# Extern Web-service-lösning vid SSAB Tunnplåt i Borlänge- för kommunikation med sina distributions lager

## External Web-service solution at SSAB Tunnplåt in Borlänge- for communication with its distribution warehouses

**Jesper Dotzky**
**Jon Wiklund**

**EXAMENSARBETE**
**Informatik**
**2004**
**Nr: C05/2004**

HÖGSKOLAN Dalarna

# EXAMENSARBETE, C-nivå i Informatik

| Program | Reg nr | Omfattning |
|---|---|---|
| Systemvetenskap, 120p | C05/2004 | 10p |
| Namn<br>Jesper Dotzky<br>Jon Wiklund | Månad/År<br>Maj 2004 | |
| | Handledare: Roger Nyberg<br>Examinator: Göran Hultgren | |
| Företag/Institution<br>SSAB Tunnplåt Borlänge | Handledare vid företaget<br>Mats Larsson | |
| Titel<br>Extern Web-service-lösning vid SSAB Tunnplåt i Borlänge- för kommunikation med sina distributions lager | | |
| Nyckelord<br>Web service, SSL, VPN, Client certificate, COM+, Förvaltning | | |


## Sammanfattning

Detta examensarbete har utförts på SSAB- Tunnplåt i Borlänge under vårterminen 2004 och omfattar 10 veckors arbete.

SSAB sköter idag sin kommunikation med distributionslagren via fax, telefon eller e-post. Eftersom detta är ett ganska tidskrävande kommunikationssätt, vill SSAB ha en smidigare och snabbare kommunikationslösning. Den lösning som SSAB vill ha är en extern Web-service-lösning för att upprätta en säker kommunikation med sina distributionslager.

Parallellt med byggandet av Web-service-lösningen arbetades en förvaltningsmodell fram. Den beskriver hur förvaltningsorganisationen med dess rutiner kan se ut vid implementering av lösningen.

För att skapa en säker förbindelse med Web-servicen skall en webbklient användas som i sin tur anropar en COM+ komponent. Detta för att kunna skicka med certifikatet ifrån webbklienten till webbservern där Web-servicen ligger. COM+ komponenten måste få tillgång till en användarprofil när den kommunicerar med Web-servicen. Detta för att kunna upprätta en SSL-förbindelse i det inledande skedet. SSL-förbindelsen skall läggas i den VPN-tunnel som mVPN tillhandahåller via WSSAL.



Dalarna University
Rödavägen 3
S-78188 Borlänge

Telephone: +46(0)23 77 80 00
Fax: +46(0)23 77 80 50
URL: http://www.du.se/

HÖGSKOLAN Dalarna

# DEGREE PROJECT in Informatics

| | | |
|---|---|---|
| Informatics and Applied Systems Science 120p | Reg number<br>C05/2004 | Extent<br>10 ects |
| Names<br>Jesper Dotzky<br>Jon Wiklund | Month/Year<br>May 2004 | |
| | Supervisor Roger Nyberg<br>Examiner: Göran Hultgren | |
| Company/Department<br>SSAB Tunnplåt Borlänge | Supervisor at the Company/Department<br>Mats Larsson | |
| Title<br>External Web-service solution at SSAB Tunnplåt Borlänge- for communication with its distribution warehouses | | |
| Keywords<br>Web service, SSL, VPN, Client certificate, COM+, System maintenance | | |


## Summary

This degree project has been created at SSAB-Tunnplåt in Borlänge during the spring term 2004 and it covers an effort of 10 weeks.

SSAB-Tunnplåt in Borlänge until today handles their communication with the distribution warehouses by fax, telephone or email. Because this way of communication demands a lot of time, SSAB wants an easier and quicker way to communicate with the warehouses. The solution SSAB wants is an external Web-service solution for communication with its distribution warehouses.

During building the Web-service the authors developed a model over how the Web-service should be administrated. The model describes how to administrate the organisation and the routines when the Web-service are installed in the system.

To create a secure communication with the Web-service SSAB must use a client who calls a COM+ component to be able to send the certificate from the client to the web server. When the COM+ component communicate with the Web-service it must have access to a user profile. Otherwise it can not create a connection with SSL in an early stage. The connection should be placed in a VPN that is provided by mVPN through WSSAL.

# Förord

Detta examensarbete har utförts på SSAB Tunnplåt i Borlänge under våren 2004 och avslutar vår Systemvetenskapliga utbildning på 120 poäng vid Högskolan Dalarna i Borlänge. Handledare vid Högskolan Dalarna har varit Roger Nyberg och examinator vid Högskolan Dalarna var Göran Hultgren.

Uppkomsten till examensarbetet var att SSAB Tunnplåt behövde en extern Web-service-lösning för att i första hand kunna kommunicera med sina distributionslager. Efter samtal med IT arkitekt Mats Larsson, som varit vår handledare på SSAB, åtog vi oss uppdraget.

En nyttig erfarenhet som vi kan dela med oss av är att snabbt sätta igång med det inledande arbetet, som till exempel faktainsamling, tidsplanering, litteraturstudier med mera. Kommer man efter i ett inledande skede blir det väldigt jobbigt i slutet av kursen och det slutgiltiga resultatet blir kanske lidande på grund av det. Tid är någonting som man aldrig kan få för lite av under ett sådant här arbete.

Vi vill rikta ett stort tack till all hjälp vi fått av IT avdelningen på SSAB Tunnplåt och ett särskilt tack till Mats Larsson, Roger Nyberg samt Pär Norlander konsult på Sogeti.

*Examensarbetare:*

Jon Wiklund
Jesper Dotzky

## Innehållsförteckning




Dalarna University
Rödavägen 3
S-78188 Borlänge

Telephone: +46(0)23 77 80 00
Fax: +46(0)23 77 80 50
URL: http://www.du.se/

# 1. Inledning

Intresset för Web-services är idag stort och många tror på framgång för den här tekniken. En Web-service erbjuder möjligheter att skapa applikationer som kopplar ihop system på ett väldigt smidigt sätt. Teknologin erbjuder inte endast utveckling av applikationer utan en snabb sådan, vilket i sin tur bidrar till minskade kostnader för företagen (Kap 6.1 Web-service).

Svenskt Stål AB, förkortat SSAB, har tidigare inte använt någon form av extern Web-service-lösning, men vill nu upprätta en förbindelse mellan Borlänge och sina distributionslager. Detta för att underlätta kommunikationen mellan parterna och för att på ett enkelt sätt kunna ta del av väsentlig lagerinformation.

Detta arbete syftar till att ta fram en prototyp på en extern Web-service-lösning för att upprätta en förbindelse, med utbyte av information, mellan SSAB i Borlänge och sina distributionslager.

## *1.1 Bakgrund*

SSAB har distributionslager utspridda i hela 40 länder i Europa och världen över. Dessa lager har man som buffert för sina produkter. Därifrån distribueras SSAB:s produkter vidare, till grossister och kunder ute i Europa och övriga världen. Nu vill man effektivisera sin kommunikation med dessa distributionslager, för att i första hand snabbt kunna inventera dessa lager.

Då bland annat e-post, som är ett mer omständligt sätt att hantera information tidigare använts för hantering av information mellan SSAB och distributionslagren, har SSAB nu för avsikt att använda en så kallad Web-service-lösning. Den här typen av lösning möjliggör kommunikation över Internet på ett smidigare sätt och hanteringen blir automatiskt snabbare än vid användningen av e-post. Detta för att slippa ha fasta linor mellan Borlänge och alla distributionslager i Europa samt övriga delar av världen. Tidigare har SSAB använt sig av en sådan lösning i några enskilda fall, men dock bara internt inom Intranätet vid SSAB.

I förlängningen ska även transaktioner kunna genomföras, som beställningar och restorder. I och med att informationen i slutändan kommer att skickas i båda led, är kontroll på in och utflöde en viktig detalj för att hålla säkerhetsnivån hög.

Genom att bygga en Web-service-lösning kan SSAB spara mycket tid vid sin hantering av lagerinformation, vid till exempel inventering. Om det sedan blir möjligt att utföra transaktioner via lösningen, kan SSAB utnyttja den inom en hel del olika områden, vid till exempel beställningar, restordrar eller inom andra behovsområden.

Förhoppningen är att lösningen tillgodoser de behov och krav som SSAB ställer på en sådan och att implementering för utnyttjande av den kan bli verklighet inom en snar framtid.

## 1.2 Problemställning

I dag finns ingen extern Web-service-lösning på SSAB, vilket innebär att det inte finns några liknande lösningar där förslag och idéer kan erhållas ifrån. Ytterligare problem vid konstruerandet av en Web-service-lösning externt, är hur SSAB ska ta sig igenom sin säkerhetslösning på ett snabbt och säkert sätt. Eftersom det inte finns någon mall, på SSAB, för hur konstruktionen av en extern Web-service-lösning går till, tillkommer ytterligare problem.

Då tekniken med en sådan här kommunikationslösning är relativt ny för SSAB, försvåras införskaffandet av information därifrån och mer tid kommer att gå åt till att samla in den information som behövs på annat håll.

Det finns en mängd olika tekniker inom det här området och det svåra med teknikvalet är att tekniken måste passa ihop med SSAB:s nuvarande säkerhetslösning. Den är i sin tur byggd av ett företag i Stockholm som heter Portwise (Kap 6.11 Portwise).

Att företaget Portwise finns i Stockholm kan ju också innebära att problem uppstår när kontakt behöver tas. Vilken tekniklösning som än förespråkas, kan den inte användas om den inte är kompatibel med den säkerhetslösning som Portwise konstruerat.

Om inte SSAB klarar av att konstruera en extern Web-service-lösning i sin verksamhet, kommer företaget att vara tvungna att fortsätta använda e-post, fax eller telefon när kontakt behöver tas med dessa distributionslager. Detta innebär att fortsätta använda ett mycket mer tidskrävande och krångligare arbetssätt, då det inte är säkert att ett svar returneras inom den tidsram som man kanske hade hoppats.

## 1.3 Syfte

Att skapa ett underlag för hur den här typen av tjänster skall dokumenteras och förvaltas, samt utreda hur man ska gå till väga för att kommunikationen ska kunna passera en säkerhetslösning obehindrat och samtidigt uppfylla säkerhetskraven.

## 1.4 Mål

Att få en principiell lösning på hur en extern Web-service-lösning ska byggas på SSAB, med fokus på vilket sätt kommunikationen genom deras säkerhetslösning skall fungera.

## 1.5 Metodbeskrivning

Litteraturstudier och intervjuer är de metoder som kommer att användas för faktainsamling och verksamhetsanalys. I övrigt måste en egen metod användas, då det inte finns någon färdig metod att utnyttja i vårt fall. Den metoden är en

modifierad variant av en metod som två tidigare examensarbetare använde vid sitt examensarbete. [26]

Metoden delas in i tre faser, varav varje fas delas in i olika delar.

- Fas 1
  - Planering
  - Verksamhetsanalys
  - Finna olika förslag
  - Sammanfattning
  - Alternativa lösningar
- Fas 2
  - Val av teknik
  - Fördjupning
  - Bygga applikationen
  - Testning
- Fas 3
  - Utvärdering
  - Rekommendation

## 1.6 Avgränsningar

Det kommer bara att skickas enkla meddelanden för att testa Web-service-lösningens funktion. Avsikten är endast att se till att kommunikationen fungerar mellan Rotterdam och SSAB Tunnplåt. Uppgiften är inte att skapa en färdig lösning, utan endast att konstruera en principiell lösning för hur en extern webblösning ska se ut med tyngdpunkt på vilken säkerhetslösning som ger hög säkerhet och fungerar ihop med SSAB:s nuvarande lösning.

Problemet med att lösa hur transaktionerna sker via Web-servicen kommer inte att lösas. Hur man förvaltar trafiken mellan lagret i Rotterdam och affärslagret, löses endast om tid finns. Lösningen måste följa SSAB:s arkitektur och riktlinjer.

## 1.7 Resurser

De resurser som finns att tillgå på eller via SSAB är:

- Datorer
- Servrar
- Väsentlig programvara
- Kontor
- Litteratur
- Mats Larsson (handledare på SSAB)
- Georgios Eliopoulos (teknikavdelningen på Nexus)
- Alex Welinder (konsult på Portwise)

De resurser som finns att tillgå på Högskolan Dalarna är:

- Datorer
- Programvara
- Kontorsutrymme
- Litteratur
- Roger Nyberg (handledare på Högskolan Dalarna)
- Andra lärare

# 2. Metod

*I detta kapitel beskrivs den metod som har använts för att uppnå examensarbetets mål och syfte. Efter omfattande intervjuer och litteraturstudier upptäcktes att det inte fanns någon generellt framtagen metod som passade examensarbetet. Detta resulterade i att en redan befintlig metod användes, som var applicerbar på den problemtyp arbetet innefattade.*

## *2.1 Fas 1 - Framtagande av tänkbara tekniker*

Under denna fas så kommer det övergripande målet vara att ta fram tänkbara tekniker för att lösa problemet som arbetet omfattar. Arbetet ska i denna fas tas från planeringsstadiet till att så långt det är möjligt veta vilka tekniker som marknaden kan erbjuda.

### 2.1.1 Planering

I det inledande skedet av fas ett så ska arbetet planeras så att det utan problem kan genomföras under den utsatta projekttiden. Planeringen påbörjas med att avgränsningarna fastläggs. Med dessa som underlag kommer sedan en tidsplanering samt en projektbeskrivning att göras. Båda dessa dokument kommer att vara ledande och ses som ett ramverk för det fortsatta arbetet.

Under planeringsstadiet fastställs alla standarder som kommer att tillämpas under arbetet upp. Lämpliga standarder att ta fram är t.ex. standarder för hur rapporten ska skrivas, hur bilagor ska skapas och hur versionshanteringen ska gå till mm.

Det stundande arbetet kan underlättas om man inom arbetsgruppen fastlägger olika ansvarsområden. Genom att i ett tidigt stadium tydligt sätter upp ansvarsområden så minskar risken för dubbelarbete och att saker och ting hamnar mellan stolarna.

### 2.1.2 Verksamhetsanalys

Det är viktigt att redan från början ha en god förståelse för hur de redan befintliga systemen fungerar. Det är utifrån dessa som förbättringarna ska göras. Det bästa sättet att få insyn i de system som används är att fråga de som använder systemen. En lämplig person att ta kontakt med är företagets IT-arkitekt. Arkitekten har god insyn i vilka system som används, hur de integrerar med varandra och hur den övergripande strukturen ser ut.

Om det skulle vara något som är oklart kan man söka svaren på sina frågor genom litteraturstudier och intervjua andra personer som antingen arbetar med systemen eller på annat sätt är inblandad i området kring problemet.

En annan bra sak att ta reda på är vilka som är ansvariga för de olika områdena i den IT-relaterade företagsstrukturen. Detta underlättar det fortsatta arbetet då

man slipper gå genom andrahandskällor utan kan gå direkt till förstahandskällan.

### 2.1.3 Söka förslag

Under denna delfas fokuseras arbetet på att ta fram så många tekniska lösningsförslag som möjligt. Alla tekniker som kan lösa problemet eller delproblem är intressanta i detta skede.

Det är viktigt att anstränga sig till det yttersta i denna delfas. Det är inte bra att halvvägs in i arbetet upptäcka att det finns en betydligt bättre lösning på problemet. Det kan både bli kostsamt, tidskrävande och i slutändan kan det äventyra hela projektet. Därför bör många tekniker som möjligt kontrolleras.

Det kan vara lämpligt att använda Internet vid sökning efter lämpliga lösningar. På Internet finns det mycket information att hämta och det är inte ovanligt att det finns andra som har stött på samma problem. Det är onödigt att uppfinna hjulet två gånger.

Det är också viktigt att dokumentera de eventuella lösningarna parallellt med sökandet. Det kan vara svårt att i efterhand komma ihåg hur lösningen egentligen såg ut eller vilken källan var. Det är viktigt att ange var informationen kommer ifrån och vilken källan är.

### 2.1.4 Sammanställning

Utvärderingen kommer att ske parallellt med sökandet efter tekniska lösningar. Utvärderingen dokumenteras noga så att arbetet i efterhand kan presenteras på ett informativt sätt.

Dokumentation kommer att kompletteras med flödesscheman och processkartor så att tekniken kan redovisas på ett överskådligt sätt.

På Internet kan man stöta på andra som har haft liknande problem. Dessa lösningar kan tas med i utvärderingen. Källan till det påstådda lösningsförslaget ska vara noggrant kontrollerat.

### 2.1.5 Alternativa lösningar

Här ska alla lösningsförslag som uppkommit under föregående faser sammanställas i ett dokument. Det utvärderade materialet ska sammanställas så att det blir lätt att jämföra de olika teknikerna mot varandra. För och nackdelar ska tydligt framkomma.

## *2.2 Fas2 - Fördjupning i vald teknik*

Målet med denna fas är att fördjupa sig i de tekniker som valts som lämpliga för att lösa problemet.

### 2.2.1 Val av teknik

I denna fas börjar arbetet med att projektgruppen i samråd med IT-arkitekten väljer ut ett av de förslag som har framkommit under fas ett av projektet. IT-arkitekten väljer lämplig teknik med utgångspunkt i sin förståelse för den övriga arkitekturen och hur den föreslagna tekniken kommer att passa in.

Inför mötet med uppdragsgivaren ska samtliga förslag vara igenomlästa och sammanställda. Dessa förslag bör vara väl dokumenterade, gärna med generella skisser som beskriver applikationens integration i systemmiljön.

Under urvalsarbetet är det viktigt att tänka på att det oftast finns många sätt att lösa ett specifikt problem på. Lösningarna kan ha nackdelar, fördelar eller begränsningar. Tänkbara begränsningar kan vara:

- **Skalbarhet**
  Fungerar applikationen i klustrade miljöer?
- **Hårdvarukrav**
  Är applikationen hårdvarukrävande?
- **Ny teknik**
  Är den stabil och kommer den att vidareutvecklas?
- **Komplexitet**
  Kostnadskrävande utveckling?
- **Integration i nuvarande system**
  Måste nya miljöer utvecklas?
- **Licenskostnad**
  Applikationsmiljöer kan vara dyra i inköp.

Efter avstämningsmötet bör ett dokument uppföras där det utvalda förslaget skrivs ner och att detta dokument skrivs under av uppdragsgivaren och arbetsgruppen. Detta för att säkerställa att inga missförstånd föreligger om vad som bestämts under mötet.

### 2.2.2 Fördjupning

I denna del av fas två fördjupas kunskaperna i det förslag som har valts ut. Med fördjupning avses att genom litteraturstudier och intervjuer fördjupar sina kunskaper i ämnet. Ett lämpligt sätt att starta fördjupningen är att börja med intervjuer av personer som har djup kunskap inom problemställningen. Detta för att få förslag på lämplig litteratur eller andra källor till information, för att snabbt hitta relevant information om problemet och för att undvika att hamna på fel spår redan i initieringsskedet.

### 2.2.3 Bygga applikation

Den första regeln vid byggandet av testapplikationer, är att inte ”uppfinna hjulet igen”. Det finns med stor sannolikhet någon som haft ett liknande problem och löst detta.

Det kan vara lämpligt att börja med att försöka hitta en dokumenterad exempelapplikation som bygger på den teknik som valts från t.ex. litteratur eller Internet. Fördelen att börja med en dokumenterad exempelapplikation är att det är lättare att få en uppfattning om hur tekniken fungerar och hur den kan implementeras.

### 2.2.4 Testning

Vid test och utvärdering är det viktigt att i förväg bestämt hur test och utvärdering skall genomföras.

Frågor som bör besvaras före testet:

- **Mot vad utvärderas förslaget?**
  Utvärderas förslaget mot andra förslag eller mot en given kravspecifikation?
- **Vilka är de prioriterade egenskaperna?**
  Är det prestanda eller säkerhet som är avgörande?

Exempel på egenskaper som kan testas är pålitlighet, funktionalitet och prestanda. Det finns olika typer av testmetoder, detta är några av dessa.

- **Benchmarktester**
  Jämför med en känd standard eller mjukvara.
- **Konfigurationstest**
  Jämförelser mellan olika mjukvaru- och hårdvaru konfigurationer.
- **Funktionstest**
  Har produkten rätt funktionalitet, gör den det den ska?
- **Installationstest**
  Kontrollerar om applikationen fungerar i olika konfigurationer.
- **Integritetstest**
  Kontrollerar pålitlighet och stabilitet av systemet.
- **Prestandatest**
  Undersöker hur systemet uppträder vid hög last.
- **Stresstest**
  Vad händer med systemet vid extrem belastning?

## *2.3 Fas 3 – Utvärdering*

I denna fas ska allt föregående arbete utvärderas för att ligga som grund för den slutgiltiga rekommendationen.

### 2.3.1 Utvärdering

Målet med detta arbetssteg är att ta fram ett underlag till en rekommendation. Rekommendationen ska vara det slutgiltiga förslaget som uppdragsgivaren kommer att erhålla.

Utvärderingen ska göras oberoende av uppdragsgivaren. Detta för att utan yttre påverkan kunna göra en objektiv bedömning. Resultatet som utvärderingen resulterar i kan både vara negativ och positiv. Hela förslaget kan förkastas.

Resultatet av bedömningen ska baseras på allt som framkommit under det föregående arbetet och ställas mot de krav och önskemål som uppdragsgivaren ställt i det inledande skedet av projektet.

Det är framför allt i denna del av metoden som betydelsen av att dokumentera parallellt med arbetet märks. Alla dokument som har produceras under arbetets gång kommer i denna fas att komma till användning.

### 2.3.2 Rekommendation

Här skrivs det dokument som ska ges till uppdragsgivaren. Detta dokument är målet med hela projektet och det sista som görs i projektarbetet. Rekommendationen lämnas över till företagskontakten.

Dokumentet ska vara objektivt utfört. Uppdragsgivaren ska själv utifrån rapporten kunna avgöra om rekommendationen ska genomföras eller inte. Rekommendationen kan både vara negativ och positiv för företaget.

Den föregående utvärderingen kan även ha kommit fram till att hela för- slaget förkastas. En komplett lösning kan helt saknas. Då är det viktigt att rekommendationen tydligt tar upp det i rapporten på samma sätt som om en komplett lösning hade hittats. Allt ska redovisas. Det är lika viktigt för företaget att få reda på att det inte går, som att det går att lösa det problem som formulerats i inledningen av projektet.

Att rekommendationen inte genomförs utan förkastas i detta skede kan bero på många faktorer. Det behöver inte bero på att rekommendationen inte var bra utan det kan bero på andra saker som t.ex. kostnader och personalbrist som gör att den inte kan genomföras.

# 3. Resultat

*I detta kapitel kommer resultatet steg för steg, utifrån metoden och dess tre faser, att presenteras. Detta leder i sin tur till den slutliga rekommendationen.*

## *3.1 Fas 1 - Framtagande av tänkbara tekniker*

### 3.1.1 Planering

SSAB ordnade ett kontor med två datorer och telefon. Datorerna var upp-kopplade till SSAB:s nätverk och hade erforderlig mjukvara installerad.

Det första arbetet efter installationen var att göra klart idéutkastet, som ska ange syfte, mål och avgränsningar med arbetet.

Ett problem var vilken metod som skulle användas, men samtal med Thomas Persson och Rickard Lindberg som varit före oss på SSAB med sitt examensarbete, ledde till att möjlighet fanns till att kunna använda deras metod som de själva konstruerat om behov. Eventuellt skulle uppstå [26]

Tidsplaneringen togs nu fram för att det fortsatta arbetet skulle kunna ske på ett strukturerat sätt.

Ett första möte med IT-arkitekten Mats Larsson genomfördes, han visade SSAB:s IT-arkitektur, samt gav rådet att kontakta de underleverantörer som levererat och implementerat SSAB:s säkerhetslösning.

De olika underleverantörer som skulle kontaktas var Portwise, arkitekt för den säkerhetslösning SSAB använder sig av. Den andra underleverantören var Nexus, som implementerat säkerhetslösningen åt SSAB.

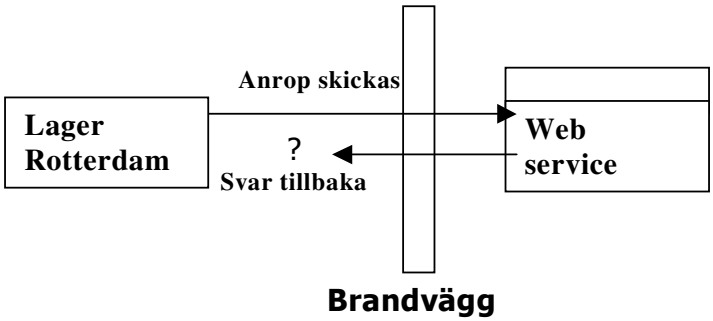
### 3.1.2 Verksamhetsanalys

Ett andra möte med Mats Larsson, gav en del uppslag om hur han mer i detalj hade tänkt arbetet. Detta gav en mall på hur Web-servicen var tänkt att fungera inom SSAB:s arkitektur, samt vilka avgränsningar de satt ut för examensarbetet.

Vad SSAB ville ha var en principiell lösning för hur en extern Web-service skulle se ut, följandes deras arkitektur och riktlinjer. Det var inte tänkt som en färdig Web-service, utan snarare en mall för hur en sådan ska byggas.
SSAB ville i första hand ha en säkerhetslösning som använder sig av säkerhetscertifikat (Kap 6.4 Säkerhetscertifikat). Andra lösningar än säkerhetscertifikat bedömdes som alldeles för osäkra.

Den lösning som kommer att rekommenderades var tvungen att vara kompatibel med Portwise säkerhetslösning, men eftersom Portwise stödjer certifikathantering skulle det inte vara några problem.
Detta gav en möjlighet att med hjälp av säkerhetscertifikat göra en egen lösning för hur Web-servicen ska kommunicera med omvärlden.

Svårigheten låg inte i att skicka ut anrop genom brandväggen, utan i att få ett svar tillbaka (Se figur 1 nedan) (Kap 6.10 Brandväggar).




*Figur 1. Visar var svårigheten ligger för passage genom brandväggen.*

De svar som kom tillbaka var tvungna att via någon säkerhetscertifikatlösning släppas igenom brandväggen, alternativt så hade Portwise en egen lösning på hur man ska integrera en extern Web-service för passage igenom brandväggen.

### 3.1.3 Söka förslag

I denna fas skulle tänkbara tekniker till lösningsförslag eller del av lösning tas fram. Detta har skedde med utgångspunkt i den verksamhetsanalys som tidigare nämnts och det mål som examensarbetet har.

En diskussion inom projektgruppen om olika lösningar utmynnade i att Portwise skulle kontaktas. Hur man än vrider och vänder på de olika lösningar som finns, måste ändå lösningen på något sätt vara kompatibel med Portwise säkerhets-lösning. Analysen av SSAB:s IT-arkitektur visade även på att en så kallad ”flaskhals” uppstod när all trafik ska passera igenom säkerhetslösningen, vilket ytterligare förstärkte argumenten för att Portwise skulle kontaktas först angående lösningsförslag.

Det visade sig att Portwise hade en lösning gjord till sitt säkerhetssystem med avseende på just Web-service, och med tanke på att SSAB:s IT arkitektur är hopbyggd med Portwise säkerhetslösning beslutades det att den lösningen skulle väljas. Onödig tid skulle inte läggas på andra tänkbara tekniker.

En anledning till att det saknades information om Portwise egen lösning på problemet, var att SSAB aldrig tidigare haft en extern Web-service. Följaktligen hade man aldrig haft behov av att fråga Portwise om en sådan lösning fanns tillgänglig.

Den lösning som Portwise har utvecklat, heter Web-service Security Application Layer, förkortat WSSAL (Kap 6.9 WSSAL). Systemet fungerar så att det ligger som ett ”lager” framför Web-servicen och tillhandahåller flera stycken olika säkerhetslösningar under ett tak så som:

Flera olika nivåer av auktorisering

- Central administration
- Användande av certifikat och kryptering för meddelanden
- Inloggnings funktioner
- Implementering av externa säkerhetslösningar

Allt ligger på en server som heter ”Portwise mVPN Access Server”, på det viset kan man ha en centraliserad administration av alla externa Web-services (Kap 6.11.1 mVPN).

För att få reda på mer om WSSAL Kontaktade vi konsulter på både Nexus och Portwise.

De personer som kontaktades var:

- Georgios Eliopoulos (konsult på Nexus)
- Alex Welinder (konsult på Portwise)
- Jon Martinsson (konsult på Portwise)

De resultat som dessa kontakter har gett upphov till kom att presenteras fortlöpande under arbetet.

Svårigheten låg i att avgöra vilken säkerhetsnivå Web-servicen ska ligga på, om skulle räcka med bara en certifikathantering eller om det skulle vara flera olika säkerhetslösningar kombinerade?.

### 3.1.4 Sammanställning

Efter att kontakt med Portwise tagits visade det sig att de hade en lösning gjord till sitt säkerhetssystem med avseende på just Web-service. Med tanke på att SSAB:s IT arkitektur är hopbyggd med Portwise säkerhetslösning beslutades det att den lösningen skulle väljas. Lösningen som Portwise tillhandahöll hette WSSAL och innehöll flera olika säkerhetslösningar, till.exempel inloggningsfunktioner. Allt låg på en server (Portwise mVPN Access Server) för att man skulle kunna ha en centraliserad administration.

WSSAL skulle ge SSAB en möjlighet till administrera flera olika externa Web servicar, enligt samma system som Windows administrerar användare i olika grupper. Varje grupp skulle sedan kunna ha olika säkerhetslösningar beroende på vilka krav man ställde på säkerheten för just den gruppen med Web servicar.

Detta ger sammanslaget en punkt där man samlar all administration för externa Web servicar, vilket gör att förvaltningen kommer underlättas betydligt. På så sätt får man en utbyggbar IT-arkitektur, om SSAB ska ha flera externa Web-service lösningar.

Den skulle sedan kompletteras med en egen designad säkerhetslösning för webbapplikationerna, att läggas under WSSAL

En annan sak som kom upp till diskussion i den här fasen, var om vi skulle använda oss av WS-Security.(Kap 6.7 ). Det är i dagsläget ingen fastställd standard för Web-service, utan ett förslag till standard. Vilket gjorde att SSAB inte ville använda sig av det.

På grund av att om man skulle implementera en lösning med WS-Security som grund och WS-Security inte blev en fastslagen standard. Alternativt om det vart ändringar i WS-Security innan den vart fastslagen standard, skulle man ha en säkerhetslösning som behövde byggas om för att anpassas till gällande standard.

Man skulle med en sådan lösning under gällande omständigheter vare sig ha standardlösningens fördelar eller ha en säkerhetslösning byggd direkt utifrån SSAB:s behov och IT-arkitektur.

### 3.1.5 Alternativa lösningar

Eftersom SSAB ansåg att Portwise lösning skulle användas, har arbetsgruppen inte sökt efter alternativa lösningar.

## *3.2 Fas 2 - Fördjupning i vald teknik*

### 3.2.1 Val av Teknik

Här togs efter diskussion med uppdragsgivarens representant Mats Larsson förnyad kontakt med Portwise. En ny kontaktperson på Portwise vid namn Jon Martinsson skulle bli bollplank och tekniskt stöd.

WSSAL är i sig bara ett administrativt verktyg för flera olika säkerhetslösningar under samma tak. Detta ledde till att uppgiften är att i samarbete med SSAB:s IT-arkitekt Mats Larsson och Portwise företrädare Jon Martinsson bestämma i första hand vilken säkerhetsnivå lösningen skulle ha, och i andra hand ta fram de olika tekniker som ska användas i samverkan.

### 3.2.2 Fördjupning

Eftersom de tekniska lösningar som finns att tillgå var helt okända för arbetsgruppen, inleddes en längre instuderingsperiod.
De arbetsmetoder som användes var följande:

- **Intervjuer**
  I första hand intervjuades personal på SSAB, Portwise och Sogeti samt lärare på Högskolan Dalarna. Intervjuerna genomfördes informellt utan att använda intervjumallar, snarare som ett slags ”brainstorming” möten där det frågades om tidigare lösningar, deras syn på saken samt luftade våra egna idéer.

- **Litteraturstudier**
  Den litteratur som användes fanns till stora delar tillgänglig på SSAB, men även skolbiblioteket användes för att hitta litteratur.

- **Informationsökning på Internet**
  Sökmotorer användes för att finna tillgänglig information.

Som tur var fanns det tidigare lösningar som Portwise gjort åt andra kunder att ha som referens. De berättade aldrig i detalj hur de lösningarna var konstruerade men de kunde i alla fall sortera bort de förslag som var orealistiska eller som inte skulle kunna jämkas ihop med SSAB:s arkitektur.

Under det här skedet gjordes ett designförslag som sedan förevisades för så väl uppdragsgivaren som för Portwise. Detta blev efter ett antal versioner godkänt som arbetsritning, men med tillägget att det kunde ändras i sin slutgiltiga version beroende på hur testerna utföll.

Designförslag var en två lagers IT-arkitektur för säkerheten på applikationsnivå, som skulle kombineras med teknisk lösning.
Kraven på den tekniska lösningen var:

- Snabb och säker inloggning
- Trafiken ska skyddas
- Innehållet skall om möjligt krypteras eller förberedas för kryptering, med tanke på framtida möjligheter för transaktioner
- Ömsesidig identifiering av klient och Web-service
- Fungera ihop med SSAB:s IT-arkitektur
- Vara kompatibel med Portwise säkerhetslösning

Den lösning som förordades var Virtual Private Network (VPN), det innebär att en så kallad tunnel upprättas på nätverksnivå (Kap 6.6 VPN). Den skulle sedan kompletteras med digitala signaturer och certifikat för en säker identifikation, godkänd av uppdragsgivaren (Kap 6.3 Digitala signaturer, Kap 6.4.1 Digitala certifikat). Lösningen är den samma som används vid identifiering av en nod, en dator. I den så kallade tunneln kommer arbetsgruppen även att kryptera informationen med SSL (Kapitel 6.5 SSL), på så sätt får Web-servicen en säkerhet för trafiken med VPN och för innehållet med SSL.

Arbetsgruppen kom att använda sig av personliga certifikat för att skapa SSL-förbindelsen, på så sätt fick arbetsgruppen en säker inloggning, som även möjliggjorde att varje användare via sitt personliga certifikat kan mappas mot ett Windows användarkonto direkt. På så sätt kunde det skiljas på personer (användare) som roller (administrators, users, etc.), vilket gjorde att Web-servicen kan per automatik administreras via SSAB:s befintliga användaradministration.

**Sammanfattning**
Det är flera olika fördelar med de tekniker valts. Tittar man på att lösningen ska vara skalbar inför framtiden, är det en stor fördel att med de utvalda teknikerna kommer säkerheten ligga under applikationsnivån.

Det gör att bara den tekniska lösningen inte applikationerna kommer behövas ändras om till exempel WS-Security skulle bli standard.

Om det kommer en standard för hur transaktioner skall ske via Web-service i framtiden, behöver bara applikationerna byggas om, de utvalda teknikerna kommer inte beröras.

Sammanslaget ger den här lösningen en separation mellan logiken och kommunikation/säkerhet, vilket gör att endera lagret kan bytas, utan att det andra lagret berörs.

### 3.2.3 Bygga applikation

*Eftersom detta kapitel har en så betydande del i rapporten, beskrivs utvecklingen i ett separat kapitel. (Kap 4 COM+ komponent)*

När applikationen byggdes uppstod flera problem med att få designen att fungera som avsett. Dessa problem låg utanför de teoretiska referensramar som författarna besitter, därför har författarna lagt lösningarna på teoretiska såväl som de praktiska problemen i det separata kapitlet.

### 3.2.4 Testning

Under detta avsnitt kommer det slutgiltiga testet av applikationen att redovisas. Som arbetsgruppen tidigare uttryckt, kommer de funktionstester som gjorts under byggnadsfasen inte att redovisas här.

Efter all utförd funktionalitetstestning flyttades webbapplikationerna ut till två riktiga servrar vid namn xborvmapp109 och xborvmapp110. Bägge två var riktiga servrar med egna IP-nummer. De låg i och för sig i SSAB:s egen test miljö, men den byggdes upp så nära verkligheten man kunde komma. Webbapplikationerna fungerade utan mankemang även här.

Nu återstår bara en testning med SSAB:s verkliga brandvägg. Webbklienten lades upp på en server utanför brandväggen, medan Web-servicen läggs upp innanför på en applikations server. Den publiceras sedan i DMZ. När sedan webbklienten

anropar Web-servicen så kommer en VPN-tunnel att skapas automatiskt omkring SSL-förbindelsen. De testerna är inte genomförda i dagens läge, en tröst är dock att det mer eller mindre kommer att handla mer om konfigurering av den befintliga brandväggen, än om test av applikationerna.

## *3.3 Fas 3 – Utvärdering*

### 3.3.1 Utvärdering

Att använda sig av ett färdigt verktyg (WSSAL) som kan administrera flera olika Web-service-lösningar, kopplat direkt till brandväggen, kommer underlätta för SSAB om dom ska ha fler externa Web-service lösningar i framtiden. På så sätt är den övergripande designen helt skalbar, den går att bygga på med många flera olika Web-service lösningar.

Att lägga säkerheten under applikationsnivån för den tekniska lösningen, gör att applikationen inte behöver ändras om tekniken ska bytas ut, och vice versa. Detta gör att även den arkitekturen uppfyller kraven på att vara skalbar.

Med VPN får man ett fullgott skydd av trafiken och krypterar man sen innehållet med SSL uppnås en fullgod säkerhet för trafiken mellan SSAB och deras distributionslager. Den här säkerhetsnivån är även fullgod inför en framtida transaktionshantering.

Användandet av klient certifikat ger flera fördelar för SSAB. SSAB får en säker identifikation av enskilda användare. Varje enskilt klientcertifikat kan mappas direkt mot ett befintligt Windows användarkonto hos SSAB, det gör att ingen extra administration behöver skapas för Web-service-lösningen.

Inloggningen kommer också ske per automatik, användaren kommer inte märka något alls, utan han öppnar bara sin klient skriver in dom uppgifter han vill ha reda på, sen skickar han det till SSAB medelst en knapptryckning. Är sen klientcertifikatet godkänt så är användaren inloggad och han får svar på sin förfrågan utan att ha märkt nån inloggningskontroll alls.

Vid en framtida transaktionshantering via Web-service-lösningen, kan om säkerheten kräver det, även en extra inloggnings kontroll byggas. Den skulle fungera så, att efter klientcertifikatet blivit godkänt, kommer man fram till en inloggningsida som kräver att man lämnar ett lösenord och användarnamn, innan man skickas vidare till Web-service-lösningen. På så sätt är det kontroll både att klientcertifikatet är giltigt, samt även att ingen olovandes använder sig av det.

Man kan givetvis ersätta användarnamn och lösenord med exempelvis, smart card, biometrisk inloggning eller en dosa liknande vad dom flesta internetbanker använder sig av. Dom sistnämda lösningarna känns dock lite avlägsna i dagsläget.

Den lösning arbetsgruppen gjort kommer att fungera som avsett. Att en COM+ komponent används för att upprätta SSL-förbindelsen är ingen nackdel för hur vare sig webbklienten eller Web-servicen är tänkt att fungera.

Det kan dock tyckas använda sig av en COM+ komponent som gränsnitt mellan två webbapplikationer, är lite kaka på kaka. Tyvärr lyckades inte författarna med en annan lösning på hur man ska få en webbklient att kunna skicka med dom privata nycklarna till Web-service-lösningen för att upprätta en SSL-trafik. Problemen med det är väl beskrivna på andra ställen (Kap.4 Lösningen för webbapplikationer och klientcertifikat) i uppsatsen.

### 3.3.2 Rekommendation

En separat rapport kommer göras och skickas till SSAB, detta för att delar av IT-arkitekturen och tekniska uppgifter inte ska komma ut som offentligt handling. Dessa uppgifter förekommer heller inte på andra ställen i examensarbetet, detta av säkerhetsskäl.

Följaktligen kommer våran rekommendation inte skrivas här.

# 4. Lösningen för webbapplikationer och klientcertifikat

*Applikationen kom att byggas i olika etapper, en webbklient och en Web-service som byggdes lokalt. När de fungerade kom de att flyttas över till en testmiljö, där man kunde simulera flera olika servrar på samma maskin. Då de fungerade i testmiljön kom en ”dummy” server sättas upp utanför SSAB:s säkerhetslösning så att en testning kunde ske för att se om allt fungerar även där. Detta test kunde ses som den slutgiltiga testningen av applikationen, det som skedde däremellan var så kallade funktionstester.*

Web-servicen som skulle byggas var i sig inget problem, eftersom den i princip bara skulle skicka någonting och få ett svar tillbaka. Efter diskussioner inom arbetsgruppen beslöts dock att det skulle göras en Web-service som uträttade någonting i alla fall, så att arbetsgruppen visste att Web-servicen fungerade som avsett. Arbetsgruppen lät Web-servicen addera två tal för att sedan returnera summan.

## *4.1 Bygga applikationen*

Funktionstestning av webbklient och Web-service utfördes under byggtiden, detta ska inte förväxlas med testningen av den färdiga applikationen.

Det första testet som innebar att applikationerna var installerade lokalt på datorn genomfördes utan problem. Steg två, att överföra Web-servicen till en server och sedan testa kommunikationen mellan klient och Web-service genomfördes även detta utan problem.

Nästa steg var att det i Web-servicen sattes upp villkor för vilka användargrupper som fick tillgång till Web-servicen och vilka olika Web Methods dessa användargrupper fick konsumera. Detta fungerade på ett tillfredsställande sätt.

Klienten flyttades nu över till en egen server och kommunikation upprättades med Web-servicen, vilket fungerade som avsett. När det upprättats en kommunikation mellan klienten och Web-servicen, liggandes på var sin server, började arbetet med att uppfylla de säkerhetsaspekter som SSAB krävde.

### 4.1.1Upprätta en SSL-förbindelse

Det första var att skapa en förbindelse med Secure Socket Layer (SSL-förbindelse) mellan klient och Web-service. När den var upprättad kunde SSL-förbindelsen sedan stoppas in i en VPN- tunnel, som WSSAL tillhandahåller, vilket visade sig fungera utan problem.

När en SSL-förbindelse skall upprättas finns det tre alternativ, antingen med webbservercertifikat eller antingen med klientcertifikat. Man kan också kombinera de båda. Den första SSL-förbindelsen gjordes med ett webbservercertifikat.

- **Webbservercertifikat**
  I ett webbservercertifikat autentiserar servern den klient som certifikatet ligger på. Det är ett ganska bra alternativ, eftersom alla användare på en specifik webbserver då kan bli automatiskt autentiserade. Vad som kan bli problem är hur man ska skilja på användargrupper och användare. Det skulle kräva att man rent kodmässigt skriver in hur man skiljer grupperna åt. Förbindelsen fungerade som avsett och inga problem uppmärksammades.

- **Klientcertifikat**
  Nästa SSL-förbindelse som upprättades var en förbindelse med ett klientcertifikat. Ett klientcertifikat autentiserar en användare. Varje certifikat kan sedan, hos den server som Web-servicen ligger på, mappas mot ett användarkonto i Windows. Det gör att man inte behöver skriva särskild kod för att skilja olika användargrupper eller användare åt, detta sköts av Windows Active Directory.

- **Kombinera teknikerna**
  Om man kombinerar dessa två tekniker får man ett så kallat ”dubbelt handslag” där först servern presenterar sig som SSAB:s server med sitt certifikat och sedan klienten presenterar sig med sitt personliga certifikat. Då får man en utväxling av privata nycklar mellan server och klient, som leder till att en SSL-förbindelse kan ha upprättats med ett ömsesidigt godkännande av både klient och server.

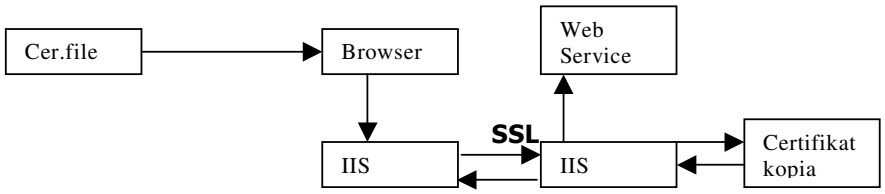
En SSL-förbindelse med klientcertifikat upprättades genom att en Windows-form applikation gjordes och en proxy-klass för en Web-service lades till den aktuella applikationen. Det fungerade som avsett, men när en webbklient upprättades i stället för en Windows-form applikation uppstod problem. Det gick tyvärr inte att upprätta en SSL-förbindelse med ömsesidigt godkännande.

## *4.2 Analys av webbapplikationer och omgivande system*

Anledningen till att det inte fungerade är svårbedömt, eftersom det är så många faktorer med i bilden när det gäller just en Web-service. Så för att förklara vad som utförts, måste det först göras en utvikning och berätta varför det utförts på det sättet.

### 4.2.1 Klient certifikat

Att via en browser anropa en Web-service med ett klientcertifikat var ganska enkelt att genomföra (Se figur 2 nedan).

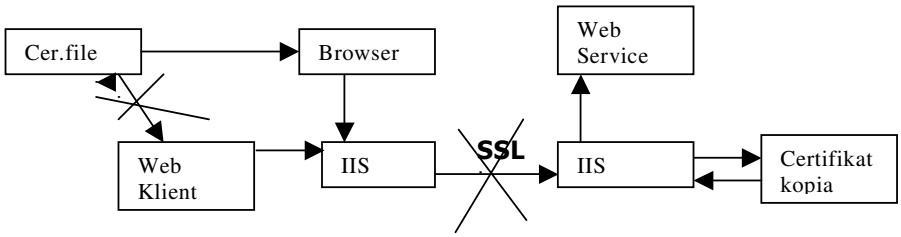



*Figur 2. Visar hur man via en browser anropar en Web-service med ett klientcertifikat.*

Det hela byggde på att man i en browser, exempelvis Internet Explorer, hade en direktkoppling till det klientcertifikat man behövde för att autentisera sig mot webbservern (IIS) som själva Web-servicen låg på. Allt som behövdes var att man exporterade en kopia (utan den privata nyckeln) av ett klientcertifikat man hade åtkomst till via sin browser, till den andra webbservern och installerade samt konfigurerade det för klientcertifikat- autentisering.

På det här sättet skapar man en krypterad förbindelse mellan två webbservrar utan att komma upp på applikations nivå. Vilket innebär att eventuella försök till att, utan rätt klientcertifikat, anropa Web-servicen inte ens når fram till densamme med det så kallade ”Response-Challenge” anropet, utan webbservern stoppar anropet.

Om man istället för att anropa Web-servicen via en browser, bygger en webbklient (Webbapplikation) som anropar Web-servicen, blir det genast mycket krångligt att använda sig av klientcertifikat, se figur 3 nedanför.




*Figur 3. Visar varför en webbklient inte kan använda browserns certifikat eller certifikaten i certifikatlagret.*

Klienten har ingen åtkomst till browserns certifikat, den har heller ingen direktkoppling till certifikatlagret för vare sig den lokala maskinens eller användarens certifikat (Kap 6.4.4 Certifikatlager). Dessutom drivs webbservern via IIS-processen (inetinfo.exe i W2k) och klienten av ASP.NET processen (aspnet_wp.exe i W2k, Network Service i W2k3) (Kap 6.15 Network service). Dessa processer är inte synkroniserade av säkerhetsskäl.

En lösning på det här är att använda sig av tekniken som en Windows forms applikation använder sig av. Det innebär att man skickar med certifikatet i en cookie. Arbetsgruppen kommer inte att gå närmare in på hur det fungerar med Windows forms applikationer, vilka processer, hur anropet sker etc. Det ligger utanför avgränsningarna för det här examensarbetet. Det har dock sina nackdelar vilket gör att vi inte kommer att använda oss av det.

Om man analyserar problemet med webbklienten, upptäcker man att webb-klienten inte har någon som helst tillgång till certifikatet. Första steget måste alltså vara att se till att webbklienten får tillgång till certifikatet.

Det finns en klass som heter X509Certificates som i sin tur ligger i biblioteket System.Security.Cryptography. Den kan hantera certifikat, dock bara av certifikattypen X509 som namnet anger. Den används på så sätt att man lägger en kopia av certifikatet i roten till C: , sen läser man in det i ett speciellt X509-objekt av klassen X509certificates, som man lägger till i ClientCertificates funktionen i proxy- klassen. Här nedan finns det exempel kod för hur man ska göra det i en Windows Forms applikation:

```
[Visual Basic]
' Instantiate proxy class to a Bank XML Web service.
Dim bank As BankSession = new BankSession()

' Load the client certificate from a file.
Dim x509 As X509Certificate =
X509Certificate.CreateFromCertFile("c:\user.cer")

' Add the client certificate to the ClientCertificates property
' of the proxy class.
bank.ClientCertificates.Add(x509)

' Call the method on the proxy class, which requires
authentication
' using client certificates.
bank.Deposit(500)
[C#]
// Instantiate proxy class to a Bank XML Web service.
BankSession bank = new BankSession();

// Load the client certificate from a file.
X509Certificate x509 =
X509Certificate.CreateFromCertFile(@"c:\user.cer");

// Add the client certificate to the ClientCertificates
property
// of the proxy class.
bank.ClientCertificates.Add(x509);

// Call the method on the proxy class, which requires
// authentication using client certificates.
bank.Deposit(500);
```

Ovanstående kod fungerar i en Windows Forms applikation. En sådan byggdes och testades av författarna för att se om koden fungerade och den fungerade utan problem.

Ett problem som uppstår direkt är att X509-klassen inte har tillgång till certifikatets privata nyckel när man använder en webbklient, det innebär att utbytet av nycklar för att bekräfta identiteten hos webbklienten fallerar igen. X509Certificates klassen stöder inte de två viktiga funktioner som måste användas i det här fallet, vilka är:

- Hämta certifikat ifrån certifikatlagret direkt.
- Den kan inte hitta den privata nyckeln som är kopplad till den publika nyckeln för ett specifikt certifikat, i nyckeldatabasen.

Anledningen till att det inte går är ASP.Net processen som driver webb klienten. Den har av säkerhets skäl inte någon som helst tillgång till de privata nycklarna. Alla sådana processer försöker man ge så lite rättigheter som möjligt av säkerhets skäl. I W2k3 (Windows server 2003) har man gått steget längre och alla rättigheter är indragna som standardkonfigurering. Det har även gett andra problem i felsökandet, men det återkommer arbetsgruppen till senare.

## *4.3 Testning av nya uppslag*

De tester som genomförts med X509-klassen för att kunna skicka med ett klientcertifikat, har gjorts med utgångspunkt av ovanstående beskrivning av hur klassen fungerar.

Alla nya tester har också uteslutande utförts i W2k3 miljö, detta för att en färdig lösning kommer att läggas på en sådan applikationsserver.

En ny analys av problemet gav oss idén att på något vis låta Network service (ASP.NET processen i W2k3) få rättigheter till certifikatens privata nycklar. Det första som gjordes var att var att söka fram mappen som innehåller de privata nycklarna och tilldela Network service läs, skriv och exekveringsrättigheter i den mappen. Det gav inget resultat, utan arbetsgruppen hade fortfarande inte lyckats skicka med certifikatets privata nycklar. Ytterligare försök gjordes med att tilldela Network service rättigheter till otaliga mappar, med lika negativt resultat.

Ett verktyg som finns i WSE 2.0 (Web-service Extension 2.0) X509 tool laddades ner från Internet. Detta verktyg gjorde att rättigheterna för ett individuellt certifikats privata nycklar kunde sättas till Network service, eller vilket annat tjänstekonto som helst, direkt. Det gav inte heller någon effekt i positiv riktning.

### 4.3.1 Alternativa uppslag

Vid det här laget inleddes samtal med en konsult på Sogeti (Pär Norlander [29]), som vid den tidpunkten utförde uppdrag åt SSAB. Han gav arbetsgruppen åtskilliga uppslag på hur vi skulle gå vidare. Ett uppslag var att lägga Web-servicen och webbklienten i en applikationspool på respektive webbserver och sedan sätta processidentitet för applikationspoolen till Network service.

Pär hade genomfört ett projekt som gick ut på att reda ut hur SSAB skulle kunna lastbalansera (Kap 6.13 Lastbalansering) sina servrar. Där hade han haft stora problem med personifieringen(det vill säga processen körs under en användare, och får automatiskt samma rättigheter som användaren) över maskingränserna. Detta hade han löst genom att skapa en applikationspool på webbservern som webbplatserna läggs i. I applikationspoolen kan han sedan konfigurera vilka användare som kan köra applikationerna.

I det här fallet använde sig arbetsgruppen av samma teknik med en applikationspool, fast arbetsgruppen fördefinierade i stället ”security account” - alternativet till IIS_WPG (IIS Worker Process Group)(Kapitel 6.15.1 IIS_WPG). Till den gruppen hör även Network service hemma. På så sätt får både IIS och ASP.NET-processerna samma rättigheter bara man tilldelar gruppen rättigheter till en mapp.

Dessvärre löste inte det arbetsgruppens problem med att få autentiseringen av klientcertifikat att fungera. För att arbetsgruppens färdiga applikation ska fungera måste den läggas i en ”applikationspool”. Detta har just med lastbalansering av webbservrarna och personifiering att göra och på det viset fick arbetsgruppen ett framtida problem löst i förväg.

## *4.4 Ny analys av problemet*

Om man skärskådade hela processen från webbklient till Web-service med alla webbservrar och tjänsteprocesser inkluderade, verkade det på något sätt som att så fort man blandade ihop en webbklient med en browser så funkade ingenting som hade med autentisering av klientcertifikat att göra. Detta tydde på att arbetsgruppen angrep problemet ur fel infallsvinkel varpå arbetsgruppen, ihop med konsulten från Sogeti, itererade igenom processen och försökte finna nya angreppsvinklar.

Mängder av forum och Internetsajter genomsöktes efter information om Web-service och klientcertifikat. Resultatet av utsökningarna påvisade att arbetsgruppen inte var ensamma om problemet. Utan att överdriva kan man säga att problemet med att skicka med de privata nycklarna dök upp i 3-4 olika versioner på nästan alla forum. Lösningförslagen på problemen var snarlika med hur arbetsgruppen gjort tidigare med att på något vis få ASP.NET-processen att tilldelas rättigheter med de privata nycklarna.

### 4.4.1 W2k3 vinklingen

En annan ide var att lösningsförslagen till största del var avsedda för W2k och applikationerna låg i en W2k3 miljö. Arbetsgruppen funderade om det var något med konfigureringen i W2k3 som ställde till problem? När man installerar en W2k3 server är alla rättigheter borttagna och man måste manuellt dela ut rättigheterna till användare och processer.

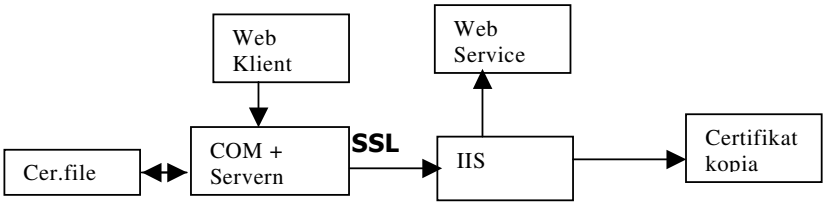
Här genomfördes det stort arbete med att försöka ta reda på hur man ska konfigurera en W2k3 applikationsserver. Problemet var att det inte på något ställe

stod i klartext hur man gör för att exempelvis få det att fungera med en Web-service-lösning och ett klientcertifikat. Informationen ligger utspridd på flera olika platser upplevs det som, sen är det upp till användaren att hitta de olika delarna och sätta ihop den konfigurering han själv vill ha.

Under det här arbetet rådfrågades flera olika personer på SSAB och flera olika personer på konsultfirmor för att få eventuella råd om hur W2k3 skulle kunna konfigureras, men någon lösning på arbetsgruppens specifika problem erhölls aldrig.

## *4.5 COM+ komponent*

Microsoft visade sig ha en alternativ lösning till hur man skulle kunna skicka med certifikatet ifrån webbklienten till webbservern där Web-servicen ligger. Arbetsgruppen hade tidigt under sitt arbete träffat på den lösningen, men lagt den åt sidan på grund av att det kändes lite som kaka på kaka att använda en webbklient som skulle anropa en COM+ komponent, som i sin tur skapade förbindelsen med Web-servicen (Se figur 4 nedan) (Kapitel 6.14 COM/COM+). Den lösningen bygger på att man kör en komponent på COM+ servern (Enterprise services) under ett användarkonto.




*Figur 4. Visar hur man med hjälp av en COM+ komponent skapar en SSL-förbindelse.*

Det här bygger på att COM+ komponenten har tillgång till en användarprofil när den kommunicerar med Web-servicen. Anledningen till det är att den behövs för den inledande ”handskakningen” när man upprättar en SSL-förbindelse.

### 4.5.1 Varför en användarprofil?

När en SSL-förbindelse upprättas utbyts ett antal detaljer mellan klient och server. Bland annat server certifikatet, klientcertifikatet (utan privata nycklar) och en krypterad bit information även kallad "pre-master secret"

Den görs med klientens privata nycklar som sedan används ihop med servern för att skapa en färdig krypteringsnyckel, även kallad ”master secret”, som sedan används för att kryptera all trafik.

För att inför webbservern verifiera att man har den privata nyckeln till det certifikat som används, måste man skapa en så kallad "pre-master secret". Nu kommer arbetsgruppen tillbaka till det här med rättigheter till olika mappar. Hela

processen kan nu få tillgång till användarens privata nycklar, eftersom dessa ligger i användarens profil. Detta sker nu med automatik eftersom din användarprofil har tillgång alla dina privata mappar.

ASP.NET kontot, som man kör alla sorters webbapplikationer under, har som standard konfigureringsinställningen "Deny interactive logon". Det gör att man inte kan logga in på datamaskinen med det kontot och som resultat får man följaktligen ingen användarprofil heller.

Man ska inte tilldela ASP.NET-kontot, eller något annat konto som man kör webbapplikationer under, möjligheten att kunna logga in. Man måste alltid tilldela så få rättigheter som möjligt av säkerhetsskäl.

## *4.6 Avslutande funktionalitetstest*

Nu fungerade applikationen till fullo och en SSL-förbindelse kunde upprättas mellan klient och server. En diskussion inom arbetsgruppen om det var den lösningen som skulle väljas för projektet genomfördes. Det beslutades i samråd med SSAB att det var den lösningen man skulle använda. Det är så pass stora fördelar med att använda webbapplikationer för SSAB. Applikationerna flyttades ut på olika servrar och testades sen igen och allt fungerade som det skulle.

# 5. Förvaltning

*Under detta kapitel presenteras ett förslag på hur förvaltnings organisationen, med dess rutiner, kan se ut på SSAB Tunnplåt vid implementering av den externa Web-service-lösningen.*

Arbetet med att ta fram en förvaltningsmodell (Kap 6.16 Förvaltningsmodell) över hur den här typen av lösningar kan förvaltas har skett parallellt med arbetet att bygga applikationen. Eftersom huvudmålet med examensarbetet var att lösa hur kommunikationen genom SSAB:s säkerhetslösningar skulle gå till, valde arbetsgruppen att fokus skulle läggas på byggandet av applikationen.

SSAB:s säkerhetslösningar förvaltas efter en modell som kallas Portwise-processen. Portwise processen omfattar behovsanmälan, drift, felanmälan och ändringsbegäran. Den är reglerad av policyregler.

Efter intervju med Jeanette Näslander, som varit med och tagit fram Portwise processen, framkom det att den Web-service-lösning som arbetsgruppen konstruerat kommer att ligga inom den arkitekturen. Detta innebär att den förvaltningsmodell som arbetsgruppen utvecklat inte kommer att skilja sig nämnvärt från Portwise processen.

Under framtagandet av Web-service-modellen har vissa författares syn tagits i beaktande, men som tidigare nämnts har Portwise processen legat som grund vid framtagandet av förvaltningsmodellen. [4][5][6][7]

## *5.1 Organisation*

Här beskrivs hur förvaltningsorganisationen bör se ut för hantering av Web-service-lösningen. Vi anser att det för den här typen av modell behövs ungefär fem personer för att den skall fungera på ett tillförlitligt sätt.

### 5.1.2 Roller

Här beskrivs de roller som bedöms som viktiga i modellen. Kommer det att införas ett flertal olika Web-service-lösningar måste naturligtvis rollerna Web-service-förvaltare och helpdesk utökas med fler personer.

- **Systemägare och IT-chef**
  De här personerna har det yttersta ansvaret för systemet. Alla större beslut bör gå genom de här personerna. Det krävs stort engagemang i och med att det medföljer ett stort ansvar i de här rollerna. Det är de här personerna som ser till att rätt personal är på rätt plats i verksamheten. Systemägaren bör upprätta ett affärsavtal med IT-chefen.

- **Web-service förvaltare**
  Måste ha god kunskap om hur modellen fungerar för att kunna åtgärda eventuella fel som uppstår. Ska ha befogenhet att hyra in externpersonal vid behov. Web-service-förvaltaren bör ha befogenhet att upprätta kontrakt

med nya användare som system ansvarig skickat dit. Uppstår problem vid kontrakterandet skickas ärendet till systemägaren/IT chefen som får ta det slutgiltiga beslutet.

Det ska finnas en förvaltare som är väl insatt i driften av Web-servicen, dvs. lägga upp nya användare mm, och en förvaltare som är väl insatt i tekniken bakom den.

- **Systemansvarig logistik**
  Den person som är ansvarig för den dagliga användningen av förvaltningsobjektet och sitter på logistikavdelningen. Den här personen skall avgöra vilken/vilka användare som skall få använda systemet och vilka befogenheter de skall få. Speciellt om framtida transaktioner kommer att utföras.
  Dessa användare skickar han/hon vidare till Web-service förvaltarna.

- **Helpdesk**
  De personer som sitter här bör ha en förståelse för tekniken och ska därmed kunna hjälpa, på ett mindre tekniskt plan, de användare som ringer in och har problem. Är problematiken för svår skickas frågan vidare till Web-service förvaltarna.

| **Verksamhet** | **IT** | **Nivå** |
|---|---|---|
| Systemägare | IT- chef | Budgetnivå |
| Systemansv. Logistik | Web-service förvaltare | Beslutsnivå |
| Helpdesk | Web-service förvaltare | Operativnivå |

*Tabell 1. Visar rollerna på respektive nivå i förvaltningsobjektet.*

- **Budgetnivå**
  Rollinnehavarna på budgetnivå har roller av styrande karaktär. De godkänner förvaltningsplaner och tecknar eventuella avtal. Dessutom beslutar de om eventuell utökning eller indragning av resurser till arbetet. Rollinnehavarna ingår alltid i styrgruppen.

- **Beslutsnivå**
  Rollinnehavarna på den beslutande nivån har det verkställande ansvaret för att förvaltningsobjektet fungerar enligt beslutad förvaltningsplan. Inom de mål som fastställs ansvarar rollinnehavarna för att prioritera och genomföra ändringar i syfte att ge största möjliga måluppfyllelse. Rollinnehavarna bemannar alltid förvaltningsgruppen.

- **Operativnivå**
  Rollinnehavarna på den här nivån arbetar med det operativa genomförandet av förvaltningen av förvaltningsobjektet. Det innebär till exempel genomförande av ändringar, användarstöd i olika led, driftaktiviteter samt text- och bildhantering.

Kommunikationen ska ske antingen horisontellt eller vertikalt, aldrig diagonalt.
Ett affärsavtal bör upprättas mellan systemägare och IT-chef.
Lägre nivåer av avtal kan upprättas om behovet uppstår, men i dagsläget ser vi inget behov av sådana avtal.

### 5.1.3 Grupper/möten

**Grupper**
Någon form av förvaltningsgrupp bör skapas. Där skall alla berörda personer i verksamheten ingå som systemägare/IT-chef, Web-service förvaltare, systemansvarig logistik, helpdesk och någon eller några av användarna i systemet.

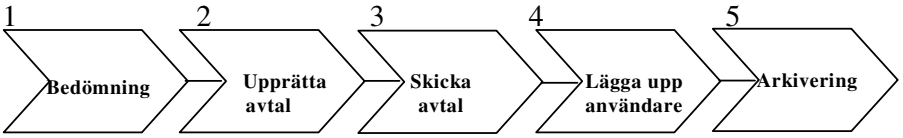
Det är viktigt att få med minst en användare i förvaltningsgruppen. Detta för att erhålla synpunkter, från någon som verkligen använder systemet, över för och nackdelar med systemet.

**Möten**
Möten i förvaltningsgruppen bör framförallt hållas när behov uppstår, som t.ex. när ett allvarligare fel uppkommer eller någon form av ändring skall införas. Någon form av rapportering till systemägaren/IT chefen bör dock ske fortlöpande. Det är alltid systemägaren/IT-chefen som har det största ansvaret och som därmed tar alla viktiga beslut.

## *5.2 Rutiner*

Här beskrivs vilka rutiner som bedöms som viktiga för att drift och förvaltning skall hållas flytande.

### 5.2.1 Behovsanmälan

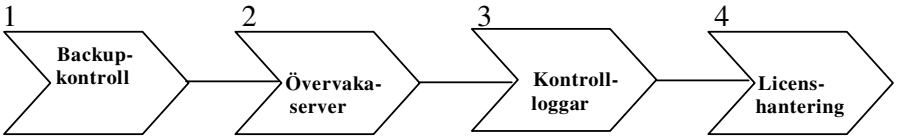



*Figur 5. Visar tillvägagångssättet när en behovsanmälan för att lägga till en ny användare i systemet har kommit in till Web-service-förvaltare.*

När en användare vill få access till systemet måste det gå genom systemansvarig logistik som sedan skickar ärendet vidare till Web-service förvaltare för upprättande av avtal. De rutiner som bör finnas är (Se även figur 5 ovan).

1. Bedömning om personen behöver erhålla access till systemet
2. Upprätta ett avtal med berörd person
3. Skicka avtal för underskrift
4. Införa användaren i systemet
5. Arkivering av avtal

### 5.2.2 Drift




*Figur 6. Visar de delar som driften innefattar.*

Ett löpande underhåll måste hållas under den här processen. Driften måste innefatta följande rutiner (Se även figur 6 ovan).

1. Backup kontroll
2. Övervakning av server
3. Kontroll av loggar
4. Kontroll av licenser

Dessa rutiner kan innebära att en felanmälan eller ändringsbegäran uppstår.

### 5.2.3 Felanmälan

**Mindre fel**
Vid frågor från användare, som kanske inte handlar om direkta fel i systemet, bör ett svar med åtgärdsförslag kunna komma direkt från helpdesk.

**Större fel**
Om inte helpdesk kan åtgärda en förfrågan om fel ska Web-service förvaltarna kontaktas. De tar i sin tur ställning till huruvida de själva kan åtgärda felet som uppstått eller om extern hjälp måste inhyras.

Om felet kräver en större ändring i systemet är det viktigt att en eller flera användarrepresentanter är med när ändringen införs, detta för att dels få synpunkter på ändringen och för att användaren/användarna snabbt skall kunna sätta sig in i hur systemet kommer att fungera efter ändringen. Det kan här vara lämpligt att hålla någon form av förvaltningsmöte.

**Dokumentation**
Felet som åtgärdats bör dokumenteras och arkiveras. Om en större ändring gjorts i systemet bör det ändras i eventuella användarmanualer.

### 5.2.4 Ändringshantering

En ändringsbegäran bör vara skriftlig. På det viset undviks missförstånd som kan innebära att problem uppstår i systemet.

Efter inkommen begäran ska Web-service-förvaltaren/förvaltarna göra en bedömning om ändringen går att genomföra. Går ändringen inte att genomföra

avslutas ärendet direkt. Går den att genomföra bör ett förvaltningsmöte ordnas där beslut skall tas om huruvida ändringen skall genomföras eller inte.

Närvarande på mötet bör vara:

- Systemägare/IT chef
- Web-service förvaltare
- Systemansvarig logistik
- Helpdesk
- Användarrepresentant
- Eventuell extern personal

Om beslutet blir positivt skall ändringen genomföras och funktionstestas, av Web-service förvaltaren/förvaltarna och användarrepresentanter, för att sedan dokumenteras och arkiveras. Eventuella användarmanualer bör ändras.

Om beslutet blir negativt avslutas ärendet direkt efter mötet. Dokumentering och arkivering bör ske även om beslutet blir negativt. Vissa idéer som luftats kan vara av vikt längre fram i tiden och därför är det viktigt att få ner dessa idéer på papper.

# 6. Teori

*Här kommer begrepp att förklaras som är viktiga att förstå, för att kunna tillgodogöra sig informationen i examensarbetet.*
*De begrepp som behandlas är:*

- Web-service
- Kryptering
- Digitala signaturer
- Säkerhets certifikat
- SSL
- VPN
- Web-service security
- SOAP
- WSSAL
- Brandväggar
- Portwise
- Kerberos
- Lastbalansering
- COM/COM+
- Network service
- Förvaltningsmodell

## *6.1 Web-service*

Web-services erbjuder möjligheter att skapa applikationer som kopplar ihop system på ett väldigt smidigt sätt. Teknologin erbjuder inte endast utveckling av applikationer utan en snabb sådan.

Användare har, som bekant, möjlighet att via webben ansluta sig till en applikation. En Web-service däremot, gör det möjligt för en applikation att ansluta sig till en annan applikation, vilket i sin tur ger flera anslutnings-möjligheter som kan spara tid och pengar för företagen.

Eftersom webbtjänster använder XML för att formatera förfrågningar och svar, kan de exponeras från och konsumeras på vilken plattform som helst som kan formatera och analysera ett XML-meddelande.

Genom att låta webbtjänster överföras via HTTP kommer man till rätta med problemet att kommunicera över Internet och över företags brandväggar. Då behöver inte tredjepartslösningar användas, som kräver att ytterligare kommunikationsportar görs tillgängliga för extern åtkomst, vilket i sin tur innebär högre kostnader. Beroendet av en enda leverantör som sköter underhåll och uppgradering försvinner.

Några typiska standarder för att bygga Web-services är Simple Object Access Protocol (SOAP) som underlättar kommunikationen mellan applikationer som är

skrivna i olika programmeringsspråk och placerade på olika plattformar. Web-service Description Language (WSDL) som utgör en standardmetod för att beskriva vilka funktioner som finns tillgängliga genom en viss XML- Web-service och vilka variabler som måste överföras för att anropa tjänsterna. Universal Description, Discovery and Integration (UDDI) som utgör en katalog över XML- Web-services som gör att det går att hitta de företag som erbjuder tjänsterna.

Ofta är det underförstått att Web-services är något som körs på en applikationsserver, exempelvis Websphere application server från IBM eller Microsofts .NET arkitektur.

XML- Web-services är enheteter av programsökvillkor som ger information och tjänster till andra program. Programmen får åtkomst till dessa via standardwebbprotokoll och dataformat som till exempel HTTP, XML och SOAP, oberoende av hur respektive Web-service används.

En mindre teknisk beskrivning av det dynamiska beteendet som Web-services uppvisar är att de kännetecknas av att:

- de själva beskriver sin funktionalitet
- beskrivningar av tjänsten publiceras
- de kan lokalisera efterfrågad funktionalitet
- de kan skicka förfrågan efter data
- de kan utbyta data med andra Web-services.

[13,18,21,22]

### 6.1.1 Krav för Web-service

För att en Web-service ska bli riktigt användbar måste den tillhandahålla följande:

- Kontrakt som anger de parametrar och datatyper som Web-servicen förväntar sig, samt de typer av svar som sänds till dem som anropar Web-servicen.
- Funktion för lokalisering av Web-servicen.
- Någon metod som gör det möjligt att upptäcka Web-services som erbjuds av en viss server eller applikation.
- Beskrivning av tjänsterna som erbjuds.

### 6.1.3 Web-service säkerhetsarkitektur

När man tittar närmare på säkerhetsarkitekturen på olika Web-services kan man dela upp den i två olika tekniker.

- Kommunikationsskyddade säkerhetslösningar
- Innehållsskyddade säkerhetslösningar

De kommunikationsskyddade säkerhetslösningarna kräver inte någon form av modifiering av koden utan tar hjälp av protokollen Secure Socket Layer/Transport Layer Security (SSL/TLS) (se kap 6.5 SSL) eller IP security protocol (Ipsec) för att skapa en säker anslutning. Eftersom protokollen ligger nedanför applikationslagret i TCP/IP modellen (Se figur 7 Kap 6.6), är allt som behövs en ändring i Web servern.

De innehållsskyddade säkerhetslösningarna bygger på att man skyddar aktuell XML- data med någon form av kryptering. Detta sker genom modifiering i koden med hjälp av en kodnyckel som bestämts i förväg. Digitala signaturer är också en möjlighet man har för att avgöra om meddelandet har blivit modifierat på vägen mellan avsändare och mottagare.

Det är givetvis möjligt att kombinera dessa två tekniker för att skapa en säkerhetslösning utifrån sina egna behov.

### 6.1.4 Web-service autentisering

Autentisering används för att avgöra om en användare eller dator har rätt att använda sig av Web-servicen. Det finns flera olika sätt för autentisering med olika säkerhetsnivåer beroende på vilka krav man ställer på säkerheten.

Kommunikationsprotokollet SOAP, som Web-services använder, har inget eget säkerhetsprotokoll utan förlitar sig på webbserverns lösningar. Anledningen till det är att SOAP är transportoberoende. För det mesta använder Web-services sig av HTTP som protokoll, men de kan även ha Simple Mail Transfer Protocol (SMTP) eller andra protokoll.

Det finns ett antal färdiga autentiserings tjänster för HTTP och SSL kombinerade. Använder man sig av en Microsoft webbserver IIS (Internet Information Server) ihop med Windows 2000/2003 server, så stödjer de dessa tjänster. Nedan beskrivs de vanligaste:

**Basic:** den vanligaste metoden för autentisering, men har den nackdelen att användarnamn och lösenord skickas i klartext, eventuellt kan de vara krypterade i base64-kryptering, base64 är dock lätt att dekryptera. Används mer för att identifiera användare, och ta fram hans personliga profil, än att skydda data. Det finns dock möjlighet att skydda uppgifterna om inloggningsinformationen med hjälp av SSL.

**Digest:** Överför inloggningsuppgifterna med hjälp av hashfunktioner. Vilket innebär att man inte behöver använda SSL för att skydda användarnamn och lösenord. Digest- autentiseringen stöds av IIS 5 och 6, men bara om Active directory är installerat. Det stöds inte heller av alla utvecklingsverktyg för Web-service.

**Windows Authentication:** Är en autentisering som är inbyggd i Windows, från och med W2k innebär det New Technology Lan Manager (NTLM) och om Active

directory är installerat, även kerberos. Den fungerar bäst inom ett Intranät, eftersom inte alla Web-service klienter stöder denna metod.

## 6.2 Kryptering

Kryptering innebär att man kodar om meddelandet så att det inte ska bli läsbart, för att sedan dekryptera det till läsbar form. Det finns två former av kryptering, symmetrisk och asymmetrisk. Bägge använder sig av nycklar för sina krypteringsalgoritmer.

**Symmetrisk kryptering:** Använder en och samma nyckel till kryptering och dekryptering, det förutsätter att nyckeln hålls hemlig. Vilket innebär att nyckeln måste skickas via en säker kanal eller med cd-rom, diskett eller annan datalagring till de inblandade.

**Asymmetrisk kryptering:** Använder två nycklar till kryptering och dekryptering av meddelanden, den ena nyckeln är privat och den andra är publik.

Data krypterat med den privata nyckeln kan bara krypteras med den publika nyckeln
Data krypterat med den publika nyckeln kan bara krypteras med den privata nyckeln.

Symmetrisk kryptering är snabbare än asymmetrisk, vilket gör den lämpligare att kryptera stora mängder, som innehållet i ett XML-meddelande. Asymmetrisk kryptering är lite långsammare men användandet av två nycklar ger generellt en högre säkerhet, därmed passar den utmärkt till autentisering med digitala signaturer och för att bekräfta att meddelandet inte ändrats på något vis. [3]

## 6.3 Digitala signaturer

Signaturen är kopplad till ett visst meddelande och vilken informationsmängd meddelandet har, detta för att man ska kunna fastställa avsändaren och att meddelandet inte förändrats på vägen ifrån avsändaren. Det behöver inte betyda att själva meddelandet är krypterat, utan man måste skilja på signering och kryptering. En digital signatur identifierar avsändaren och säkerställer att meddelandet är oförändrat, man kan sen med annan teknik kryptera själva meddelandet för att uppnå en högre säkerhetsnivå.

Message Digest (kontrollsumma) är det värdet som man får efter man kört en informationsmängd (meddelandet) igenom en säker hashfunktion. Banker och postgiro använder liknande men mycket enklare system för bank och postgirokontonummer, där används de sista siffrorna i numret för kontroll att det verkligen är ett korrekt kontonummer.

- **Beräkning av digest**
  Ett program räknar fram digesten med hjälp av en känd formel, utifrån meddelandet som ska signeras.

- **Kryptering av digesten**
  Digesten krypteras sen med avsändarens privata nyckel. Den krypterade digesten bifogas sedan med meddelandet som bilaga, det är alltså den bilagan som är själva digitala signaturen.

- **Meddelandet skickas**
  Nu skickas meddelandet tillsammans med bilagan. Det kan skickas med e-post, distribueras över nätverk eller via http.

- **Dekryptering av digesten**
  Mottagaren dekrypterar bilagan med avsändarens publika nyckel, vilket innebär att digesten tagits fram i klartext.

- **Ny digest tas fram**
  Mottagaren räknar fram en digest med samma formel som avsändaren använde.

- **Digestarna jämförs**
  Den nya digesten jämförs med den man fått fram genom att dekryptera bilagan till meddelandet, om summorna stämmer vet man att avsändaren är den man tror och att meddelandet är oförändrat. Stämmer inte summorna så är meddelandet inte att lita på.

En digital signatur ska inte blandas ihop med begreppet elektronisk signatur. Det sistnämnda kan jämföras med en juridisk handling som man med hjälp av teknik kan koppla till en fysisk eller juridisk person så att denne kan skriva under ett juridiskt avtal.

## *6.4 Säkerhetscertifikat*

### 6.4.1 Digitala certifikat

Ett digitalt certifikat är ett sätt att koppla en publik nyckel till en specifik person, företag eller organisation. Man kan även utfärda certifikat för webbservrar, som intygar att nyckeln som en viss server använder verkligen hör ihop med den specifika servern och till företaget som har servern. Nyckeln kan sedan användas för att med hjälp av t.ex. protokollet SSL skapa en säker webbkommunikation.

Certificate authority (CA), som är utfärdaren av certifikatet, kan vara ett tredjepartsföretag eller en myndighet som tillhandahåller en publik nyckel på ett säkert sätt. CA signerar certifikatet digitalt så att innehållet inte ska kunna modifieras. En signatur ifrån en CA innebär att den har kontrollerat ägarens identitet och kan bekräfta detta. Standarden för digitala certifikat är ISO-standarden X.509v3**,** det förekommer även andra format både på Internet och i andra sammanhang.

Själva certifikatet innehåller ägarens namn, e-post adress och andra data. CA anger även dess publika nyckel, adressen till CA:n så att man ska kunna

kontrollera den, giltighets period och serienummer. En digital signatur av allt detta skapas med CA:ns egna nycklar, nu har man ett certifikat som kan användas som del i ett SOAP meddelande eller läggas ut på en webbsida.

### 6.4.2 Servercertifikat

Servercertifikat, som även kallas server-ID:n, gör att servrar som använder ett säkert kommunikationsprotokoll, till exempel SSL, kan identifieras för en säker kommunikation över nätet.

Det innebär att när en webbläsare initierar kommunikationen med en verifierad webbserver via HTTPS protokollet, som är den säkra varianten av HTTP, får den serverns öppna nyckel tillsammans med dess certifikat. Detta gör att webbläsaren på ett säkert sätt kan identifiera serverns ägare och använda dess öppna nyckel för att kryptera data. [3]

### 6.4.3 Klient certifikat

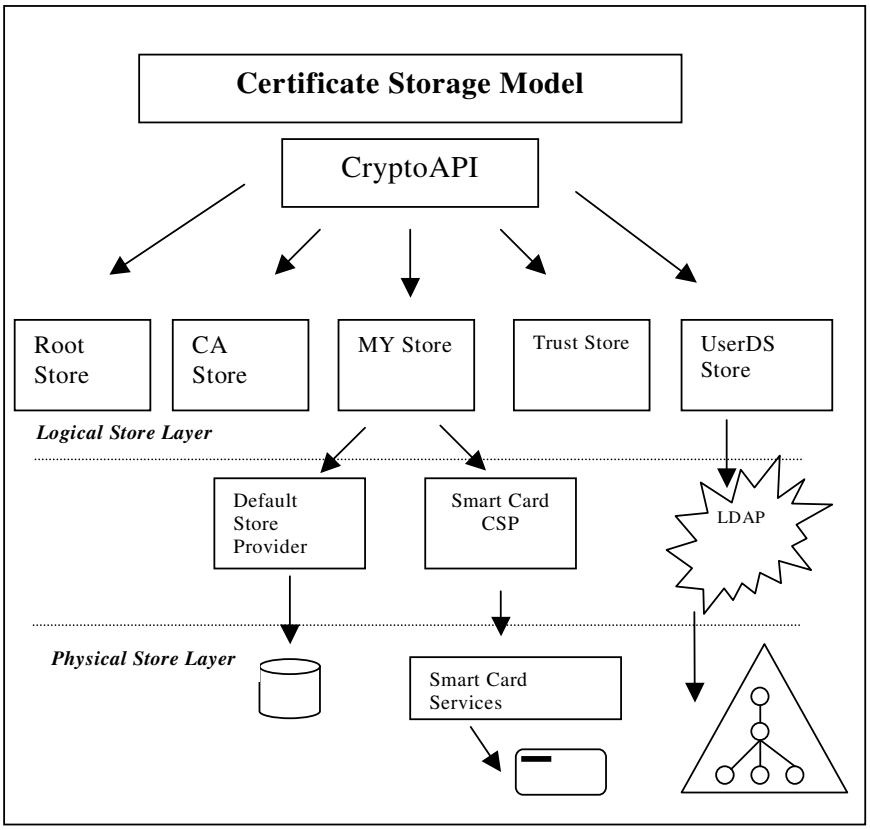
Personliga certifikat, dvs. klientcertifikat gör att människor och servrar kan verifiera en individs identitet. Man kan använda dem med sitt e-postprogram för att skicka krypterade meddelanden och man kan också använda dem med din webbläsare för att komma åt webbplatser som kräver klientverifiering.

Den här typen av verifiering är pålitligare än den klassiska inloggningen som baseras på användarnamn och lösenord. Skälet till detta är att ett stulet certifikat är värdelöst utan den matchande privata nyckeln medan kombinationen användarnamn och lösenord i princip låter vem som helst komma åt webbplatser.

Det krävs heller inte att användaren kommer ihåg sitt användarnamn och lösenord och ett certifikat associerar definitivt en användare med hans eller hennes transaktioner, vilket i sin tur gör att det inte kan ske något återtagande. [3]

### 6.4.4 Certifikatlager

Windows 2000 har ett systemområde där certifikaten förvaras. Detta område kallas certifikatlagret (Certificate Store) och det spelar en central roll för certifikatfunktionaliteten i Windows 2000 (Se figur 7 nedan).




*Figur 7. Bild över certifikat lager modellen.* [24]

Du hanterar certifikat som är installerade i certifikatlagret i ditt system med hjälp av snapin-modulen för certifikat för Microsoft Management Console (MMC). Du öppnar MMC-fönstret genom att skriva ”mmc.exe” i dialogrutan RUN från startmenyn.

Snapin-modulen för Certificate Manager låter dig utföra de vanligaste bearbetningarna av certifikat, t.ex. begära ett nytt certifikat, exportera eller importera certifikat och förnya certifikat.

Förutom detta kan du läsa certifikatinformationen samt kopiera eller ta bort ett certifikat. [3]

**Lägga upp certifikatlager på den lokala maskinen**

| System Store Location Name | Description | System Stores Registry Location |
|---|---|---|
| Current User | Certificate store for the currently logged-on user | HKEY_CURRENT_USER\Software\Microsoft\SystemCertificates |
| Local Machine | Certificate store for the local computer | HKEY_LOCAL_MACHINE\Software\Microsoft\SystemCertificates |
| Current service | Certificate store for the current service | HKEY_LOCAL_MACHINE\Software\Microsoft\crytografy\Services\<ServiceName>\SystemCertificates |
| Services | Certificate store for a specified service account | HKEY_LOCAL_MACHINE\Software\Microsoft\crytografy\Services\<ServiceName>\SystemCertificates |
| User | Certificate store for the users group of this computer | HKEY_USERS\<UserName>\Software\Microsoft\SystemCertificates |
| Current User Group Policy | Certificate store for the currently logged-on group | HKEY_CURRENT_USER\Software\Microsoft\SystemCertificates |
| Local Machine Group Policy | Certificate store for the local machine group policy downloaded from a network setting | HKEY_LOCAL_MACHINE\Software\Microsoft\SystemCertificates |
| Local Machine Enterprise | Certificate store for the local machine enterprise downloaded from a network setting | HKEY_LOCAL_MACHINE\Software\Microsoft\EnterpriseCertificates |

*Tabell 2. Översiktbild över hur Windows 2000 lägger upp ett certifikatlager på den lokala maskinen.*

Certifikatlagret är uppbygg på sådant sätt att varje användare, varje service, nuvarande användare, nuvarande service mm har en egen rent fysisk adress som lagrar sina egna certifikat (Se tabell 2 ovan). Detta för att säkerheten skall vara så hög som möjligt för varje certifikat.

## *6.5 SSL*

Secure Socket Layer (SSL) är ett säkerhetsprotokoll som används för att kryptera känslig information över Internet mellan klient och server. Det är idag implementerat i de flesta större webbläsare och webb servrar.

Skriver man HTTPS istället för HTTP i adressfältet på webbläsaren, så upprättas en SSL-förbindelse, dock under förutsättning att servern är konfigurerad för att upprätta en sådan. Normalt kommer också en så kallad pop-up ruta upp som meddelar att du nu kommer att kommunicera över en säker förbindelse. Accepteras detta meddelande så kommer det i nedre högra hörnet av webbläsaren visas ett låst hänglås som visar att förbindelsen är upprättad.

SSL har blivit Internet standard i dag men då under namnet Transport Layer Security (TLS), men det vanligaste är att man fortfarande benämner det SSL. Det använder sig av digitala certifikat och digital signering för att skapa en säker kommunikationskanal mellan klient och server.

När en klient ansluter till en server så utbyter de information med varandra, på så sätt ”förhandlar” man fram en gemensam krypteringsstandard. Det finns flera olika krypteringar och identifikationssätt som SSL stödjer, vilket leder till att man kan ha olika säkerhetsnivåer på olika applikationer på servern.

SSL används ofta för att skapa säkra webblösningar, men tiden det tar att ”förhandla” gör att det kan bli en långsam lösning. Om man behöver kryptera bara en del av SOAP- meddelandet är det heller inte effektivt att skydda hela kommunikationskanalen. Sammanslaget gör det att det finns andra alternativ som passar bättre ibland.

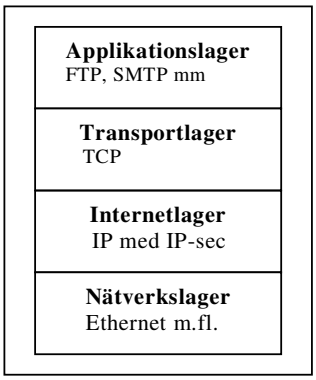
Delkryptering av meddelanden är av extra vikt för Web-services, exempelvis vid transaktioner över nätet. En kund beställer en vara hos ett företag via en Web-service. Kunden skickar med information om vilken vara han/hon vill köpa, samt information om betalsätt, som t.ex. kontonummer eller kreditkortsnummer. Företaget skickar betalningsuppgifterna vidare till kundens bank via en annan Web-service. Företaget ska inte ha möjlighet att kunna se bankens information och banken ska inte ha möjlighet att kunna se företagets information.

En end-to-end säkerhetslösning, det vill säga en lösning över flera noder, samt delkryptering, är lösningen på ovanstående problem. Detta klarar inte SSL av så man måste ta till andra tekniker.

## *6.6 VPN*

Virtual Private Network (VPN) är en metod som används för att skapa ett säkert kommunikationssätt över ett osäkert nätverk exempelvis Internet. Metoden kan också med fördel användas för att skapa en säker förbindelse inom ett Intranät. Om man ska upprätta en säker förbindelse mellan två kontor med olika adresser inom ett företag eller om en medarbetare behöver jobba på distans (hemarbete) är ett virtuellt privat nätverk en säker lösning.

VPN-tekniken kan upprätta virtuella grupper av noder på flera olika nivåer, i TCP/IP modellen (Se figur 8 nedan), inom nätverket som exempelvis nätverksnivå etc. Huvudsakligen upprättas dock ett VPN inom nätverksnivån.



*Figur 8. Visar nivåerna i TCP/IP modellen*

Den som administrativt sköter om de olika VPN- sessionerna är brandväggen, den tar emot anrop ifrån en klient som vill upprätta en säker förbindelse till nätverket, brandväggen fastställer nodens (datorn, servern etc.) identitet. När identitet är fastställd så kontrolleras vilken behörighet noden har inom nätverket mot en databas, behörigheten kan sättas såväl till hela nätverket som bara vissa noder inom nätverket. Är noden behörig så skapas en session.

Själva sessionen som skapas brukar benämnas ”tunnel, tunnling”- man tunnlar trafiken mellan två olika noder. All trafik i tunneln krypteras med digitala signaturer, certifikat, engångslösenord etc. Användarna upplever det som om de satt ihopkopplade inom samma nätverk.

Allmänt kan sägas att VPN-tekniken används för att identifiera en nod. Om man vill identifiera en användare för en applikation på applikationsnivån måste man komplettera VPN med SSL, denna kombination används ofta av Internetbanker för att erhålla en högre säkerhets nivå. [9][11]

## 6.7 Web-service Security

Det finns ingen standard för säkerhetsspecifikationer åt Web-service i dagsläget. Det som finns är Web-service Security (WS-Security) en säkerhetsspecifikation som är framtagen av IBM, VeriSign och Microsoft som blev publicerad 2002-04-05. Den gäller inte som standard i dag, men ovan nämnda företag arbetar på att få den godkänd som säkerhetsstandard för Web-service i framtiden. Tilläggas skall dock att det arbetet fortskrider väldigt långsamt.

WS-Security är inte det enda förslaget till standard som tagits fram, Man föreslog samtidigt andra lösningar: WS-Routing, WS-Coordination, WS-Inspection, WS-Referal samt WS-Transaction. Alla de här förslagen ingår i ramverket *Global Xml Web-service Architecture (GXA).*

Med WS-Security kan man överföra användarnamn/lösenord, digitala certifikat (säkerhetssymboler) på ett standardiserat sätt oberoende av vilken typ av säkerhetssymbol man använder sig av. WS-Security-standarden ska stödja en mängd olika format. Hur kryptering av kerberosbiljetter och X.509-certifikat ska

göras finns specificerat. Sammantaget gör detta att identiteten för ett meddelande kan fastställas.

Säker kommunikation kan genomföras med användning av olika säkerhetssymboler och XML- kryptering av meddelandet. Delkryptering av meddelanden stöds också som standard av WS-Security.

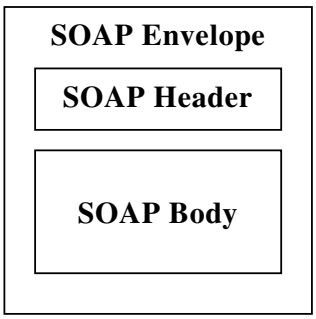
## 6.8 SOAP

Simple Object Access Protocol (SOAP) bygger på XML. Denna teknik gör det möjligt för olika program att anropa varandra på ett löst sammansatt standardsätt, vilket i sin tur gör det möjligt att bygga program som kan distribueras över Internet. SOAP har stöd av i stort sett alla större programvaruföretag som t.ex. Microsoft, IBM, HP, Sun och Oracle. [20]

SOAP protokollet består av fyra delar:

- Ett kuvert (Envelope) som definierar ett ramverk för att beskriva vad som finns i ett meddelande och hur det ska processas.
- Ett transportramverk för utbyte av meddelanden på ett underliggande protokoll.
- En uppsättning regler för hur instanser av applikationsdefinierade datatyper skall uttryckas.
- En konvention för att representera fjärrfunktionsanrop (Remote Procedure Calls) och svar.

Ett typiskt SOAP meddelande består av ett SOAP Envelope, SOAP Header och en SOAP Body (Se figur 9 nedan).




*Figur 9. Innehållande SOAP Envelope, SOAP Header och SOAP Body*

SOAP Envelope specificeras som ett rotelement och är giltigt genom att använda rätt namespace (namnrymd). Namnrymder används för att bland annat återanvända XML kod.

SOAP:s headerelement används för att tillhandahålla information som är besläktad med vad som finns i SOAP:s bodyelement. Informationen som placeras där behöver inte den slutgiltiga användaren veta något om. Exempel på användningsområden för SOAP headern är betalning, inloggning och hantering av transaktioner. Headerelementet måste placeras direkt under envelopeelementet.

Bodyelementet måste finnas med i ett SOAP meddelande, till skillnad från header elementet. Bodyelementet innehåller det data som skall skickas och utgör därmed den större delen av meddelandet. Bodyelementet innehåller den data som är specifik för det aktuella metodanropet, som metodnamn, inparametrar och utparametrar vilka skickas till XML Web-servicen. Web-servicen använder sedan bodyelementet för att skicka tillbaks data till klienten. Bodyelementet kan också innehålla information om olika fel som kan ha inträffat, vilka skickas tillbaka av Web-servicen. [1]

## 6.9 WSSAL

En Web-service är utvecklad för att snabbt kunna implementeras och användas för kommunikation internt och externt. Kommunikationen kan ske med redan kända användare eller med helt nya. För att stödja detta behöver säkerheten kunna implementeras snabbt, vara flexibel och vara lätthanterlig vid förändringar. Detta uppnås genom att implementera Web-service Security Abstraction Layer, förkortat WSSAL, som Portwise utvecklat.

WSSAL består av flera delar. Den första delen är en Web-service Proxy, förkortat WSP. Den fungerar ungefär som en grindvakt, som bara släpper igenom den trafik som uppfyller den policy som är uppsatt hos Web-servicen. Policyn består av en uppsättning regler som kräver identifikation av användaren, även om en tillåten kanal används. WSP samarbetar med den andra delen som är Web-service Security Platform Layer, förkortat WSSPL. Samarbetet gör att WSP får hjälp med saker som rättigheter, identifikation och engångslösenord mm.

Kraven på WSSAL är att den måste kunna hantera extern kommunikation som kommer in och att den måste kunna hantera intern kommunikation som går ut. Ett exempel på detta är att WSSAL ska kunna hantera SOAP- meddelanden som kommer in och även kunna säkra utgående svar.

## 6.10 Brandväggar

Allt fler företag ansluter sig till Internet, därmed ökar risken att någon olovligen tar sig in i företagets nätverk. De vanligaste hoten som ett företag kan råka ut för är att någon tar sig in i nätverket eller servrarna för att lagra olaglig data, tar sig in för att använda företagets mailserver för att skicka ut tusentals reklam e-post eller tar sig in för att använda företagets servrar för riktade attacker mot andra företag. Det finns programvaror som kan genomsöka miljontals IP-adresser per timme på sin jakt att hitta en oskyddad dator, ett nätverk eller en mailserver.

När man har ett nätverk med internetanslutning brukar man börja med det så kallade skalskyddet, det vill säga brandväggen. En brandväggs uppgift är att, till exempel, isolera två nätverk ifrån varandra för att inte få in oönskade saker i nätverket. Detta är den vanligaste uppgiften som en brandvägg utför.

En brandvägg har två extrema lägen, helt stängd eller helt öppen. Om den är helt stängd fungerar den som en stenvägg, alltså helt säker vilket dock medför att användarna innanför brandväggen inte kan utföra sina arbetsuppgifter. Men om

den är helt öppen blir den som en del av Internet, det vill säga att vem som helst kan komma åt vårt nätverk. Hemligheten ligger i att öppna brandväggen exakt så mycket som är nödvändigt. Säkerheten blir då optimal, det vill säga att det nödvändigaste kan göras utan att onödiga säkerhetsrisker behöver tas.

En brandvägg fungerar åt båda håll, det vill säga att den även kan användas för att kontrollera personal i det egna företaget. Brandväggen styr vad som får och inte får göras och mellan vilka avsändare och mottagare olika tjänster får användas. Genom att låta alla datorer kommunicera med
Internet via brandväggen, är det bara brandväggen som är direkt ansluten till Internet. Det räcker då att göra brandväggsdatorn så svårhackad som möjligt.

Om datorer ute på Internet skall kunna kommunicera med datorer i företagets nätverk måste regler sättas upp i brandväggen. Detta för att bibehålla så hög säkerhet som möjligt. Samma sak gäller vid omvänt fall.

En brandvägg kan även användas för att logga allt som händer mellan Internet och det interna nätet. Om man vill spionera på användarna passar alltså en brandvägg utmärkt i detta syfte. Möjlighet finns alltså till att övervaka och kontrollera allt som användarna gör. Men spionage på de anställda är kanske inte allt som företagsledningen vill göra, brandväggen passar utmärkt för att mäta vad företagets resurser verkligen används till.

En viktig funktion en brandvägg har är att hålla reda på IP-adresser. IP-adressen är den unika adress som alla datorer som är anslutna till Internet har. Denna adress används för att hålla reda på avsändare och mottagare när två datorer kommunicerar med varandra.

Det finns tre typer av brandväggar. Typen paketfiltrerande router, typen Application Gateways (Proxy) eller typen Reverse proxy. [9] [10]

### 6.10.1 Paketfiltrerande Router

En brandvägg av den här typen använder en router och en uppsättning regler för vilka paket som skall släppas igenom och vilka som skall stoppas. Dessa regler baseras på avsändarens adress, mottagarens adress och port. Den här typen av brandvägg ger relativt låg säkerhet, men kostnaden blir låg och det är sällan som det uppstår problem. [10]

### 6.10.2 Application Gateways (Proxy)

En brandvägg av den här typen använder serverprogram, så kallade proxies, som körs på brandväggen. Proxies tar emot en utifrån kommande begäran, undersöker den och vidarebefordrar en godkänd begäran till en Internet ansluten dator som kan ge den efterfrågade tjänsten. Om högsta möjliga säkerhet krävs, bör all Internettrafik gå igenom en Applicaton Gateway.

Man kan säga att en paketfiltrerande router bara kollar uppkopplingen medan en proxy dessutom kollar vad uppkopplingen används till.[10]

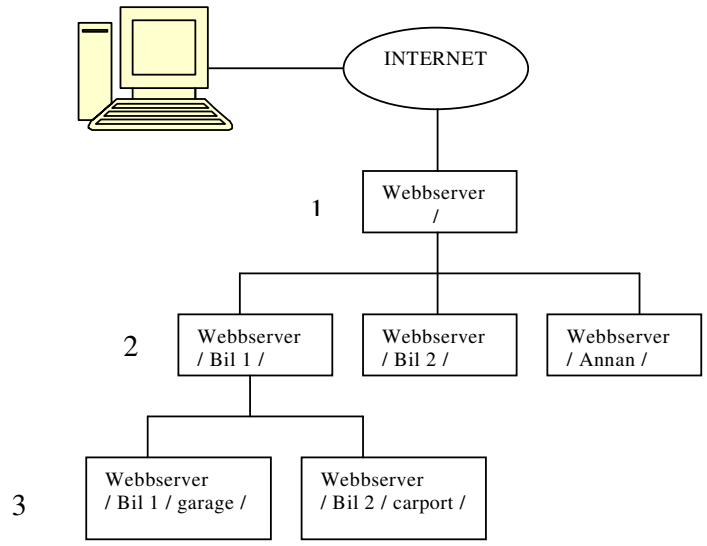
### 6.10.3 Reverse proxy

En reverse proxy möjliggör flera skyddsfunktioner, till exempel att den framförliggande proxyn tar första smällen istället för den riktiga webbservern vid en attack. Det vill säga att den gömmer den bakomliggande nätstrukturen.

En reverse proxy server används för att få flera webbservrar att se ut som en. De gömda webbservrarna blir då en undernivå i den första webbserverns filträd (Se figur 10 nedan).

I bildexemplet visas sex stycken webbservrar, men bara nummer ett har ett publikt IP-nummer och kan nås över Internet, medan de andra har privata nummer. Genom att använda reverse proxy i de webbservrar som har en annan under sig, det vill säga nummer ett och två, blir den undre webbservern en del i hans eget filträd.

Detta betyder att webbservernivå två syns som en mapp i den första webbserverns filträd och den tredje webbservernivån syns som en mapp i den andra webbservers filträd. För att nå webbserver tre går man till webbserver ett och går igenom dess filträd tills man kommit till webbserver tre. Webbserver tre verkar nu vara en del av webbserver ett, det vill säga att användaren kommer att tro att de sex webbservrarna är en enda webbserver.




*Figur 10. Visar hur filträdsstrukturen kan se ut vid användandet av en reverse proxy.*

En reverse proxy har en funktion som kallas caching. Det innebär att man med reverse proxy caching kan få användaren att tro att cachen som befinner sig nära originalservern är webbservern.

En reverse proxy gör det möjligt för användare utifrån att komma åt Internet servrar som ligger lokalt.[16] [17]

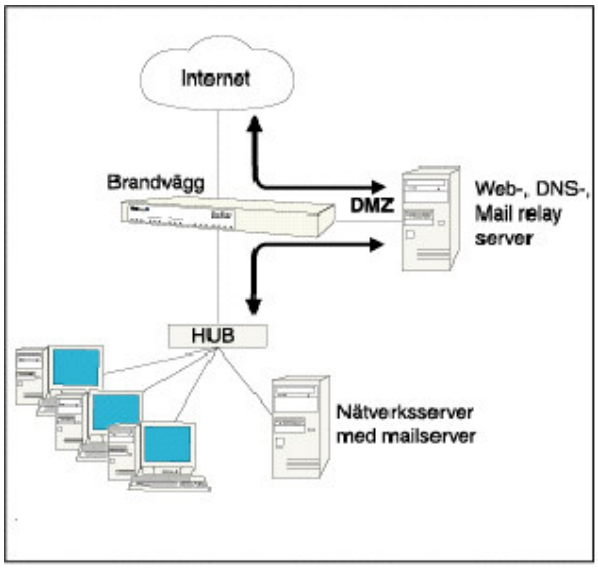
### 6.10.4 DMZ

I realiteten används brandväggar inte bara mellan intranät och Internet. De används också emellan interna nät, mellan servicenät och Internet och mellan interna nät och servicenät. Ett servicenät är ett nät där vi lägger services som allmänheten skall komma åt, som till exempel filhämtning via FTP. Ett annat namn för servicenät är DeMilitarized Zone, demilitariserad zon eller förkortat DMZ.

Med detta menas att det så kallade servicenätet ligger i ett ingenmansland mellan det interna nätet och Internet (Se figur 11 nedan). DMZ skyddas av brandväggen men tillåter oftast flera olika sorters trafik utifrån.

I och med att DMZ ligger på en extra nätverksutgång, det vill säga att den ger en extra nätverksdel skild från det vanliga nätverket, så görs nätverket säkrare genom att man begränsar möjligheterna att ta sig in i nätverket, även om någon lyckas hacka sig in i webbservern.

Vitsen med att ha ett DMZ-nät är att man kan ha olika regler beroende på om trafiken kommer inifrån eller utifrån. En webbserver, till exempel, skall ju kunna uppdateras inifrån det interna nätet men absolut inte utifrån Internet. [23]




*Figur 11. Visar hur DMZ nätet är åtskiljt från det övriga nätet.*[23]

## *6.11 Portwise*

Portwise AB, f.d. Lemonplanet, är ett svenskt IT-säkerhetsföretag som utvecklar och säljer lösningar för säker access, autentisering och auktorisation. Portwise tillhandahåller två produktgrupper. Portwise mVPN och Portwise mID. Portwise mVPN ger en säker access till interna system genom SSL-baserade VPN och

Portwise mID erbjuder säkra inloggningsfunktioner, där användaren ges ett specifikt engångslösenord vid varje accesstillfälle.[15]

### 6.11.1 mVPN

Portwise mobile Virtual Private Network (mVPN) är ett SSL-baserat VPN. Fördelen med SSL-baserade VPN är att kunderna inte behöver installera en klient på alla sina pc, eftersom det finns SSL stöd i vanliga webbläsare. En användare kan till exempel gå in på ett internetkafé och på ett säkert sätt koppla upp sig mot sitt företagsnät och läsa e-post. Detta endast med hjälp av en internetuppkoppling och en webbläsare.

Portwise mVPN gör att en användare har möjlighet att passera routrar, brandväggar och proxyservrar med en hög säkerhetsnivå. Portwise mVPN ger dessutom tillgång till applikationer som e-post och ftp med mera. [14]

### 6.11.2 mID

Portwise mobile Identification (mID), är en server som gör användarkontroller. Beroende av säkerhetspolicyn och inblandade parter, kan Portwise erbjuda fyra olika metoder för kontroll av användare, dessutom finns alla metoder i en och samma server.

Alla metoderna är baserade på så kallade One-Time Passwords, förkortat OTP, och kan fås genom en installerad mjukvarudosa eller genom SMS. Detta erbjuds till en förhållandevis låg kostnad. [14]

## *6.12 Kerberos*

Kerberos är en metod för verifiering som ursprungligen togs fram för att förbättra säkerheten i UNIX-system och som tillämpar principerna med kryptering för att lösa problemet med användarverifiering.

Verifiering med kerberos bygger på kortlivade ”biljetter” eller verifieringsnycklar som bevisar att du är den du säger att du är, men har så kort livslängd att när en hacker kommer över dem är de redan föråldrade och värdelösa. Därigenom försvinner den praktiska möjligheten att stjäla lösenord och uppträda under falskt namn. [8]

Varje gång användaren vill göra något som karaktäriseras som en tjänst behöver han en korrekt biljett. En tjänst kan vara att få komma åt filer, hämta post från postdatorn, skicka meddelanden eller använda en annan dator. Varje sådan tjänst kräver en enskild biljett. Om användaren saknar rätt sorts biljett, och har rätt att utnyttja tjänsten, kan han få en sådan biljett mot uppvisande av den första biljetten, ticket-granting-ticket. Detta är något som kerberos tar hand om helt och hållet och som inte användaren märker någonting av. Har användaren en korrekt biljett används denna.[25]

## 6.13 Lastbalansering

Enligt SSAB:s standard skall maskiner som driver och innehåller webbsidor särskiljas från de som innehåller Web servicar med åtföljande COM+ komponenter. Allt för att få en så skalbar miljö som möjligt.

Ett typfall för SSAB:s affärslösningar är att man önskar hålla reda på vem som anropar en webbsida och dess tillhörande resurser i form av Web servicar och komponenter. Anledningen kan vara att man önskar styra funktionalitet utifrån roller som var och en har tilldelats. Denna metod att hålla reda på vem som äger frågan i en resurs kallas personifiering.

När en webbsida befinner sig i samma maskin som övriga resurser är detta inte ett problem. Då finns all information som behövs inom maskingränsen. Men för SSAB:s del så finns det ett behov att kunna skala ut en maskinpark så att rätt prestanda erhålles beroende på uttaget utnyttjande och antal användare av resurser.

Detta gör att SSAB:s arkitektur strävar mot att separera webbsidorna och affärsskikten på separata maskiner.

I och med att processgränserna utåt sett från dessa maskiner kommer att vara webbifierade (ASP.NET och Web-service) så kan flera maskiner via lastbalansering se till att rätt prestanda erhålles.

## 6.14 COM/COM+

### 6.14.1 COM

COM är en teknik som handlar om hur klienter och ett COM-objekt talar med varandra på en binär nivå.

En COM-komponent kan ses som en svart låda, du behöver inte veta hur den fungerar, utan kan anropa dess funktioner via ett eller flera gränssnitt. Sedan byggs den in på ett sådant sätt att klienten aldrig märker att han använder ett COM-objekt. Tex. när du använder dig av ett Microsoft Office-makro så använder du ett COM-objekt. IIS (Internet Information Server) är till stor del uppbyggd med COM. I stort sett varenda Microsoft produkt använder sig av COM.

COM är helt objektbaserat, vilket gör att du kan skriva kod för en komponent med de flesta programspråk. Klienterna som i sin tur anropar komponenten kan skriva sin kod i det programspråk som de föredrar.

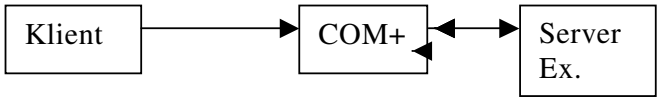
Tekniken möjliggör platsoberoende åtkomst mellan klienten och de objekt som används. Komponenten och klienten kan även köras i olika tillämpningar (processer) eller i olika datorer. COM tillhandahåller tekniken för att överbygga stora avstånd. Du kan alltså använda en komponent på samma sätt oavsett avståndet till var den är.

Som utvecklare behöver du bara veta hur du ska använda komponenten. Exakt hur den är byggd eller fungerar är inget du behöver veta för att den ska fungera, utan det är bara och anropa den. Ska du där emot bygga en komponent ställs ju självfallet helt andra krav.

COM har tre grundläggande delar:

- En binär specifikation
- Ett körnings bibliotek
- Tjänster

Ungefär så här fungerar det med COM-komponenten som ett gränssnitt emot exempelvis en server (Se figur 12 nedan):




*Figur 12. Visar hur det fungerar med COM-komponenten som ett gränssnitt mot en server.*)

### 6.14.2 COM+

COM+ är en vidareutveckling av COM som tillhandahåller en rad tjänster som inte vanlig COM har.

Några av de viktigare tjänsterna som tillkommit är:

- Transaktionsstöd mellan komponenter
- Säkerhetsmodell
- Processisolering

En COM/COM+ komponent passar alldeles utmärkt att använda när man ska upprätta en kommunikationskanal mellan två objekt på en binär nivå.

## *6.15 Network service*

I tidigare versioner av IIS har arbetsprocesser körts under Local System kontot. Problemet är att Local System har tillgång och rättigheter till nästan alla resurser i operativsystemet och det har resulterat i omfattande säkerhetsproblem för Microsoft.

Så ifrån IIS 6.0 har man alla arbetsprocesser samlade i en ”default application pool identity”, Network Service. Du kan även själv konfigurera vilket konto ”the application pool” processerna ska köras under. Det finns tre fördefinierade

konton, Network Service, Local Service och Local System, att välja på. Det finns även möjlighet att skapa ett eget konto att köra arbetsprocesserna under.

Man kan jämföra Network Service med det gamla IUSR_MachineName kontot som finns i tidigare versioner av IIS, men det är mycket säkrare.

Problemet med Network Service kontot är att det inte fungerar över maskingränserna. Om man ska köra en ASP-process över två maskiner krävs det åtskilliga konfigureringar för att få det att stämma säkerhetsmässigt. Detta eftersom Network Service är ett lokalt maskinkonto.

### 6.15.1 IIS_WPG

IIS_WPG är en användargrupp som finns fördefinierad i IIS 6.0, det tillhandahåller ett minimum av användarrättigheter i olika mappar för att kunna köra en webbapplikation. Network Service, Local Service och Local System är i standardkonfigureringen medlemmar i gruppen.

Det som är intressant med IIS_WPG är att man kan skapa egna konton som man lägger till gruppen. Sedan är det bara att sätta rättigheter till IIS_WPG. Det leder till att man kan bygga över problemen med att synkronisera två ASP-processer när de körs på två olika maskiner.

Det ligger alltså både rena tjänstekonton som ASP.NET samt, att man kan skapa, egna personifierade konton man lägger till IIS_WPG. Detta underlättar konfigureringen för utvecklare och administrativ personal.

## *6.16 Förvaltningsmodell*

En förvaltningsmodell beskriver hur ett system förvaltas. Den kan innefatta ett antal aktiviteter. Exempel på aktiviteter är:

- Ändringar
- Drift
- Ansvarsroller
- Rutiner
- Grupper
- Utbildning

Det är viktigt att alla system har en så bra förvaltningsmodell som möjligt. Detta för att öka livslängden på systemet, kunna hålla systemet i drift, ha en bra felhantering och ändringshantering mm.

# 7. Diskussion

*I dagens läge när tekniken med Web-service börjar användas externt i allt större omfattning av företag, ställs det krav på en säker kommunikation mellan webbklient och Web-service. Den kommunikationen ska även kunna passera igenom företagens olika säkerhetslösningar, på ett sådant sätt att den verifierar sig själv automatiskt. Detta för att kunna få ett snabbt och smidigt system som helst ska klara sig ifrån manuell handpåläggning för att kunna fungera.*

*Hur man sen driver och förvaltar en extern Web-service är också av intresse för flera företag. Vilket arbetsgruppen även gjort en modell på.*

## *7.1 Tekniker*

Att konstruera och bygga en Web-service och en webbklient gick bra. Bägge applikationerna byggdes i Visual Studio.NET med programmeringsspråken VB.NET och C#. Anledningen till att arbetsgruppen använde två olika programmeringsspråk var av testsyfte. Författarna kunde inte märka någon avgörande skillnad i vare sig prestanda eller funktion mellan de olika språken. Den slutgiltiga varianten skrevs dock i C#.

**Säkerhetstekniker**
När författarna började titta på dom olika säkerhetstekniker som finns, för att kunna sätta ihop en säkerhetslösning åt SSAB, uppstod det problem. Ingen av dom tekniker som skulle användas hade författarna kunskap om, så en instuderings period inleddes. Teknikerna var inte allt för komplicerade att vare sig förstå eller använda sig av visade det sig. Detta eftersom det finns mycket skrivet om det, både på Internet och i litteraturen. Ett problem var dock att mycket av tekniken med certifikat inte var anpassad för en Web-service - Webb klient lösning, men detta lyckades arbetsgruppen till slut pussla ihop.

**Nyheter i W2k3**
Windows 2003 Application Server (W2k3) är en ny serverteknik från Windows. Den skiljer sig på flera punkter ifrån tidigare varianter av som Windows NT och Windows 2000 (W2k). Det som arbetsgruppen hade mest problem med var standard konfigureringen där man manuellt måste gå ut och tilldela rättigheter till användare och processer. När arbetsgruppen dessutom var tvungna att sätta sig in i diverse nyheter som införts i W2k3 som t.ex. Network Service, IIS_WPG och ”default application pool”, kändes det som att lösningen på problemen var långt borta. Generellt kändes det som att arbetsgruppen överhuvudtaget inte hade någon kunskap ämnet.

**Svårt att hitta information**
Att få tag i litteratur till W2k3 var inte det lättaste, och när man väl fick tag i något stod det sällan någonsin något i klartext. Detta ledde till att arbetsgruppen var tvungna att läsa in stora mängder information. Svårigheten var att man bara hittade en bit information här och där, varpå man skulle sätta ihop alla de bitarna och få en fungerande lösning. Samma sak gällde med att söka information på Internet om W2k3, det fanns mycket beskrivet men dock väldigt utspritt där också. Författarna hade dock förmånen att jobba på SSAB:s IT-avdelning , där

viss hjälp med specifika problem kunde erhållas. Men även personalen på SSAB ansåg det vara krångligt att hitta väsentlig information om ämnet. Allt detta kan bero på att W2k3 är en ny produkt som inte satt sig på marknaden än och som inte fått sina barnsjukdomar bortrensade.

**Fler tekniker**
ASP.NET processerna och IIS-processerna var också tekniker som arbetsgruppen inte hade en aning om hur de fungerade. När dessutom mycket hade gjorts om, av säkerhets skäl, i W2k3 ställde det till problem för felsökningen på webb-applikationerna. Att man sen kan personifiera vissa processer för att uppnå resultat, i arbetsgruppens fall måste personifiera processerna, var någonting som arbetsgruppen var lyckligt ovetande om. Det är dessutom saker som i många avseenden skulle varit med redan i designfasen. Med facit i hand blev arbetsgruppens applikationer totalt ombyggda jämfört med den design författarna kom fram till i början av byggandet.

COM+ komponenter var också okänd teknik för arbetsgruppen. Där lyckades dock arbetsgruppen bättre med införskaffandet av kunskap i ämnet. Det ska tilläggas att den tekniken till stor del var vanlig kodning.

Certifikat i sig är inte svårt att lära sig använda, men kopplar man ihop de med W2k3, arbetsprocesser och Web-service tekniken uppstår det problem med på vilket sätt som man ska få de att samverka.

**Sammanfattning om tekniker**
Sammanfattningsvis kan arbetsgruppen säga att den teknik som skolan gett undervisning i inte förorsakade några problem. En del annan teknik som SSL, VPN och certifikat var inte svårare än att det gick att läsa in. Men allt som handlar om processer, personifiering, W2k3 och rättigheter för åtkomst på mappar var för arbetsgruppen helt okänd mark.

För mycket tid lades ner i designfasen med att rita i hop en ”tjusig” design, som sedan kompletterades med diverse tekniker som författarna trodde skulle samverka utan problem. De problem som uppstod var dels svårlösta men framför allt hade författarna ingen aning om vad det var som var fel. Vilket gjorde felsökningen nästintill omöjlig. Hade inte SSAB bistått med handledning för att komma vidare hade nog projektet havererat här. Men med nya infallsvinklar och ny design av applikationerna så fungerade till slut allt som avsett.

## *7.2 Design*

Designen som gjordes är av nödvändighet helt anpassad till SSAB:s IT-arkitektur. Men det hade arbetsgruppen inga större problem med. De problem som möjligen fanns var att det krävdes en hel del inläsning i det här stadiet.

När sen en arkitektur gjordes för hur säkerhetslösningen på applikationsnivån skulle se ut, lades alltför stor möda på att få den logiskt riktig snarare än att försöka få ihop en fungerande arkitektur. När sedan den teknik som användes för säkerhetslösningen inte fungerade ihop över domängränser, utan någon form av

handpåläggning, uppstod det problem för författarna. De teoretiska referensramar som införskaffats under tre års högskolestudier räckte inte till för att kunna göra en så omfattande design.

När det dessutom visade sig att vissa processer måste personifieras (det vill säga processen körs under en användare, och får automatiskt samma rättigheter som användaren) för att fungera. Fick en omdesign lov att göras för att uppnå den funktionalitet som krävdes gentemot SSAB:s krav på säkerhet och funktion.

**Sammanfattning design**
Det visade sig vara mycket vara mycket mer att göra, än att bara få ihop en snygg och logiskt riktig design för att få allt att fungera. När man kommer ut i verkligheten är den mycket komplexare än vad man föreställt sig i skolan.

Hade inte SSAB ställt upp med handledning i form av egna resurser så väl som externa konsulter, hade nog inte omdesignen av applikationerna lyckats, på grund av att arbetsgruppens teoretiska referensramar inte räckte till för att få en så omfattande design att fungera ihop med den valda tekniken över två domäner.

Resan har blivit lång och lärorik för författarna, det räcker inte med att dra några streck på ett papper för att få en fungerande design, man måste veta om teknikens begränsningar också.

## 7.3 Metod

Den metod arbetsgruppen använt sig av är en metod som två studenter som genomförde sitt examensarbete före oss, på SSAB, utvecklade. Metoden har arbetsgruppen verkligen haft stöd av vid genomförandet av examensarbetet. De problem som arbetsgruppen stött på, beror mer på teknisk okunskap än av att metoden varit bristfällig. Faserna i metoden är klara och genomtänkta och varje steg i den enskilda fasen driver utvecklingsarbetet framåt. Det är tydliga gränser mellan varje steg och den skapar inte mer dokument än nödvändigt.

## 7.4 Förvaltning

*Här hålls en diskussion över varför vissa beslut tagits under framtagandet av Web-service- modellen.*

Det första steget som arbetsgruppen tog, vid framtagandet av Web-service-modellen, var att intervjua berörd personal. Det visade sig, efter intervjuerna, att SSAB:s säkerhetslösningar förvaltas under en modell som kallas Portwise processen och att Web-service-modellen skulle ligga i den arkitekturen.

När vi tog fram roller och rutiner så har Portwise processen följts i väldigt stor utsträckning. Vi hade naturligtvis studerat vissa författares syn på dessa begrepp, bland annat Nordström och Welanders, men deras syn skiljde sig inte nämnvärt från de uppgifter vi fick ifrån Portwise processen.

I Portwise processen fanns inte någon information över förvaltningsgrupper. Därför valde författarna att studera vad, framförallt Nordström och Welander, tycker om dessa. De tycker att de grupper som måste finnas med är styrgrupp och förvaltningsgrupp. Men eftersom endast ett fåtal personer behövs för att kunna förvalta Web-service-lösningen, ansåg arbetsgruppen att det skulle räcka med att skapa en förvaltningsgrupp där alla berörda ska ingå. Gällande möten i förvaltningsgruppen, anser Nordström och Welander att de skall hållas 2 ggr/månad. Författarna anser dock att det räcker att hålla möten när ett behov av möte uppstår, t.ex. som när en felanmälan kommer in.

När författarna undersökte affärsmässigheten i Portwise processen hittades en ganska intressant företeelse. Helpdesk var nämligen motpart till portwise-förvaltarna, dvs. IT-avdelningen blev motpart till sig själv (Se tabell 1 Kap 5.1.2 Roller). Men det här systemet fungerar idag bra och SSAB upplever inte några problem med den rollstrukturen. Med tanke på detta och med tanke på att arbetsgruppens förvaltningsmodell är mindre än Portwise processen, fanns det ingen anledning att göra några ändringar gentemot den.

Genom att upprätta avtal uppnår SSAB en högre affärsmässighet i förvaltningen. I Portwise processen finns inte några klara avtal, därför anser arbetsgruppen att SSAB ska inse vikten av detta och börja upprätta avtal i en större omfattning än idag. För Web-service-modellen tyckte arbetsgruppen att det skulle vara tillräckligt med att upprätta ett affärsavtal mellan systemägaren och IT-chefen, detta för att förvaltningen är så pass liten.

Möjligen skulle ett avtal kunna upprättas mellan helpdesk och Web-service-förvaltarna på operativ nivå för att klargöra avtal mellan parterna. Detta för att IT-avdelningen blir motpart till sig själv.

## *7.5 Avgränsningar*

Nu när examensarbetet har analyserats, kan arbetsgruppen till sin stora glädje konstatera att de avgränsningar och tidsramar som sattes upp när arbetet inleddes i mitten av Januari inte behövde ändras. Arbetet med att färdigställa projektet slutfördes i och för sig inte förrän näst sista dagen.
Arbetet har inte varit för stort utan den tidspress som upplevdes mot slutet berodde mer på att tekniken som valts inte fungerade utan handpåläggning utanför maskingränserna.

# 8. Slutsats

Det har varit ett intressant arbete med att få flera olika trådar att utmynna i en principiell lösning som sedan ska kunna implementeras i olika miljöer. Generellt kan sägas att den lösning författarna kommit fram till skulle kunna implementeras av de flesta företag, vissa delar är dock specifika för SSAB:s IT-arkitektur, men dessa bitar har lämnats utanför examensarbete i mesta möjliga mån av säkerhetsskäl.

## *8.1 Resultat*

Lösningen som arbetsgruppen tagit fram åt SSAB för hur en framtida lösning på en externa Web-service lösning ska se ut, är en lösning i från ax till limpa. Den sträcker sig ifrån anpassning till befintlig IT-arkitektur, design av den tekniska lösningen, ner till kodning av enskilda applikationer. Dessutom togs en modell fram för hur man ska förvalta en sådan tjänst.

Vad som gjordes först var att studera SSAB:s IT-arkitektur, detta för att lösning måste av naturliga skäl vara kompatibel med densamma. Eftersom det inte fanns en extern Web-service förut, fanns det heller plan på hur man skulle administrera en sådan med tanke på inloggning, åtkomlighet, säkerhetsnivå och den rent tekniska lösningen.

Ett administrativt verktyg WSSAL som tillhandahölls av Portwise, företaget som byggt SSAB:s säkerhetslösning, visade sig tillhanda hålla en central punkt, där alla externa Web-service-lösningar kan administreras.

Det beslutades i samråd med SSAB att en extern säkerhetslösning skulle byggas och implementeras i WSSAL, så att lösningen skulle vara kompatibel med SSAB:s IT-arkitektur.

Den design som gjordes för den tekniska lösningen, är gjord så att vi separerat logik och kommunikation/säkerhet ifrån varandra för att kunna ha en skalbar lösning. Det ger en möjligheterna att bygga ut både applikationerna eller säkerheten var för sig, utan att den andra berörs. Detta innebär att SSAB kan bygga ut tjänsten i framtiden för en framtida transaktionshantering, utan att behöva ändra på säkerhetslösningen. Med tanke på att det inte finns en standard för hur transaktioner ska ske via Web-service, är det viktigt att ha en lösning som gör det möjligt att bara byta ut ett lager den dagen man ska börja med transaktioner.

Vid användande av klientcertifikat får man också en säker och automatisk inloggning mot SSAB:s intranät, och eftersom man kan mappa varje certifikat mot ett befintligt användarkonto hos SSAB. Så det blir ingen extra administration som behöver göras för externa användare av en Web-service hos SSAB.

Att det en COM+ komponent måste användas för att kunna skicka med certifikatets privata nycklar känns designmässigt klumpigt. I många avseenden är en COM+ komponent och en Web-service väldigt lika varandra, funktionalitet är liknande dom två emellan. Den avgörande skillnaden är i vårt fall att en

användarprofil kan laddas in i en COM+ komponenten, vilket leder till att dom privata nycklarna kan skickas med för att skapa en krypterad förbindelse.

Ur användarsynpunkt märks ingen som helst skillnad när man använder en COM+ komponent i kombination med webbapplikationer. Den påverkar inte funktionaliteten för Web- service-lösningen på något sätt, enär dess enda funktion är att skapa en krypterad förbindelse.

Arbetet blev till sist klart och det är en väl genomtänkt lösning på en extern Web-service-lösning som följer SSAB:s IT-arkitektur. Fokus i arbetet kom dock att ligga på den tekniska lösningen. Anledningen till det var problemen med att kunna skicka med de privata nycklarna.

En förvaltnings modell skapades också, med SSAB:s förvaltningsplan som förebild. Den skiljer sig på några punkter i från SSAB:s egen förvaltning, dock är det inga stora avsteg från den.

En förvaltningsgrupp bör skapas där alla berörda personer ska ingå. Möten behöver dock endast hållas när ett behov uppstår, som t.ex. när en felanmälan kommer in. Avtal bör upprättas i en större utsträckning än idag. Detta för att kunna uppnå en högre affärsmässighet i organisationen. För Web-service-modellen räcker det om ett avtal upprättas mellan systemägaren och IT-chefen, detta för att förvaltningen är så pass liten.

## *8.2 Metod*

Metoden som användes är en egenutvecklad metod, av två studenter som gjorde sitt examensarbete på SSAB före oss har gjort. Den är uppdelad i tre olika faser, och varje fas har en klar och tydlig avgränsning.

Generellt har författarna inte haft några problem med metoden, utan arbetet har framskridit på ett naturlig sätt. Alla steg i metoden driver arbetet framåt utan att man på något vis känner sig styrd av metoden. Man har aldrig någonsin fått ägna tid åt att jobba med själva metoden, utan den har funnits där som ett stöd hela tiden.

Problemen som uppstått berodde inte på att metoden var bristfällig. De problem som dök upp låg helt enkelt utanför de teoretiska referensramar som författarna hade. Detta föranledde åtgärden att bryta ut en del av fas 3 och skapa ett eget kapitel för den (Kapitel 4).

Dokumentationen metoden genererar är heller inte överdriven på något vis, utan bara de dokument som behövs finns med.

## *8.3 Framtid*

Att kunna skapa en webbapplikation som klarar av att hämta de privata nycklarna till certifikatet, utan att ha en COM+ komponent som gränssnitt, borde vara genomförbart. Författarna anser att det borde vara möjligt att lösa det problemet

utan att använda sig av en COM+ komponent, det handlar om att ge rätt arbetsprocess (exempelvis Network Service) rättigheter i rätt mapp.

En annan lösning kan vara att lyckas konfigurera W2k3, i dagsläget känns det som om problemet kan ha sin lösning där. Detta på grund av att i en normal installation är allt förbjudet som grundinställning för säkerheten. Under arbetets gång upptäcktes många säkerhetskonfigureringar som man manuellt fick ställa om för att applikationer skulle fungera som avsett. Allt detta är medvetet gjort av Microsoft för att höja säkerheten för användarna.

Sammanfattningsvis fungerar W2k3 på sådant sätt att användaren/administratören för Intranätet manuellt ska konfigurera vad som är tillåtet och inte tillåtet, detta av säkerhetsskäl. Arbetsgruppen ansåg att konfigureringen här gjorde att man inte kunde hämta de privata nycklarna.

# Källförteckning

## *Litteratur*


1. Ejven Bill (2002) *XML Web Services for ASP.NET.* ISBN:0-7645- 4829-

2. Short Scott (2002) *Building XML Web services for the Microsoft .NET platform.* ISBN:0-7356-1406-7

3. Homer Alex, Sussman Dave, Francis Brian (2000) *Active server Pages 3.0 för professionella.* ISBN:91-636-0610-0

4. Nordström, M. & Welander T.(2002), *Affärsmässig förvaltningsstyrning*, Studentlitteratur
   ISBN 91-888-62-13-5

5. Bergvall, M. & Welander, T. (1996), *Affärsmässig systemförvaltning*, Studentlitteratur
   ISBN 91-888-62-01-1

6. Brandt, P (1998), *Systemförvaltningshandboken 1998*, Nacka, Itligence

7. Bättre systemförvaltning. *Hur man undviker de vanligaste fällorna.* (1997). Stockholm. Riksrevisionsverket.


## *Internet*


8. Pagina. Sökord: kerberos. Hämtad från <http://pagina.se/itord/default.asp?SokOrd=kerberos>

9. IT-säkerhet. Sökord: brandväggar +vpn. Hämtad från < http://www.nextcom.se/_brandvagg-info.html>

10. Brandväggar. Sökord: brandvägg. Hämtad från < http://www.atremo.se/isite-cache/filer/130/Introduktion_Brandv%E4ggar.pdf>

11. Encyclopedia Intranetica. Sökord: vpn. Hämtad från < http://www.intranetica.com/intranetica/vpn/>

12. CIW Internet System Management. Sökord: reverse +proxy. Hämtad från< http://www.linuxkurser.nu/system.pdf>

13. An introduction to Web services. Sökord: webservice. Hämtad från < http://www.ariadne.ac.uk/issue29/gardner/>

14. Networks Telecom. Sökord: portwise. Hämtad från

<http://networks.stofair.se/common/stand/aboutshow.asp?StandId=4528&ProductID=0>

15. Portwise. Sökord: portwise. Hämtad från < http://www.certezza.net/pdf/mVPN%20product%20sheet.pdf>

16. Att skydda ett nätverk från intrång från obehöriga via Internet. Sökord: firewalls +proxy. Hämtad från < http://www2.du.se/kurser/2097697692/dokument/Firewalls%20vt-02%20för%20Sy2.pdf?iKursid=2097697692>

17. Chapter 7 Reverse proxy. Sökord: reverse +proxy. Hämtad från <http://developer.netscape.com/docs/manuals/proxy/adminux/revpxy.htm>

18. Skapa och använda webbtjänster. Sökord: webbtjänst. Hämtad från < http://nezzo.se/na.asp?id1=170&id2=3040>

19. Standarder inom XML världen. Sökord: XML. Hämtad från < http://www.statskontoret.se/publi/xmlstandard/index.html>

20. Terminologi. Sökord: SOAP. Hämtad från < http://www.microsoft.com/sverige/net/thisis/terms/default.asp>

21. Web services: The next big thing? Sökord: webservice. Hämtad från < http://www.internetnews.com/dev-news/article.php/10792_970851_1>

22. Wikipedia den fria Encyklopedin. Sökord: webbtjänst. Hämtad från < http://sv.wikipedia.org/wiki/Webbtj%E4nst>

23. Nextcom. Sökord: DMZ. Hämtad från <http://www.nextcom.se/_brandvagg-info.html>

24. Cryptography in Microsoft.NET Part III. Sökord: certificate. Hämtad från http://www.c-sharpcorner.com/Code/2003/Jan/DigitalCertIII.asp

25. Kerberos. Sökord: kerberos. Hämtad från <http://datorn.e.kth.se/upe/node47.html>

## *Examensarbete/Dokumentationer*

26. Persson, Tomas, Lindberg, Richard (2003) *Händelsestyrd uppdatering av Webbgränssnitt i .NET.* Examensarbete nr:E2919IT. Informationsteknologi, SSAB Tunnplåt AB, Högskolan Dalarna, Borlänge.

## *Intervjuer*

27. Larsson Johan (2004), Univ. Adj.informatik, Högskolan Dalarna,

023-77 88 68, e-post: jls@du.se

28. Nyberg Roger (2004), Univ. Adj.Informatik, Högskolan Dalarna, 023-778873, e-post: rny@du.se

29. Larsson Mats (2004), It-Arkitekt SSAB Tunnplåt, 0243-71105 e-post: mats.larsson3@ssab.com

30. Norlander Per (2004), Konsult Sogeti, 0243-71467, e-post: pear.norlander@ssab.com

31. Eliopoulos Georgios (2004), Konsult Nexus, 073 345 87 31 e-post: georgios.eliopoulos@nexus.se

32. Welinder Alex (2004), Konsult Portwise, e-post: alex.welinder@portwise.com

33. Martinsson Jon (2004), Konsult Portwise, 070 269 14 27 e-post: jon.martinsson@portwise.com

34. Näslander Jeanette (2004), .....,0243-7...., e-post: jeanette.naslander@ssab.com