Handelshögskolan
VID GÖTEBORGS UNIVERSITET
Institutionen för informatik

2006-05-30

Hur designas en SOA-tjänst?

– Argument för val av processer som skall understödjas av tjänster.

En fallstudie bland systemutvecklare på Skanska och Zystems

Idag möts företag av en ständigt föränderlig marknad. För att möta detta krävs det att företagen är flexibla så verksamheten kan förändras allt eftersom nya krav uppstår. Ett relativt nytt begrepp är SOA – Service Oriented Architecture. SOA är ett designkoncept som i korthet innebär att skapa återanvändbara tjänster utifrån identifierade processer i en organisation. Inom ämnet SOA så har hittills inte så mycket forskningslitteratur presenterats. Litteraturen vi har funnit grundar sig främst på erfarenheter på den amerikanska marknaden. Syftet med studien är att bidra till ökad kunskap om SOA samt förståelse för hur design av SOA-tjänster går till i Sverige. Frågeställningen innebar att utröna hur en SOA-tjänst designas och vilka argument som finns för val av processer som skall understödjas av tjänster. Ett antal djupgående intervjuer med anställda på Skanska och Zystems har ägt rum och insamlat empiriskt material har legat till grund för den jämförelse som gjorts mellan intervjumaterial och presenterad teori. En del faktorer som har betydelse för hur en SOA-tjänst designas har framkommit under studien. När det gäller hur valet av processer som skall understödjas av tjänster är svaren tvetydiga, men rådande behov verkar vara en avgörande faktor.

Nyckelord: design, lös koppling, process, service-oriented architecture, SOA, tjänst

Författare: Marcus Johansson, Magnus Lindström, Fredrik Ström
Handledare: Maria Bergenstjerna
Magisteruppsats, 20 poäng

Innehållsförteckning

1 Inledning 3
1.1 Bakgrund 3
1.2 Syfte och frågeställning 4
1.3 Avgränsning 4
1.4 Målgrupp 4
1.5 Översättning av begrepp från engelska 4
1.6 Disposition 5
2 Metod 6
2.1 Filosofiska synsätt 6
2.1.1 Metodval 6
2.1.2 Intervjuer 7
2.1.3 Tillförlitlighet 7
2.1.4 Argument för val av vetenskapligt tillvägagångssätt 7
2.1.5 Genomförande 8
3 Teori 10
3.1 Centrala begrepp 10
3.2 Zachmans ramverk 13
3.3 Business Process Management 16
3.4 Övergripande principer för tjänsteorientering 21
3.4.1 Fiktivt exempel på design av tjänster 23
3.5 Detaljerade anvisningar för SOA analys och design 29
3.5.1 Integrationsstrategier 29
3.5.2 Tjänsteorienterad analys 33
3.5.3 Tjänsteorienterad design 39
3.5.4 Designrekommendationer och riktlinjer 41
4 Empiri 44
4.1 Företagsintroduktion 44
4.2 Respondenter 44
4.3 Analys 45
5 Diskussion 60
5.1 Slutsats 62
6 Referenser 63
Litteratur 63
Artiklar 63
WWW 64
7 Bilagor 66

1 Inledning

Ett relativt nytt begrepp inom systemutveckling är Service Oriented Architecture – SOA. Det är ett designkoncept som i korthet innebär att skapa återanvändningsbara tjänster utifrån identifierade processer i en organisation. Inom ämnet SOA så har hittills inte så mycket forskningslitteratur presenterats. Detta kan man snabbt sluta sig till genom att göra en sökning på den akademiska vetenskapsdatabasen Science Direct [Sd1]. Som resultat fås idag enbart 21 träffar. Däremot finns det mycket icke-vetenskapligt material att finna runt om på Internet, vilket i sig kan betyda att SOA inte är någon vetenskaplig ”upptäckt”, utan snarare en företeelse skapad av etablerade systemutvecklings- och konsultföretag på marknaden, t.ex. IBM, utifrån ett reellt behov i företag, organisationer och myndigheter.

1.1 Bakgrund

Idag möts företag av en ständigt föränderlig marknad, vilket innebär ständigt nya krav ifrån kunder, nya regelverk osv. För att kunna möta detta krävs det att företagen är flexibla så att de kan förändra verksamheten efter de nya krav som uppstår (Kim et al, 2006). I en undersökning av 500 företag gjord av Fortune Magazine (Sprott 2004) visade det sig att över 80 % de senaste två åren gjort stora förändringar i sin affärsmodell. Ungefär två tredjedelar av dessa angav att förändringarna har försvårats på grund av dålig flexibilitet inom IT-stöd. I en undersökning gjord av IBM Business Consulting ansåg 90 % av tillfrågade verkställande direktörer, att de räknar med förändring av verksamheten på något sätt inom de kommande två åren (Sprott 2004).

SOA bygger på principen att använda tjänster. Tjänstebegreppet har sitt ursprung i affärsprocesser och affärsdesign. Generellt sett så är en tjänst en upprepande uppgift i en affärsprocess. Det gäller att identifiera sina affärsprocesser, bryta ner dem så långt att det blir möjligt och identifiera de uppgifter som utförs i processerna. Dessa upprepande uppgifter utgör tjänster. En tjänst kan vara alltifrån simpel dokumenthantering till avancerade affärstjänster där en behörig användare, som kan vara ett annat system, kan kontrollera sin status på en leverans (Sprott 2004).

En princip med SOA är att tjänsterna är löst kopplade, vilket innebär att de inte har någon relation till varandra, till skillnad från tidigare objektorienterade system där objekten oftast har någon form av relation till varandra. Eftersom tjänsterna inte har någon direkt relation med varandra så är det lättare att genomföra omstruktureringar av IT-arkitekturen. Med hjälp utav väldefinierade tjänstekontrakt, som är en slags överenskommelse för hur kommunikationen skall gå till, samt väldefinierade tjänstegränssnitt som beskriver hur tjänsten fungerar, är tjänsterna tänkta att fungera oberoende av vilken plattform de existerar på. Detta möjliggör lättare interorganisatorisk samverkan mellan olika IT-system (Erl 2005).

Idag existerar ingen standardiserad definition på SOA. Det finns heller ingen standardiserad metod för hur en implementation av SOA i en verksamhet skall gå till. Problemet är att många tror att ett SOA-projekt enbart är en teknisk implementation av Web Services. Med hjälp av en väldefinierad design och bättre förståelse går det att undvika detta problem (Erl 2005).

Av litteraturen ovan framgår att det finns ett behov bland företag att möta utmaningar i omgivningen genom lämpliga förändringar av såväl affärsmodell och som IT-stöd. Det är emellertid inte en lätt uppgift och många företag står rådvilla inför uppgiften och möjligheten att använda sig av SOA. Utifrån vår kunskap om systemutveckling har vi förståelse för litteraturens påstående att väldesignade tjänster är en viktig del i en lyckad implementation av SOA. Vi anser emellertid att litteraturen speglar den amerikanska marknaden och amerikanska företag. Den säger med andra ord ingenting om vilket behov svenska företag har, än mindre hur svenska företag går till väga för att designa en SOA-tjänst och vilka argument som ligger bakom valet av processer som ska understödjas av tjänster.

1.2 Syfte och frågeställning

Syftet med studien är att bidra till ökad kunskap om SOA samt förståelse för hur design av SOA-tjänster går till i Sverige. Med hjälp utav befintliga teorier samt en studie hos Skanska och Zystems så är syftet att komma fram till hur en tjänst designas samt utröna vilka argument svenska företag har för val av processer som skall understödjas av tjänster. Detta leder oss fram till följande frågeställningar:

Huvudfråga
Hur designas en SOA-tjänst?

Delfråga
Vilka argument ligger bakom valet av processer som skall understödjas av tjänster?

Vid användning av ordet design i vår studie menas både analysfas och designfas i utvecklingsarbetet. Drar vi paralleller till vår frågeställning, ser vi att vår huvudfråga ”Hur designas en SOA-tjänst?” avser just design, medan vår andra delfråga ”Vilka argument ligger bakom valet av processer som skall understödjas av tjänster?” är en kombination av analys och design.

1.3 Avgränsning

Studiens fokus ligger främst i att försöka jämföra hur teorin föreslår hur en tjänst bör designas kontra hur det ser ut i praktiken. Studien berör förhållanden inom byggbranschen. Vi har studerat införandet av SOA på Skanskas huvudkontor i Stockholm. Zystems är det företag som direkt har ansvarat för implementationen på Skanska. Endast personer som är inblandade i Skanskas SOA-projekt har intervjuats.

1.4 Målgrupp

Den målgrupp som uppsatsen först och främst riktar sig till är de personer som idag arbetar med eller har intresse av systemutveckling och implementation av SOA.

1.5 Översättning av begrepp från engelska

Den litteratur som används i vår studie är mestadels skriven på engelska, vilket har medfört att en del termer som beskrivs och används i uppsatsen inte är översatta till

svenska, detta på grund av att ingen svensk term som är helt liktydig med den engelska har funnits. Detta medför att risken för att förlora kommunikativ precision minskar. Se Bilaga 1 för begreppsförklaringar.

1.6 Disposition

Kapitel 1: Inledning
Kapitlet redogör för vad begreppet SOA innebär, bakgrunden till SOA och hur det beskrivs idag. Sedan Presenteras syftet med studien, vilka avgränsningar som gjorts, eventuella förtydliganden, vilken målgrupp som avses och språkkommentarer.

Kapitel 2: Metod
Kapitlet redogör för vilka filosofiska synsätt som finns idag, vilken metod som valts och hur tillvägagångssättet för insamling av empiriskt material utförts.

Kapitel 3: Teori
Kapitlet presenterar de teorier som legat till grund för studien och kapitlet avslutas med designrekommendationer och riktlinjer.

Kapitel 4: Empiri
Kapitlet inleds med en introduktion av de företag och respondenter som medverkat i vår studie. Kapitlet tar upp det insamlade empiriska material som legat till grund för studien och materialet kopplas senare till presenterade teorier i vår analysdel.

Kapitel 5 Diskussion
Kapitlet tar upp vårt resultat av den analys som skett av vårt empiriska material.

Kapitel 6: Slutsats
Kapitlet presenterar vår slutsats och svarar på de uppställda frågeställningar som legat till grund för vår studie.

2 Metod

Avsnittet förklarar innebörden av respektive vetenskapligt tillvägagångssätt, för att därefter argumentera för de val vi gjort för att kunna svara på vår frågeställning. Vi kommer även att ta upp praktiskt tillvägagångssätt under de intervjuer som skett, vilken utrustning som varit tillgänglig och vilka personer som blivit intervjuade och deras respektive position inom aktuellt företag.

2.1 Filosofiska synsätt

Idag finns tre filosofiska synsätt, positivistisk, konstruktivistiskt och relativistiskt. Det första är det positivistiskt synsättet som innebär att forskaren ser sig själv som det externa och är således inte en påverkande del av världen. Detta synsätt ses som det mest objektiva av de tre just på grund av att forskaren bara observerar det som sker och inte beblandar sig med det som studeras. Forskaren angriper det som studeras på ett objektivt sätt, vilket resulterar i att egna värderingar som kan påverka resultatet minimeras. Den kunskap som erhålls är endast intressant om den fås genom iakttagelser av den externa världen. När det gäller att dra slutsatser så använder det positivistiska synsättet sig av deduktion. Detta innebär att slutsatser härleds logiskt utifrån allmänna lagar (Easterby-Smith et al 2001).

Det andra är socialt konstruktivistisk synsätt vilket fokuserar mer på att verkligheten kretsar kring forskaren och dennes medmänniskor snarare än objektiva och externa faktorer. Fokus vid undersökning ligger således på människorna, hur dessa agerar i vissa situationer, samt hur de kommunicerar. Med konstruktivistiskt synsätt är forskaren med i olika situationer, där denne är del av den, och en ren objektiv studie är därför inte möjlig. Med konstruktivistiskt synsätt är det möjligt att dra slutsatser på både induktivt och deduktivt sätt (Easterby-Smith et al 2001). Med induktion menas att slutsatser härleds utifrån tidigare erfarenheter. Detta sker utifrån ett antal händelser som inducerar en slutsats [Wp1].

Det tredje är det relativistiska synsättet. Synsättet utgår ifrån att försöka få en mångfacetterad bild av problemet, detta genom att försöka hitta så många olika infallsvinklar av ett fenomen som möjligt. Den uppfattningen av teorier och koncept har inte bara en saklig betydelse, utan är även sociala konstruktioner som kallas kritisk realism.
I likhet med det konstruktivistiska synsättet betraktas inte objektivitet som en möjlighet och inte heller som sann verklighetsbild (Easterby-Smith et al 2001). Det relativistiska synsättet är som ett mellanting mellan de båda ovan nämnda synsätten. Det faktum att syftet är att skapa en bred och djup bild och att inte fokus läggs på det objektiva, gör att sambandet till det konstruktivistiska synsättet kan antas vara något starkare än till det positivistiska (Easterby-Smith et al 2001).

2.1.1 Metodval

Det finns två olika metoder att välja mellan, kvantitativ och kvalitativ metod. Vid kvantitativ metod samlar forskaren in empirisk och kvantifierbar data, som sammanfattas i statistisk form och sedan analyseras med hjälp av testbara hypoteser. Utgångspunkten är oftast stora populationer och med hjälp av mätinstrument försöker

forskaren fånga samband, fördelning och variation i det som studeras. En kvalitativ undersökning eftersträvar en helhetsbeskrivning av det som undersöks, så metoden tenderar därför att omfatta mindre populationer än vid kvantitativa undersökningar gör eftersom ett lämpligt, för studien, antal intervjuer äger rum [Ne1].

2.1.2 Intervjuer

Det finns fyra olika typer av intervjuer, personlig-, grupp-, telefon- och e-postintervju. Den personliga intervjun sker mellan respondenten och den som skall intervjua. Detta tillvägagångssätt är det mest personliga när respondenten och den som intervjuar befinner sig på samma plats och möjlighet till personlig kommunikation möjliggörs, där kroppsspråk kan uppfattas och tolkas. Gruppintervjun innebär att det finns flera respondenter som intervjuaren ställer sina frågor till vid samma tillfälle (Nordberg 2000).
Telefonintervjun sker över telefon, vilket tyvärr innebär att kroppsspråk försvinner helt (Krag Jacobsen 1993). Den sista formen av intervju är e-postintervjun. Här skickas de frågor som intervjuaren tänkt ställa till respondenten via e-post. Detta möjliggör att respondenten får lång tid på sig att svara på ställda frågor. Även här faller möjligheten till personlig interaktion bort mellan respondent och intervjuare.

Det finns två olika typer av intervjuer, strukturerade och standardiserade. Strukturering handlar om vilka möjligheter som finns när det gäller upplägg av de frågor som skall ställas (Andersen 1994). Det är upp till den som intervjuar om det skall vara hög eller låg grad av strukturering (Patel et al 2003). Vid hög grad av strukturering av presenterade svarsalternativ är ordningen mer förutbestämd, medan låg grad av strukturering innebär ett öppnare upplägg (Hartman 1998). Standardisering av intervjufrågor handlar om hur själva frågorna är utformade och formulerade. Vid hög grad är frågorna ordnade i ett bestämt mönster och ställs i den ordningen, medan låg grad innebär att frågorna kan ställas utan inbördes rangordning (Andersen 1994).

2.1.3 Tillförlitlighet

Reliabilitet
När en forskningsstudie bedrivs, är det viktigt att beakta det resultat som forskningen resulterar i (Ejvegård 2003). Reliabilitet mäter hur tillförlitlig en forskningsmetod är. Ju fler oberoende studier som får samma eller liknande resultat, desto högre tillförlitlighet uppnås (Bell 2000).

Validitet
När forskning bedrivs är det viktigt att alla delar i forskningsprocessen uppnår hög validitet. Validitet kan generellt sägas vara ett mått på hur väl själva mätningsförfarandet mäter det som skall mätas (Bell 2000).

2.1.4 Argument för val av vetenskapligt tillvägagångssätt

Metoden som vi valde att använda var kvalitativ metod med relativistiskt synsätt. Valet av kvalitativ metod beror på att vi står i direkt kontakt med den sociala verklighet som analyseras, det vill säga Skanska och Zystems. Eftersom undersökningar ofta sker med större populationer, är enkäter en bra teknik att använda

sig av, men för vår del är denna metod inte att föredra. Det beror på att SOA som begrepp inte har en tydlig definition idag, därför skulle ett rent kvantitativt val av metod ge sämre kvalité på de svar vi söker.

Intervjuernas uppbyggnad var av semistrukturerad karaktär med låg grad av strukturering och standardisering. Detta för att inte ha en för fast struktur på de frågor som ställts och att eventuella följdfrågor enkelt kan ställas.
Personlig intervju är oftast att föredra för den ger möjligheten att se och använda kroppsspråk, vilket är viktigt för att få en så tydlig tolkning som möjligt av det material intervjun mynnar ut i.

Telefonintervjun är ett alternativ till personlig intervju när respondenten befinner sig ”för långt bort” för att det skall vara befogat att ta sig till aktuell plats. Den ger möjlighet att tolka röstläge, vilket också är ett slags kroppsspråk.

E-postintervjun kan användas men är en ganska strikt och intetsägande metod för inget kroppsspråk finns med i bilden, det är bara orden som kan och skall tolkas.

Det är ett faktum att i och med att forskaren själv inte deltar i områden som denne studerar, uppfattas detta som en fördel av objektivitetsskäl [Ne2]. Datainsamling och analys skedde kontinuerligt genom arbetets gång, och vår fokus låg i att fånga såväl människors handlingar samt vad som motiverar dessa handlingar.
Varför vårt val föll på det relativistiska synsättet beror på att vi ville bilda oss en så mångfacetterad bild av vad SOA egentligen är och hur det uppfattas av andra människor. Detta gör vi genom att studera multipla källor såsom teoretisk litteratur och genom en fallstudie.

2.1.5 Genomförande

Vi har genomfört totalt fem intervjuer, varav fyra djupintervjuer, med personer som är anställda på Skanska och Zystems. Den resterande intervjun har skett via e-post och telefon. Zystems är det företag som ansvarade för den tekniska implementationen av en SOA hos Skanska. Skanska hade huvudansvaret att planera och övervaka den implementation som skett. En mer utförlig introduktion av ovan nämnda företag finns i analysavsnittet 4.4.1.

De intervjuer som genomförts har inkluderat personer från ledning, projektledning och tekniksidan, i den ordningen. Representanten från ledningen på Skanska är integrationschef på nämnd firma. De andra två personerna är från Zystems. De frågor som ställdes var av olika karaktär beroende på vilken position personen hade på företaget, det vill säga vi ställde frågor om bakgrund och syfte till implementationen av SOA, till den person som var ansvarig för den delen.

Djupintervjuerna har skett på plats hos Skanska i Stockholm. Dessa tre intervjuer var nödvändiga att ske samma dag med tanke på den stora geografiska skillnaden mellan Göteborg och Stockholm. En intervju skedde på Zystems kontor i Göteborg och den resterande via e-post och telefon. Intervjuerna med de anställda hos Skanska i Stockholm skedde med alla tre informanter i gruppen på plats och spelades in på två olika typer av inspelningsutrustning. Vi använde oss av en bärbar hårddiskinspelningsapparat och en bärbar dator och spelade in allt via de inbyggda

mikrofonerna. Valet att ha med två typer av inspelningsutrustning beror enbart på att eliminera de eventuella risker som existerar gällande fallerande utrustning, vilket skulle ha fått allvarliga konsekvenser med tanke på att inga manuella anteckningar fördes under de intervjuer som utfördes. Kvalitén blev mycket god, vilket medförde att det blev en problemlös transkribering av varje utförd intervju. Den huvudsakliga anledningen till att inspelningsutrustning användes är för att inte missa väsentlig information och minimera eventuella missförstånd som kan uppkomma vid manuell anteckningsform.

Viktigt att nämna är alla de frågor som vi arbetat fram har en mycket viktig del i arbetets resultat. Detta medförde att vi ansträngde oss mycket för att se till att de frågor som vi skulle ställa var genomarbetade och tydliga. Lika viktigt var att vi var väl införstådda i varje frågas kärna och vad varje fråga betydde, detta för kunna ge en djupare förklaring till frågan om något skulle vara oklart för den som intervjuades. Vi hade även med oss en beskrivning om hur vi tolkar begreppet SOA och dess innebörd om eventuella meningsskiljaktigheter eller missförstånd skulle uppkomma.

De intervjuer som skett i Stockholm transkriberades för att underlätta vårt analysarbete för uppsatsen. De svar vi spelade in kom textmässigt att filtreras till den grad att eventuellt talspråk skrevs om till skriftspråk. Analys av intervjumaterialet skedde först enskilt, för att senare delas ut till resterande gruppmedlemmar för enskild analys. Efter det genomfördes en gemensam tolkning och analys med alla gruppmedlemmar närvarande, detta för att minimera eventuell risk att de svar som erhållits misstolkats eller hanterats utanför dess rätta kontext.

Det empiriska material som extraherades utifrån genomförda intervjuer, analyserades för att jämföra Skanskas sätt att implementera SOA och vad SOA är för dem, gentemot den uppfattning vi bildat oss under vår teoretiska studie av SOA och hur det är tänkt att implementera detta utifrån de teorier vi tidigare beskrivit under teoriavsnittet. Slutligen diskuteras resultatet och en sammanfattning görs under rubrikerna 4.2 Analys och 5 Diskussion.

När det gäller den reliabilitet och validitet som vår studie har uppnått, är det viktigt att poängtera att SOA är ett relativt nytt begrepp på marknaden och eftersom det utfördes ett fåtal intervjuer, kan detta ha påverkat vårt resultat negativt. Med tanke på att de litterära referenser vi haft tillgång till har kommit fram till samma sak angående hur en SOA skall designas, anser vi att detta väger över åt att ha en god reliabilitet i vår studie. Däremot när det gäller validitet kan det vara så att den har en lägre nivå med tanke på det fåtal intervjuer som genomförts. Dessa intervjuer har förvisso gett samma resultat, men att detta empiriska material endast kommer ifrån ett och samma företag måste beaktas för det kanske inte hade blivit samma svar på de frågor vi haft om insatta personer på andra företag intervjuats. Vi har valt en kvalitativ inriktning där ett fåtal djupgående intervjuer har skett med personer som var insatta i ämnet utifrån deras sätt att se på SOA och vad ämnet innebar, och har erfarenhet av implementering av SOA i ett företag, som i vårt fall är Skanska.

3 Teori

Avsnittet nedan presenterar de teorier som ligger till grund för studien och de jämförelser som framöver kommer att utföras och redovisas i kapitlen Analys och Diskussion. Först ger vi en förklaring till centrala begrepp, därefter presenteras Zachmans ramverk. Denna ger en övergripande uppfattning om vilka roller som berörs i systemutveckling och vilka modeller som vanligtvis används. Sedan ges en detaljerad bild av hur SOA designas enligt den litteratur som presenterad teori hämtats från.

3.1 Centrala begrepp

Här presenteras de centrala begrepp som används i huvudfråga och delfråga. Begreppen är design, SOA, tjänst och process. Begreppet tjänst inkluderar även några varianter.

Design

Design är en internationellt använd term för formgivning, det vill säga gestaltning av hantverkligt eller industriellt framställda produkter och miljöer. Arkitekturstrategi i ett informationssystem är framställd ur en modell som skapas med hjälp av systemdesign. En person som skapar systemdesign kallas systemdesigner [Wp06].

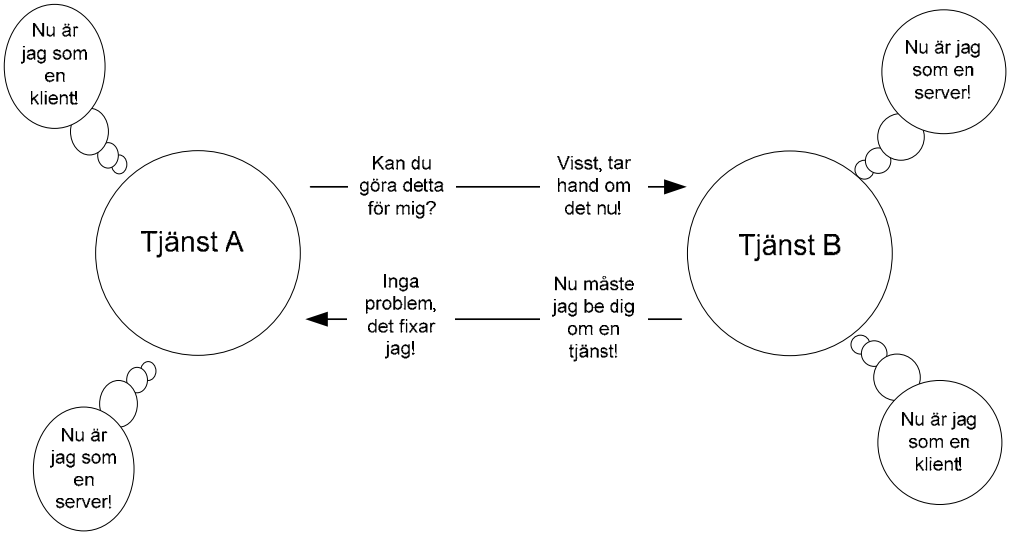
SOA

SOA är en förkortning för Service Oriented Architecture, eller tjänsterorienterad arkitektur på svenska. SOA är ett systemarkitektiskt koncept som beskriver användningen av tjänster för att uppfylla affärsmässiga krav på ett IT-system. Tjänsterna är löst kopplade till skillnad från tidigare system och arkitekturer där tjänsterna är hårt kopplade. För att förstå SOA måste först vissa begrepp gås igenom, såsom tjänst, lös koppling, systemarv och även Web Services, som är en väl använd teknik för att implementera SOA [Ibm2]. I ett system som är uppbyggt enligt SOA-principen så är resurserna tillgängliga för andra program på ett nätverk, i form av oberoende tjänster och kan anropas på ett standardiserat sätt. Ofta förknippas SOA med webbtjänster baserade på XML, SOAP, WSDL och UDDI, men detta är inte på något sätt ett krav. Ett SOA-system kan byggas upp med vilken tjänstebaserad teknik som helst. Förutsättningen att SOA skall lyckas är just kravet på en standardisering (Erl 2005).

Tjänst

En tjänst är ett visst arbete som utförs till nytta för någon annan. Tjänster sätts ofta i motsats till produkter, som resultat av utfört arbete. Tjänster är en av grundpelarna i tjänsteorienterad arkitektur. Generellt sett så är en tjänst en iterativ uppgift i en affärsprocess. Följande citat beskriver en process: *”en process är ett repetitivt använt nätverk av i ordning länkade aktiviteter som använder information och resurser för att transformera objekt in till objekt ut, från identifiering till tillfredsställelse av kundens behov”(Ljungberg et al 2001).* Med andra ord, om identifikation av sin affärsprocess är möjlig och i förlängningen identifiering av de uppgifter som måste utföras i denna process, så är lokaliseringen av de tjänster som ingår i SOA klar. En tjänst kan vara allt från simpel dokumenthantering till avancerade affärstjänster där en behörig användare (som kan vara ett annat system) kan kontrollera sin status på en leverans [Ibm1. Figur 1 illustrerar hur en tjänst kan beskrivas (Erl T, 2004).

Tjänsteförsörjaren (service provider) beskriver vad tjänster består av och definierar den. Tjänsteagenten (service broker) katalogiserar tjänsten som sedan används (Erl 2004). Tjänsterna är tillståndslösa. Inkommande meddelande innehåller all den information som tjänsten behöver för att utföra sin uppgift och det utgående meddelandet innehåller all information användaren behöver få tillbaka (Erl 2004). Därför består kommunikationen mellan användaren av tjänsten och tjänsten själv enbart av ett anrop istället för ett flertal anrop som det är med vanliga objekt.



Figur 1.Illustration av en tjänst (Erl T, 2004).

Affärstjänster
Informationsflödet inom en organisation, oberoende storlek, kan brytas ner i en samling av affärstjänster. Det beror på att affärstjänster helt enkelt representerar små enheter av arbetsuppgifter inom en organisation. Hur en verksamhet dokumenterar sin affärsverksamhet varierar dock mycket. Med hjälp utav till exempel Business Process Management (BPM), kan man lätt skaffa sig en överblick över sin verksamhet. Därigenom blir det lättare att bryta ner processerna och sedan identifiera var möjliga tjänster skulle kunna passa in (Erl 2005).

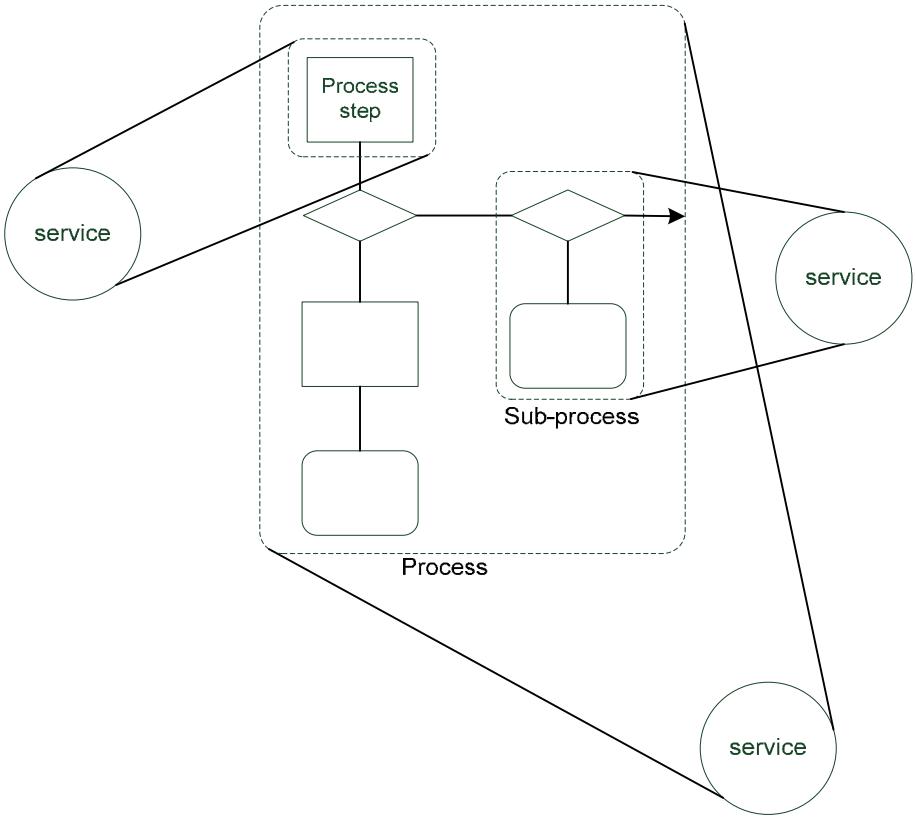
Andra verktyg som kan användas vid framtagandet av affärstjänster är entitetsmodeller. Entitetsmodeller representerar dokument- och transaktionsområden inom en verksamhet, exempelvis beställningsorder, faktura och kund med flera. Dessa är likadana som de som används inom objektorienterad systemutveckling, och namnges därefter ofta på samma sätt. Det kan tyckas förvirrande att blanda in objekt eller entiteter i tjänsteorientering men de fyller faktiskt en bra funktion inom en SOA som snart skall presenteras (Erl 2005).

Applikationstjänster
Applikationstjänsterna fungerar som förmedlare av teknikspecifik funktionalitet. Deras syfte är att tillhandahålla återanvändbara funktioner som är relaterade till att behandla data inom nya eller gamla ärvda informationssystem (Erl 2005).

Atomära tjänster
Är inte beroende av andra tjänster och är vanligtvis associerade med enkla och rakt på sak tjänster. Exempel på sådana tjänster kan vara ”Hämta faktura” (Newcomer et al 2005).

Entitestjänster
En entitetstjänst hanterar objekt. Det som gör att entitetstjänsterna blir återanvändbara är att de inte är knutna till någon specifik affärsprocess inom verksamheten, som funktionstjänsterna är. Detta medför inte bara hög återanvändbarhet utan det medför även att dessa är smidigare vid processförändringar, då dessa slipper att förändras, vilket funktionstjänsterna kan komma att göra (Erl 2005).

Funktionstjänster
Funktionstjänster tillhandahåller uträknad data efter förfrågan och fungerar som stöd i specifika processer. Tjänsterna har operationer som är relaterade till särskilda arbetsuppgifter inom processerna. Dessa funktionstjänster är ofta de som utkommer ur analys av processerna i företaget, där BPM samt även användarfall kan vara behjälpliga. Dessa tjänster är lätta att modellera men saknar ofta återanvändbar kapacitet. För att göra funktionstjänsterna till återanvändbara tjänster, analyseras flera olika processer samt eventuella användarfall, för att försöka finna gemensamma funktioner (Erl 2005).



Figur 2. Illustration av hur delar av en process kan inkapslas av en affärstjänst (Erl T, 2005)

Hybridtjänster
Tjänster som innehåller både applikationslogik och affärslogik presenterar Erl (2005) som hybrida tjänster. Den typen av tjänster är enligt den vanligaste typen ute bland verksamheter idag

Sammansatta tjänster
Använder andra tjänster, en sammansatt tjänst består minst av två tjänster. För övrigt har en sammansatt tjänst samma karakteristik som en vanlig tjänst:

- Har ett väldefinierat tjänstekontrakt
- Finns registrerade någonstans i ett tjänsteregister där det kan sökas upp och exekveras som vilken tjänst som helst
- Implementerad, styrda och säkrade som övriga tjänster
- Kan använda andra tjänster, även andra sammansatta tjänster

Process
Process i sin vidaste betydelse betecknar någon sorts förlopp. En process innebär nästan alltid ett förändringsförlopp mellan två tidpunkter [Wp04].
En process är en sammanhängande serie händelser i tiden. Det kan vara en industriell process t.ex. för tillverkning av papper av pappersmassa som kokas av träfiber, eller ett exekverande program i en dator vars instruktioner utförs en i taget [Su01].

3.2 Zachmans ramverk

Avsnittet presenterar en beskrivning av Zachmans ramverk och hur det är uppbyggt. Zachmans ramverk påvisar problemet med kommunikation mellan de olika öar som finns i en organisation. Öar finns i systemarkitekturer – men också bland människor (Ö = avgränsat område). Detta är också anledningen till att ramverket finns med i vår uppsats för detta anser vi vara en grundbult i alla verksamheter. Alla aktörer ser verksamheten på olika sätt. De har olika motiveringar till hur och varför saker skall göras och kommunikationen dessa emellan fyller en viktig funktion.

Zachmans ramverk för informationssystemarkitekturer presenterades första gången 1987 och senare som en utbyggd version 1992 och har enligt Office of Information and Technology blivit en standard inom utveckling av informationssystemarkitekturer [Cs1]. Ramverket är ett resultat av Zachmans studier av processen när komplexa ingenjörsprodukter byggs, gällande både arkitektur, konstruktion och själva tillverkningen. Resultatet av dessa studier blev ett ramverk att använda inom mjukvaruindustrin eftersom denna inte har tillgång till samma erfarenhet som den traditionella industrin har (Zachman 1987). Zachmans grundläggande utgångspunkt var att en organisations förmåga att planera och styra sina resurser är beroende av konsistent information. Av flera faktorer som bidrar till framgångsrik IT-management, lyfter Zachman särskilt fram förmågan att uppfatta och förstå den IT-managementverksamhet som den strategiska IS/IT-planen skall stödja (Magoulas et al 1998). I Zachmans ramverk delas verksamheten upp i en matris, där varje rad får representera ett perspektiv ur vilket man betraktar verksamheten. Dessa perspektiv börjar med ett mycket övergripande

perspektiv, och går sedan mot en mer och mer detaljerad beskrivning (utförarens perspektiv). Dessa perspektiv delas sedan in i ett antal dimensioner vilka representeras av kolumner i matrisen [Al1].

Zachman menar att om inte IS-arkitekturen ges denna dynamiska och sammanhängande roll kommer projekten när det färdigställts, inte att bygga på någonting, utan resulterande system existerar utan krav på externa relationer till andra system.

"Without an understanding of this management system, an I/S planning effort seems to be a one-time, stand alone analysis that has no continuing impact on the organization and its ability to manage its resources more effectively."

Citat från John A. Zachman taget ifrån [Zach87] sida 9.

Det resulterade systemets värde har inget annat värde, än värdet av att lösa enskilda problem, vilket i sig är meningslöst om inte helheten fungerar. Det strategiska IT-ledningens system består av fyra delar: verksamhetssystemet, IS-arkitekturen, projekt samt slutligen den strategiska IS/IT-planen [Sim98].

1. Verksamhetssystemet utgörs av människor som verkar tillsammans och omvandlar resurser till produkter och resultat. Här i existerar relationer mellan alla olika delar som i sig bygger upp verksamheten, det kan vara processer, resurser, mellan produkter och aktörer, o.s.v. Varje relation mellan två entiteter förutsätter ett informationsflöde.
2. Strategisk IS/IT-planering är den process som syftar till att bestämma målsättning, strategier och resurser för IS/IT. Utformningen av den strategiska planen förutsätter att verksamhetsansvariga har selekterat och beskrivit de entiteter som är relaterade med väsentliga informationsflöden.
3. IS-arkitekturen utgörs av alla informationsflöden, strukturer, funktioner, mm som syftar till att förbättra vår förståelse av verksamheten. Arkitekturens datorbaserade del utgörs av system och data. Data representerar relationer mellan de entiteter som tillsammans utgör verksamheten. Systemen utgörs av funktioner som försörjer verksamhetsansvariga med information om relationerna som råder mellan entiteterna.
4. Projekt utgör IS/IT-managementsystemets fjärde område. Projekten omfattar aktuella investeringar i resurser för att driva utvecklingsaktiviteter och eliminering av ”störningar” som verksamhetens aktörer upplever. Kostnadsreducering, förbättrad effektivitet i beslutsfattande, förbättrad vinst mm utgör exempel på effekter som man eftersträvar genom effektivisering av informationsförsörjning. Ur det perspektivet kan varje projekt ses som ett system under utveckling.

Zachmans bestämda uppfattning om den strategiska processen är att IS-arkitekturen härleds utifrån verksamhetens struktur betraktat utifrån ett informationsförsörjningsperspektiv.
Bland de uppgifter som ingår i IT-management, lyfter Zachman särskilt fram följande[Sim98]:

- Öka kunskaperna om de faktorer som skapar behov av en IS-arkitektur. Arkitekturens roll är att representera de informationsflöden som är både nödvändiga och väsentliga för att driva verksamheten och som bör fungera på ett effektivt och säkert sätt.
- Analysera verksamhetsstrukturen och definiera dess informationsflöden. Utifrån denna definition görs en beskrivning av IS-arkitekturen. Verksamheten bör studeras utifrån ett dynamiskt perspektiv.
- Välja lämpliga verksamhetsanalytiska metoder och modelleringstekniker som bland annat fokuserar på informationsflödenas kvalitativa egenskaper.
- Framställa en handlingsplan för att migrera från en olämplig informationsmiljö till en arkitekturbaserad informationsmiljö (notera att Zachman betraktar verksamheten som den miljö en IS/IT-plan syftar till att effektivisera).
- Bedriva projekt som härleds från IS/IT-planen och IS-arkitekturen. Validera och välja utvecklingsmetoder och tekniker.
- Utforma en IT-infrastruktur som ”matchar” IS-arkitekturens egenskaper.
- Planera hur de fyra delarna som omfattas av IT-management skall integreras.

Zachmans ramverk illustreras i form av en matris, se Figur 3.

	data (what)	**process** (how)	**network** (where)	**people** (who)	**time** (when)	**motivation** (why)
Scope (planner)	List of important enfities	List of business processess	List of operating locations	List of important organizationals	List of significant events	List of business Goals
Enterprise Model (Owner)	Semantic Model	Business Process Model	Logistic Network	Work Flow Model	Master Schedule	Business Plan
System Model (Designer)	Logical Data Model	Application Arch.	Distributed System Arch.	Human Interface Arch.	Processing Structure	Business Rule Model
Technology Model (Builder)	Physical Data Model	System Design	System Arch.	Presentation Arch.	Control Structure	Rule Design
Detailed Representation (Subcontractor)	Data Definition	Program	Network Arch.	Security Arch.	Timing Defintion	Rule Specification
Functioning Enterprise	Data	Function	Network	Organization	Schedule	Strategy

Figur 3. Zachmans ramverk för informationssystemarkitektur. Kombination av alla celler på en rad ger en fullständig beskrivning av den aktuella vyn [Cs03].

Raderna i matrisen representerar olika aktörers syn på utvecklingsprocessen, det vill säga olika vyer. Dessa är, enligt The Zachman Institute for Framework Advancement [Zif02]:

- **Scope** (Planerarvy) - definition av verksamhetens mål och inriktning.
- **Enterprise Model** (Ägarvy) - definierar verksamheten och dess struktur, funktion, organisation med mera.
- **System Model** (Desigvy) - definition av verksamheten som i rad 2 men mer informationsinriktat.
- **Technology model** (Utvecklarvy) - definierar hur teknik kan användas för att hantera informationsprocessen som identifierats i tidigare rader.
- **Detailed representations** (Underleverantörsvy) - specificerar till exempel databasen, nätverk och så vidare. Uttrycks i speciellt språk.
- **Functioning enterprise** - systemet är färdigutvecklat. Kolumnerna i matrisen representerar det som ska undersökas av aktörerna och kan enligt, Zachman, beskrivas som nedan.
- **Data** (vad) - beskriver verksamhetens data och relationen mellan denna data.
- **Function and Processes** (hur) - beskriver hur verksamhetens data används, till exempel hur ordrar görs/läggs.
- **Network** (var) - beskriver verksamhetens nätverk. Till exempel om all data förflyttas mellan datorer i ett och samma hus eller om datorer måste vara ihopkopplade över hela världen.
- **People** (vem) - beskriver ansvarshierakier och människor inom verksamheten och det arbete de utför.
- **Time** (när) - beskriver tidsplanering av processer, vilket kan vara en specifik tidpunkt som startar en process, tiden på en händelse som startar en annan process eller en tidssekvens som beskriver i vilken ordning processer måste köras.
- **Motivation** (varför) - beskriver verksamhetens mål och drivkrafter [Cs03].

3.3 Business Process Management

Enligt Mike Rosen (2004), ”Chief Scientist” hos Wilson Consulting Group, så är SOA och Business Process Management (BPM), starkt relaterade till varandra. SOA kan användas för att skapa återanvändningsbara tjänster, men utan BPM så går det inte att utnyttja dessa tjänster för att skapa en flexibel verksamhet. BPM kan användas för att skapa applikationer som skall stödja affärsprocesser, men utan SOA är det svårt att sprida detta till hela verksamheten. Vi anser därför att det är nödvändigt att förklara begreppet BPM, samt visa relationen till SOA.

Grundläggande Business Process Management koncept
En affärsprocess är en verklig aktivitet eller mängd händelser som, om dessa utförs i rätt ordning, resulterar i ett värdeskapande resultat. Citat: - *En process är ett repetitivt använt nätverk av i ordning länkade aktiviteter som använder information och resurser för att transformera ”objekt in” till ”objekt ut”, från identifiering till tillfredsställelse av kundens behov* (Ljungberg et al 2001) sida 44. Business process management adresserar hur organisationer kan identifiera, modellera, utveckla, implementera och hantera sina verksamhetsprocesser. I dessa ingår processer som involverar IT-system och mänsklig interaktion (Newcomer et al 2005).

Några av huvudmålen med BPM är att:

- Reducera problem med olikheter mellan affärskrav och IT-system, detta genom att låta affärsverksamheten modellera processer som sedan IT-avdelningen kan skapa en infrastruktur till för att ge stöd för de befintligt uppställda krav som ställts av specifika affärsprocesser.
- Öka produktivitet och reducera driftskostnader genom att automatisera och strömlinjeforma affärsprocesser.
- Öka verksamhetens förmåga till snabba förändringar beroende på omgivningens krav, detta genom att specifikt skilja på processlogik från övriga affärsverksamhetsregler, för att sedan presentera processer på ett sätt som är lätt att förändra och överblicka. Detta i sig tillåter organisationer att enklare svara mot förändringar och anpassa sig allteftersom marknaden förändras.
- Minska utvecklingskostnader och arbetsinsatser genom att använda ett grafiskt programmeringsverktyg som tillåter utvecklare att bygga och uppdatera IT-system. (Varför skulle kostnaderna minska på grund av ett grafiskt utvecklingsverktyg?)

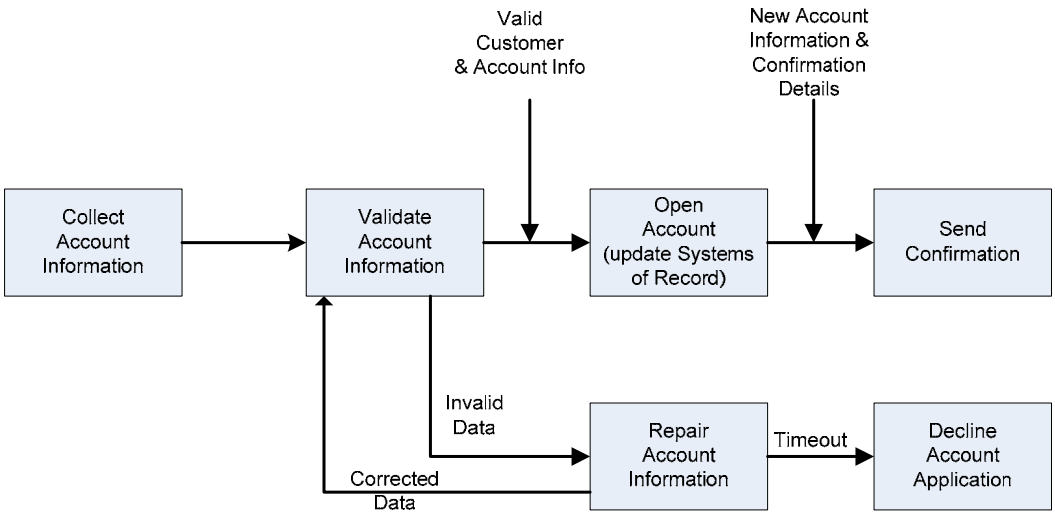
Enligt Eric Newcomer och Greg Lomow är det uppenbart att alla IT-system stödjer någon form av processer. Vad som gör business process management unikt är att den skiljer på affärsprocesslogik från befintliga affärsregler. Inom andra utvecklingsverktyg är processlogiken djupt inbyggd i själva applikationerna (Newcomer et al 2005).

Business Process Management Systems
BPM är ett koncept som används för att definiera, hantera och realisera affärsprocesser till att bli en företagstillgång. Business Process Management Systems - BPMS är istället själva verktyget för att realisera detta koncept. Den förser användaren med de verktyg som behövs för att konstruera en eller flera av de nyckelelement som BPM bkräver (Newcomer et al 2005).

Ett av nyckelelementen i BPMS hanterar modellering av affärsprocesser, det är denna del vi anser vara relevant för vår studie av BPMS, därför tar vi endast upp processmodellering. Målet med processmodellering är att hitta verksamhetskrav tidigt i utvecklingsarbetet, för att sedan låta dessa ligga till grund för den fortsatta utvecklingen. Processmodellering inleds med en verksamhetsanalys för att definiera behov med hjälp av verksamhetsmodelleringsverktyg. Processmodellen definierar beroenden mellan olika applikationer, hur applikationerna är sekvenserade, när och

hur en applikation används, och vem som kan utnyttja vilka applikationer. Se Figur 1. Normalt överlämnas dessa modeller till mjukvaruutvecklare eller andra tekniskt kunniga för att koppla modellerna till verksamhetens IT-tillgångar eller utveckling av nya IS-stöd för att fylla de behov som processmodellen påvisar (Newcomer et al 2005).

Verksamhetsanalytiker och IT-avdelning kan använda processmodellen för att testköra simuleringar. Verksamhetsanalytiker kan t.ex. använda simuleringar för att lokalisera flaskhalsar i processer.
Dessa flaskhalsar kan vara allt ifrån vilka tjänster/applikationer som arbetar hårdast under specifik belastning, till hur detta i sin tur kan påverka informationsflödet på servern, vilket den tekniska expertisen tittar på (Newcomer et al 2005).



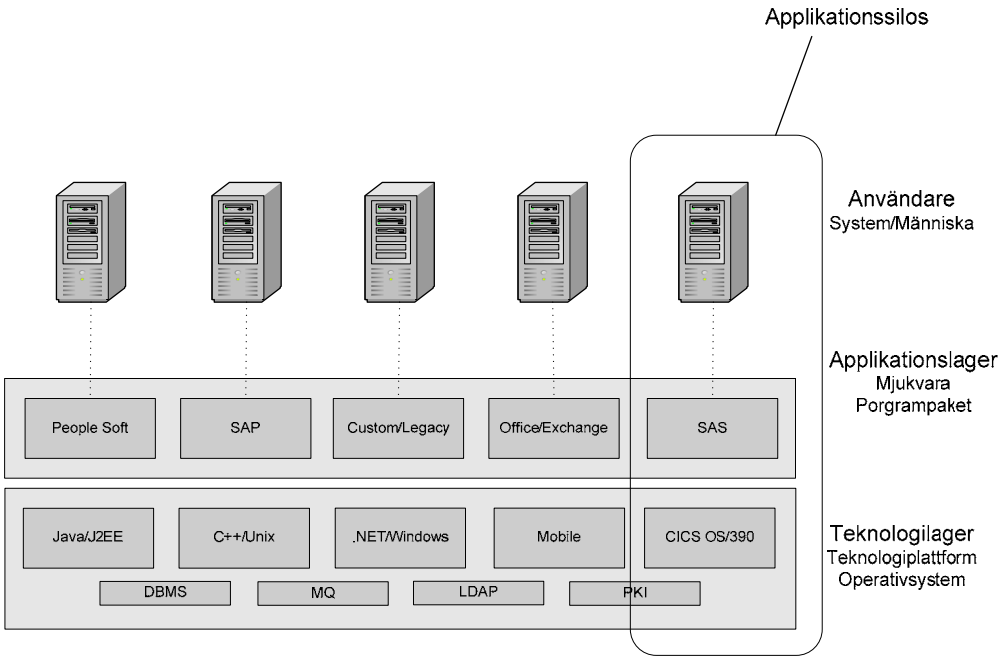
Figur 4. Illustration av en enkel process (Newcomer et al 2005).

I Figur 4 visas en enkel modell över en process, i det här fallet handlar det om att öppna ett konto. Generellt inkluderar en verksamhetsprocess vilka aktiviteter som skall utföras, kopplingar mellan aktiviteter som talar om i vilken ordning dessa aktiviteter skall utföras, data som skall passera mellan aktiviteter och till sist vilka regler som måste uppfyllas för att möjliggöra koppling mellan aktiviteter (Newcomer et al 2005).

Avsnittet nedan presenterar en kombination av BPM, SOA och Web Services kan se ut. Detta genom att visa hur affärstjänster kan göras tillgängliga, dels baserade på redan existerande äldre grundsystem så kallade ”legacy systems” och nyutvecklade linjer av affärstjänster (Newcomer et al 2005).

De flesta organisationer (enligt Newcomer, Lomow) har en mängd olika ”IS-applikationer” och ”IT-teknologier” som grund för sin verksamhet, se Figur 5. Det typiska är att det finns ett flertal olika applikationssilos. Newcomer och Lomow kallar dessa för silos på grund av deras isolerade natur, och med isolerande menas att allt från gränssnitt till applikationsspecifik affärslogik och databaser hålls åtskiljda. Dessa silos kan innehålla alltifrån GUI:s till applikationsspecifik affärslogik och

applikationstillhörande databaser. Att dela information dessa silos emellan kan vara mycket svårt på grund av deras olika typer av teknologiska plattformar och databaser (Newcomer et al 2005).



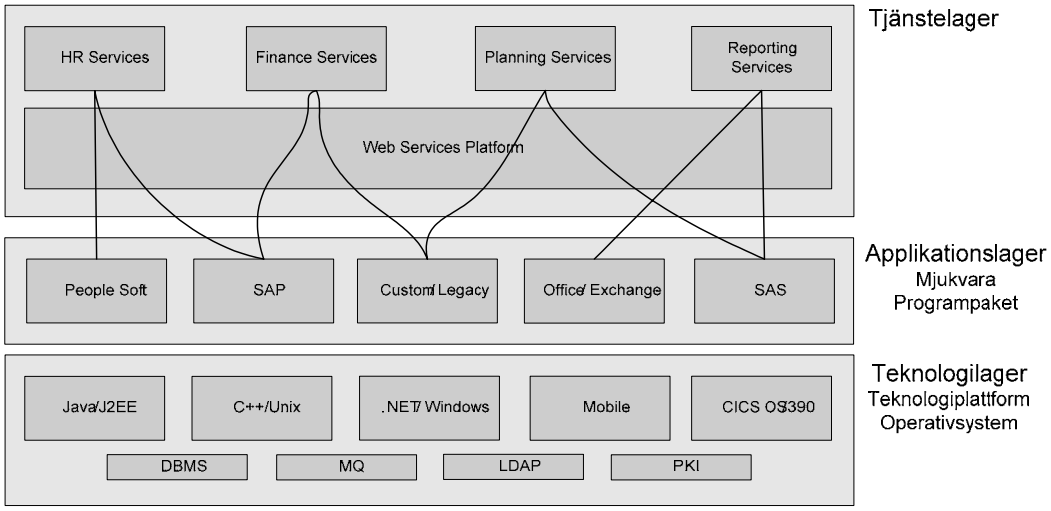
Figur 5. Typiskt IS och IT landskap (Newcomer et al 2005).

När sedan tillämpning av SOA och Web Services påbörjas introduceras ett tjänstelager. Detta lager består av en eller flera linjer av affärstjänster som är formade efter en specifik verksamhetsdomän (detta inkluderar de specifika datamodeller som passar varje domän), återanvändningsbara tekniska tjänster som kan nyttjas av flera olika affärsverksamheters domäner. Detta ger möjligheten att definiera och tillämpa tjänster, på ett sätt som gör det möjligt att bli oberoende av underliggande applikationer och teknologier, se Figur 6 (Newcomer et al 2005).

Ett tjänstelager bidrar med en ideal plattform som stöd för processlagret.
En anledning är att kunna dra nytta av fördelen med SOA genom detta steg.
Detta gör det alltså möjligt att ha ett enkelt och föränderligt tjänstelager med tillgång till de undre applikationerna, detta genom tjänster som i sin tur kan väljas av processerna vid behov. Nedan visas ett antal skäl som visar fördelarna med att dra nytta av tjänstelager för stöd till processerna i verksamheten (Newcomer et al 2005).

- Linjen av verksamhetstjänster förser en grovkornig verksamhetsfunktionalitet som formas efter eller kopplas till verksamhetsuppgifterna i processerna.
- Tjänstekontrakten för själva verksamhetstjänsterna förser väldefinierade och entydiga gränssnitt till de intressenter som använder tjänstelagret. Detta gör det möjligt för processerna att enbart bry sig om resultatet av tjänsterna och i slutändan betyder detta att tjänsten inte behöver bry sig om hur detta utförs.
- Tjänsteregistret och tjänstelokaliseraren som ligger i själva tjänstelagret gör det möjligt för processlagret att dynamiskt hitta och utnyttja de tjänster som behövs.

- Tjänstelagrets datamodell är definierad på verksamhetsdomänen och är oberoende av den datamodell som applikationerna använder sig av. Vidare ansvarar XML för det kanoniska format som krävs av tjänsteregistret vid förfrågningar av tjänstelagret, exempelvis när en process kräver något från en underliggande applikation med hjälp av tjänstelagret som mellanhand. I detta förfarande är XML-bäraren av själva förfrågan.



Figur 6. Tjänstelager baserat på SOA (Newcomer et al 2005).

- Tjänstelagrets säkerhetsmodell ansvarar för att rätt process får tillgång till rätt tjänst och har rätt tillstånd, dessutom förenklar den för processerna vid hanteringen av de olika säkerhetskraven från applikationerna under.
- Tjänstelagrets managementmodell genererar statistik gällande nyttjande av olika tjänster. Dessa statistiska data kan i sin tur utnyttjas av BAM-verktyg som är en del av processlagret.

Komplexitetsproblem

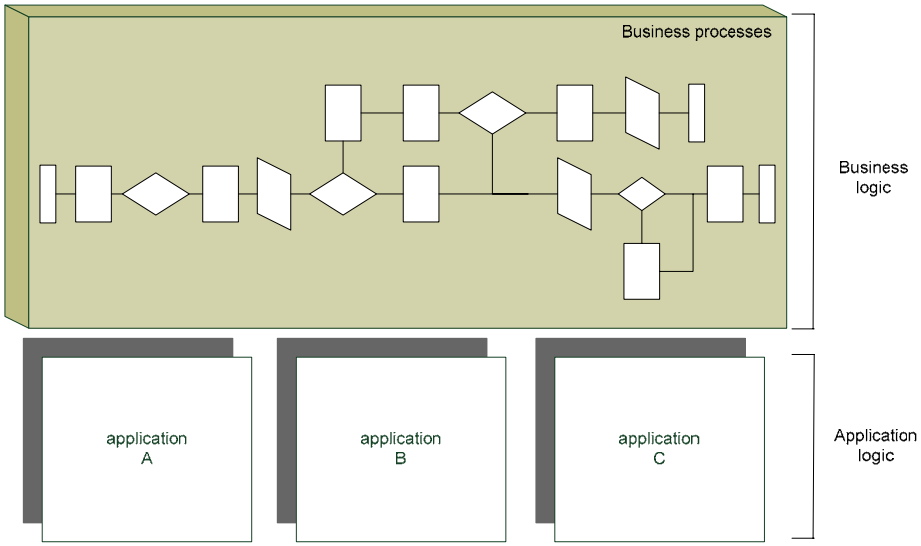
Tidigare tillämpades inte möjligheterna som SOA och Web Services skapar. Om en verksamhet använder sig av någon form av BPM utan ett tjänstelager, blir resultatet ett onödigt komplext och instabilt system. Komplexiteten ökar tack vare att det måste vara direkta kopplingar från processernas behov ned till de mer grundläggande applikationerna (se figur 7 och 8). Detta görs genom att skapa flera olika gränssnitt som i sin tur är definierade av applikationen i fråga (t.ex. API:er, meddelanden, eller databastabeller). För att gå vidare i tillämpningen av applikationerna för dessa processer behövs en bättre förståelse för vad applikationerna erbjuder i form av gränssnitt och anpassningar i processerna för att hantera bristande definitioner gällande vad applikationerna erbjuder. Detta kan i sin tur innebära förändringar av den specifika data som behövs, för att skapa s.k. kanoniska dataobjekt som affärsprocessen behöver tillgång till (Newcomer et al 2005).

Denna tillämpning av applikationer är inte bara mer komplex som visades ovan. Den är dessutom mer bräcklig. Detta på grund av att det behövs en starkare koppling mellan själva applikationen, som utför den beräkning processen ovan är beroende av.

Bräckligheten skapas här när det uppstår behov att uppdatera applikationer, vilket i sin tur skapar problem för processernas förmåga att tillämpa den förändrade applikationen (Newcomer et al 2005).

3.4 Övergripande principer för tjänsteorientering

En verksamhets sammantagna affärs- och applikationslogik är kontinuerligt föränderlig genom både externa och interna faktorer. Erl (2005) delar in logiken i två skilda delar, affärslogik och applikationslogik, se Figur 7.



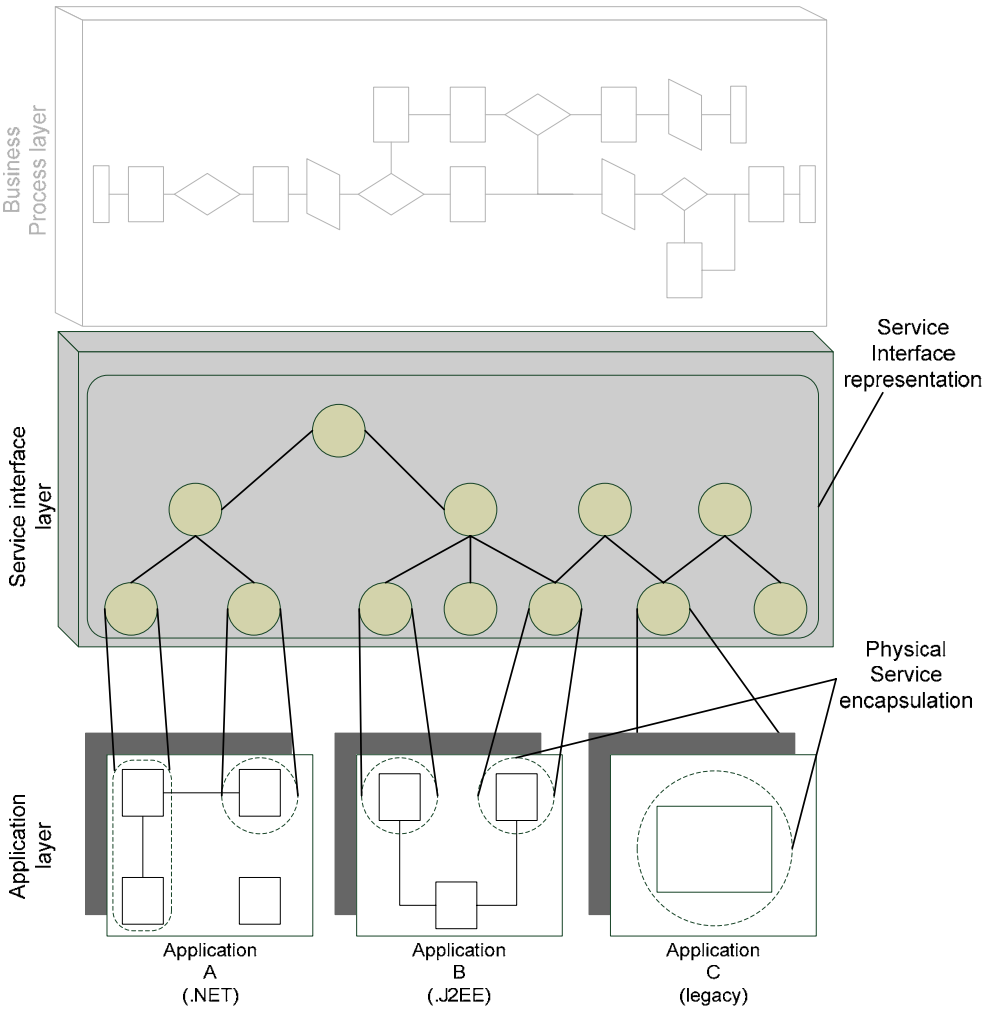
Figur 7. Illustration av affärs- och applikationslogik (Erl T, 2005).

Affärslogiken är den delen som affärsverksamheten i grunden representerar, och är ofta en dokumenterad modellering utav flödet i en organisation. Applikationslogiken är en automatiserad implementation av affärslogiken organiserad i olika tekniska lösningar. Applikationslogiken kan vara inköpt eller bestå av egentillverkade system som skall stödja affärsverksamheten. (Erl 2005)

Tjänsteorientering har sina rötter i en software engineering teori som heter ”separations of concerns”. Teorin grundar sig på idén att det lönar sig att bryta ner stora problem i delproblem för att lättare kunna lösa dem. På samma sätt kan man bryta ned logik, som till exempel affärsprocesser, i mindre delar för att lokalisera specifika händelser i processen. Teorin har tidigare använts i utvecklingsplattformar. Två exempel är objektorienterad programmering och komponentbaserad programmering som utnyttjar ”separations of concerns” genom att använda objekt, klasser och komponenter (Erl 2005).

Tjänsteorientering är ett annat sätt att realisera teorin om ”separations of concerns”. De principer som tjänsteorientering vilar på stödjer också teorin, samtidigt som de kan fungera som vägledning vid byggandet av en SOA. De teorier som Erl (2005) presenterar är inga standardiserade principer eftersom det ännu inte finns någon standardiserad definition av SOA. Erl (2005) hävdar dock att de är vida accepterade

av de flesta intressenter och tittar man närmare på dem upptäcker man att de härrör ifrån teorin om ”separations of concerns” antingen direkt eller indirekt.



Figur 8. Illustration av tjänstelager positionerat mellan affärslogik och applikationslogik (Erl 2005).

Principerna som Erl (2005) nämner är följande:

- Tjänster är återanvändbara – Oavsett om direkt återanvändning existerar så är tjänster designade för möjligt återanvändning.

- Tjänster delar ett gemensamt kontrakt – För att tjänster skall kunna interagera, så behöver de ett gemensamt kontrakt som beskriver hur samt vilka tjänster som existerar.

- Tjänster är löst kopplade – Tjänster måste vara designade för att kunna interagera med varandra utan att behöva ha ett beroende dessa emellan.

- Underliggande logik för tjänsteabstrakt – Den enda del av en tjänst som är synlig inför övriga tjänster, är det som är definierat i tjänstekontraktet. All underliggande logik är osynlig utåt, det vill säga det som tjänster utför för att kunna presentera resultatet. Tjänsten talar inte om hur den gör bara att det blir gjort.

- Tjänster är sammansättningsbara - Tjänster kan vara sammansatta av fler tjänster. Detta gör att logiken kan presenteras på olika nivåer, samt att det möjliggör skapandet av tjänstelager.

- Tjänster är autonoma – Den underliggande logiken i en tjänst existerar inom fasta gränser. Tjänsten har kontrollen inom dessa gränser och är inte beroende av någon annan tjänst för att kunna utföra det som ligger inom dessa.

- Tjänster är tillståndslösa – Tjänster skall inte behöva ta hand om tillståndsinformation som kan förhindra den att vara löst kopplad. Tjänster bör alltid designas för att uppnå maximal tillståndslöshet, även om det betyder att behandling av tillstånd förflyttas till annat håll.

- Tjänster är upptäckbara – Tjänster bör tillåta att deras beskrivningar kan upptäckas och förstås av människor och tjänsteförfrågare.

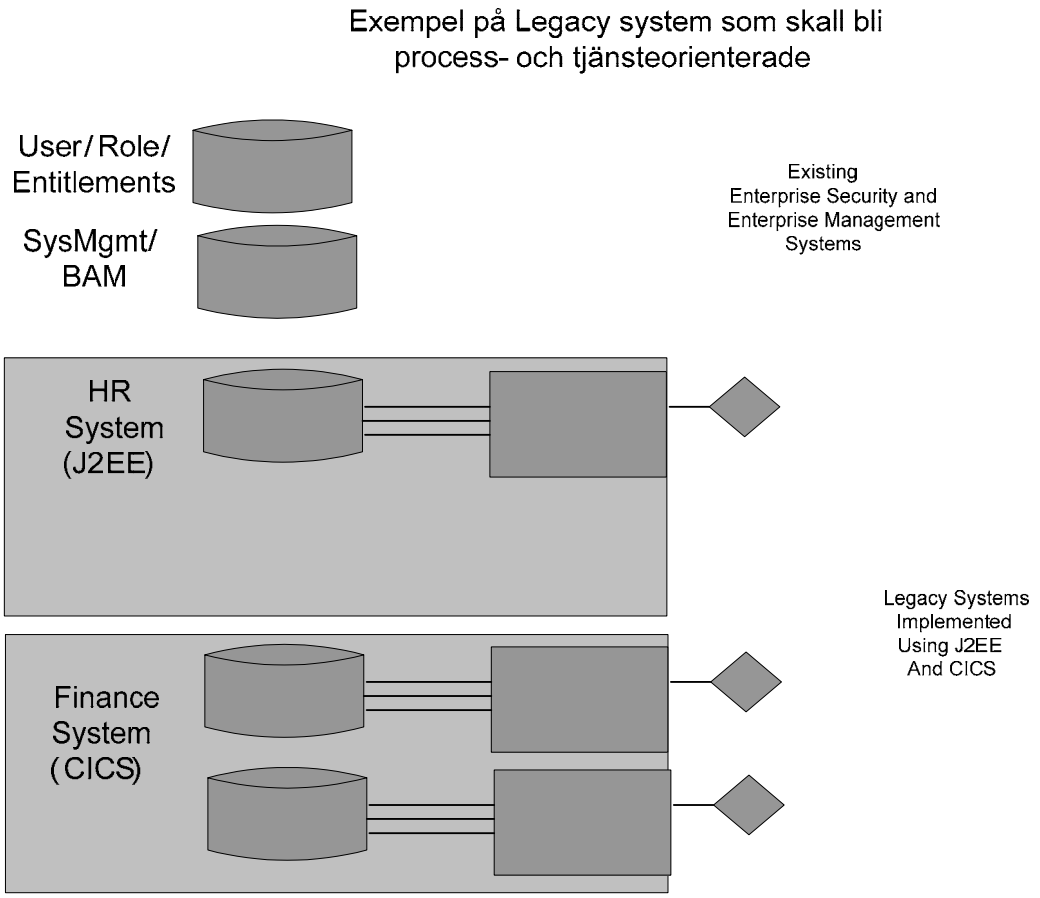
Av dessa åtta principer som Erl (2005) presenterar, anser han att fyra av dem; behovet av att dela ett gemensamt kontrakt, lös koppling som möjliggör oberoende, abstrakt, osynlig underliggande logik samt autonom, avgränsad underliggande logik; är grunden i en SOA.

3.4.1 Fiktivt exempel på design av tjänster

Anledningen till av vi väljer att ta med detta exempel är att göra det lättare för läsaren att se kopplingen mellan BPM, SOA och Web Services, och hur dessa används för design av en lösning med hjälp av dessa tre begrepp. Exemplet baserar sig på hur verksamhetstjänster görs tillgängliga för yttre och inre användare baserad på ett så kallad Legacy-system (ursprungssystem) (Newcomer et al 2005).

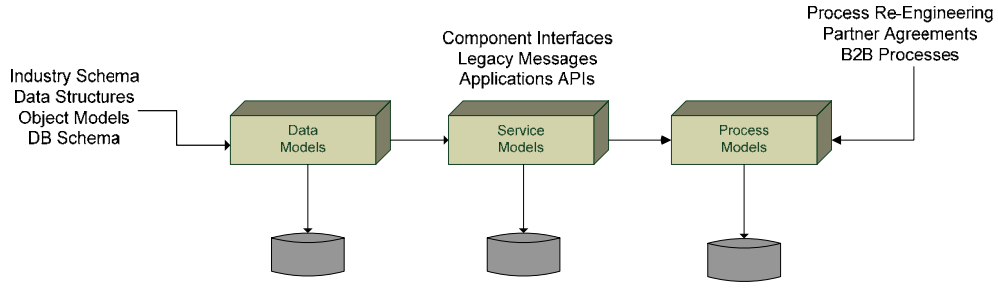
Den första figuren, se Figur 9, illustrerar det initiala systemet som består av ett grundsystem, HR-system, ett befintligt finanssystem, ett redan befintligt säkerhetssystem för verksamheten som hanterar user/role/entitlement information och till sist ett redan existerande verksamhetshanteringssystem (Newcomer et al 2005).

HR-systemet implementeras med hjälp av J2EE teknologi och består i sin tur av applikationsdatabaser och en objektmodell. Den förser även en EJB (Enterprise Java Bean)-baserat gränssnitt. Finanssystemet körs på själva servern/stordatorn, den implementeras genom användandet av CICS transaktioner mot själva databasen och är tillgänglig genom klientapplikationer via exempelvis WebSphere MQ (Newcomer et al 2005).



Figur 9. Exempel på grundsystem som skall bli process- och tjänsteorienterat (Newcomer et al 2005).

Första steget mot att använda sig av återanvändbara tjänster i SOA, är att definiera verksamhetsbaserade domändata-, tjänste-, och processmodeller, se Figur 10.



Figur 10. Data-, tjänste- och processmodeller (Newcomer et al 2005).

- **Datamodellen**: Definierar verksamhetsdata som kommer att bli utbytt mellan tjänster och kommer att bli tillgängliga till tjänsteförfrågningar. Modellen inkluderar även data definitioner (XML-scheman, datavalideringsregler (XML-schemabegränsningar och XPath), och datatransformationsregler

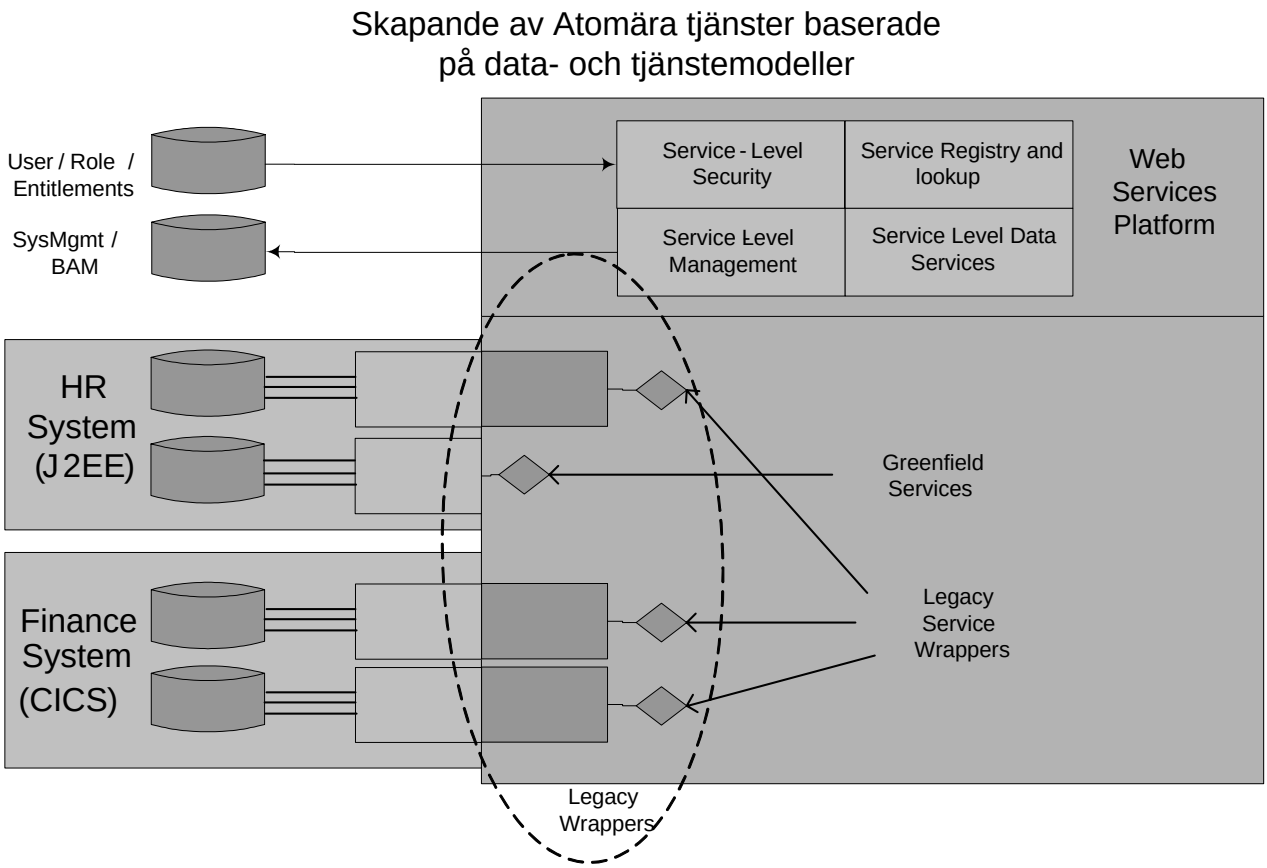
(XLST). Idealiskt skall tjänstenivåmodellen skapas efter existerande industristandard (schema), detta är inte alltid möjligt och i sådana lägen måste man vara mycket försiktig, så att sann och enhetligt språk/datakonvention upprätthålls för att i sin tur bevara applikations- och teknikoberoende.

- **Tjänstemodellen**: Definierar de så kallade tjänstekontrakten (WSDL alternativt även WS-policy) för tjänsterna i verksamhetslinjen. Tjänstekontrakten definierar följande.
  Input och outputparametrar för själva tjänsten i fråga, detta baseras i sin tur på det dokument som definierats av datamodellen.
  Säkerhetsprofilen för tjänsterna (rättigheter, tillgänglighetslista, vad som är hemligt och till slut tillgängligt).
  Kvalitén på tjänsten för tjänsterna, exempel kan vara (prioritet, leverans-säkerhet och transaktionskaraktär).
  Överenskommelser när det gäller tjänster (svarstider och tillgänglighet).
  Det finns vanligtvis två breda kategorier av tjänster: atomära tjänster (tjänster som i sig inte är sammansättningar av andra tjänster), och sammansatta tjänster (tjänster som i sig är sammansatt av två eller fler tjänster). Precis som datamodellen är tjänstemodellen baserad på redan existerande industri-orienterade tjänstedefinitioner från tidigare system, men precis som tidigare kan dessa vara tvungna att härledas från underliggande komponentgränssnitt, om en sådan definition (industriorienterad tjänstedefinition) inte redan existerar.

- **Processmodellen**: Processmodellen definierar verksamhetsprocesserna som implementeras genom att använda sig av exempelvis WS-BPEL, för att få fram densamma. Varje processdefinition inkluderar processuppgiften, kontrollflödet, dataflödet och andra verksamhetsregler som är associerade med processerna (se figur 6 för illustration på hur detta kan se ut) (Newcomer et al 2005).

Processmodellen är normalt skapad på bas av redan existerande processdefinitioner. Normalt skapas datamodellen först, som i sin tur fungerar som bas för att definiera tjänstemodellen och tillslut skapas processmodellen baserad på tjänstemodellen.

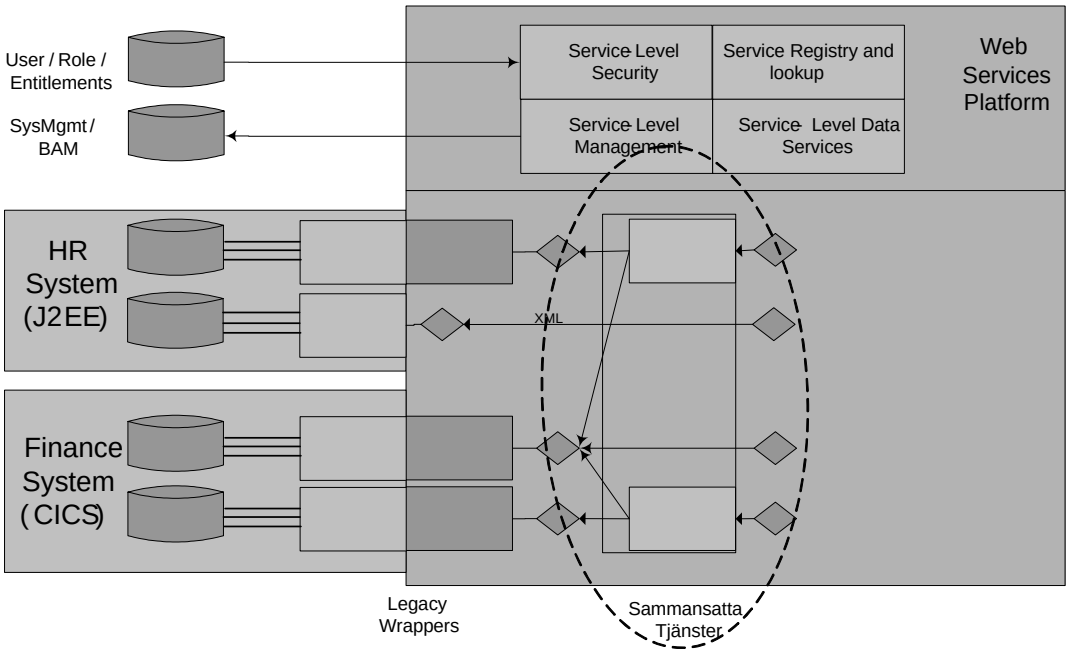
Värt att notera är att detta bara är ett exempel, dessa modeller kan skapas i vilken ordning som helst, och sker ofta iterativt. Nästa illustration, se Figur 11, visar ett exempel på atomära tjänster baserade på data- och tjänstemodellerna (Newcomer et al 2005).



Figur 11. Skapande av atomära tjänster baserade på data- och tjänstemodeller (Newcomer et al 2005).

I exemplet är tre av tjänsterna definierade genom att skapa ett så kallat tjänstehölje för grundsystemet. Tjänstehöljet skapar ett Web Service-gränssnitt (SOAP och WSDL, se Bilaga 1) mot ett grundsystem. Detta gränssnitt ansvarar för att ta emot inkommande SOAP-meddelanden, översätta dessa till ett format som grundsystemet kan förstå, för att sedan skicka själva förfrågan till rätt del. I detta exempel är en av tjänsterna definierad genom att implementera en ny J2EE komponent och exponera denna som en Web Service. Figuren visar även själva Web Service-plattformen som förser tjänsterna med definitioner på registret, säkerhet och management av de såkallat atomära tjänsterna (Newcomer et al 2005).

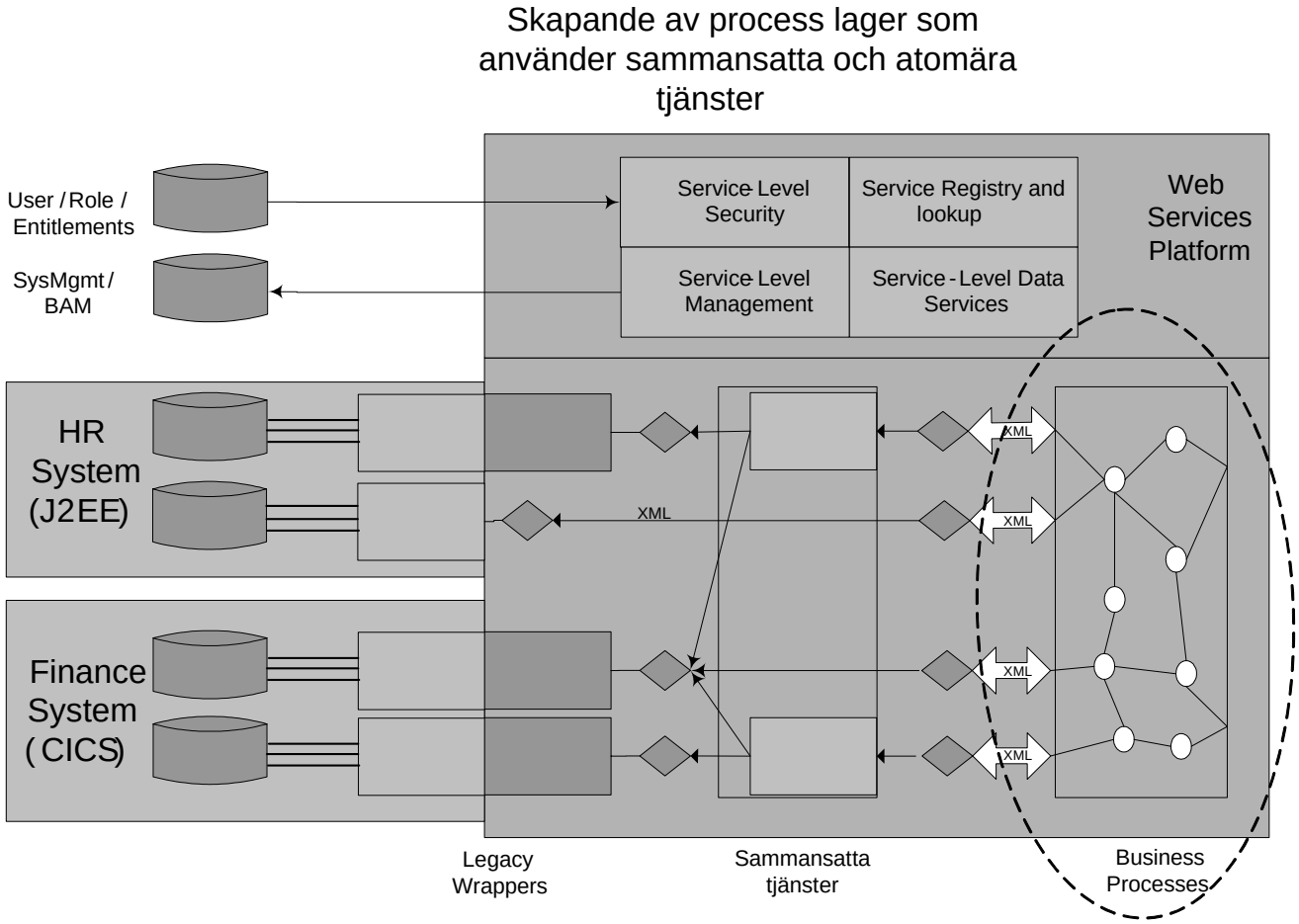
I Figur 12 visar vi nästa steg där vi definierar de så kallade sammansatta tjänsterna. Sammansatta tjänster består av flera olika Web Services (Newcomer et al 2005).



Figur 12. Skapande av sammansatta Web Services (Newcomer et al 2005).

Sammansatta tjänster är precis som vilken annan Web Service, i den meningen att de har ett WSDL tjänstekontrakt och anropas genom så kallade SOAP-meddelanden. Ett sätt att skapa sammansatta tjänster är att använda sig av Web Service orkestrering och WS-BPEL. Här användes oftast en produkt som stödjer grafiskt användargränssnitt. Den genererar också en WS-BPEL processdefinition som även kan exekveras. Fördelarna att sätta samman Web services med hjälp av WS-BPEL är enkelheten och flexibiliteten. WS-BPEL gör det möjligt att förändra sammansättningen av tjänsterna utan att behöva skriva till någon ny kod. I vissa fall kan det dock vara att föredra med explicit kod, detta kan vara aktuellt vid väldigt simpla sammansatta tjänster eller prestandakänsliga tjänster (Newcomer et al 2005).

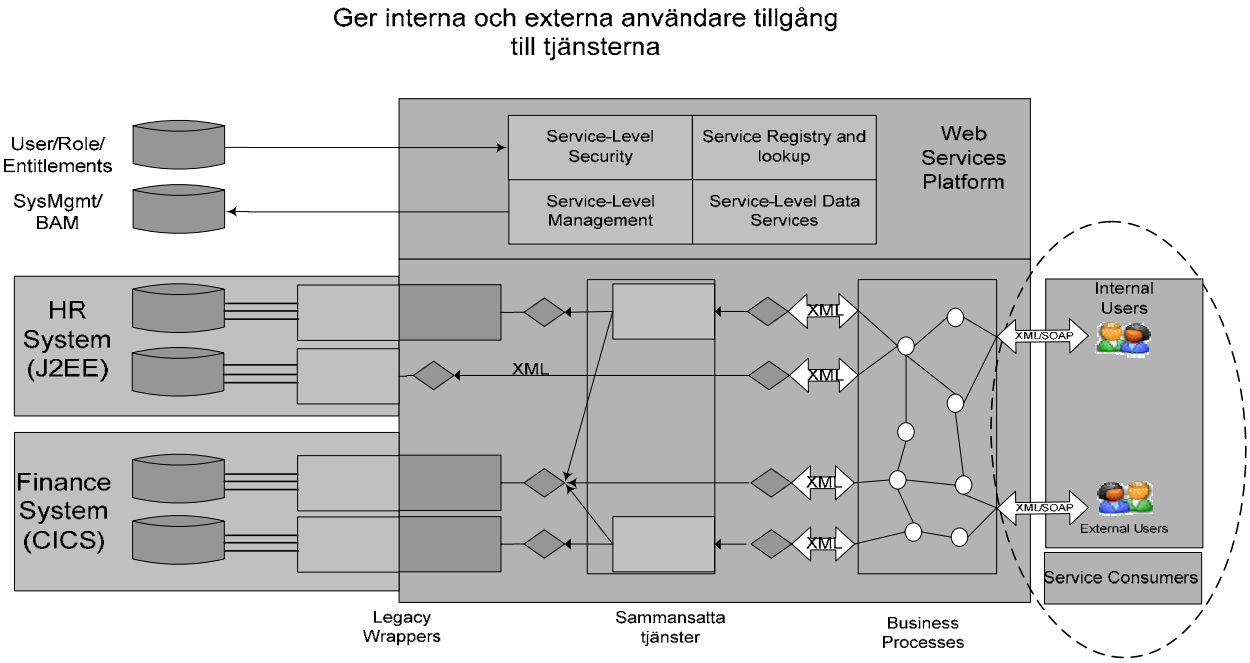
I Figur 13 ser vi själva processlagret. Vad vi ser är olika processer som är sammankopplade i varandra. Varje nod är en process. Verksamhetsprocesserna är en komplex del som i sig består av flera lager av processer/funktioner. Det normala är att flera olika typer av användare eller system använder sig av dessa processer. Varje process i sig är också en tjänst, detta betyder alltså att den kan anropas av en annan tjänst eller process (Newcomer et al 2005).



Figur 13. Processlager som använder sammansatta och atomära tjänster (Newcomer et al 2005).

Figur 14 visar nästa och sista lagret på vår SOA-modell. I modellen ser vi hur externa och interna användare får tillgång till våra tjänster (exempelvis Web Services), i detta inkluderas verksamhetsprocesser i form av tjänster. Vem som helst med rätt behörighet (även andra system) kan här komma åt tjänsterna när som helst, och varifrån som helst.

Ibland synliggörs enbart ett urval av tjänsterna till externa användare detta är upp till var och en att bestämma, oftast krävs ett mer omfattande säkerhetstänkande i hanteringen av externa användare, till skillnad från interna användare (Newcomer et al 2005).



Figur 14. Exempel på hur interna och externa användare får tillgång till systemet genom tjänsterna (Newcomer et al 2005).

3.5 Detaljerade anvisningar för SOA analys och design

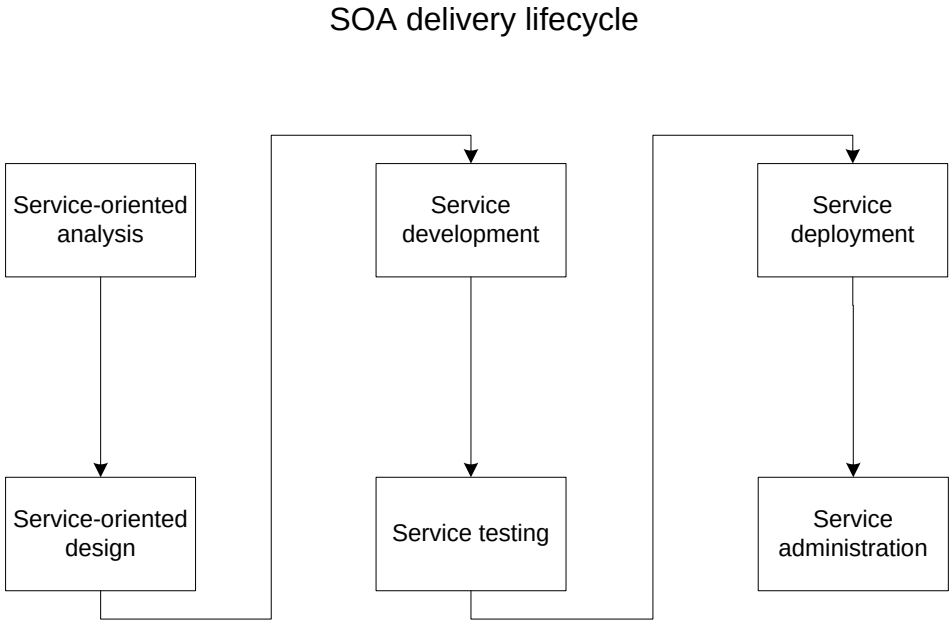
Följande del tar upp ett förslag på hur man kan gå tillväga för att på ett effektivt sätt analysera samt modellera nödvändiga tjänster. Därefter följer några av designrekommendationerna av Erl (2005). Innan själva analysen börjar är det dock lämpligt att fundera över hur man skall planera utvecklingsprojektet (Erl 2005)

3.5.1 Integrationsstrategier

Fastän ett utvecklingsprojekt för tjänsteorienterade lösningar på ytan liknar utvecklingsprojekt för andra IT-lösningar, så behövs det justeras i viss mån. Det beror på att traditionella utvecklingsprojekt saknar stöd för att på rätt sätt kunna utveckla och positionera tjänsterna i en korrekt struktur. Erl (2005) presenterar i Service Oriented Arcitecture, livscykeln för ett SOA-system. Livscykeln består av ett antal olika faser som visas på bilden här nedan.

Av Figur 15 framgår att det bara är de två första faserna som har namn med ordet tjänsteorienterad inblandat. Detta är på grund av att de andra faserna tar upp konstruktionen av själva tjänsterna, medan de två första tar upp hur tjänsterna skall struktureras. Figuren ovan visar ett generellt tillvägagångssätt för hur man kan införa en tjänste-orienterad lösning. Stegen bör sedan anpassas efter verksamhetens syfte och mål med projektet. Ett bra sätt är att försöka hitta en balans mellan långsiktiga

mål och kortsiktiga krav. Tre vanliga strategier har dykt upp för att möta detta problem, top-down, bottom-up, agile (meet in the middle) (Erl 2005).



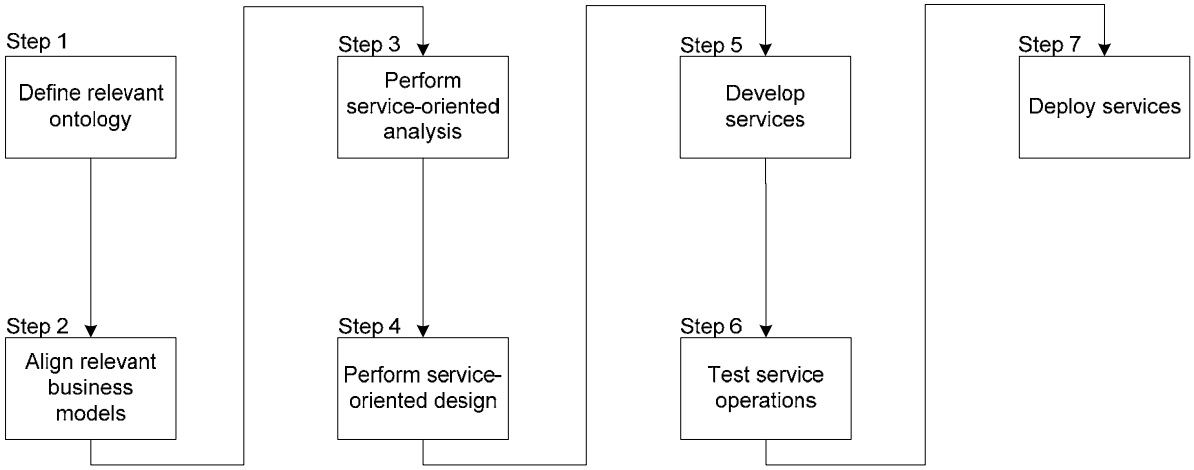
Figur 15. Visar hur tjänsteorienterad lösning införs (Erl 2005).

För att kunna kartlägga hur implementation generellt går tillväga vid införandet av tjänsteorienterade lösningar så anses det relevant att ta upp dessa strategier för att ta redan på om företagets metoder passar in på någon av strategierna.

Top-down strategi
Top-down strategin, se Figur 16, innebär att man utgår ifrån verksamhetens affärslogik och strömlinjeformar den efter det tjänsteorienterat tankesättet. Man utgår ifrån att analysera och kartlägga processerna för att därigenom anpassa de tjänster som framöver skall tas fram för att stödja processernas behov. Top-down strategin är lämplig vid ett storskaligt SOA-projekt som oftast resulterar i många återanvändbara tjänster (Erl 2005).

Fördelar och nackdelar med top-down strategi
Fördelen med top-down strategin är, enligt Erl (2005) att SOA-projektet ofta resulterar i välstrukturerad tjänstearkitektur. Varje tjänst har genomgått en noggrann analys, vilket ökar chanserna för potentiell återanvändbarhet och för strömlinjeformad struktur. Det medför att organisationen får en hög flexibilitet då tjänsterna är strukturerade så att en låg koppling mellan dem råder (Erl 2005).

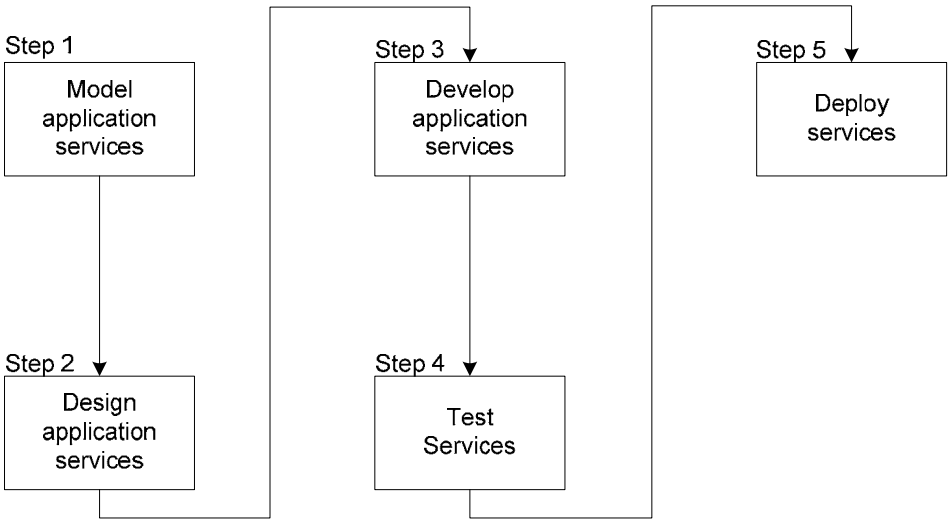
Nackdelen i sin tur är att top-down strategin ofta innebär mycket tid och pengar. Verksamheterna måste ofta investera mycket tid och pengar på förstudier av företaget och dess processer, vilket kan ta väldigt lång tid beroende på storleken av verksamheten ifråga, vilket i sin tur leder till att det dröjer innan man märker några resultat av projektet (Erl 2005).



Figur 16. Olika steg som bör ingå i en top-down analys (Erl 2005).

Bottom-up strategi

Bottom-up strategin, se figur 17, innebär att man utgår helt ifrån existerande system och tar fram tjänster som är tänkta att stödja redan befintliga funktioner. Web Services eller liknande byggs efter behov för att möta kortsiktiga krav. Oftast är integration en drivande faktor när man tillämpar denna strategi.



Figur 17. Olika steg som bör ingå i en bottom-up analys (Erl 2005).

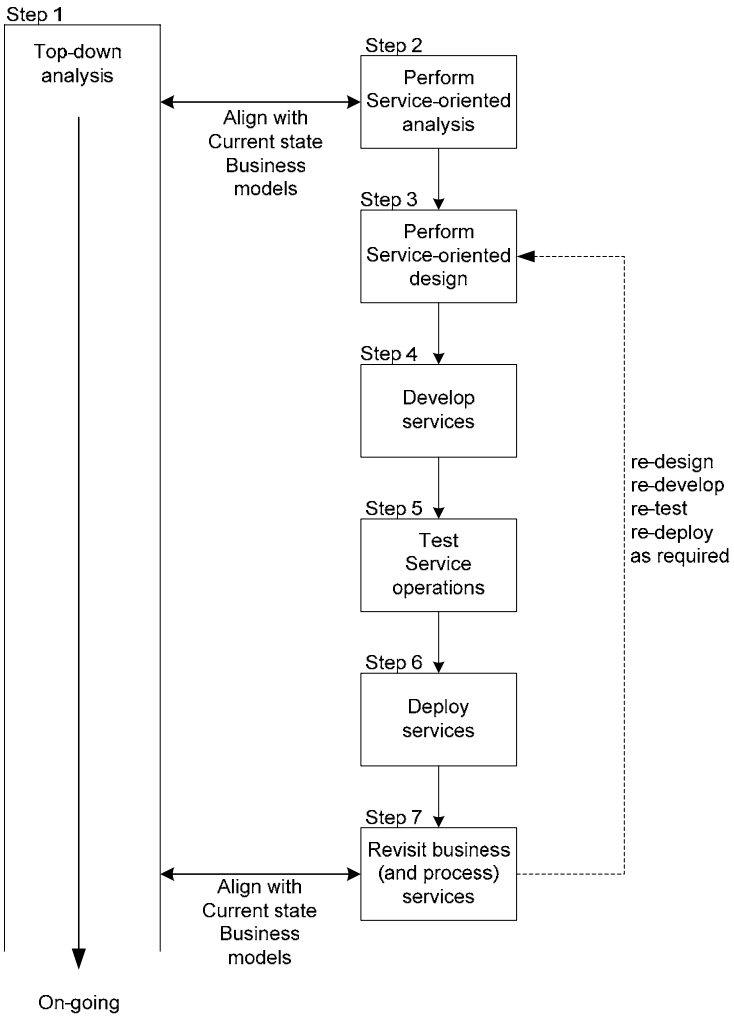
Fördelar och nackdelar med bottom-up strategi

Enligt Erl (2005) så är bottom-up strategin den vanligaste metoden hos verksamheter som idag bygger Web Services-lösningar eller liknande på redan existerande system. Det medför att strukturen på systemet förblir oförändrad, och principerna för tjänsteorientering sällan tas i beaktning. Eftersom verksamheterna, enligt Erl (2005) i stort sett aldrig tar de tjänsteorienterade principerna i beaktning så är inte bottom-up

strategin ett bra verktyg för att införa en SOA, eftersom man inte infört någon egentlig tjänsteorienterad lösning. Erl (2005) anser att detta upptäcker verksamheterna när de börjar med större SOA-projekt. Vad som händer är att trots att tjänsterna har gett bra kortsiktiga resultat så behövs de ofta göras om vid införandet av en större SOA. (Erl 2005).

The agile strategy

Erl (2005) föreslår att man bör hitta en mellanväg mellan de två ovan nämnda strategierna i syfte att kunna få ta del utav de kortsiktiga resultaten samtidigt som man anpassar verksamheten så att den blir tjänsteorienterad. En tänkbar strategi för detta är vad Erl (2005) kallar för ”The agile strategy”, eller ”meet in the middle”, se Figur 18. Den går ut på att man skall utgå ifrån top-down analysen, med startfokus på de processerna som verksamheten finner viktigast och är i behov utav kortsiktiga resultat. Samtidigt som analysen pågår så börjar man designa och bygga tjänsterna utefter så långt man har kommit i analysen (Erl 2005).



Figur 18. Illustration av hur en agile strategy kan se ut (Erl 2005).

Fördelar och nackdelar med the agile strategy
The agile strategy är ett försök att ta en mellanväg mellan top-down och bottom-up strategierna. Lyckas detta fås både kortsiktiga resultat, samt att verksamheten struktureras efter ett tjänsteorienterat mönster. En kombination av top-down och bottom-up ställer höga krav på dem som är involverade i projektet eftersom det måste tas med i beräkningen att både kortsiktiga och långsiktiga mål skall gå hand i hand med varandra, så att de lösningar som tas fram inte motarbetar varandra längre fram under projektets gång. Det krävs också ett större underhållsarbete då det även måste ingå kontroller av tjänsterna, så att dessa stämmer överens med de affärsprocesser som de skall stödja (Erl 2005).

3.5.2 Tjänsteorienterad analys
Huvuduppgiften med den tjänsteorienterade analysen är att ta reda på hur kraven kan representeras genom tjänsteorienterad struktur. De två huvudfrågorna under analysfasen presenteras nedan, och de svar som erhålls beror på hur storskaligt projektet är.

1. Vilka tjänster behövs?
2. Vilken logik (applikations- eller affärslogik) skall de olika tjänsterna kapsla in?

Enligt Erl (2005) så finns det ett antal delmål som skall uppfyllas under analysfasen. Dessa är:

- Definiera preliminära operationskandidater – de operationer som skall kunna utföras i systemet
- Gruppera de preliminära operationskandidaterna i logiska kontexter
- Definiera preliminära tjänstegränser så att de inte överlappas med existerande eller planerade tjänster
- Identifiera funktioner med potentiell återanvändbarhet
- Säkerställ att inkapslad logik är korrekt för dess tilltänkta användning
- Definiera kända preliminära kompositionsmodeller

Erl (2005) är noga med att påpeka att ovan nämnda punkter inte på något sätt är tänkta att ersätta en verksamhets redan beprövade analysmetoder. Men däremot är de tänkta att fungera som ett komplement vid införandet av en tjänsteorienterad lösning.

Analysmodellen kan användas på olika nivåer beroende på vilken strategi man valt, och hur omfattande projektet är tänkt att vara och vilka tjänstelager som skall införas med lösningen. De olika tjänstelagren kräver olika slags modeller, därför är det viktigt att innan man börjar med analysen, har valt strategi så att man vet omfattningen på projektet. Andra frågor som bör vara besvarade innan man börjar med analysen är:

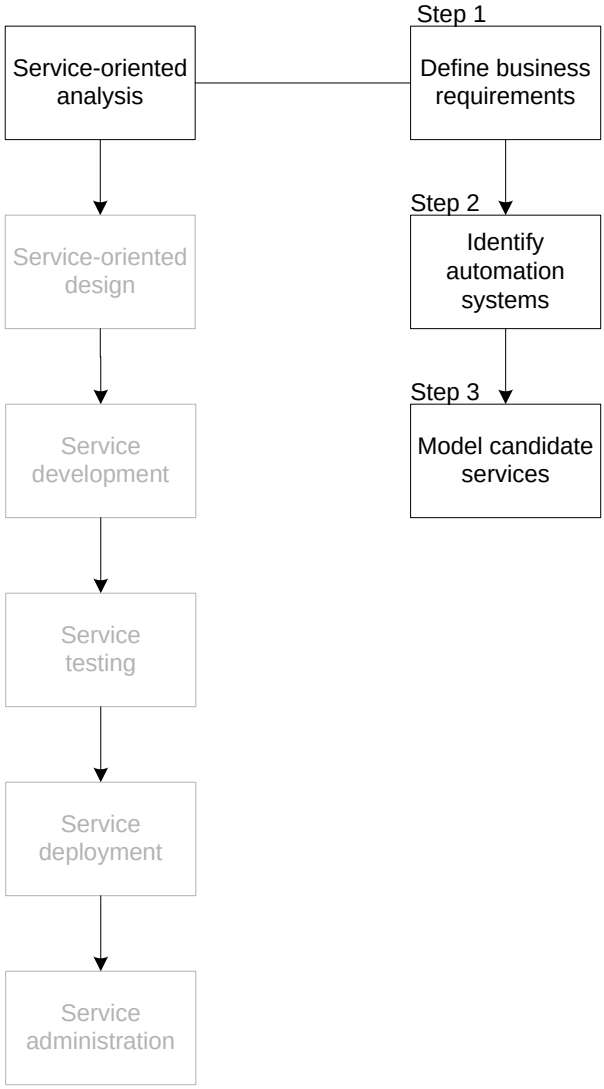
1. Vilket förarbete behövs för att ta fram affärsmodeller över de processer som tjänsterna skall stödja?
2. Vilka modelleringsverktyg skall användas i analysfasen för att modellera tjänsterna?
3. Kommer analysen vara förarbete på ett senare större projekt?

3.5.2.1 Analysfasens tre steg

Analysfasen består enligt Erl (2005)av tre steg; 1) definiera affärsautomatiserings-kraven,2) identifiera existerande automatiseringssystem och 3) modellera kandidattjänster . Figur 19 ger en överblick över proceduren.

Steg 1: Definiera affärsautomatiseringskraven

Enligt Erl (2005) bör man börja med att analysera kravspecifikationen. Detta bör göras på det sätt som verksamheten är van vid. Eftersom denna analys syftar till att ta fram information angående vilka tjänster som behövs för att stödja en tjänsteorienterad lösning så bör bara de krav, som anses vara relaterade till just detta, tas i beaktning.



Figur 19. Illustration av de övergripande stegen i en tjänsteorienterad analys. Steg ett och två omfattar till största delen insamling av information som skall ligga till grund för modelleringen i steg 3 (Erl 2005).

Utifrån kravspecifikationen skall man identifiera de processer som skall stödjas. De processerna som nu dokumenteras, är de processerna som man skall utgå ifrån i steg tre när det är dags att modellera själva tjänsterna.

Steg 2: Identifiera existerande automatiseringssystem

Existerande applikationer, som på något sätt redan stödjer de krav som tidigare identifierades, skall tas fram. Även om analysen i sig inte inkluderar hur tjänsterna skall inkapsla eller ersätta den applikationslogiken, så hjälper det att veta vilka applikationer som påverkas av tjänsterna. Informationen hjälper även till att identifiera applikationstjänsterna, det vill säga de tjänster som direkt skall kommunicera med existerande applikationer. Detaljerna kring exakt hur tjänsterna skall hantera dessa applikationer tas upp senare i designfasen.

Enligt Erl (2005), är detta steget mest till för att understödja storskaliga SOA-projekt. Dock så kan informationen som tas fram i detta steg vara väldigt hjälpsam för små projekt också, men ju mindre projektet är desto mindre kraft bör man lägga på just detta steg.

Steg 3: Modellera kandidattjänster

Utifrån informationen som samlats i de två tidigare stegen skall man sedan modellera de preliminära tjänstekandidaterna. Dessa skall sedan grupperas i logisk kontext så att de blir lättöverskådliga. Tanken är dessa modeller skall resultera i en komponerad modell som skall representera all logik som skall representeras i den tjänsteorienterade lösningen.

3.5.2.2 Steg för att modellera kandidattjänster

Erl (2005) går vidare med att presentera 12 underliggande steg för att modellera tjänstekandidaterna.

Steg 1: Bryt ner affärsprocesserna

Det är viktigt att bryta ner processerna i på den lägsta nivå som går, även om det innebär en lägre nivå än vad som dokumenterats. Anledningen är att man skall inte skall kunna missa någon viktigt händelse/operation som kan inträffa.

Steg 2: Identifiera operationerna i affärstjänsterna

Ta bort de operationer som inte är relevanta eller inte kan bli automatiserade.

Steg 3: Separera orkestreringslogiken

Om verksamheten har beslutat sig för att ha ett orkestreringslager, så är detta steget

till för att identifiera och separera den logik som skall ingå i det lagret. Erl (2005) ger några exempel på vad detta kan vara:

- Affärsregler
- Villkor för händelser – eventuella villkor som skall uppfyllas innan en process sätts igång
- Undantag – eventuella undantag i processerna
- Sekvens – i vilken ordning processtegen utförs

Steg 4: Skapa affärstjänstekandidaterna

Se över processtegen och försök att gruppera stegen i olika logiska kontext. Varje gruppering är tänkt att bli en tjänstekandidat. Erl (2005) påpekar att det viktiga med detta steg inte är att alla processteg hamnar korrekt just för tillfället, utan för att få en överblick på vilka tjänster som eventuellt kommer att byggas.

Steg 5: Applicera principerna för tjänsteorientering

Detta steg är till för justering av de tjänstekandidaterna som man fick fram i det tidigare steget. Tanken är att se till så att tjänstekandidaterna följer de enligt Erl (2005) åtta principerna för tjänsteorientering som tidigare beskrivits i SOA-beskrivningen. Notera dock att vissa principer inte tas i beaktning förrän i designfasen.

Steg 6: Identifiera de vanligaste scenarierna

Gå igenom de olika tänkbara scenarierna som kan uppstå inom varje process. Använd de steg som tidigare tagits fram. Syftet med detta menar Erl (2005) är att ta reda på hur korrekta grupperingarna är. Det framkommer även ifall några steg har missats.
Steg 7: Se över tjänstegrupperingarna

Utifrån resultaten i föregående steg, se över de steg och grupperingar som kräver justeringar. Gå tillbaka i den mån det behövs. I vissa fall kan man behöva göra helt nya grupperingar.

Steg 8: Analysera applikationsprocesskrav

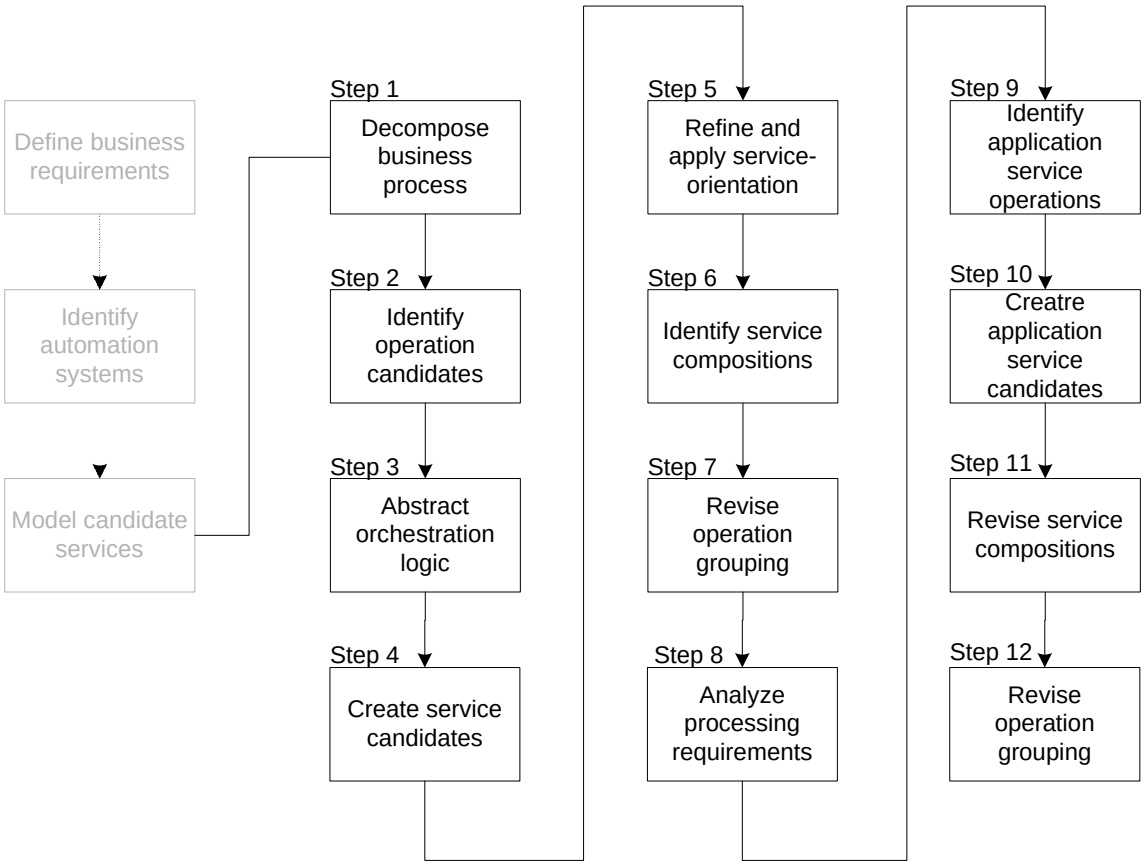
Detta och resten av stegen är enligt Erl (2005) till för stora och komplicerade affärsprocesser. I detta steg gäller det att identifiera vilka applikationer som stödjer vilka processer. Syftet är bland annat för att sålla fram de tjänster som kommer att utgöra applikationstjänstelagret. Varje process som tagits fram i tidigare steg bör genomgå en minianalys.

Det som man bör få veta är:

- Vilken underläggande applikationslogik måste exekveras för att händelsen som beskrivs i operationskandidaten skall utföras?

- Existerar den applikationslogiken som krävs för händelsen?
- Är applikationslogiken som krävs är spridd över fler system? Med andra ord krävs fler än ett system för att utföra händelsen

I detta steg används ”Steg 2: Identifiera existerande automatiseringssystem” som referens.



Figur 20. Illustration av de underliggande stegen som fungerar som guide vid modelleringen av tjänstekandidaterna (Erl 2005).

Steg 9: Identifiera operationerna i applikationstjänsterna

Bryt ner operationerna som applikationslogiken utför i små steg. När stegen namnges, se till så att de refererar till funktionen de utför och inte till processteget de understödjer.

Steg 10: Skapa applikationens tjänstekandidater

Gruppera stegen från föregående steg till en fördefinierad kontext. I detta fall bör enligt Erl (2005) vara logisk relation mellan operationskandidaterna. Denna relation kan till exempel vara:

- Gemensam association med ett specifikt grundsystem
- Association mellan en eller flera lösningskomponenter
- Logisk gruppering kring en speciell funktion

En del andra faktorer tas upp senare i designfasen så länge dessa representerar grupperingar ett applikationstjänstelager.

Steg 11: Se över tjänstegrupperingarna med applikationstjänsterna

Återblicka på Steg 6, där det gällde att gå igenom alla möjliga scenarion. Gör om det steget, men denna gång skall applikationstjänstekandidaterna ingå. Man bör då få en god bild av hur olika aktiviteter och händelser utförs i systemet. Det är viktigt att denna bild dokumenteras så att man kan ha fortsatt god överblick över systemet.

Steg 12: Se över operationsgrupperingarna bland applikationstjänsterna

Steg 11 brukar enligt Erl (2005) resultera i att man får förändra en del i grupperingarna samt definitioner. Det kan även förekomma att man får lägga till en del operationer och även i vissa fall helt nya tjänst

Erl (2005) anser att det är lämpligt att behålla samtliga tjänstekandidater dokumenterat. Om man vid senare tillfälle går igenom stegen på nytt så bör man ta existerande kandidater i beaktning innan man skapar nya.

Erl (2005) påpekar dock att det kan vara svårt att hitta återanvändbarhet på detta sätt, eftersom tjänstekandidaternas beskrivningar kan förändras mycket under fasernas gång och därmed försvinner den ursprungliga betydelsen. Erl (2005) vill därmed se detta val som frivilligt eftersom det mycket väl kan hända att man inte får någon direkt användning för det.

3.5.3 Tjänsteorienterad design

För att design av ett tjänsteorienterat system skall vara möjlig, är det viktigt att först ha gjort en analys, som är beskrivet i tidigare stycke hur det går till. Erl (2005) presenterar Service-oriented design som är den process där tjänstekandidaterna omvandlas till konkreta tjänster som sammansätts på ett sätt som de är tänkta att stödja en affärsprocess. De frågor som bör kunna besvaras under designprocessen är:

- Hur kan fysiska tjänstegränssnitt definieras utifrån de tjänstekandidaterna som togs fram under analysfasen?
- Vilka SOA-drag vill vi realisera och stödja?
- Vilka standarder och tilläggsmjukvara kommer att krävas av vår SOA för att implementera de planerade tjänsterna och SOA-dragen?

Huvudmålen med designprocessen är:

- Fastslå kärnan för arkitekturella utvidgningar
- Gränsdragning för arkitekturen
- Identifiera krav på designstandarder
- Definiera design på tjänsteprotokoll
- Identifiera möjliga tjänstesammansättningar
- Skapa resursstöd för tjänsteorienterade principer

3.5.3.1 Designfasens fem steg

Nedan beskrivs designfasens fem steg, se även Figur 20, kortfattat eftersom flertalet steg innehåller en mängd ytterligare understeg som kan uppfattas som alltför detaljerade för studiens syfte.

Steg 1 – Bestäm sammansättningen av SOA

Erl (2005) anser att varje SOA är unik. Då alla SOA delar många gemensamma teknologier så finns det mycket utrymme för anpassning för varje enskild SOA. Steg 1 kan delas in i tre underpunkter:

- Välj tjänstelager.

Det som skall göras är att först bestämma en konfiguration för tjänstelagren som kommer att representera en standardiserad logisk definition inom arkitekturen. Detta steg kompletteras av att studera tjänstelagerkandidaterna ifrån analysfasen.

- Positionera SOA-standarder.

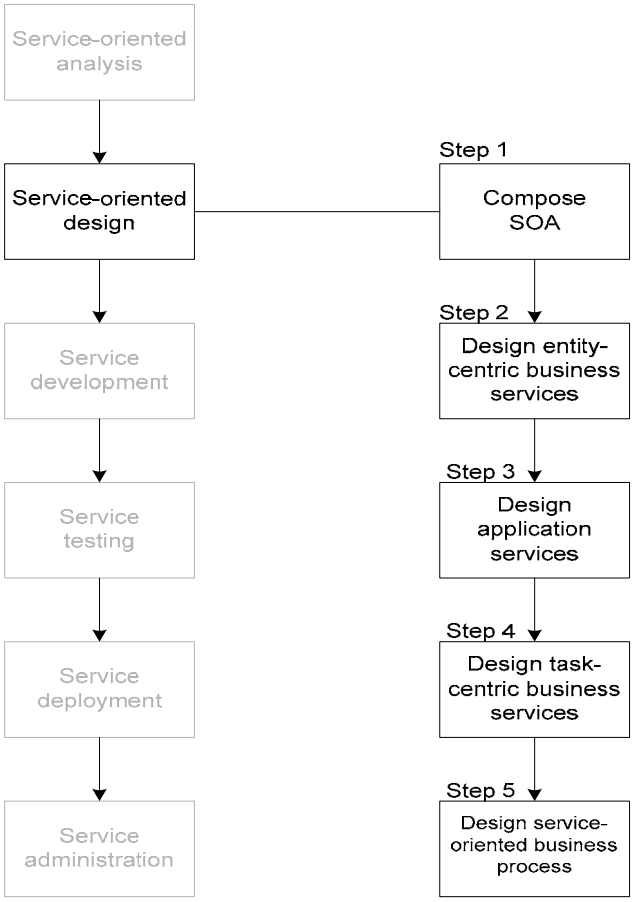
Här gäller det att bestämma vilka standardteknologier som den tjänsteorienterad arkitekturen skall bestå utav.

- Välj SOA-tillägg.

Här skall man bestämma SOA-egenskaper som skall stödjas i den tjänsteorienterade arkitekturen.

Steg 2 – Designa entitetstjänster

De affärstjänster som kretsar kring entiteter är de tjänster som är mest autonoma. Deras uppgift att representera dataentiteter (till exempel kund, order), som är definierade inom en verksamhet. Erl (2005) anser att eftersom de är mest autonoma bör de även designas före de andra tjänsterna.



Figur 21. Illustration av de steg som ingår i designfasen (Erl 2005).

Steg 3 – Designa applikationstjänster

Applikationstjänstelagret är en ren tjänsteorienterad abstraktion av verksamhetens tekniska miljöer. De personer som bör vara involverade i utvecklingen av dessa tjänster är först och främst de personer som förstår denna miljö bäst. Applikationstjänsterna är främst de tjänster som betraktas som mest återanvändbara, eftersom de ofta är finmalda med några få operationer.

Steg 4 – Designa funktionstjänster

Affärstjänster som kretsar kring specifika uppgifter kräver ofta mindre arbete än applikationstjänster, eftersom återanvändning inte är någon hög prioritering för dessa.

Därför används nästan enbart de operationskandidaterna som togs fram under analysfasen när dessa designas.

Steg 5 – Designa affärsprocesser

Detta steg är tänkt som ett stöd för att designa ”orkestreringstjänstelagret”. Fördelen med att använda ett orkestreringslager är enligt Erl (2005):

- Applikations- och affärstjänster kan fritt designas att vara processagnostiska och därmed återanvändbara
- Affärslogiken blir centraliserad istället för att bli utspridd bland ett otal olika tjänster

3.5.4 Designrekommendationer och riktlinjer

Erl (2005) anser för att lyckas med att införa en SOA i verksamhet så är det oerhört viktigt att standardisera så mycket som möjligt av tjänsterna. Nedan följer ett par rekommendationer ifrån boken *Service-Oriented Arcitecture Concepts, Technology and Design*.

Att namnge tjänsterna är ekvivalent med att namnge en IT-infrastruktur. Det är därför viktigt att tjänsterna är namngivna på ett sådant sätt att de är självbeskrivande. Namnstandarder är därför viktigt för:

- Tjänsternas ”endpoint”-namn
- Tjänsternas operationsnamn
- Meddelandevärden

Exakt hur man namnger sina tjänster kan variera från verksamhet till verksamhet. Det som är viktigt är att man är konsistent när man namnger sina tjänster. Erl (2005) har dock några förslag på saker man kan tänka på när man namnger sina tjänster.

- Tjänster som har en hög återanvändningspotential bör aldrig namnges på ett sådant sätt att de avslöjar den affärsprocess som de ursprungligen var tilltänkta för. Försök därför att korta ner namnet till ett så enkelt som möjligt, men att den fortfarande ger en beskrivning av vad den utför.

- Applikationstjänsterna bör namnges enligt den kontexten som den tillhör. Exempelvis kan man använda verb plus substantiv vid namngivningen. Exempelvis SalesReporting och GetStatistics.

- Namnen på applikationstjänsternas operationer bör vara så beskrivande som möjligt vad gäller deras tilltänkta funktionalitet.

- Entity-centric affärstjänsternas namn bör representera entiteten som de ursprungligen togs fram ur. Därför bör de namnges utifrån verksamhetens ursprungliga entitetsmodeller. Förslagsvis kan de namnges med ett substantiv, t.ex. Customer och Employee.

- Tjänsteoperationerna för entity-centric tjänsterna bör inte reflektera tjänsten i sig. Alltså om en tjänst heter t.ex. Employee så bör det inte finnas en operation som heter AddEmployee.

En annan viktig del är att se över hur grovmalda eller finmalda tjänsterna skall vara. En trend har varit att utveckla grovmalda tjänstestrukturer för att undkomma prestandaproblem som mer finmalda strukturer innebär, det vill säga att om strukturen är finmalen så krävs fler meddelandeomvandlingar vilket höjer prestandakraven. Saker som Erl (2005) anser är att ju mer grovmald en tjänstestruktur är desto mindre återanvändning kan den erbjuda. Om multipla funktioner är grupperade i en enskild operation, så kan den bli oanvändbar för tjänsteanvändare som bara behöver en enda funktion. Erl (2005) ger några riktlinjer för hur detta bör gå till.

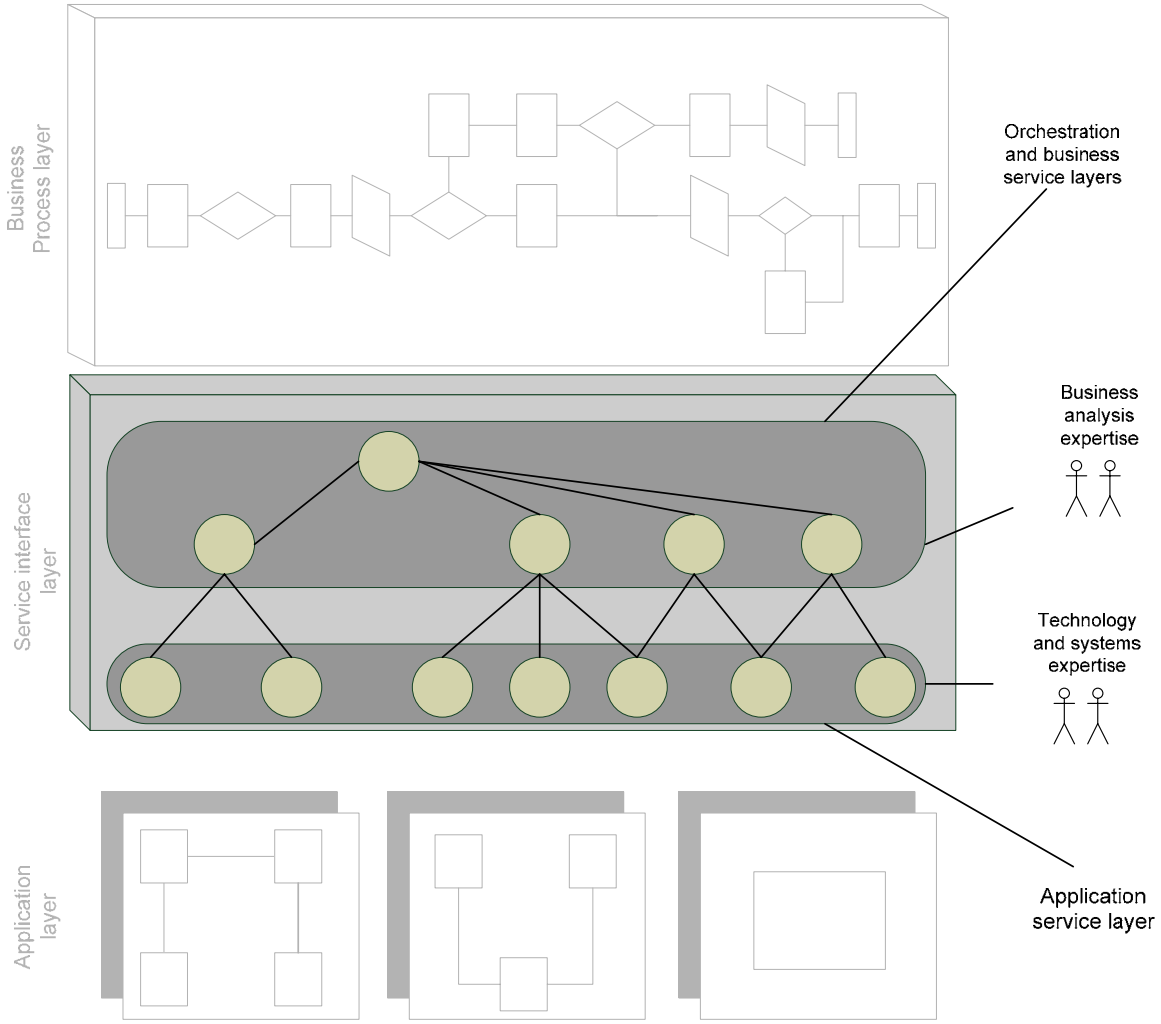
- Bestäm minimikravet för den prestanda som krävs i den miljön som tjänsterna kommer att fungera i.

- Undersök möjligheterna till att alternera med både en grovmald tjänstedefinition och en finmald tjänstedefinition för samma tjänst.

- Försök i så stor utsträckning som möjligt att använda grovmalda tjänster vid endpoints och tillåt mer ”finmalda” inom fördefinerade gränser. Exempelvis så kan externt tillgängliga tjänster vara relativt grovmalda så att de kan ta emot all data som krävs för att utföra en specifik aktivitet. Tjänsterna kan sedan anropa andra interna tjänster som är mer finmalda, för att på så sätt utnyttja återanvändningmöjligheterna samtidigt som man får en hög interoperabilitet med externa partners.

Oavsett hur bra tjänsterna är designade, så är det svårt att se exakt vad framtiden kräver. En del affärsprocesser kan komma att förändras. Därför är det viktigt att designen av tjänsterna gör att det finns utrymme för att kunna lägga till funktioner. Om design av operationer och meddelanden sker så att de innehåller så mycket ickespecifika värden och funktioner som möjligt, medför detta att det går att införa tilläggsfunktioner utan att behöva förändra hela strukturen i tjänsterna.
En viktig sak att tänka på är att det finns fasta rutiner för versionskontroll. Tjänster är nästan alltid byggda som en del av en automatiseringslösning. Det är dock bra att försöka att inte begränsa tjänsterna till att bara kunna fungera inom denna ram. Erl (2005) anser att det är bra med en analys av potentiella tjänsteanvändare utanför dess ursprungliga funktionsram, och ta detta i beaktning vid designen av tjänsterna.

Eftersom tjänsterna i en SOA representerar olika saker inom en verksamhet så är det enligt Erl (2005) också lämpligt att dela upp modelleringen av tjänsterna. Detta kan göras så att experter inom respektive område modellerar tjänster som passar in i just deras område. Figur 22 visar en illustration hur detta kan se ut.



Figur 22. Illustration av tjänster vilka bör modelleras av personer med systemexpertis respektive affärsanalytisk expertis (Erl 2005).

Eftersom affärstjänsterna är de tjänster som direkt representerar affärsprocesser och liknande i verksamheten, bör också de personer med bäst insyn hur denna del av verksamheten fungerar också modellera dessa tjänster. Applikationstjänsterna bör således modelleras av de personer med bäst insyn i hur befintliga system fungerar (Erl 2005).

4 Empiri

Avsnittet börjar med en presentation av de företag och dess respondenter som bidragit med svar på de intervjufrågor vi använt för att sammanställa vårt empiriska material. Efter att varje fråga presenterats har vi skrivit en förklaring till vad frågan betyder och varför vi valt att ställa frågan. Utifrån det insamlade materialet har vi gjort en tolkning och genomför längre fram en diskussion om vad som framkommit under intervjuerna och vilka kopplingar vi gör till de teorier vi presenterat under Teori. I slutet av rapporten görs en sammanfattning och slutsats utifrån syfte och valda frågeställningar.

4.1 Företagsintroduktion

Zystems
Zystems by Semcon är ett företag som arbetar med integrationslösningar. Deras affärsidé är att med hjälp av paketerade lösningar på etablerade integrations-plattformar integrera informationssystem för företag. Företaget använder ett eget utvecklat koncept som de kallar för BaseLine. Konceptet hjälper Zystems att skapa en plattform för en fullt skalbar infrastruktur för alla typer av systemintegrationer. BaseLine är en best-practice för effektivt utnyttjande av en integrationsplattform och bygger på användandet av standardprodukter i form av integrationsmjukvara från IBM. BaseLine sätter en enhetlig struktur för all kommunikation mellan de olika informationssystemen. Den centrala integrationsplattformen ansvarar för vidarebefordran och eventuell konvertering av informationen [Zt1].

Skanska
Skanska är ett svenskt företag som arbetar med entreprenadverksamhet som omfattar nybyggnation, renoverings- och underhållsarbeten på fastigheter. Skanska initierar, utvecklar, hyr ut samt säljer kommersiella fastigheter, exempelvis kontor. Företaget grundades 1897 och började med att tillverka cementprodukter, men verksamheten breddades snabbt till att bli ett byggföretag. Under 50-talet expanderade verksamheten till att bli internationell. Skanska omsätter årligen 22 miljarder inom Sverige, inom koncernen 125 miljarder. Företaget har 12000 anställda i Sverige, 54000 inom koncernen och huvudkontoret ligger i Solna [Sk1].

4.2 Respondenter

De respondenter vars svar har legat till grund för vårt empiriska material har varierande uppgifter inom Zystems och Skanska. Tillvägagångssättet för intervjuerna med nedan nämnda respondenter har beskrivits i Kapitel 2 Metod och kopplingar till Kapitel 3 Teori sker kontinuerligt i texten allteftersom kopplingar blir möjliga. Intervjuade personer hade följande befattningar:

- BaseLine-ansvarig
- Integrationschef
- IT-chef
- Projektledare, gedigen erfarenhet av systemintegration
- Systemintegratör

4.3 Analys

Avsnittet presenterar vår analys av det insamlade intervjumaterial som baserades på frågorna, vilka presenteras som Bilaga 2. Analysen är uppdelad på sådant sätt att vi först presenterar en fråga samt syftet med frågan. Efter syftet kommer de citat som uppkommit i samband med just den frågan. Efter citaten presenterar vi vår tolkning av citaten samt i så stor utsträckning som möjligt gör vi också kopplingar till vår teori om vi anser att en sådan koppling går att göra. Ibland har flera citat uppkommit i samma fråga, då kan det förekomma en kort analys av varje citat innan nästa citat presenteras. Anledningen till detta är att vi ibland har kunnat urskilja att citaten tar upp problem som är skiljda ifrån varandra trots att de har uppkommit ur samma fråga.

Vad var bakgrunden till projektet?
Bakgrunden till att frågan ställdes var för att få veta vilka faktorer, om det fanns några, som eventuellt låg bakom projektet att implementera SOA eller om Skanska bara valt SOA för att det är nytt och ville vara med i det främre ledet av ”SOA-vågen”.

> ***Integrationschef:*** *”Ja, det har flera bakgrunder, dels att vår gamla integration- plattform var död mer eller mindre.”*
>
> ***Projektledare:*** *”Bakgrunden till det här projektet kan jag inte svara på, det vet inte jag. Jätteenkelt. Vad jag kan säga är att eftersom jag gjort detta på andra ställen så kan ju jag säga om drivkrafterna är i huvudsak en enda och det är att hitta ett sätt att bryta den ständiga kostnadsspiral som IT-kostnaderna har i koncernen., och det är den enda orsaken. Och sedan letar alla företag efter olika sätt att uppnå det. För om man följer kurvans förlängning så här så blir det absurt, fullständigt absurt. Man måste göra något. Så enkelt är det.”*

Skanskas integrationsplattform var uttjänad och de behövde en ny. Skanskas huvudsyfte enligt projektledaren är enbart en sak, att bryta kostnadsspiralen för IT inom koncernen.

Varför föll valet på SOA? Vilka effekter var önskvärda vid ett införande?
Frågan ställdes för att undersöka om Skanska har tittat på andra lösningar än SOA i samband med att integrationsplattformen skulle bytas. Vi vill även få reda på vilka eventuella effekter Skanska förväntade sig efter ett införande, exempelvis lägre kostnader för underhåll av IT-system, lägre utbildningskostnader och så vidare.

> ***Integrationschef:*** *”…nu ville vi lyfta upp det här på en nivå högre därför integrationen blir allt viktigare och att vi vill sänka kostnaderna för det, och det är väl den största drivkraften egentligen, återanvändbarhet och att vi skalar.”*
>
> ***Projektledare:*** *”Vad man uppnår med SOA är att man, man uppnår att man inte behöver träna sin personal i oändlighet för varje nytt verktyg för varje ny si varje ny så. Och det är ju samma grundtanke som egentligen ligger bakom SOA, grundtanken. Gör någonting som en generell funktion som du kan återanvända i många integrationer, i många… för många IT-system.”*

> ***Integrationschef:*** *"…finns inte så många alternativ om man vill vara flexibel och kunna styra om snabbt och ser man hur hela branschen trummade på, var det ju bara en riktning."*

Svaren speglar på en strävan att sänka sina kostnader för IT-sidan, och att minska de utbildningskostnader som ett systembyte innebär. Ett argument för lägre kostnader är att skapa återanvändbarhet i sina system.

Har ni märkt av några av de önskade effekter ni eftersträvade?
Frågans kärna är att få reda på vilka effekter Skanska räknar med att få ut av sin implementation och jämföra dessa med vad vår litteraturstudie nämner som generella effekter av ett införande av SOA.

> ***Integrationschef:*** *"Vi börjar väl se effekterna av det i och med att vi började projektet i höstas tidigt i höstas, så vi har ju inte full lösning på plats, vi håller på med migreringsarbetet såväl som nya projekt just nu men vi ser ju styrkan med det just nu, vi ser vinster vi kommer att kunna plocka hem."*

Ovan nämner integrationschefen att de förväntar sig se vinster framöver och en naturlig följdfråga var då:

> **Vi ställde en fråga:** "Kan du specificera vinsterna som ni tror ni kommer att kunna få?**"**

> ***Integrationschef:*** *"Just det här att när man tagit fram ett, ett generellt affärsobjekt att man lätt kan haka på nya prenumeranter eller tjänstenyttjare, på det, det ser vi ju att det i den migreringsprocessen kommer, vi kommer jacka på vartefter, då anpassar vi ju oss bara mot den ändpunkten och inte utifrån början och då har vi ju sparat mycket redan där."*

Vår studie av SOA:s effekter inkluderar lägre kostnader tack vare återanvändbarhet. Om något går att återanvända kommer detta minska kostnader mer eller mindre direkt. Ovan nämner Skanskas integrationschef att *"man lätt kan haka på nya prenumeranter eller tjänstenyttjare"*. Lägre kostnader tack vare återanvändbarhet är den främsta vinsten. Detta är precis det som Newcomer nämner i teoriavsnittet tidigare i uppsatsen.

Vad är det som gör er tjänstestruktur till ett SOA? Alltså vad är er definition på SOA, och kan du ge exempel på en tjänst som har något karaktärsdrag som är specifikt?
Frågorna ställdes för att få veta hur Skanska ser på SOA och vad SOA är för dem. Världsbilden av någonting behöver ju inte stämma överrens bara för att begreppet är känt. Fortfarande är SOA ett relativt nytt begrepp och har enligt vår studie hittills ingen standardiserad definition.

Vi ville också veta om någon av de befintliga tjänster som implementerats har något karaktärsdrag som är specifikt för just hur en SOA-arkitektur skall se ut.

> ***Projektledare:*** *"Komponentisering, flexibilitet och kundfokusering, alltså det är mycket mera verksamhetsfokusering än vad vanliga integrationer är tidigare. Just det här att man börjar prata affärsobjekt och tjänster kring dom här affärsobjekten, gör att man redan där börjar titta på och göra någonting som är återanvändbart och lättare och kommunicera till egentligen systemägare och organisationen i stort än om man bara pratar flatfiles och ftp överföringar. Därför det, det är någonting som få kan ta till sig men när man tar fram typer, definitioner det här är en kund, det här är en leverantör det här är en faktura och visar det, det är väl på något sätt SOA är ju inte någonting nytt alls., Egentligen det är en smart paketering men på något sätt, för mig känns det som, ja det både databasteori och objektorientering och göra alltså det känns liksom ganska naturligt. Evolution, ingen revolution."*

En fråga som kom upp i samband med ovan nämnda är om Skanska ser några konkurrensfördelar med att implementera SOA, och vad dessa skulle kunna vara. Frågan är högst relevant med tanke på att vad vi vet om verklighetens företagsklimat så skall alla förändringar bidra med någon typ av förbättring, vanligtvis ekonomisk.

> **Vi ställde en fråga:** Ser ni några nya affärsmöjligheter för Skanska med det här? Ger SOA några nya affärsmöjligheter?
>
> ***Integrationschef: "****Definitivt, kraven våra kunder blir ju mycket mycket mer, mer och mer krävande i att öppenhet..."*
>
> ***Integrationschef:*** *"… det här att kunna publicera information erbjuda en, ett smörgåsbord av tjänster egentligen är ju, kommer vara en konkurrens faktor egentligen mot andra byggbolag eller andra aktörer…"*

Frågan nedan är en följdfråga på frågan ovan och ställdes för att utröna mer exakt vad projektledaren kortfattat anser vara de styrkor som SOA skall ge.

> **Vi ställde en fråga:** Men om du sammanfattar SOA med bara ett par ord, vad är styrkorna som du ser med SOA?
>
> ***Projektledare: "****Styrkorna med SOA är att… SOA ger en betydligt snabbare förändringsbarhet i en organisation. Det ger en möjlighet att bryta kostnadsspiralen pga. återvinning."*

Svaret visar på att det är verksamheten som fokus läggs på i jämförelse med tidigare integrationer. Att hela tiden diskutera runt vad det är som verksamheten innebär och vad den gör är viktigt för att kunna bygga upp rätt typ av tjänster i företaget. Har företaget inte förstått vad sin egen verksamhet kräver blir det svårt att få att bygga upp en fungerande arkitektur kring tjänster för då ökar risken för att fel smyger sig in och skapar onödiga kostnader och ett mindre flexibelt system.

Skanska ser klara konkurrensfördelar med SOA. Företaget kan ha en öppenhet gentemot kunden som inte var möjlig förut, att kunna förändra sin organisation efter hur vinden blåser och snabbt anpassa sig till förändringar på marknaden för att alltid ligga i framkant och inte missa viktiga marknadsandelar. Om Skanska kan visa att

företaget tar till sig nya tekniker som fungerar skapar detta fördelar gentemot andra liknande bolag för då visar Skanska att de hela tiden letar efter lösningar som kan förenkla och sänka kostnader för det verksamheten gör, vilket i slutändan skapar lägre kostnader för kunden. Att vara lyhörd är enligt oss viktigt för att överleva i dagens hårda företagsklimat.

När vi sedan ställde samma fråga som ovan till systemintegratören så uppkom följande diskussion:

> ***Systemintegratör:*** *”SOA för mig är tjänste ...baserade system. Sedan i praktiken så innebär tjänster en massa saker. Meddelandeorienterade, ofta asynkrona etc etc. Men man landar ganska snabbt i governancebiten. Det är lika viktigt att det inte bara är tjänster. Måste vara reglerat i kontraktsmässiga avtal. Jag skulle inte säga att det räcker med att knacka ”dit” en tjänst. Det måste också regleras på något sätt, finnas någon styrning.”*
>
> **Vi ställde en följdfråga: Vi talade tidigare med Integrationschefen och Projektledaren och det verkade som att dokumenthanteringen har varit svår.**
>
> ***Systemintegratör:*** *”Ja, den är svår om man använder word. Worddokument är ju värdelösa på att strukturera information. Men nu har vi dem inte i word-dokument längre som jag har förstått att ni har fått förklarat det för er. Så nu ser jag egentliga inga problem med det längre... det är långt ifrån optimalt men det är ju hästlängder bättre än tidigare.”*
>
> **Vi ställde en följdfråga: Det vi har fått förklarat är att ni har någon form utav wikilösning. Vad menar ni med wikilösning?**

Tidigare hade Integrationschefen nämnt Wikibaserad teknik för oss, vi var inte helt säkra på vad de menade och ville därför veta mer.

> ***Systemintegratör:*** *”Vi menar… att vi helt enkelt har migrerat våra dokument till conference som är ett kommersiellt informationshanteringssystem som bygger på wikiprincipen, om wikiprincipen är bekant?”*
>
> **Vi ställde en följdfråga: Jag är osäker, det kan hända att jag vet, men jag vågar inte svära på det.**
>
> ***Systemintegratör:*** *”Nä.. alltså dels är väl wiki från början är att systemet är öppet för alla att redigera.. men riktigt så kör ju inte vi det. Vi har ju en rättighetsmodell på det. Men mycket har ju handlat om att wiki har en platt struktur av sidor.. det finns ingen hierarki egentligen. Det finns korslänkar som strukturerar informationen.*
>
> *Sedan har det här systemet även ett hierarkiskt begrepp som.. ja jag vet vad man skulle förklara det som.”*

Vi ställde en följdfråga: Men är det här systemet ett sätt för dig att hålla koll på tjänsterna eller är det ett sätt för tjänster att hålla koll på andra tjänster? Eller är det både och?

Bakgrunden till frågan är tanken på UDDI. Vi ville veta om detta Wiki var någon sorts variant på ”udditekniken”. Eftersom vi på fler ställen tidigare, i våra teoristudier läst och sett exempel på just UDDI, låg detta i bakgrunden hela tiden i vår diskussion kring överblickbarhet av tjänster.

Systemintegratör: *”Nej, det är ett sätt för oss att hålla koll på dokumentartifakterna kring våra tjänster,* ***andra tjänster har ingen möjlighet att tala med det systemet. I dagsläget****.”*

Vi ställde en följdfråga: Skulle du vilja att det var så?

Systemintegratör: *”Då är vi inne på hela registrytanken och UDDI-snacket och det tror jag inte särskilt mycket på.. det kan hända att det kommer men det ligger väldigt långt fram i tiden. Folk kan inte ens bygga tjänster i dagsläget.. rent generellt. Det byggs tjänster för glatta livet men dom är ju inte tjänster. Det finns wizards där du trycker på next next next och finish, sedan exponerar befintliga metoder i en klass. Så tror man att man har SOA bara för man har Web Services.* ***För att gå tillbaka till registrydiskussionen så nej.. vi vill utveckla det som ett, det blir lättare att dra ut metadata ur det men det handlar mer om att vi t.ex. vill.. när vi packeterar en integration kunna dra ut metadata om vilka endpoints integrationen skall prata med****.*

Vi ställde en följdfråga: Men UDDI.. finns inte det? Eller du menar att det används inte?

Systemintegratör: *” Microsoft lade ju ner sitt publika UDDI-registry med en snygg bortförklaring att nu har vi (konceptat oss) så nu behövs inte UDDI mer. När det i själva verket inte var någon som använde det som jag tolkade det som. Men UDDI finns, det kommer eller har kommit en ny version.”*

Vi ställde en följdfråga: Men om man inte använder det vad använder man istället? Och är det överflödigt? Finns det inget behov av UDDI?

Systemintegratör: *”Alltså ett tjänste registry behöver man. Men det har ju vi i vårt* ***informationshanteringssystem****, där tjänsterna är presenterade och katalogiserade... dom finns ju där alltså vad har vi för tjänster.. så kan vi haka på beskrivningsmetadata som t.ex. vilka objekt jobbar dom här tjänsterna på eller vilka tjänster jobbar på det här informationsobjektet.* ***Men när jag säger UDDI så tänker jag på hur det ursprungligen var tänkt. Att tjänsterna själva skulle kunna slå upp andra tjänster och anropa dom****.”*

Systemintegratör: *”Och... Tekniskt är det ballt, Men jag har inte sett någon verksamhet som kräver det överhuvudtaget och jag kan ju tänkta så här ja om vi har UDDI för Skanskas jag vet inte vad... leverantörstjänst och slå upp kreditstatistik, värdighet och automatiskt välja den leverantör*

som levererar billigast kreditvärderingsinformation per transaktion så är det ju en optimering men då måste du ha juridiskt bindande avtal så att informationen som står i UDDI:n är korrekt. Och då måste man ju ha preppat juridiska avtal med alla dom parter... för vi vet ju inte vilken tjänst vi kommer att använda eftersom vi tar den tjänst som är billigast för dagen enligt UDDI:n, och jag ser inte det på ganska många år."

Vi har här följt en diskussion om problemen med just tjänsteorientering. Detta är ett mycket stort och tungt problem i all användning av tjänster men kanske speciellt i målet att kunna få till ett SOA-system. Enligt de vi pratat med är det kanske också här den allmänna uppfattningen av SOA haltar mest. Förslag från litteraturen att enbart använda sig av t.ex. Web Service och därigenom förlita sig på den s.k. UDDI-lösningen är inte att rekommendera. Anledningen till detta som vi ser det, är att tekniken ännu inte riktigt fungerar som den var tänkt.

Följde projektet någon bestämd metod eller rutin, RUP eller liknande? Om inte, vilka komplikationer uppkom vid projektets gång? Om ja, vilka komplikationer uppkom vid projektets gång och vilken specifik metod användes?
Syftet med frågan var att få reda på hur strukturerat projektet har varit. Ifall de har använt någon form utav metod anpassad för SOA så ville vi veta vilken och huruvida de ansåg att de saknade något. Med hjälp utav svaren hade vi hoppats kunna utröna ifall det behövs standardiserade metoder vid framtagningen av SOA-tjänster. Vi fick följande svar:

Integrationschef: *"För projekt, ren projektstyrning använder vi Tieto Enators modell PPS inom Skanska för projektstyrningsbitarna och sedan för dokumentationsbitarna så har vi köpt in Zystems modell som de kallar för Baseline, så för att dokumentera och utveckla, men för projektledningsbitarna är det PPS"*

Systemintegratör: *"Vi jobbar enligt en metodik som heter Basline, som i grunden är en uppsättning mallar och konceptmodell, hur dom förhåller sig till varandra, om man har t.ex konceptintegration där varje integration har en eller flera tjänster osv. Vårt arbete är att försöka göra så mycket som möjligt av specifikationen som innebär då att författa så mycket som möjligt av baselinemodellen vad gäller dokument osv , och sedan implementera dom därefter"*

I Skanskas fall hade dels en intern konceptmodell använts som är utvecklad utav Zystems, samt en ren projektstyrningsmodell. Det vi var mest intresserade utav var "Baseline" som tydligen är en konceptmodell för SOA projekt. Det ledde till en följdfråga där vi ville utröna hur bra metoden fungerat. **Har metoden fungerat bra eller är det något ni känt att ni har saknat?** Så här lydde svaren:

Integrationschef: *"Vi har väl lyckats ganska bra, nej det viktigaste skulle vara branschstandarder i så fall. Och mer patterns från, från leverantörer alltså från IBM och från dom andra liksom vill du göra det här anser vi vara en bra lösning för detta, så ja Service repositories egentligen från alla dom stora leverantörerna vore ganska, väldigt värdefullt. Därför, ja, avsaknaden av gemensamma standarder är det största problemet så då får*

> *man ju höfta någonting. Så metoden, ja det är ju, det är oftast ”här är kraven” ser vi på lite längre sikt ja då tar vi fram ett.*”

När vi talade med projektledaren och ställde samma fråga så svarade han så här:

> ***Projektledare:*** *”Det svåra är inte metoder eller verktyg, det svåra är om vi nu känner t.ex. att vi… alltså det som det väldigt snabbt hamnar på det är att SOA-tjänster kan ju IT-sidan bygga till… vad skall vi säga… en given nivå. Sedan måste affärssidan träda in och säga ”grabbar så här skall vi göra”... ”*

Så här svarade respondent ***Systemintegratör:***

> ***Systemintegratör:*** *”Den fungerar bra... måste jag säga. Jag är ju rätt så ny med den eftersom jag är ny på bolaget. Men ... från början tyckte vi inte att den passade in riktigt när vi höll på att tråckla med tjänsterna, men det har den faktiskt gjort. Det är kanske snarare så att den är såpass bra så att den är svår att förstå...*”

Svaren varierade lite, men huvuduppfattningen verkar ändå vara att metoden de använt har fungerat bra och inget särskilt har saknats. Integrationschefen ansåg dock att det hade varit värdefullt med ”service repositories” ifrån de stora leverantörerna.

Blev resultatet det önskade eller blev det stora avvikelser från ursprungsplanen?
Det visade sig att Zystems och Skanska har använt sig utav en internt utvecklad metod. Det visade sig också att den verkar har fungerat bra. Så här svarade Integrationschefen***:***

> ***Integrationschef:*** *”Nej det tycker jag inte i och med att vi börjat i det lilla och vi är ganska klara på vad vi vill att, vi definierar affärsobjekt utifrån det vi har och dom krav vi ser nu plus en liten educated guess och sedan så får man ta det därifrån, så den ansatsen funkar, och mognaden på system och tredje parts leverantörer i övrigt är inte så pass långt gången så att,, vi kan vara med att forma dom så pass mycket så det känns inte som vi kan göra så fel*”

Svaret vi fick, är ett direkt svar på frågan om det blev stora avvikelser från ursprungsplanen? Resultatet av projektet verkar hittills ha varit lyckat. Skanska har börjat i liten skala med ett fåtal tjänster. Det verkar som om integrationchefen anser att det är anledningen till att det gått så bra, eftersom chansen att göra fel blir mindre ifall projektet är i liten skala. Vi dock inte utesluta att deras metod Baseline faktiskt har hjälpt dem en hel del och kanske kan även den ha spelat roll.

Hur resonerade ni när ni valde ut funktioner som skulle bli tjänster? Hur började ni?

Vi ville veta hur Skanska valt ut funktioner som skulle bli tjänster för att kunna göra en koppling till de teorier som vi studerat för att se hur det går till hos Skanska, och om dessa teorier använts i någon utsträckning i implementationsarbetet.

> ***Integrationschef: "****Ja det var väl främst behov, det vi började med var, eller bland annat där vi började med var en tredjeparts leverantör, extern leverantör som vi har som hade ett speciellt informationsbehov. Och det låg ganska snart i tiden, då var det ganska naturligt att börja med dom objekten, och bygga upp arkitekturen utifrån det och det är så vi kommer jobba och jobba framgent också. Men då hoppas man att i och med det arbetet har höjd för ja hyfsat mycket i alla fall för kommande krav, så när dom som man inte räknat med eller nya parter kommer bara kan jacka in dom och anpassa sig efter deras krav hyfsat enkelt."*

Både Erl och Newcomer nämner i sina teorier att det är behovet som skall styra vid val av vilka funktioner som skall bli tjänster. Detta är precis det Skanska gjort. Skanska började där behovet var störst och valde ut en leverantör som hade ett speciellt informationsbehov, och byggde upp en SOA-arkitektur runt denne för att kunna tillgodose behovet hos tredjepartsleverantören, men även inför framtida behov av utbyggbarhet.

Har ni en tydlig koppling mellan de tjänster som skapades och befintliga processer?
Vi ville med denna fråga försöka ta reda på i hur stor utsträckning som befintliga affärsprocesser stöds av de tjänster som byggs. Så här svarade Integrationschefen:

> ***Integrationschef:*** *"Vi har ingen processmotor på plats idag, det är fas två för oss egentligen. Därför på något sätt måste vi bli mogna i vår kunskapsutveckling mellan våra system innan vi bygger processer ovanpå. Men vi tänker givetvis på det nu. Men dels har vi inte identifierat i tillräckligt stor utsträckning processansvariga i företaget som är mogna och liksom kunna ansvara för en process och kunna modellera den och ta fram key performance indicator och sådär. Det tänket finns inte riktigt än"*

I teorin presenterade vi hur nära SOA och BPM står varandra. Enligt Mike Rosen kunde man med SOA bygga återanvändningsbara tjänster, men utan BPM ansåg han att man inte kunde utnyttja dessa för att skapa en flexibel verksamhet. Skanska har varken använt sig utav processmodeller eller BPM när de har utvecklat sina tjänster. Främsta orsaken verkar vara att de inte anser sig vara mogna för att kunna göra det.

Vem/ vilka var ansvariga för val och utveckling av tjänsterna? Det vill säga vilka positioner besitter dessa personer, samt vilka tidigare erfarenheter av SOA har de?
Med denna fråga ville vi få reda på vilka som ligger bakom valet av tjänsterna som skall byggas.

> ***Integrationschef:*** *"Valet är nog jag undertecknad som har gjort valet i och med att vi, vi på It enheten måste ligga före organisationen, vi måste förutse vad de vill ha om ett år, vi kan inte komma när dom vill ha någonting och säga, ja men då skall vi börja titta på det, så hela den här uppsättningen av ett integrationscenter och igångrullandet av tjänstefiering av våra integrationer det har varit ett projekt och en boll*

som drivs enbart härifrån och under min försorg tillsammans med vår chefsarkitekt”

När vi ställde samma fråga till systemintegratören så uppstod följande dialog kring frågan:

Systemintegratör: *”...vi bygger inte tjänster uttryckligen från början heller. Vi bygger integrationer, och då är det så att man skall ta data från system A till system B. Då identifierar vi möjligheten till att bygga tjänster och möjligtvis så bygger vi också tjänsten. Eller så väntar vi tills vi får en konsulent till innan vi bygger tjänsten*”

Följdfråga: Kraven på tjänsterna, är det den som vill ha tjänsten som kommer med kraven eller är det ni som talar om för dem vilka krav man skall ha?

Systemintegratör: *”Alltså systemet ställer krav på vilken data den skall ha. Sen försöker vi ställa oss över det och ta fram en rimlig tjänst som inte nödvändigtvis är knuten till just det systemet. Sen är ju tjänstebegreppet där lite intressant, alltså tjänster blir det ju för varje integration. När jag säger att vi inte bygger tjänster, menar jag att vi inte explicit gör publika tjänster. Så det blir tjänster men dessa är privata. Sen försöker vi tänka så långt som möjligt för att vi lätt skall kunna göra dem publika när behovet uppstår*”

Följdfråga Men det är någon som talar om för dig då, vad dom vill ha?

Systemintegratör: *”Nej det behöver det absolut inte vara utan vi ser utåt, sen kommer system X och vill ha data som visar sig vara samma data. Då ser ju vi som sitter i mitten och håller i det där, möjligheten till att lyfta upp det där till en publik tjänst. Så att både system A och X nyttjar samma tjänst*”

Följdfråga: Men är det ni som tar det beslutet eller är det någon annan?

Systemintegratör: *”Nej.. han(Systemägaren) får ju ta beslutet att göra det. Vi identifierar ju möjligheten så kommunicerar vi den*”

I teoriavsnittet presenterade vi ett par olika slags tjänster. Erl anser att affärstjänsterna lämpligast bör tas fram utav personer som är väl insatta i affärprocesserna i verksamheten, medan de tekniskt kunniga bör ta fram de tekniska applikationstjänsterna som skall kapsla in befintliga system. Projektet på Skanska verkar dock ha varit ett rent IT-projekt utan några som helst inblandade ifrån ”affärssidan”. I övrigt verkar det som att behovet styr helt och hållet.
När vi senare fortsatte diskussionen med projektledaren om hur tjänsterna hade tagits fram så var detta några av de saker han nämnde:

Projektledare: *”...SOA-tjänsterna som är här de är ju också väldigt starkt färgade av att det finns ett stort vått täcke som heter Oracle E-business Suite, dvs. ett affärssystem som egentligen täcker oerhört stor mängd utav de SOA-tjänster som skulle komma fram ur en topdown-analys*”

> ***Projektledare:*** *"Och då är ju alla de här så kallade tjänsterna eller vad man kallar dem, funktionerna egentligen inuti Oracle, och de kommer aldrig ut därifrån för de är väldigt väldigt svåra att plocka ut som en delmängd, presenteras som en separat tjänst. Därför bli SOA-tjänsterna här färgade utav det och då blir dem mer i form utav vad lämnar affärssystemet ut för information kontra vad behöver affärssystemet för information och vilka levererar det?*"

Det verkar som att valet av tjänster grundar sig på behov i form av informationsutbyte mellan olika system. Men eftersom det system som tjänsterna utgår ifrån är ett affärssystem så kan det tänkas att tjänsterna blir relativt affärsinriktade. Skanskas tjänster skulle vi tolka som det som Erl benämner som hybridtjänster eftersom de både behandlar affärsobjekt samt att de faktiskt står i direkt relation till systemet.

Vilka slags tjänster tror ni kommer att implementeras framöver?
Vi ville med denna fråga se hur projektet är tänkt att fortskrida framöver. Vad är målet med Skanskas SOA projekt. Är det i första hand enbart en teknisk integration eller är målet att införa SOA globalt inom Skanska koncernen?

> ***Projektledare:*** *"Jag tror att vi väldigt snabbt kommer att hamna på funktionstjänster. Typiskt är en tjänst som orderläggning så till vida att man bygger en tjänst utanför Oracle som småapplikationer ute i grustagen anropar. Men de går inte in i Oracle och gör det, alltså utan man bygger en tjänst som sedan gör i andra led för att hantera teknikskillnader, dåliga kommunikationslinor tidsaspekter och andra sådana där saker*"

Någonting som visades nu var att det planeras tjänster som går runt affärssystemet. Tjänster som inte använder affärssystemet eftersom det saknar de funktioner som Skanska har som mål att kunna stödja med hjälp utav IT.

I följande text använder vi uttryck som finkorniga och grovkorniga tjänster, det vi menar är finkornig = atomär, grovkornig = sammansatt. Detta för att läsaren skall förstå kopplingen till teoriavsnittet.

En fördel att använda finkorniga tjänster är att man får stor återanvändningsbarhet. Finns det något som ni anser vara negativt med att använda enbart finkorniga tjänster?
Syftet var att vidare undersöka hur Skanska har designat sina tjänster med tanke på hur tjänsterna var sammansatta. Så svarade de olika respondenterna.

> ***Integrationschef:*** *"Bra fråga…det beror ju på vad, problemet vi har sett det är ju när man skall börja versionshantera sina tjänster, det har ju vi inte stött på än men vi kommer ju stöta på det, och då tror jag man får en ökad komplexitet när du skall versionshantera en sammansatt tjänst mot att du skall versionshantera en singeltjänst eller en finkornigtjänst sedan tror jag det kan vara fördelar med och ha lite, ha sammansatta tjänster därför dom är lättare och sälja internt i organisationen för har du för finkorniga tjänster då är du återigen bits and bytes och små informationsmängder medans en grovkornig tjänst kan du, här kommer en leverantör med fullständig kreditvärdering och vår försäljningshistorik på*

denna leverantören och så publicerar man det till dom som vill ha det, då har du ju raffinerat datan mycket mer"

Projektledare: *"jag tror att det är… det blir väldigt svårt att hålla ihop dem om man har dem separerade hela vägen, det blir praktiskt att bygga ihop en tjänst utav tre delkomponenter. Jag tror, så att jag tror att det är en mix man kommer att landa på. Och mixens innehåll och struktur kommer att skilja sig från bolag till bolag, beroende på vad man gör och vilken ände man börjar. Jag tror inte på väldigt hög nivå av tjänster*"

Följdfråga: Men om man skulle nämna fördelar och nackdelar med respektive finkorniga eller grovkorniga tjänster...

Systemintegratör: *"Grovkorniga..."*

Följdfråga: Ok, men vad är just fördelen med grovkorninga i så fall tycker du? Och om den har några nackdelar?

Systemintegratör: *"Den jobbar på mer kompletta dataobjekt skulle jag vilja säga. Risken med för mycket finkornighet är ju att man hamnar på RPC.. du måste göra vissa sekventiella anrop för att uppnå någonting. Du kommer däremot kanske skicka mer onödig data till en grovkornig tjänst.. så kan man dividera om det*"

Thomas Erl anser att man bör se över hur "grovmalda" respektive "finmalda" tjänsterna skall vara. Vidare påstår han att finmalda tjänster ger högre återanvändningsbarhet. Det verkar inte som det inom Skanska finns något entydigt svar på vilket som är att föredra, finkorniga eller grovkorniga tjänster.

Vill man ha mer/mindre integration mellan system och hur tycker ni SOA hanterar detta? (Tänk på möjliga motsättningar ex loose coupling.)

Frågan togs upp för att få reda på Skanskas syn på hur de ser på integration mellan system och vad integration betyder för dem. Vår litteratur nämner detta som en viktig punkt. För att SOA skall vara riktig SOA så skall det vara lösa kopplingar mellan system. När vi tog upp frågan med projektledaren så uppstod följande dialog:

Vi ställde en fråga: "Vi har ju stött på… vi tänker ju på det här med integration mellan system och SOA. Och varför tanken kom upp det är ju för att SOA på något sätt främjar integration samtidigt som det ställer som krav att det skall vara lösa kopplingar. Hur ser du på den tycker jag, nästan emotsättning? Alltså hur…"

Projektledare: *"Hur då menar du motsättning?"*

Följdfråga: "Jo, när man tänker sig integration så tänker jag mig en närmare bindning, att man kommer närmare varandra."

Projektledare: *"Ja."*

Följdfråga: "Det kan hända att jag har fel uppfattning där men."

Projektledare: *”Ja, det skulle jag vilja påstå.”*

Följdfråga: ”Ja, okej.”

Projektledare: *”Det som jag har gjort, började med på Telia och det som vi gjort här det har ett enda syfte och det är lösare kopplingar. Lösare koppling, lösare koppling, lösare koppling.”*

Följdfråga: ”Och lösare koppling främjar integrationen menar du? Ja, det gör den ju.”

Projektledare: *”Det främjar integration men den främjar framförallt förändring.”*

Vi valde att ta med en stor del av konversationen för att inte förlora helhetsperspektivet på de svar vi erhöll på frågan. Svaren vi fick tyder på att Skanska inte anser integration vara något som medför en hårdare bindning mellan system utan snarare medför löst kopplade system som då samtidigt medför större förändringsmöjligheter. Projektledaren på Skanska har enligt intervjun god erfarenhet av vad integration innebär, och integration resulterar enligt denne uteslutande i lösare kopplingar mellan system, vilket är en av grundtankarna med SOA.

När vi ställde samma fråga till Integrationschefen så svarade han så här:

Integrationschef: *”Ja det finns en ganska klar risk, om man säger att systemen är mogna och börjar tjänstepublicera så har vi system A här som 500 tjänster och system B som har 500 tjänster och C som har en massa tjänster och så låter man de här kommunicera fritt med varandra då har du ett ormbo direkt igen, fast du har ”soafierat” det och du har inte gjort en vinst. Vad vi kommer göra åt hos oss är att… system A får internt använda sina tjänster hur mycket dom vill, men internt i systemet A, men så fort den där tjänsten skall börja användas av något annat system så måste den replikeras på integrationsplattformen eller vår* ***service bus*** *då. Därför då får vi den övervakad, vi får den versionshanterad, vi får den dokumenterad och vi har en tydlig ägare på den. Det är jätteviktigt, därför låter vi de här systemen, när tjänsten börjar exponeras tala direkt med varandra då har vi ingen loose coupling längre och då kan vi inte gå in och säga att nej men vi vill modifiera den här tjänsten här borta, så det är väldigt viktigt att integrationsplattformen eller* ***tjänstebussen…*** *dels skall ha den som publicerar tjänsterna både externt och mot andra nyttjande system i organisationen. Just för att uppnå den här versionshanteringen.”*

Här fick vi en fingervisning om hur Skanska hanterar själva tjänsteorienteringen med hjälp av något de kallar **Service bus**, med andra ord Enterprise Service Bus. Här visar Skanska att det försöker skapa sig en överblick vilka tjänster som finns. Det var också här vi började ana problem som kan uppstå vid ett införande av ett SOA system.

Något annat att tillägga?

Frågan ställdes för att se om respektive respondent hade något att tillägga som denne ansåg att vi missat, eller kanske ville förtydliga något som sades tidigare extra noggrant för oss.

> ***Projektledare: "****Ja, jag tror att… jag tror att det är väldigt viktigt att titta på drivkrafterna för varför man tittar på SOA, och att de inte har med någonting annat än med kostnadsbesparingar att göra i grund och botten. Sedan kan de kalla det för en massa saker. Det är en sak men det är kostnadsbesparingar. Och det är ett… och tänker man rätt, återanvändningsbart, löst kopplat till IT-system tror jag är oerhört viktigt. Då finns det stora stora vinster att göra. Det är jag helt övertygad om."*

Svaret ovan tyder på att det är framförallt ur två aspekter som SOA införs, dels för att minska kostnader för framförallt IT-sidan, men också skapa flexibla, stabila och utbyggbara IT-system som enligt projektledaren kommer att vara mer långlivat enligt dennes tidigare erfarenheter av systemintegration inom företag.

Integrationschefen svarade så här:

> ***Integrationschef:*** *"Ja vad skulle det vara? Jag vet inte hur mycket ni tittar på dokumentationsverktyg och sådär för att stödja SOA, gör ni det?"*
>
> **Vi svarade:** Nej lite grann, nej inte så mycket.
>
> ***Integrationschef:*** *Vi började i höstas när vi handlade upp Zystems arbetssätt och dokumentationssätt. Består av en mängd word-mallar egentligen eller word-dokument som då refererar till andra word-dokument. Här har vi en integrations-*
> *Specifikation och sedan har vi end pointspecifikation och så har vi integrationskontrakt här. När man börjar etablera tjänster som är återanvändbara, så kan vissa delar av den här förekomma flera gånger. Vilket dokument, hur strukturerar man, det här växer. Man kommer ganska snabbt fram till det här kommer inte funka och då börjar vi titta på andra verktyg för att dokumentera det här… och då har vi kommit fram till ett wikibaserat verktyg för att uppnå den här . Därför där får vi kopplingar automatiskt informationen finns en gång och länkas vidare till andra.* ***Så, det är ett viktigt sätt och just dokumenteringen av tjänster är rätt viktigt eller väldigt viktigt****.*
>
> **Vi ställde en följdfråga: Men det är någonting som ni, någon slags dokumenthantering som skulle vara.**
>
> ***Integrationschef:*** *"Informationshantering, vi tycker väl vi har hittat någorlunda rätt form för just, för oss internt i projektet då och även så småningom våra, våra kunna exponera delar av den här informationen utåt men just det här och, att det är informationen som är viktig och att man får med rätt taggar och sådär. Det här är tjänsterna och det är nyttjarna och sådär* ***för i ett stelt dokument eller wordmall system med ett filstruktur så kan du aldrig hantera det där, det stjälper ganska fort****.* ***Så någon form av Web Service repository, ja eller tjänstekatalog är ju väldigt viktig****."*

Det var alltså först här vi verkligen fick det bekräftat vad vi tidigare bara anade, hur viktigt det är att hålla god överblick över sina tjänster. Detta för att inte hamna i ett nytt ormbo. Vad vi ser här, är en bekräftelse på hur och varför det är viktigt att ha ett smart system för att ha kontroll över sina tjänster.

Generella citat från projektledaren om grundproblem inom systemutveckling.
Eftersom projektledaren arbetat inom flera olika koncerner så tenderade samtalet att övergå i generella problem som projektledaren ansåg vara desamma på alla de ställen han tidigare jobbat. Så här uttryckte han sig:

> ***Projektledare:*** *"För det är ju såhär att det är ju väldigt många bolag som har stora affärssystem och med stora menar jag att de täcker väldigt många funktioner utav de SOA-tjänster som egentligen man skulle kunna ta fram om man gör dem småskaliga och rörliga och allt det här som… och det är många gånger som varför har man valt ett stort affärssystem som täcker så mycket? Ett är vill du låta IT-systemet brotta ner dem problem som vi inte klarar av själva organisatoriskt. Egentligen är ju* ***Telefons AB***[1] *före detta 57 inget problem om det hade varit så att ekonomisidan hade styrt hur kontoplanen hade sett ut, hur redovisningsprinciper hade sett ut. Då hade ekonomisystemets färg inte spelat någon som helst roll men de klarar inte av det. Därför att de är för svaga själva, de har för dåliga kunskaper om hur de skall styra, de styr och då lyssnar inte de andra och då skiter de bara i dem där på huvudkontoret, de struntar i det, vi gör som vi vill. Och så har man inte maktmedel och ta till runtomkring det där så det blir liksom inte den styrning som man behöver och tillslut ger koncernledningen upp och så köper man det här stora… affärssystemet och så låter man på något sätt affärssystemet bli det verktyg som de genomför den här standardiseringen i sättet att jobba via. Så i själva verket är det ett införande av ett affärssystem men lika mycket är det gemensamma affärssregler, gemensamma redovisningsprinciper, gemensamma si gemensamma så och det är därför alltid de har en tendens att ta oändligt med tid och resurser i organisationen*"

Samtalet med projektledaren tenderade att gång på gång glida in på problemet med samarbetet inom verksamheterna. Detta hände vid ett flertal tillfällen, vilket medför att det inte går att blunda för att det onekligen är ett stort problem inom verksamheter idag. Faktumet att tjänsternas huvudsakliga uppgift är informationsutbyte mellan Skanskas affärssystem och andra system gör att Erls metod inte direkt passar in. Det som Erl presenterar är en metod som är anpassad till att verksamheten har ett antal olika system som stödjer olika processer i en verksamhet. Men likväl skulle metoden kunna vara användbar eftersom affärssystemet är ett IT-system som används som stöd i företagets olika processer. Den egentliga skillnaden är att ett affärssystem är så mycket större än små enskilda system. Följande citat ger oss en fingervisning på vad man särskilt bör ta hänsyn till när det är ett affärssystem tjänsterna pratar med.

> ***Projektledare:*** *"...när man designar en SOA-tjänst, och du t.ex. har databaser eller i grunden, som i det här fallet då eller andra då gäller det att vara lite vaksam. För att du kan t.ex. ställa en fråga startar ett väldigt väldigt stort jobb i databasen och är databasen i själva verket ett affärssystem, som i det här fallet, den är ju ett affärssystemskal ovanpå en databas, så låser ju den då prestandamässigt upp databasen för alla andra processer och ingen kan göra någonting här på Skanska. Lite läskigt sådär. Så att när man designar SOA-tjänster så måste man också ta hänsyn till den här typen av problem och frågeställningar, vad är det*

[1] Fiktivt företag.

egentligen SOA-tjänsten skall prata med bakom skynket. Är det ett affärssystem, kan vi ställa den här typen utav frågor till affärssystemet".

Till skillnad ifrån fler mindre isolerande system så kan alltså ett affärssystem låsa flera processer ifall något går fel. Det är nog därför en bra idé att beakta det som projektledaren berättade angående att man i detta fallet med affärssystem måste ta hänsyn till hur saker fungerar så att man inte låser systemet på grund av att en tjänst skickar en för komplicerad förfrågan till affärssystemet.

Faktumet att projektledaren väldigt ofta återkom till samarbetssvårigheter inom verksamheten gjorde att det uppkom en följdfråga vid ett tillfälle. Vi frågade projektledaren rakt ut hur samarbetet fungerar, så att vi fick mer konkret svar. Så här svarade han:

> ***Projektledare:*** *"Nej, det är svårt, det är oerhört svårt. Det är oerhört svårt att framförallt komma överens*"

> ***Projektledare:*** *"Det är oerhört svårt att få affärssidan att faktiskt ställa sig upp och ta ansvar för det som de skall göra. Att beskriva hur gör vi affärer, hur kommer vi att vilja göra affärer. D.v.s. sådant slår ju sedan ner på teknik, för de brukar oftast fastna i att träta om… Alltså man fastnar i detaljdiskussioner som inte ger någonting, vad är en kund, vad är en produkt"*

Samarbetet mellan "affärsidan" och IT-avdelningen verkar har varit en utav de största svårigheterna i projektet. Projektledaren som har jobbat många år inom IT-relaterade områden säger att det är ett återkommande problem inom ett flertal verksamheter.

5 Diskussion

Diskussionsavsnittet baseras på det material som tidigare presenterats i Kapitel 2 Analys. De kopplingar som finns mellan våra intervjufrågor och presenterade teorier ligger som en bilaga. De frågor som vi ställde var utformade för att utifrån presenterade teorier kunna dra slutsatser som vi kan använda för att slutligen kunna svara på vår frågeställning i vår slutsats. Vi har valt att bryta ut det empiriska material som samlats in och delat in denna i lämpliga underrubriker för att skapa en rigid och lättförståelig struktur på presenterad diskussion.

Vi tror att i princip alla företag finns en ständig strävan sänka sina IT-kostnader. Detta kan uppnås genom att skapa återanvändbarhet i sina IT-system. Det är grundtanken med SOA, att skapa återanvändbarhet. Detta för också med sig är att sänka kostnaderna för utbildning av sin personal, detta på grund av att systembyte kommer att ske mer sällan i och med IT-systemet är mer flexibelt och utbyggbart tack vare tjänstestrukturen.
Vår studie av SOA:s effekter inkluderar lägre kostnader tack vare återanvändbarhet. Om något går att återanvända kommer detta minska kostnader mer eller mindre direkt. I de svar vi fått nämner Skanskas integrationschef att *”man lätt kan haka på nya prenumeranter eller tjänstenyttjare”*. Detta är precis det som Newcomer nämner i teoriavsnittet tidigare i uppsatsen.

Metoder
Skanska och Zystems har använt sig utav en internt utvecklad metod vid framtagning av tjänster. Integrationschefen nämner att det skulle vara en fördel att ha någon form av branschstandard. Kanske kan Erls förslag på hur analys och design bör ske vara Skanska till hjälp. Projektledaren ansåg dock inte att bristen på metoder eller modeller spelade någon roll och eftersom han arbetat inom många olika koncerner, så vi antar att han har en hel del erfarenhet utav liknande projekt. Men faktum att en metod för utvecklingen av tjänster har använts, tolkar vi som att det kan vara användbart med en sådan vid utveckling. Den metod som Erl presenterar är ingen form utav standard utan bara hans syn och förslag.
Vi anser att en standardisering av utvecklingsverktyg är en fördel vid utveckling av tjänster, men än så länge ingen nödvändighet så länge projekten är relativt små, som i exempelvis Skanskas fall.

Kopplingen mellan SOA och BPM
Skanska har inte använt sig utav några modeller av sina affärsprocesser när de tagit fram sina tjänster. Erl förespråkar att personer med god kunskap om affärsprocesser bör ta fram de tjänster som direkt skall stödja dessa, och att personer med god kunskap om befintliga system skall ta fram de tjänster som direkt pratar med systemet. Faktum att Skanska inte har gjort på detta sätt kan innebära ett problem vid utveckling av affärsrelaterade tjänster eftersom ”IT-sidan” inte har lika stor kunskap om hur affärsprocesserna ser ut. Mike Rosen (2004) påstår att SOA och BPM är starkt relaterade till varandra. I Erl:s analys- och designmetod är affärsprocessernas inblandning i SOA-projekt en självklarhet. Rosen menar att det är fullt möjligt att bygga återanvändbara tjänster utan hjälp av BPM, men detta kan leda till svårigheter att applicera på hela verksamheten. Integrationschefen på Skanska verkar vara bekant

med påståendet att SOA och BPM hör ihop och kanske i framtiden om SOA växer sig större inom koncernen, kanske BPM kommer att spela en större roll vid utveckling av tjänster.

Angående affärssystem
Något som framkom vid intervjuerna var att Skanska idag använder ett stort affärssystem. De tjänster som hittills tagits fram är tänkta att fungera som informationsförmedlare mellan detta affärssystem och andra system. Affärssystemet innehåller idag affärsfunktioner som skulle kunna modelleras som tjänster i framtiden. Tanken med SOA är att kunna göra verksamheten mer flexibel för förändringar. Något vi har funderat på är om ett affärssystem egentligen är så förändringsvänligt. Projektledaren trodde att tjänster i framtiden kommer att vara mer funktionsinriktade till skillnad från de tjänster idag som fungerar som informationsförmedlare. Vidare nämner han att vissa av dessa funktioner inte stöds i affärssystemet utan kommer att byggas utanför. Affärssystemet klarar inte av de nya krav som ställs och planer finns för att bygga funktioner utanför systemet. Detta är ett intressant fenomen och vad som kan inträffa är att allt fler funktioner byggs utanför affärssystemet när nya krav ställs på Skanska. Detta anser vi tyder på en svårighet att förändra i befintliga affärssystem, som inte är uppbyggda med SOA-tanken i grunden och detta är en detalj som SOA kan lösa.

Tjänsteuppbyggnad
Erl anser att sammansättningen av befintliga tjänster bör ses över. Han påtalar betydelsen av att se över hur sammansättningen av grovmalda och finmalda tjänster görs. Vidare påstår Erl att finmalda tjänster ger högre återanvändbarhet men en hög grad av finkornighet innebär större krav på prestanda i systemet. Vi fick tvetydiga svar från Skanska och Zystems. Vi fick inget egentligt svar på vad som är att föredra, finkorniga eller grovkorniga tjänster. Det verkar uteslutande vara så att Skanska har bestämt sig för var de skall använda fin- respektive grovkorniga tjänster. Erl rekommenderar att finkorniga inom fördefinierade gränser och grovkorniga tjänster används vid end points. Skanska ger inget sken av att tänkt på detta sätt, men det behöver inte betyda att de inte tänkt på det över huvudtaget. Därför är det svårt att entydigt avgöra hur viktigt det är att tänka på sammansättningen som helhet.

Problemet med samarbete inom verksamheten
Under utförda intervjuer framkom att ett stort problem är att komma överens inom en organisation. Detta problem ligger utanför vårt ursprungliga fokus men det innebär inte att det är mindre intressant, för det pekar på att det existerar ett stort problem som SOA inte hanterar. En lösning med SOA kräver mer eller mindre att det existerar gemensamma definitioner på objekt, rutiner, struktur och ansvar inom en verksamhet. Allt eftersom fler externa organisationer inkluderas i verksamheten, medför detta ytterligare utmaningar eftersom det redan är svårt nog att komma överens inom en verksamhet. SOA har vuxit fram ur en gemensam satsning på en standardiserad teknologi och det är tack vare detta som SOA har vunnit mark. Men SOA är så mycket mer än teknik. Det svåra verkar vara att komma överrens, även på ett plan som ligger över tekniken.

Enligt Erl och Newcomer är en av grundpelarna med SOA, återanvändbarhet. Det innebär kort att man skall kunna flytta på och återanvända tjänster där det behövs i systemet. Detta innebär alltså att det är oerhört viktigt med god kontroll och överblick av de tjänster som befinner sig i det systemet man utvecklar. Både Erl och Newcomer förespråkar en UDDI-lösning. Skanska har här en egen lösning, istället använder man sig av en så kallad Wikibaserad lösning som innebär att man helt enkelt skriver in metadata i text dokument som sedan orienteras med hjälp av html länkar. Wikilösningen ger också användaren möjlighet till att söka på, uppdatera och versionshantera informationen allteftersom det behövs.

Eftersom SOA främjar föränderlighet är det enormt viktigt att kontrollera vad vad som gör vad (tjänster), i ett sådant perspektiv är det oerhört viktigt att undvika en spagettistruktur/linssoppa. För att uppnå en klar och tydlig struktur av tjänster, finns det olika tekniker, varav UDDI är en.

5.1 Slutsats

För att göra en snabb återkoppling till uppsatsens syfte och frågeställning, var målet med studien att bidra till ökad kunskap om SOA, samt att ta reda på hur design av SOA-tjänster ser ut hos ett byggnadsföretag i Sverige. Vår huvudfråga ”Hur designas en SOA-tjänst?” låg till grund för studien. Vår delfråga var vilka argument finns för val av processer som skall understödjas av tjänster. Inget i vår studie talar direkt emot de saker som nämns i teorin angående hur tjänster bör designas. Vi anser därför att det är gångbart att använda sig utav den teori vi presenterat för design av tjänster. Däremot har vi funnit att det i praktiken är väldigt viktigt att veta hur befintliga system fungerar och vikten av detta tas upp i presenterade teorier. Det har även framkommit att det är viktigt med ett tjänstregister för att lättare hålla ordning på tjänsterna.

När det gäller vår andra fråga ”Vad ligger bakom valet av processer som skall understödjas av tjänster”, så kan vi inte påstå att vi har fått ett tydligt svar. Det visade det sig att Skanska och Zystems inte har använt sig av processer vid framtagandet av sina tjänster. Vi kan dock inte utesluta att de tjänster som idag är i bruk hos Skanska faktiskt ingår i någon process. Svaret på frågan verkar vara att det är dels behovet som styr vilka tjänster som skall byggas, men även att det bör vara möjligt att börja i liten skala.

6 Referenser

Litteratur

Andersen, H (1994) *Vetenskapsteori och metodlära – En introduktion.* Lund: Studentlitteratur.

Backman, J (1998). *Rapporter och uppsatser*. Studentlitteratur .

Bell, J (2000) *Introduktion till forskningsmetodik* Lund: Studentlitteratur.

Easterby-Smith M, Thorpe R, Lowe A. (2002). *Management Research – An introduction*. *Second edition*, Great Britain: The Cromwell Press Ltd.

Ejvegård, R (2003) *Vetenskaplig metod*, Lund: Studentlitteratur.

Erl, T (2005) *Service-Oriented Architecture – Concepts, Technology and Design*. United States: R.R. Donnelley, Third printing.

Erl, T (2004) *Service-Oriented Architecture – A Field Guide to Integrating XML and Web Services*. United States, Seventh printing.

Hartman, J (1998) *Vetenskapligt tänkande – Från kunskapsteori till metodteori.* Lund: Studentlitteratur.

Krag Jacobsen, J (1993) Intervju – *Konsten att lyssna och fråga.* Lund: Studentlitteratur.

Magoulas, T Pessi K. (1998*). Strategisk IT-management.* Västra Frölunda: Vasastadens Bokbinderi AB

Newcomer E, Lomow E, (2005). *Understanding SOA with Web Services*. United States: Phoenix Color Corp, Second printing.

Nordberg, K (2000) *Projekthandboken – Metod och process.* Förlags AB Björnen.

Patel, R & Davidson, B (2003) *Forskningsmetodikens grunder – Att planera, genomföra och rapportera en undersökning.* Lund: Studentlitteratur.

Artiklar

Kim T, Lee S, Kim K, Kim C, (2005) A modeling framework for agile and interoperable virtual enterprises. Computers in Industry 57 (2006) 204–217, *Science Direct*

Rosen M, (2006), BPM and SOA Where does one end and the other begin?, *A BPT COLUMN*

Sprott D, (2004), Service Oriented Architecture: An introduction for managers. (2004) 3, 6 CBDI Forum Limited, *1.ibm.com*

Zachman J, (1987). *A Framework for information system architecture* IBM Systems Journal, Vol26 1987

WWW

[Al1]: http://www.anderslundgren.se/gpage.html [2006-03-20]

[Cs1]: http://www.cs.umu.se/education/examina/Rapporter/456.pdf [2006-04-20] Sida 68

[Ibm1]:ftp://ftp.software.ibm.com/software/websphere/literature/cbdireport_soa_july2004.pdf [2006-04-15] Sida 6

[Ibm2]: http://www-128.ibm.com/developerworks/webservices/newto/ [2006-04-28]

[Ne1]:http://www.ne.se/jsp/search/article.jsp?i_art_id=234209&i_word=kvalitativ%20metod [2006-05-12]

[Ne2]:http://www.ne.se/jsp/search/article.jsp?i_art_id=234260&i_word=kvantitativ%20metod [2006-05-12]

[Sb1]: http://dsv.su.se/soa/serviamdokument/SUF-10%20UDDI%20versikt.pdf [2006-05-05] Sida 12

[Sd1]: www.sciencedirect.com [2006-05-28]

[Sk1]: http://www.skanska.se [2006-05-25]

[St01]: http://www.statskontoret.se/upload/Publikationer/2001/2001312.pdf [2006-05-05] Sida 24

[Su01]: http://susning.nu/Process [2006-05-25]

[Su02]: http://susning.nu/Tj%e4nst [2006-05-25]

[Wp1]: http://sv.wikipedia.org/wiki/Induktion_%28filosofi%29 [2006-05-27]

[Wp02]: http://sv.wikipedia.org/wiki/UDDI [2006-03-24]

[Wp03]: http://sv.wikipedia.org/wiki/ESB [2006-05-26]

[Wp04]: http://sv.wikipedia.org/wiki/Process [2006-05-25]

[Wp05]: http://sv.wikipedia.org/wiki/Webbtj%C3%A4nster [2006-05-20]

[Wp06]: http://sv.wikipedia.org/wiki/Design [2006-06-05]

[Wp07]: http://sv.wikipedia.org/wiki/Reliabilitet [2006-05-29]

[Wp08]: http://sv.wikipedia.org/wiki/Validitet [2006-05-29]

[Zach87]: http://delivery.acm.org/10.1145/1050000/1040722/p8-zachman.pdf?key1=1040722&key2=9875820511&coll=GUIDE&dl=GUIDE&CFID=73622383&CFTOKEN=46067770 Sida 9 [2006-04-23]

[Zt1]: http://www.zystems.se [2006-05-25]

7 Bilagor

Bilaga 1 - Begreppsförklaringar

BAM
Business Activity Monitoring analyserar händelseförlopp genererade av affärsprocesser och information inhämtade från affärsprocesser. Detta för att förse data angående prestationsförmågan baserad på realtidshändelser.

Best practice
Är ett uttryck som hävdar att det finns en teknik, metod, process, aktivitet, incitament eller belöning som är mer effektiv att leverera ett visst resultat än någon liknande teknik, metod eller dylikt [Wp07].

End point
En tjänst som kan vara den sista i en kedja av tjänsteförfrågningar som tidigare startats av en annan tjänst. Den tjänsten kallas för end point-tjänst.

ESB
Enterprise Service Bus, kombinerar meddelandehantering, transformering, säkerhet och transaktioner för att skapa en applikationsmässig infrastruktur byggd på Web Services. SOA-arkitekturen stöds genom att ESB implementerar SOAP, WSDL och eventuellt UDDI.

De viktigaste funktionerna för en ESB är:

Meddelandetransformering – Transformering av data från proprietära dataformat till ett gemensamt XML-baserat format som kan förstås av båda sändande och mottagande applikationer.

Innehållsbaserad dirigering – Bestämning av ett meddelandes destination baserat på dess innehåll, vilket befriar den sändande applikationen från att känna till alla tänkbara mottagare.

Publicering och prenumeration – En händelsedriven modell som bygger på att en händelse som inträffar i en viss applikation får en annan händelse att inträffa i en annan applikation.

Säkerhet – Ett ramverk för att möjliggöra säkra överföringar av meddelanden mellan applikationer i ett distribuerat system.
Transaktioner – Möjlighet till att utföra transaktioner som är både asynkrona och synkrona. [Wp03]

Finkornig tjänst: Se atomära tjänster.

Grovkornig tjänst: Se sammansatta tjänster.

Lös koppling
Ingen kan förutse hur en IT-miljö kommer att utvecklas, hur automatiserade lösningar kommer att växa, integreras eller blir ersatta över tid. Detta på grund av att sådana förändringar ofta har att göra med orsaker som ligger utanför den miljö som systemet verkar i. Att kunna anpassa sig efter sådana förändringar ligger som grund för nyttan med tjänsteorientering. Mot denna bakgrund hamnar kravet och nyttan med lösare koppling. Lös koppling är ett tillstånd där en tjänst kan få information om en annan tjänst utan att vara beroende av densamma. För att uppnå lös koppling krävs det att man har någon form av tjänstekontrakt som tillåter tjänster att interagera med fördefinierade parametrar. Tack vare lös koppling mellan tjänsterna så behöver en applikation inte få någon teknisk information om den andra applikationen innan de kan kommunicera (Erl 2005).

UDDI

Universal Description, Discovery and Integration är en standardiserad katalogmodell för webbtjänster. En UDDI-katalog - innehåller information om webbtjänsters funktionalitet och beskrivning [Wp02].

Vad är UDDI?
UDDI är ett plattformsoberoende ramverk för att beskriva tjänster/service, upptäcka företag, och att integrera företagstjänster genom att använda Internet.

- UDDI står för Universal Description, Discovery and Integration
- UDDI är en katalog/bibliotek för att spara information om Web Services
- UDDI är en katalog/bibliotek av Web Service-gränssnitt, beskrivet av WSDL
- UDDI kommunicerar via SOAP

Tanken med UDDI är att hjälpa företag att hitta varandra och förmedla företagens behov och utbud. Det fungerar ungefär som en elektronisk telefonkatalog, och det är också den metafor som används inom UDDI. Det finns vita sidor för namn och kontaktinformation, gula sidor för verksamhetskategorier och gröna sidor för de tjänster som erbjuds. Ansvariga för UDDI var ursprungligen IBM, Microsoft och Ariba. Idag är drygt 130 företag med i utvecklingen [St01]. Stig Berild (2001) konstaterar att det finns en hel del frågetecken kring driften av UDDI. Någon måste ansvara för driften och detta kan knappast bygga på idealitet. Det måste finnas någon form av incitament för den ansvarige, kanske rent ekonomiska. Kommer en prislapp att införas? Det måste också finnas någon som ansvarar för innehållets kvalitet. Det är klart fördelaktigt om informationen är aktuell. Hur kan kvaliteten kontrolleras? Vad händer om uppdaterare missköter sig? [Sb1].

Use case
Inom programvaruutveckling är ett användarfall ett sätt att inhämta krav på ett nytt system eller ändring på befintlig programvara. Användarfall innehåller inte tekniska beskrivningar utan mer användbara beskrivningar från slutanvändaren.

Web Services
Web Services är en teknik som använder sig av XML för att tillåta olika applikationer att kommunicera med varandra. Det är plattforms- och programspråksoberoende och fungerar som verktyg för att implementera SOA i en verksamhet. Web Services är dock inte det enda sättet att implementera SOA men i dagsläget det vanligaste (Erl 2005). Språket som används är XML. Det strukturerar och märker informationen så att den blir lättare att hålla reda på, söka i, publicera och återanvända. Med hjälp av XML kan man definiera taggar, det vill säga märkord som är speciellt anpassade till den information som skall beskrivas. Dess struktur baseras på, för ändamålet, definierade etiketter. Dessa etiketter delar upp ett dokument i en hierarkisk struktur och identifierar dokumentets olika delar eller element (Erl 2004). Till skillnad från HTML, som är ett rent märkspråk, kan användaren själv definiera taggarna och därigenom skapa sin egen struktur anpassad för lagringen av sina data
När vi utvecklar Web Services så utvecklar vi med hjälp av en teknik som heter SOAP eller Simple Object Access Protocol. Med hjälp av SOAP kan vi på standardiserade sätt skicka meddelanden beskrivna i XML över det vanliga transportprotokollet som används över Internet, nämligen HTTP. Tack vare att vi skickar ren XML över HTTP så får vi också möjligheten att skapa lösningar som är standardiserade och kan oavsett programmeringsspråk eller operativsystem användas på klient och serversidan.
World Wide Web Consortium definierar en webbtjänst för tillfället enligt följande: ”A Web service is a software application identified by a URL, whose interfaces and bindings are capable of being defined, described, and discovered as XML artifacts. A Web service supports direct interactions with other software agents using XML based messages exchanged via internet-based protocols.”

- För att skicka data mellan applikationerna används protokollet HTTP.
- För att beskriva applikationernas gränssnitt används XML.
- Typiska standarder för att bygga Web Services inkluderar SOAP, WSDL och UDDI.
- Den dator som kör en webbtjänst brukar kallas applikationsserver.

Web Services kan samarbeta med varandra automatiskt och utan avbrott. Det är möjligt eftersom Web Services per definition är utvecklade i enlighet med samma standarder för självbeskrivning, publicering, lokalisering, anrop, kommunikation och datautbyte. En mindre teknisk beskrivning av det dynamiska beteendet som Web Services uppvisar är att de kännetecknas av:

- att de är applikationer
- att de officiellt och publikt beskriver sina funktionalitet
- att de kan lokalisera efterfrågad funktionalitet
- att de kan efterfråga och utbyta data med andra Web services [Wp05].

Bilaga 2 - Intervjufrågor

Frågeställning:

- Hur designas en SOA-tjänst?
- Argument för val av processer som skall understödjas av tjänster

Projektet, projektansvariga:

1. Följde projektet någon bestämd metod eller rutin, RUP eller liknande?
   Om inte, vilka komplikationer uppkom under projektets gång?
   Om ja, vilka komplikationer uppkom under projektets gång och vilken specifik metod användes?

2. Hur resonerade ni när ni valde ut funktioner som skulle bli tjänster? Hur började ni?

3. Använde ni någon typ av modell av verksamhetens struktur, processer och funktioner?

4. Har ni en tydlig koppling mellan de tjänster som skapades och befintliga processer?

5. Blev resultatet det önskade eller blev det stora avvikelser från ursprungsplanen?

6. Är det något särskilt moment ni känner att ni skulle vilja förbättra vid utveckling av tjänster?

Personal:

7. Vem/ vilka var ansvariga för val och utveckling av tjänsterna? Dvs vilka positioner besitter dessa personer, samt vilka tidigare erfarenheter av SOA har dessa?

8. Har någon som var inblandad i projektet haft någon form av utbildning inom SOA-området?

SOA teknikerfrågor:

9. Vad är det som gör er tjänstestruktur till ett SOA? Alltså vad är er definition på SOA, och kan du ge exempel på en tjänst som har något karaktärsdrag som är specifikt för SOA?

10. Vi har identifierat ett antal olika tjänster i vår litteraturstudie av SOA. De som presenteras är applikationstjänster, affärstjänster samt orkestreringstjänster. Är detta tjänster som låter bekanta? Vilka slags tjänster har ni infört hittills?

11. Vilka slags tjänster tror ni kommer att implementeras framöver?

12. En stor fördel att använda små finkorniga tjänster är att man får stor återanvändningsbarhet. Finns det något som ni anser vara negativt med att använda enbart finkorniga tjänster? (kanske lite ledande? men ändå intressant att få veta deras syn på det hela)

13. Anser ni att det skulle vara bra med någon form av analysmetod vid framtagningen av tjänster?

Ledningsfrågor:

14. Vad var bakgrunden till projektet?

15. Varför föll valet på SOA? Vilka effekter var önskvärda vid ett införande?

16. Har ni märkt av några av de önskade effekter ni eftersträvade?

17. Har ni märkt av effekter som ni inte räknade med eller inte var önskvärda?

18. Fanns några andra alternativ än tjänsteorienterad lösning som ni tittade på i samband med förändringsplanerna?

19. Tror ni att ni kommer att utöka er nuvarande tjänstearkitektur till att omfatta hela organisationen?

Övrigt:

20. Vill man ha mer/mindre integration mellan system och hur tycker ni SOA hanterar detta? (Tänk på möjliga motsättningar ex loose coupling.)

21. Något annat att tillägga?

Bilaga 3 – Intervjufrågornas teoretiska förankring

Nedan visas den koppling som finns mellan utformade intervjufrågor och presenterade teorier.

Teoretisk förankring **Intervjufråga**	**Zachman**	**Erl 2004 2005**	**Newcomer**
1			x
2			x
3			x
4			x
5		x	x
6		x	x
7		x	x
8		x	x
9		x	x
10		x	x
11		x	x
12		x	x
13			x
14	x		
15	x		
16		x	x
17		x	x
18	x		
19		x	x
20			x