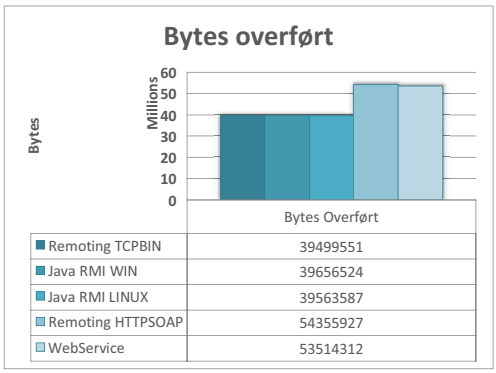








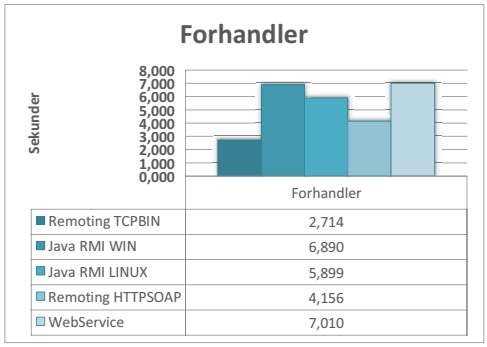


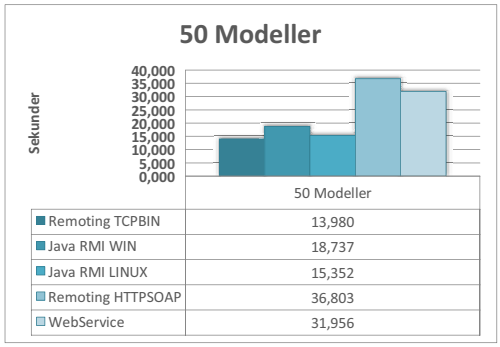


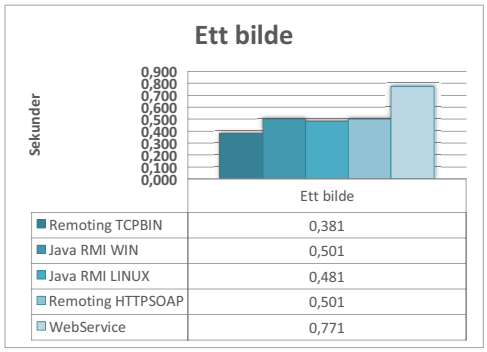
Test fra en ADSL-linje på 1,5 Mb/s mot en server som står på NTNUs nett.

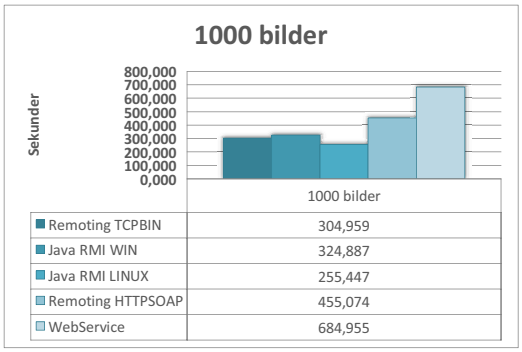
SERVER: ***Athlon 1,53GHZ 512 MB RAM, Windows XP/Mandriva Linux***
CLIENT: ***Pentium 3, 1GHZ, 512 MB RAM, Windows 2003 Server***

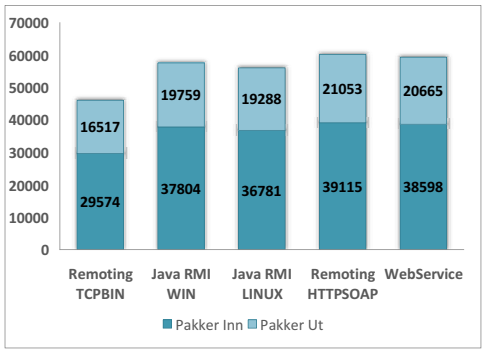
RUN 1	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	2,714	13,980	0,381	304,959	322,394	29574	16517	39685383
Java RMI WIN	6,890	18,737	0,501	324,887	351,025	37804	19759	40008399
Java RMI LINUX	5,899	15,352	0,481	255,447	277,199	36781	19288	39924200
Remoting HTTPSOAP	4,156	36,803	0,501	455,074	496,614	39115	21053	54588816
WebService	7,010	31,956	0,771	684,955	727,166	38598	20665	53670372

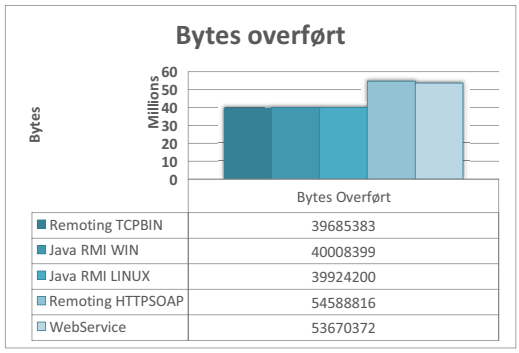




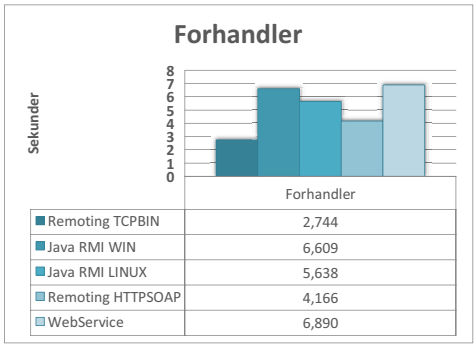


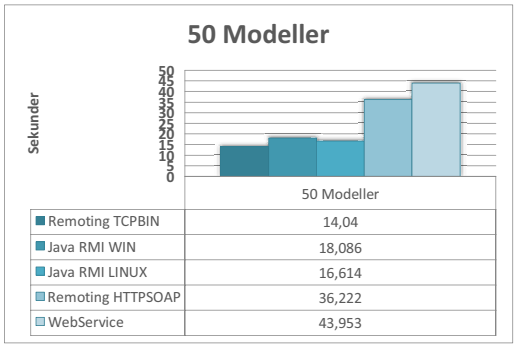


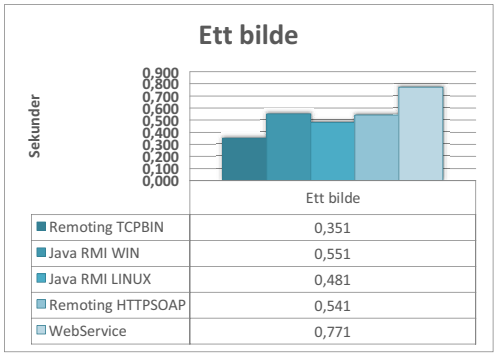


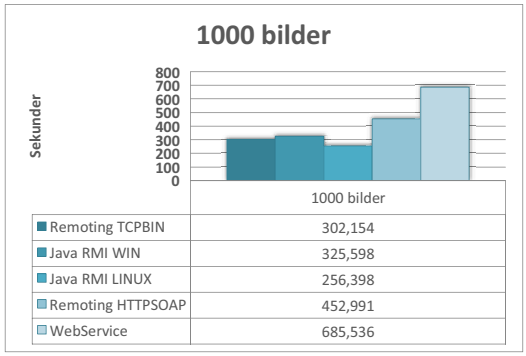


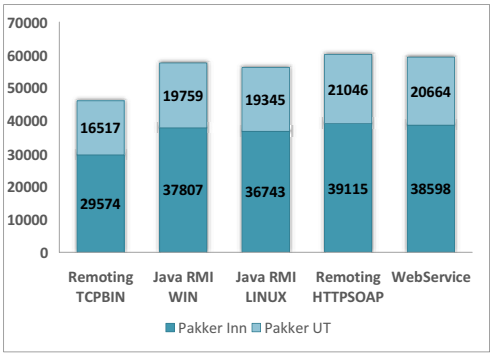
RUN 2	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker UT	Bytes Overført
Remoting TCPBIN	2,744	14,04	0,351	302,154	319,469	29574	16517	39684237
Java RMI WIN	6,609	18,086	0,551	325,598	350,874	37807	19759	40008581
Java RMI LINUX	5,638	16,614	0,481	256,398	279,141	36743	19345	39925208
Remoting HTTPSOAP	4,166	36,222	0,541	452,991	494,01	39115	21046	54588426
WebService	6,890	43,953	0,771	685,536	739,784	38598	20664	53670342

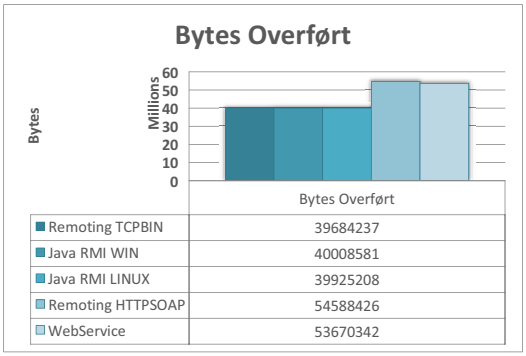








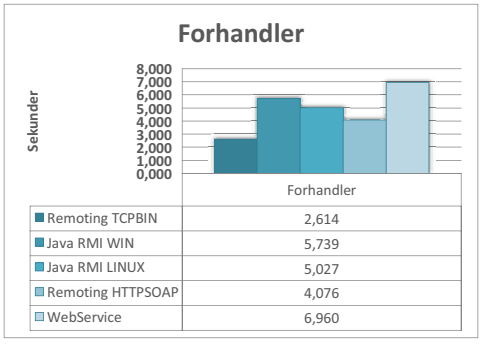


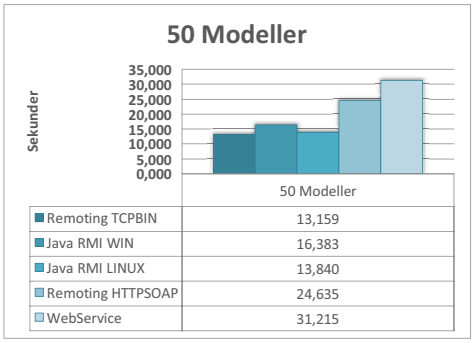


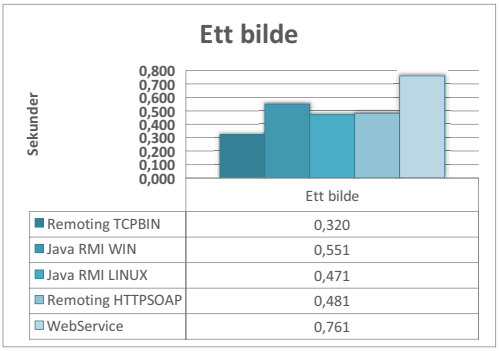
Test fra en ADSL-linje på 1,5 Mb/s mot en server som står på NTNUs nett.

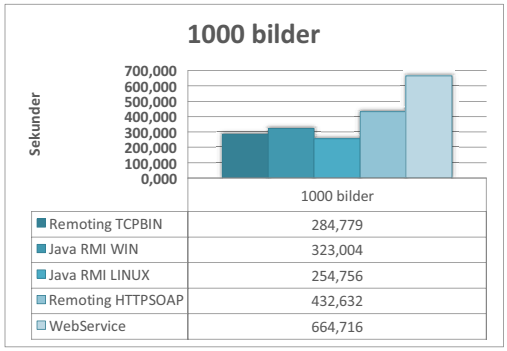
SERVER: ***Pentium 4, 3GHZ, 1GB RAM, Windows XP/Mandriva Linux***
CLIENT: ***Pentium 3, 1GHZ, 512 MB RAM, Windows 2003 Server***

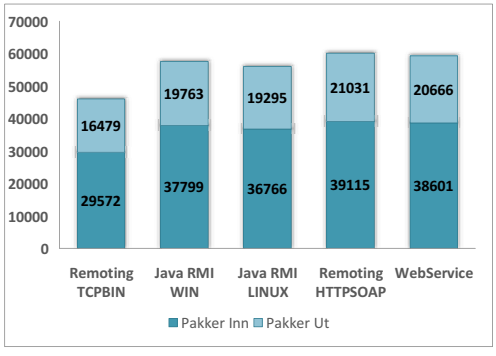
RUN 1	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	2,614	13,159	0,320	284,779	301,043	29572	16479	39683235
Java RMI WIN	5,739	16,383	0,551	323,004	345,697	37799	19763	40008311
Java RMI LINUX	5,027	13,840	0,471	254,756	274,114	36766	19295	39925355
Remoting HTTPSOAP	4,076	24,635	0,481	432,632	461,914	39115	21031	54587434
WebService	6,960	31,215	0,761	664,716	706,075	38601	20666	53666818

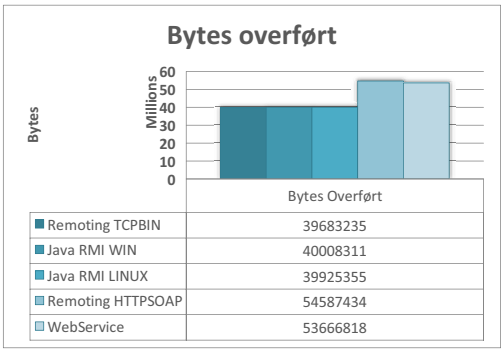




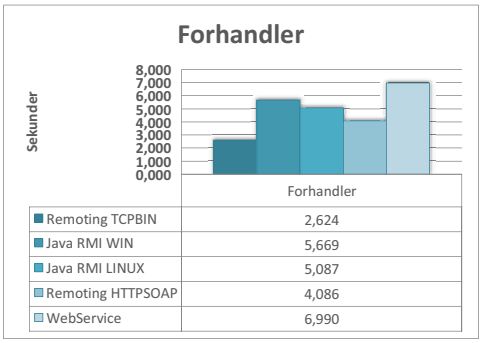


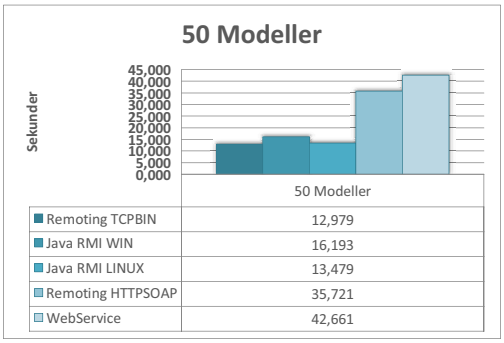


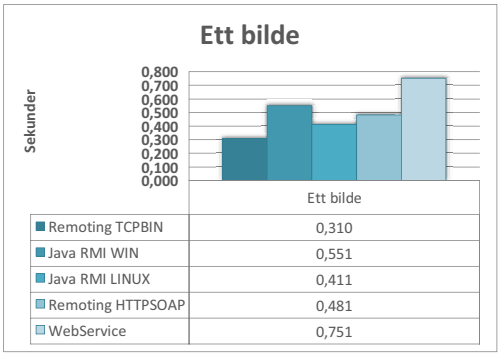


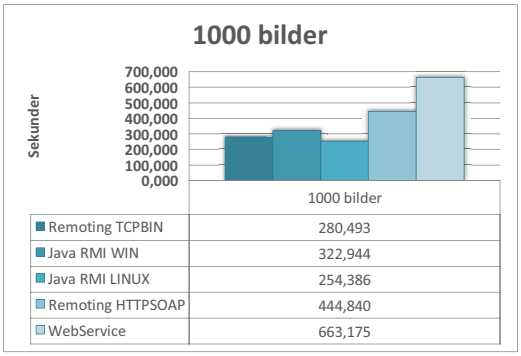


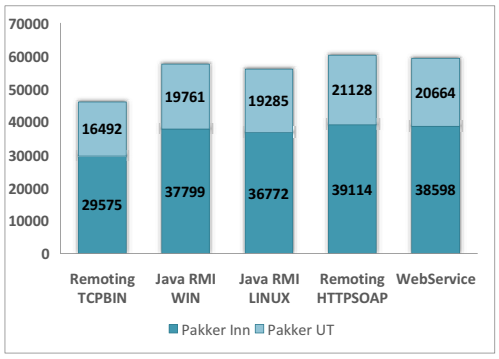
RUN 2	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker UT	Bytes Overført
Remoting TCPBIN	2,624	12,979	0,310	280,493	296,586	29575	16492	39685559
Java RMI WIN	5,669	16,193	0,551	322,944	345,387	37799	19761	40008203
Java RMI LINUX	5,087	13,479	0,411	254,386	273,383	36772	19285	39925129
Remoting HTTPSOAP	4,086	35,721	0,481	444,840	485,228	39114	21128	54592702
WebService	6,990	42,661	0,751	663,175	716,160	38598	20664	53665055

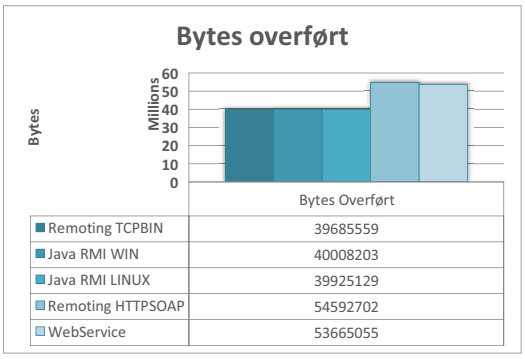












A.3 Sammenligning av hovedtestene

God HW, ADSL

RUN 2	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	2,624	12,979	0,310	280,493	296,586	29575	16492	39685559
Java RMI WIN	5,669	16,193	0,551	322,944	345,387	37799	19761	40008203
Java RMI LINUX	5,087	13,479	0,411	254,386	273,383	36772	19285	39925129
Remoting HTTPSOAP	4,086	35,721	0,481	444,840	485,228	39114	21128	54592702
WebService	6,990	42,661	0,751	663,175	716,160	38598	20664	53665055

Dårlig HW, ADSL

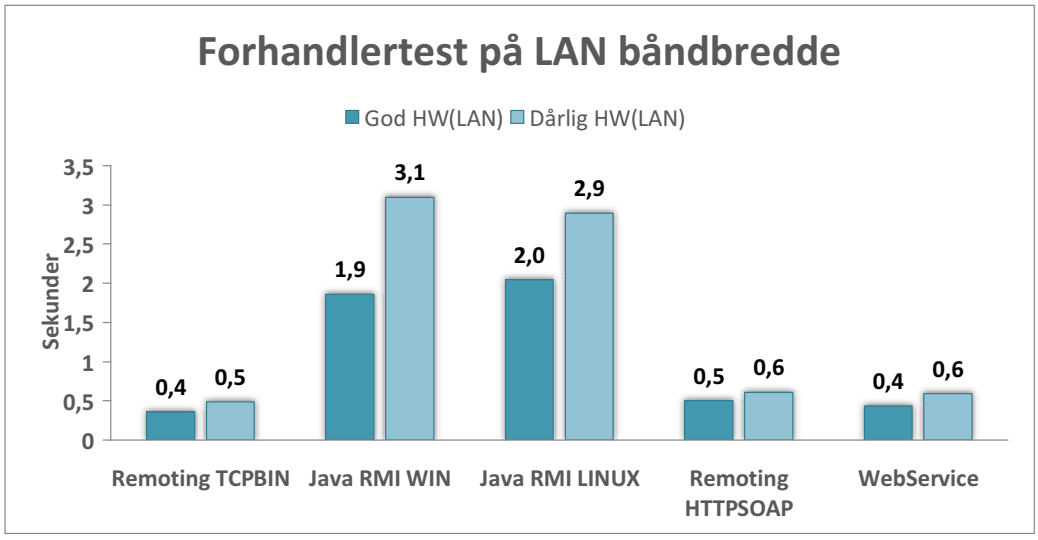
RUN 2	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	2,744	14,04	0,351	302,154	319,469	29574	16517	39684237
Java RMI WIN	6,609	18,086	0,551	325,598	350,874	37807	19759	40008581
Java RMI LINUX	5,638	16,614	0,481	256,398	279,141	36743	19345	39925208
Remoting HTTPSOAP	4,166	36,222	0,541	452,991	494,01	39115	21046	54588426
WebService	6,890	43,953	0,771	685,536	739,784	38598	20664	53670342

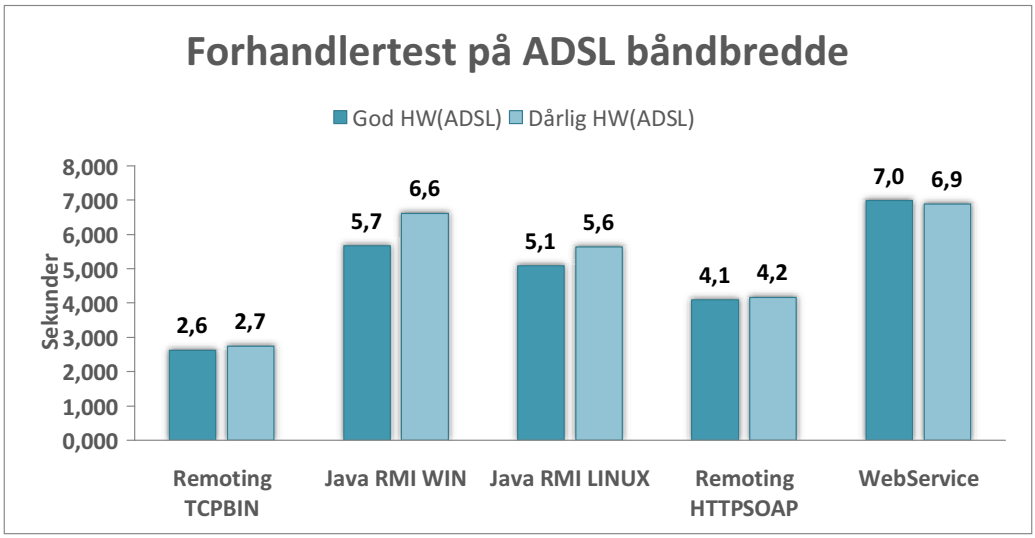
God HW, LAN

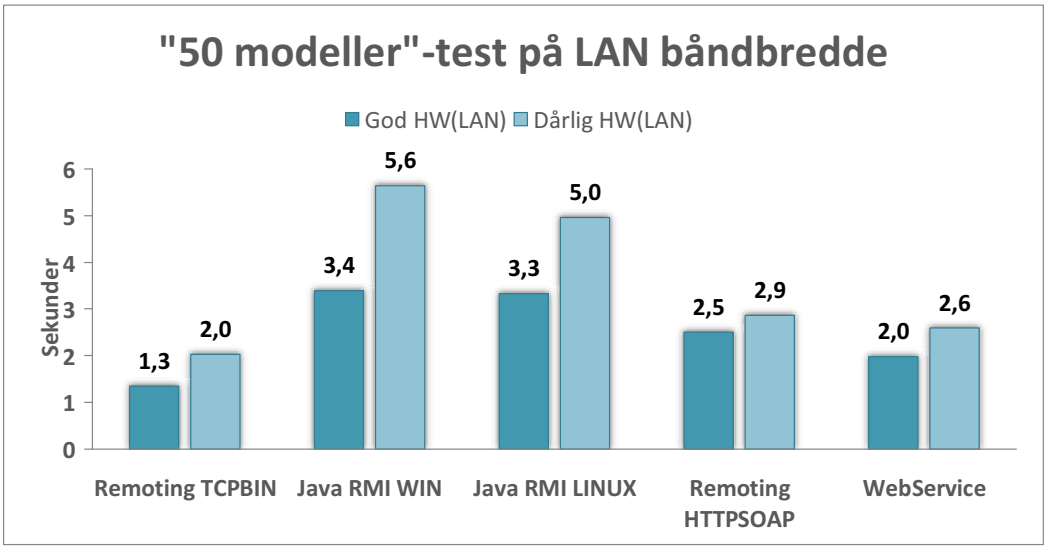
RUN 2	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	0,359	1,344	0,047	36,953	38,781	29573	13077	39499551
Java RMI WIN	1,859	3,391	0,031	14,609	19,89	37769	13291	39656524
Java RMI LINUX	2,047	3,328	0,015	13,672	19,602	37345	12013	39563587
Remoting HTTPSOAP	0,5	2,5	0,063	47,891	51,016	39114	16722	54355927
WebService	0,438	1,984	0,047	56,531	60,266	38602	17868	53514312

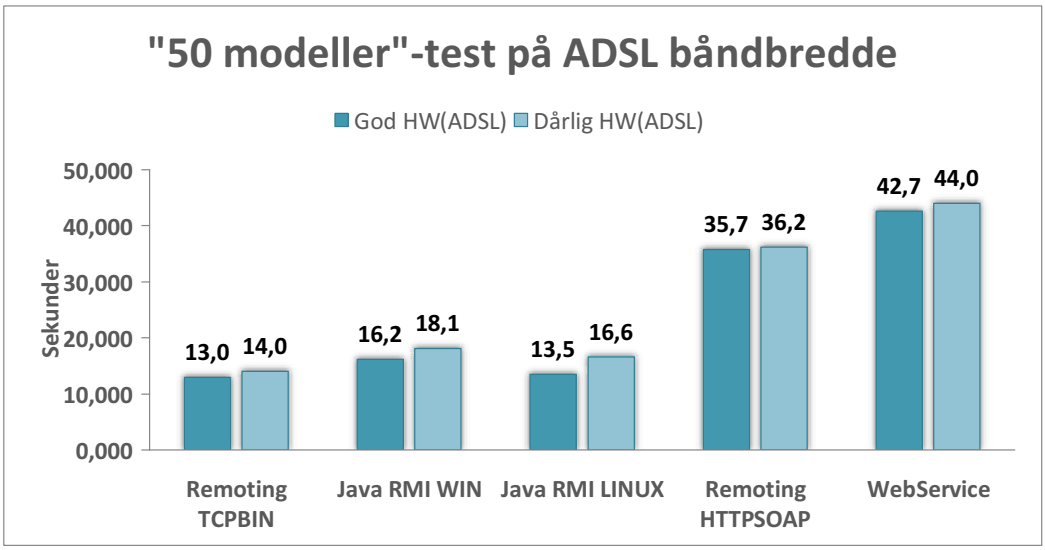
Dårlig HW, LAN

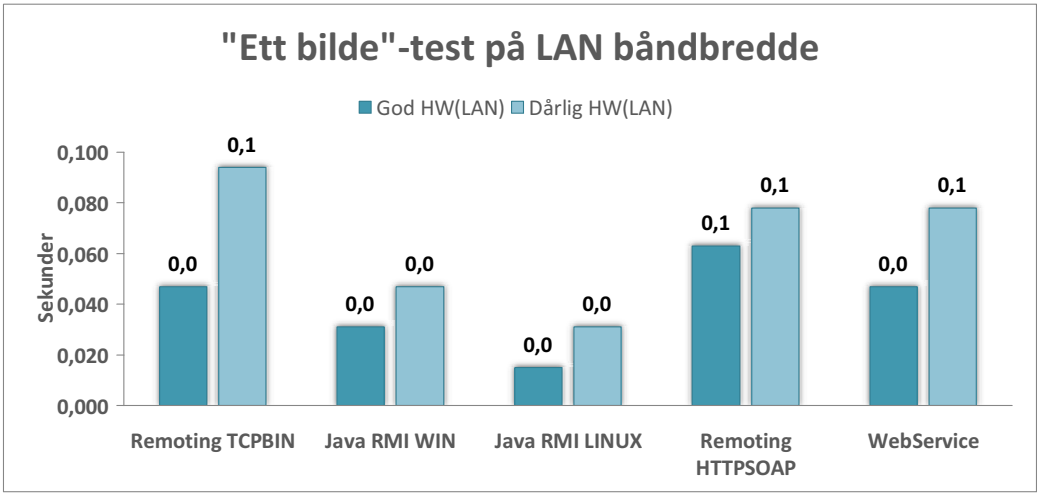
RUN 2	Forhandler	50 Modeller	Ett bilde	1000 bilder	TOTAL	Pakker Inn	Pakker Ut	Bytes Overført
Remoting TCPBIN	0,484	2,031	0,094	61,016	63,75	29577	13984	39549231
Java RMI WIN	3,094	5,641	0,047	18,359	27,156	37785	15435	39774720
Java RMI LINUX	2,891	4,953	0,031	15,047	22,922	37224	11596	39532897
Remoting HTTPSOAP	0,609	2,859	0,078	68,719	72,328	39114	17643	54405662
WebService	0,594	2,594	0,078	73,719	78,031	38601	16383	53439738

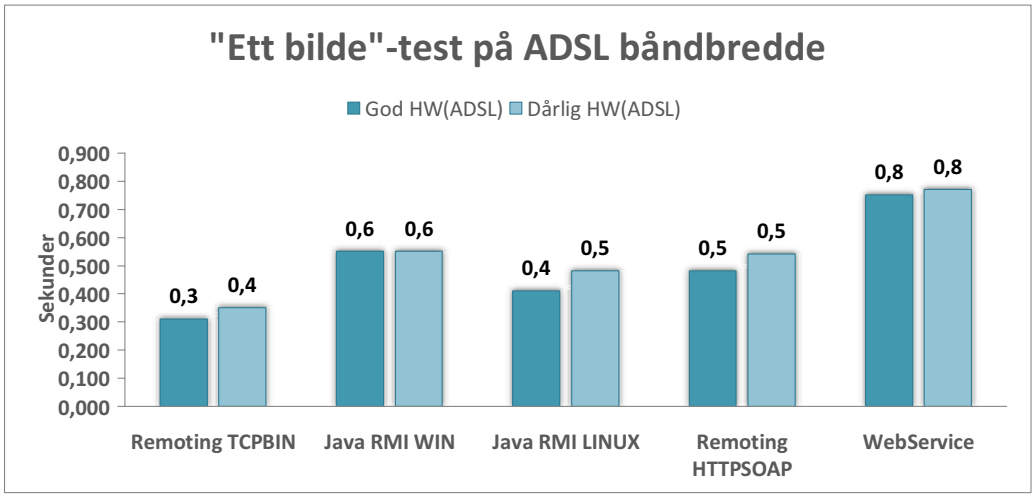

Forhandlertest på LAN båndbredde
God HW(LAN)
Dårlig HW(LAN)
Sekunder
3,5
3
2,5
2
1,5
1
0,5
0
0,4
0,5
1,9
3,1
2,0
2,9
0,5
0,6
0,4
0,6
Remoting TCPBIN
Java RMI WIN
Java RMI LINUX
Remoting HTTPSOAP
WebService

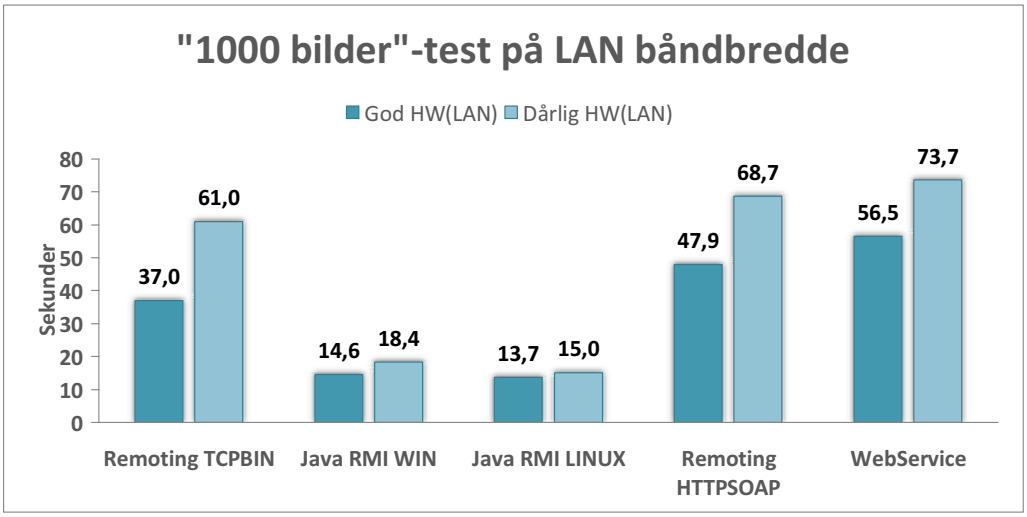

Forhandlertest på ADSL båndbredde
God HW(ADSL)
Dårlig HW(ADSL)
Sekunder
8,000
7,000
6,000
5,000
4,000
3,000
2,000
1,000
0,000
2,6
2,7
5,7
6,6
5,1
5,6
4,1
4,2
7,0
6,9
Remoting TCPBIN
Java RMI WIN
Java RMI LINUX
Remoting HTTPSOAP
WebService

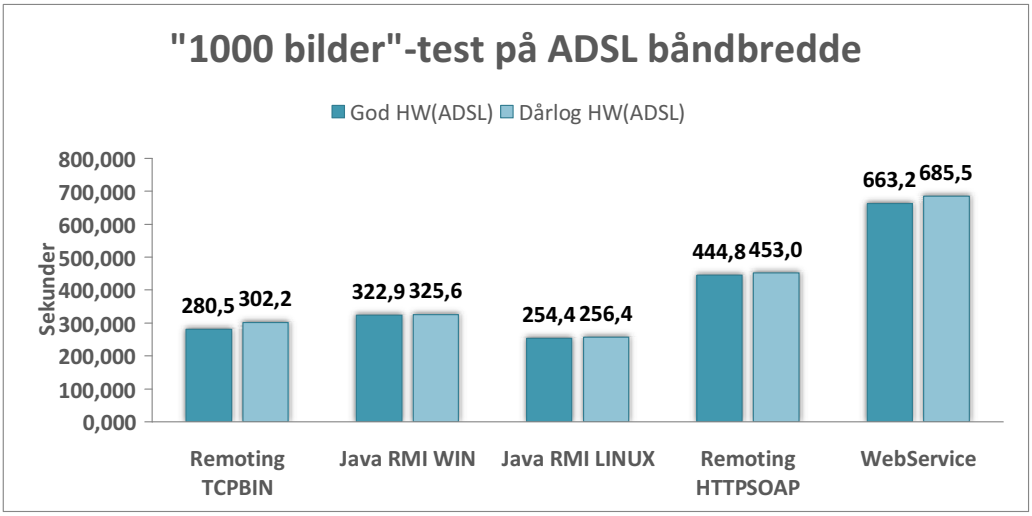

"50 modeller"-test på LAN båndbredde
God HW(LAN)
Dårlig HW(LAN)
Sekunder
0
1
2
3
4
5
6
1,3
2,0
3,4
5,6
3,3
5,0
2,5
2,9
2,0
2,6
Remoting TCPBIN
Java RMI WIN
Java RMI LINUX
Remoting HTTPSOAP
WebService


"50 modeller"-test på ADSL båndbredde
God HW(ADSL)
Dårlig HW(ADSL)
Sekunder
0,000
10,000
20,000
30,000
40,000
50,000
13,0
14,0
16,2
18,1
13,5
16,6
35,7
36,2
42,7
44,0
Remoting TCPBIN
Java RMI WIN
Java RMI LINUX
Remoting HTTPSOAP
WebService


"Ett bilde"-test på LAN båndbredde
God HW(LAN)
Dårlig HW(LAN)
Sekunder
0,100
0,080
0,060
0,040
0,020
0,000
0,0
0,1
0,0
0,0
0,0
0,0
0,1
0,1
0,0
0,1
Remoting TCPBIN
Java RMI WIN
Java RMI LINUX
Remoting HTTPSOAP
WebService


"Ett bilde"-test på ADSL båndbredde
God HW(ADSL)
Dårlig HW(ADSL)
Sekunder
0,900
0,800
0,700
0,600
0,500
0,400
0,300
0,200
0,100
0,000
0,3
0,4
0,6
0,6
0,4
0,5
0,5
0,5
0,8
0,8
Remoting TCPBIN
Java RMI WIN
Java RMI LINUX
Remoting HTTPSOAP
WebService


"1000 bilder"-test på LAN båndbredde
God HW(LAN)
Dårlig HW(LAN)
Sekunder
80
70
60
50
40
30
20
10
0
37,0
61,0
14,6
18,4
13,7
15,0
47,9
68,7
56,5
73,7
Remoting TCPBIN
Java RMI WIN
Java RMI LINUX
Remoting HTTPSOAP
WebService


"1000 bilder"-test på ADSL båndbredde
God HW(ADSL)
Dårlog HW(ADSL)
Sekunder
800,000
700,000
600,000
500,000
400,000
300,000
200,000
100,000
0,000
280,5
302,2
322,9
325,6
254,4
256,4
444,8
453,0
663,2
685,5
Remoting TCPBIN
Java RMI WIN
Java RMI LINUX
Remoting HTTPSOAP
WebService

Tillegg B

Testresultater

B.1 Tidsbruk hovedtest

B.1.1 LAN - Dårlig HW

.NET Remoting HTTP SOAP - RUN1


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingHttpSoapNUnit" total="4" failures="0" not-
    run="0" date="2006-04-24" time="01:36:27">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingHttpSoapNUnit" success="True"
      time="72.266" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingHttpSoapNUnit" success="True"
          time="72.266" asserts="0">
        - <results>
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.FemtiModeller"
              executed="True" success="True" time="2.984" asserts="0" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.ForhandlerTest"
              executed="True" success="True" time="0.594" asserts="1" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.HentBildeEtt" executed="True"
              success="True" time="0.094" asserts="0" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.HentBildeTusen"
              executed="True" success="True" time="68.516" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 13:36:10

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	39115	17681	Out	1	8088, 59196	v190c.studby.ntnu.no	54 408 236	13:35:52
129.241.152.190	224.0.0.252	0	3	Pass	0	137		276	13:35:22

This report was generated by CommView.

.NET Remoting HTTP SOAP - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingHttpSoapNUnit" total="4" failures="0" not-
    run="0" date="2006-04-24" time="01:38:17">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingHttpSoapNUnit" success="True"
      time="72.328" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingHttpSoapNUnit" success="True"
          time="72.328" asserts="0">
        - <results>
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.FemtiModeller"
              executed="True" success="True" time="2.859" asserts="0" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.ForhandlerTest"
              executed="True" success="True" time="0.609" asserts="1" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.HentBildeEtt" executed="True"
              success="True" time="0.078" asserts="0" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.HentBildeTusen"
              executed="True" success="True" time="68.719" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 13:38:04

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	39114	17643	Out	1	8088	v190c.studby.ntnu.no	54 405 662	13:37:50

This report was generated by CommView.

.NET Remoting TCP Binary - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingHttpSoapNUnit" total="4" failures="0" not-
    run="0" date="2006-04-24" time="02:14:41">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingHttpSoapNUnit" success="True"
      time="50.641" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingHttpSoapNUnit" success="True"
          time="50.641" asserts="0">
        - <results>
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.FemtiModeller"
              executed="True" success="True" time="2.484" asserts="0" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.ForhandlerTest"
              executed="True" success="True" time="0.484" asserts="1" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.HentBildeEtt" executed="True"
              success="True" time="0.063" asserts="0" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.HentBildeTusen"
              executed="True" success="True" time="47.563" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 13:16:53

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	29574	14020	Out	1	8085	v190c.studby.ntnu.no	39 550 533	13:16:20

This report was generated by CommView.

.NET Remoting TCP Binary - RUN2

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingTcpBinaryNUnit" total="4" failures="0" not-
    run="0" date="2006-04-24" time="01:18:21">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingTcpBinaryNUnit" success="True"
      time="63.750" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingTcpBinaryNUnit" success="True"
          time="63.734" asserts="0">
        - <results>
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.FemtiModeller"
              executed="True" success="True" time="2.031" asserts="0" />
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.ForhandlerTest"
              executed="True" success="True" time="0.484" asserts="1" />
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.HentBildeEtt" executed="True"
              success="True" time="0.094" asserts="0" />
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.HentBildeTusen"
              executed="True" success="True" time="61.016" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 13:18:31

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	29577	13984	Out	1	8085, 59196	v190c.studby.ntnu.no	39 549 231	13:18:21
129.241.152.190	129.241.152.255	0	1	Pass	0	138		233	13:17:12

This report was generated by CommView.

Web Service - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornesWebServiceNUnit" total="4" failures="0" not-run="0"
    date="2006-04-24" time="02:04:15">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornesWebServiceNUnit" success="True" time="80.609"
      asserts="0">
    - <results>
      - <test-suite name="TestEkornesWebService" success="True" time="80.609"
          asserts="0">
        - <results>
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.FemtiModeller"
              executed="True" success="True" time="3.984" asserts="0" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.ForhandlerTest"
              executed="True" success="True" time="0.578" asserts="1" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeEtt"
              executed="True" success="True" time="0.109" asserts="0" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeTusen"
              executed="True" success="True" time="74.922" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 14:04:01

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	38605	16377	Out	1	8080, 59196	v190c.studby.ntnu.no	53 440 080	14:03:48
129.241.150.133	129.241.150.101	1	1	Out	0	28681	v101a.studby.ntnu.no	197	14:03:19

This report was generated by CommView.

Web Service - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornesWebServiceNUnit" total="4" failures="0" not-run="0"
    date="2006-04-24" time="02:06:08">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornesWebServiceNUnit" success="True" time="78.031"
      asserts="0">
    - <results>
      - <test-suite name="TestEkornesWebService" success="True" time="78.031"
          asserts="0">
        - <results>
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.FemtiModeller"
              executed="True" success="True" time="2.594" asserts="0" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.ForhandlerTest"
              executed="True" success="True" time="0.594" asserts="1" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeEtt"
              executed="True" success="True" time="0.078" asserts="0" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeTusen"
              executed="True" success="True" time="73.719" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 14:05:54

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	38601	16383	Out	1	8080	v190c.studby.ntnu.no	53 439 738	14:05:41

This report was generated by CommView.

Java RMI Linux - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	24.860

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		4.782
testOpprettModellFemti	Success		5.046
testHentBildeEtt	Success		0.063
testHentBildeTusen	Success		14.969

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 22:07:06

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.252	37294	11633	Out	3	1099, 32771	v252a.studby.ntnu.no	39 539 785	22:06:33

This report was generated by CommView.

Java RMI Linux - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	22.922

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		2.891
testOpprettModellFemti	Success		4.953
testHentBildeEtt	Success		0.031
testHentBildeTusen	Success		15.047

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 22:10:48

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.252	37224	11596	Out	2	1099, 32771	v252a.studby.ntnu.no	39 532 897	22:10:36

This report was generated by CommView.

Java RMI Windows - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	30.516

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		3.266
testOpprettModellFemti	Success		6.578
testHentBildeEtt	Success		0.047
testHentBildeTusen	Success		20.625

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 12:07:30

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	37738	15218	Out	2	59196, 1099, 1045	v190c.studby.ntnu.no	39 759 880	12:07:06
129.241.152.190	129.241.152.255	0	13	Pass	0	138, 137		1 701	12:05:17

This report was generated by CommView.

Java RMI Windows - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	27.156

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		3.094
testOpprettModellFemti	Success		5.641
testHentBildeEtt	Success		0.047
testHentBildeTusen	Success		18.359

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 12:08:51

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	37785	15435	Out	2	1099, 1045, 59196	v190c.studby.ntnu.no	39 774 720	12:08:45

This report was generated by CommView.

B.1.2 LAN - God HW

.NET Remoting HTTP SOAP - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingHttpSoapNUnit" total="4" failures="0" not-
    run="0" date="2006-04-24" time="02:14:41">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingHttpSoapNUnit" success="True"
      time="50.641" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingHttpSoapNUnit" success="True"
          time="50.641" asserts="0">
        - <results>
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.FemtiModeller"
              executed="True" success="True" time="2.484" asserts="0" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.ForhandlerTest"
              executed="True" success="True" time="0.484" asserts="1" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.HentBildeEtt" executed="True"
              success="True" time="0.063" asserts="0" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.HentBildeTusen"
              executed="True" success="True" time="47.563" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 14:14:27

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	39115	16778	Out	1	8088	v101a.studby.ntnu.no	54 358 919	14:14:12

This report was generated by CommView.

.NET Remoting HTTP SOAP - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingHttpSoapNUnit" total="4" failures="0" not-
    run="0" date="2006-04-24" time="02:16:06">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingHttpSoapNUnit" success="True"
      time="51.016" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingHttpSoapNUnit" success="True"
          time="51.016" asserts="0">
        - <results>
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.FemtiModeller"
              executed="True" success="True" time="2.500" asserts="0" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.ForhandlerTest"
              executed="True" success="True" time="0.500" asserts="1" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.HentBildeEtt" executed="True"
              success="True" time="0.063" asserts="0" />
            <test-case name="EkornerRemotingHttpSoapNUnit.
              EkornesRemotingHttpSoapNUnit.HentBildeTusen"
              executed="True" success="True" time="47.891" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 14:15:52

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	39114	16722	Out	1	8088, 28681	v101a.studby.ntnu.no	54 355 927	14:15:38

This report was generated by CommView.

.NET Remoting TCP Binary - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingTcpBinaryNUnit" total="4" failures="0" not-
    run="0" date="2006-04-24" time="02:10:58">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingTcpBinaryNUnit" success="True"
      time="38.750" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingTcpBinaryNUnit" success="True"
          time="38.750" asserts="0">
        - <results>
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.FemtiModeller"
              executed="True" success="True" time="1.344" asserts="0" />
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.ForhandlerTest"
              executed="True" success="True" time="0.375" asserts="1" />
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.HentBildeEtt" executed="True"
              success="True" time="0.047" asserts="0" />
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.HentBildeTusen"
              executed="True" success="True" time="36.766" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 14:10:45

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	29573	13162	Out	1	8085	v101a.studby.ntnu.no	39 504 147	14:10:43
129.241.150.101	129.241.151.255	0	1	Pass	0	138		240	14:10:33

This report was generated by CommView.

.NET Remoting TCP Binary - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingTcpBinaryNUnit" total="4" failures="0" not-
    run="0" date="2006-04-24" time="02:12:08">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingTcpBinaryNUnit" success="True"
      time="38.781" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingTcpBinaryNUnit" success="True"
          time="38.781" asserts="0">
        - <results>
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.FemtiModeller"
              executed="True" success="True" time="1.344" asserts="0" />
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.ForhandlerTest"
              executed="True" success="True" time="0.359" asserts="1" />
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.HentBildeEtt" executed="True"
              success="True" time="0.047" asserts="0" />
            <test-case name="EkornerRemotingTcpBinaryNUnit.
              EkornesRemotingTcpBinaryNUnit.HentBildeTusen"
              executed="True" success="True" time="36.953" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 14:11:57

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	29573	13077	Out	1	8085	v101a.studby.ntnu.no	39 499 551	14:11:56

This report was generated by CommView.

Web Service - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornesWebServiceNUnit" total="4" failures="0" not-run="0"
    date="2006-04-24" time="01:57:58">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornesWebServiceNUnit" success="True" time="62.578"
      asserts="0">
    - <results>
      - <test-suite name="TestEkornesWebService" success="True" time="62.578"
          asserts="0">
        - <results>
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.FemtiModeller"
              executed="True" success="True" time="3.500" asserts="0" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.ForhandlerTest"
              executed="True" success="True" time="1.641" asserts="1" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeEtt"
              executed="True" success="True" time="0.063" asserts="0" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeTusen"
              executed="True" success="True" time="55.953" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 13:57:41

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	38603	17938	Out	1	80, 28681	v101a.studby.ntnu.no	53 518 234	13:57:19
129.241.150.133	129.241.152.190	3	3	Out	0	59196	v190c.studby.ntnu.no	804	13:57:25

This report was generated by CommView.

Web Service - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornesWebServiceNUnit" total="4" failures="0" not-run="0"
    date="2006-04-24" time="01:59:42">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornesWebServiceNUnit" success="True" time="60.266"
      asserts="0">
    - <results>
      - <test-suite name="TestEkornesWebService" success="True" time="60.266"
          asserts="0">
        - <results>
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.FemtiModeller"
              executed="True" success="True" time="1.984" asserts="0" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.ForhandlerTest"
              executed="True" success="True" time="0.438" asserts="1" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeEtt"
              executed="True" success="True" time="0.047" asserts="0" />
            <test-case
              name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeTusen"
              executed="True" success="True" time="56.531" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 13:59:26

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	38602	17868	Out	1	80	v101a.studby.ntnu.no	53 514 312	13:59:13

This report was generated by CommView.

Java RMI Linux - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	22.344

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		3.375
testOpprettModellFemti	Success		3.204
testHentBildeEtt	Success		0.046
testHentBildeTusen	Success		15.704

Properties »
System.out »

IP Statistics Report

Generated on 09.05.2006 at 11:12:48

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.169	129.241.151.255	0	21	Pass	0	32770, 137, 138		3 246	11:12:40
129.241.150.133	129.241.150.169	37369	11019	Out	2	1099, 32772	v169a.studby.ntnu.no	39 511 181	11:12:18

This report was generated by CommView.

Java RMI Linux - RUN2

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	19.062

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		2.047
testOpprettModellFemti	Success		3.328
testHentBildeEtt	Success		0.015
testHentBildeTusen	Success		13.672

Properties »
System.out »

IP Statistics Report

Generated on 09.05.2006 at 11:14:40

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.169	37345	12013	Out	2	1099, 32772	v169a.studby.ntnu.no	39 563 587	11:14:26

This report was generated by CommView.

Java RMI Windows - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	21.391

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		1.968
testOpprettModellFemti	Success		3.485
testHentBildeEtt	Success		0.047
testHentBildeTusen	Success		15.875

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 14:18:59

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	37774	13521	Out	2	1099, 3929, 28681	v101a.studby.ntnu.no	39 669 652	14:18:35

This report was generated by CommView.

Java RMI Windows - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	19.890

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		1.859
testOpprettModellFemti	Success		3.391
testHentBildeEtt	Success		0.031
testHentBildeTusen	Success		14.609

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 14:20:10

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	37769	13291	Out	2	1099, 3929	v101a.studby.ntnu.no	39 656 524	14:19:56

This report was generated by CommView.

B.1.3 ADSL - Dårlig HW

.NET Remoting HTTP SOAP - RUN1


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingHttpSoapNUnit" total="4" failures="0" not-run="0" date="2006-04-24"
    time="01:49:10">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft Windows NT 5.2.3790
      Service Pack 1" platform="Win32NT" cwd="D:\Inncoming\TESTRESULTATER!!
      \REMOTING_HTTPSOAP_152" machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingHttpSoapNUnit" success="True" time="496.624" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingHttpSoapNUnit" success="True" time="496.614" asserts="0">
        - <results>
            <test-case
              name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.FemtiModeller"
              executed="True" success="True" time="36.803" asserts="0" />
            <test-case
              name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.ForhandlerTest"
              executed="True" success="True" time="4.156" asserts="1" />
            <test-case
              name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.HentBildeEtt"
              executed="True" success="True" time="0.501" asserts="0" />
            <test-case
              name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.HentBildeTusen"
              executed="True" success="True" time="455.074" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 13:49:33

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	39115	21053	Out	1	8088	v190c.studby.ntnu.no	54 588 816	13:49:32

This report was generated by CommView.

.NET Remoting HTTP SOAP - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite  -->
- <test-results name="EkornerRemotingHttpSoapNUnit" total="4" failures="0" not-run="0" date="2006-04-24"
    time="02:01:41">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft Windows NT 5.2.3790
      Service Pack 1" platform="Win32NT" cwd="D:\Inncoming\TESTRESULTATER!!
      \REMOTING_HTTPSOAP_152" machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingHttpSoapNUnit" success="True" time="494.020" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingHttpSoapNUnit" success="True" time="494.010" asserts="0">
        - <results>
            <test-case
              name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.FemtiModeller"
              executed="True" success="True" time="36.222" asserts="0" />
            <test-case
              name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.ForhandlerTest"
              executed="True" success="True" time="4.166" asserts="1" />
            <test-case
              name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.HentBildeEtt"
              executed="True" success="True" time="0.541" asserts="0" />
            <test-case
              name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.HentBildeTusen"
              executed="True" success="True" time="452.991" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 14:01:20

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	39115	21046	Out	1	8088	v190c.studby.ntnu.no	54 588 426	13:59:52

This report was generated by CommView.

.NET Remoting TCP Binary - RUN1


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingTcpBinaryNUnit" total="4" failures="0" not-run="0" date="2006-04-24"
    time="01:25:03">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft Windows NT 5.2.3790
      Service Pack 1" platform="Win32NT" cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_TCPBIN_152"
      machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingTcpBinaryNUnit" success="True" time="322.414" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingTcpBinaryNUnit" success="True" time="322.394" asserts="0">
        - <results>
            <test-case
              name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.FemtiModeller"
              executed="True" success="True" time="13.980" asserts="0" />
            <test-case
              name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.ForhandlerTest"
              executed="True" success="True" time="2.714" asserts="1" />
            <test-case
              name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.HentBildeEtt"
              executed="True" success="True" time="0.381" asserts="0" />
            <test-case
              name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.HentBildeTusen"
              executed="True" success="True" time="304.959" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 13:24:47

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	29574	16517	Out	1	8085	v190c.studby.ntnu.no	39 685 383	13:24:22

This report was generated by CommView.

.NET Remoting TCP Binary - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornerRemotingTcpBinaryNUnit" total="4" failures="0" not-run="0" date="2006-04-24"
    time="01:32:21">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft Windows NT 5.2.3790
      Service Pack 1" platform="Win32NT" cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_TCPBIN_152"
      machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornerRemotingTcpBinaryNUnit" success="True" time="319.469" asserts="0">
    - <results>
      - <test-suite name="EkornesRemotingTcpBinaryNUnit" success="True" time="319.469" asserts="0">
        - <results>
            <test-case
              name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.FemtiModeller"
              executed="True" success="True" time="14.040" asserts="0" />
            <test-case
              name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.ForhandlerTest"
              executed="True" success="True" time="2.744" asserts="1" />
            <test-case
              name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.HentBildeEtt"
              executed="True" success="True" time="0.351" asserts="0" />
            <test-case
              name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.HentBildeTusen"
              executed="True" success="True" time="302.154" asserts="0" />
          </results>
        </test-suite>
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 13:31:52

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	29574	16496	Out	1	8085	v190c.studby.ntnu.no	39 684 237	13:30:56

This report was generated by CommView.

Web Service - RUN1


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="TestEkornesWebService" total="4" failures="0" not-run="0"
    date="2006-04-24" time="02:19:24">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft
      Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\WEBSERVICE_152" machine-
      name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="TestEkornesWebService" success="True" time="727.166"
      asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.FemtiModeller"
          executed="True" success="True" time="31.956" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.ForhandlerTest"
          executed="True" success="True" time="7.010" asserts="1" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeEtt"
          executed="True" success="True" time="0.771" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeTusen"
          executed="True" success="True" time="684.955" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 14:19:08

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	38598	20665	Out	1	8080	v190c.studby.ntnu.no	53 670 372	14:18:40

This report was generated by CommView.

Web Service - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="TestEkornesWebService" total="4" failures="0" not-run="0"
    date="2006-04-24" time="02:32:41">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft
      Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\WEBSERVICE_152" machine-
      name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="TestEkornesWebService" success="True" time="739.784"
      asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.FemtiModeller"
          executed="True" success="True" time="43.953" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.ForhandlerTest"
          executed="True" success="True" time="6.890" asserts="1" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeEtt"
          executed="True" success="True" time="0.771" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeTusen"
          executed="True" success="True" time="685.536" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 14:32:57

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	38598	20664	Out	1	8080	v190c.studby.ntnu.no	53 670 342	14:32:45

This report was generated by CommView.

Java RMI Linux - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	277.199

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		5.899
testOpprettModellFemti	Success		15.352
testHentBildeEtt	Success		0.481
testHentBildeTusen	Success		255.447

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 09:54:16

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	213.236.166.135	600	731	In	0	3389	pc135.teknonett.com	214 706	09:54:16
192.168.1.37	129.241.150.252	36781	19288	Out	2	1099, 32772	v252a.studby.ntnu.no	39 924 200	09:53:34
192.168.1.1	224.0.0.1	0	6	Pass	0		ALL-SYSTEMS.MCAST.NET	360	09:53:25
192.168.1.1	224.0.0.9	0	3	Pass	0		RIP2-ROUTERS.MCAST.NET	180	09:53:06
192.168.1.37	192.168.1.255	0	1	Out	0	138		249	09:50:00

This report was generated by CommView.

Java RMI Linux - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	279.141

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		5.638
testOpprettModellFemti	Success		16.614
testHentBildeEtt	Success		0.481
testHentBildeTusen	Success		256.398

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 10:12:44

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	213.236.166.135	890	1159	In	0	3389	pc135.teknonett.com	353 156	10:12:44
192.168.1.37	129.241.150.252	36743	19345	Out	2	1099, 32772	v252a.studby.ntnu.no	39 925 208	10:12:26
192.168.1.37	217.118.32.12	2	2	Out	0	53	thufir.bluecom.no	533	10:11:20
192.168.1.1	224.0.0.1	0	6	Pass	0		ALL-SYSTEMS.MCAST.NET	360	10:12:22
192.168.1.1	224.0.0.9	0	3	Pass	0		RIP2-ROUTERS.MCAST.NET	180	10:12:03
192.168.1.37	192.168.1.255	0	1	Out	0	138		243	10:11:19

This report was generated by CommView.

Java RMI Windows - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	351.025

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		6.890
testOpprettModellFemti	Success		18.737
testHentBildeEtt	Success		0.501
testHentBildeTusen	Success		324.887

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 11:57:31

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.105.181	1930	2310	Out	0	1119	dhcp-105-181.idi.ntnu.no	585 193	11:57:31
192.168.1.37	217.118.32.12	4	4	Out	0	53	thufir.bluecom.no	1 115	11:51:03
192.168.1.37	192.168.1.255	0	1	Out	0	138		249	11:50:00
192.168.1.37	129.241.150.133	12	12	Out	0	1099	v133a.studby.ntnu.no	1 464	11:50:12
192.168.1.1	224.0.0.1	0	7	Pass	0		ALL-SYSTEMS.MCAST.NET	420	11:57:18
192.168.1.1	224.0.0.9	0	4	Pass	0		RIP2-ROUTERS.MCAST.NET	240	11:57:18
192.168.1.37	129.241.152.190	37804	19759	Out	3	1099, 1045	v190c.studby.ntnu.no	40 008 399	11:56:56

This report was generated by CommView.

Java RMI Windows - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	350.874

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		6.609
testOpprettModellFemti	Success		18.086
testHentBildeEtt	Success		0.551
testHentBildeTusen	Success		325.598

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 12:05:05

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.105.181	1192	1568	In	0	3389	dhcp-105-181.idi.ntnu.no	305 645	12:05:05
192.168.1.37	129.241.152.190	37807	19759	Out	3	1099, 1045	v190c.studby.ntnu.no	40 008 581	12:04:24
192.168.1.1	224.0.0.1	0	6	Pass	0		ALL-SYSTEMS.MCAST.NET	360	12:03:55
192.168.1.1	224.0.0.9	0	3	Pass	0		RIP2-ROUTERS.MCAST.NET	180	12:03:37
192.168.1.37	192.168.1.255	0	2	Out	0	138		492	12:05:00

This report was generated by CommView.

B.1.4 ADSL - God HW

.NET Remoting HTTP SOAP - RUN1


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornesRemotingHttpSoapNUnit" total="4" failures="0" not-run="0" date="2006-04-
  24" time="03:41:56">
  <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft Windows NT
    5.2.3790 Service Pack 1" platform="Win32NT" cwd="D:\Inncoming\TESTRESULTATER!!
    \REMOTING_HTTPSOAP_150" machine-name="SERVER" user="Administrator" user-
    domain="SERVER" />
  <culture-info current-culture="nb-NO" current-uiculture="en-US" />
- <test-suite name="EkornesRemotingHttpSoapNUnit" success="True" time="461.914" asserts="0">
  - <results>
      <test-case
        name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.FemtiModeller"
        executed="True" success="True" time="24.635" asserts="0" />
      <test-case
        name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.ForhandlerTest"
        executed="True" success="True" time="4.076" asserts="1" />
      <test-case
        name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.HentBildeEtt"
        executed="True" success="True" time="0.481" asserts="0" />
      <test-case
        name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.HentBildeTusen"
        executed="True" success="True" time="432.632" asserts="0" />
    </results>
  </test-suite>
</test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 15:42:14

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	39115	21031	Out	1	8088	v101a.studby.ntnu.no	54 587 434	15:40:38

This report was generated by CommView.

.NET Remoting HTTP SOAP - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornesRemotingHttpSoapNUnit" total="4" failures="0" not-run="0" date="2006-04-
  24" time="03:52:34">
  <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft Windows NT
    5.2.3790 Service Pack 1" platform="Win32NT" cwd="D:\Inncoming\TESTRESULTATER!!
    \REMOTING_HTTPSOAP_150" machine-name="SERVER" user="Administrator" user-
    domain="SERVER" />
  <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornesRemotingHttpSoapNUnit" success="True" time="485.228" asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.FemtiModeller"
          executed="True" success="True" time="35.721" asserts="0" />
        <test-case
          name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.ForhandlerTest"
          executed="True" success="True" time="4.086" asserts="1" />
        <test-case
          name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.HentBildeEtt"
          executed="True" success="True" time="0.481" asserts="0" />
        <test-case
          name="EkornerRemotingHttpSoapNUnit.EkornesRemotingHttpSoapNUnit.HentBildeTusen"
          executed="True" success="True" time="444.840" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 15:52:14

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	39114	21128	Out	1	8088	v101a.studby.ntnu.no	54 592 702	15:52:07

This report was generated by CommView.

.NET Remoting TCP Binary - RUN1


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornesRemotingTcpBinaryNUnit" total="4" failures="0" not-run="0" date="2006-04-24" time="04:13:00">
  <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT" cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_TCPBIN_150" machine-name="SERVER" user="Administrator" user-domain="SERVER" />
  <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornesRemotingTcpBinaryNUnit" success="True" time="301.043" asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.FemtiModeller"
          executed="True" success="True" time="13.159" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.ForhandlerTest"
          executed="True" success="True" time="2.614" asserts="1" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.HentBildeEtt"
          executed="True" success="True" time="0.320" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.HentBildeTusen"
          executed="True" success="True" time="284.779" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 16:14:14

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	29572	16479	Out	1	8085	v101a.studby.ntnu.no	39 683 235	15:58:11

This report was generated by CommView.

.NET Remoting TCP Binary - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="EkornesRemotingTcpBinaryNUnit" total="4" failures="0" not-run="0" date="2006-04-24" time="04:24:30">
  <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT" cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_TCPBIN_150" machine-name="SERVER" user="Administrator" user-domain="SERVER" />
  <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="EkornesRemotingTcpBinaryNUnit" success="True" time="296.586" asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.FemtiModeller"
          executed="True" success="True" time="12.979" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.ForhandlerTest"
          executed="True" success="True" time="2.624" asserts="1" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.HentBildeEtt"
          executed="True" success="True" time="0.310" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.EkornesRemotingTcpBinaryNUnit.HentBildeTusen"
          executed="True" success="True" time="280.493" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 16:24:43

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	29575	16492	Out	1	8085	v101a.studby.ntnu.no	39 685 559	16:23:40

This report was generated by CommView.

Web Service - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="TestEkornesWebService" total="4" failures="0" not-run="0"
    date="2006-04-24" time="03:15:49">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft
      Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\WEBSERVICE_150" machine-
      name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="TestEkornesWebService" success="True" time="706.075"
      asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.FemtiModeller"
          executed="True" success="True" time="31.215" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.ForhandlerTest"
          executed="True" success="True" time="6.960" asserts="1" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeEtt"
          executed="True" success="True" time="0.761" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeTusen"
          executed="True" success="True" time="664.716" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Table

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	38601	20666	Out	1	80	v101a.studby.ntnu.no	53 666 818	14:53:36

Web Service - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="TestEkornesWebService" total="4" failures="0" not-run="0"
    date="2006-04-24" time="03:28:18">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-version="Microsoft
      Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\WEBSERVICE_150" machine-
      name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="TestEkornesWebService" success="True" time="716.160"
      asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.FemtiModeller"
          executed="True" success="True" time="42.661" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.ForhandlerTest"
          executed="True" success="True" time="6.990" asserts="1" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeEtt"
          executed="True" success="True" time="0.751" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.TestEkornesWebService.HentBildeTusen"
          executed="True" success="True" time="663.174" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 15:28:42

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	38598	20664	Out	1	80	v101a.studby.ntnu.no	53 665 055	15:27:54

This report was generated by CommView.

Java RMI Linux - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	274.114

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		5.027
testOpprettModellFemti	Success		13.840
testHentBildeEtt	Success		0.471
testHentBildeTusen	Success		254.756

Properties »
System.out »

IP Statistics Report

Generated on 03.05.2006 at 10:24:32

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.169	36766	19295	Out	2	1099, 32778	v169a.studby.ntnu.no	39 925 355	10:14:07

This report was generated by CommView.

Java RMI Linux - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	273.383

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		5.087
testOpprettModellFemti	Success		13.479
testHentBildeEtt	Success		0.411
testHentBildeTusen	Success		254.386

Properties »
System.out »

IP Statistics Report

Generated on 03.05.2006 at 10:42:02

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.169	36772	19285	Out	2	1099, 32778	v169a.studby.ntnu.no	39 925 129	10:29:33

This report was generated by CommView.

Java RMI Windows - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	345.697

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		5.739
testOpprettModellFemti	Success		16.383
testHentBildeEtt	Success		0.551
testHentBildeTusen	Success		323.004

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 16:38:17

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	37799	19763	Out	3	1099, 4486	v101a.studby.ntnu.no	40 008 311	16:36:49

This report was generated by CommView.

Java RMI Windows - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class GUITest

Name	Tests	Errors	Failures	Time(s)
GUITest	4	0	0	345.387

Tests

Name	Status	Type	Time(s)
testOpprettForhandler	Success		5.669
testOpprettModellFemti	Success		16.193
testHentBildeEtt	Success		0.551
testHentBildeTusen	Success		322.944

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 16:45:09

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	37799	19761	Out	3	1099, 4486	v101a.studby.ntnu.no	40 008 203	16:44:30

This report was generated by CommView.

B.2 Filstørrelsetest

B.2.1 LAN - Dårlig HW

.NET Remoting HTTP SOAP - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingHttpSoap" total="2" failures="0" not-run="0"
    date="2006-04-24" time="09:52:18">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingHttpSoap" success="True" time="42.016"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="22.594" asserts="0" />
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="18.984" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 21:52:01

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	41291	15747	Out	1	8088	v190c.studby.ntnu.no	59 389 281	21:51:41
129.241.150.133	129.241.150.101	1	1	In	0	51711	v101a.studby.ntnu.no	200	21:51:56

This report was generated by CommView.

.NET Remoting HTTP SOAP - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingHttpSoap" total="2" failures="0" not-run="0"
    date="2006-04-24" time="09:53:59">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingHttpSoap" success="True" time="37.609"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="21.359" asserts="0" />
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="15.984" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 21:53:43

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	41629	16252	Out	1	8088, 59196	v190c.studby.ntnu.no	59 889 963	21:53:34
129.241.150.133	129.241.150.101	1	1	Out	0	28681	v101a.studby.ntnu.no	197	21:53:28

This report was generated by CommView.

.NET Remoting TCP Binary - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingTcpBinary" total="2" failures="0" not-run="0"
    date="2006-04-24" time="09:49:23">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingTcpBinary" success="True" time="28.844"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="16.844" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="11.891" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 21:49:08

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	31218	11976	Out	1	8085	v190c.studby.ntnu.no	44 883 958	21:48:57
129.241.150.133	129.241.150.101	1	1	In	0	51711	v101a.studby.ntnu.no	200	21:48:56

This report was generated by CommView.

.NET Remoting TCP Binary - RUN2

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingTcpBinary" total="2" failures="0" not-run="0"
    date="2006-04-24" time="09:50:40">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingTcpBinary" success="True" time="29.781"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="17.359" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="12.328" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 21:50:24

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	31220	12219	Out	1	8085	v190c.studby.ntnu.no	44 898 448	21:50:14
129.241.150.101	129.241.151.255	0	1	Pass	0	138		240	21:49:50

This report was generated by CommView.

Web Service - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSize" total="2" failures="0" not-run="0" date="2006-04-24"
    time="09:56:52">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSize" success="True" time="46.156" asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeEnMegaTjueganger"
          executed="True" success="True" time="24.703" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeTjueMegaEnGang"
          executed="True" success="True" time="20.078" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 21:56:39

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	42932	16763	Out	1	8080, 59196	v190c.studby.ntnu.no	59 980 490	21:56:35
129.241.150.133	129.241.150.101	1	1	Out	0	28681	v101a.studby.ntnu.no	197	21:56:29

This report was generated by CommView.

Web Service - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSize" total="2" failures="0" not-run="0" date="2006-04-24"
    time="09:58:26">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSize" success="True" time="44.797" asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeEnMegaTjueganger"
          executed="True" success="True" time="25.484" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeTjueMegaEnGang"
          executed="True" success="True" time="18.094" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 21:58:11

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	42940	16953	Out	1	8080	v190c.studby.ntnu.no	59 989 346	21:57:58
129.241.150.133	129.241.150.101	2	2	In	0	51711	v101a.studby.ntnu.no	669	21:58:07

This report was generated by CommView.

Java RMI Linux - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	11.281

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		5.656
testTjueBilderEnMega	Success		5.609

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 22:17:25

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.252	29185	9603	Out	2	1099, 32771	v252a.studby.ntnu.no	44 051 668	22:17:09

This report was generated by CommView.

Java RMI Linux - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	12.203

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		5.688
testTjueBilderEnMega	Success		6.500

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 22:19:25

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.252	29218	9645	Out	2	1099, 32771	v252a.studby.ntnu.no	44 055 888	22:19:13

This report was generated by CommView.

Java RMI Windows - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	41.297

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		20.468
testTjueBilderEnMega	Success		20.813

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 21:19:14

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	41017	20822	Out	2	1099, 1784	v190c.studby.ntnu.no	45 296 369	21:19:01
129.241.150.133	129.241.150.101	1	1	In	0	51711	v101a.studby.ntnu.no	200	21:18:52
129.241.152.190	129.241.152.255	0	1	Pass	0	138		233	21:18:53

This report was generated by CommView.

Java RMI Windows - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	40.922

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		20.782
testTjueBilderEnMega	Success		20.140

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 21:22:39

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.152.190	40995	20984	Out	2	1099, 1784	v190c.studby.ntnu.no	45 303 957	21:22:21
129.241.150.133	129.241.150.101	2	2	In	0	51711	v101a.studby.ntnu.no	669	21:21:52

This report was generated by CommView.

B.2.2 LAN - God HW

.NET Remoting HTTP SOAP - RUN1


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingHttpSoap" total="2" failures="0" not-run="0"
    date="2006-05-09" time="12:03:02">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingHttpSoap" success="True" time="35.797"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="19.125" asserts="0" />
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="16.078" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 09.05.2006 at 12:03:16

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	41352	14501	Out	1	8088	v101a.studby.ntnu.no	59 408 387	12:02:30

This report was generated by CommView.

.NET Remoting HTTP SOAP - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingHttpSoap" total="2" failures="0" not-run="0"
    date="2006-04-24" time="05:13:00">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingHttpSoap" success="True" time="31.156"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="15.875" asserts="0" />
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="14.500" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 17:12:43

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	40984	14426	Out	1	8088	v101a.studby.ntnu.no	58 876 265	17:12:20

This report was generated by CommView.

.NET Remoting TCP Binary - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingTcpBinary" total="2" failures="0" not-run="0"
    date="2006-04-24" time="05:03:32">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingTcpBinary" success="True" time="19.453"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="10.422" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="8.938" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 17:03:12

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	30739	10515	Out	1	8085	v101a.studby.ntnu.no	44 119 378	17:02:58

This report was generated by CommView.

.NET Remoting TCP Binary - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingTcpBinary" total="2" failures="0" not-run="0"
    date="2006-04-24" time="05:05:27">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingTcpBinary" success="True" time="19.375"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="10.484" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="8.797" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 17:05:02

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	30646	10687	Out	1	8085	v101a.studby.ntnu.no	43 996 668	17:05:01

This report was generated by CommView.

Web Service - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSize" total="2" failures="0" not-run="0" date="2006-04-24"
    time="05:48:59">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSize" success="True" time="32.359" asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeEnMegaTjueganger"
          executed="True" success="True" time="18.156" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeTjueMegaEnGang"
          executed="True" success="True" time="13.109" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 17:48:41

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	42896	17160	In	1	51711, 4028	v101a.studby.ntnu.no	59 997 971	17:48:22

This report was generated by CommView.

Web Service - RUN2

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSize" total="2" failures="0" not-run="0" date="2006-04-24"
    time="05:51:04">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.1.2600 Service Pack 2" platform="Win32NT"
      cwd="E:\SKOLE\NTNU_06\Masteroppgaven\TestResultater" machine-
      name="CAMJAC" user="camilla" user-domain="CAMJAC" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSize" success="True" time="33.547" asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeEnMegaTjueganger"
          executed="True" success="True" time="17.703" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeTjueMegaEnGang"
          executed="True" success="True" time="14.547" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 17:50:51

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	42915	17147	Out	1	80, 28681	v101a.studby.ntnu.no	59 998 447	17:50:39

This report was generated by CommView.

Java RMI Linux - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	13.203

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		6.781
testTjueBilderEnMega	Success		6.422

Properties »
System.out »

IP Statistics Report

Generated on 09.05.2006 at 11:17:06

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.169	29183	10044	Out	2	1099, 32772	v169a.studby.ntnu.no	44 075 416	11:16:34
129.241.150.169	129.241.151.255	0	2	Pass	0	138		514	11:16:41

This report was generated by CommView.

Java RMI Linux - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	9.828

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		4.937
testTjueBilderEnMega	Success		4.875

Properties »
System.out »

IP Statistics Report

Generated on 09.05.2006 at 11:18:08

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.169	29165	10051	Out	2	1099, 32772	v169a.studby.ntnu.no	44 074 740	11:17:53

This report was generated by CommView.

Java RMI Windows - RUN1

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	13.172

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		7.250
testTjueBilderEnMega	Success		5.890

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 16:55:52

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	40046	12715	Out	2	28681, 1099, 4486	v101a.studby.ntnu.no	44 738 175	16:55:40
129.241.150.133	129.241.152.190	3	3	Out	0	59196	v190c.studby.ntnu.no	804	16:55:11

This report was generated by CommView.

Java RMI Windows - RUN2

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	12.375

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		6.000
testTjueBilderEnMega	Success		6.375

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 16:57:07

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
129.241.150.133	129.241.150.101	40352	12747	Out	2	1099, 4486	v101a.studby.ntnu.no	44 756 304	16:56:47

This report was generated by CommView.

B.2.3 ADSL - Dårlig HW

.NET Remoting HTTP SOAP - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingHttpSoap" total="2" failures="0" not-run="0"
    date="2006-04-24" time="05:41:34">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_HTTPSOAP_FS_152"
      machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingHttpSoap" success="True" time="386.095"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingHttpSoap.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="186.588" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingHttpSoap.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="198.005" asserts="0" />
      </results>
    </test-suite>
</test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 17:41:16

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	40986	20712	Out	1	8088	v190c.studby.ntnu.no	59 215 820	17:40:52

This report was generated by CommView.

.NET Remoting HTTP SOAP - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingHttpSoap" total="2" failures="0" not-run="0"
    date="2006-04-24" time="05:48:33">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_HTTPSOAP_FS_152"
      machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingHttpSoap" success="True" time="365.245"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingHttpSoap.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="191.045" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingHttpSoap.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="173.790" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 17:48:11

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	40984	20688	Out	1	8088	v190c.studby.ntnu.no	59 214 392	17:47:44

This report was generated by CommView.

.NET Remoting TCP Binary - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingTcpBinary" total="2" failures="0" not-run="0"
    date="2006-04-24" time="05:16:50">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_TCPBIN_FS_152"
      machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingTcpBinary" success="True" time="314.903"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="141.554" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="173.009" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 17:17:09

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	30739	15517	Out	1	8085	v190c.studby.ntnu.no	44 389 498	17:15:41

This report was generated by CommView.

.NET Remoting TCP Binary - RUN2

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingTcpBinary" total="2" failures="0" not-run="0"
    date="2006-04-24" time="05:29:16">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_TCPBIN_FS_152"
      machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingTcpBinary" success="True" time="269.197"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="128.004" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="141.013" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 17:28:58

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	30739	15532	Out	1	8085	v190c.studby.ntnu.no	44 390 308	17:22:04

This report was generated by CommView.

Web Service - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSize" total="2" failures="0" not-run="0" date="2006-04-24"
    time="06:04:32">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\WEBSERVICE_FS_152" machine-
      name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSize" success="True" time="680.809" asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeEnMegaTjueganger"
          executed="True" success="True" time="348.782" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeTjueMegaEnGang"
          executed="True" success="True" time="329.103" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 18:04:50

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	42942	21497	Out	1	8080	v190c.studby.ntnu.no	60 234 666	18:03:54

This report was generated by CommView.

Web Service - RUN2

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSize" total="2" failures="0" not-run="0" date="2006-04-24"
    time="06:18:39">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\WEBSERVICE_FS_152" machine-
      name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSize" success="True" time="692.886" asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeEnMegaTjueganger"
          executed="True" success="True" time="361.700" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeTjueMegaEnGang"
          executed="True" success="True" time="328.683" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 18:18:13

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	42942	21490	Out	1	8080	v190c.studby.ntnu.no	60 234 300	18:17:33

This report was generated by CommView.

Java RMI Linux - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	230.622

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		114.766
testTjueBilderEnMega	Success		115.846

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 09:53:54

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
81.191.26.84	129.241.152.190	36760	19269	Out	1	1099	v190c.studby.ntnu.no	40 055 603	09:53:34

This report was generated by CommView.

Java RMI Linux - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	230.111

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		114.294
testTjueBilderEnMega	Success		115.797

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 10:12:46

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
81.191.26.84	129.241.152.190	36724	19326	Out	1	1099	v190c.studby.ntnu.no	40 160 460	10:12:26

This report was generated by CommView.

Java RMI Windows - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	335.262

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		167.250
testTjueBilderEnMega	Success		168.002

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 16:59:53

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	40969	20544	Out	4	1099, 1430	v190c.studby.ntnu.no	45 211 450	16:59:34

This report was generated by CommView.

Java RMI Windows - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	350.194

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		174.201
testTjueBilderEnMega	Success		175.973

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 17:08:57

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.152.190	40969	20536	Out	4	1099, 1430	v190c.studby.ntnu.no	45 211 018	17:08:07

This report was generated by CommView.

B.2.4 ADSL - God HW

.NET Remoting HTTP SOAP - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingHttpSoap" total="2" failures="0" not-run="0"
    date="2006-04-24" time="07:52:55">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_HTTPSOAP_FS_150"
      machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingHttpSoap" success="True" time="353.238"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="186.779" asserts="0" />
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="164.827" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 19:52:35

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	41624	21022	Out	1	8088	v101a.studby.ntnu.no	60 146 748	19:52:02

This report was generated by CommView.

.NET Remoting HTTP SOAP - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingHttpSoap" total="2" failures="0" not-run="0"
    date="2006-04-24" time="07:59:29">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_HTTPSOAP_FS_150"
      machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingHttpSoap" success="True" time="352.377"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="188.721" asserts="0" />
        <test-case
          name="EkornerRemotingHttpSoapNUnit.FileSizeRemotingHttpSoap.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="163.275" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 19:59:43

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	41625	21071	Out	1	8088	v101a.studby.ntnu.no	60 149 466	19:59:29

This report was generated by CommView.

.NET Remoting TCP Binary - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingTcpBinary" total="2" failures="0" not-run="0"
    date="2006-04-24" time="08:07:23">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_TCPBIN_FS_150"
      machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingTcpBinary" success="True" time="245.002"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="126.832" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="117.799" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 20:07:05

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	31219	15808	Out	1	8085	v101a.studby.ntnu.no	45 090 952	20:05:58

This report was generated by CommView.

.NET Remoting TCP Binary - RUN2


```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSizeRemotingTcpBinary" total="2" failures="0" not-run="0"
    date="2006-04-24" time="08:12:25">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\REMOTING_TCPBIN_FS_150"
      machine-name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSizeRemotingTcpBinary" success="True" time="243.991"
      asserts="0">
    - <results>
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeEnMegaTjueganger" executed="True" success="True"
          time="125.661" asserts="0" />
        <test-case
          name="EkornerRemotingTcpBinaryNUnit.FileSizeRemotingTcpBinary.
          HentBildeTjueMegaEnGang" executed="True" success="True"
          time="118.130" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```


IP Statistics Report

Generated on 24.04.2006 at 20:12:04

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	31219	15986	Out	1	8085	v101a.studby.ntnu.no	45 100 552	20:11:51

This report was generated by CommView.

Web Service - RUN1

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite -->
- <test-results name="FileSize" total="2" failures="0" not-run="0" date="2006-04-24"
    time="07:29:53">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\WEBSERVICE_FS_150" machine-
      name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSize" success="True" time="682.071" asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeEnMegaTjueganger"
          executed="True" success="True" time="354.770" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeTjueMegaEnGang"
          executed="True" success="True" time="324.817" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 19:30:10

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	42942	21496	Out	1	80	v101a.studby.ntnu.no	60 234 519	19:29:17

This report was generated by CommView.

Web Service - RUN2

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<!-- This file represents the results of running a test suite  -->
- <test-results name="FileSize" total="2" failures="0" not-run="0" date="2006-04-24"
    time="07:43:28">
    <environment nunit-version="2.2.7.0" clr-version="2.0.50727.42" os-
      version="Microsoft Windows NT 5.2.3790 Service Pack 1" platform="Win32NT"
      cwd="D:\Inncoming\TESTRESULTATER!!\WEBSERVICE_FS_150" machine-
      name="SERVER" user="Administrator" user-domain="SERVER" />
    <culture-info current-culture="nb-NO" current-uiculture="en-US" />
  - <test-suite name="FileSize" success="True" time="680.749" asserts="0">
    - <results>
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeEnMegaTjueganger"
          executed="True" success="True" time="353.628" asserts="0" />
        <test-case
          name="EkornesWebServiceNUnit.FileSize.HentBildeTjueMegaEnGang"
          executed="True" success="True" time="324.406" asserts="0" />
      </results>
    </test-suite>
  </test-results>
```

IP Statistics Report

Generated on 24.04.2006 at 19:43:02

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	42944	21494	Out	1	80	v101a.studby.ntnu.no	60 234 543	19:41:36

This report was generated by CommView.

Java RMI Linux - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	231.233

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		115.406
testTjueBilderEnMega	Success		115.807

Properties »
System.out »

IP Statistics Report

Generated on 03.05.2006 at 22:42:13

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.169	28837	14950	Out	3	1099, 32772	v169a.studby.ntnu.no	42 069 775	22:39:35

This report was generated by CommView.

Java RMI Linux - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	229.750

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		114.094
testTjueBilderEnMega	Success		115.636

Properties »
System.out »

IP Statistics Report

Generated on 03.05.2006 at 22:47:50

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.169	29855	15446	Out	3	1099, 32772	v169a.studby.ntnu.no	43 510 826	22:46:22

This report was generated by CommView.

Java RMI Windows - RUN1

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	330.325

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		165.107
testTjueBilderEnMega	Success		165.208

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 20:20:13

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	41027	20569	Out	4	1099, 4897	v101a.studby.ntnu.no	45 284 032	20:19:43

This report was generated by CommView.

Java RMI Windows - RUN2

Unit Test Results

Designed for use with JUnit and Ant.

Class TestFileSize

Name	Tests	Errors	Failures	Time(s)
TestFileSize	2	0	0	329.504

Tests

Name	Status	Type	Time(s)
testEttBildeTjueMega	Success		164.176
testTjueBilderEnMega	Success		165.298

Properties »
System.out »

IP Statistics Report

Generated on 24.04.2006 at 20:26:52

Local IP	Remote IP	In	Out	Direction	Sessions	Ports	Hostname	Bytes	Last packet
192.168.1.37	129.241.150.101	41027	20565	Out	4	1099, 4897	v101a.studby.ntnu.no	45 283 822	20:25:58

This report was generated by CommView.

Tillegg C

WSDL

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://tempuri.org/"
xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
targetNamespace="http://tempuri.org/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://tempuri.org/">
      <s:element name="opprettForhandler">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="forhandler"
type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="opprettForhandlerResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="opprettForhandlerResult"
              type="tns:Forhandler" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="Forhandler">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="id"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Navn"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Land"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Adresse"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Tlf"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Kontakt"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Faks"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Email"
type="s:string" />
          <s:element minOccurs="1" maxOccurs="1"
name="opprettelsesDato" type="s:dateTime" />
          <s:element minOccurs="0" maxOccurs="1" name="alModell"
type="tns:ArrayOfModellObjekt" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfModellObjekt">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded"
name="ModellObjekt" nillable="true"
          type="tns:ModellObjekt" />
```

```
        </s:sequence>
      </s:complexType>
      <s:complexType name="ModellObjekt">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="Forhandler"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="AktuellModell"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Storrelse"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="alMatchModell"
type="tns:ArrayOfString" />
          <s:element minOccurs="0" maxOccurs="1" name="alMatchStr"
type="tns:ArrayOfString1" />
          <s:element minOccurs="0" maxOccurs="1" name="alTilbehor"
type="tns:ArrayOfString2" />
          <s:element minOccurs="0" maxOccurs="1" name="alMobel"
type="tns:ArrayOfMobelObjekt" />
          <s:element minOccurs="0" maxOccurs="1" name="bArray"
type="s:base64Binary" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfString">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded"
name="MatchModell" nillable="true"
          type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfString1">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded"
name="MatchStr" nillable="true"
          type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfString2">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded"
name="Tilbehor" nillable="true"
          type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfMobelObjekt">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="Mobel"
nillable="true"
          type="tns:MobelObjekt" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="MobelObjekt">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="Modellnr"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Bredde"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Dybde"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Hoyde"
type="s:string" />
```

```
          <s:element minOccurs="0" maxOccurs="1" name="Beskrivelse"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Mobelnavn"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="Kategori"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="PrisTekstil"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="PrisHud"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="PrisMicrofiber"
type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="Hud"
type="s:boolean" />
          <s:element minOccurs="1" maxOccurs="1" name="Tekstil"
type="s:boolean" />
          <s:element minOccurs="1" maxOccurs="1" name="Microfiber"
type="s:boolean" />
          <s:element minOccurs="1" maxOccurs="1" name="Safe"
type="s:boolean" />
          <s:element minOccurs="1" maxOccurs="1" name="LosePuter"
type="s:boolean" />
          <s:element minOccurs="1" maxOccurs="1" name="NakkeStotte"
type="s:boolean" />
          <s:element minOccurs="1" maxOccurs="1" name="Sving"
type="s:boolean" />
          <s:element minOccurs="1" maxOccurs="1" name="Byggesystem"
type="s:boolean" />
          <s:element minOccurs="0" maxOccurs="1" name="mobelBilde"
type="s:base64Binary" />
        </s:sequence>
      </s:complexType>
      <s:element name="opprettModellObjekt">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="forhandler"
type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="modell"
type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="opprettModellObjektResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="opprettModellObjektResult"
              type="tns:ModellObjekt" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="hentTrekkFarger">
        <s:complexType />
      </s:element>
      <s:element name="hentTrekkFargerResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="hentTrekkFargerResult"
              type="tns:TrekkFarger" />
```

```
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="TrekkFarger">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="Trekk"
type="tns:ArrayOfString3" />
          <s:element minOccurs="1" maxOccurs="1"
name="opprettelsesDato" type="s:dateTime" />
          <s:element minOccurs="0" maxOccurs="1" name="tt"
type="tns:Trekk" />
          <s:element minOccurs="0" maxOccurs="1" name="alTrekk"
type="tns:ArrayOfTrekk" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfString3">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="string"
nillable="true"
          type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="Trekk">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="trekkType"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="trekkNr"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="trekkNavn"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="alFarge"
type="tns:ArrayOfFarge" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfFarge">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="Farge"
nillable="true"
          type="tns:Farge" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="Farge">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="fargenr"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="navn"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="fargekode"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="fargeBilde"
type="s:base64Binary" />
          <s:element minOccurs="0" maxOccurs="1" name="al"
type="tns:ArrayOfString4" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfString4">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="al"
nillable="true"
          type="s:string" />
```

```
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfTrekk">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="Trekk"
nillable="true"
          type="tns:Trekk" />
        </s:sequence>
      </s:complexType>
      <s:element name="opprettTre">
        <s:complexType />
      </s:element>
      <s:element name="opprettTreResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="opprettTreResult" type="tns:Tre" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="Tre">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="alTre"
type="tns:ArrayOfTreverk" />
          <s:element minOccurs="1" maxOccurs="1"
name="opprettelsesDato" type="s:dateTime" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfTreverk">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="Treverk"
nillable="true"
          type="tns:Treverk" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="Treverk">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="treNr"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="treNavn"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="treBilde"
type="s:base64Binary" />
        </s:sequence>
      </s:complexType>
      <s:element name="opprettFarge">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="trekknr"
type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="trekktype"
type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="fargenr"
type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="opprettFargeResponse">
        <s:complexType>
          <s:sequence>
```

```
            <s:element minOccurs="0" maxOccurs="1"
name="opprettFargeResult"
            type="tns:Farge" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="opprettTrekk">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="trekktype"
type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="trekknr"
type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="opprettTrekkResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="opprettTrekkResult"
            type="tns:Trekk" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="opprettTreverk">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="treverknr"
type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="opprettTreverkResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="opprettTreverkResult"
            type="tns:Treverk" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="opprettModellTreFarge">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="modell"
type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="opprettModellTreFargeResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="opprettModellTreFargeResult"
            type="tns:ModellTreFarge" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ModellTreFarge">
```

```
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="alModellTre"
type="tns:ArrayOfModellTre" />
          <s:element minOccurs="0" maxOccurs="1" name="treNavn"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="trenummer"
type="tns:ArrayOfString3" />
          <s:element minOccurs="1" maxOccurs="1"
name="opprettelsesDato" type="s:dateTime" />
          <s:element minOccurs="0" maxOccurs="1" name="mobelnr"
type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfModellTre">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded"
name="ModellTre" nillable="true"
          type="tns:ModellTre" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ModellTre">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="treNavn"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="treNr"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="alModellTrekk"
          type="tns:ArrayOfModellTrekk" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfModellTrekk">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded"
name="ModellTrekk" nillable="true"
          type="tns:ModellTrekk" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ModellTrekk">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="trekkType"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="trekkNr"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="trekkNavn"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="alModellFarge"
          type="tns:ArrayOfModellFarge" />
          <s:element minOccurs="0" maxOccurs="1" name="mFarge"
type="tns:ModellFarge" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfModellFarge">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded"
name="ModellFarge" nillable="true"
          type="tns:ModellFarge" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ModellFarge">
        <s:sequence>
```

```
          <s:element minOccurs="0" maxOccurs="1" name="fargeNavn"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="fargeNr"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="trekkfarge"
type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:element name="hentBilde">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="bilde"
type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="hentBildeResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1"
name="hentBildeResult" type="s:base64Binary" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="bildeDato">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="fil"
type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="bildeDatoResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1"
name="bildeDatoResult" type="s:dateTime" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="filDato">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="fil"
type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="filDatoResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="filDatoResult"
type="s:dateTime" />
          </s:sequence>
        </s:complexType>
      </s:element>
    </s:schema>
  </wsdl:types>
  <wsdl:message name="opprettForhandlerSoapIn">
    <wsdl:part name="parameters" element="tns:opprettForhandler" />
```

```
    </wsdl:message>
    <wsdl:message name="opprettForhandlerSoapOut">
      <wsdl:part name="parameters"
  element="tns:opprettForhandlerResponse" />
    </wsdl:message>
    <wsdl:message name="opprettModellObjektSoapIn">
      <wsdl:part name="parameters" element="tns:opprettModellObjekt" />
    </wsdl:message>
    <wsdl:message name="opprettModellObjektSoapOut">
      <wsdl:part name="parameters"
  element="tns:opprettModellObjektResponse" />
    </wsdl:message>
    <wsdl:message name="hentTrekkFargerSoapIn">
      <wsdl:part name="parameters" element="tns:hentTrekkFarger" />
    </wsdl:message>
    <wsdl:message name="hentTrekkFargerSoapOut">
      <wsdl:part name="parameters"
  element="tns:hentTrekkFargerResponse" />
    </wsdl:message>
    <wsdl:message name="opprettTreSoapIn">
      <wsdl:part name="parameters" element="tns:opprettTre" />
    </wsdl:message>
    <wsdl:message name="opprettTreSoapOut">
      <wsdl:part name="parameters" element="tns:opprettTreResponse" />
    </wsdl:message>
    <wsdl:message name="opprettFargeSoapIn">
      <wsdl:part name="parameters" element="tns:opprettFarge" />
    </wsdl:message>
    <wsdl:message name="opprettFargeSoapOut">
      <wsdl:part name="parameters" element="tns:opprettFargeResponse" />
    </wsdl:message>
    <wsdl:message name="opprettTrekkSoapIn">
      <wsdl:part name="parameters" element="tns:opprettTrekk" />
    </wsdl:message>
    <wsdl:message name="opprettTrekkSoapOut">
      <wsdl:part name="parameters" element="tns:opprettTrekkResponse" />
    </wsdl:message>
    <wsdl:message name="opprettTreverkSoapIn">
      <wsdl:part name="parameters" element="tns:opprettTreverk" />
    </wsdl:message>
    <wsdl:message name="opprettTreverkSoapOut">
      <wsdl:part name="parameters" element="tns:opprettTreverkResponse" />
    </wsdl:message>
    <wsdl:message name="opprettModellTreFargeSoapIn">
      <wsdl:part name="parameters" element="tns:opprettModellTreFarge" />
    </wsdl:message>
    <wsdl:message name="opprettModellTreFargeSoapOut">
      <wsdl:part name="parameters"
  element="tns:opprettModellTreFargeResponse" />
    </wsdl:message>
    <wsdl:message name="hentBildeSoapIn">
      <wsdl:part name="parameters" element="tns:hentBilde" />
    </wsdl:message>
    <wsdl:message name="hentBildeSoapOut">
      <wsdl:part name="parameters" element="tns:hentBildeResponse" />
    </wsdl:message>
    <wsdl:message name="bildeDatoSoapIn">
      <wsdl:part name="parameters" element="tns:bildeDato" />
    </wsdl:message>
    <wsdl:message name="bildeDatoSoapOut">
```

```
      <wsdl:part name="parameters" element="tns:bildeDatoResponse" />
    </wsdl:message>
    <wsdl:message name="filDatoSoapIn">
      <wsdl:part name="parameters" element="tns:filDato" />
    </wsdl:message>
    <wsdl:message name="filDatoSoapOut">
      <wsdl:part name="parameters" element="tns:filDatoResponse" />
    </wsdl:message>
    <wsdl:portType name="ServiceSoap">
      <wsdl:operation name="opprettForhandler">
        <wsdl:input message="tns:opprettForhandlerSoapIn" />
        <wsdl:output message="tns:opprettForhandlerSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="opprettModellObjekt">
        <wsdl:input message="tns:opprettModellObjektSoapIn" />
        <wsdl:output message="tns:opprettModellObjektSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="hentTrekkFarger">
        <wsdl:input message="tns:hentTrekkFargerSoapIn" />
        <wsdl:output message="tns:hentTrekkFargerSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="opprettTre">
        <wsdl:input message="tns:opprettTreSoapIn" />
        <wsdl:output message="tns:opprettTreSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="opprettFarge">
        <wsdl:input message="tns:opprettFargeSoapIn" />
        <wsdl:output message="tns:opprettFargeSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="opprettTrekk">
        <wsdl:input message="tns:opprettTrekkSoapIn" />
        <wsdl:output message="tns:opprettTrekkSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="opprettTreverk">
        <wsdl:input message="tns:opprettTreverkSoapIn" />
        <wsdl:output message="tns:opprettTreverkSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="opprettModellTreFarge">
        <wsdl:input message="tns:opprettModellTreFargeSoapIn" />
        <wsdl:output message="tns:opprettModellTreFargeSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="hentBilde">
        <wsdl:input message="tns:hentBildeSoapIn" />
        <wsdl:output message="tns:hentBildeSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="bildeDato">
        <wsdl:input message="tns:bildeDatoSoapIn" />
        <wsdl:output message="tns:bildeDatoSoapOut" />
      </wsdl:operation>
      <wsdl:operation name="filDato">
        <wsdl:input message="tns:filDatoSoapIn" />
        <wsdl:output message="tns:filDatoSoapOut" />
      </wsdl:operation>
    </wsdl:portType>
    <wsdl:binding name="ServiceSoap" type="tns:ServiceSoap">
      <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
      <wsdl:operation name="opprettForhandler">
        <soap:operation soapAction="http://tempuri.org/opprettForhandler"
  style="document" />
        <wsdl:input>
```

```
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="opprettModellObjekt">
      <soap:operation
soapAction="http://tempuri.org/opprettModellObjekt" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="hentTrekkFarger">
      <soap:operation soapAction="http://tempuri.org/hentTrekkFarger"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="opprettTre">
      <soap:operation soapAction="http://tempuri.org/opprettTre"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="opprettFarge">
      <soap:operation soapAction="http://tempuri.org/opprettFarge"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="opprettTrekk">
      <soap:operation soapAction="http://tempuri.org/opprettTrekk"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="opprettTreverk">
      <soap:operation soapAction="http://tempuri.org/opprettTreverk"
style="document" />
      <wsdl:input>
```

```
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="opprettModellTreFarge">
      <soap:operation
soapAction="http://tempuri.org/opprettModellTreFarge"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="hentBilde">
      <soap:operation soapAction="http://tempuri.org/hentBilde"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="bildeDato">
      <soap:operation soapAction="http://tempuri.org/bildeDato"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="filDato">
      <soap:operation soapAction="http://tempuri.org/filDato"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:binding name="ServiceSoap12" type="tns:ServiceSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="opprettForhandler">
      <soap12:operation
soapAction="http://tempuri.org/opprettForhandler" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
```

```
    <wsdl:operation name="opprettModellObjekt">
      <soap12:operation
soapAction="http://tempuri.org/opprettModellObjekt" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="hentTrekkFarger">
      <soap12:operation soapAction="http://tempuri.org/hentTrekkFarger"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="opprettTre">
      <soap12:operation soapAction="http://tempuri.org/opprettTre"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="opprettFarge">
      <soap12:operation soapAction="http://tempuri.org/opprettFarge"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="opprettTrekk">
      <soap12:operation soapAction="http://tempuri.org/opprettTrekk"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="opprettTreverk">
      <soap12:operation soapAction="http://tempuri.org/opprettTreverk"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
```

```
    <wsdl:operation name="opprettModellTreFarge">
      <soap12:operation
soapAction="http://tempuri.org/opprettModellTreFarge"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="hentBilde">
      <soap12:operation soapAction="http://tempuri.org/hentBilde"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="bildeDato">
      <soap12:operation soapAction="http://tempuri.org/bildeDato"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="filDato">
      <soap12:operation soapAction="http://tempuri.org/filDato"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="Service">
    <wsdl:port name="ServiceSoap" binding="tns:ServiceSoap">
      <soap:address
location="http://129.241.150.101/ekornes/service.asmx" />
    </wsdl:port>
    <wsdl:port name="ServiceSoap12" binding="tns:ServiceSoap12">
      <soap12:address
location="http://129.241.150.101/ekornes/service.asmx" />
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```