Handelshögsskolan vid
Göteborgs Universitet
Institutionen för informatik

Objektmodellering och säkerhet vid systemutveckling

Examensarbete 1 vid institutionen för informatik. Vårterminen 1999.

Författare: **Jonas Thorsell**
Magnus Wahlqvist

Handledare: **Wera Tegner-Johansson**

Sammanfattning

Vid konstruktionen av ett system som skall implementeras på ett företag krävs att hänsyn tas till ett antal olika faktorer. Det är av stor vikt att systemet blir lätt att vidareutveckla och att det följer företagets säkerhetsrutiner. Kravet på dokumentation är ofta omfattande.

Vi har valt att studera modellering, brandväggar och lösenord för att kunna ta hänsyn till dessa då vi utvecklat ett WWW-anpassat kundsystem åt EHPT(tidigare Ericsson Hewlett-Packard Telecommunications). Företaget konstruerar och säljer bland annat billing-system till teleoperatörer, det vill säga system som fakturerar telefonabbonenten. EHPT:s kunder upptäcker vid körning av sina system att de vill ha vissa förändringar i programmet. Då de meddelar detta till EHPT kallas deras önskemål Request For Change(RFC). Varje RFC är unik och information om den sparas i en MSAccess databas. Om kunden vill ha information om status på sin RFC måste hon idag ringa EHPT där den ansvarige går in i databasen. För att erbjuda kunderna ökad service vill man ge dem möjlighet att gå in via WWW och avläsa statusen. Vi fick i uppdrag av EHPT att konstruera detta system.

För att motsvara de krav som ställs på systemets utvecklingsbarhet har vi valt att studera UML(Unified Modeling Language). Detta är en standard för objektorienterad utveckling antagen av OMG(Object Management Group). Det är ett mycket kompetent modelleringsspråk vilket är lämpligt att använda dels som kommunikationsplattform mellan användare och utvecklare samt som stöd för utvecklarna. Vår slutsats är att det är användbart men en aning komplicerat. Vi rekommenderar att vidare studier inom området görs som en jämförelse mellan UML och ett annat språk.

Då brandväggar är ett hinder för vårt system har vi även fördjupat oss i dessa. Vårt system måste hämta data från en server innanför brandväggen och kopiera denna till en Web-server utanför brandväggen. Vi har gått igenom olika typer av brandväggsarkitekturer för att förbereda oss på de problem som kan uppstå vid implementeringen av vårt system. Vi har dock inte implementerat uppdateringen av information genom brandväggen utan endast lämnat ett förslag på hur detta kan lösas.

För att skydda informationen från obehöriga användare har vi gått igenom olika säkerhetsrutiner, dels kryptering, dels lösenord. Kryptering av information har inte ansetts nödvändig då informationen inte har ett kritiskt värde för företaget. Däremot har vi valt att skydda systemet med ett lösenordsförfarande. Hur detta i praktiken fungerar kan vi inte gå djupare in på av sekretesskäl. Implementeringen av lösenordsfunktionen är inte utförd i skrivande stund.

Inledning

EHPT (tidigare Ericsson Hewlett-Packard Telecommunications), utvecklar och säljer bland annat system för fakturering till teleoperatörer. När kunden använder programvaran i sin verksamhet uppstår ofta ett behov av ny funktionalitet. Kundens önskemål skickas till EHPT och kallas då en Request For Change (RFC). När RFC:n inkommer till EHPT så tilldelas den en ansvarig person. Vidare lagras RFC:n(ett Word-dokument) på företagets server. I en Accessdatabas lagras information om RFC:n, bland annat dess status, vem som är ansvarig för den och en länk till själva dokumentet. Idag måste kunden ringa till sin kontaktperson på EHPT för att få information om hur arbetet med RFC:n fortskrider. EHPT vill att kunden själv ska kunna avläsa statusen på sin RFC via WWW. Vi har fått i uppdrag att utveckla detta system.

Problemställning

I analysen av uppdraget har vi stött på en rad problem vilka skulle kunna behandlas i vår uppsats. Vi väljer här att inledningsvis redovisa dessa problem i en probleminventering. Därefter går vi igenom de problem vi valt att analysera i uppsatsen.

Probleminventering

- Ett problem är hur kommunikationen mellan systemets olika delar lämpligast sker. Här finns olika sätt att lösa det hela på. Vid ställningstagandet måste hänsyn tas till systemets snabbhet och säkerhet.

- Användarpolicy är ytterligare ett problem. Här avser vi då upprättandet av en policy för i vilken utsträckning användarna skall kunna ändra data i systemet, vilken typ av information som de ska få tillgång till och vilka användargrupper som ska få tillgång till systemet.

- Ett tredje problem är att göra en analys av vilket program/scriptspråk som systemet skall byggas i. Här är det lämpligt att göra en inventering av hur systemet ska användas, det vill säga om det finns behov av plattformsoberoende, på vilket sätt systemet skall uppdateras i framtiden, samt sist men inte minst vilka kunskaper utvecklarna har.

- Vi vill skapa ett system som är lätt att vidareutveckla. Hur görs detta på bästa sätt? Här bör vi välja en modelleringsmetod och beskriva den samt använda metoden för att beskriva vårt system.

- Våra framtida arbetsuppgifter kommer med stor sannolikhet innebära att hänsyn tas till olika typer av säkerhetsproblem. Ett sådant är att de flesta företag använder sig av brandväggar som måste kringås om man vill presentera information utanför företaget. Detta blir ett problem då informationen som skall presenteras kontinuerligt uppdateras

innanför brandväggen. Vi kommer att beskriva olika typer av brandväggar samt hur vi tar hänsyn till dessa i arbetet med vårt system.

- För att skydda informationen som skall exponeras på WWW krävs att vi skapar ett lösenordsförfarande och eventuell kryptering vilket hindrar obehöriga från att få tillgång till information. Vi avser att översiktligt beskriva olika metoder för detta samt ange hur vi valt att implementera dessa i vårt system.

Vi har valt att fokusera på de tre senaste problemställningarna. Dessa är de vi finner mest intressanta. Dessa kan sammanfattas som följer: Hur utvecklar man ett välmodellerat system vilket när det implementeras tar hänsyn till organisationens säkerhetsrutiner?

Avgränsning

Vi beskriver nedan hur vi avgränsat de tre problemställningarna så att de blir hanterbara inom ramen för uppsatsen.

Modelleringsmetod
Genomgången av modelleringsmetod kommer att inriktas på de delar av vald metod som kan tänkas vara användbara vid modellering av vårt system. Detta kommer givetvis att innebära att vår genomgång ger en förenklad bild av metoden. I analysdelen av uppsatsen beskriver vi hur vi tillämpat metoden för att bygga systemet åt EHPT.

Brandväggar
Olika brandväggsarkitekturer kommer att presenteras i uppsatsens teoridel. Presentationen har för avsikt att ge oss en grundinsyn i hur brandväggar fungerar för att kunna hantera den brandvägg som finns på EHPT. Därefter anges i analysen vilken arkitektur som förekommer på EHPT samt hur vi har för avsikt att hantera de problem som denna orsakar.

Säkerhet
Uppsatsens teoridel presenterar kortfattat olika säkerhetsförfaranden och viktiga punkter att tänka på vid lösenordshantering. Vi har inte för avsikt att på ett djupare plan beskriva olika krypteringsalgoritmer. I analysen redogör vi för hur vi löst problemen.

Metod

Vetenskapligt förhållningssätt

Inom det vetenskapliga förhållningssättet[i] brukar två huvudinriktningar nämnas; positivism och hermeneutik. De kan sägas representera ytterligheterna på en skala. En av de stora skillnaderna mellan inriktningarna är den strävan de representerar; generaliserbar kunskap respektive djupare förståelse.

Positivism
Positivismen härstammar från människans behov av att få fenomen tydligt förklarade. Det är genom insamlande av objektiva och mätbara data som går att jämföra sinsemellan som förklaringen nås.[ii] Med sådan data kan sedan slutsatser dras för att i slutändan ge generaliserbar kunskap. Positivism är en kunskapssyn som hävdar att vetenskapen är begränsad av det erfarenhetsmässigt givna. Inom naturvetenskapen har detta förhållningssätt varit det dominerande sedan en lång tid tillbaka. Det positivistiska förhållningssättet ger upphov till studier som är kvantitativa till sin art.

Hermeneutik
Hermeneutik kallas den vetenskap som har sin grund i tolkning av texter. Det hermeneutiska förhållningssättet avser att skapa förståelse för det beforskade fenomenet. Det är via förståelse som djupare kunskap erhålls.[iii] Den djupare kunskap som erhålls är, menar hermeneutikerna, mer fullständig och rättvisare än kunskap som framhåller sin objektivitet. Vissa grenar inom hermeneutiken menar till och med att det inte finns objektiva data utan den data som finns är alltid till viss del subjektiv. Detta förhållningssätt är betydligt yngre är positivismen men är det förhärskande inom samhällsvetenskapen. Det hermeneutiska förhållningssättet ger upphov till kvalitativa studier.

Vårt förhållningsätt
Att välja ett specifikt förhållningsätt och göra ett klart ställningstagande anser vi vara mycket svårt och inte heller önskvärt. I vårt arbete kommer förhållningssättet till viss del vara hermeneutiskt. Vi kommer att tolka de svar och åsikter, vi erhåller på EHPT, med avseende på vårt problem. Vi anser att den kunskap vi får genom våra kontaktpersoner är mycket värdefull och att den kan ytterligare förädlas genom att vi sätter den i ett större sammanhang och tolkar den. Det hermeneutiska synsättet lämnar betydligt större utrymme för oss att kunna dra egna viktiga slutsatser.

Att helt utelämna det positivistiska förhållningssättet är inte bra då det också fyller en viktig funktion inom forskning. Vissa typer av data går att kvantifiera utan att det innehållsmässiga värdet går förlorat. I sådana fall är det dumt att inte möta frågan med ett öppet sinne och se på den på det sätt som gynnar resultatet av arbetet oavsett om det är ett hermeneutiskt eller positivistiskt synsätt.

Vetenskapligt angreppssätt

Inom forskning arbetas det med så kallad teoriproduktion. Denna produktion bygger på data och information om det område som studeras. Det vanliga är att kalla erfarenhetskunskap (byggnadsmaterialet) för empiri. Ett av forskningens huvudmål är att relatera teori och verklighet till varandra. Vi kommer att behandla tre alternativa arbetssätt för teoriproduktionen, dessa är deduktion, induktion samt abduktion.

Deduktion
Patel & Davidson säger att den som arbetar deduktivt följer bevisandets väg.[iv] Den deduktiva ansatsen kännetecknas av att forskaren utifrån befintliga principer och teorier härleder hypoteser som sedan prövas mot verkligheten. Den befintliga teorin styr forskaren mot den information som berörs och hur den skall tolkas. Resultaten från undersökningen skall sedan relateras tillbaka till den redan befintliga teorin.

Induktion
Den som arbetar induktivt följer, enligt P & D, upptäckandets väg. Forskaren studerar här studieobjektet utan att utgå från etablerad och vedertagen teori. Detta innebär dock inte att han arbetar utan idéer och föreställningar. Målet med den induktiva ansatsen är utifrån det undersökta skapa nya egna teorier.

Abduktion
Alvesson presenterar ett tredje angreppssätt, abduktion. Det innebär att ett enskilt fall tolkas med ett hypotetiskt övergripande mönster som, om det vore riktigt, förklarar fallet ifråga. Tolkningen bör sedan styrkas genom nya iakttagelser (nya fall). Metoden blir härigenom en kombination av de tidigare nämnda deduktiva och induktiva angreppssätten, men tillför även nya aspekter. Under processens gång utvecklas dels det empiriska tillämpningsområdet, dels justeras och förfinas även teorin (dvs det föreslagna övergripande mönstret).

Vårt vetenskapliga angreppssätt
Vi är av samma uppfattning som Alvesson att induktiv respektive deduktiv ansats inte är ömsesidigt uteslutande i en undersökning. Inom det beforskade finns det alltid olika områden. Det gäller då att för varje område se vilken ansats som ger bästa resultatet. I vårt fall passar den abduktiva ansatsen bäst. Vi avser att förklara problemet bland annat genom att konstruera ett program. Om programmet fungerar så förklarar det faktiskt fallet ifråga, åtminstone på ett sätt. Ofta blir det att det abduktiva angreppssättet får en dragning åt antingen induktion eller deduktion. Vad gäller vår undersökning kan vi snabbt konstatera att vi har en klar dragning åt det induktiva hållet. Vi utgår inte från vedertagen teori utan försöker skapa något som passar just vårt problem. Det sker givetvis genom att ta delar av redan etablerade teorier men vi kombinerar dem så att de passar oss på bästa sätt.

Forskningsmetod

När forskningsmetod diskuteras är det vanligaste att tala om två huvudområden, nämligen den kvantitativa och den kvalitativa metoden. Det finns mängder av andra metoder som också används inom forskningen men ofta har de rötter i någon av de ovan nämnda metoderna. Vi avser att belysa, som vi anser det, de två huvudinriktningarna; kvantitativ och kvalitativ metod.

Kvantitativ
Den kvantitativa metoden har som vi tidigare nämnt sitt ursprung i positivismen. Det som utmärker metoden är att forskaren har ett så kallat "utifrån" -perspektiv på forskningsobjektet. Detta innebär att forskaren studerar objektet som helt skilt från sin egen påverkan och aldrig sätter sig in i informandens situation. En viktig del i undersökningen är att forskaren styr undersökningen med t ex ett strikt frågeformulär. För att det skall gå att jämföra den data som erhålls är det viktigt att informanderna erhåller exakt samma stimuli. Oftast sker undersökningarna i enkätform. Det som sökes är det gemensamma, generella och representativa hos undersökningsenheterna. De svar som erhålls omarbetas till siffror för att kunna bearbetas statistiskt. Genom att relatera olika variabler till varandra kan samband mellan dem undersökas. Det är dessa samband som är de viktigaste resultaten. Den kvantitativa metoden utgår i första hand från forskarens idéer om vad som skall stå i centrum i undersökningen.[v]

Kvalitativ metod
Med ett hermeneutiskt förhållningssätt är det naturligt att begagna sig av den kvalitativa metoden. Utmärkande för denna metod är att forskaren har ett "inifrån" -perspektiv. Detta synsätt innebär att forskaren anser sig som en del av forskningsobjektet. Han försöker sätta sig in i informandens situation och se världen med dennes ögon. Forskaren försöker inte styra intervjun utan följer informanden för att få med det som denne tycker är viktigt. Detta leder till att ingen undersökning blir den andra lik inom undersökningsområdet.

Forskaren försöker att erhålla en djupare förståelse om frågeställningen han har. Resultaten från undersökningarna bearbetas inte efter några bestämda regler utan forskaren utgår ofta från något som kallas den hermeneutiska spiralen. Forskaren pendlar i den mellan förförståelse och förståelse. Detta innebär att forskaren först förstår en liten del och med hjälp av den del kan han förstå en lite större och med hjälp av den lite större delen är det möjlig att förstå ytterligare delar som bildar, till slut, helheter. De viktigaste resultaten i bearbetningsprocessen är forskarens egna tolkningar.

Kvantitativ kontra kvalitativ metod
Vi har tidigare redogjort för vårt vetenskapliga förhållningssätt och konstaterat att vi i grunden bekänner oss till den hermeneutiska skolan. Med detta förhållningssätt är det naturliga att använda sig av en kvalitativ metod. Vi anser dock att liksom det är farligt att bekänna sig till ett specifikt förhållningssätt att det är farligt att bara bekänna sig till en typ av metod. Morgan & Smicich stöder denna uppfattning och menar att det inte är metodvalet som är det väsentliga utan det är den intellektuella ansträngningen och

resultatet som ger tyngd.[vi] Alvessons teori om att metoderna bör kombineras för att på bästa sätt passa det valda problemområdet.[vii] Denna uppfattning stöder vi till fullo och påpekar än en gång det viktiga med att möta problemen med ett öppet förhållningssätt.

Den största svårigheten med kvantitativ metod som vi ser är möjligheten till att vara helt objektiv. Att helt avskärma sig från forskningsobjektet torde inom samhällskunskapen vara omöjligt. Detta är något som Holme & Solvang också påpekar när de menar att det är forskarens skyldighet att tolka och analysera utifrån de kunskaper och förhållningsätt han har.[viii]

Vår uppsats kommer till stor del att bygga på en kvalitativ metod. För att kunna genomföra undersökningen på ett gripbart sätt är personliga intervjuer/samtal den bästa metoden för oss. Vi anser att möjligheten att kunna utveckla viktiga resonemang med informanden är avgörande för resultatet av vår undersökning. Med tanke på att vi inte använder ett strikt frågeformulär utan ställer olika frågor till de personer vi pratar med, beroende på deras kompetens, kan vi inte jämföra de svar vi får. Detta är inte heller önskvärt då vi vill ha en bred utgångspunkt att bygga på.

Datainsamlingsmetod

När en undersökning skall genomföras finns det i huvudsak två typer av data att tillgå; primärdata och sekundärdata. Primärdatan hämtas från empirin och sekundärdatan är data som redan insamlats och bearbetats vid tidigare forskning.[ix]

Primärdata
Vid kvalitativ metod används främst fallstudier och frågeundersökning med avsiktliga urval. Vi har valt att använda oss av fallstudien för att analysera problemen. Dock använder vi oss av, som tidigare nämnts, frågeundersökningar med personer som kan tänkas kunna tillföra viktig information för att vi skall kunna analysera problemet

Sekundärdata
Sekundärdatan används för att ge oss en teoretisk referensram och de teoretiska verktyg vi behöver för att kunna angripa problemet. Källorna som används (böcker, artiklar, avhandlingar och rapporter) hjälper oss att få en överblick av problemområdet och gör det möjligt för oss att ställa rätt typ av frågor.

Urval
När det gäller urval finns det enligt Lundahl och Skärvad två huvudtyper av urvalsmetoder; sannolikhetsurval och icke-sannolikhetsurval. Sannolikhetsurval karaktäriseras av att alla urvalsenheter i populationen har en känd, men inte nödvändigtvis lika stor sannolikhet att bli utvald.[x] Vi kommer att använda oss av ett riktat urval som är ett icke-sannolikhetsurval.

Holme och Solvang anser att man kan öka informationsinnehållet genom att använda sig av intervjupersoner som på goda grunder kan antas ha riklig kunskap om de företeelser

man undersöker.[xi] Dessa personer är ofta mer medvetna än andra och reflekterar oftare över sin situation. En risk med att intervjua dessa personer är att de kan ge friserade och förvrängda beskrivningar samt att de ofta besitter förtroendegivande positioner. Tar man dessa faktorer i beaktande kan det avsiktliga urvalet användas med fördel anser vi.

Intervjuer
Enligt Carlsson finns det fyra typer av intervjuer; strukturerad-, ostrukturerad-, fokuserad- och fri intervju.[xii] I den strukturerade intervjun är intervjuaren bunden till ett formulär och svaren skall härigenom bli jämförbara. Den ostrukturerade intervjun är friare och intervjun rör ett tema och utgår ifrån en intervjuguide. När det gäller den fokuserade intervjun testar forskaren i förväg uppsatta hypoteser, och omformulerar dessa om så krävs. I användandet av en fri intervju är rollerna närmast ombytta jämfört med en traditionell intervju, då intervjuaren håller sig så passiv som möjligt och låter intervjuobjektet styra intervjuförloppet.

I vårt fall har vi valt att göra kvalitativa intervjuer och då använda oss av den ostrukturerade formen. Detta gör vi av flera skäl. Vi anser att styrkan i den kvalitativa intervjun ligger i att den liknar en vardaglig situation och ett vardagligt samtal. Detta möjliggör för intervjuaren att läsa mellan raderna och ur samtalet "vaska fram" den information som han/hon är intresserad av. Den ostrukturerade intervjuns fria form skapar då en möjlighet för intervjuaren att leda och att ledas in på intressanta och fruktsamma spår. Av de frågeundersökningar vi utfört kan de allra flesta anses som rena samtal och inte som en intervju.

Sanningskriterier

För att en undersökning skall kunna betraktas som trovärdig måste vissa kriterier vara uppfyllda. Oftast talas det om tre kriterier, dessa är; validitet, reliabilitet och generalitet.[xiii]

Validitet
Validitet handlar om att veta vad det är som skall undersökas. Ofta finns det ett "glapp" mellan vad vi vill undersöka och vad vi faktiskt undersöker. För att försäkra sig om en god validitet finns det många sätt att gå tillväga. Ett sätt är att någon som är väl insatt i problemområdet granskar instrumentet som avses att användas. Om det till exempel är en intervju som skall ske granskar den utomstående personen frågorna. Ett annat sätt är att göra provundersökningar dvs testa instrumentet på en undersökningsgrupp som liknar den som är föremål för huvudundersökningen.

Reliabilitet
Begreppet reliabilitet innebär att instrumentet som används skall stå emot slumpinflytande av olika slag. När t ex en intervju sker kommer sanningar och sådant som inte är riktigt sant fram. Instrumentets förmåga att särskilja dessa olika fenomen kan sägas vara instrumentets förmåga att ge ett resultat med högre eller lägre grad av reliabilitet. För att försäkra sig om att erhålla en god reliabilitet, om vi tar intervjun som

exempel igen, kan olika tekniker användas. Ett sätt är att intervjuaren åtföljs av en eller flera observatörer vars uppgift blir att studera olika beteenden hos informanden när denne svarar. För uppnå så hög reliabilitet som möjligt gäller att intervjuaren och observatören är tränade.

Generalitet
Med generalitet menas undersökningens möjlighet eller förmåga att vara applicerbar på andra miljöer än den undersökta. Viktiga inslag för att göra generaliseringen genomförbar är att det gäller att kunna forma de resultat som uppnås på ett universellt, allmängiltigt, sätt som möjligt. Här kan t ex metaforer hjälpa forskaren att finna nya möjliga lösningar på problemet. I övrigt är det bara forskarens fantasi som sätter gränserna för generaliteten.

Användbarheten av sanningskriterierna
Är det då möjligt att uppnå tillräckligt hög reliabilitet, validitet och generalitet med en i huvudsak kvalitativ metod, som trots allt är vår huvudsakliga metod? Holme & Solvang menar att validiteten inte har central roll inom kvalitativ forskning, syftet är att nå en bättre förståelse och inte att ha den statistiska representativiteten i centrum.[xiv] Vi tror dock att en viss validitet är önskvärd, trots metodval, och att det är möjligt att uppnå. En möjlighet till en bättre validitet är om någon som är insatt i ämnet kan hjälpa till med att se om intervjuaren kan, till en början, leda in informanden på rätt kurs. I vårt fall är vi nästan uteslutande hänvisade till vår egen kunskap för att förmå leda in våra personer mot rätt område. Det faktum att vi är två gör att vi kunnat komplettera varandras frågor allt eftersom samtalet fortgått.

Problemet att få reliabilitet är inte lika överhängande i en kvalitativ studie. Det finns en närhet till informanden som hjälper forskaren. Dock finns möjligheten att forskaren inte förstår och upplever situationen på ett riktigt sätt. Med en stor medvetenhet tror vi oss kunna uppnå tillräckliga sanningskriterier för att uppsatsen skall bli trovärdig.

Generalitet är inget som vi strävar efter i vår kvalitativa undersökning. Det kriteriet är mycket hårt kopplat till den kvantitativa metoden. Vi anser dock att en viss generalitet är möjlig att nå även vid användandet av den kvalitativa metoden. Våra slutsatser kommer att vara användbara även för andra organisationer, än EHPT. Genom det väl avgränsade problemet anser vi att om en organisation känner igen problemet så kommer delar av våra slutsatser att vara tillämpliga.

Modellering

Det finns enligt oss flera faktorer att ta hänsyn till vid inledningen av ett utvecklingsarbete. Den första frågan borde vara: Varför ska jag använda en modell? Användarna är ju inte intresserade av modellen, de vill ha en programkod som exekverar och uträttar vad de bett om. Den andra frågan är: Behövs en modell i just det här fallet? Det finns ju inte två projekt som är exakt likadana. En liten applikation som löser ett tillfälligt problem kanske inte behöver en modell. Det är viktigt att ha i åtanke att modellen är till för att snabba upp kodningen, minska på kodens omfång och att kommunicera med andra programmerare och användare. Om systemet utvecklas med hjälp av objektorienterad teknik resulterar det i att metoder hamnar utspridda i olika klasser. Spridningen kan medföra att det blir svårt att följa koden, för att underlätta läsningen bör modeller och diagram användas. Ett annat tungt skäl för att använda sig av modellering är att systemet blir lättare att bygga ut i framtiden.

Vilken modelleringsmodell är lämplig?

”Det är i stort sett omöjligt att bygga ett modernt, komponentbaserat system utan att först ha tänkt igenom en arkitektur och en väl inarbetad metodik, om man sedan ska få förvaltningen av systemen att fungera.” Säger Michael Welin-Berger, konsultchef, Cap Gemini[xv]. Ivar Jacobson på Rational Software menar att tillämpandet av UML kommer att leda till att efterfrågan på programmerare kommer att minska. I framtiden kommer programutvecklare främst att använda sig av modelleringsverktyg som genererar kod. Tyngdpunkten i utvecklingsarbetet kommer att ligga på analys- och designfasen i stället för på programmeringen. Utvecklingen kommer att bli mer ingenjörsmässig. Användandet av moduler kommer att bli ett centralt inslag i programutvecklingsprocessen vilket innebär att det är mer betydelsefullt för utvecklaren att kunna läsa och förstå vad olika moduler utför. UML ger en förståelse som gör det lättare att förstå vad andra har utvecklat. Därmed blir det lättare att återanvända kod.[xvi]

Då vi ville arbeta med ett modelleringsspråk som troligtvis kommer att användas som standard under den närmaste framtiden valde vi att titta närmare på UML (Unified Modeling Language). Detta har utvecklats av Ivar Jacobsson, James Rumbaugh och Grady Booch vid Rational Software. Dessa tre har tidigare varit modelleringsgurus på varsitt håll och därmed konkurrenter. Numera samarbetar de i Rational Software vilket har lett till utvecklingen av UML. Standard för modellering i branschen sätts av OMG (Object Management Group). Gruppen består av över 800 företag vilka enas kring dess standarder. UML:s notationsstandard fastslogs av OMG i september 1997.
Vi har konstaterat att det är av yttersta vikt att vi lär oss ett språk väl. UML borde vara det naturliga valet att använda för kommunikation mellan systembyggarna och användarna men även systembyggarna sinsemellan. En av våra frågeställningar innebär att vi ska ta reda på hur vi kan skapa ett system som är lätt att bygga ut. UML är en modelleringsmetod som, vad det verkar, kan skapa en god grund för kommunikation och framtida utbyggnad. Vi måste dock tillägga att även UML har fått ta emot kritik. Bland

annat har det ansetts att det är onödigt komplicerat och att det fortfarande är under förändring.[xvii] Kritiken har givetvis tillbakavisats av Rational Software.

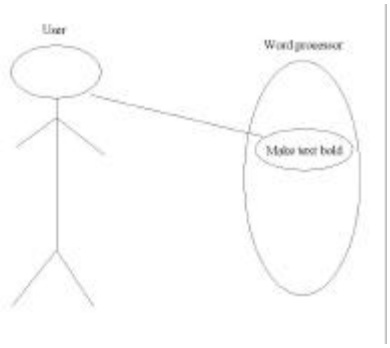
Definition av UML

Ett beskrivningsspråk för objektorienterad systemutveckling. Kombinerar notationen från Booch, Use-cases och OMT[xviii]. Notation är den grafiska delen av en modell. UML skapar ett universellt modelleringsspråk som kan användas med vilken metod som helst. Med grafiken beskrivs hur klasser, associationer och förhållanden representeras.[xix] UML har som syfte att hjälpa programmerare att skapa bra objektorienterade modeller.[xx]

Use-Case diagram

Use–Case diagram utvecklades ursprungligen av Ivar Jacobson och har infogats i UML. För att skapa en modell som beskriver vad som ska göras inleds arbetet med att bygga Use-case diagram. Dessa beskriver användarens agerande i systemet. Ett exempel är om användaren skriver på en ordbehandlare. Då kan ett Use-case vara ”Skapa fet text.”.[xxi] Detta beskrivs grafisk som i figur 1.

Fig. 1.

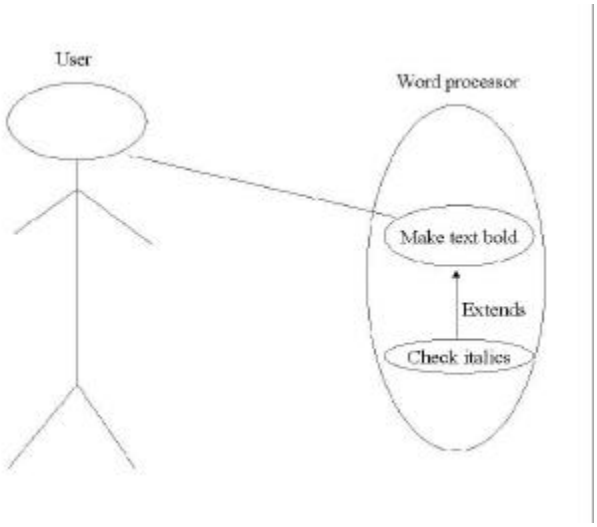


Syftet med use case diagram är tredelat.

1. Att skapa utvecklarens bild av vad användarna vill ha.
2. Att skapa en utgångspunkt för att hitta objektklasserna.
3. Att skapa en utgångspunkt för att hitta vilka operationer som ska utföras av respektive klass.[xxii]

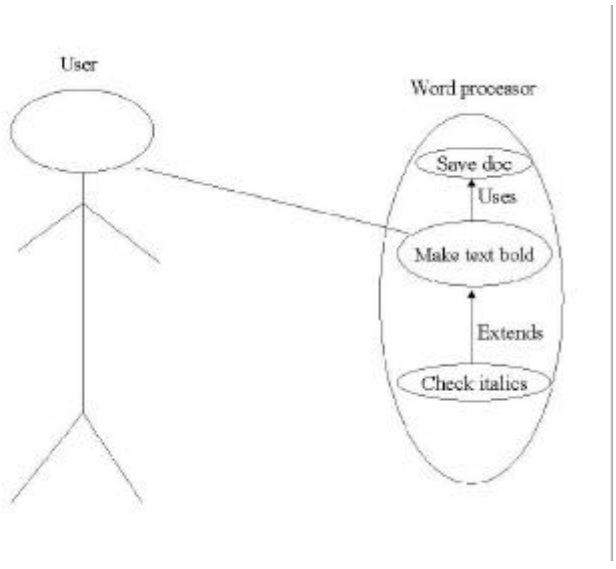
En modell av det här slaget ger en bra kommunikationslänk mellan utvecklare och användare. Båda grupperna kan förstå den förhållandevis enkla layouten och dessutom är den en bra grund att bygga vidare på för utvecklaren. Ett use case kan sägas vara en utveckling av ett annat use case om det är likadant som ett annat men uträttar lite mer. Ett exempel på detta är en utveckling av fig. 1, se fig. 2. Där ett use case lagts till vilket hindrar formatering till fet text om texten redan är kursiv. En use Case modell skapas genom att modellkonstruktören inledningsvis ritar upp de enklare förhållandena och därefter tänker: ”Vad kan gå fel i det här use caset?” och ”Hur kan detta fungera på ett annat sätt?”.[xxiii] Dessa alternativ ritas därefter in som egna use case ”bubblor” och på pilen mellan dem och ursprungsbubblan anges att det är en utveckling av den.

Fig. 2



Ett annat exempel på use case ”bubblor” kan vara om en funktion kommer användas av flera andra use cases. Istället för att klippa och klistra in funktionen i flera andra läggs den in som ett eget use case och på pilen anges att andra använder sig av den. Se fig. 3.

Fig . 3.



Klassdiagram

Klassdiagram är en central del inom objektorientering. Klassdiagrammet beskriver objekten i ett system och de olika relationerna mellan dem. Klassdiagrammet består inledningsvis av entitetsklasser, det vill säga klasser som står att finna i användarens verklighet. Därefter utökas diagrammet med gränssnittsklasser, kontrollklasser och abstrakta klasser i designfasen.[xxiv] Det finns enligt Fowler tre olika perspektiv på hur den som konstruerar ett klassdiagram kan se på det. De är: konceptuellt, specificerat och implementering.[xxv]

Konceptuell design

Designern ritar ett diagram som representerar begreppen i systemet såsom användarna uppfattar det. Vi tolkar det som att detta innebär att designern utgår från de begrepp som framkommer vid intervjuer med användarna och utifrån dessa ritar en modell. Ingen hänsyn tas till vilket programspråk som ska användas. På så vis blir diagrammet språkoberoende.

Specifikation

Under specifikationsfasen utvecklas modellen som konstruerats under den konceptuella designfasen. Nu tas även hänsyn till mjukvaran och utvecklaren kontrollerar vilka gränssnitt som ska förekomma i systemet.

Implementering

Vid implementering är klasserna färdigritade och modellören ser över hur systemet ska implementeras.

Fowler skriver även att förståelse av perspektiven är grundläggande för att kunna rita och läsa diagram. Han fortsätter att det inte finns en klar gräns mellan de olika perspektiven.

Att skapa klasser

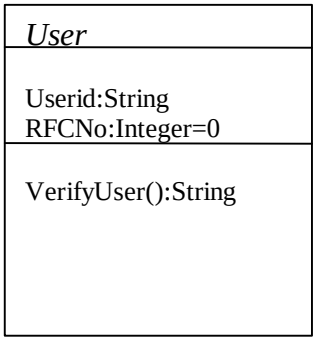
För att skapa ett bra klassdiagram är en lämplig metod *KRB seven-step method* som beskrivs av Brown. Denna innehåller följande:

1. Ta fram kandidatklasser
2. Definiera klasser
3. Etablera relationer
4. Expandera många till många relationer
5. Lägg till attribut
6. Normalisera
7. Definiera beteenden.[xxvi]

Klasser

Då en designer ritar upp en klass anges dess namn, attribut och operationer. En klass kan ritas på följande vis.

Fig. 4



Överst anges klassens namn. Därunder dess attribut i syntaxen:
Attributnamn:typ=standardvärde
Längst ner anges dess operationer/funktioner(operations). De skrivs i syntaxen Namn(argumentlista):retur-typ-uttryck[xxvii]

Kardinalitet

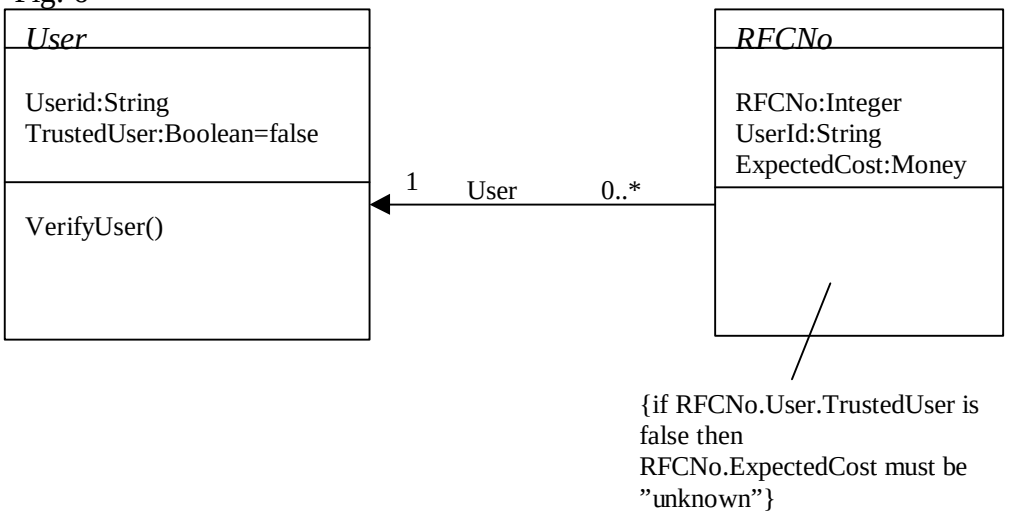
Kardinalitet beskrivs på följande vis i UML.[xxviii]

Fig. 5

An A is always associated with one B	An A is always associated with one or more B	An A is associated with zero or one B	An A is associated with zero, one, or more B
A ——1—— B	A ——1..*—— B	A ——0..1—— B	A ——*—— B

De olika klasserna knyts samman genom associationer. Dessa tecknas genom att en linje dras mellan de olika klasserna. Associationen har två roller(eng. Roles), en i vardera riktningen på associationen. Antingen ger designern rollen en etikett eller så namnges den efter den klass den hänvisar till. Pilen anger att RFCNo innehåller information om vilka RFC:s som är knutna till en viss User. Det går däremot inte att avgöra vilka RFC:s som hör till en viss användare genom att avläsa User.[xxix]

Fig. 6



Texten inom hakparenteser är ett villkor. UML har ingen strikt syntax för hur villkor ska skrivas mer än att de måste skrivas inom hakparenteser.[xxx]

Säkerhet

I dag har utbredningen av Internet gjort det möjligt för i stort sett vem som helst att ta del av information från företagen exempelvis genom dess hemsidor. Problemet i dag är dock att all information som finns på företagens nätverk inte skall vara tillgänglig via nätet. Vidare har år 2000 problematiken också lyft fram IT och säkerhetsfrågorna på bordet.[xxxi] Datasäkerhet är ett rymligt begrepp. För att göra det mer gripbart måste en finare uppdelning göras. En uppdelning kan vara att tala om intern och extern säkerhet.

Intern

Traditionellt har personer inom företagen stått för 80-95 procent av alla säkerhetsrisker.[xxxii] Vi avser med intern datasäkerhet skyddet av den information som personer (anställda) med fysisk tillgång till hårddiskar, servrar osv. kan komma åt. Att upptäcka intrång av intern natur är svårt. Genom att registrera var och när personerna på företaget loggar på kan en viss kontroll upprätthållas. Att införa olika behörighetsnivåer är också ett sätt att skydda viss information. Det kan dock vara så att en person som har behörighet att komma åt värdefull information kan ha ett bedrägligt beteende och då hjälper inga behörighetsnivåer i världen. Det vanligaste skyddet är att ha någon form av lösenordsskydd. Ofta lämnas datorerna olåsta och då är det fritt fram för vem som helst att undersöka den oskyddade datorn. Ytterligare ett sätt för organisationer att skydda sig är att de anställda får skriva på avtal om tystnadsplikt. För ett företag kan en brist i den interna datasäkerheten leda till stora förluster både i reda pengar och i prestige.

Extern

Med extern datasäkerhet avser vi den säkerhet som används för att skydda ett företags information från utomstående, dvs. ej anställda. Enligt en amerikansk undersökning[xxxiii] gjordes 1995 hela 900 miljoner försök till intrång på 2,5 miljoner datorer. Svenska Riksrevisionsverket rapporterar[xxxiv] i år att utomstående numera svarar för de flesta databrotten. Ett av de tydligaste exemplen där den externa datasäkerheten inte var tillräcklig var när någon crackade sig in hos Telia och lade ut 200 000 Internet-abonnenters lösenord på Internet.[xxxv]

Olika sätt att uppnå säkerheten

Kryptering
Ett av de bästa sätten att skydda information är att kryptera den. Att kryptera information är ingen ny företeelse. Redan Julius Caesar[xxxvi] använde kryptering för sina meddelanden. Vid kryptering undviks att informationen skrivs i klartext. Ett enkelt sätt att kryptera är till exempel att byta ut bokstäver efter ett mönster. Numer skapas ofta en avancerad algoritm för att kryptera informationen. För att kunna läsa information som är krypterad används en nyckel. Den innehåller uppgifter om vilket mönster som används vid krypteringen. Desto längre och mer komplicerad nyckel som krävs för att läsa

informationen desto svårare är det att för s.k. crackers att knäcka krypteringen och därmed komma åt informationen.

På dagens marknad finns många olika krypteringsprogram. Dock finns det faktorer som verkat hämmande på den fria konkurrensen. I USA klassas krypteringsprogram som vapen[xxxvii] vilket innebär att de måste ha exporttillstånd för att få föras ur landet. Hittills har det varit mycket svårt att få ett sådant tillstånd. USA har nu börjat att ge tillstånd[xxxviii] för export av krypteringsprogram som använder sig av 56-bitars nyckel.

Det kanske mest kända krypteringsprogrammet på senare tid är PGP, Pretty Good Privacy. Det skapades av en amerikan vid namn Phil Zimmermann. Hans avsikt med programmet var att alla skulle få tillgång till bra kryptering. Då de algoritmer som används bara var patenterade i USA valde Zimmermann att lansera programmet via Internet och på så sätt kringgå kraven på exporttillstånd.. PGP vidareutvecklas numer, i huvudsak, av andra utanför USA. Detta pga. de lagar som gäller där men också pga. att Zimmermann fått utstå mycket ”myndighetsutövning”.

PGP bygger på en teknik som kallas RSA och har ett öppet nyckelsystem. RSA, efter uppfinnarna Rivest, Shamir och Adleman, är en så kallad asymmetrisk algoritm vilket innebär att den är lätt att räkna i en riktning men svårare att räkna baklänges. Det är exempelvis lätt att räkna fram stora tal som är produkten av två primtal men det är mycket svårare att genom produkten få fram vilka de två talen är. Öppna nycklar innebär att sändaren och mottagaren har två nycklar var, en dold och en öppen. Krypteringen sker på följande sätt: Om A skall skicka ett meddelande till B, tar han reda på B:s öppna nyckel. Han krypterar med sin egen hemliga nyckel och B:s öppna nyckel. B får meddelandet och tar reda på A:s öppna nyckel. Algoritmen är utformad så att B nu kan dekryptera meddelandet med sin egen hemliga nyckel och A:s öppna nyckel.[xxxix]

Inloggning
Ett av de mest kritiska avsnitten inom datasäkerhet är lösenordsförfarandet. Kommer lösenorden på drift är mycket av det övriga säkerhetsarbetet till ingen eller liten nytta. Rekommendationer om lösenordshantering finns det många av. Lösenorden anses som extra utsatta vid fem tillfällen:[xl]

- *Vid skapandet av nytt lösenord.* Lösenordet bör skapas maskinellt för att undvika att ett allt för lätt väljes.

- *Byte av lösenord.* Lösenorden bör inte användas över en längre tid, <1 år. Användaren skall kunna byta lösenordet själv.

- *Lagringen av lösenord på servern.* När lösenorden lagras måste de vara krypterade. En lösenordsfil i klartext inbjuder till intrång.

- *Komma ihåg lösenordet (gäller användaren).* Lösenord får inte vara så svåra att användaren måste skriva ner dem för att komma ihåg.

- *Inloggning i systemet.* Lösenordet måste skickas i ett krypterat format vid påloggning. Det är väldigt lätt att avlyssna trafik på TCP/IP-nätet. Genom att använda så kallade ”packet sniffers”[xli] kan lösenord registreras när de skickas i nätverket.

Grundläggande säkerhetsrekommendationer

Forskaren John D Howard[xlii] vid Carnegie Mellom-universitetet ger följande rekommendationer för att upprätthålla en bra intern och extern säkerhetsnivå:

* Gör backup av viktiga filer
* Använd bra lösenord
* Använd rätt filaccessprivilegier
* Känsliga filer skall krypteras eller lagras off-line
* Skicka inte okrypterade användar-ID
* Kryptera E-post
* Genomför riskanalys
* Skydda lösenordsfiler
* Leverera programvaror med aktiverade säkerhetsfunktioner

SKF[xliii] arbetar för att höja sin säkerhetsnivå. Det utförs genom att utbilda informationssäkerhetschefer. Detta sker i en flerstegsprocess där utbildningsstegen är:

* Riskanalys
* Dataåterställning (Business recovery)
* Strategier för att höja säkerhetsmedvetandet

Chefernas uppgift blir sedan att:

* Upprätta lokal säkerhetspolicy
* Beskriva lokala åtgärder
* Fungera som rådgivare till dotterbolagens VD
* Utse ägare till datasystemen
* Upprätta en hotanalys

Värdet av information är svårt att värdera men ofta så rör det sig om enorma summor. För exempelvis SKF rör sig värdet om flera miljarder kronor. Skulle viss information hamna fel kan det till och med kunna innebära konkurs. Det är då enkelt att förstå att säkerhetsarbetet får kosta ett antal[xliv] miljoner kronor varje år. Totalt omsätter industrin inom datasäkerhet över 10 miljarder kronor och då är företagens arbetskostnader oräknade.

Alternativ

Den maximala datasäkerheten kan man bara nå i ett stängt rum utan kommunikation med omvärlden.[xlv] Det är orealistiskt eftersom företag i sin verksamhet måste ha kontaktytor med omvärlden.

Åsblom[xlvi] menar att: ”Man kan ha ett öppet, lättanvänt nätverk. Man kan ha ett säkert nätverk. Men man kan inte ha bägge delarna.”

Brandväggar[xlvii]

Brandväggens uppgifter

Brandväggens huvuduppgift är att skydda information på ett lokalt nätverk som är kopplat mot ett externt nätverk(läs Internet). All kommunikation mellan de båda nätverken kontrolleras av brandväggen. Endast trafik som godkänts får passera brandväggen. Fysiskt består brandväggen av flera komponenter. Dels router och dels en dator. Vanligtvis loggas den trafik som flödar mellan de båda nätverken.

Hot mot det lokala nätverket

Vid anslutning av sitt lokala nätverk till ett externt finns alltid en risk att användare av det externa nätet lockas att göra intrång. En vanlig typ av angrepp, riktade mot lokala nätverk, utförs av Internets ”vandaler”. Dessa gör intrång endast i syfte att visa att de kan klara av det eller för att sabotera delar av det lokala nätverket. Rena felaktigheter i programvaror kan erbjuda hackers en väg in i nätverket. Kunskapen om systembuggar, som lämnar oavsiktliga hål efter sig, sprids snabbt.[xlviii] Med enkla scannerprogram söker så potentiella crackers igenom nätet efter just den typen av sårbara operativsystem. Ett exempel på detta är att i Backoffice server 4.0 ligger en fil som innehåller administrationslösenordet i klartext åtkomlig för vem som helst.[xlix] Lösenordet skrivs in under installationen och ligger i filen för att användas av installationsprogrammet. Problemet är att den inte avlägsnas då installationen är avslutad.

Att göra intrång med hjälp av lösenord är en metod. Om systemet är oskyddat kan lösenordsfilen laddas ner och avkrypteras. Ett problem här är att många användare har alltför enkla lösenord. Därefter kan angreppet fortsätta och angriparen kan röra sig som en vanlig användare i systemet.

Vad kan en brandvägg inte skydda emot ?

Nätet är inte det enda stället där information kan läcka ut. Information lagras och fraktas på många olika sätt. Ett vanligt sätt är disketter och cd-rom skivor. Informationen på dessa medier måste skyddas exempelvis genom kryptering för att vara säker om de av någon anledning skulle hamna i fel händer. Någon som vill bryta sig in på ditt nätverk har många olika möjligheter. Det allra lättaste, och säkraste, sättet att är få tag i någon anställds användar-id och lösenord för då går det att röra sig fritt i nätverket då brandväggen inte skyddar mot interna förehavanden. Sker ”attacken” utifrån är den något lättare att upptäcka. Kommer det många ovanliga anrop från samma ställe kan dessa undersökas för att se om anropen trots allt är riktiga och tillåtna.[l] För att en brandvägg ska fungera, måste den vara en del av ett välutvecklat säkerhetstänkande (arkitektur). En brandvägg ger inte något större skydd mot virus av olika slag.

OSI-modellen[li]

Open Systems Interconnection(OSI)-modellen är avsedd att tillämpas vid datorkommunikation i öppna system. Det finns sju olika skikt(lager) i modellen där de tre översta definierar behandlingsorienterade funktioner och de fyra undre definierar kommunikationsrelaterade funktioner. De benämns högnivå respektive lågnivåskikten.

Högnivåskikten:
- Applikations Lager: Lagret handhar datautbytet mellan användare och applikation.

- Presentations Lager: Detta lager ansvarar för att utväxlad data tolkas och översätts till den representation som tillämpningarna förstår.

- Sessions Lager: Innehåller funktioner för att upprätta samband mellan två tillämpningsprocesser.

Lågnivåskikten:
- Transport Lager: Här finns funktioner för att styra dataöverföringen från ändpunkt till ändpunkt.

- Nätverks Lager: Lagret innehåller funktioner för att överföra block av data från ett system till ett annat.

- Data Länk Lager: Lagret ansvarar för att överföra data mellan två när liggande noder.

- Fysiskt Lager: Detta lager innehåller funktioner för styrning av det fysiska mediet.

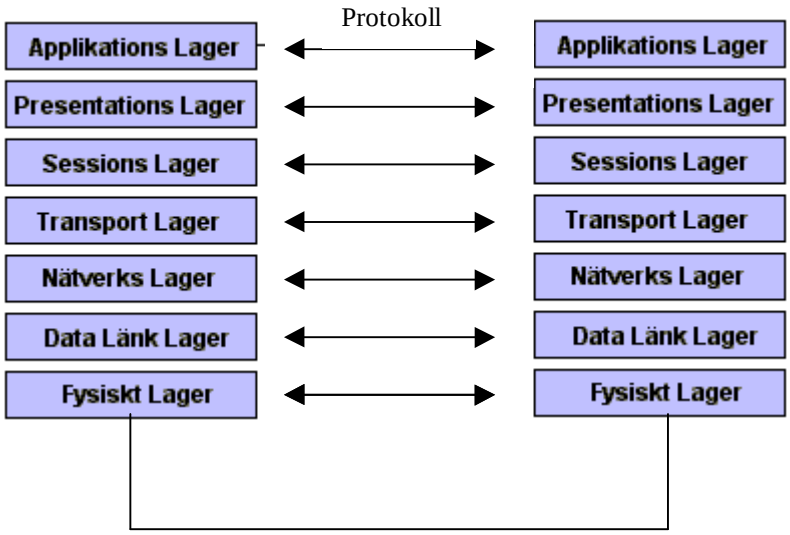


Fig.7. OSI-modellen

Application Gateway

Med Application Gateway behandlas data på applikationsnivå. Sjunde lagret i OSI-modellen.

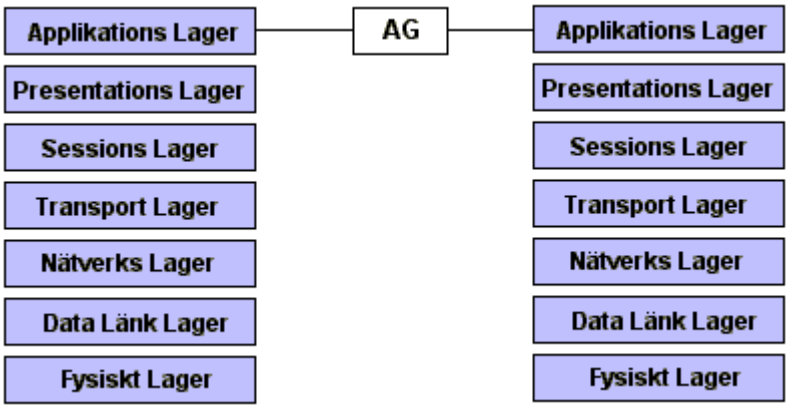


Fig. 8. OSI-modellen med markerat lager där Application gateways arbetar.

Application gateways kallas ibland för proxyservers. En proxyserver är en applikation som körs på den dator som agerar som brandvägg. All trafik mellan Internet och det lokala nätverket måste gå genom denna dator för att proxyservern skall fungera. När till exempel en användare på det lokala nätverket vill köra Telnet mot en dator på Internet, körs Telnet programmet egentligen emot proxyservern. Användaren märker ingen skillnad. Proxyservern kontrollerar varje kommando som skickas så att det inte bryter emot, internt på företaget, fastställda säkerhetspolicys.

På liknande sätt fungerar det åt andra hållet. En dator utanför det interna nätverket som vill kommunicera med en maskin på det interna nätverket anropar som vanligt exempelvis genom FTP. Datorn utanför kommunicerar sedan inte med den maskin den anropat utan med proxyservern. Kommunikationen mellan proxyservern och den maskin som anropats (en så kallad proxyklient, ligger på datorerna i det lokala nätverket) sker med en specialversion av det kommunikationsprogram som används. Proxyklienten pratar med proxyservern istället för maskinen ute på Internet. Proxyservern kontrollerar att de kommandon som begärs är tillåtna och om så är fallet ser den till att de blir utförda.

En proxyserver är en mjukvarulösning. En FTP-proxy kan kanske tillåta export av filer från det lokala nätverket, men inte import av filer från Internet. Vissa mera avancerade proxyservrar kan tillåta FTP från vissa datorer på Internet men inte andra.

En av nackdelarna med proxyservrar är att de kan anses vara oflexibla då de kräver specialversioner av alla program som används för att kommunicera med Internet. Om ett företag har ett eget Internetprogram kan det inte användas av en proxyserver. För att lösa detta problem finns något som heter SOCKS. Det är ett toolkit som kan konvertera

existerande applikationer till proxyversioner. I och med brandväggarnas intåg så görs de flesta nya Internetapplikationer med proxyfunktioner redan från början.

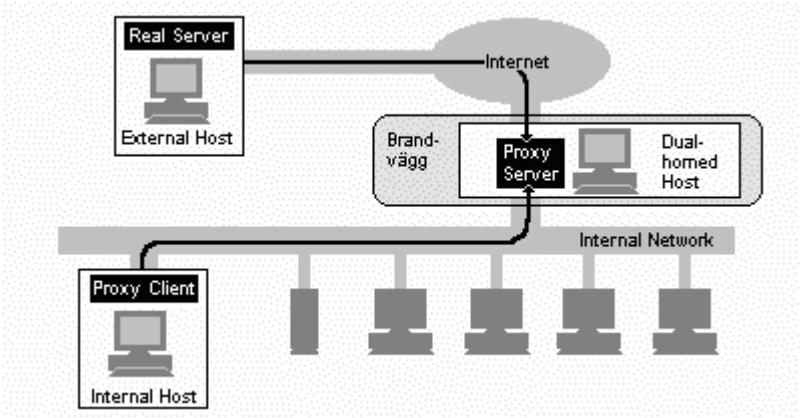


Fig. 9. En proxyserver mellan Internet och det lokala nätverket.

Packet-filtering

Med packet-filtering analyseras all nätverkstrafik genom innehållet i IP-paketen. Data behandlas på transportlagret, fjärde lagret i OSI-modellen (se bild nedan). Varje paket undersöks för att se om det matchar någon restriktion som definierats. Om det matchar någon av restriktionerna, stoppas paketet. I annat fall släpps paketet igenom. Packet-filtering tillåter direkt koppling mellan datorer på det lokala nätverket och datorer på andra nätverk, exempelvis Internet. Med packet-filtering är det som inte har definierats som förbjudet, tillåtet. Detta därför att de flesta packet-filtering brandväggar implementeras på routers. En routers uppgift är att hålla kommunikationen mellan nätverk så genomskinlig som möjligt. Den kan liknas vid en trafikdirigerare. En användare på en dator A i ett nätverk som är uppkopplad mot en dator B i ett annat nätverk, genom en eller flera routers ska känna som om datorerna finns i samma nät. Filtering brandväggar är i regel snabba.

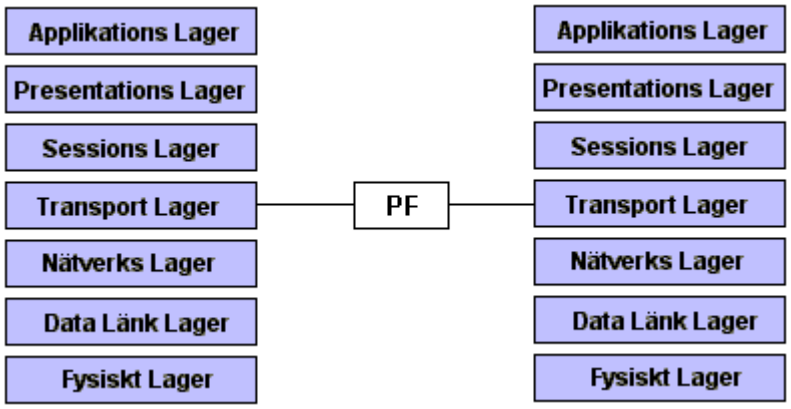


Fig. 10. OSI-modellen med markerat lager där Packet-filtering brandväggar arbetar.

Routern som används i en packet-filtering brandvägg kallas screening router. Till skillnad från en vanlig router är denna typen något mer intelligent. En vanlig router tittar endast på destinationsadressen för varje paket. Två möjligheter finns, routern vet var den ska sända paketet, och gör så, eller så vet routern inte var paket ska sändas och returnerar paket via ett destination unreachable meddelande. En screening router tittar lite mer noggrant på paketet och bestämmer inte bara om den kan skicka paket till sin destination utan också om den ska skicka iväg paketet. Vilket beslut som tas bestäms av hur routern har blivit konfigurerad. Det är möjligt att ha en enda screening router mellan det lokala nätverket och Internet (se bild nedan). Emellertid placerar detta en enorm börda på routern. Den ensamma routern har inte endast att utföra alla routingbeslut, den är också den enda säkerhetspunkten. Skulle routern inte klara av att stå emot en cracker, exponeras hela nätverket.

En screening router kan förbjuda en tjänst, men den kan inte skydda från individuella operationer inom denna tjänst. Om en eftertraktad tjänst har osäkra funktioner kan inte packet-filtering ensamt skydda mot detta.

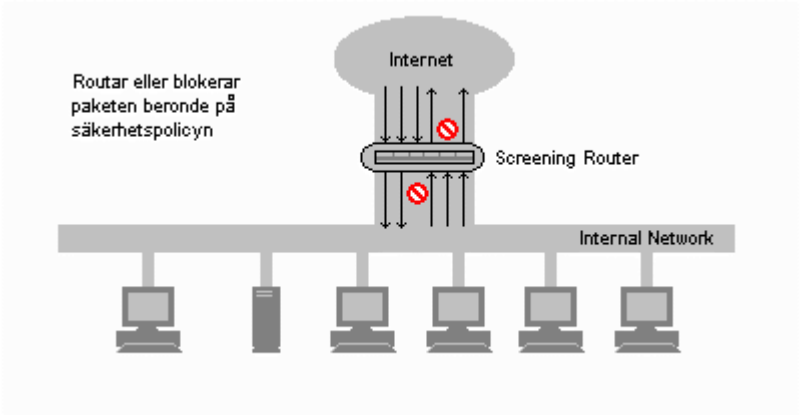


Fig. 11. Screening Router mellan Internet och det lokala nätverket.

Dual-home host arkitektur

En dual-home dator är en dator som är kopplad till två nätverk samtidigt. Emellertid etableras ingen direkt förbindelse mellan dessa nätverk. Denna dator skulle kunna fungera som en router. Vid implementationen av dual-home stängs den ordinarie routern av. Datorer som är inne på det lokala nätverket kan kommunicera med dual-home datorn och datorer på Internet kan kommunicera med dual-home datorn, men dessa system kan alltså inte kommunicera direkt med varandra. Genom att inte tillåta att några paket skickas mellan det lokala nätverket och Internet är det lätt att upptäcka när någon försöker skicka paket direkt till det andra nätet. Sker detta bör en extra kontroll göras av paketet för att se om det innehåller hemlig information. Dual-home ger vid sådan konfiguration en mycket hög säkerhetsnivå. Tillåts paket att gå mellan de olika nätverken kan dual-home neka att skicka paket som utger sig för att innehålla viss typ av

information men i själva verket innehåller något annat. Sådan kontroll är mycket svår att utföra om ett packet-filtering system används.

En dual-home dator kan erbjuda tjänster endast genom att använda en proxy eller genom att låta användarna logga in direkt på dual-home datorn. Att låta användare logga in direkt är i sig en säkerhetsrisk eftersom användar-ID och lösenord ofta inte är säkert. En proxy-server är troligen är mer smidig att använda, men är kanske inte tillgänglig för alla tjänster(läs mer på Application Gateway) som är av intresse för verksamheten.

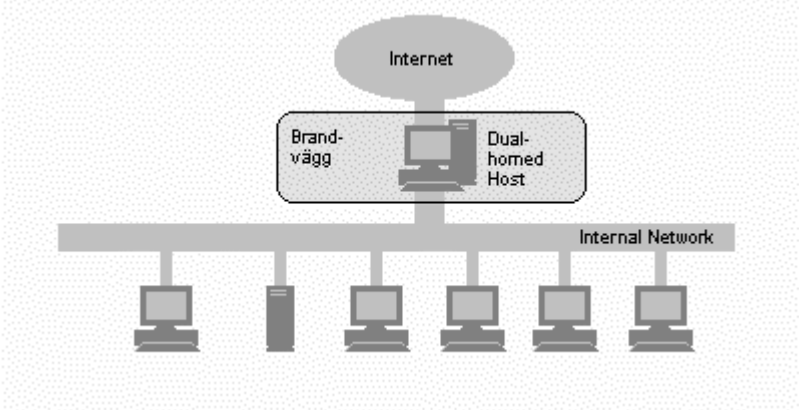


Fig. 12. En dual-home host mellan Internet och det lokal nätverket.

Screened router med Bastiondator

Till skillnad från en dual-home dator som är ihopkopplad på två nät är en bastiondator endast inkopplad på det lokala nätverket. Ut mot Internet finns sedan en screening router. Tanken är att all kommunikation går via bastiondatorn. Screening routern är konstruerad så att det bara är bastiondatorn som kan kommunicera med nätet.

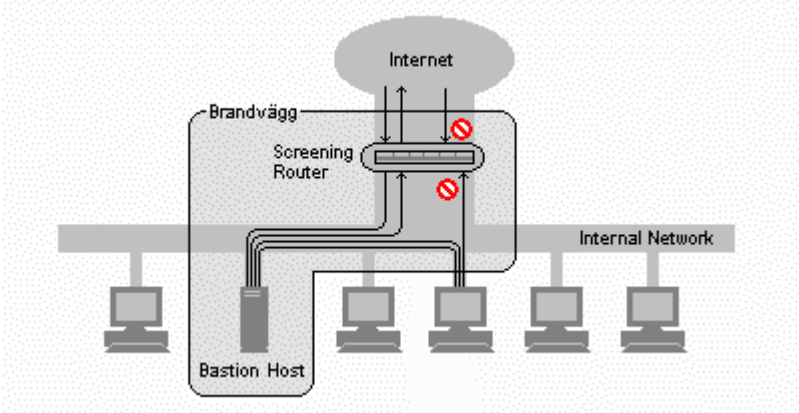


Fig. 13. En screening router i samverkan med en bastiondator

En bastiondator är den huvudsakliga kommunikationspunkten med Internet. Packet filtering funktionen hos screened routern är gjort så att det är endast hos bastiondatorn

som Internet kan kommunicera. Dessutom är det endast viss form av kommunikation som är tillåten. Bastiondatorn måste därför ha en hög säkerhets nivå. Genom att göra så här kan både proxy-server och packet-filtering arkitekturen kombineras att kunna:

Tillåta andra datorer, på det lokala nätverket, att komma åt Internet. Direkt kommunikation med Internet är dock förbjuden vilket innebär att de lokala klienterna måste gå via proxy-servern på bastiondatorn.

Då denna arkitektur tillåter att paket går ifrån Internet till det lokala nätverket, kan den ses som mer farlig än dual-home arkitektur, vilken är designad att se till att inga paket når det lokala nätverket (se tidigare avsnitt). Dock gäller att en dual-home inte heller är säker då paket kan ”slinka” igenom. Genom att skydda det lokala nätverket med både en router och en dator (bastiondatorn) erhålls ändå ett relativt bra skydd.

Screened subnet arkitektur

Genom att lägga ett extra nätverk mellan det externa nätet och det lokala nätet ökas säkerheten. Detta mellannätverk kallas även Perimeter nätverk. Bastiondatorn är det enda stället där ett intrång kan ske och genom att denna placeras på Perimeter nätverket så kommer en eventuell hacker endast in på det mindre betydelsefulla Perimeter nätverket. All trafik från och till bastionen kan emellertid avlyssnas.

I screened subnet arkitektur används två screening routers vilka är kopplade till Perimeter nätverket. Detta innebär att en angripare måste ta sig genom tre olika säkerhetspunkter.

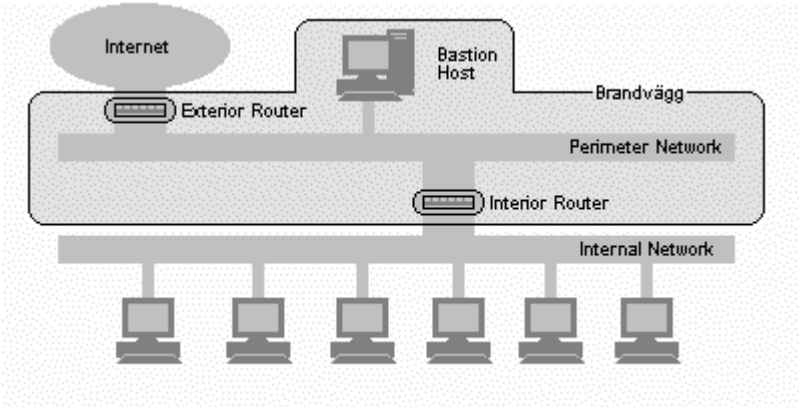


Fig. 14. Ett perimeter nätverk uppsatt mellan det inre nätverket och Internet.

Analys

Analysdelen av uppsatsen innebär en tillämpning av de olika avsnitt vi gått igenom i teoridelen. Detta innebär att vi anger hur vi tillämpat modellering, brandväggar och säkerhet i vårt system.

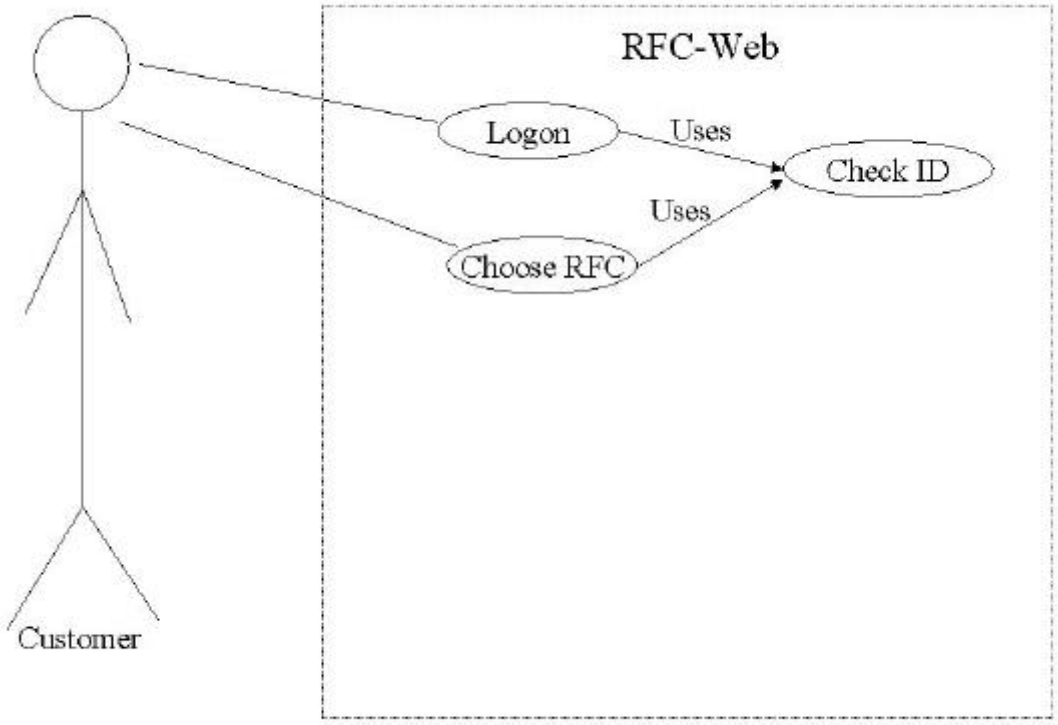
Modellering

Vi har använt oss av Use-Case diagram och klassdiagram från UML notationen samt en egen notation för att översiktligt beskriva relationerna mellan systemets delar.

Use-Case diagram RFCweb

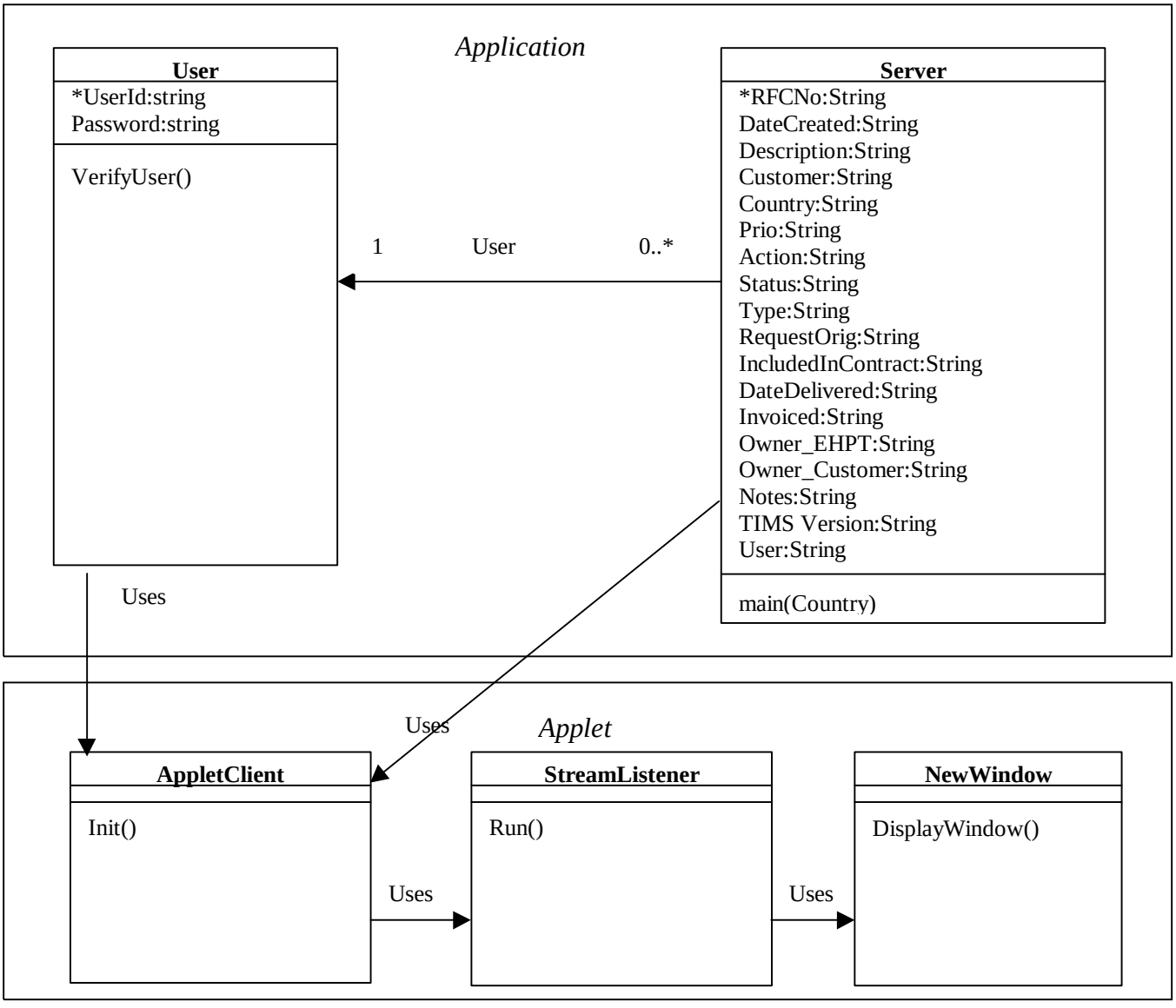
Vårt användarscenario är förhållandevis enkelt. Användarens aktiviteter i systemet är endast att logga in med sitt användarnamn och lösenord samt att välja vilken RFC hon vill få information om. Enligt UML notationen anger vi att systemets två användarscenarios använder sig av ett tredje som är en idkontroll.

Fig. 15.



Klassdiagram RFCWeb

För att få fram lämpliga kandidatklasser har vi använt oss av vår Use-case modell.
Denna metod rekommenderas av Brown liksom brainstorming, användarintervjuer och Delphi metoden.[lii] Vår uppgift är ju dock begränsad varför vi inte anser det nödvändigt att använda fler metoder än use case modellen. Denna har ju i sig kommit till under en sorts brainstorming. Tilläggas skall att vi inte använder oss av en strikt UML notation. Relationen mellan Application och Applet anges med streckade pilar eftersom vi anser att applet är en typ av gränssnitt. UML notationen för kommunikation mellan vanliga klasser och gränssnittsklasser anges med streckade pilar.[liii] Kandidatklasser är User och Server i applikationen och AppletClient, StreamListener samt NewWindow i appleten.. Vi har en gränssnittsklass, GUI vilken i systemet kommer att representeras av en applet. Övriga klasser hanteras av en applikation som körs på servern. I diagrammet nedan beskrivs de olika klasserna som ingår i vårt system och deras relationer. Identifierare är markerade med en asterisk (*).
Fig. 16.



Systembeskrivning

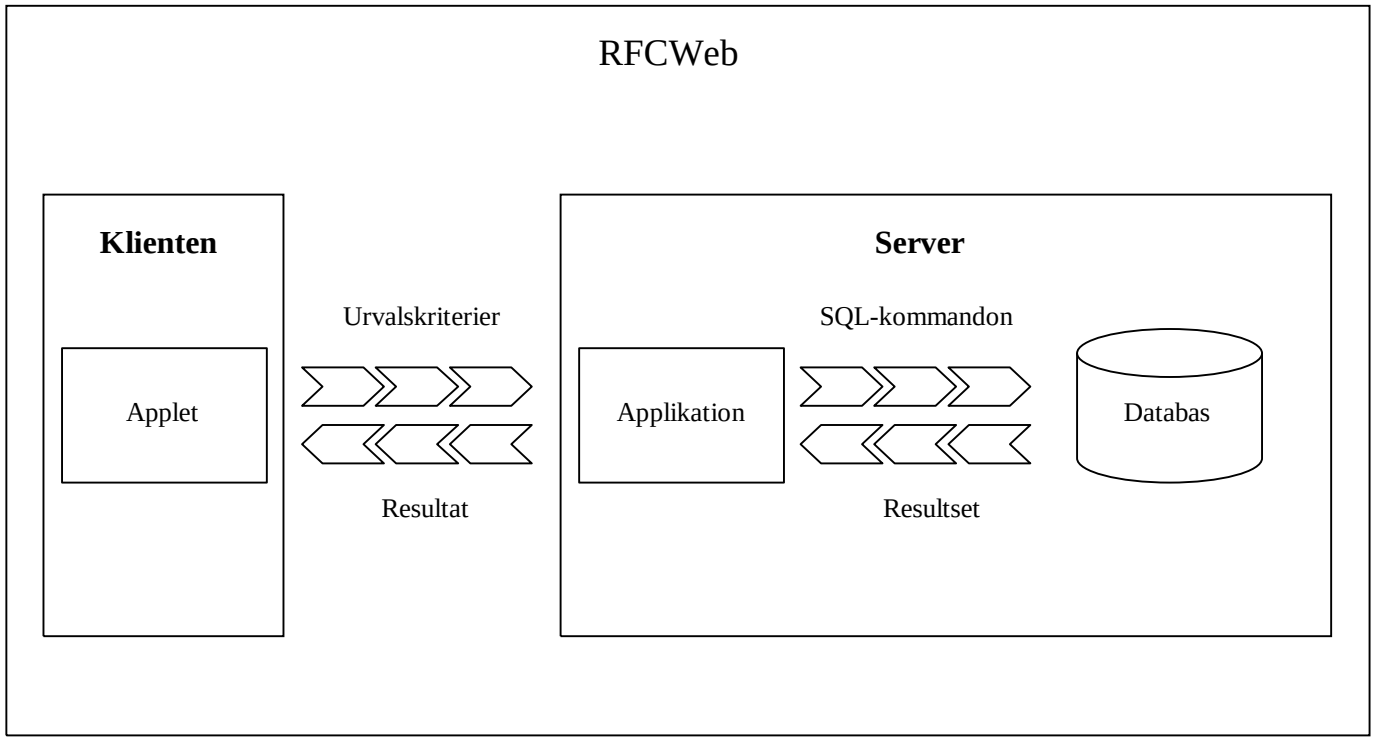
Systemet kan delas upp i två delar, en klientdel och en serverdel. Klientdelen motsvaras av en java-applet och körs lokalt hos klienten. Serverdelen består av en java-applikation som ligger på webbservern och är ständigt igång. Systemet aktiveras när en klient (kund) klickar på en länk på EHPT:s hemsida. Appleten aktiveras och laddas ner på kundens dator. Kunden anger, genom att fylla i formuläret på appleten, vilka urvalkriterier som skall gälla och skickar dem till applikationen på servern. Applikationen översätter de önskade urvalkriterierna till SQL-kommandon som den använder mot en databas för att hämta upp efterfrågad data, ett så kallat resultset skapas. Informationen skickas slutligen tillbaka till appleten där den presenteras för användaren.

Kommunikationen mellan systemets applet och applikationen sker via en port med hjälp av så kallade sockets. En port är ett tänkt fysisk ställe på datorn där kommunikation mellan en server och en klient kan ske.[liv] En socket är en abstraktion av nätverksmjukvaran som möjliggör kommunikation till och från ett program.[lv] Sockets i java skapas relativt enkelt om det finns tillgång till en giltig Internetadress och ett giltigt portnummer. När en klient skrivit det som skall sändas till applikationen trycker han på en skickaknapp. Informationen i fälten skickas ut på den överenskomna porten. Serverapplikationen ligger hela tiden och ”lyssnar” om det kommer något på detta portnummer. Klienten i sin tur lyssnar sedermera på den port där servern returnerar resultatet. Se fig 17. För mer detaljerad information se bilaga 1 –Programkod.

Uppkopplingen sker vid ”skicka”- kommandot och nerkopplingen av förbindelsen sker efter mottagandet av informationen. I och med att vi låter klient och server behålla kontakten med varandra genom hela processen, från skickandet av information till mottagandet av den samma, blir det inga problem med att klienter skulle kunna få fel information. Dock innebär detta en begränsning i att bara en klient åt gången kan göra förfrågningar. Då det går fort att få sin information torde inte detta innebära några problem med för långa väntetider för att kunna göra förfrågningar. Väntetiden blir delar av en sekund.

Javas plattformsoberoende gör att systemet kan tillhandahålla sina tjänster oavsett vilken typ av maskin som kunden sitter vid när han är intresserad av att göra förfrågningar. Systemet har, genom sin enkla uppbyggnad, ett par begränsningar. Det finns bara möjlighet att erhålla information men inte att manipulera och uppdatera den. Detta innebär att risken att kunderna förstör eller förvanskar informationen i databasen elimineras. Databasen uppdateras, via ett gränssnitt i Access, av anställda på EHPT för att bibehålla kontrollen över den data som förs in i basen.

Fig. 17.



Säkerhet

Vårt system innebär två olika säkerhetsrisker. Den första innebär att vi måste hindra obehöriga från att komma in i systemet. Den andra att obehöriga kan lyssna av trafiken mellan vår applet och server. I det senare fallet anses inte informationen ha sådant värde att det är nödvändigt att använda krypterad överföring. Däremot är det ett minimum att vi skapar ett lösenordsförfarande som hindrar obehöriga från att gå in i systemet och ladda hem information. Lösenordsförfarandet kommer att bygga på kryptering av användarnamn och lösenord. Tyvärr kan vi inte gå djupare i detta avsnitt då företagets säkerhetsregler inte tillåter att vi publicerar mer detaljerad information.

Brandväggar

För vårt system innebär närvaron av brandväggen ett par saker som vi måste ta hänsyn till. Kunderna (klienten) tillåts inte att komma åt information som ligger på EHPT:s nätverk. Detta innebär att den databas, som används av kunderna, måste göras åtkomlig. Med hjälp av ett script, som körs varje natt, läggs delar av databasen på en server som är tillgänglig för användarna. I vårt fall innebär detta att delar av databasen kopieras från en plats i det lokala nätverket till en webbserver som står utanför brandväggen. Uppdateringen av databasen sker således en gång varje dygn. Att informationen inte visas

i realtid kan i stort sett negligeras då RFC-processen som sådan inte är tillräckligt snabb för att kräva ständig uppdatering.

Ett sätt att göra informationen mer “upp to date” skulle vara att köra scriptet fler gånger per dag. Dock kan andra problem uppstå, exempelvis att informationen inte är tillgänglig under tiden som databasen uppdateras. Genom att köra scriptet när varken kunder eller administratörer arbetar med någon av databaserna (den på det lokala nätverket och den på webbservern) minimeras riskerna att något går fel. Det är väsentligt att informationen som kunderna får är korrekt. Uppstår tveksamheter om det som presenteras verkligen stämmer (är uppdaterat osv.) och kunden bestämmer sig för att ringa till EHPT, tappar systemet helt sin funktion.

EHPT har idag ingen egen brandvägg. De använder sig av Ericsson koncernens vägg och uppkoppling mot Internet. Detta kommer att ändras inom en snar framtid där EHPT får en egen brandvägg och uppkoppling mot Internet. Nu finns det ingen möjlighet, som vi tidigare nämnt, att komma åt information som finns på insidan väggen. Som planerna ser ut nu så skissas det på en lösning där en SQL-server placeras så att den går att komma åt både från utsidan och insidan. Detta kommer i hög grad att påverka vårt system. Om det blir möjligt att använda SQL-servern för RFC-infomationen så kommer systemet att kunna köras mot denna och då behövs inte vårt script längre. När hela RFC-informationen läggs på samma ställe som vår applikation kör kommer systemet att bli lättare att bygga ut och om efter framtida önskemål. I och med att dessa förändringar är förestående så kommer vårt system troligen inte att tas i drift innan de är genomförda.

Vad Ericsson har för typ av brandvägg idag har vi inte kunnat få reda på. Den anses dock inte ge ett fullgott skydd utan beskrivs som ett filter med lite för stora hål. Den har inte tillräcklig flexibilitet och kan inte tillhandahålla den service som krävs. I framtiden kommer kunderna, i större utsträckning än idag, att kunna ta del av den information som gäller dem. Säkerhetsaspekten kommer då att bli än mer betydelsefull. Genom att få en egen brandvägg kan EHPT bättre kontrollera vad som skall vara tillgängligt och vad i den egna verksamheten som behöver extra skydd.

Att EHPT får en egen server som kan verka mot nätet gör att nya möjligheter öppnas. Det finns då möjlighet att pröva nya lösningar som både är snabbare och enklare än vårt system. En utveckling helt i linje med utvecklingen av mjukvara.

Avslutande diskussion

Modellering

Bakgrund och problem
Då vi inledningsvis övervägde vilken modelleringsmodell som kunde komma ifråga för vår studie fokuserade vi snabbt på UML. Detta eftersom vi redan innan arbetet inleddes visste att detta är en standard som branschen i stor utsträckning använder. Vi ville lära oss metoden grundligt. Som källor använde vi dels en bok skriven av Fowler, dels själva standarddokumentationen för UML utgiven av OMG (Object Management Group). Den senare är en välstrukturerad men svårläst beskrivning av UML. Den modell som vi efterhand utarbetade var till god hjälp vid konstruktionen av systemet. Vi hade väntat oss att den modell vi konstruerade skulle kunna ligga till grund för utvecklingen utan större förändringar. Det ska dock tilläggas att den fortlöpande uppdaterats för att samtidigt beskriva det färdiga systemet. Vi anser det naturligt att man vid utvecklingen av ett system blir nödgad att rita om sin modell efterhand som utvecklarna kommer på nya sätt att lösa problem. Omfattningen på förändringen står med all säkerhet i proportion till erfarenheten hos de som konstruerar modellen.

Resultat
Vi hyser ingen tvekan om modellens fördelar som kommunikationslänk mellan användare och programmerare samt mellan programmerare själva. Vår modell kommer att bifogas programmet för att skapa förutsättningar för en vidareutveckling av systemet.

Alternativa angreppssätt
Det skulle vara intressant att ha gjort en jämförelse mellan UML och ett annat modelleringsspråk för att på så vis utröna skillnaderna i hur komplicerade språkens uppbyggnad är. Även olika språks förmåga att förmedla vad systemet skall innehålla skulle vara intressant att utforska.

Säkerhet

Bakgrund och problem
Vid modelleringen av systemet tog vi hänsyn till dess säkerhet beträffande inloggningsrutiner. Vi ritade in ett lösenordsförfarande. Hur detta skulle implementeras hade vi dock inte klart för oss. Efter studier av olika typer av säkerhetsrutiner kom vi fram till den lösning som kommer att konstrueras för EHPT. Vid framtagandet av en lösning upptäckte vi vikten av att lägga kryptering och algoritm på en nivå som är lämplig för vårt projekt. Vi fick ta hänsyn till vår tidigare kunskap på området och projektets tidsramar. Detta kommer slutligen att innebära att vi lämnar en rekommendation för hur lösenförfarandet kan se ut.

Resultat
Vi kan som tidigare nämnts inte ange den exakta utformningen av vårt säkerhetsförfarande av hänsyn till EHPT:s säkerhetsrutiner. Vi har dock beslutat att inte

kryptera själva överföringen av information från serverdelen av systemet till klienten. Med tanke på innehållet i de överföringar som görs krävs inte detta. Vi har nöjt oss med ett lösenordsförfarande.

Alternativa angreppssätt
Då det gäller säkerhetslösningar finns det en uppsjö av information att gå igenom. Lämpliga områden att studera är standardlösningar som kan implementeras i nya system. Användbart skulle vara att kunna utnyttja NT:s lösenordsförfarande i andra applikationer.

Brandväggar

Bakgrund och problem
Brandväggar och dess funktion behövde tidigt kartläggas för att vårt system skulle kunna användas så som det var tänkt. I och med att kunder, i nuläget, inte kan komma åt information på EHPT:s lokala nätverk behövdes en lösning på detta problem. Att komma runt brandväggar är en utmaning för systemutvecklare och en mardröm för säkerhetsansvariga. Två motstridiga intressen möts och behöver, närmast fordrar, en bra lösning. Vi konstaterar att brandväggarna blir allt bättre, dvs. mer osynliga för användaren.

Resultat
Vår lösning att med ett script exportera de delar av databasen som kunderna skall kunna komma åt kan kännas något långsam och omständlig. Att behöva ha delar av informationen på två ställen är inte att utnyttja ett ofta begränsat utrymme på ett bra sätt. Scriptet erbjuder emellertid en okomplicerad lösning som gör jobbet, dock kanske inte på det effektivaste sätt. Vi kan i dag inte se någon annan lösning på detta problem som passar inom ramen för vårt projekt.

Alternativa angreppssätt
När EHPT får sin egna brandvägg och SQL-server bjuds det nya alternativa lösningar. Med möjligheten att lägga all information direkt på en server som har direkt åtkomst via nätet underlättas framtida ändringar i systemet då all information redan finns på plats. Problemet med uppdateringar och att scriptet körts som det skall kan då läggas till handlingarna och fokus kan vridas mot nya angelägna områden.

Avslutning

För att summera projektet så kan vi konstatera att vi planerat och genomfört en lösning på EHPT:s uppdrag. I efterhand tror vi oss kunna konstatera att Java inte är det bästa sättet att lösa en sådan här uppgift. Vi anser att ASP(Active Server Pages) på ett enklare sätt kan lösa de uppgifter uppdraget innebar. Framför allt skulle ASP som scriptspråk kräva mindre kapacitet av serverdatorn då det endast aktiveras när någon startar systemet. Framtida utökning och annat underhåll skulle ta mindre resurser i anspråk då ASP är betydligt lättare att använda. Förmodligen skulle en ASP-lösning göra systemet något snabbare då ett led i kommunikationen skulle kunna tas bort, den mellan appleten och

applikationen. Det finns dessutom en tendens till motvilja att installera applikationer som ligger och kör kontinuerligt på webbserven.

Bilaga 1. Koden

Applet

```
//Title:       AppletClient
//Version:
//Copyright:   Copyright (c) 1999
//Author:      Magnus Wahlqvist & Jonas Thorsell
//Company:     EHPT
//Description:
/*Programmet skapar en socket för att kommunicera med en server, där koden till
appleten ligger. Kommunikationen sker på port 6789. Det skapas också gränssnitt
för in och utmatning av data.
*/

//package AppletClient;

import java.awt.event.*;
import java.applet.*;
import java.awt.*;
import java.io.*;
import java.net.*;
import java.util.*;

public class AppletClient extends Applet
{
/**********************Deklarationer*************************/
    public static final int PORT = 6789;
    Socket s;
    DataInputStream in;
    PrintStream out;
    TextField inputfield_country;
    TextField inputfield_priority;
    TextField inputfield_status;
    TextField inputfield_rfcno;
    //TextArea outputarea2;
    //TextArea outputarea;
    StreamListener listener;
    String sendString;
    static String country;
    static String priority;
    static String status;
    static String RFCNo;
    static String SQL;
    static String SQL_count;
    private Button sendButton;

  // Create a socket to communicate with a server on port 6789 of the
  // host that the applet's code is on.  Create streams to use with
  // the socket.  Then create a TextField for user input and a TextArea
  // for server output.  Finally, create a thread to wait for and
  // display server output.

  //Anger layouten och definierar de ingående komponenterna för välkomst och inmatningsfönstret.
    public void init()
```

```
    {


        inputfield_country = new TextField();
        inputfield_priority =new TextField();
        inputfield_status = new TextField();
        inputfield_rfcno = new TextField();

        //outputarea = new TextArea();
        //outputarea2 = new TextArea();
        sendButton = new Button("Skicka");

        this.setLayout(new BorderLayout());

        Panel great = new Panel();
        great.setLayout(new GridLayout(5,3));

        great.add(new Label("        Country:"));
        great.add(inputfield_country);
        great.add(new Label(""));

        great.add(new Label("        Priority:"));
        great.add(inputfield_priority);
        great.add(new Label(""));

        great.add(new Label("        Status:"));
        great.add(inputfield_status);
        great.add(new Label(""));

        great.add(new Label("        RFCno:"));
        great.add(inputfield_rfcno);
        great.add(new Label(""));

        great.add(new Label(""));
        great.add(sendButton);
        great.add(new Label(""));
        this.add("Center", great);
        //this.add("East", outputarea);
        //this.add("West", outputarea2);
        //this.add("South", sendButton);

        this.resize(280,100);
        //this.setSize (280,100);
        //show();
}

public boolean action(Event e, Object what) {
    if (e.target == sendButton) {

        //***********Skapar en SQL-sträng att skicka till servern**********

        country = null;
        priority = null;
        SQL = null;
        SQL_count = null;
```

```
        priority = inputfield_priority.getText().trim();
        country = inputfield_country.getText().trim();
        SQL = "SELECT RFCNo, Country FROM request WHERE country LIKE "'+country+'"";
        SQL_count = "SELECT COUNT(*) AS antal FROM request WHERE Country LIKE
"'+country+'"";



        if (priority != null) {
           //System.out.println ("###"+priority+"###");

          SQL = SQL + "AND prio LIKE "'+priority+'"";
          SQL_count = SQL_count + "AND prio LIKE "'+priority+'"";
          }


          SQL_count = SQL_count + "¤";
          SQL = SQL + "¤";
          SQL_count = SQL_count + SQL;

        try {

           s = new Socket(this.getCodeBase().getHost(), PORT);        //this.getCodeBase().getHost()

           in = new DataInputStream(s.getInputStream());
           out = new PrintStream(s.getOutputStream());

           listener = new StreamListener(in/*, outputarea, outputarea2*/);


           this.showStatus("Connected to "
              + s.getInetAddress().getHostName()
              + ":" + s.getPort());

         }
        catch (IOException y)
          { this.showStatus(y.toString()); }




        //sendString = inputfield_country.getText();
        out.println(SQL_count);    //sendString
        //inputfield_country.setText("");

        return true;
        }

      return false;
   }


 public AppletClient() {
   try  {
     jbInit();
   }
```

```
    catch (Exception e) {
      e.printStackTrace();
    }
  }

  private void jbInit() throws Exception {
    this.setSize(new Dimension(712, 466));
  }


}

// Väntar på output från servern för att kunna visa den i de specificerade fälten.

class StreamListener extends Thread {
    DataInputStream in;
    TextArea output;
    TextArea output2;
    static String line;
    static String NoOfRows;
    NewWindow bmw;


    public StreamListener(DataInputStream in/*, TextArea output, TextArea output2*/) {
        this.in = in;
        //this.output = output;
        //this.output2 = output2;
        this.start();

    }


    public void run() {

        String Tnext;
        int NoOfRowsInt;
        int NoOfFields;
        String [] values = new String[1000];


//Delar upp strängen i delsträngar för att kunna lägga in dem i rätt fack på det nya fönstret.
        try {
            for(;;) {
                line = in.readLine();
                  StringTokenizer T = new StringTokenizer(line, ",");


                  NoOfRows = T.nextToken();
                  values[0]=NoOfRows;



                NoOfRowsInt = Integer.valueOf (NoOfRows.trim()).intValue();
                NoOfFields = NoOfRowsInt * 2;

                for (int i = 1;i <=NoOfFields;i++) {
```

```
                Tnext = T.nextToken();
                values[i] = (Tnext);
               }//Slut for

               bmw = new NewWindow(values);

            }//Slut for
         }//Slut try
         catch (IOException e) { output.setText(e.toString()); }

         finally { output.setText("Connection closed by server." + '\n'); }
      } // Slut run

}  //Slut klass

class NewWindow

{

  String [] valuearray;
  static String NoOfRowsString;
  static int NoOfRowsInt;


    Frame pop_up;
    Button register, end;



  public NewWindow (String [] valuearray) {
      this.valuearray = valuearray;
      pop_up = new Frame("RFC Info");
      this.displayWindow();

 }// Slut konstruktor


//Tar fram det nya fönstret för resultatet av den gjorda sökningen
  public void displayWindow () {

      NoOfRowsString = valuearray[0];
      NoOfRowsInt = Integer.valueOf (NoOfRowsString.trim()).intValue();

      System.out.println(NoOfRowsInt);



      Panel pa = new Panel();
      ScrollPane p = new ScrollPane ();
      pa.setLayout(new GridLayout(NoOfRowsInt, 2));



      NoOfRowsInt = NoOfRowsInt*2;

      for(int i=1; i<NoOfRowsInt +1; i++)
```

```
        pa.add(new Label (valuearray[i]));
        p.add("Center", pa);
        pop_up.add("Center", p);
        pop_up.pack();
        pop_up.resize(600,400);
        pop_up.show();

    }//Slut displayWindow



 }//Slut klass
```

Serverapplikation

```
//Title:      myread
//Version:
//Copyright:  Copyright (c) 1999
//Author:     Jonas Thorsell & Magnus Wahlqvist
//Company:    EHPT
//Description:
/* En applikation som kör på servern. Den får information från en applet och tar sedan information ur en
databas och skickar tillbaka informationen till appleten*/

//package myread;


import java.awt.*;
import java.net.*;
import java.io.*;
import java.sql.*;
import java.util.*;
import java.lang.*;


public class Server {
//////////////////Deklarationer//////////////////////////////////
    static int antalRaderInt;
    static int NoOfRows;
    static String antalRader;
    static String clientSentence;
    static String replystring;
    static String replystringCountry;
    static String FirstReply;
  boolean packFrame = false;

  //Construct the application

  public Server() {
    }
//Main method

  public static void main(String argv[]) throws Exception  {
```

```
   System.out.println("Test1");

   //Databasdeklarationer tänk på objektorientering senare...

    ServerSocket listenSocket = new ServerSocket(6789);    // ServerSocket


    String url = "jdbc:odbc:testodbc2";


    while(true)
     {
       Socket connectionSocket = listenSocket.accept();
       DataInputStream inFromClient = new DataInputStream(connectionSocket.getInputStream());
       DataOutputStream outToClient = new DataOutputStream(connectionSocket.getOutputStream());
       //clientSentence = "";
       replystring = null;

        clientSentence = inFromClient.readLine();

        if (clientSentence != "") {

          //String numberOfRows = "SELECT COUNT(*) AS antal FROM request WHERE Country LIKE
'"+clientSentence+"' ";
          //String query1 = "SELECT RFCNo, Country FROM request WHERE Country LIKE
'"+clientSentence+"' ";

/* Delar upp strängen som kommer från klienten i de två SQL-delar som används.
Första delen av strängen är ett SQL-kommando för att få ut hur många rader   resultsetet kommer
att innehålla. Den andra delen av strängen, också ett SQL-kommando, skapar resultsetet.*/

          StringTokenizer T = new StringTokenizer(clientSentence, "¤");
          String numberOfRows = T.nextToken();
          String query1 = T.nextToken();

          Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
          Connection con0 = DriverManager.getConnection (url, "", "");
          Connection con = DriverManager.getConnection (url, "", "");

          Statement stmt0 = con.createStatement();
          Statement stmt = con.createStatement();
          System.out.println("Test3,5");
          System.out.println("sssssssssssssssssssssssssssssssssssssssssssssssssssss");
          ResultSet NumberRows = stmt0.executeQuery(numberOfRows);
          ResultSet rs = stmt.executeQuery(query1);
          rs.next();
          //NoOfRows = 0;



          NumberRows.next();
////////Plockar ut antalet rader ur rs NumberRows///////////////
          try {
            antalRader = NumberRows.getString("antal");

            System.out.println(antalRader);
```

```
            }//Slut try
            catch (SQLException gt){}
          try {

            antalRaderInt = Integer.valueOf (antalRader.trim()).intValue();

/////////Bygger sträng att skicka till klienten//////////////////
            // antalRaderInt ++;
            for (int i=0; i < antalRaderInt; i++)
              {
              FirstReply = rs.getString("RFCNo");
              System.out.println("antalRaderInt");
              FirstReply = FirstReply + ",";
              replystringCountry = rs.getString("Country");
              replystringCountry = replystringCountry + ",";
              if (i==0) {
                replystring = FirstReply + replystringCountry;
              }//Slut if
              if (i !=0) {
                replystring =replystring + FirstReply + replystringCountry;
              }//Slut if
              rs.next();
            }//Slut for
            antalRader = antalRader + ",";
            //System.out.println(replystring);
            replystring = antalRader + replystring;
            outToClient.writeBytes(replystring + '\n');
            System.out.println(replystring);

            }//Slut try
          catch (SQLException rt) {
           outToClient.writeBytes("Not an appropriate value, try again." + '\n');
           }


        }//slut if
      }//slut while
  } //slut main
} //slut klass
```

Källförteckning

Alvesson, **M.** *Tolkning och reflektion vetenskapsfilosofi och kvalitativ metod*, Studentlitteratur, 1994.
Andréasson, S-A & Carlsson, C. *Datakommunikation för informatik*, Institutionen för datavetenskap Chalmers tekniska högskola, 1997.
Brown, D. Object-Oriented Analysis John Wiley & Sons, Inc. 1997.
Byttner, K-J. *Fler än Telia har brister*. Computer Sweden, nr 69, 1998.
Carlsson, B. *Grundläggande forskningsmetodik för medicin och beteendevetenskap*, Almqvist & Wiksell, 1984.
Engholm, A. *En sak är säker – kryptering*. Computer Sweden, nr 33, 1998.
Engholm, A. *Total datasäkerhet inget att eftersträva*. Computer Sweden, nr 62, 1997.
Enryd, T & Linberg A. *Datasäkerhet vid lösenordshantering* Examensarbete Inst. För Informatik, Handelshögskolan, Göteborgs Universitet, 1997.
Fowler, M. *UML Destilled*, Addison Wesley 1997.
Holme, I. H. & Solvang B. K. *Forskningsmetodik-om kvalitativa och kvantitativa metoder*, Studentlitteratur, 1991.
IDG News. *Förenade hackare – ny hotbild på nätet*. Computer Sweden, nr 79, 1998.
Johansson, A. *Inga hemligheter kvar*. Computer Sweden, nr 37, 1998.
Karlsén K. *Efterfrågan på pogrammerare kommer att minska* Computer Sweden nr 26, 1999.
Karlsén K. *Nödvändigt med metoder* Computer Sweden nr 26, 1999.
Lobråten, P. *Expert slår larm om dålig IT-säkerhet*. Computer Sweden, nr 21, 1998.
Lotsson A. *UML bör stå på tillväxt* Computer Sweden nr 17, 1998.
Lundahl U. & Skärvad P-H. *Utredningsmetodik för samhällsvetare och ekonomer*, Studentlitteratur, 1992.
Magnusson, P S. *Jag är en kryptoanarkist*. Datateknik, nr 6, 1994.
Morgan, G. & Smircich, L. *The case for qualitative research*, Academy of Management Review, nr 4 1980
Nilsson, Å. *Säkerhetsexperten: ”Många företag slarvar med säkerheten*”. Computer Sweden, nr 7, 1997.
Norén, L. *Fallstudiens trovärdighet* FE-rapport 1990-305. FEK. Göteborg, 1990.
Patel, R.& Davidsson, B. *Forskningsmetodikens grunder*, Studentlitteratur, 1994.
Rosengren K. E &Arvidson P. *Sociologisk metodik* Almqvist & Wiksell, 1992.
Taget från lösenordsuppsatsen (Department of Defense).
Thurén, T. *Vetenskapsteori för nybörjare* Runa förlag, 1991.
UML Notation Guide, version 1.1, Rational Software 1/9 1997.
van der Brink, R. *Datasäkerhet inte bara på burk*. Computer Sweden, nr 5, 1997.
Åsblom J. *Lösenord läsbart I backoffice* Computer Sweden nr 16, 1999.
Åsblom, J. *Datasäkerhet – bra, om man kan få det*. Computer Sweden, nr 4, 1997.

Fotnoter

[i] Thurén, T. *Vetenskapsteori för nybörjare* Runa förlag, 1991.
[ii] Alvesson, M. *Tolkning och reflektion vetenskapsfilosofi och kvalitativ metod,* Studentlitteratur, 1994.
[iii] Ibid.
[iv] Patel, R.& Davidsson, B. *Forskningsmetodikens grunder*, Studentlitteratur, 1994.
[v] Alvesson, M. *Tolkning och reflektion vetenskapsfilosofi och kvalitativ metod,* Studentlitteratur, 1994.
[vi] Morgan, G. & Smircich, L. *The case for qualitative research*, Academy of Management Review, nr 4 1980
[vii] Alvesson, M. *Tolkning och reflektion vetenskapsfilosofi och kvalitativ metod,* Studentlitteratur, 1994.
[viii] Holme, I. H. & Solvang B. K. *Forskningsmetodik-om kvalitativa och kvantitativa metoder*, Studentlitteratur, 1991.
[ix] Rosengren K. E &Arvidson P. *Sociologisk metodik* Almqvist & Wiksell, 1992.
[x] Lundahl U. & Skärvad P-H. *Utredningsmetodik för samhällsvetare och ekonomer*, Studentlitteratur, 1992.
[xi] Holme, I. H. & Solvang B. K. *Forskningsmetodik-om kvalitativa och kvantitativa metoder*, Studentlitteratur, 1991.
[xii] Carlsson, B. *Grundläggande forskningsmetodik för medicin och beteendevetenskap,* Almqvist & Wiksell, 1984.
[xiii] Norén, L. *Fallstudiens trovärdighet* FE-rapport 1990-305. FEK. Göteborg, 1990.
[xiv] Holme, I. H. & Solvang B. K. *Forskningsmetodik-om kvalitativa och kvantitativa metoder*, Studentlitteratur, 1991.
[xv] Karlsén Karin *Nödvändigt med metoder* Computer Sweden nr 26, 1999
[xvi] Karlsén Karin *Efterfrågan på pogrammerare kommer att minska* Computer Sweden nr 26, 1999
[xvii] Lotsson Anders *UML bör stå på tillväxt* Computer Sweden nr 17, 1998
[xviii] För mer om OMT läs sid 166 I Brown
[xix] Fowler, Martin *UML Destilled,* Addison Wesley 1997.
[xx] Ibid.
[xxi] Ibid.
[xxii] Brown, David Object-Oriented Analysis John Wiley & Sons, Inc. 1997.
[xxiii] Fowler, Martin *UML Destilled,* Addison Wesley 1997.
[xxiv] Brown, David Object-Oriented Analysis John Wiley & Sons, Inc. 1997.
[xxv] Fowler, Martin *UML Destilled,* Addison Wesley 1997.
[xxvi] Brown, David Object-Oriented Analysis John Wiley & Sons, Inc. 1997.
[xxvii] *UML Notation Guide,* version 1.1, Rational Software, 1/9 1997.
[xxviii] Ibid.
[xxix] Fowler, Martin *UML Destilled,* Addison Wesley, 1997.
[xxx] *UML Notation Guide,* version 1.1, Rational Software, 1/9 1997.
[xxxi] Johansson, A. Inga hemligheter kvar. Computer Sweden, nr 37, 1998.
[xxxii] Engholm, A. Total datasäkerhet inget att eftersträva. Computer Sweden, nr 62, 1997.
[xxxiii] Ibid.
[xxxiv] Ibid.
[xxxv] Byttner, K-J."Fler än Telia har brister". Computer Sweden, nr 69, 1998.
[xxxvi] Engholm, A. En sak är säker – kryptering. Computer Sweden, nr 33, 1998.
[xxxvii] Magnusson, P S. Jag är en kryptoanarkist. Datateknik, nr 6, 1994.
[xxxviii] Ibid.
[xxxix] Engholm, A. En sak är säker – kryptering. Computer Sweden, nr 33, 1998.
[xl] Enryd, T & Linberg A. *Datasäkerhet vid lösenordshantering* Examensarbete Inst. För Informatik, Handelshögskolan, Göteborgs Universitet, 1997.
[xli] Nilsson, Å. Säkerhetsexperten: "Många företag slarvar med säkerheten". Computer Sweden, nr 7, 1997.
[xlii] Engholm, A. Total datasäkerhet inget att eftersträva. Computer Sweden, nr 62, 1997.
[xliii] van der Brink, R. Datasäkerhet inte bara på burk. Computer Sweden, nr 5, 1997.
[xliv] Ibid.
[xlv] Åsblom, J. Datasäkerhet – bra, om man kan få det. Computer Sweden, nr 4, 1997.

[xlvi] Ibid.
[xlvii] http://www.mds.mdh.se/föreningar/small/firewall
[xlviii] Lobråten, P. Expert slår larm om dålig IT-säkerhet. Computer Sweden, nr 21, 1998.
[xlix] Joel Åsblom *Lösenord läsbart I backoffice* Computer Sweden nr 16, 1999
[l] IDG News. Förenade hackare – ny hotbild på nätet. Computer Sweden, nr 79, 1998.
[li] Andréasson, S-A & Carlsson, C. Datakommunikation för informatik. Institutionen för datavetenskap Chalmers tekniska högskola, 1997.
[lii] Brown, D. Object-Oriented Analysis John Wiley & Sons, Inc. 1997
[liii] *UML Notation Guide*, version 1.1, Rational Software, 1/9 1997
[liv] Bishop, J. Java Gently, Programming Principles Explained. Addison Wesley Longman, 1997.
[lv]Ibid