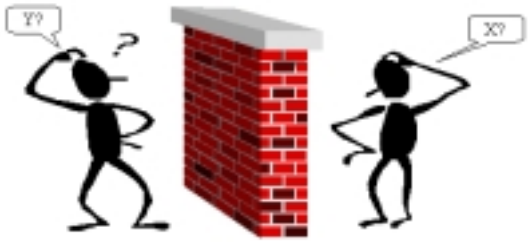
Institutionen för Informatik
Handelshögskolan vid
Göteborgs universitet

Systemutvecklare vs. kund

- hur överbrygga kommunikationsproblem?



Abstrakt

Vi upptäckte i vårt första systemutvecklingsprojekt utanför skolans regi att det fanns stor risk för missförstånd i kommunikationen mellan oss utvecklare och kunden. Detta gjorde att vi blev nyfikna på varför kommunikationsproblem uppkom och hur de skulle undvikas. Vi fann att det till stor del berodde på faktorerna *tidsbrist*, motparternas *olika bakgrund*, *otydlighet*, *okunskap*, *oerfarenhet*, *språk*, *komplicerade system* samt *dålig sammanhållning* inom projektgruppen. Vi undrade även om det fanns standardiserade metoder inom systemutvecklingsvärlden för att undvika dessa problem.

Av våra litteraturstudier, intervjuer och egna erfarenheter fann vi att så inte var fallet, men vi lyckades ändå få fram såpass mycket information om ämnet att vi ansåg oss kapabla att ge några riktlinjer för hur kommunikationen kan underlättas. Dessa var att *våga fråga, undvika fackspråk, lyssna uppmärksamt, träffa motparten, göra prototyper, dokumentera, ha gott om tid, ha kunskap om människans kognitiva förmåga* samt *ha kunskap om kundens organisation.*

Theresia Höglund & Niklas Borg
Examensarbete 1 10p IA5840
ADB-programmet 80p
Vårterminen 2000
Handledare: Maria Bergenstjerna fil.mag.

”Erfarenhet är det namn
människorna ger
åt sina misstag”
Oscar Wilde

Tack till…

Ett speciellt tack vill vi ge vår handledare Maria Bergenstjerna som ställde upp för oss i vått och torrt och som ständigt sökte rätt på oss i någon av datasalarna och undrade hur det gick.

”Även den längsta resa börjar
alltid med ett enda steg.”
Kinesiskt ordspråk

INNEHÅLLSFÖRTECKNING

1 Introduktion

I februari år 2000 kom vi genom en vän på vår skola i kontakt med ett företag som ville ha ett system byggt.

Vid första mötet mellan oss och det aktuella företaget försökte vi ta till oss vad de ville ha för typ av system. Efter ytterligare ett par möten och ett par e-mail fram och tillbaka tyckte vi att vi fått en någorlunda klar bild av hur systemet skulle se ut och satte igång med att göra en kravspecifikation. Då denna refuserades och omarbetades ett antal gånger så insåg vi till slut att vi talade helt olika språk. Vi talade vårt systemutvecklingsspråk och företaget talade sitt fackspråk. Någon sorts brygga för att träda in i varandras språkvärldar saknades.

Under skoltiden hade vi fått många tillfällen att öva praktiskt på att jobba i projekt. Skillnaden var att vi då själva spelade de flesta roller i systemutvecklingsprocessen såsom användare, utvecklare och beställare. Att helt plötsligt möta en verklig kund och försöka ta till sig dennes krav och önskemål var en ny företeelse för oss. Vi insåg ganska snart att vi inte hade tillräckliga kunskaper för att utan problem kunna göra oss förstådda för och att förstå kunden. Under denna tid föddes idén om att skriva en uppsats för att belysa de svårigheter som kan uppstå i kommunikationen mellan utvecklare och kund.

2 Bakgrund

2.1 Problembeskrivning

I samband med systemutveckling är det väsentligt att utvecklaren är helt på det klara med kundens önskemål så att det färdiga systemet kommer att fungera på ett tillfredsställande sätt. Ett system är numera en oupplöslig del av kundens organisation och det är därför också viktigt att utvecklaren förstår de delar av organisationen och dess funktioner som systemet kommer att beröra.

Ändå uppstår ofta situationer där ett system inte blir som kunden tänkt sig. Detta framhålls av Langefors (1995): *”... En viktig del av projektet är att tillsammans med användaren komma fram till vilka behov och önskemål som finns. Trots detta var det vanligt att användaren sa att det nya systemet inte var vad de frågat efter”.*

Detta anser vi oftast bero på missförstånd i kommunikationen mellan utvecklare och kund. Problemet, svårigheten att förstå varandra, kan egentligen ha två vinklar, dels att de båda parterna inte förstår varandra och är medvetna om det, och dels att de tror att de förstår varandra fast det i själva verket kan vara tvärtom. Den sistnämnda vinkeln anser vi vara den ”farligaste” formen av missförstånd eftersom systemutvecklingen kan ha hunnit långt och systemet kan t.o.m. vara färdigbyggt innan det upptäcks att utvecklaren och kunden talat om olika saker.

Keller (1998) skriver ”… bland annat har systemvetarna svårt att lyssna på icketeknisk kunskap. Användarnas språk blir helt enkelt för amatörmässigt”.
Han menar vidare att skillnaderna i tekniskt kunnande även leder till att användaren har svårigheter att uttrycka sin krav för systemutvecklaren, just eftersom systemutvecklaren har svårt att lyssna på/inhämta kunskap som beskrivs med ett icketekniskt språk.

Svårigheter i kommunikationen är inget som bör ignoreras. De leder till onödiga kostnader, förseningar, dålig PR för utvecklingsföretaget och i värsta fall projekt som går i stöpet. För att spara tid och resurser är det därför viktigt att utvecklare och kund redan på ett tidigt stadium får varandras visioner om systemet att stämma överens.

Hur görs då detta? Hur ska kommunikationsproblemen minimeras och i bästa fall avlägsnas? Varför är det så svårt att förstå varandra? Det är dessa frågor som vi ska försöka belysa i denna uppsats.

2.2 Syfte och frågeställning

Som vi tidigare nämnt kan missförstånd i kommunikationen innebära allvarliga problem. Det är därför ett viktigt ämne som tidigare, enligt våra efterforskningar, har fått alldeles för lite uppmärksamhet. Vi upplever även att just den mer ”mjuka” och psykologiska biten av systemutveckling, som kommunikationsproblemen bör tillhöra, är något som har tagits upp lite för lite under vår utbildning.

Genom att belysa och analysera detta ämne förväntar vi oss få en fördjupad kunskap om vad som orsakar kommunikationsproblemen samt finna standardiserade metoder för att undvika dem eftersom det är ett så stort och vanligt problem. Förhoppningsvis

kan vi även komma fram till någon slags riktlinje eller punkter att följa för hur utvecklare kan bete sig för att ge kunden ökad förståelse.

Detta anser vi vara nyttig kunskap för oss inför framtida arbete inom branschen. Vi hoppas även att andra har nytta av att ta del av vår undersökning eftersom ämnet som sagt verkar vara lite undanskymt och nästan ignorerat.

För att klara av den här uppgiften måste vi ställa nedanstående huvudfråga:

- *Hur underlättas kommunikationen mellan utvecklare och kund?*

För att kunna lösa någonting, i detta fall kommunikationsproblemen, anser vi att källan till dem måste vara känd. Därför blir vår delfråga:

- *Varför uppstår kommunikationsproblem?*

2.3 Avgränsning

Uppsatsen behandlar *främst* de kommunikationsproblem som kan uppstå i förhandlings- och utvecklingsarbetet när systemutvecklare möter kunder inom andra branscher än IT. Uppsatsens perspektiv är satt ur systemutvecklarens synvinkel. Kommunikation handlar visserligen om två parter som försöker förstå varandra men i vår uppsats koncentrerar vi oss i första hand på att göra *oss* förstådda för kunden.

2.4 Uppsatsens disposition

Vi har med hjälp av litteraturstudier, egna erfarenheter i vårt nuvarande systemutvecklingsprojekt samt intervjuer försökt förstå problemet med kommunikationssvårigheter.

Uppsatsen består av dels en teoridel och dels en empirisk del och vi har delat upp de olika delarna enligt nedanstående:

Kapitel 3 behandlar relevanta begrepp.
Kapitel 4 behandlar de olika litteraturstudier vi gjort och lyfter fram de teorier om meningsfull och lättförståelig kommunikation mellan människor som vi har hittat.
Kapitel 5 redogör för de intervjuer vi gjort samt behandlar våra egna erfarenheter under vårt första systemutvecklingsprojekt utanför skolans regi.
Kapitel 6 innehåller analyserar och diskuterar det resultat vi kommit fram till.
Kapitel 7 avslutar själva uppsatsen med sammanfattning, faktorer som kan ha påverkat uppsatsens reliabilitet och validitet samt förslag till forsatta studier.
Kapitel 8 utgörs av våra referenser.
Kapitel 9 presenterar våra intervjufrågor i form av en bilaga.

3 Relevanta begrepp

Innan vi ger oss i kast med problemet vill vi klargöra ett antal begrepp som vi uppfattar som viktiga.

Ämnet vi har valt att behandla är alltså kommunikationsproblem. Vissa frågeställningar uppkommer då som vi redan nu vill klargöra.

- Vilka är det som kommunicerar?
- Varför kommunicerar de?
- Vad kommunicerar de om?

I något omvänd ordning är svaren, markerade nedan, på dessa tre frågor följande:

Systemutvecklaren kommunicerar med **kunden** för att få fram **specifikationer** om **krav** och riktlinjer för det **system** som ska **utvecklas**.

3.1 Vad är en kund respektive systemutvecklare?

Med ordet *kund* menar vi *beställare, ägare, användare* eller *annan person* som har intresse av hur systemet kommer att se ut och därför kommunicerar på olika sätt med *systemutvecklaren* under systemutvecklingsprocessen. Systemutvecklaren är alltså motparten till kunden, och är den eller de personer som utvecklar, konstruerar och levererar systemet.

3.2 Vad innebär systemutveckling?

Vi har valt att kortfattat beskriva systemutvecklingen för att försöka ge klarhet i var någonstans i systemets livscykel som kommunikationen mellan kund och utvecklare äger rum.

För att förstå systemutveckling vill vi först klargöra vad ett system är. Det beskrivs av Sommerville (1996) som en samling av sammanhängande delar som tillsammans, i en viss omgivning, arbetar för att uppnå vissa mål.

Utvecklingen av detta kan beskrivas som en process och är enligt Sommerville (1996) en tvärvetenskaplig aktivitet som kräver samarbete mellan flera personer. Detta för att de tillsammans har den kunskap som krävs för att förstå innebörden av alla systembeslut. Processen brukar delas upp i olika faser där antalet varierar lite beroende på vilken litteratur man läser. Enligt Brown (1997) består processen av följande fem faser:

Analys – under denna fas (som av Eklund & Fernlund (1998) kallas definitionsfasen) studeras användarens företag för att komma fram till vad systemet ska utföra för att kunna hjälpa användarna. Riktlinjer för projektet ska tas fram och programmets funktioner ska bestämmas. Dessa dokumenteras med hjälp av ett antal modeller, dokument och diagram. Enligt Sommerville (1996) ska denna fas också producera en kravspecifikation där krav och mål med systemet kartläggs. Vad en sådan innebär mer specifikt tas upp i nästa paragraf.

Design – under denna fas produceras en plan som visar hur det är tänkt att systemet ska utföra de funktioner som identifierades i analysfasen. Beslut tas i samförstånd av utvecklare och kund om plattform, språk, operativsystem och databasmjukvara.

Databas, program, skärmbilder och rapporter designas vilka sedan ska ligga till grund för nästa fas.

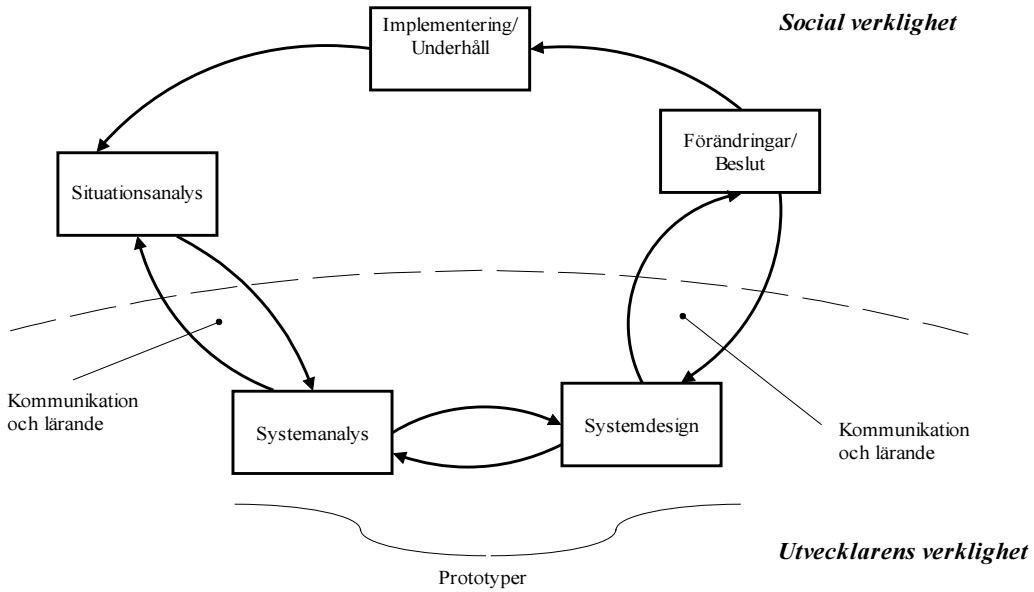
Konstruktion – under denna fas kodas programmen och databasen byggs. Hela systemet testas sedan på olika sätt.

Implementering – under denna fas installeras systemet och användarna utbildas.

Underhåll – under denna fas genomförs kontinuerliga förbättringar och felkorrigeringar av systemet.

Det är främst under de första två faserna, analys och design, som det är viktigt att kunden och utvecklaren kommunicerar med varandra och kommer överens. Det är ju dessa två faser som till stor del avgör hur arbetet under de följande faserna ska läggas upp och om inte kunden har fått fram sina önskemål ordentligt är risken stor att systemet inte kommer att fungera tillfredsställande.

Checkland (1989) beskriver faserna på ett lite annorlunda sätt. Detta visar figur 3.1.

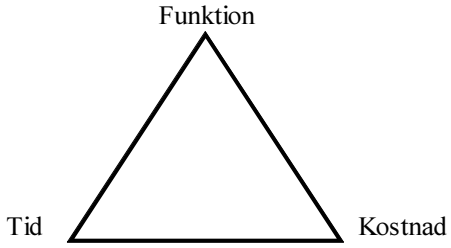


Figur 3.1 Systemutvecklingsprocessen, en omarbetad modell enligt Checklands(1989) SSM[1]

3.3 Vad är en kravspecifikation?

Under analysfasen ska alltså en överenskommelse om en *kravspecifikation* nås. Denna ska sedan ligga till grund för kontraktet mellan utvecklingsföretaget och kunden. Det är alltså till stor del kravspecifikationens innehåll som är föremål för diskussion. Kravspecifikationen beskrivs av Eklund & Fernlund (1998) som ett kontrakt där hela åtagandet gentemot kunden fastställs, ex funktioner, pris, omfattning, leveransvillkor m.m.

[1] Soft Systems Methodology behandlas i kapitel 4.4



Figur 3.2 ur ”Programkonstruktion” av Eklund & Fernlund(1998)

De tre hörnen hos Figur 3.2, ”Åtagandetriangeln”, representerar enligt Eklund & Fernlund som de grundläggande och sammanlänkade egenskaperna hos ett åtagande – *tid, kostnad* och *funktionalitet.*

Kravspecifikationen kan tas fram på tre olika sätt. Kraven kan vara:

- *Definierade av kunden*

Detta är ovanligt p. g. a att det ställs mycket höga krav på kundens tekniska kompetens.

- *Definierade av leverantören*

Detta är vanligt vid program som säljs i stora upplagor, ex Microsoft Office

- *Framtagna i dialog*

Detta är den metod som bör vara mest eftersträvansvärd

Eklund & Fernlund tar vidare upp att en leverantör bör, förutom att diskutera kundens krav på programvaran, gå igenom följande punkter:

- *Identifiera kunden*

Ta reda på vem på kundens sida som kan fatta beslut om systemet.

- *Analysera existerande system*

Beskriv det existerande systemet.

- *Intervjua användare*

Sätta sig in i användarnas situation och försöka skapa sig en bild av den.

- *Titta på hur omgivningen ser ut*

Ta reda på om omgivningen ställer speciella krav på programvaran som kunden tar för givet och inte berättar om(ex. driftsstörningar).

- *Göra en hierarkisk problembeskrivning*

Beskriv problemet på mer och mer detaljerade nivåer och tala om vad som skall lösas och inte.

- *Rubricera kraven i olika nivåer*

Få kunden att förstå det svåra i att få allting till låg kostnad och med kort leveranstid.

- *Dokumentera*

Skriv ner allt klart och tydligt.

- *Låta kunden känna ansvar och delaktighet i kravspecifikationen*

Få dokumentet med kundens krav, kravspecifikationen, gradvis godkänd av kunden allteftersom arbetet fortlöper.

Viktigt att tänka på när det gäller själva innehållet i kravspecifikationen är enligt Eklund & Fernlund (1998) att:

- kravspecifikationen inte är något designdokument, den säger inget om *hur* problemen skall lösas.
- allt som programvaran ska kunna göra ska vara specificerat i kravspecifikationen.
- kraven är skrivna på ett sådant sätt att de kan kontrolleras mot den färdiga programvaran.
- inga krav står i konflikt med varandra.

Lubars, Potts & Richter (1993) skriver att kravspecifikationen helst ska valideras av kunden och om den *inte* valideras är risken stor att det finns fel i den. Korrigeras inte detta innan själva programmeringsfasen börjar blir det dyrare att reparera felet.

4 Litteraturstudier

Det finns en uppsjö av litteratur kring systemutveckling i allmänhet, men vi har inte hittat någon som specifikt behandlar kommunikationen mellan utvecklare och kund. Därför börjar vi med att ta reda på om kommunikationsproblem förekommer för att sedan studera vad kommunikation innebär i allmänhet. Därefter undersöker vi hur forskare inom Informatik valt att uttrycka sig angående problematiken gällande kommunikation mellan kund och utvecklare.

4.1 Förekommer kommunikationsproblem?

Innan man börjar fundera på varför kommunikationsproblem uppstår under systemutvecklingsarbetet och hur de ska kunna lösas anser vi att det klart bör fastställas att de verkligen förekommer samt att de verkligen får allvarliga konsekvenser.

Detta behandlas i Håkan Enquists (1999) uppsats som gick ut på att ta reda på frekvensen av kommunikationsproblem samt vad de brukar leda till.
Enquist tar upp att svårbegripligheten i systemutvecklingsprocessen delvis ”skylls” på involverandet av ett stort antal aktörer med olika ”språk”, erfarenheter, ambition, kunskap och intressen. Personliga skillnader leder till avvikelser i hur dessa aktörer uppfattar de riktigt väsentliga sakerna i systemutvecklingsarbetet. Enquist tror vidare att den redan ansträngda tidsplanen gör att aktörerna inte hinner ’lära sig’ en gemensam referensram som båda förstår.

Enquist refererar till en studie som visar att 56% av felen i de undersökta installerade systemen berodde på dålig kommunikation mellan användare och analytiker, samt att dessa feltyper var de mest kostsamma att rätta till (upp till 82% av tillgänglig personaltid). Vidare skriver han att designers och programmerare kan missförstå varandra eftersom de ofta har inkorrekt, förvillande eller ej komplett information om varandras arbetsrutiner.

Största delen av Enquists uppsats grundade sig på en empirisk undersökning bestående av en kvalitativ tvådagars workshop samt en kvantitativ enkät.
Resultatet visar att missförstånd anses ofta förekomma bland aktörerna och är källa till störningar i systemutvecklingsprocessen när det gäller komplexa systemprodukter. Ca 30% av de svarande anser att konsekvenserna av missförstånd *vanligtvis* orsakar mindre störningar medan 5% anser att de skapar större störningar.
Försening av aktiviteter rankas som den vanligaste konsekvensen av missförstånd, tätt följt av *oenighet/motvilja mellan aktörer* och på tredje plats hamnar *fel i produkten*.

Enquists undersökning visar att den vanligaste orsaken till att *inte* reagera på missförstånden är att det *inte anses vara viktigt*. Det visar ju att det saknas en medvetenhet hos aktörerna om de negativa konsekvenser som missförstånd kan leda till. Vidare visar undersökningen att den vanligaste orsaken till missförstånd är *oklar/ej komplett uttryckt information* och den näst vanligaste är *skillnader i begrepp och referensram.* Även *osäkerhet angående uppgifter, ansvar, auktoritet och/eller syfte från andra aktörer* ansågs spela stor roll. Enquist drar slutsatsen att inte bara *vad* som sägs utan också *av vem* är mycket viktigt.

Några av de punkter Enquist kom fram till var:

- Missförstånd är väl införlivade i systemutvecklingen och därför värda att analysera.
- Missförstånd mellan aktörer i systemutvecklingen orsakar ofta mindre (och då och då mer omfattande) negativa konsekvenser. Flera olika och av varandra oberoende konsekvenser kan uppkomma från samma missförstånd.
- Skillnader i begrepp och referensramar mellan aktörer i systemutveckling är en ofta förekommande orsak till missförstånd med negativa konsekvenser.

4.2 Vad är kommunikation?

Vi försöker här redogöra för vad kommunikation är. Valda teorier försöker vi relatera till kommunikationen mellan kunden och utvecklaren. För att förstå kommunikationsprocessen mellan kund och utvecklare är det bra att känna till kommunikationsprocessens uppbyggnad och vad kommunikation i allmänhet egentligen handlar om. Vi kommer att koppla det mer specifikt mot systemutvecklingprocessen i kapitel 6.1.

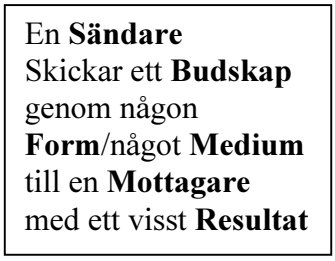
Dimbleby & Burton (1999) påpekar att det faktum att kunna tala med någon inte betyder att det som ska sägas går att överföra. Efter att ha skapat en förbindelse måste vi människor lära oss att använda den efter bästa förmåga. Vidare delar de in kommunikation i fyra kategorier.

- Intrapersonell kommunikation – Äger rum inom oss själva, exempelvis tankar.
- Interpersonell kommunikation – Äger rum mellan två personer, exempelvis intervju, samtal med vänner.
- Gruppkommunikation – Äger rum mellan personer som tillhör samma grupp och mellan olika grupper av människor, exempelvis familj, kompisgrupp.
- Masskommunikation – Innebär kommunikation som mottages eller används av ett stort antal personer, exempelvis konsert, massmedia.

Av dessa är det främst den andra och tredje*, interpersonell*, d.v.s. kommunikationen mellan två personer, och eventuellt *gruppkommunikation*, som rör vårt ämne. I första hand är det ju en kund och en utvecklare som pratar med varandra, eller en grupp av kunder som pratar med en grupp av utvecklare.

Kommunikation sägs ha ett behov och ett syfte. Människor måste ha en anledning att kommunicera och Dimbleby & Burton påpekar att det inte minst i arbetslivet är bra att vara klar över detta syfte. Boken tar upp flera olika syften med kommunikation och delar sedan in dem i tre kategorier: *personliga behov, sociala system* och *ekonomiskt tvång*. Vi menar då att det främst är de två sistnämnda som rör vårt ämne. Inom dessa tar D & B upp punkter som *samarbete, övertalning* och *information*.

Vidare beskriver de kommunikationen som en dynamisk process. Ett aktivt flöde av idéer, fakta och åsikter. Processen är fortlöpande och framskrider av feedback på och justering av sättet att kommunicera. De refererar till Harold Lasswell som 1984 beskrev kommunikationsprocessen i sina olika delar enligt figur 4.1.



Figur 4.1 Lasswells beskrivning

Figur 4.2 Vår omarbetning av Lasswells beskrivning

Dimbleby & Burton tar därefter upp att kommunikationen sker i något sammanhang, d.v.s. i någon slags omgivning. Denna är av två sorter, dels *det fysiska sammanhanget* och dels det *sociala sammanhanget*. Den förstnämnda tar upp var kommunikatörerna befinner sig rent fysiskt. Med det sociala sammanhanget avses den aktuella tilldragelsen och de berörda personerna. Till detta hör även det kulturella sammanhanget som syftar på en ännu bredare uppsättning omständigheter och övertygelser som kan påverka hur vi talar.

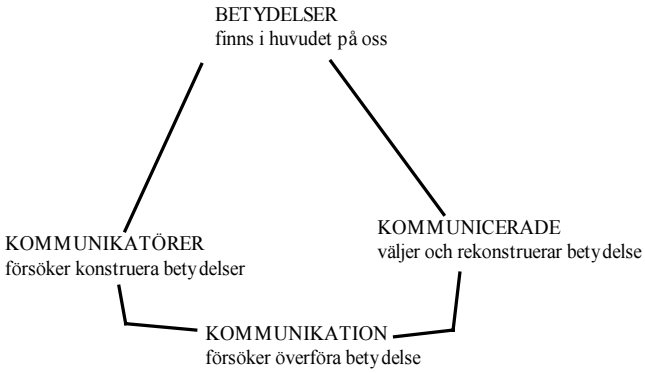
Vidare beskrivs hur ett budskap förs fram och tas emot i kommunikationsprocessen. Vid mottagningen registrerar, tolkar och lagrar vi budskap eller handlar genast. Det finns alltså två tillfällen då det kan bli fel. Dels vid *registreringen* och dels vid tolkningen.

Dimbleby & Burton hävdar att budskap sällan är neutrala, inte ens faktabudskap. Detta beror på sammanhanget de framförs i. Budskap kan vara öppna och tydliga eller förtäckta, beroende på om avsikterna visas eller inte.

Dimbleby & Burton refererar även till amerikanen D. K Berlo som beskriver budskapet i tre delar: *koden, innehållet* och *behandlingen*. Han hänvisar också till det faktum att vår kunskap, kulturella bakgrund, attityd och kommunikationsfärdighet påverkar hur vi kommunicerar med varandra. Detta påverkar vår effektivitet som kommunikatörer. Vad som händer när vi kommunicerar är alltså någonting som vi lärt oss att göra, på vissa sätt och av vissa orsaker.

Boken beskriver att budskap egentligen handlar om betydelser som vi utbyter mellan varandra. Dimbleby & Burton hävdar att det skulle uppstå färre problem i umgänget med andra om alla människor kunde överföra sina budskap exakt som de avser dem. Så är dock inte fallet eftersom tolkningen är olika. Det är därför inte bra att dra förhastade slutsatser om vad andra menar med vad de säger. Varje kommunikation betyder något för de personer som är med om att skapa eller ta emot den. De flesta kommunikationer betyder mer än en sak och inte samma sak för alla. Betydelsen ändras också efter tid och rum. En människas respons på ett budskap beror på vad det betyder för henne samt utbytessammanhanget och relationen med sändaren. Källan till betydelsen ligger främst inom mottagaren, i hennes huvud.

Betydelserna är sedan knutna till tecken, exempelvis ord, bilder och kroppsspråk, och det är dessa tecken som ges och tas emot. Om vi förstår varandras tecken går budskapet fram, annars inte.



Figur 4.3. Kommunikation och betydelse, allt finns i huvudet.
Ur ”Kommunikation är mer än ord”.

Dimbleby & Burton tar upp fyra problem med tecken som leder till missförstånd:

- att säga att någonting är ett tecken talar inte om vad dess betydelse är
- samma tecken kan ha olika betydelser på olika platser eller vid skilda tidpunkter
- ett tecken kan ha mer än en betydelse
- samma tecken kan ha olika betydelse för olika människor

Dimbleby & Burton förklarar att lösningen på problem nummer ett är att vi lär oss att associera ett tecken med betydelse, t.ex. genom att lära oss ett språk. Om man inte vet betydelsen är tecken helt värdelösa. Dimbleby & Burton skriver dock att vi kan förstå ett teckens betydelse eller innerbörd genom andra, omgivande tecken.

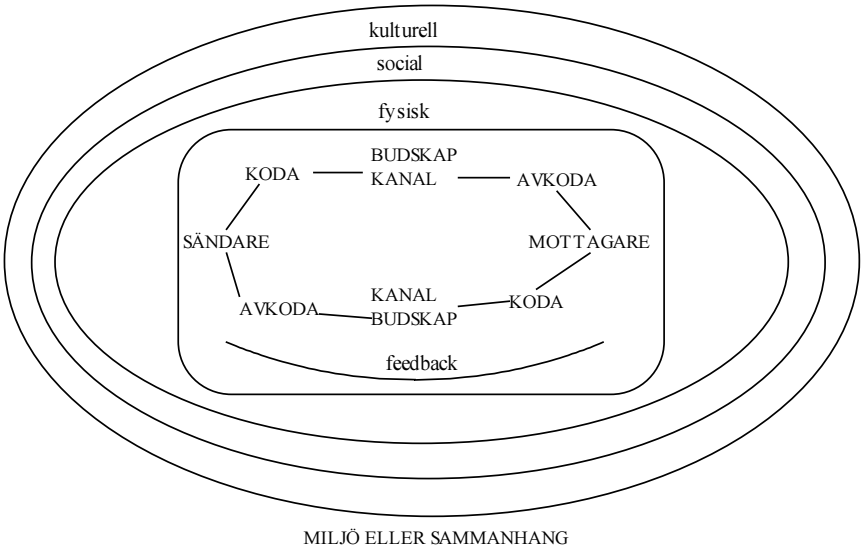
Dimbleby & Burton beskriver sedan att vi använder koder när vi kommunicerar genom tecken. Detta innebär ett system för hur vi använder tecknen. Detta system är grundat på regler och konventioner, exempelvis stavning och grammatik, som delas av kodens användare. Koder är i princip uppsättningar av tecken som kan vara primära eller sekundära. De primära är exempelvis bildform och verbal form medan de sekundära är speciella uppsättningar av tecken som fungerar inom den primära koden. Det är här kundens och utvecklarens terminologi kommer in då de sekundära ofta är relaterade till arbete eller sociala/subkulturella grupper.

Kommunikation beskrivs som ett beteende. Alltså något vi kan lära oss och bli bättre på. Att kommunicera är en inlärningsprocess som inte är oföränderlig utan hela tiden utvecklas. Det går alltså att lösa kommunikationsproblem genom att bli bättre på tekniken att kommunicera. Det handlar främst om att vi måste lära oss att lyssna aktivt och med stort tålamod försöka förstå det som vi hör. Koncentrera oss på innehållet istället för framförandet.

Dimbleby & Burton tar också upp att själva kommunikationsprocessen kan beskrivas i tre olika modeller. Vi har valt att avbilda en sammanhangsmodell, figur 4.4,

eftersom den tillfogar dimension eller omgivning. Detta innebär bl.a. olika erfarenheter och skilda antaganden.

Sammanhangsmodellen visar även feedback som en central del i kommunikationsprocessen. Dimbleby & Burton beskriver att människor anpassar sitt sätt att föra samtal efter den feedback som mottagaren ger.



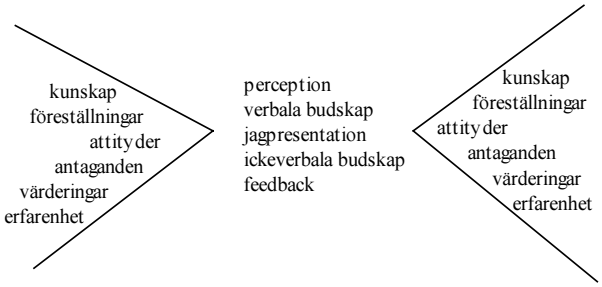
Figur 4.4. En sammanhangsmodell. Ur ”Kommunikation är mer än ord”.

Detta beskriver Dimbleby & Burton som strategi. Effektiv kommunikation är en fråga om att vara uppmärksam på den andre, att ta hänsyn till vad han behöver. Vi använder avsiktligt vissa verbala och ickeverbala tecken för att nå ett visst resultat. Det kan handla om t.ex. strategier för att övertala, gilla eller ogilla och be om ursäkt. De tar som exempel att expediter får lära sig vissa strategier för hur de ska ta människor och få dem att köpa aktuell produkt.

Vidare tar Dimbleby & Burton upp att vi presenterar oss själva för andra på olika sätt. Allt handlar om föreställningar och roller. Dessa roller påverkar relationen mellan personerna vilket också Dimbleby & Burton påpekar. Och denna relation påverkar i sin tur till stor del kommunikationen.

Det är dock skillnad på den roll en person uppfattas ha och den roll han verkligen har. Dimbleby & Burton beskriver detta som perception. Alltså hur vi uppfattar människor. Ju mer tid vi tillbringar med en person desto mer upptäcker vi hos honom vilket leder till en bättre bedömning.

Resultatet av en kommunikationsprocess beror alltså på flera olika faktorer. Dessa sammanställs i figur 4.5.



Figur 4.5. Faktorer i interpersonell kommunikation. Ur ”Kommunikation är mer än bara ord”.

De faktorer som står i vägen för en fri och fullständig kommunikation beskriver Dimbleby & Burton som barriärer. Ofta finns dessa barriärer inom oss. De indelas i tre grupper:

- Mekaniska barriärer (t.ex. störande ljud)
- Semantiska barriärer (ordens betydelse, koden)
- Psykologiska barriärer (attityder, övertygelser, värderingar)

Vi behöver erkänna dessa barriärer innan vi kan göra något åt dem.

Enligt Eysenck (1993) är förståelse av språk viktigt för att minska användningen av de kognitiva resurserna. Annars läggs resurser till att tyda vad som sägs istället för att tyda innebörden av budskapet. Användningen av formella språk vid systemutveckling gör att mycket av användarnas kognitiva resurser går åt till att hantera språkförståelsen. Det lämnar lite plats över till annat, ex. validering. Ett annat problem i anknytning till detta som Eysenck tar upp är att vi människor inte klarar flera nya saker samtidigt eftersom de tar samma resurser i anspråk. Det vi inte förstår ersätter vi med antaganden. Detta utgör en stor risk i kommunikationen och skapar grund för missförstånd.

4.3 ”Infologiska ekvationen”

Som vi tidigare har nämnt, angående vad kommunikation är, kan kommunikations-problem bero på någon form av tolkningsfel. Det handlar alltså om att ta till sig data och omvandla den till information på ett felaktigt sätt. Detta samt vilka aspekter som spelar in tar Langefors (1986) hänsyn till och pekar ut i sin infologiska ekvation.

Hur kan data tillhandahålla information? Vad krävs för att specifika data skall kunna åstadkomma information? Enligt Langefors krävs en process som tar emot data som en av dess inputs. Denna process kallar han ’informationsprocessen’ eller ’tolkningsprocessen’, *i*. Informationsanvändaren, eller processen *i*, måste sedan ha kunskap om betydelsen av de ord som används för att forma data samt de regler som används för att göra kombinationerna av ord meningsfulla. För att få informationen användbar måste användaren alltså ha någon sorts förförståelse av det språk som datan är formerad i samt vissa förkunskaper om den relevanta delen av verkligheten. Detta namnger Langefors som *S,* vilket alltså innebär förförståelse och referensramar.

Vidare hävdar Langefors att processen *i* tar tid, *t*. Om tiden (*t)* är begränsad så måste mängden information följaktligen begränsas i samma mängd och så även mängden

data. Ovanstående slutsatser utmynnar i en ’begreppsekvation’ eller ’den infologiska ekvationen’ som visas i figur 4.6.

$$I=i(D,S,t)$$

I = den åstadkomna informationen
i = informations- eller tolkningsprocessen
D = data
S = förkunskaper eller referensramar hos informationsmottagaren
t = tiden som krävs eller finns tillgänglig för processen

Figur 4.6. Den infologiska ekvationen.

Eftersom *I* egentligen är kunskap så adderas den till *S* (vilket ger *S+I*) vilket innebär att efter processen, om individen skulle ta emot samma data en gång till, får han informationen *I = i(D, S+I, t)*.

Även tiden som krävs, *t*, beror på datan som skall tolkas såväl som på förförståelsen hos mottagaren. Detta uttrycks i formeln *t = T(D,S)*. alltså tiden som krävs, beror på datan som skall tolkas såväl som på förförståelsen hos mottagaren.

Langefors poängterar att även om den infologiska ekvationen är ganska enkel och pekar på ganska självklara saker så visar den på ett antal viktiga fakta. Den ger insikten att för att utforma data så att de informerar den tänkta mottagaren på rätt sätt måste vi människor ha kunskap om förförståelsen och referensramar, d.v.s. *S*, hos mottagaren. S kan alltså t.ex. utökas till att omfatta S_b(begrepp), S_w(verklighetsbild) och S_v(värderingar).

Vidare skriver Langefors att detta antyder att det är ett mycket problematiskt förehavande att utforma data så att den informerar. Så länge det inte finns någon metod för att dokumentera *S* hos individerna är deltagande av användarna i utformningen av data nödvändig. Utformandet är också oberoende av datorteknologi.

En slutsats av den infologiska ekvationen blir enligt Langefors att allt förvärvande av kunskap, om det så är från (formell) data, från språktexter eller från observationer från verkligheten, är helt och hållet beroende av vår (individuella) förförståelse.
En annan slutsats av ekvationen är att två olika personer inte kan förväntas att tillägna sig samma information från samma data. Det bästa vi kan hoppas på är en *tillfredsställande grad* av tillägnandet av informationen. Langefors antyder att för att öka förståelsen av varandra så är det mer effektivt om vi kan använda det egna ”lokala” arbetsspråket, d.v.s. fackuttryck o. dyl.

Människor är alltså olika, beroende på skilda egenskaper, kunskaper och erfarenheter, vilket innebär *S*, och detta påverkar deras tolkningsprocess, *i*. Denna olikhet tas även upp av Magoulas & Pessi (1998) som anser att olikheterna speglas dels i våra mentala modeller (d.v.s. våra visioner om systemet – vår anmärkning) och dels i vår förmåga att ständigt utveckla dessa. De tar upp ett antal olika typer av olikheter:

- Funktionella olikheter, dvs yrkeskunskaper och kompetens. Hit hör också fysiologiska och kognitiva olikheter.

- Infologiska olikheter, dvs kulturella olikheter i form av språk, estetik, etik, värderingar och känslor.
- Strukturella olikheter, dvs olikheter i sociala intressen, makt, ansvar, resurser.
- Filosofiska (meta-arkitekturella) olikheter, dvs olikheter i perspektiv på funktionella, infologiska och strukturella olikheter.

Magoulas & Pessi menar att den infologiska ekvationen egentligen kan delas upp i två olika synsätt när det gäller systemutveckling. Det ena synsättet beskriver ekvationen ur ett ”hårt” systemtänkande som använder ett regelstyrt (R) perspektiv, alltså man har ett bestämt regelverk för hur utvecklingen skall gå till och man följer detta slaviskt. De beslut som tas med detta synsätt kan beskrivas som en process som – i enlighet med vissa *regler* – omvandlar information till handling.

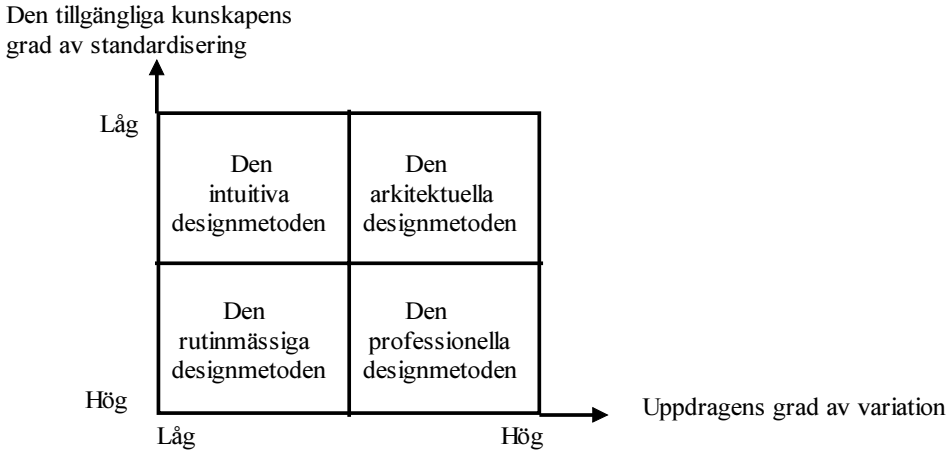
Det andra synsättet beskriver ekvationen ur ett ”mjukt” systemtänkande där utvecklingen sker ur ett motivationsbaserat (M) perspektiv. Beslut som tas kan då karakteriseras som en process som – i enlighet med vissa etablerade *mål* – omvandlar information till handling.
De olika synsättens ekvationer kan då se ut som på figur 4.7.

I=i((D,S,t),R) **I=i((D,S,t),M)**

Figur 4.7 Den infologiska ekvationen ur ett hårt resp. mjukt systemtänkande.

4.4 Design och utvecklingsfilosofier

Bergenstjerna, Johansson & Wojtasik (1999) beskriver en modell för design -och utvecklingsfilosofier som vi anser väl avspeglar de utfall som intervjuerna med systemutvecklarna och våra egna erfarenheter givit. Vi beskriver dessa utfall i kapitel 6.



Figur 4.7 Den tillgängliga kunskapens grad av standardisering i relation till uppdragens variation. Ur ”Metoder för strategisk IT-management”.

Bergenstjerna, Johansson & Wojtasik menar att vid den *rutinmässiga designmetoden* finns en hög grad av standardisering och en låg grad av variation. Detta innebär att metoden är styrande och att man följer den slaviskt. Detta är vanligt i små projekt.
Vid den *intuitiva designmetoden* finns däremot en låg grad av standardiserad kunskap och en låg grad av variation på uppdragen. Detta innebär att metodkunskapen istället för att vara dokumenterad utgörs av utvecklarnas kunskaper och erfarenheter.

Vid den *professionella designmetoden* finns både en hög grad av standardiserad kunskap och variation på uppdragen. Det innebär att metoden är både styrande och stödjande. Slutligen anger Bergenstjerna, Johansson & Wojtasik att vid den *arkitektuella designmetoden* har den tillgängliga kunskapen låg grad av standardisering medan uppdragens variation är stor. Detta erbjuder maximal balans mellan standardisering och specialisering. De menar att det krävs en mängd intuition och kreativitet för att slutföra uppgiften.

Vidare nämner de tre viktiga kunskapsmässiga grunder för att använda ovanstående metoder. Dessa är överblickbarhet, medvetenhet och meningsfullhet.

Överblickbarhet är ett av de främsta målen i systemdesign. Utan denna finns få förutsättningar för att förstå och utan att förstå finns få förutsättningar att handla på ett sunt sätt.
Medvetenheten är grunden för sunt handlande och denna grundas på direkt och ömsesidig kommunikation mellan utvecklare och kund.
Meningsfullheten innebär i detta sammanhang att varje teknisk lösning skall vara ett medel för människors strävan att uppnå avsedda mål i den organisation som systemet är tänkt att införlivas i.

4.5 Soft Systems Methodology

I vårt sökande efter litteratur om att förebrygga kommunikationsproblem inom systemutveckling är den mest väsentliga och genomtänkta infallsvinkeln vi hittat det som handlar om Soft Systems Methodology(SSM). Vi finner det därför passande att ägna en del av detta kapitel åt detta ämne.

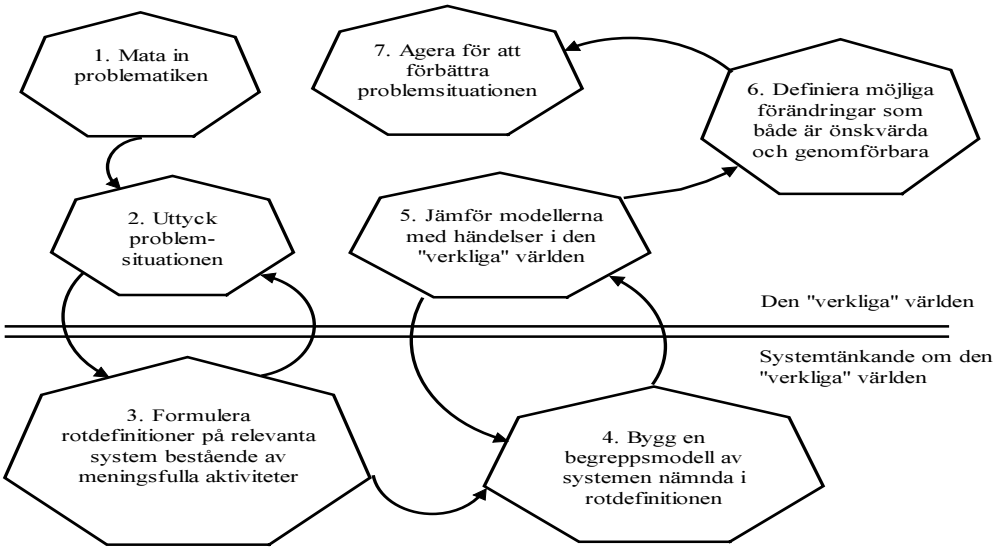
I motsats till traditionell systemkonstruktion (av Checkland (1989) kallad System Engineering) har Checkland utvecklat metoder han kallar Soft Systems Methodology (SSM).

I sin bok ger han sin syn på traditionell systemkonstruktion, där systemkonstruktion, såväl som andra ”hårda” närmanden, utgår från en relativt välstrukturerad problemsituation där man har kommit överens om vad som utgör problemet. Den utgår från att myter och betydelser är på plats och statiska. Istället kan energi läggas ner på fakta och logik.

Checkland insåg här behovet av vissa metoder. SSM är en metod som stödjer inlärning, medan systemkonstruktion mer handlar om att nå mål. Inlärandet handlar om en komplex, problematisk mänsklig situation och leder till handlingar som är meningsfulla för de som berörs. SSM innebär undersökning som leder till handling.

Olika individer gör olika bedömningar av en situation, vilket leder till olika handlingar och tolkningar.

Vi måste acceptera att det (a) finns flera möjliga beskrivningar av en verklig händelse och (b) att den beskrivning som ska användas analytiskt måste vara tydlig angående de antaganden om världen som den beskrivningen tar för givna. Detta kallar han Weltanschauung eller världsbild. Vår Weltanschauung är lagret av bilder i våra huvuden, skapade av våra ursprung, uppfostran och erfarenhet av världen som vi använder för att förstå världen och som normalt inte ifrågasätts. Systemkonstruktion ignorerar Weltanschauung men SSM gör det inte. SSM lär sig genom att jämföra modeller av aktiviteter, baserade på skilda uppfattningar av världen. Figur 4.8 visar de sju steg som SSM:s inlärningscykel går igenom.



Figur 4.8 SSM's inlärningscykel

Dahlbom & Mathiassen (1993) tar upp Checklands teori och menar att "soft systems thinkers" inser att vår värld är formad av våra erfarenheter. Vi ser olika saker, har olika perspektiv, strukturerar världen olika beroende på intressen, bakgrund, utbildning och kultur.

Enligt dem lever vi i den värld vi *uppfattar* och den världen är mer intressant än den verkliga. För systemutvecklare är det *den* som är den verkliga världen. När vi ser system i världen är de resultatet av våra försök att organisera våra upplevelser, övertygelser och visioner. Ett system baseras på antaganden om världen. Olika antaganden ger olika system. Som en konsekvens finns det alltid flera perspektiv, vilket resulterar i olika system i samma konkreta situation.

För "hard systems thinkers", finns systemen därute och de byggs, förändras och förbättras genom utveckling. Utvecklarna ser dem och tror på vad de ser. För "soft systems thinkers" finns systemen i våra huvuden, de är perspektiv som vi ändrar och förbättrar genom att konfronteras med andra perspektiv, genom att färdas runt i världen och uppleva nya saker, genom att lära oss. Dahlbom & Mathiassen menar

dock att själva inlärningen är svår, speciellt när det betyder att vi måste ändra vår uppfattning om världen, våra grundläggande antaganden om vad som är viktigt och inte. När vi konfronteras med folk med radikalt olika perspektiv på världen, är vår naturliga reaktion oförståelse. Vi förstår helt enkelt inte vad de säger. Det som för oss är terrorism, kan för andra vara en kamp för frihet.

För att kunna arbeta enligt ”soft system” är det enligt dem nödvändigt att utveckla en metodik som ska hjälpa oss förstå perspektiv som skiljer sig från våra. Metodiken hos ”soft system” är tolkning. Vi uppmuntras, genom den här metoden, att ta hänsyn till olika perspektiv; målet är att för att lära sig om världen måste vi förstå, uttrycka och debattera en variation av radikalt olika perspektiv.

De skriver vidare att för att förbättra sättet som arbetet organiseras på i en systemutvecklingsgrupp kan vi inte bara förlita oss på det abstrakta system som uttrycks i den standardprojektmodell som används av gruppen. För det första borde vi jämföra detta system med övertygelserna och attityderna hos projektmedlemmarna och lära oss från skillnaderna mellan den ideala världen i projektmodellen och erfarenheterna och idéerna i projekten. Men viktigast av allt borde vara att formulera alternativa system, uttrycka perspektiven hos de involverade aktörerna vad gäller ämnen såsom projektorganisation, kommunikation och samarbete, programmering m.m.

En annan anmärkning, menar Dahlbom & Mathiassen, är att vi har en tendens att vara blinda för olikheter i perspektiv. Vi går omkring och tar för givet att folk ser på världen som vi gör. Vi vet givetvis att det finns ”konstiga främlingar”, men det kommer oftast som en överraskning att de kan befinna sig på vår egen arbetsplats.
Synsättet om ”soft system” påminner oss om att vi är olika och att olika perspektiv gör skillnad. Det erbjuder en strategi för att uttrycka sådana olika perspektiv och för att engagera folk i en debatt med syftet att nå någon slags överenskommelse som kan leda till formuleringen av krav för handling.

Dahlbom & Mathiassen hänvisar återigen till Checkland (1989) och nämner CATWOE, som en metod för att få fram de rotdefinitioner som nämns i figur 4.9.

C Customer	Vem har nytta av/”drabbas av” den meningsfulla aktiviteten?
A Actors	Vem skall utföra aktiviteten?
T Transformation-process	Hur uttrycks aktiviteten som input respektive output?
W Weltanschauung	Vilken världssyn gör denna definition meningsfull?
O Owners	Vem kan stoppa aktiviteten?
E Environmental constraints	Vilka begränsningar i dess miljö tar detta system för givet?

Figur 4.9 CATWOE

De menar dock att både ”hard” och ”soft systems” tänkande delar en underbart naiv syn på världen som grundläggande harmonisk. Antingen är världen själv i ordning och stabil eller så finns det flera olika perspektiv på världen, men de kan alla sammanföras i en rationell dialog, med ett ömsesidigt intresse av att nå den slutgiltiga

syntesen av överenskommelse, sanning och förståelse. De skriver att vår värld dock inte ser ut så. Den är istället full av kaos, konflikter och mörka, irrationella krafter, en farlig och skrämmande djungel av olika intressen och maktkamper.

Avslutningsvis förklarar Dahlbom & Mathiassen att för att kunna hantera en komplex miljö, förlitar sig organismer på en strategi som bara observerar det som är nytt eller annorlunda och tar resten för givet. Människor är inget undantag. Det som är för familjärt blir praktiskt taget osynligt. Vi ser det inte förrän det är borta. Det är lättare att studera en främmande kultur eller organisation än sin egen.

Lewis (1994) skriver att en viktig aspekt av ”soft system”-tänkande är att det tillåter att en situation kan anses vara annorlunda av betraktare med olika personliga övertygelser och värden. Situationens betydelse beror alltså på värderingarna, övertygelserna och tidigare historia hos betraktaren. ”Hard system”-tänkande har inga metoder för att ta hand om sådana olikheter i uppfattningar av situationen, men ”soft system” tänkande tar dessa, och deras orsaker, som en viktig komponent i problemsituationen.

Här går att se, menar han, att medan utvecklarens rekommendationer kan vara både möjliga och rationella i deras egna termer, har de väldigt lite mening för andra. Såvida inte verklighetsdeltagarna sympatiserar med de synsätt och värderingar på vilka analysen har baserats, är det föga troligt att de kommer att hålla med om utvecklarens syn på vad som är ”problemet”, eller betrakta hans rekommendationer som önskvärda eller ens relevanta för deras syften.

Systemutvecklaren måste förstå organisationen som systemet ska stödja och förstå samt bidra till debatten om vad organisationen behöver eller inte.

4.6 Prototyping

Sommerville (1996) skriver att vid utveckling av system är det ofta svårt för kunderna att specificera vad det är som de egentligen behöver i ett system. Som hjälp kan då *prototyper* användas.

Flynn & Warhurst (1994) beskriver prototypen som en förenklad version av systemet där inte allt fungerar men det ger kunden en bild av vilka funktioner som kommer att finnas tillgängliga och hur systemet kommer att se ut när det är färdigt. Det validerar krav samtidigt som nya ospecificerade krav på systemet kommer fram.

En prototyp ska, enligt Sommerville (1996), vara en grundform, en mycket elementär beskrivning av ett system, där vissa funktioner medvetet väljs bort för att förenkla och göra systemet mer konkret. Det går att hindra många dyrbara misstag, som ofta sker då människor med olika uppfattningar och erfarenheter möts, genom att redan tidigt i utvecklingsprocessen ta fram en prototyp.

Sommerville visar på några andra av fördelarna med prototyping:

1. Missförstånd mellan systemutvecklare och kund uppdagas genom att utvecklarna konkret kan visa hur de har uppfattat att funktioner skall fungera och användarna kan korrigera dessa.

2. Kunden får möjlighet att kontrollera att alla nödvändiga funktioner finns.
3. Systemets användbarhet kan testas och modifieras.
4. Systemutvecklarna får en möjlighet att stämma av krav på systemet och lägga till, modifiera eller ta bort krav.
5. Med utgångspunkt från en prototyp kan utvecklarna skapa en specifikation.

Dessa punkter innebär även ekonomiska fördelar.

Vidare tar Sommerville upp tre olika sorters prototyping. Den första är *evolutionary prototyping* och innebär att kunden får ett ofullständigt system som utvecklas och förfinas i många steg allteftersom kundens krav blir tydliga. Detta görs tills ett fullständigt system har utvecklats som kunden är nöjd med. Denna sorts prototyping lämpar sig för system som är svåra att specificera. Prototypen bygger på att de krav från användaren som är lätta att förstå införlivas först och sedan går man vidare till de krav som är mer svårförståeliga.

Den andra sorten kallas *throw away-prototyping* och dess funktion är att konkretisera krav på systemet. Prototypen utvecklas från ett systems specifikationsutkast. Parterna experimenterar med den och modifierar den tills den stämmer överens med kundens funktionskrav. Efter utvärdering kastas prototypen bort. Specifikationen som framtagits med hjälp av prototypen används för att göra färdigt den slutgiltiga mjukvaruprodukten.

Den tredje och sista är *incremental development* vilket innebär att krav framarbetas och systemet levereras i små steg. Detta gör det möjligt för kunden att testa det levererade systemet och lägga fram synpunkter så att senare levererade delar kan påverkas. Varje steg planeras och dokumenteras, vilket gör det lätt att i utveckling av nya delar i dokumentationen hitta de krav som ställs på dem utifrån äldre delar.

Pressman (1997) skriver att ett problem med prototyping är att kunden måste vara tillgänglig under hela processen för att kunna utvärdera prototypen kontinuerligt så att inga avbrott i processen uppkommer till följd av detta. Vonk (1990) menar att denna kundmedverkan kräver mycket av kundens tid. Han skriver att kostnaden för prototyping beror på antalet försök som behövs för att nå en av användarna accepterad prototyp. Antalet försök är i sin tur relaterad till osäkerhetsgraden i kravbilden vid början av projektet. Vonk påpekar dock att det faktum att kunden medverkar också innebär att han/hon har möjlighet att bättre utvärdera gränssnittet vilket är en fördel.

En fara med prototyping, som både Martin (1991) och Pressman (1997) tar upp är att kundernas förväntningar kan bli för höga. De förstår inte att det endast är en prototyp som utvärderas och att det mesta av arbetet återstår för att få ett stabilt system, utan tror ofta att systemet skall vara levereras omedelbart eftersom prototypen fungerar som det framtida systemet.

Martin (1991) menar vidare att prototyping stimulerar kunderna att ständigt förändra sina krav samt finna nya så att prototypen tar mycket lång tid att slutföra. Han tar dock upp att ett stort plus med prototyping är att det är ett redskap för utbildning. De kunder som är med under utvecklingsarbetet får samtidigt färdigheter på det framtida systemet innan det är levererat.

Flynn & Warhurst (1994) tar upp ett alternativ till prototyping: *brainstorming*. Detta innebär ett möte där deltagarna ohämmat ”slänger ur sig” förslag utan att fundera över lämpligheten eller användbarheten av förslaget. Alla förslag antecknas. Detta arbetssätt skapar en massa förslag som sen kan gås igenom och förkastas eller behållas allt efter användbarhet. Författarna anser dock att kraven lätt blir motsägelsefulla och dessutom är det lätt att missa detaljerade krav när dessa två metoder används. De tar också upp att om systemutvecklaren ska förmedla kravspecifikationen till användaren så blir kunskapsklyftan svår att överbrygga även där. Det kan exempelvis vara svårt för användarna att förstå de grafer och formella språk som används för att beskriva kraven. Det är ju helt klart svårt att bekräfta om något stämmer med det önskade om man inte förstår kraven.

Systemutvecklaren måste komma runt detta problem och enligt Flynn & Warhurst är det vanligaste sättet att exemplifiera kravspeficikationen. Systemutvecklaren skriver då *scenarier* (skeenden som uppstår i användarens arbetssituation) på de krav som har tagits fram. Då blir det lättare för användarna att förstå innebörden av de grafer och diagram som finns i specifikationen. Ett problem, enligt Langefors (1995) är dock att ett scenario kan uppfattas på olika sätt av olika användare. Samma ord har olika innebörd i olika situationer.

5 Empirisk del

I teoriernas värld anses alltså kommunikationssvårigheter vara ett stort och allvarligt problem, men ändå verkar det inte finnas några direkta metoder för att lösa dem.
Hur går det då till ute i verkligheten? Det är ju trots allt en relativt liten andel av alla systemutvecklingsprojekt som anses lyckade fullt ut, så hur upplever egentligen branschfolk kommunikationsproblemen och hur gör de för att undvika dem ute i arbetslivet?

För att ta reda på detta förflyttade vi oss från böckerna ut i verkligheten för att tala med folk inom branschen. Vi ansåg att det i vårt fall var mest väsentligt att få fram kvalitativa svar och valde därför att göra semistrukturerade intervjuer istället för en enkät. Dessa intervjuer utgör första delen av empirikapitlet. I den andra och sista delen har vi beskrivit våra egna erfarenheter ifrån det projekt som var upprinnelsen till den här uppsatsen. Där har vi valt att hålla företaget anonymt med hjälp av ett fingerat namn för att undvika att företaget tar illa upp över att bli exemplifierat som källa till kommunikationsproblem. Detta gäller även den andra kundintervjun som skett med vår egen kund.

5.1 Intervjuer

Intervjuerna har skett dels med branschfolk inom IT-sektorn, så kallade utvecklare, dels med de personer på företag som kommer i kontakt med utvecklare, så kallade kunder, då ett behov av en systembeställning uppkommer. De personer vi valt att intervjua är alla utom två oberoende av varandra.

Intervjuerna nedtecknades för hand under intervjuns gång och på plats i respondentens miljö i alla fall utom ett, den första utvecklarintervjun, då vi befann oss i universitets lokaler.

Vi intervjuade utvecklarna **Johan Öst**, f d anställd på *Digitron*, **Glenn Geidemar** och **Fredrik Magnusson** från *SKF Dataservice AB* samt **Henrik Wassenius** på *Ericsson Microwave AB*. Vi gjorde även intervjuer med kunderna **Jan Johansson** f d anställd på *SKF Vehicle Parts AB* samt ”**Arne”** (fingerat namn) på läkemedelsföretaget ”*HÄLSA* AB”(fingerat namn).

För att få en bredare bild och inte begränsas av vår egna upplevelser, som säkerligen berodde mycket på oerfarenhet, valde vi att intervjua andra systemutvecklare. Genom att se hur de personer vi intervjuade hanterar kommunikationen med kunden hoppades vi få exempel på lösningar eller tillvägagångssätt för att underlätta kommunikationen.
Vi valde medvetet till största delen relativt stora företag med många års erfarenhet i branschen eftersom vi ansåg att chansen var större att där träffa på någon form av utarbetade metoder för att undvika kommunikationsproblem.

Vi valde att intervjua kunder för att om möjligt få bekräftat att de på sätt och vis lever i en annan värld än systemutvecklarna. Vad det gäller de specifika kunderna försökte vi hitta motparter till utvecklarna för att få även deras bild av situationen. Vi lyckades dock, av sekretess- och geografiska skäl, endast i ett av fallen där Jan Johansson var kund till Digitron. Den andra kundintervjun gjorde vi med vår egen kund ”Arne” från HÄLSA.

5.1.1 Syfte med intervjuerna

De frågor vi ställde ligger i bilaga 9.1. Kundfrågorna och utvecklarfrågorna hade ungefär samma syften men ställdes på lite olika sätt då kunden och utvecklaren befann sig på olika sidor om problemet. Frågorna är indelade i grupper beroende på syfte.

Den första gruppen syftade till att ge information om respondentens bakgrund, hans typ av företag samt hans arbetssituation. Detta för att det till stor del påverkade hur resten av intervjun utvecklade sig. Därefter ville vi ta reda på hur själva mötet med utvecklaren, respektive kunden, under systemutvecklingsarbetet praktiskt hanterades.
Den tredje gruppen frågor syftade därefter till att verifiera att missförstånd verkligen uppstod i denna situation. Nästa grupp avsåg att fånga upp exempel på vanliga typer av kommunikationsproblem och därpå få fram i vilken grad de förekommer.

Vi kom sedan in på den första delen av själva kärnan i intervjun där vi försökte fånga upp anledningar till kommunikationsproblemen. Den sista, och sjunde, gruppen av frågor syftade till att få fram eventuella lösningar för att reda ut kommunikationsproblemen.
Allt eftersom intervjuerna framskred kom det dock ibland upp nya, intressanta frågor eller frågetecken som vi då följde upp och införlivade i intervjuerna.

5.1.2 Utfall av intervjuer med utvecklare

A. Intervju med Systemutvecklare 1

Namn: Johan Öst
Yrke: F d systemutvecklare på Digitron AB, idag anställd på Cap Gemini
Bakgrund: Systemvetarprogrammet samt fyraårig teknisk linje på gymnasiet.
År inom branschen: Fem och ett halvt

Om företaget: Digitron är ett av flera underbolag i logistikkoncernen Swisslog. Digitron har hand om utvecklingen av mjukvaran, samt försäljning och projektledning rörande denna, medan de andra underbolagen sysslar med olika former av ”hårdvara”, t.ex. truckar, banor och kranar. Den mjukvara som behövs är främst transportstyrsystem men kan även vara administrativa system, lagersystem eller orderhanteringssystem. Systemen styr utrustningen och håller koll på var i logistikkedjan produkterna befinner sig.

Respondentens arbetsuppgifter: Systemutvecklingen på Digitron består av olika steg där steg 1 främst utförs av ett säljteam och innebär att man specificerar funktioner på grov nivå. Johan arbetar med programmering, systemering och design och kommer in i steg 2 då man mer i detalj beskriver vad man menar med olika saker och hur de ska fungera och se ut. Han berättar att han som programmerare, tillsammans med säljare och projektledare, möter kundens projektteam ett antal gånger där de diskuterar igenom de olika delarna av vad man beställt och ”stångas lite grand”. Ett utkast skrivs som sedan omarbetas efter hand och så småningom ska godkännas av kunden genom påskrift.

Johan berättar att Digitron egentligen inte har några direkta rutiner för hur kundmötet ska gå till men påpekar att det är viktigt att veta SHARK:s (systemet som ligger till grund för de system kunderna beställer) möjligheter och begränsningar så att man inte lovar saker som inte går att bygga till. Det är viktigt att kunna uppskatta vad ändringar och tillägg innebär vad det gäller extra tid och kostnader eftersom systemen levereras till fast pris. Johan poängterar att det är viktigt att vara detaljerad i kravspecifikationen för att undvika problem och missförstånd. Ofta rör det sig om 60-70 sidor med designspecar, testspecar och manualer. 10-15% av tiden för systemutvecklingen brukar gå till själva programmeringen. Resten går åt till möten, dokumentation och testning.

Vilken roll kunden spelar i systemutvecklingsarbetet beror mycket på hur mycket kunden kan. Johan berättar att de kunder som inte är så kunniga inom området kan få aha-upplevelser i slutfasen. Ibland blir då systemet inte riktigt som de tänkt sig även om de skrivit under kravspecifikationen. Kunden har helt enkelt trott att det som han skriver under beskriver något annat än vad systemutvecklarna tolkar det som.
Systemutvecklaren får försöka förklara vad som ingår så gott det går. Förr eller senare trillar polletten ner, tyvärr ibland försent. Det underlättar då att det inte finns så stor variation och svängrum i de system som Johan utvecklar. Det är värre med administrativa system. Det finns även kunder som vet precis vad de vill ha och Johan menar att han föredrar den sorten.

Johan berättar att han brukar ta grafer till hjälp för att få kunden att förstå. Grova och förenklade flödesdiagram kan hjälpa kunden att se flödet i systemet. Om kunden gör si händer så… Det är viktigt att vara tydlig och göra allt så enkelt som möjligt. Johan tar också nytta av de erfarenheter han får i varje projekt. Vidare säger han att de brukar göra prototyper på operatörsgränssnittet så att man har något konkret att diskutera runtomkring. Kunderna ser, även om det bara är på ytan, hur det är tänkt och får lite känsla för vad systemet ska göra. Johan tycker att prototyping är en bra metod. Det tar inte särskilt långt tid att göra. Prototyperna ritas upp utan kod med hjälp av 4-GL-verktyg som ex. Uniface. Därefter är det bara att fylla på skalet.

Johan anser att kommunikationen med kunden för det mesta går ganska bra, men visst förekommer missförstånd och problem om än inte i direkt stor utsträckning. Dock har det hänt att Digitron gått back på projekt. I stort sett alla projekt innehåller mer eller mindre allvarliga missförstånd. Han minns ett projekt med ett danskt företag. Det största problemet här var det faktum att kunden pratade danska och utvecklaren svenska. Parterna förstod inte varandra lika bra som de trodde och ibland fick de ta till engelska.

Annars handlar de flesta kommunikationssvårigheter om att systemet inte är tillräckligt specificerat. Johan beskriver att han som utvecklare gör sin tolkning och kunden sin om det är otydligt från början. Då är det förhandlingar och kompromisser som gäller angående vilka funktioner som ingår. Han säger att de ibland ger med sig och bjuder på omarbetningar för att visa goodwill och göra kunden glad, vilket dock kan bli resurs- och tidsmässigt kostsamt. Det är därför viktigt att vara tydlig i ett så tidigt stadium som möjligt. Ibland kommer kunden kanske med något galet förslag

som inte går att förverkliga men då lyckas oftast systemutvecklarna övertyga dem om andra idéer.

Ett annat problem kan vara att kunden fokuserar på ”fel” del av systemutvecklingen. Johan berättar att i projektet han jobbar i just nu fokuserar de på själva formalian och hur saker och ting uttrycks istället för innehållet och det kan bli jobbigt för projektledaren.

Johan tror också att en annan orsak till problemen kan vara att systemutvecklarna har sitt språk och kunderna har sitt verksamhetsspråk. De använder olika begrepp för saker och ting. Kunden kan inte heller så mycket om systemutvecklarnas saker. Det beror på om kunden har liknande automatiserade system innan. Då hänger det på oss att förklara, berättar Johan.

Johan påpekar återigen att det är viktigt att redan på ett tidigt stadium vara tydlig för att undvika missförstånd. Man måste noggrant specificera vad som ska ingå. Det underlättar även för den händelse att någon i projektteamet skulle hoppa av. Då blir det lättare för den nye att sätta sig in i projektet.
En ordentlig förstudie är också viktigt. Man ska inte börja ”hacka kod” för tidigt eftersom kunden oftast inte riktigt vet vad han vill ha.

B. Intervju med Systemutvecklare 2

Namn: Henrik Wassenius
Yrke: Systemutvecklare på Ericsson Microwave AB
Bakgrund: Fyra års studier i datateknik på Chalmers
År inom branschen: Fyra år

Om företaget: Ericsson Microwave är ett litet företag i ett stort, globalt företag. Microwave-delen inom Ericsson-koncernen bygger radarsystem. Deras främsta kund är den svenska försvarsmakten och beställningarna kommer från Försvarets MaterielVerk (FMV). Produkterna, alltså radarsystemen, är sk inbyggda datasystem, men kunden använder dem som en sorts informationssystem.

Respondentens arbetsuppgifter: Henriks arbetsuppgifter består av att sitta som teknisk delprojektledare, där den främsta arbetsuppgiften är systemanalys. Kundkontakten består av att han sitter med vid kundmöten när det gäller tekniska frågor, alltså inte när det gäller t. ex. kontraktsförhandling. Även användare av det blivande systemet sitter ofta med vid dessa möten. Väldigt förenklat går utvecklingsprocessen till som så att Ericsson Microwave får en beställning från försvaret och sedan bygger de systemet. All utveckling är väldigt teknisk, så systemet som byggs är extremt kundspecifikt.

Hur ett möte mellan utvecklare och kund praktiskt hanteras beror mycket på hur projektet är uppbyggt, hur ansvaret är fördelat o.s.v., berättar Henrik. Som det ser ut nu så tar de bägge parterna kontakt med varandra efter behov och träffas då det behövs. Kunderna är aktiva med tekniskt kunnig personal som har åsikter och synpunkter på det mesta. Dialogen mellan utvecklare och kund fungerar oftast mycket bra, båda är intresserade av att få fram en bra produkt. Det sätts ofta ett fast

pris på produkten. Ibland uppkommer givetvis diskussioner om saker man ej är överens om, men detta löser sig oftast på ett smidigt sätt.

I utvecklingsarbetet kommer Henrik ofta i kontakt med olika språkterminologier. Kunder från den marina delen av försvarsmakten använder ett visst ”språk”, kunder från luftvärnsdelen ett annat, och användare av systemen ett tredje o.s.v. I vissa projekt har de för att undvika missförstånd i kommunikationen med varandra utvecklat en gemensam terminologi. Detta händer om projekten är tidsmässigt väldigt långa och då skaffar man fram denna redan från början. I ett projekt som Henrik arbetar med nu har t.ex. arbetet med att ta fram gemensam terminologi pågått hela vintern. En viktig faktor här är tiden. Är det gott om tid så läggs mer kraft ner på att få fram en gemensam terminologi, men med mindre tid till förfogande så blir de oftast tvungen att hoppa över denna del.

Ett annan metod som Ericsson Microwave använder sig av för att undvika missförstånd i kommunikation är att göra en ordentlig verksamhetsanalys av kundens verksamhet. Här stöter de då på olika termer som måste klargöras och förklaras så att de kommer överens om innebörden.

Henriks uppfattning om kommunikationen mellan utvecklare och kund är att den fungerar bra. Det är lätt att göra sig förstådd. Detta kan bero på att det oftast sitter tekniker på båda sidor av förhandlingsbordet och de har i regel nästan samma bakgrund vilket underlättar.

Ett vanligt problem som dock kan uppstå, främst kanske av användarkaraktär men ibland även av kundkaraktär, är att användare ofta refererar till sin verklighet som den ser ut idag. De har svårt att förstå hur systemet *kommer* att se ut och vad som rent tekniskt går att genomföra.

Graden av missförstånd varierar mycket, men när missförstånd uppkommer så kostar det både tid och pengar. Upptäcks det försent, ex. när produkten är klar, får de börja förhandla igen. Henrik tror att missförstånd säkerligen finns i alla projekt men det mesta löser sig alltid genom givande och tagande från båda sidor.

Eftersom Ericsson Microwave bara har svenska kunder så finns inga rent språkliga barriärer mellan utvecklare och kund. Missförstånd kan istället bero på hur mycket som läggs in i olika begrepp, hur ambitionsnivån ser ut på exempelvis en funktion. Kunden kan ibland tro att systemutvecklaren gör saker och ting på ett mer komplicerat sätt än nödvändigt.

Ytterligare en orsak till missförstånd är att då projekten är långa så hinner personal sluta och ny personal skall introduceras i projektet. Den nya personalen kanske då tolkar saker och ting på ett annat sätt om inte specifikationen är tydlig nog.
Vid grafiska presentationer kan det uppstå en del diskussioner men det handlar mer om att folk uttrycker olika synpunkter än om missförstånd.

Henrik anser att det absolut viktigaste är att dokumentera allt man kommer fram till. Alltså inte bara säga ”-Vi gör så här…” utan verkligen föra protokoll på allt och detta protokoll skall båda parter vara överens om. En väsentlig del är också att våga fråga

om något är oklart. Då gäller det även att tänka på att ställa frågan så att motparten inte missförstår den.

När det gäller terminologi så upplever Henrik det som att Ericsson Microwave har i stort sett samma terminologi som luftvärnet medan behovet av att ta fram gemensam terminologi är större när det gäller samarbetet med marinen.
Det används inte så mycket grafer och bilder för att skapa förståelse under utvecklingen av inbyggda system. Istället används förklaringar i textform.

Oftast har de på Ericsson en ganska klar bild över vad kunden vill ha. Har de inte det så gör man en grundläggande förstudie/analys som klargör vad kunden är ute efter.
Ericsson Microwave har en enhet på 15 personer som enbart sysslar med förstudier.
Metoden med prototyping som används mycket inom övrig systemutveckling börjar komma mer och mer även inom försvarsindustrin. Man bygger en demovariant som sedan byggs på allteftersom.

C. Intervju med Systemutvecklarna 3/4

Namn: Glenn Geidemar
Yrke: Systemutvecklare på SKF Dataservice AB
Bakgrund: Systemvetarlinjen med magisterexamen
År inom branschen: Ett års traineeutbildning samt två års anställning på SKF

Namn: Fredrik Magnusson
Yrke: Systemutvecklare på SKF Dataservice AB
Bakgrund: ADB-linjen med kandidatexamen
År inom branschen: Ett års traineeutbildning samt två års anställning på SKF

Om företaget: SKF Dataservice är en IT-avdelning inom SKF-koncernen. Alla uppdrag som avdelningen tar på sig kommer från SKF. Avdelningen är tänkt att vara icke vinstdrivande men den avsikten är på väg att luckras upp. Nya möjligheter börjar skönjas. Med samarbete mellan olika företag kommer gemensamma program mer och mer att utvecklas.

Respondenternas arbetsuppgifter: Fredrik Magnusson arbetar som IT-koordinator, vilket i praktiken innebär att han är ”hjälpreda” åt ”IT-strategikoordinationsgruppen” inom SKF. Arbetsuppgifterna innebär att när ett utvecklingsprojekt skall genomföras försöker Fredrik styra in det till SKF Dataservice och försöker hitta olika lösningar.
Glenn Geidemar jobbar som ”alltiallo” inom utveckling. Mestadels med analys och design men ibland även som projektledare.

I mötena mellan SKF Dataservice och deras kunder ingår mycket telefonkontakt då kunderna ofta är geografiskt spridda. Videokonferenser och netmeetings ingår också då det finns behov av att förklara saker och ting visuellt.

För att göra sig förstådda för kunderna så försöker både Glenn och Fredrik att sätta sig in i kundens affärsprocess *innan* själva ”IT-arbetet” startar. Detta givetvis om tid finnes vilket inte alltid är fallet. Något som Glenn och Fredrik upptäckt är att det är

mycket viktigt för båda parter är att träffas och umgås. Kommunikationen fungerar mycket bättre efter det.

Kommunikationen mellan utvecklare och kund fungerar bra på det hela taget. Glenn gjorde en gång ett försök att med UML, Unified Modelling Language [2] visa en kund hur hans idé om systemet såg ut. Kunden förstod inte mycket av det och blev ombedd att rita om det så att han själv förstod. Glenn hade då inga svårigheter att förstå kundens modeller. UML kan kanske vara ett bra redskap för båda att använda, dock under förutsättning att båda lär sig det ordentligt. Annars verkar det vara svårt att hitta ett gemensamt språk.

Ibland är det svårt för kunden att förklara vad han vill ha ut av systemet eller kanske vissa funktioner. En anledning, tycker både Glenn och Fredrik, är att de som utvecklare har för liten kännedom om kundens organisation. Vanliga ”käppar i hjulet” är också språksvårigheter då utvecklarna och kunderna under möten ofta talar engelska men ingen av parterna har det som modersmål. Detta försvåras även av att kunskaperna i engelska hos vissa länder är mindre bra, ex. i Tyskland, Frankrike och Italien. Utöver språksvårigheter mellan nationaliteter så förekommer även språksvårigheter inom svenska språket. Ex. ordet ’komponent’ betyder en sak för utvecklarna på SKF Dataservice och en helt annan sak för deras kund. Detta ställer till det ibland.

Ytterligare svårigheter som upplevts är kulturella skillnader, stereotypa könsroller, annat chefsklimat o.s.v. Även rädsla hos kunderna för att ta beslut utan att ha sanktionerat det med sin chef upplever de som en bromskloss ibland.
För många deltagare i ett projekt kan också skapa förvirring. Frågor som ”är det rätt person som tagit ett visst beslut”, ”den här rapporten saknas, vem skulle gjort den” o.s.v. kan vara svårare att bena ut om det är många projektmedlemmar
Ett annat problem som börjat synas i takt med att det blir lättare och lättare för ”Svensson” att utveckla egna program är att kunden försöker leka systemutvecklare själv och knåpar ihop program med egna lösningar (oftast utan struktur) utan att ha tillräckliga kunskaper. Detta skapar system som är väldigt svåra att underhålla och som sedermera blir dyra att byta ut.

Tidsbristen är en annan faktor som kan öka risken för missförstånd. I och med att kunderna ofta är spridda över världen i ett globalt företag som SKF så är det svårt att få alla inblandade på samma plats samtidigt.

Fredrik och Glenn anser att missförstånd förekommer i så gott som alla utvecklingsprojekt, kanske mer i början än i slutet. Det är en lång sträcka av osäkerhet i början. Parterna pratar om samma sak men uttrycker sig luddigt. Det tar tid innan de förstår att de talar om samma sak. Ibland vet inte kunden vad han vill ha förrän han ser det. Eller tvärtom, när han ser det vet han att han inte vill ha det.
Det ultimata scenariot hade varit om en tjock lunta med precisa specificerade funktioner hade kunnat tas fram i samförstånd mellan kund och SKF Dataservice. Först då skulle man påbörja själva ”IT-arbetet”. Så är dock sällan fallet. Istället blir det mycket givande och tagande och kompromisser från båda sidor. Ofta får SKF

[2] ett grafiskt språk för att visualisera, specificera, konstruera samt dokumentera ett system, t.ex. beskriva klasser, funktioner o.dyl.

Dataservice agera ”mäklare” mellan olika krav och mellan olika verksamheter. En annan roll de får ikläda sig är ”polisens”, exempelvis om kunden har ett krav på att systemet skall vara anpassat för NT-miljö och det inte går. Då gäller det att styra undan kunden till den miljö som går att använda. Det går inte att säga ”du får inte…” och sedan nöja sig med det utan man måste komma med konstruktiva alternativa lösningar för att inte stöta sig med kunden. Resonerar man ordentligt med kunden så slutar det oftast med en vettig och godtagbar lösning.

En lösning på problemet med att få alla inblandade på plats samtidigt kan vara att istället för att ha själva *kunden* på plats för diskussion så kan man ta in nån som kan det som kunden *kan*, t. ex en konsult av något slag.

5.1.3 Utfall av intervjuer med kunder

D. Intervju med Kund 1

Namn: Jan Johansson
Yrke: F d produktionschef på SKF Vehicle Parts AB
Bakgrund: Civilingenjör på Chalmers samt fysik på Göteborgs Universitet
År inom branschen: Tjugonio år på SKF

Om företaget: SKF Vehicle Parts AB är ett delföretag inom SKF-koncernen.

Respondentens arbetsuppgifter: Produktionschef

I samband med lokalbyte av verksamheten som Jan var chef för så bestämdes att SKF Vehicle Parts skulle arbeta fram ett nytt datasystem. Detta skulle styra de olika maskinerna som hade hand om plockning och förpackning av produkterna som avdelningen fått beställningar på. Det nya systemet skulle vara Windows-baserat.
Jan tog kontakt med försäljarna på systemutvecklingsföretaget Swisslog AB som skulle utveckla systemet genom sitt dotterbolag Digitron AB. Eftersom SKF Vehicle Parts hade ett gammalt system att jobba efter innan och de visste ganska precis vad de ville ha så gick arbetet med systemutvecklarna på Digitron väldigt smidigt. Jan hade dessutom förberett en grov specifikation på vad han ville att systemet skulle göra så det var bara att jobba vidare på den tills en fullständig kravspecifikation som båda parter var nöjda med hade tagits fram.

I mötena mellan SKF Vehicle Parts och Digitron upplevde Jan det som att systemutvecklarna agerade bollplank, att det var ett givande och tagande på lika villkor. Ingen kände sig överkörd av den andra parten under utvecklingsarbetet utan det fanns hela tiden en dialog som fungerade bra. Utvecklarna förklarade sig bra med hjälp av bilder på gränssnittet i systemet samt med prototyper. Trots att de från SKF Vehicle Parts sida från början av utvecklingsarbetet trott sig ha en klar bild över alla funktioner de ville ha i systemet, så tillkom några till efter att bilder och prototyper visats upp för dem. Tio månader efter att arbetet hade börjat var systemet klart och all utrustning stod på plats.

Jan upplevde det som att utvecklarna på Digitron snabbt lyckades sätta sig in i SKF Vehicle Parts’ organisation. De var tillsammans ute och tittade på andra anläggningar

för att få idéer till upplägget. Fem olika grupper skapades där det sinsemellan diskuterades funktionalitet på det blivande systemet, och där försäljarna av systemet ingick. Till slut skapades en känsla av att även utvecklarna och försäljarna ingick i deras organisation.

Kommunikationen mellan parterna upplevde Jan som sagt som mycket bra och graden av missförstånd var mycket liten. Om det var någon detalj han inte kunde förklara som utvecklarna undrade över så var det bara att ta in folk från verkstaden eller kontoret som hade bättre koll på just den delen av det blivande systemet. Vidare var båda parter på samma nivå, ingen satte sig över den andra. De hade helt enkelt samma roller med ett gemensamt mål: att skapa ett bra system. Även de anställda hade stor del i systemet. Både praktiskt och pedagogiskt eftersom de då fick känslan av ”vårt system” och blev otåliga på att få prova det.

En liten småsak som dock störde var när de bytte projektledare vilket gjorde att en del detaljer glömdes bort, men det var inget stort problem utan ordnades till snabbt. Jan berättar att han hade för vana att alltid föra dagbok över alla möten och dylikt där han satt med. Detta för att få med alla detaljer och sedan kunna gå tillbaka och se exakt vad som sagts. Han fyllde ca 10 st kollegieblock med anteckningar under den tid projektet fortlöpte.

E. Intervju med ”Arne”

Namn: ”Arne” (fingerat namn)
Yrke: VD för ”HÄLSA AB”(fingerat namn)
Bakgrund: Fil kand inom biokemi och företagsekonomi
År inom branschen: Har jobbat som marknadsförare och säljare inom läkemedelsbranschen i 20 år.

Om företaget: HÄLSA är ett marknadsförings- och försäljningsbolag som sysslar med konsultationer, försäljning, marknadsanalyser m.m. inom hälso- och sjukvårdsbranschen.

Respondentens arbetsuppgifter: Eftersom företaget är så litet gör Arne det mesta.

Arne kom i kontakt med systemutvecklare då hans företag beställde en webplats för produktförsäljning samt kundportal. Eftersom Arne inte hade någon tidigare erfarenhet av systemutveckling hade han inga speciella rutiner för att praktiskt hantera ett möte med en systemutvecklare. Han upplevde det som om hans ovana vid möten med systemutvecklare även gjorde att han och hans kollega inte hade några förutfattade meningar eller uppfattningar om hur saker och ting skulle lösas. De visste bara vad de ville ha efter att ha tittat runt lite på andra sidor och skapat en egen blandning, men inte hur de skulle nå dit. Han påpekar att det säkert innebar både för- och nackdelar för systemutvecklarna. Det handlade om ett givande och tagande där man försöker jämka.

Eftersom HÄLSA arbetar på ett lite annorlunda sätt ansåg Arne att det var viktigt att systemutvecklarna förstod organisationen för att förstå hur systemet skulle se ut. Han

visade därför en powerpoint-presentation av företaget. Han upplever det lättare än att förklara organisationen i bara ord.

Ett problem i kommunikationen med systemutvecklare som Arne märkt av är att de använder annorlunda terminologi. Både vad det gäller att få systemutvecklare att förstå sjukvårdsbranschen med dess ”fikonspråk” och när de ska förstå hur han vill att systemet ska uppföra sig. Han beskriver det som att var och en lever i sin egen lilla värld och tror att alla andra lever i samma. I många fall kände han att han kanske inte använde rätt ”datorspråk” för att systemutvecklarna skulle förstå en viss funktion. Han tolkade det ändå som att de ganska fort fick klart för varandra vad de ville göra.

Det var även svårt för honom att förstå den terminologi som systemutvecklarna använde. Han menar att han inte har så djup kunskap om datorer men intresset finns där. Han förstår dock att systemutvecklarna måste prata lite tekniska metoder sinsemellan för att komma fram till hur saker och ting går att lösa. Det är ju systemutvecklarna som är specialisterna och vet bäst vad det är för skillnad på ena och andra lösningen. Arne berättade vidare att om han inte har någon insikt i problemet tror han att det är lättare än vad det är. Det är svårt att känna till alla de mellansteg som krävs. Som kund måste man därför vara beredd att tänka om ifall något inte går att göra eller visar sig vara väldigt dyrt att realisera.

Språksvårigheterna var dock inget oväntat. Det får man räkna med när folk inom olika yrken möts, anser Arne. Han förväntade sig inte att systemutvecklarna skulle förstå fullt ut vad han sa eller att han skulle begripa exakt vad de sa. Om mottagaren inte begriper har sändaren av budskapet helt enkelt varit för oklar. Ibland tar det sin lilla tid. Han ser dock ingen mening i att ge sig in i tekniska diskussioner angående program ditt eller datt utan utgår ifrån att systemutvecklarna bäst avgör vad som ska användas. Graden av svårigheter var alltså ganska liten.

Förutom att språket är annorlunda har kunden och utvecklaren även olika erfarenheter och bakgrund. Arne ser dock det hela positivt att han, genom att möta människor ifrån andra yrkesgrupper, får möjlighet att lära sig många nya saker. Det är viktigt att vara öppen till sinnet. Oftast är det outsagda det väsentliga.

Han tycker att modeller är ett bra sätt att visa saker och ting på. Med visualiseringar och flödesscheman är det lättare att begripa hur informationen ska gå och det blir lättare att hänga med.

5.2 Egna erfarenheter

Som vi skrev i introduktionen så har vi skaffat oss en del egna erfarenheter utanför skolans regi vad gäller systemutveckling, och att vi tyckte att dessa var ett lämpligt ämne att skriva uppsats om. Vi tänkte behandla några av dessa erfarenheter i denna del av den empiriska undersökningen som vi helt enkelt döpt till ”Egna erfarenheter”.

Vi blev alltså kontaktade av företaget, ovan döpt till HÄLSA AB, som ville ha ett system byggt på deras befintliga hemsida på Internet. HÄLSA hade gjort en egen marknadsundersökning bland sina kunder och fått fram att intresset var stort för deras idé som vi beskriver nedan under rubrik 5.2.3.

5.2.1 Vilka är vi?

För eventuella läsare av denna uppsats kan det vara av intresse att veta den bakgrund och utbildning vi hade då vi tog oss an det här projektet. Projektmedlemmarna är tre till antalet:
Niklas Borg – 30 år, f d banktjänsteman, hittills 70 poäng inom Informatik.
Theresia Höglund – 21 år, ettårig utbildning inom Media och Kommunikation, hittills 70 poäng inom Informatik.
Peter Akcan – 20 år, hittills 70 poäng inom Informatik.

5.2.2 HÄLSA:s organisation

HÄLSA är, som tidigare beskrivits, ett marknadsförings- och försäljningsbolag som sysslar med konsultationer, försäljning, marknadsanalyser m.m. inom hälso- och sjukvårdsbranschen. Företaget består idag av två personer men en del av försäljningen utförs av inhyrda konsulter. Syftet med systemet var alltså att slippa behöva anlita en del av dessa konsulter. På sin befintliga hemsida på Internet ger HÄLSA idag även ut ett nyhetsbrev till de som besöker sidan. Utöver detta finns information om de produkter de saluför.

5.2.3 Systemet

HÄLSAs målsättning med systemet var att för kunden ge översikt, kontroll, kontakt, snabbhet i beställningar, svar på terapifrågor, trygghet gällande tillgång och priser samt dessutom skapa en positiv bild av HÄLSA.

Systemet skulle dels tillhandahålla en ”butiksdel”, d.v.s. en möjlighet för företagets kunder att beställa en del av deras produkter via Internet. Efter beställning av produkter skulle en sida med historik skapas där kunden kunde se sina tidigare beställningar och även kunna ändra på sina personliga uppgifter, såsom adress och telefonnummer, valda intresseområden o.s.v.

Utöver detta skulle det finnas tillgång till en lösenordsskyddad ”egen sida” för kunden, en sk portal. Här skulle kunden själv kunna styra vad för slags information från HÄLSA AB denne ville ha tillgänglig på sidan eftersom det rörde sig om så vitt skilda intresseområden. Meddelanden om exempelvis erbjudanden som företaget ville ha ut till sina kunder skulle kategoriseras och endast nå de kunder som valt en viss kategori.

5.2.4 Möten med HÄLSA AB

Vi har hittills haft 4-5 möten med HÄLSA. Vid första mötet förklarade de hur de tänkt sig att deras system skulle se ut. I första hand var de intresserade att få butiksdelen byggd och den skulle vara klar på ca två månader. Tiden var därmed ganska knapp.

När HÄLSA började prata om sina visioner om systemet så började vi omedvetet ganska snart att i huvudet ”översätta” visionerna till kod och tekniska lösningar (ex. knappar, boxar, listor m.m.). En annan detalj var att redan vid första mötet började båda parter prata fackspråk med varandra som motparten inte förstod, och både vi och de fick ”översätta” till vanlig svenska

Vi har efter första mötet i februari träffats ca en gång per månad i syfte att komma överens om vad systemet skall innehålla och kunna få fram en kravspecifikation.
Det gick dock ganska lång tid mellan första och andra mötet och vi blev osäkra på om vi över huvudtaget skulle få projektet.

Efter det andra mötet blev det muntligen klart att systemet skulle byggas av oss och sedan dess började vi ha ganska täta kontakter via e-mail. Nästan uteslutande handlade e-mailen om vilka funktioner som skulle ingå i systemet. Under denna period uppstod nästan fler frågor för varje svar vi fick, och vi upplevde det även som att de frågor vi ställde inte var de vi fick svar på. En fråga som vi uppfattade som en enkel ja- eller nej-fråga kunde resultera i ett långt svar, samtidigt som en uttömmande fråga eller en flervalsfråga kunde besvaras med ett ja eller nej.

Vi började väl så smått inse att vi borde träffas ansikte mot ansikte betydligt oftare än vi hittills gjort, men vi ansåg då att tidsbristen gjorde att det gick snabbare att kommunicera på detta sätt. Samtidigt får vi nog erkänna att vi var ”rädda” för att ytterligare möten skulle skapa än fler missförstånd och oenigheter om systemets innehåll, vilket väl får anses vara ett kardinalfel av oss.

Vi kände oss även lite ”dumma” när vi tyckte att vi måste fråga samma fråga om igen när vi fått svar på något annat än det vi frågat efter, och det skapade även en viss irritation hos oss när vi inte var nöjda med svaret. Här lyste vår orutin från den verkliga världens systemutveckling igenom. Vi hade inte räknat med att det skulle vara så svårt att förstå varandra.

Ganska snart upptäckte vi dock ett hjälpmedel som både vi och HÄLSA kunde använda för att lättare förstå varandra: prototyper. Vi byggde ganska snabbt ihop en prototyp som visade hur vi hade uppfattat hur de ville att systemet skulle fungera. Detta blev mycket uppskattat och efter denna första prototyp gjordes ändringar allteftersom oklarheter i kommunikationen oss emellan klarades upp.

När det gäller portaldelen verkade det som om kunden hade god kännedom om hur han ville att det skulle se ut, han hade en vision. Vi frågade dock inte om han sett något liknande någonstans vilket säkerligen kunde ha hjälpt oss. Kunden verkade ha relativt god kännedom om tekniken, Internet o. s. v. men det var svårt för oss att uppskatta exakt hur mycket/lite.

Avslutningsvis kan väl sägas om mötena med HÄLSA att vi upplevde att vi inte hade tillräcklig kunskap om hur man går tillväga vid ett kundmöte, hur man börjar, vad man ska skapa för ”dokument”, systemutvecklingens faser o. s. v. De objektmodeller, klasskort m.m. som vi initialt använde för att försöka beskriva delar av systemet var alldeles för abstrakta för kunden och sade ingenting om funktionerna i systemet.

5.2.5 Kravspecifikationen/kontraktet

Största problemet för oss var egentligen att systemet bestod av två delar, butiksdelen och portaldelen. Förvirring rådde angående hur mycket av de båda delarna av systemet som skulle ”byggas ihop” eller om de skulle vara två fristående system.
Eftersom vi uppfattat att endast butiksdelen skulle byggas av oss hade vi satt oss för lite in i hur portaldelen skulle se ut.

När det gällde arbetet med att ta fram en kravspecifikationen som båda parter kunde acceptera så hade vi alltså här ett ganska stort missförstånd på gång redan från början. Vi hade uppfattat att de ville ha en offert och en kravspecifikation på endast butiksdelen och att portaldelen skulle vara en senare och separat del av systemet som *eventuellt* skulle byggas. Så var inte fallet utan HÄLSA ville ha en offert på *hela* systemet. När vi hade fått iväg denna vår första offert var det förvånade ögonbryn och en del muttrande innan vi fick rett ut detta lilla missöde.

En nackdel med att behöva jobba så mycket med kravspecifikationen och koda våra prototyper var att vi kände oss osäkra på om vi över huvudtaget skulle få betalt. Det var ju så mycket jobb som skulle göras innan vi kunde skriva kontrakt och vi hade definitivt ingen lust att jobba gratis.

Till slut fick vi ihop en slutgiltig offert och en kravspecifikation som båda kunde godkänna och kontraktet skulle slutligen skrivas under. Tyvärr fick vi då nya direktiv från HÄLSA. Butiksdelen skulle skjutas på till framtiden och portaldelen hade största prioritet. Samtidigt sköts tidsplanen framåt, sjösättning av *hela* systemet skulle bli i augusti. Nu fick vi lägga butiksdelen åt sidan som vi ändå hunnit ganska långt med och börja från noll med portaldelen. Eftersom vi är oerfarna vid dessa snabba omkastningar i tidsplaneringen skapade detta en del irritation hos oss. Vi hade väl förväntat oss att det skulle gå till ungefär som i skolans trygga värld där man hade en klar uppgift och ett fast datum för när projektet skulle vara klart. Så kan man lugnt säga att det *inte* går till ute i verkligheten.

Efter den sista omarbetningen av tidsplanen i kravspecifikationen kunde kontraktet slutligen undertecknas. Vi tycker att vi i kravspecifikation givit en ganska bra beskrivning av hur systemet ska fungera. Givetvis är det en ingalunda komplett kravspecifikation p. g. a tidsfaktorn (återigen) och det kan säkert bli problem när kunden ska validera den mot systemet, men för att göra den komplett hade vi behövt arbeta med den så länge att tidpunkten för när systemet skulle installeras skulle ha passerats.

När det gäller förhandling av den ekonomiska biten så kände vi oss väldigt vilsna. Det var extremt svårt att försöka uppskatta hur lång tid hela projektet skulle ta, och därmed sätta ett pris på systemet. Ingen av oss hade någon större erfarenhet av att sitta i ekonomiska förhandlingar av den här typen och det kändes jobbigt att sitta och diskutera pengar med kunden. Vårt utgångsbud accepterades i alla fall utan invändningar vilket väl tyvärr innebär att det var alldeles för lågt.

5.2.6 Funktionella önskemål

Vissa önskemål om funktioner i systemet som vi inte var överens om, och därför blev föremål för diskussion och missförstånd, tänkte vi redovisa här.

HÄLSA uttryckte vid ett tillfälle önskemål om att kunna skicka e-mail till ”kundens sida”, alltså inte till någon e-mailadress. Detta stötte på en del problem eftersom vi inte hade någon aning om hur man skulle kunna lösa det rent tekniskt. Till slut enades vi om att det inte var fråga om e-mail utan enkla textmeddelanden om exempelvis

erbjudanden och rabatter som skulle synas på respektive kunds portalsida, vilket var betydligt smidigare att lösa.

Svårigheter som uppstod i portaldelen var de olika intresseområden m.m. som HÄLSA ville att deras kunder skulle ha att välja mellan.
Det fanns från början 4 st huvudområden, vi kan kalla dem A, B, C och D. Under dessa huvudområden skulle man kunna välja att få information om mer detaljerade delområden, 3-4 st under varje huvudområde. Dessa kan vi kalla x, y, z och å. Slutligen skulle man kunna välja vilket av följande informationsslag man ville få av dessa underområden: information, aktiviteter, nyheter och övrigt.

Man skulle alltså bli tvungen att först välja huvudområde A, därefter något av delområdena och slutligen vilken typ av information man ville ha skickad till ”sin sida”. Vi insåg ganska snabbt att om en person satt och läste igenom alla valmöjligheter skulle denne ganska snart tröttna på dessa så vi fick HÄLSA att begränsa valmöjligheterna ganska drastiskt.

HÄLSA hade även stora visioner om att kunna anordna videokonferenser, direkt telefonkontakt m.m. via portaldelen i framtiden. Detta ansåg vi oss inte kompetenta att utföra och tid till att utbilda sig fanns ju inte heller så vi fick göra klart redan från början att det inte kunde ingå i kravspecifikationen.

5.2.7 Övrigt

Som avslutning på kapitlet om våra egna erfarenheter kan vi bara påpeka att själva systemutvecklingen legat i träda ett tag p. g. a. denna uppsats men kommer att återupptas inom kort och förhoppningsvis har vi fått alla delar på plats i augusti när det är dags för installation av systemet som helhet. Då lär vi också se om vi har lyckats ta till oss de erfarenheter som vi skrivit om i uppsatsen.

6 Resultat, analys och diskussion

6.1 Analys och diskussion av teoridelen

Under denna rubrik analyserar och diskuterar vi en del av litteraturen vi gått igenom och tar fasta på punkter som vi tycker är väsentliga och extra viktiga. Vi argumenterar också emot där vi tycker att det inte stämmer med våra erfarenheter.

Vi reagerade starkt på Enquists (1999) undersökning som visade att utvecklarna inte reagerade på kända missförstånd eftersom det inte ansågs vara viktigt. Att man som oerfaren, nyutexaminerad systemutvecklare kan brista på den här punkten må kanske vara förlåtet, men när det rör sig om rutinerade utvecklare som rimligtvis har lärt sig vikten av att vara överens med kunden så borde det finnas ett ansvar hos dem att reagera på all form av missförstånd. Ett litet missförstånd i början på utvecklingsprocessen kan leda till ett stort fel i den färdiga programvaran som kan bli dyrt att reparera.

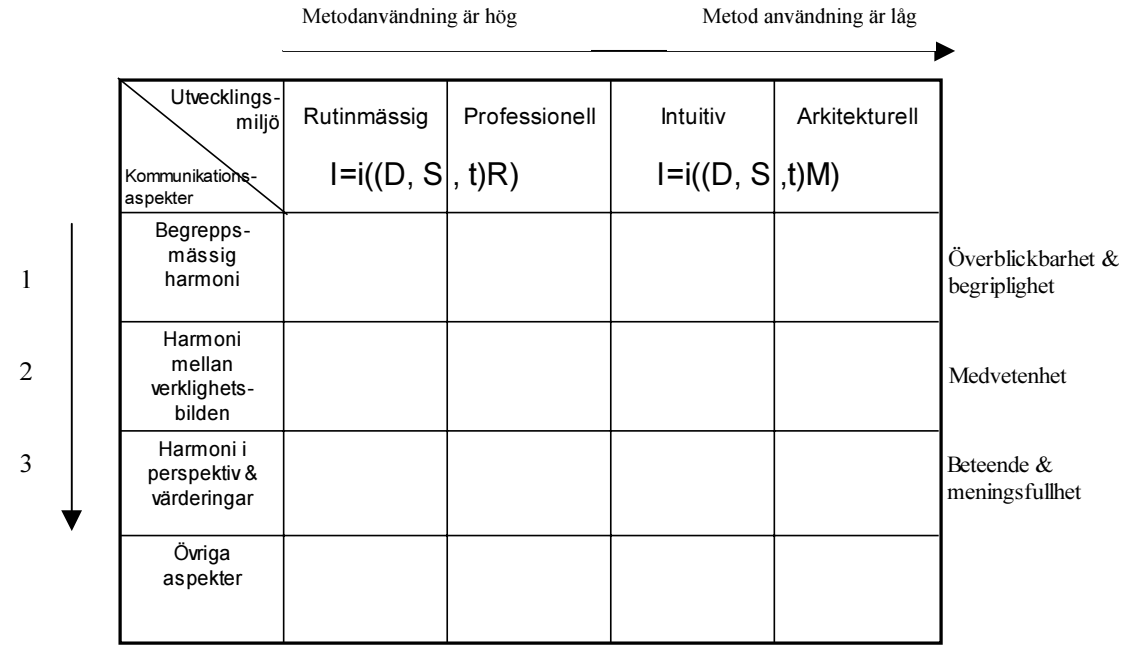
Vi gillar Checklands beskrivning av SSM-metoden där man måste inse att det finns flera möjliga beskrivningar av en verklig händelse, vilket beror på den världsbild eller Weltanschauung vi har. Han nämner i sin bok exemplet att Nicaraguas regering kallade landets gerilla i början på 1980-talet för terrorister medan USA kallade dem för frihetskämpar.

Vid studerandet av Langefors (1986) *infologiska ekvation* slog det oss att tiden, *t* i ekvationen, är en oerhört väsentlig del av problemet med missförstånd i kommunikationen. Pengar är sällan något problem när det gäller systemutveckling men tid verkar vara en faktor som det ständigt saknas tillräcklig mängd av. Om tidsmängden i systemutvecklings-processen kunde ökas skulle antagligen förståelsen mellan utvecklare och beställare öka väsentligt.
Vidare tycker vi att en av slutsatserna man kan dra av den infologiska ekvationen som rör vår uppsats är insikten att för att utforma data så att de informerar de tänkta användarna på rätt sätt så måste man ha kunskap om *S*, alltså förförståelse och referensramar hos kunden.

När vi tittade på Bergenstjerna, Johansson & Wojtasiks modell av olika designfilosofier tyckte vi att det gick att sammanföra deras modell med Langefors infologiska ekvation. De talade om *rutinmässig* (hög grad av standardiserad kunskap, låg grad av variation på uppdrag), *professionell* (hög grad av standardisering, hög variation), *intuitiv*(låg grad av standardisering, låg grad av variation) samt *arkitekturell* (låg grad av standardisering, hög variation) *design*.
Beroende på vilken utvecklingsmiljö systemutvecklaren befinner sig i finns antingen metodstöd i form av regler (*R* i infologiska ekvationen) eller inte. Finns inga regler får man förlita sig på intuition och känsla (*M* i infologiska ekvationen). Den översta pilen i figur 6.1 visar olika grader av stöd i form av metodanvändning med högsta graden till vänster och lägsta graden till höger. Har man väldigt lite stöd av metoder behövs, enligt den infologiska ekvationen, ytterligare data (*D*) och tid (*t*).

För att kunna hantera systemutvecklingsuppdraget och ta till sig en gemensam verklighetsbild (kunskaper, handlingsstilar, tolkningsstilar, kommunikationsstilar)

måste man först skaffa sig en gemensam begreppsbas (lokalt språk, globalt språk) och för att kunna få samordnade perspektiv och värderingar (motivation, intressen) krävs gemensam verklighetsbild. Alltså måste stegen tas i tur och ordning från 1 till 3 enligt figur 6.1.



Figur 6.1 Teoretisk karta över designfilosofier och den infologiska ekvationen

När begreppsmässig samordning nås leder det till överblickbarhet och begriplighet. När försök görs att harmonisera verklighetsbilden kan vi med hjälp av begreppen öka medvetenheten. När steg 1 och 2 är överstökade har förhoppningsvis insikt nåtts om varandras olikheter och det blir lättare att ändra sina värderingar och anpassa dem efter motpartens. Genom att ge och ta på ett meningsfullt sätt kan man nå en ”win-win”-situation för båda parter.

När Boland (1979) skriver att beslutsfattare inte vet vad de vill ha och det de vill ha inte nödvändigtvis är det de behöver så hittar vi här ett mycket starkt argument för systemutvecklaren att verkligen sätta sig in i kundens organisation. Det kan inte nog understrykas hur viktigt detta är.

I Dahlbom & Mathiassen (1993) påpekas att det som är familjärt praktiskt taget blir osynligt. Det tror vi stämmer väldigt väl i utvecklingsfasen av ett system. Om kunder sitter med och skall diskutera exempelvis detaljer i funktioner som är självklara för dem så är det lätt hänt att de glömmer att nämna dem för utvecklaren. Detta skapar givetvis problem i ett senare skede av utvecklingsprocessen.

Dahlbom & Mathiassen skriver också att världen vi erfar är den verkliga världen och att den är mer intressant än den verkliga världen. Precis denna punkt pekar ”Arne” från ”HÄLSA AB” på då han säger att missförstånd kan bero på att alla går omkring ”i sin egen lilla värld” i tron att alla andra lever i samma.

När det gäller metoden prototyping för att öka förståelse i kommunikationen så har vi ju själva använt oss av den metoden i vårt utvecklingsprojekt och vi finner den vara

det oslagbart bästa sättet att göra sig förstådd för kunden. Martin (1996) nämner i och för sig en fara med prototyping då kunden kan ledas att tro att systemet snart kan vara klart eftersom han ser gränssnittet. Vi har dock själva inte upplevt detta som något problem p. g. a att vi hela tiden varit noga med att påpeka att det är just bara en prototyp som inte klarar av några avancerade funktioner. Kunden har inte haft några svårigheter att förstå detta så är man bara tydlig nog så behöver det inte innebära något problem.

Flynn & Warhurst (1994) föreslår brainstorming som ett bra sätt att kommunicera. Det är nog inte fel, men återigen krävs det en del tid i anspråk för att reda ut varenda förslag som föreslås. Enligt brainstorming-metoden får ju inga idéer förkastas utan allt skall antecknas, men som sagt, om tid finnes anser vi detta vara en mycket bra metod för att undvika att vare sig utvecklare eller kunder skall känna sig rädda för att komma med förslag som kanske inte är riktigt genomtänkta. Gör man klart från början att allt som sägs är av vikt så vågar man nog slänga ur sig fler förslag.

Flynn & Warhurst andra förslag är scenarier och vi insåg då att det hade varit en bra metod att tillämpa i mötet med HÄLSA. Vi tror att kunden då lättare kan urskilja om alla hans önskemål angående systemfunktioner kommit med. Han får också, redan i ett tidigt skede av systemutvecklingsprojektet, en klarare bild över hur det färdiga systemet är tänkt att fungera och kan då bedöma om systemutvecklaren förstått honom korrekt. Vi tror att scenarier kan vara ett bra alternativ (eller komplement) till prototyper om tiden för prototypskapande är knapp.

Eklund & Fernlund (1998) anser att kravspecifikationen är ett kontrakt där hela åtagandet gentemot *kunden* fastställs. Detta är vi ej överens om då vi anser att ett kontrakt alltid innebär två parter som avser att uppfylla ett åtagande gentemot *varandra.* Även kundens skyldigheter gentemot *utvecklaren* skall ju noteras här när det gäller ex. betalning m.m.

Ur Dimbleby & Burtons (1999) texter har vi nedan försökt ”översätta” bokens innehåll till att specifikt handla om kommunikation i systemutvecklingsprocessen och de liknelser vi kan göra med ”vanlig” kommunikation.

De påpekar bl. a. att kommunikation sägs ha ett behov och ett syfte.
Med detta i åtanke så kan man utläsa att om utvecklaren och kunden verkligen är medvetna om varför de kommunicerar och att de arbetar mot ett gemensamt mål så kommer de att ha lättare för att samarbeta och därmed kommunicera bättre.
De måste ta reda på åt vilket håll det är meningen att kommunikation ska gå, ex. om det är kunden som ska berätta sin vision av systemet eller utvecklaren sin. Vi anser att det kan vara av stor vikt att ta till sig att kommunikationsprocessen även har ett resultat, att den leder till någonting. Om detta resultat sedan blir som man tänkt sig beror på om budskapet gått fram på rätt sätt eller inte. En utvärdering av detta bör vara på sin plats.

Vidare tror vi liksom Dimbleby & Burton att var man fysiskt befinner sig under systemutvecklingsprocessen kan påverka resultatet av kommunikationen. Om man befinner sig på utvecklarens eller kundens hemmaplan och hur rummet ser ut kan påverka vem det är som blir den ”ledande” i samtalet, vem som vågar ställa frågor och

erkänna att han/hon inte förstår etc. Vi tror att det kan vara lättare att ”hamna på lika nivå” och bli mer familjär med varandra om man befinner sig i en mindre, och för båda parter neutral lokal.

Hur utvecklaren och kunden tolkar varandras budskap anser vi delvis ha att göra med om de kommunicerar med ”öppna sinnen” eller om de redan har en idé om hur systemet ska se ut. I sådana fall är det lätt hänt att man tar det personligt om ens idé ratas, och det kan därför vara viktigt att påpeka att hela kommunikationen rör själva systemet och inte personerna som deltar i utvecklingsprocessen. Här spelar givetvis även tidigare erfarenheter och bakgrund hos båda parter in.

Vad gäller själva kommunikationen mellan kunden och utvecklaren menar vi att man bör tänka på om det man vill ha sagt verkligen framgår tydligt. Kunden ska inte tvingas ”läsa mellan raderna” för att uppfatta hela budskapet från utvecklaren. Det är också viktigt att de frågor som uppstår blir öppet ställda utan tanke på om frågan är pinsam eller har ställts förut. Kunden kan ju endast svara på det som han/hon hör. Om båda är så tydliga och öppna som möjligt är det självklart en förutsättning att mottagaren inte tolkar in för mycket och letar efter dolda budskap, för då uppstår ju likväl missförstånd.

När det gäller responsen av ett budskap från ex. kund till utvecklare, så kan man tänka sig ett fall där kunden har ett önskemål om en viss funktion som han vill att systemet ska tillhandahålla. För utvecklaren betyder kanske budskapet extra jobb och ny problematisk teknik och responsen kan då bli påverkad av detta.

Missförstånd sker när ett visst tecken inte har någon betydelse knuten till sig hos personen som tar emot tecknet, eller om hans betydelse är olik tecknets betydelse hos den som skickade budskapet. Det är alltså detta som händer när utvecklaren och kunden använder ord ifrån sin egen terminologi.
Av detta utläser vi att det inte är någon mening att kommunicera om man inte förstår den andres tecken, alltså är det meningslöst både för utvecklaren och kunden att använda fackuttryck från sina respektive världar om inte dessa förstås av mottagaren.
Detta låter måhända väldigt elementärt men det händer ändock alltför ofta.

Vi anser därför att om utvecklaren eller kunden inte förstår eller inte håller med motparten så måste de visa detta. Det vill säga att mottagaren av budskapet ger rätt typ av feedback så att sändaren kan reagera på detta. Det innebär givetvis också att sändaren måste vara observant på hur mottagaren reagerar och vid behov kan ändra sin kommunikation så att budskapet bättre går fram och syftet lättare uppnås.

Dimbleby & Burton skriver om olika roller i kommunikationen och då anser vi det värt att notera att utvecklaren i relationen med kunden visserligen främst ska inta rollen av utvecklare, men även tänka på att han/hon ska vara till tjänst för kunden.
Med det menar vi att vara både professionell utvecklare och förstående människa,
och att utvecklaren måste tänka på att spela en sådan roll så att kunden får en slags känsla av trygghet och inte känner sig dum.
Detta kan dock visa sig vara något av en balansgång; att både visa att man behärskar det man håller på med så att kunden får förtroende för att projektet kommer att utföras

på ett riktigt sätt, samtidigt som man inte ska visa sig alltför teknologiskt proffsig så att kunden känner sig bortkommen.

Kommunikation, säger Dimbleby & Burton slutligen, är ett beteende som kan förbättras.

Detta anser vi styrka att frågeställningen i vår uppsats, d. v. s hur man ska lösa kommunikationsproblemen mellan kund och utvecklare, är möjlig.

6.2 Analys och diskussion av empirin

Vi har valt att nedan kalla våra respondenter för Systemutvecklare 1, Kund 1 o.s.v. för att lättare klargöra när det rör sig om kommentarer från kund respektive utvecklare.

Systemutvecklare 1: *Johan Öst, Digitron AB*
Systemutvecklare 2: *Henrik Wassenius, Ericsson Microwave AB*
Systemutvecklare 3/4: *Glenn Geidemar och Fredrik Magnusson, SKF Dataservice AB*
Kund 1: *Jan Johansson, SKF Vehicle Parts AB*
Kund 2: *”Arne”, ”HÄLSA AB”*

Vårt möte med utvecklare och kunder ute i verkligheten resulterade i att vi återigen fick bekräftat att kommunikationsproblem verkligen förekommer. Alla fyra systemutvecklarna vi intervjuade hade upplevt någon form av kommunikations-problem. Ändå tyckte de alla att dialogen oftast fungerar bra. Även Kund 2 upplevde att kommunikationsproblem endast förekom i liten grad och inga av dem var något oväntat. Till vår förvåning upplevde Kund 1 knappt några problem alls utan ansåg att kommunikationen fungerat mycket bra.

Systemutvecklare 3/4 påpekade att missförstånd oftare förekommer i början av projektet än i slutet. Ibland pratar kunden och systemutvecklaren t.o.m. om samma sak men det tar tid innan de inser det. Detta nämndes även av Systemutvecklare 1 som beskrev det som att icke kunniga kunder oftast får någon form av aha-upplevelser i slutet. Alla fyra poängterade vikten av att vara tydlig och detaljerad. Detta gäller både i kravspecifikationen och i dialogen med kunden. Systemutvecklare 1 ansåg att det är just otydlighet som är största källan till kommunikationsproblem. Otydlighet gör det inte bara svårt för kunden att förstå utan innebär också svårigheter vid eventuellt byte av personal inom utvecklingsföretaget. Under vårt samarbete med HÄLSA var det alltså inte så konstigt att vi upplevde relativt stor grad av kommunikationsproblem. Till en början var det oklart om vi skulle göra projektet överhuvudtaget och därefter i vilken ordning som de olika delarna skulle vara klara.

Vi tror att det ultimata vore att kunden är så pass insatt och medveten om vad han vill ha att han själv kan förbereda en grov kravspecifikation inför mötet med systemutvecklarna. Detta gjorde Kund 1 och det gav resultat.
Med vissa faktorer bestämda redan från början blir mängden dialog och resurser som krävs för att komma överens och få klarhet i systemets funktioner mindre. Av detta utläser vi att om mängden dialog som *krävs* blir mindre minskar också risken för kommunikationsproblem, eftersom de tillfällen då man kan missförstå varandra reduceras. Vi vill dock påpeka att detta absolut inte är samma sak som att mängden dialog som *används* minskar, då risken finns att effekten istället kan bli den motsatta.

Med andra ord, den mängd dialog som används måste självfallet vara lika stor eller större än den mängd som krävs. Det kan låta som en självklarhet men trots det så har många projekt inte fler möten även om det egentligen vore nödvändigt. Vårt eget projekt med HÄLSA är ett exempel på detta. Vi insåg redan då att vi borde träffas oftare, men valde ändå, av flera anledningar, att inte göra det. En annan nackdel med att träffas för sällan anser vi vara att en del tankar och idéer glöms bort. Vi hann också arbeta för mycket med kodningen vilket gjorde att det var svårare för oss att ta till oss nya synpunkter och idéer eftersom de, i viss fall, skulle spoliera det arbete vi lagt ner.

Istället för att träffas utnyttjade vi e-mail. Visserligen är det ett bra och snabbt sätt att utbyta information men vi märkte att ”dialogen” ändå inte blir tillräckligt omedelbar samt att responsen inte var tillräckligt klar. Det fanns inte utrymme till diskussion och upprepning på samma sätt som under ett samtal. Eventuella reaktioner, exempelvis oförståelse, genom kroppsspråk föll också bort.

Systemutvecklare 2 tyckte att det absolut viktigaste inom systemutveckling är att dokumentera allt man kommer fram till. Detta visade även intervjun med Kund 1. Hans goda vana att noggrant anteckna under alla möten lönade sig. Detta tycker vi är något som är bra att ta efter. I hans projekt var graden av kommunikationsproblem så gott som obefintlig. Vi anser att det även spelade stor roll att Kund 1 hade en bakgrund som ingenjör och programmerare. Vi drar slutsatsen att det helt enkelt är lättare att möta människor med snarlik bakgrund och då förhoppningsvis använder liknande terminologi. Det påpekade även Systemutvecklare 2 som oftast möter tekniskt kunniga personer med nästan samma bakgrund som honom själv.

Detta tolkar vi som att kunskap och insikt i kundens respektive systemutvecklarens världar är till stor hjälp. Okunskap är alltså en faktor som kan skapa dålig kommunikation, vilket Systemutvecklare 1 och 3/4 också nämnde. Att i projektet ta in människor som har koll på rätt del av verksamheten, kan i viss mån lindra detta problem, sade Kund 1. Genom att få med olika sorters människor i projektet skapar man också en känsla av ”vårt system”.

Något som kan hänvisas till okunskapen, eller kanske rättare sagt omedvetenheten om varandras arbetsområden, är hur mycket kunden respektive systemutvecklaren lägger in i olika begrepp, påpekade Systemutvecklare 2. Han hade även upplevt att kunden och utvecklaren hade olika ambitionsnivå på funktioner. Systemutvecklare 3/4 påpekar vikten av att, på ett konstruktivt sätt, komma med ett alternativ om kunden kommer med en ”dålig” eller omöjlig lösning på någon funktion. Med detta i åtanke anser vi att kunden, och givetvis även systemutvecklaren, måste vara öppen till sinnet och vara beredd att tänka om. Detta påpekar även Kund 2. Detta anknyter till hur vi, när HÄLSA började prata om sina visioner om systemet, omedvetet ”översatte” visionerna till kod och tekniska lösningar. Denna ”översättning” gjorde vi nog alldeles för tidigt. Vi borde nog istället försökt lyssna med öppet sinne och smält intrycken och informationen innan vi började fundera på lösningar. Annars är det lätt hänt att man missar väsentlig information.

Kunden och systemutvecklaren är ofta dåligt insatta i varandras organisationer. Det kan i viss mån avhjälpas genom att göra en ordentlig förstudie, vilket var något som alla fyra utvecklarna tog upp. Detta givetvis om tid finnes vilket inte alltid är fallet.

Under förstudien kan man även stöta på termer som måste förklaras sade Systemutvecklare 2.

Just språksvårigheter är ett annat problem som de alla stöter på. Vare sig det gäller vanliga ”traditionella” språk som exempelvis svenska/engelska, eller fackuttryck och organisationsterminologi. Graden av språksvårighet beror helt enkelt på vilken typ av kund man träffar. Systemutvecklare 2:s alla kunder befinner sig i Sverige så där var problemet med fackuttryck mest markant, medan 1:an och 2:an har lite problem med båda varianterna. Systemutvecklare 2 har för vana att, i mån av tid, utveckla en gemensam terminologi i samarbete med kunden. De tre andra verkar inte ha någon direkt metod för att överbrygga just det problemet. Även Kund 2 hade registrerat språkproblemet men tyckte att utvecklaren måste få prata sitt språk för att komma underfund med vissa funktioner o.dyl. Under vårt samarbete med HÄLSA märkte vi bl.a. svårigheten för dem att förstå begrepp gällande gränssnittet till systemet. Vi tror att ett sätt att överbrygga denna svårighet vore att skapa någon form av ”ordbok” med bilder på vanliga funktioner och objekt där man kan peka ut aktuella termer för kunden.

Kund 2 påpekar att detta dock inte bara handlar om att kunden inte förstår systemutvecklarens ”fikonspråk” och tvärtom, utan även att systemutvecklaren inte förstår när kunden försöker prata IT-språk. Detta framhåller även Keller (1998) som menar att systemutvecklarna har svårt att lyssna på icketeknisk kunskap.
Kanske skulle det här ha underlättat om vi frågat kunden om han sett något som liknande hans vision om systemet någon annanstans. Då hade vi, utan att kunden skulle behövt förklara dem, lättare kunnat ta till oss funktioner och teknik.
En annan variant kan vara att be kunden berätta om syftet med systemet. Kanske kan utvecklaren, som är den kunnige inom området, då komma med ett smidigare sätt att nå samma syfte. I vårt fall ville exempelvis HÄLSA kunna skicka mail till kundens sida. Istället för att börja diskutera omöjligheten att maila till en hemsida skulle vi ha frågat om HÄLSAs syfte med att maila kunden och då fått fram ett antal faktorer som vi förhoppningsvis skulle kunnat arbeta fram en alternativlösning runt.

Systemutvecklare 2 poängterade vikten av att våga fråga om det är något man inte förstår. Det är också viktigt att tänka på att ställa frågan så den inte missförstås. Att våga fråga är dock svårt, det visade inte minst våra egna erfarenheter i samband med samarbetet med HÄLSA. Vi tror här att det då kan underlätta om man blir mer familjär med kunden. Kund 1 tog upp att det faktum att båda parter i hans projekt befann sig på samma nivå, att ingen ställde sig över den andra, var en klar fördel.

Systemutvecklarna 3/4 hade märkt att dialogen med kunden blir lättare om man träffas och umgås. De tog dock upp att tiden sätter begränsningar även här. När så skedde brukade de ha telefonkontakt, videokonferenser eller netmeetings. De föreslog även en annan lösning på tidsproblemet; att hyra in en konsult som innehar samma kunskap som kunden. Vi anser att det till viss del även skulle lösa problemet med okunskap och språksvårigheter, eftersom konsulten då skulle kunna vara någon slags ”alltiallo” som kunde lite av varje ifrån både utvecklarens och kundens värld.

Under vårt eget utvecklingsarbete i samarbete med HÄLSA upptäckte vi att prototyper var en användbar metod för att få kunden att förstå. Detta tog även alla fyra

systemutvecklarna upp. Ibland kan det vara så att kunden inte vet vad han vill ha förrän han ser det, vilket Systemutvecklare 3/4 tog upp, och då underlättar ju prototyper. Motsatsen förekommer också, d.v.s. att han vet att han inte vill ha det först när han ser det. Systemutvecklare 2 nämnde att grafiska presentationer ofta ledde till diskussioner men då handlade det mer om synpunkter än om missförstånd. Vi tolkar detta som att bilder kan vara ett sätt att göra det lättare för kunden att aktivt delta i diskussionen. Systemutvecklare 2 påpekade dock att det är svårt att göra bilder och prototyper av inbyggda system. I de fall detta förekom brukade han använda beskrivningar i textform.

Bilder är inte bara användbart vid presentation av själva systemet utan även när en organisation eller vissa begrepp ska förklaras. Kund 2 förklarade att han brukar ta en Powerpoint-presentation till hjälp när han ville få någon att förstå HÄLSAs organisation. Han hade själv lättare för att hänga med i visualiseringar och tyckte därför att även modeller över systemet är bra. Också Kund 1 nämnde bilder på gränssnitt och prototyper som en bra metod för att skapa förståelse för systemet. Trots att Kund 1 trott sig ha en klar bild över alla funktioner de ville ha i systemet så tillkom några efter att bilder och prototyper visats upp för dem.
En viktig sak att tänka på vid användandet av bilder är dock, anser vi, att inte ha för detaljerade bilder som visar för mycket på en gång. För mycket effekter under en presentation kan också stjälpa mer än det hjälper.

Systemutvecklare 3/4 insåg efter några projekt att det fungerar bättre om kunden själv får rita en modell över sin idé om systemet. De upplevde att de som utvecklare då hade mindre svårighet att förstå kundens modell än om de hade gjort tvärtom. Det är viktigt att dra nytta av sina erfarenheter, påpekade Systemutvecklare 1. Det var alltså inte så konstigt att vi upplevde relativt stora kommunikationsproblem i vårt samarbete med HÄLSA. Vi hade ingen erfarenhet av att möte en verklig kund med verkliga intressen och visioner om systemet. Kommunikation är dock ett beteende (vilket vi tog upp i kapitel 4.2) som kan förbättras genom övning och inlärning. Förhoppningvis har vi redan nu lärt oss en hel del inför framtida projekt.

Oerfarenhet eller orutin kan dock vara ett plus ibland ansåg Kund 2. Han beskrev det som att man inte har några förutfattade meningar eller uppfattningar om lösningar. Istället får systemutveckaren fria händer vilket kan underlätta.
Oerfarenhet avhjälps dock genom erfarenheter. Ett större problem kan då vara ointresse. Även om vår kund visste relativt lite om tekniken bakom systemutveckling var han åtminstone väldigt intresserad av Internet o.dyl., vilket vi ser som ett stort plus. Men om en kund är tekniskt helt ointresserad, hur mottaglig är han då för systemutvecklarnas information och hur väl kan han förklara sina visioner om vad systemet ska göra?

6.3 Resultat

När det gäller själva orsaken till kommunikationsproblemen har vi tagit fasta på våra egna och de intervjuade utvecklarnas erfarenheter. Litteraturen vi gått igenom hade även den en hel del att erbjuda när det gäller uppkomsten av nämnda problem.

Varför uppstår kommunikationsproblem?

1. *Bristen på tid* gör att utvecklare och kund inte hinner lära känna och förstå varandra och ger därmed upphov till sämre kommunikation.

2. Utvecklarens och kundens *olika bakgrund* gör att missförstånd i kommunikationen kan uppstå. Bakgrunden hos aktörerna kan bestå av värderingar, kulturell bakgrund, attityder och kommunikationsfärdigheter.

3. *Otydlighet* skapar en grogrund för missförstånd. Var så tydlig som möjligt när det gäller ex. frågeställningar eller förklaringar. En otydlig kravspecifikation ger ofelbart upphov till missförstånd längre fram i utvecklingsfasen.

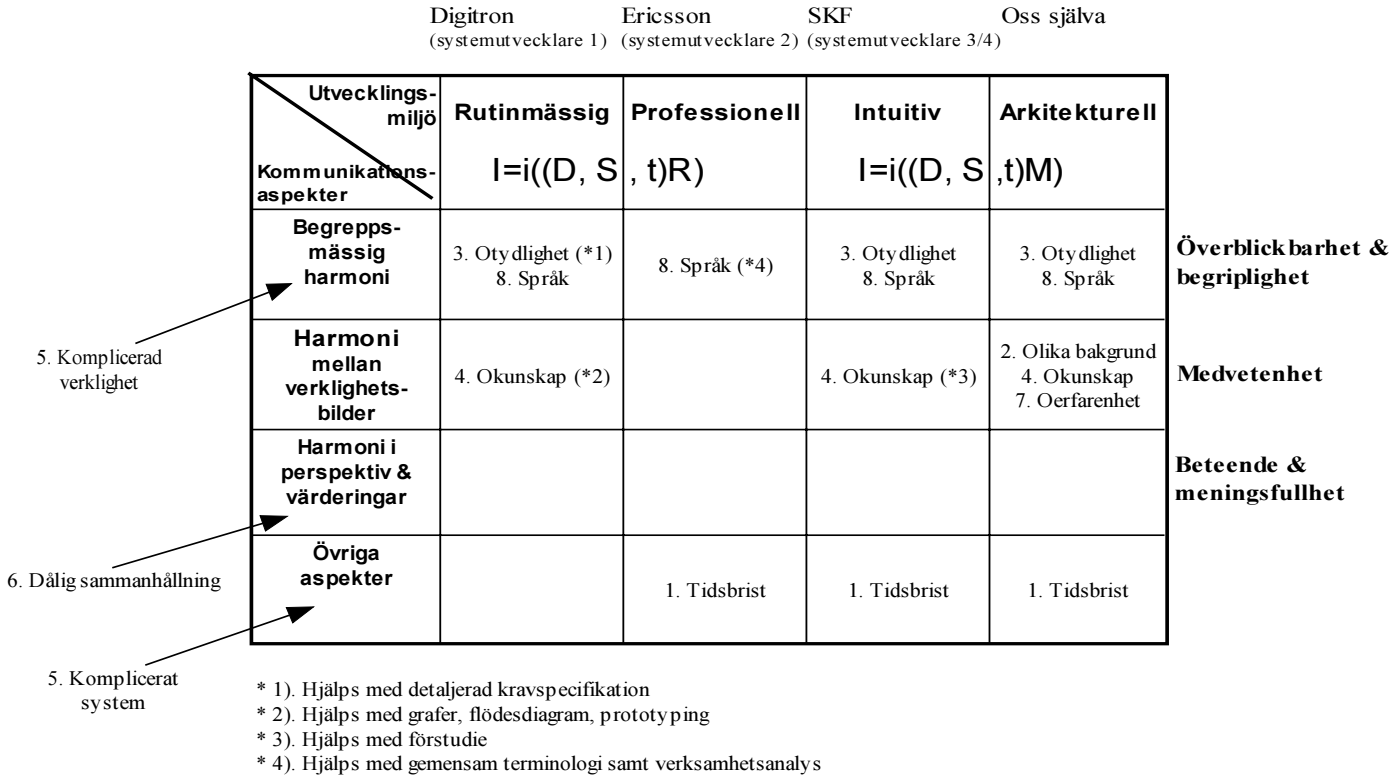
4. *Okunskap* hos endera parten kan orsaka problem, framförallt om någon part känner sig obehaglig till mods över att behöva fråga samma fråga mer än en gång.

5. Ett *komplicerat system/verklighet* tvingar fram mer detaljer att gå igenom vilket skapar fler risker för missförstånd.

6. *Dålig sammanhållning* inom projektgruppen ger upphov till en känsla av att man arbetar mot varandra istället för mot det gemensamma mål som systemet faktiskt är.

7. *Oerfarenhet* av systemutveckling, möten på andra parter o.dyl. leder till osäkerhet om hur de problem som uppstår ska hanteras. Man vet inte heller hur man ska gå tillväga för att minska riskerna för problem.

8. *Språkliga skillnader,* både vad det gäller fackterminologi och språk såsom svenska/engelska, kan leda till missförstånd och oförståelse.



Figur 6.2 Den teoretiska kartan har fyllts med vår empiriska undersökning

Vi har i matrisen i figur 6.1 fört in resultatet av vår empiriska undersökning, alltså både intervjuerna med systemutvecklarna och våra egna erfarenheter. Detta har resulterat i figur 6.2 som alltså kopplar ihop teoridelen i uppsatsen med empiridelen.

Vi har något hårdraget kategoriserat de fyra systemutvecklarna enligt Bergenstjerna, Johansson & Wojtasiks modell över designfilosofier. Vi fann något tveksamt att SKF Dataservice i viss mån kunde placeras i kategorin *intuitiv utvecklingsmiljö* då de hade relativt låg grad av standardiserad kunskap och låg grad av variation på uppdragen. De använde inga metoder utan förlitade sig mest kunskaper och egna erfarenheter. Ericsson Microwave placerade vi i *professionell utvecklingsmiljö* då de hade hög grad av standardiserad kunskap och hög grad av variation på uppdragen. Digitron hamnade i *rutinmässig utvecklingsmiljö* med en hög grad av standardisering och låg grad av variation. De hade ex. ett grundsystem att utgå ifrån och gjorde vissa ändringar på det på kunders begäran. Vi placerade oss själva i *arkitekturell utvecklingsmiljö* eftersom vi hade en låg grad av standardiserad tillgänglig kunskap och eftersom vi aldrig jobbat med systemutveckling förr så hade vi en hög grad av variation i uppdraget.

Vi har därefter placerat ut de punkter vi fick fram som svar på frågan varför kommunikationsproblem uppstår på de platser(vertikalt) där vi anser att de hör hemma och på de platser(horisontellt) med hänsyn till vilka systemutvecklare som upplevt dem. Exempelvis så anser vi att *språksvårigheter* hör till kategorin *Begreppsmässig harmoni* och problemet uppmärksammades av alla fyra systemutvecklarna. Punkterna *dålig sammanhållning* och *komplicerat system* nämndes inte av någon av utvecklarna men har ändå sin plats i den vertikala axeln.

Som avslutning på detta resonemang kan nämnas att det inte finns någon metod som går att använda i alla situationer utan metodstödet är beroende av situationen, vem man är samt av tidsfaktorn.

På vår fråga om det fanns standardiserade metoder i systemutvecklingsvärlden för att undvika kommunikationsproblem så tycker vi att svaret vi fått fram blev ett tveksamt ”nej”. Dock har vi tagit fasta på en del av de nyttiga erfarenheter systemutvecklare skaffat sig från tidigare projekt, samt även funnit en hel del matnyttigt ur våra litteraturstudier. Alltså, några råd för att undvika missförstånd i kommunikationen kan se ut som nedan.

Hur underlättas kommunikationen mellan utvecklare och kund?

1. *Våga fråga!* Det finns inga dumma frågor, och man måste skapa en såpass familjär miljö mellan utvecklare och kund att man skall kunna ställa samma fråga om och om igen tills man fått ett svar man är nöjd med. Tänk också på att ställa frågan så att den inte missförstås. Var noga med att ge *respons* och *feedback* vid förståelse respektive oförståelse.

2. *Undvik fackspråk* som motparten inte begriper. I mån av tid, utarbeta en gemensam terminologi som båda förstår.

3. *Lyssna uppmärksamt* på motparten. Tänk mer på *innehållet* i vad som sägs istället för på *hur* det framförs. Försök ha ett öppet sinne och övertolka inte.

4. *Träffa motparten* så ofta det behövs. Det är bättre att träffas och räta ut frågetecken än att försöka sig på kvalificerade gissningar och antaganden som kan bli dyrbara att korrigera i ett senare skede. Om det är svårt att få till stånd möten med kunden p.g.a ex. tidsbrist eller geografiska avstånd kan man ta in en konsult som besitter kundens kunskap.

5. Använd *prototyper* som hjälpmedel. Att detta är ett mycket bra hjälpmedel är både vi, våra intervjuoffer och författarna alla rörande överens om. En förenklad bild av systemet gör det mycket lättare för kunden att förstå utvecklarens syn på systemet och vice versa.

6. *Dokumentera* alla möten mellan utvecklare och kund noggrant. När ett beslut tagits om någon del av systemet så skall man alltid kunna gå tillbaka i dokumentationen och se vad som beslutats. På så sätt undviker man återigen misstaget att göra antaganden.

7. Se till att ha *gott om tid* att utföra projektet på. En god tidsmarginal ökar chansen att hinna göra sig ordentligt förstådd för motparten och därmed minska risken för missförstånd. Tyvärr är det sällan detta kan styras av utvecklarna själva ex. vid anbudsförfarande om ett systembygge.

8. Som projektledare i ett systemutvecklingsprojekt är det viktigt att ha *kunskap om människans kognitiva förmåga* och vi tror det är viktigt att vara utbildad inom områden som pedagogik, psykologi eller gruppdynamik.

9. Att *sätta sig in i kundens organisation* och göra en ordentlig förstudie är extremt viktigt för att få förståelse för hur systemet kommer att hjälpa kundens organisation.

7 Sammanfattning och slutsats

7.1 Slutsatser

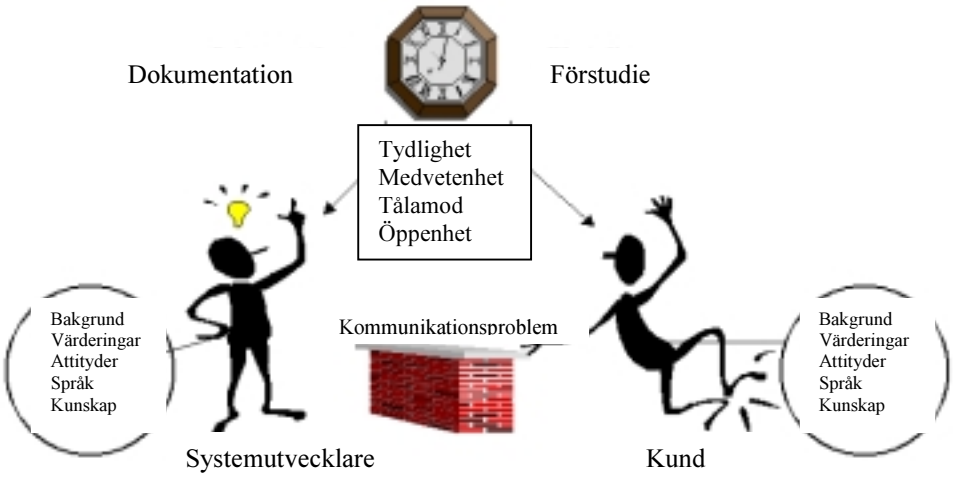
Till viss del vill vi påstå att vår förhoppning om att det skulle finnas standardiserade metoder för att undvika kommunikationsproblem inom systemutveckling blev något av en besvikelse. Vi tycker ändå att vi fått såpass bra kunskaper från våra intervjuade utvecklare och kunder samt genom litteraturen vi läst att vi kunnat sammanställa de råd vi angav i kapitel 6.3 för att på ett tidigt stadium i utvecklingsprocessen undvika missförstånd i kommunikationen.

De intervjuade systemutvecklarna hade t. ex. var och en sina egna små tips för att lösa problemen vi diskuterat och vi tycker vi fått många värdefulla råd att ta med oss i vår framtid inom IT-branschen. Även litteraturen har givit oss en bra grund i förståelsen av hur kommunikationen mellan utvecklare och kund bör se ut. Vi har även blivit mer medvetna om problemet och som vi tidigare nämnt är just medvetenheten en början på lösningen av problemet.

Vi ansåg själva i början av arbetet med vårt projekt att vi hade alldeles för lite på fötterna när det gäller att kommunicera tydligt och klart med kunder, men allteftersom arbetet fortskred så kom vi själva på lösningar för att underlätta kommunikationen, exempelvis användandet av prototyper. Att detta var en förträfflig lösning fick vi ju bekräftat av både litteraturen samt intervjuade kunder och utvecklare.

Vi tycker dock att problemet med missförstånd i kommunikationen bör tas upp på dagordningen inom projektgruppen om den inte redan finns där. Finns punkten på dagordningen blir det ett ämne att tala om och därmed möjlighet att kunna undvika problemen redan från början. Att vara medveten om varför man missförstår varandra, gör det ju lättare att lösa problemen.

Vi menar också att det behövs mer undervisning i kognitiv psykologi och kommunikationsvetenskap på högskoleutbildningar inom systemvetenskap eftersom det är en så väsentlig del av systemutvecklingsprocessen.



Figur 7.1 Bilden på omslaget har nu fått nytt utseende

.
Den bild vi visade på uppsatsens omslag har vi förhoppningsvis, genom att skriva den här uppsatsen, ändrat till att se ut som figur 7.1. I och med att de inblandade i systemutvecklingsprocessen insett att de bär på varsin ”ryggsäck” med olika bakgrund och genom att visa varandra respekt genom tålamod, öppenhet, tydlighet och medvetenhet samtidigt som de noggrant studerat organisationen som systemet skall införlivas i så har parterna lyckats att betydligt sänka muren som symboliserar kommunikationsproblemen.

7.2 Uppsatsens validitet

Med fakta och stöd både från teorin och empirin anser vi att vi har fått relativt klara och tillförlitliga svar på vår frågeställning. Vi har försökt att vara så objektiva som möjligt och sparat våra egna åsikter till de sista två kapitlen, men faktum kvarstår; vi är inte mer än människor och redan när vi väljer ut relevanta delar blir uppsatsen på sätt och vis subjektiv.

För att få en ännu bredare bild och fler förslag på tillvägagångssätt hade vi kanske kunnat intervjua fler systemutvecklare och kunder. När vi började intervjua såg vi dock att de flesta respondenter nämnde ungefär samma faktorer i olika ordalag, så frågan är om det inte hade blivit mest upprepningar.

Det hade varit önskvärt att intervjua systemutvecklare och kunder som var varandras motparter. Kanske hade man då ännu mer kunnat urskilja orsak till och verkan av kommunikationsproblem. Det tror vi dock hade skapat risken att systemutvecklarna valt ut en ”bra” kund där problemen var minimala. Vi tror att så var fallet med Kund 1, kund till Systemutvecklare 1, som knappt upplevde några problem alls.
Nu var det ju tyvärr inte möjligt att hitta motparten till de respondenter vi valde. En variant hade då varit att välja andra systemutvecklare men tidsresurserna svek oss.

Vi har i teoriavsnittet gjort vårt bästa för att få med lite olika vinklar och teorier om ämnet, både ur informatiksynpunkt och mer allmänt. Detta för att ge en så bred bild som möjligt av ämnet. Det finns säkert ytterligare matnyttig litteratur men tiden räckte tyvärr inte till för att finna all.

7.3 Uppslag till fortsatta studier

För att få fram konkreta *metoder* för att lösa kommunikationsproblem, eller rättare sagt utarbeta en metod för samarbete där kommunikationsproblem minimeras, anser vi att det hade krävts ytterligare bearbetning av de faktorer vi fick fram. Med stöd av detta kan man sedan utforma några slags praktiska tester som utvecklaren och kunden får genomgå. Dessa kan handla om t.ex. olika termer, förståelse av olika sorters bilder o.dyl. och kan sedan ligga till grund för metoder. Detta kan alltså vara ett uppslag till fortsatta studier.

En annan idé är att man mer detaljrikt undersöker och följer upp ett visst avslutat systemutvecklingsprojekt. Man gör ordentliga intervjuer med kunder och systemutvecklaren angående samarbetet, projektets resultat, problem, framtida önskemål o.dyl samt läser igenom kravspecikationen och alla andra dokument som rör

projektet. Genom att studera ett speciellt projekt, från a till ö, är det lättare att finna specifika lösningar och ta reda på vad parterna kunde gjort annorlunda.

En annan variant på ämnet kommunikationsproblem vore att studera kommunikationen mellan andra parter i systemutvecklingsprojektet. Detta kan vara exempelvis systemets ägare vs. dess användare eller programmerare vs. säljare.

8 Referenser

Ackoff, R.L. (1967). Management Misinformation Systems. *Management Science*, December 1967, pp. B147-B156.

Boland, R.J.Jr. (1979). Control, Causality and Information System Requirements. *Accounting Organizations and Society*, Vol 4, No. 4, 259-272.

Brown, D. (1997). *An introduction to Object-Oriented Analysis – objects in plain English*. Toronto: John Wiley & Sons.

Checkland, P.B. (1989). Soft Systems Methodogy. *Human Systems Management*, *8 1989*, 273-289.

Dahlbom, B. & Mathiassen, L. (1993). Computer in context – The philosophy and Practise of Systems Design. Oxford: Blackwell.

Dimbleby, R. & Burton, G. (1999). *Kommunikation är mer än bara ord*. Lund: Studentlitteratur.

Eklund, S. & Fernlund, H. (1998*). Programkonstruktion med kvalitet – projekthantering och ISO 9000*. Lund: Studentlitteratur.

Enquist, H. (1999). *IT-management för komplexa ledningssystem - arkitekturbegrepp för samverkan verksamhet – leverantör.* Göteborg: Department of Informatics, Göteborg University.

Bergenstjerna, M., Johansson, L. & Wojtasik, M (1999). *Metoder för strategisk IT-management.* Göteborg: Institutionen för informatik, Göteborgs universitet

Eysenck, W.M. (1993). *Principles of Cognitive Psychologym*, Erlbaum: Taylor % Francis.

Flynn, D.J, & Warhurst, R. (1994). An empirical study of validation process within requirements determination. *Information Systems Journal 4*, pp 185-212.

Hedberg, B. (1980). Using Computerized Information Systems to design better organizations and jobs. In N. Bjorn-Andersen (Ed.), *The Human Side of Information Processing*. North Holland Publishing Company.

Keller, A. (1998-02-01). Användarvänlig dator spar pengar. *Dagens Nyheter*, Ekonomi/Söndag.
Keil, M., Carmel, E. (1995). Customer-Developer Links in Software Development. *Communications of the ACM,* May 1995 Vol.38 No.5.

Langefors, B. (1986). Information and Management Systems in E. Johnsen, ed, *Trends and Megatrends in the Theory of Management*. Copenhagen**:** Bratt International**.**

Langefors, B. (1995). *Essays on Infology*. Lund: Studentlitteratur.

Lewis, P. (1994). *Information-Systems development*. London: Pitman Publishing

Lubars, M., Potts, C., & Richter, C. (1993). *A review of the state of the practise in requirements modelling*. Proc IEEE Symposium on Requirements Engineering, San Diego, California.

Magoulas, T., & Pessi, K. (1998). *Strategisk IT-management*. Göteborg: Department of Informatics, Göteborg University.

Martin, J. (1991). *Rapid application development*. New York: Macmillan Publishing Company.

McGinnes, S. (1992). How Objective is Object-Oriented Analysis?. In P. Loucopoulos (Ed.), *Advanced Information Systems Engineering*. Manchester: Springer-Verlag.

Moody, D. (1996). *Graphical Entity Relationship Model: Towards a More User Understandable Representation of Data*. Intallheim Bernhard (Red.) Conceptuell Modelling-Entity Relationship 96, 15th International Conference on the ER approach, Cottbus, Germany, October 96.

Pressman, R.S. (1997). *Software Engineering – Fourth Edition*. New York: McGraw-Hill.

Roland, C (1998). *A comprehensive View of Process Engineering*
Advanced Information Systems Engineering, 10th International Conference, CaiSE'98, Pisa, Italy, June 1998, Proceedings

Sommerville, I. (1996). *Software Engineering*. Essex: Addison Wesley.

Vonk, R. (1990), *Prototyping*. Hertfordshire: Prentice Hall International.

9 Bilaga

9.1 Intervju 1

Respondent: Systemutvecklare

1.1 Hur skulle du vilja beskriva företaget? Är det t.ex. en IT-avdelning eller ett konsultföretag? Vilken typ av uppdrag åtar ni er?
1.2 Vilka arbetsuppgifter har just du? Ingår systemanalys som en naturlig del i dina arbetsuppgifter? På vilket sätt har du kontakt med kunden?
1.3 Vilken bakgrund har du? Utbildning? Antal år I branschen? Annat av intresse?

2.1 Hur hanterar ni praktiskt ett möte mellan kund och utvecklare?
2.2 Finns det t.ex. klara regler och rutiner för hur det ska gå till?
2.3 Hur vill du beskriva kundens roll i mötet?
2.4 På motsvarande sätt – hur vill du beskriva systemutvecklarens roll?
2.5 Hur gör ni er förstådda för kunden?

3. Hur skulle du vilja beskriva kommunikationen mellan dig och kunden i allmänhet? Tycker du att den fungerar tillfredsställande? Upplever du att det är lätt att göra sig förstådd?

4. Vilken typ av svårigheter menar du är vanligast förekommande?

5. Hur skulle du vilja beskriva förekomsten av missf./kommunikationssvårigheter? Liten, stor, mycket stor?

6.1 Vad tror du att det beror på/var uppstår problemen?
6.2 Har du någon uppfattning om varför problemen uppstår?

7.1 Hur gör/hanterar du då detta?
7.2 Använder ni er av några metoder för att lösa dem? Har ni i så fall utarbetat egna eller fått lära er några standardmetoder?
7.3 I de fall där problem uppstod, med facit i handen, hur hade det bäst kunnat lösts?

9.2 Intervju 2

Respondent: Kunden

1.1 Hur skulle du vilja beskriva företaget? Vilken typ av uppdrag åtar ni er?
1.2 Vilka arbetsuppgifter har just du? På vilket sätt har du kontakt med utvecklaren?
1.3 Vilken bakgrund har du? Utbildning? Antal år I branschen? Annat av intresse?

2.1 Hur hanterar ni praktiskt ett möte mellan er som kund och utvecklare?
2.2 Finns det t.ex. klara regler och rutiner för hur det ska gå till?
2.3 Hur vill du beskriva utvecklarens roll i mötet?
2.4 På motsvarande sätt - hur vill du beskriva din egen kundroll?
2.5 Hur går ni tillväga för att få dem att förstå er organisation?

3. Hur skulle du vilja beskriva kommunikationen mellan dig och utvecklaren I allmänhet? Tycker du att den fungerar tillfredsställande? Upplever du att det är lätt att göra sig förstådd?

4. Vilken typ av svårigheter menar du är vanligast förekommande?

5.1 Hur skulle du vilja beskriva förekomsten av missf./kommunikationssvårigheter?
5.2 Liten, stor, mycket stor?

6.1 Vad tror du att det beror på/var uppstår problemen?
6.2 Har du någon uppfattning om varför problemen uppstår?

7.1 Hur hanterades problemen?
7.2 Användes det några metoder för att lösa dem?
7.3 I de fall där problem uppstod, med facit i handen, hur hade det bäst kunnat lösts?