Use case som teknik för identifiering och dokumentering av krav

(HS-IDA-EA-02-306)

Helén Fredh (b99helfr@student.his.se)

Institutionen för datavetenskap
Högskolan i Skövde, Box 408
S-54128 Skövde, SWEDEN

Examensarbete på det systemvetenskapliga programmet under vårterminen 2002.

Handledare: Jessica Lindblom

Use case som teknik för identifiering och dokumentering av krav

Examensrapport inlämnad av Helén Fredh till Högskolan i Skövde, för Kandidatexamen (B.Sc.) vid Institutionen för Datavetenskap.

[2002-06-07]

Härmed intygas att allt material i denna rapport, vilket inte är mitt eget, har blivit tydligt identifierat och att inget material är inkluderat som tidigare använts för erhållande av annan examen.

Signerat: _____

Use case som teknik för identifiering och dokumentering av krav

Helén Fredh (b99helfr@student.his.se)

Sammanfattning

Ett effektivt användande av ett informationssystem förbättrar informationshanteringen inom en verksamhet. För att ett informationssystem ska kunna fungera effektivt krävs att det motsvarar de krav som ställts på informationssystemet av olika intressenter.

Requirements Engineering (RE) är en viktig del av systemutvecklingsprocessen för att kunna säkerställa en väl fungerande kravhantering. Use case är en teknik som kan användas som hjälpmedel i RE-processen för att identifiera och dokumentera krav.

Syftet med detta examensarbete är att undersöka om use case är tillräcklig som enda teknik för att identifiera och dokumentera krav samt vilka eventuella kompletterande tekniker som används bland systemutvecklare. Resultatet av undersökningen visar att use case-tekniken inte är tillräcklig utan måste kompletteras med andra tekniker för att möjliggöra att samtliga krav kan identifieras och dokumenteras.

Nyckelord: Kravhantering, Requirements Engineering, UML, Use case

Innehållsförteckning

Bilagor

1 Introduktion

Idag använder de flesta verksamheter och organisationer någon form av informationssystem. Eriksson och Penker (2000) påstår att informationsteknik är en integrerad del av det dagliga arbetet hos majoriteten av företag. De menar vidare att ett effektivt användande av datoriserade informationssystem förbättrar de flesta verksamheter och kan ibland vara helt nödvändiga för att hantera den information som finns i företaget. Många företag är dock inte nöjda med det sätt som deras informationssystem fungerar. För att komma till rätta med detta problem och utveckla väl fungerande och effektiva informationssystem måste olika förbättringar göras i systemutvecklingsprocessen (Eriksson och Penker, 2000).

Requirements Engineering (RE)[1] är en mycket viktig del i systemutvecklingsprocessen och innefattar främst identifiering och dokumentering av krav på det tänkta informationssystemet men som finns med som en del i hela systemutvecklingsprocessen. Det är viktigt att åstadkomma en gemensam förståelse mellan användare och utvecklare av ett informationssystem för den verksamhet som informationssystemet ska verka i samt de krav som framkommer i RE-processen. En kravspecifikation är ett dokument kring vilket användare och utvecklare kan diskutera kraven för informationssystemet och komma överens om en gemensam syn på dessa krav. RE är en viktig del av systemutvecklingsprocessen eftersom det är av stor vikt att korrekta och fullständiga krav kan tas fram och dokumenteras i kravspecifikationen för att sedan kunna utveckla ett väl fungerande informationssystem (Karlsson, 1996; Loucopoulos & Karakostas, 1995). Det finns olika sorters krav, den vanligaste kategoriseringen av krav är enligt Karlsson (1996) funktionella och ickefunktionella krav. Vissa krav kan uttryckas klart och tydligt, det vill säga de är explicita. Denna kategori av krav är lättare att hantera än implicita krav vilka är outtalade krav som ändå tas för givet av intressenterna till informationssystemet (Karlsson, 1996). Samtliga sorters krav ska dokumenteras i en kravspecifikation som sedan utgör en grund för den fortsatta systemutvecklingsprocessen. Brister i kravspecifikationen kan få stora konsekvenser längre fram i utvecklingen av ett informationssystem.

För att förenkla och förbättra arbetet med att hantera krav och skapa en förståelse mellan användare och utvecklare för kraven används ofta någon form av modelleringsspråk (Booch *et al*., 1999). Med hjälp av ett modelleringsspråk skapas en representation av verkligheten, till exempel i form av grafiska bilder såsom diagram. Dessa är lättare att hantera än stora textmassor och de är samtidigt också lättare för användarna att förstå då de inte alltid har kunskap om systemutveckling och de specifika begrepp som kan förekomma inom systemutvecklingsområdet. Enligt Eriksson och Penker (2000) är ett exempel på ett modelleringsspråk som har haft, och fortsätter att ha, ett stort inflytande i systemutvecklingskretsar *The Unified Modeling Language (UML)*. De menar vidare att är det många företag som idag använder UML som enda modelleringsspråk för sin systemutveckling. Enligt Kruchten (2000) skapar UML:s modeller av informationssystemet förståelse och fungerar som ett hjälpmedel i utformandet av både problem och lösningar i informationssystemet.

[1] Det finns inget direkt motsvarande svenskt begrepp för Requirements Engineering, dock kommer i denna rapport även begreppet kravhantering att användas, det motsvarar då det engelska begreppet Requirements Engineering.

UML innehåller flera olika diagram som är anpassade för olika stadier i systemutvecklingsprocessen. För hantering av krav i systemutvecklingsprocessen finns bland annat *use case-tekniken*[2] i UML. Use case kan användas i såväl text- som diagramform eller som en kombination av båda. Enligt Regnell (1999) samt Lee och Xue (1999) har intresset för use case-modellering i systemutvecklingsprocessen ökat, särskilt i de tidiga faserna, vilka inkluderas i RE-processen.

Use case är en teknik som kan användas som hjälpmedel vid identifiering och dokumentering av krav i systemutvecklingsprocessen. Enligt Cockburn (2001) kan use case ses som en beskrivning över ett informationssystems beteende då det interagerar med aktörer av olika slag vilka existerar utanför informationssystemet. Det kan till exempel vara en ordermottagare (aktör) som registrerar en order (use case) i informationssystemet. Ett use case kan också beskrivas som en steg-för-steg-beskrivning över de steg som en aktör utför i informationssystemet (Schneider & Winters, 2001). För att identifiera vilka use case som finns i en verksamhet förs ofta diskussioner mellan systemutvecklare och användare till det tänkta informationssystemet. Användarna kan fungera som domänexperter och systemutvecklarna som experter på use case-tekniken samt kan belysa möjligheter och begränsningar för tekniska lösningar.

1.1 Problemställning

Eftersom use case-tekniken blivit alltmer populär finns det mycket som talar för att denna teknik är fördelaktig att använda vid identifiering och dokumentering av krav. Dock finns det begränsningar med tekniken vilka också är viktiga att belysa eftersom tekniken vunnit så stor popularitet. Med hänsyn till ovanstående har i examensarbetet undersökts hur väl use case-tekniken passar för identifiering och dokumentering av krav samt vilka andra tekniker som systemutvecklare eventuellt använder som komplement till use case-tekniken. Följande problemprecisering har formulerats:

> *Är use case tillräcklig som enda teknik för att identifiera och dokumentera krav. Om inte; hur kompletterar systemutvecklare de former av krav som use case-tekniken inte fångar in?*

1.2 Tillvägagångssätt

För att undersöka examensarbetets frågeställning ansågs intervjuer vara ett lämpligt tillvägagångssätt. En fallstudie var också planerad att genomföras om möjlighet fanns, dock gavs negativt svar från de tillfrågade företagen varför en fallstudie inte kunde genomföras. Slutligen blev intervjuer i kombination med ytterligare litteraturstudier det tillvägagångssätt som valdes för undersökningen.

[2] Motsvarande svenskt begrepp för use case är användningsfall men eftersom use case är ett vedertaget begrepp inom systemutveckling kommer det engelska begreppet att användas i denna rapport.

Innan intervjuerna genomfördes framställdes ett antal intervjufrågor som vägledning för intervjuerna. Dock var intervjuerna relativt öppna och det fanns utrymme för eventuella nya frågor under intervjuernas gång. Kontakt togs med ett antal systemutvecklingsföretag och intervjuer genomfördes slutligen på två olika företag med totalt sex intervjudeltagare. Urvalet av intervjudeltagare gjordes av kontaktpersoner på respektive företag och visade sig vara ett gott urval av både erfarna och oerfarna systemutvecklare. Intervjuerna bandades för att inte förlora värdefull information som kunde framkomma under intervjuerna. När intervjuerna var genomförda transkriberades samtligt material. Utifrån textmaterialet genomfördes en sammanställning och analys.

1.3 Resultat och slutsatser

Det resultat som framkom av undersökningen överensstämde till stor del med den litteratur som studerats. Use case anses inte vara tillräcklig som enda teknik för att identifiera och dokumentera krav. Use case-tekniken fungerar väl för att fånga upp funktionella krav på ett informationssystem men tekniken har stora begränsningar vad gäller ickefunktionella krav. Use case-tekniken anses vara lätt att lära och förstå och den fungerar väl i kommunikationen mellan systemutvecklare och användare. Dock anses det vara svårt att avgöra vilken detaljnivå ett use case ska ha samt svårt att få en uppfattning om processen/flödet i informationssystemet.

Under undersökningen har flera olika kompletterande tekniker framkommit. I den litteratur som undersökts har det inte funnits mycket information om kompletterande tekniker. Under intervjuerna gavs dock en hel del information kring vilka tekniker som används tillsammans med use case-tekniken. Dock har det framkommit att det saknas en formell beskrivning över kompletterande tekniker på ett motsvarande sätt som finns för use case. Det bör vara värdefullt att utveckla en eller flera tekniker, som på samma sätt som use case-tekniken fungerar för funktionella krav, kan fungera för ickefunktionella krav samt till exempel processbeskrivningar.

Under examensarbetets gång har det tydligt framkommit att kravhantering är en mycket viktig del i systemutvecklingsprocessen och att många systemutvecklare anser att mer resurser bör läggas på kravhantering. Use case-tekniken har visat sig vara en god hjälp i kravhanteringen men den är dock inte tillräcklig. Det är av stor vikt att ständigt försöka förbättra kravhanteringen och de tekniker som används i denna process då det finns mycket att vinna för en verksamhet om det finns en väl fungerande kravhanteringsprocess.

1.4 Rapportens uppbyggnad

I nästa kapitel presenteras den bakgrund och de centrala begrepp som skapar en grund för förståelse för det problemområde som kommer att behandlas i detta examensarbete. Bland annat presenteras Requirements Engineering och use case. I kapitel 3, Problembeskrivning, ges en ytterligare beskrivning av det aktuella problemområdet och den problemställning som detta examensarbete har som utgångspunkt presenteras. Fortsättningsvis i kapitlet presenteras även de avgränsningar som gjorts för området samt förväntat resultat av undersökningen.

Kapitel 4, Tillvägagångssätt, presenterar de metoder som ansetts vara lämpliga för att utföra examensarbetets undersökning. I kapitlet beskrivs också den arbetsprocess som slutligen genomfördes för att få svar på examensarbetets problemställning. Dessutom presenteras de företag samt intervjudeltagare som medverkat i undersökningen. I kapitel 5, Resultat och analys, presenteras en sammanställning av det material som samlats in samt vilka resultat som framkommit. En analys genomförs också kring dessa resultat. Kapitel 6, Slutsatser och diskussion, består av en presentation av de slutsatser som tagits av det resultat som framkommit utifrån examensarbetets frågeställning samt en diskussion kring dessa. Dessutom förs en diskussion kring use case-tekniken i allmänhet och den information som framkommit under intervjuerna som ej direkt kan kopplas till problempreciseringen. Slutligen presenteras de erfarenheter som erhållits under arbetsprocessen med examensarbetet samt förslag på fortsatta arbeten inom området.

2 Bakgrund

I detta bakgrundskapitel kommer de områden som ligger till grund för examensarbetet att presenteras. Först presenteras en övergripande beskrivning av *informationssystem* och *systemutveckling*[3]. Efter denna översikt presenteras kravhanteringsprocessen, Requirements Engineering (RE) mer ingående samt beskrivs krav och vilka kategorier av krav som finns. Därefter ges en introduktion till Unified Modeling Language (UML) för att sedan ge en mer grundlig beskrivning av use case och hur denna teknik kan användas för att identifiera och dokumentera krav, samt dess möjligheter och begränsningar.

2.1 Informationssystem och systemutveckling

Idag är det vanligt att ett informationssystem är en del av flertalet verksamheter. Enligt Eriksson och Penker (2000) påverkas det dagliga arbetet i de flesta företag på olika sätt av informationsteknik. Andersen (1994, sid 15) definierar ett informationssystem som *”ett system för insamling, bearbetning, lagring, överföring och presentation av information”*. Han nämner även att ett informationssystem dessutom förmedlar information mellan människor i en verksamhet. Även Avison och Fitzgerald (1995) betonar de manuella aspekterna av ett informationssystem. De menar att ett informationssystem ofta beskrivs som ett datoriserat system för hantering av information, men att de mänskliga aspekterna är minst lika viktiga. De människor som arbetar i verksamheten och de interaktioner som de har med det datoriserade systemet är i högsta grad en del av informationssystemet. Ett informationssystem tillhandahåller också den information som behövs för ett effektivt styrande av organisationen enligt Avison och Fitzgerald (1995). De menar att ett väl fungerande informationssystem är ett viktigt konkurrensmedel för en organisation. Med ökat användande av Internet som bas för kommunikation ökar kraven på företagens informationssystem ytterligare (Eriksson & Penker, 2000).

Andersen (1994) påpekar att ett informationssystem inte har något egentligt syfte i sig utan måste existera i en verksamhet för att inneha något specifikt ändamål. Även Booch, *et al*. (1999) beskriver att ett informationssystem interagerar med antingen mänskliga eller automatiserade aktörer vilka använder informationssystemet för ett visst syfte och har vissa förväntningar på hur informationssystemet ska agera. I dagens informationssystem är det inte längre bara rådata och text som hanteras utan även mer komplex data såsom bilder och ljud (Avison & Fitzgerald, 1995).

[3] I den litteratur som refereras till i detta examensarbete förekommer olika beskrivningar kring informationssystem, systemutveckling, programvaruutveckling samt andra typer av system. För att underlätta för läsaren kommer endast begreppen informationssystem och systemutveckling att användas i detta examensarbete.

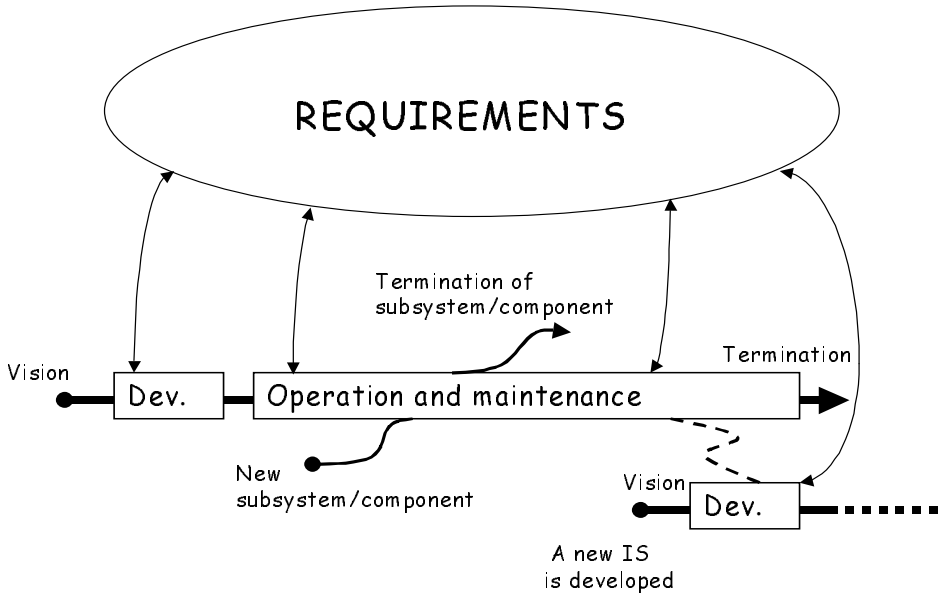
Enligt Avison och Fitzgerald (1995) är ett misslyckat informationssystem inte längre en följd av brister av teknisk karaktär, vilket tidigare varit fallet, utan troligtvis en verkan av mänskliga och organisatoriska skäl. Exempel på sådana skäl kan enligt Avison och Fitzgerald (1995) vara bristfällig planering och otillräcklig utbildning för de anställda men även bristfälliga utvecklingsmetoder, tekniker och verktyg. Även Kruchten (2000) nämner ett antal problem med informationssystemutveckling. Ett par av de problem som nämns är missuppfattningar av användarnas behov och oförmåga att hantera föränderliga krav, vilket är viktiga skäl till att förbättra kravhanteringen inom informationssystemutveckling.

Den process som sker vid utveckling av ett informationssystem kallas för systemutveckling. Avison och Fitzgerald (1995) beskriver systemutveckling som en iterativ process eller livscykel. Den traditionella systemutvecklingens livscykel innehåller olika steg från att undersöka om kraven för det nya informationssystemet är genomförbara, via analys och design, till implementation och underhåll. Målet för systemutvecklingen är ofta ett väl fungerande informationssystem som utvecklats enligt de krav som framkommit under systemutvecklingsprocessen. Dessa krav samlas in från olika intressenter till det informationssystem som ska utvecklas. Intressenter till ett informationssystem är personer eller representanter för en organisation som har ett intresse, ofta ekonomiskt, av det resulterande informationssystemet. Exempel på intressenter är användare, utvecklare samt kunder (Kruchten, 2000).

Kortfattat kan kravhanteringsprocessen, eller Requirements Engineering, beskrivas som att den verksamhet där det nya informationssystemet ska finnas analyseras, samt att krav på det nya informationssystemet samlas in. Detta kan ske på ett flertal sätt, till exempel genom diskussioner med användarna av det nya informationssystemet men även genom att ta hänsyn till eventuellt tidigare system samt till exempel verksamhetsspecifika aspekter och lagar. De krav som samlats in kan representeras på olika sätt, textdokument, diagram, gränssnittsprototyper och liknande. Av alla de krav som samlats in väljs vissa krav ut som ska dokumenteras i en kravspecifikation. Denna kravspecifikation utgör sedan grunden för att skapa det nya informationssystemet och följer hela systemutvecklingsprocessen eftersom kraven kan förändras och kravspecifikationen ständigt behöver kompletteras. I följande avsnitt presenteras Requirements Engineering i mer detalj.

2.2 Requirements Engineering

Requirements Engineering (RE) är ett relativt nytt begrepp som beskriver hantering av krav i systemutvecklingsprocessen. Enligt Sommerville och Sawyer (1997) innefattar RE de första faserna i systemutvecklingsprocessen, vilka främst är identifiering, dokumentering och underhåll av krav. Fokus i RE-processen ligger på de första faserna men RE-processen löper över hela systemutvecklingsprocessen i och med att kraven måste hanteras under hela denna process. Det kan tillkomma nya krav likväl som gamla krav förändras med tiden. Figur 1 nedan, beskriver den traditionella systemutvecklingens livscykel och hur RE-processen förhåller sig till denna. Tidigare räknades RE som de tidigare faserna i systemutvecklingsprocessen (Development i figur 1) men enligt detta synsätt anses RE-processen sträcka sig över hela systemutvecklingsprocessen. Detta synsätt innebär därmed att krav ständigt kan återanvändas och underhållas genom hela systemutvecklingsprocessen (Dahlstedt, 2001).



Figur 1. RE-processens ur livscykelperspektiv, från Dahlstedt (2001, sid 51).

I litteraturen finns det flera olika definitioner av vad RE innebär. Å ena sidan definierar Loucopoulos och Karakostas (1995, sid 13, egen översättning) RE som:

> *”En systematisk process bestående av utveckling av krav genom en iterativ samverkansprocess bestående av analys av problemet, dokumentation av de resulterande observationerna i olika representationsformer och att kontrollera korrektheten i den förståelse som erhållits”.*

Pohl (1996, sid 4) å andra sidan definierar RE enligt IEEE standard (IEEE-610.12, 1991):

1. Den process som innebär att studera användares behov för att komma fram till en definition av system-, hårdvaru- eller mjukvarukrav.
2. Den process som innebär att studera och förfina system-, hårdvaru- eller mjukvarukrav.

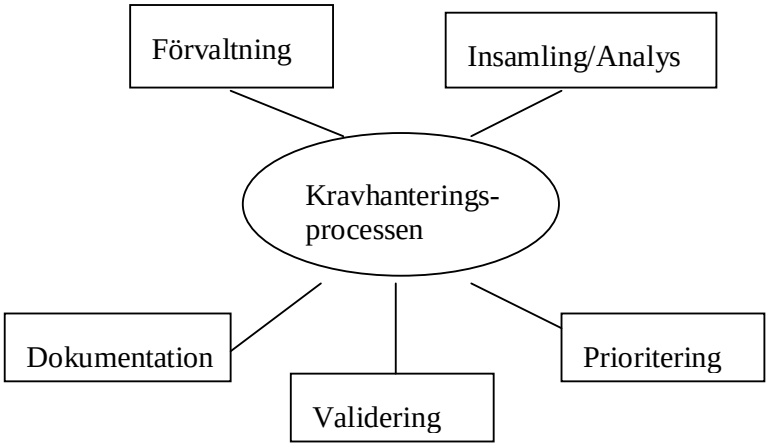
Dessutom definierar Gause och Weinberg (1989) RE som den del i systemutvecklingsprocessen där personer försöker upptäcka vad som är önskvärt.

Loucopoulos och Karakostas (1995) förtydligar sin definition av RE som en systematisk process genom att hävda att RE innebär att försöka förstå användarnas behov av informationssystemet och att formulera och beskriva dessa behov på ett sätt som kan användas i systemutvecklingen. Viktigt att poängtera är att användarnas behov ofta är en kombination av de problem som finns i det dagliga arbetet vilka användaren förväntar sig att informationssystemet ska lösa och de specifika lösningar som användaren själv har tänkt sig för det aktuella problemet (Wiegers, 1997). Saiedian och Dale (2000) påpekar att det även är viktigt under RE-processen att skapa en förståelse hos användarna för vilka begränsningar som finns, till exempel i form av tekniska begränsningar eller lagar, samt hur dessa begränsningar kommer att påverka deras behov och det färdiga informationssystemet.

Ytterligare ett sätt att se på RE-processen är Pohls (1996) tredimensionella ramverk för kravhantering. De tre dimensionerna är ’specification dimension’, ’representation dimension’ och ’agreement dimension’. I specification dimension utvecklas en kravspecifikation som ska vara så komplett som möjligt och då används oftast någon form av mall för att avgöra graden av kompletthet. I representation dimension behandlas användning av olika sorters representationer eller språk som används i systemutvecklingsprocessen. Det gäller då till exempel informella språk (tex naturligt språk, bilder), semiformella språk (tex diagram) och formellt språk (tex specifikationsspråk som Z). I agreement dimension arbetas en överenskommelse fram mellan de olika intressenterna till det tänkta informationssystemet så att en gemensam syn på kraven på informationssystemet finns.

En viktig del i RE-processen är hanteringen av krav och framförallt *identifiering* av krav, vilket innebär att med användarnas hjälp finna och specificera de önskemål som finns på det tänkta informationssystemet. I kravhanteringen ingår också *dokumentation* av de identifierade kraven. Dokumentation av kraven leder till en kravspecifikation som kan ses som målet för RE-processen. Kravspecifikationen är en mycket viktig del i systemutvecklingen då denna utgör grunden för det fortsatta arbetet med utvecklingen av informationssystemet. För att åstadkomma ett effektivt informationssystem krävs en väl genomarbetad kravspecifikation som alla parter är överens om. Med en gemensam uppfattning av kraven i kravspecifikationen kan utvecklingen av informationssystemet fortgå på ett korrekt sätt.

Hantering av krav innefattar enligt Kruchten (2000) tre aktiviteter vilka är identifiering, organisering samt dokumentering av informationssystemets funktionalitet och restriktioner. Karlsson (1996, sid 12) beskriver kravhanteringsprocessen som ”de aktiviteter som utförs för att på ett effektivt sätt hantera krav på ett programvarusystem”. Till skillnad mot Kruchtens (2000) tre aktiviteter hävdar Karlsson (1996) att kravhanteringsprocessen består av fem aktiviteter, nämligen insamling/analys, prioritering, dokumentation, validering och förvaltning, se figur 2 nedan.



Figur 2. Aktiviteter i kravhanteringsprocessen (från Karlsson, 1996, sid 13)

Det som skiljer Kruchtens (2000) och Karlssons (1996) beskrivningar av kravhanteringsprocessen är aktiviteterna prioritering, validering och förvaltning. Samtliga dessa tre aktiviteter är viktiga för att korrekta krav ska kunna framställas och bevaras. Prioritering av krav är nödvändigt ur kostnads- och tidsperspektiv, validering säkerställer att de olika intressenterna är överens om vad kraven innebär och förvaltning av krav möjliggör en enhetlig behandling av kraven genom hela utvecklingsprocessen (Karlsson, 1996). Det föreligger en risk för att RE-processen förenklas, med följden att den blir bristfällig, om hänsyn enbart tas till de tre aktiviteter Kruchten (2000) nämner. Eftersom RE-processen lägger grunden för det informationssystem som ska utvecklas är det viktigt att förståelse finns hos de iblandade för RE-processens omfattning och komplexitet, varför RE-processen i detta arbete anses bestå av Karlssons (1996) fem aktiviteter.

Karlsson (1996) menar att de allvarligaste problemen som inträffar under utvecklingen av ett informationssystem kan hänföras till kravhanteringsprocessen och att effekten av en väl fungerande kravhanteringsprocess bör leda till nöjdare kunder och användare av de informationssystem som utvecklas. Enligt Loucopoulos och Karakostas (1995) finns det flera problem som kan uppkomma vid kravinhämtning:

- Användare har inte alltid en klar bild över sina krav på det nya informationssystemet.
- Användare kan ha svårt för att beskriva sin domänkunskap.
- Ofta använder systemutvecklare begrepp som är dataorienterade medan användarna har uttryck och begrepp som är domänspecifika. Detta leder till brister i kommunikationen.
- Användare kan ibland vara negativt inställda till ett nytt informationssystem och vill på grund av detta inte medverka i kravhanteringsprocessen.

Även Kruchten (2000) menar att en väl fungerande kravhantering löser många problem i systemutvecklingsprocessen. Bland annat nämner han att mycket kommunikation mellan olika intressenter sker via kraven samt att krav måste kunna prioriteras och spåras. I en systemutvecklingsprocess kan motstånd uppkomma hos användarna mot det informationssystem som är under utveckling, Saiedian och Dale (2000) påpekar att kunskap om hur sådant motstånd kan identifieras och minskas är mycket viktigt för att utvecklingen ska förlöpa väl. Även här är en väl fungerande kommunikation mellan utvecklare och användare viktig för att minska det eventuella motstånd som framträder. Om användarna informeras om vad som sker under utvecklingen av informationssystemet och även får ta del i utvecklingen förekommer en mindre risk för att de blir negativt inställda till förändringen. Ovanstående faktorer understryker behovet av väl definierade krav. Enligt Kruchten (2000) ger en effektiv kravhantering också bättre kontroll över komplexa projekt och ökar kvaliteten på informationssystemet. Han nämner vidare minskade kostnader och mindre risk för förseningar i projektet som resultat av en effektiv kravhantering.

Ett av de grundläggande begreppen i RE är *krav*, det vill säga till exempel de önskemål som finns på det informationssystem som ska utvecklas. Det är viktigt att det finns en förståelse för vad som menas med krav i detta sammanhang och vilka olika kravkategorier som finns. Kravbegreppet kommer att diskuteras närmare i nästa avsnitt.

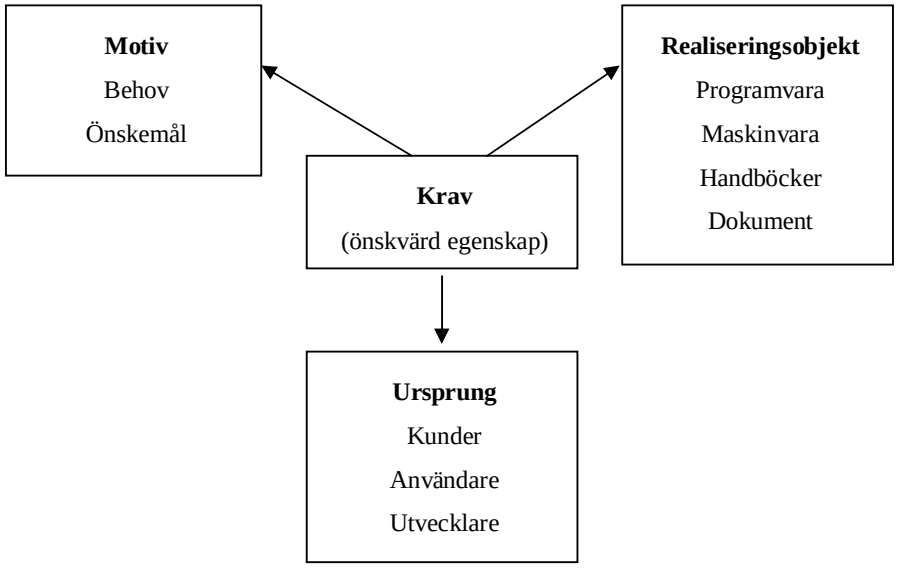
2.3 Vad är krav?

Kruchten (2000) beskriver ett krav som ett villkor som ett informationssystem måste uppfylla. Karlsson (1996) däremot beskriver krav som önskvärda egenskaper hos ett informationssystem. Karlsson (1996) menar att krav inte alltid måste uppfattas som en ovillkorlig egenskap även om han poängterar att lagar, förordningar och standarder kan vara exempel på sådana ovillkorliga egenskaper och dessa kan tvinga systemutvecklare att utforma informationssystemet på ett visst sätt. Oftast förknippas dock krav med de, enligt användarna och intressenterna, önskvärda egenskaper som ett tänkt informationssystem ska inneha. Det är dock viktigt att ta hänsyn till båda dessa typer av krav och att avvägningar görs i de fall oförenliga krav uppkommer.

Loucopoulos och Karakostas, (1995, sid 2) definierar vad ett krav är enligt IEEE-standard (IEEE-610, 1990):

1. Ett villkor eller en egenskap som en användare behöver för att lösa ett problem eller uppnå ett mål.
2. Ett villkor eller en egenskap som måste uppnås eller innehas av ett system eller systemkomponent för att tillfredsställa ett kontrakt, en standard, en specifikation, eller andra formella dokument.
3. En dokumenterad representation av ett villkor eller en egenskap såsom i 1 eller 2.

Det finns krav som beskriver *vad* ett informationssystem ska göra och det finns krav som beskriver *hur* det ska göras. Enligt Karlsson (1996) har ett krav även ett visst ursprung ifrån en eller flera intressenter, ett motiv till varför det finns, är det ett behov eller önskemål, samt ett realiseringsobjekt, det vill säga det slutliga resultatet, till exempel ett informationssystem eller en programvara. Figur 3 nedan visar de olika delar som ett krav består av, enligt Karlsson (1996).



Figur 3. Ett kravs tre beståndsdelar (från Karlsson, 1996, sid 9)

Enligt Kruchten (2000) är ett av de vanligaste problemen i systemutvecklingsprojekt att kraven på ett informationssystem inte är konstanta utan förändras ständigt allt eftersom utvecklingen av informationssystemet fortskrider. Även Harker *et al.* (1993) menar att en ständig förändring av kraven är mer regel än undantag. Om förändringar av kraven inte kan hanteras korrekt menar Kruchten (2000) att detta kan leda till förseningar och missnöjda kunder. Karlsson (1996) beskriver ett annat problem som kan uppkomma gällande krav nämligen hur de beskrivs. Han hävdar att det finns två orsaker till att krav beskrivs felaktigt. Först kunskapsfel, vilket innebär att de inblandade inte vet vilka som är de korrekta kraven och sedan specificeringsfel, vilket innebär att de inblandade inte vet hur kraven ska dokumenteras.

Sommerville och Sawyer (1997) beskriver ytterligare fyra problem som ofta uppstår vid hantering av kraven på ett informationssystem:

1. Kraven reflekterar inte kundens/användarens verkliga behov av systemet.
2. Kraven är inkonsistenta eller inte kompletta.
3. Det är kostsamt att göra förändringar i kraven efter att de fastställts.
4. Det uppstår missförstånd mellan olika intressenter.

Väl definierade krav är mycket viktigt i systemutvecklingsprocessen på grund av att de sätter kriterium för den acceptans, framgång och användbarhet som krävs av det informationssystem som ska utvecklas (Loucopoulos & Karakostas, 1995). Var börjar då processen med att identifiera dessa krav? För det första är det viktigt att ha kunskap om olika kategorier av krav och detta presenteras närmare i nästa avsnitt.

2.3.1 Kategorisering av krav

Krav kan delas upp i olika kategorier. Enligt Karlsson (1996) är den mest traditionella kategoriseringen av krav att dela in kraven i *funktionella* och *ickefunktionella* krav.

Karlsson (1996) menar att funktionella krav beskriver de tjänster som informationssystemet ska tillhandahålla användaren, medan Kruchten (2000) beskriver funktionella krav som de krav som specificerar det beteende som informationssystemet har samt det resultat beteendet ska leda fram till. Ett funktionellt krav kan till exempel vara att när användaren klickar på spara-knappen ska datan lagras på hårddisken. Ickefunktionella krav beskriver de egenskaper som ett informationssystem ska uppfylla men som inte kategoriseras som tjänster enligt Karlsson (1996), medan Kruchten (2000) beskriver ickefunktionella krav som en mängd krav som finns på informationssystemet men som inte kan härledas till de funktionella kraven. Detta kan till exempel innebära informationssystemets kvalitet, användbarhet och effektivitet. Kruchten (2000) påpekar dock att det är viktigt att inte endast se till vad informationssystemet ska kunna utföra utan även varför. På frågan vad informationssystemet ska göra finns svaren bland de funktionella och ickefunktionella kraven. Frågan varför tar hänsyn till förståelse för olika intressenters behov som måste uppfyllas av informationssystemet (Kruchten, 2000).

Enligt Karlsson (1996) kan krav också kategoriseras enligt deras förmåga att tillfredsställa de olika intressenterna. Han beskriver vidare tre typer av krav, vilka är *sensationella*, *normala* och *förväntade* krav. Sensationella krav är inte uttalade av användare men kan leda till stor tillfredsställelse om informationssystemet uppfyller dem. Det kan till exempel vara tekniska möjligheter som användaren inte känner till men som skulle vara fördelaktiga för informationssystemets förmåga att stödja användarnas arbetsuppgifter. Om dessa krav inte uppfylls kommer detta dock inte att leda till minskad tillfredsställelse för användarna eftersom de inte förväntar sig dem. Normala krav är ofta uttalade av användare och kan därför identifieras. Normala krav kallas också för explicita krav. Ju bättre denna typ av krav är tillgodosedda desto högre är användarnas tillfredsställelse med informationssystemet. Ett exempel på ett normalt krav kan vara effektivitet. Förväntade krav är inte uttalade av användare men kommer ändå att leda till minskad tillfredsställelse om informationssystemet inte uppfyller dem. Exempel på sådana krav är grundläggande behov hos användarna som är så vanliga att de inte blir uttalade, användarna tar för givet att informationssystemet kommer att uppfylla dessa krav (Karlsson, 1996). Krav som användarna inte känner till eller inte kan uttrycka kan också kallas för implicita krav (Lauesen, 2000). Även om användarna kan uttrycka kraven så gör de så med sina egna termer och med implicit kunskap om sina arbetsuppgifter som kan påverka kraven men som inte uttrycks (Sommerville & Sawyer, 1997). Den kunskap som användarna har men inte kan uttrycka kallas även för tyst kunskap (Waern, 1993) och det krävs stor domänkunskap hos de som utvecklar informationssystemet för att förväntade och implicita krav ska kunna identifieras.

Förutom användarnas krav på informationssystemet finns också *systemkrav*. Systemkrav är oftast mycket mer detaljerade är användarkrav och de kan uttryckas med hjälp av matematiska eller grafiska modeller, dock finns alltid kommentarer till modellerna vilka skrivs i naturligt språk (Sommerville & Sawyer, 1997).

Att hantera krav på ett informationssystem är svårt då kravhanteringsprocessen är mycket komplex. Det finns många olika kategorier av krav och det är svårt för systemutvecklare att överskåda samtliga krav. Det krävs stor kunskap om kravhantering såväl som om problemdomänen för att möjliggöra korrekt hantering av de krav som ställs på det informationssystem som ska utvecklas. Första steget i kravhanteringsprocessen är att identifiera och dokumentera de olika krav som finns på det tänkta informationssystemet. Hur identifiering och dokumentering av krav kan gå till kommer att diskuteras i följande avsnitt.

2.3.2 Identifiering och dokumentering av krav

Enligt Kruchten (2000) identifieras krav genom en process som börjar med att samla in krav från olika intressenter till informationssystemet för att sedan strukturera, analysera och dokumentera dessa krav. Det är i detta skede viktigt att kunna tillgodogöra sig och förstå den domänkunskap som de olika intressenterna innehar. Domänkunskap är den kunskap som innehas av olika personer gällande det aktuella området, det vill säga det aktuella problemet eller den verksamhet som det tänkta informationssystemet ska utvecklas för. Det kan till exempel vara att undersöka specifika arbetsuppgifter och hur informationssystemet ska stödja dessa uppgifter.

Det är viktigt att hålla kontakten med intressenterna för att kunna diskutera eventuella brister och inkonsistens i de uttryckta kraven (Kruchten, 2000). Denna första fas i RE-processen är mycket viktig för den fortsatta utvecklingen. Även Loucopoulos & Karakostas (1995, sid 21) beskriver detta på följande sätt *”för att kunna lösa någon annans problem så måste man först ta reda på mer information om problemet”*.

Enligt Pohl (1996) finns relevant information om problemet bland olika intressenter, lagar, standarder och även dold i existerande informationssystem. Han menar vidare att för att kunna identifiera kraven är det viktigt att kunna identifiera de olika informationskällor som finns i verksamheten. I och med den ökade komplexiteten av den information som finns i verksamheter idag, finns denna representerad på flera olika sätt (Pohl, 1996), till exempel i form av dokument men också i form av de anställdas yrkeskunskaper och erfarenhet. I dokument kan informationen till exempel representeras med hjälp av bilder, skisser, naturligt språk eller formella modeller. Denna representation dokumenteras i någon form av kravspecifikation som sedan ligger till grund för vidare utvecklingsarbete.

Kravspecifikationen kan ses som den produkt som är resultatet av RE-processen (Pohl, 1996). Kravspecifikationen beskriver kundens och användarnas krav på informationssystemet samt den funktionalitet och de villkor som måste uppfyllas av informationssystemet (Loucopoulos & Karakostas, 1995). Lauesen (2000) menar att en kravspecifikation ska beskriva information som tillförs informationssystemet och information som hämtas ut från informationssystemet, eller det som användaren kommer att se i det färdiga informationssystemet. Kvaliteten på kravspecifikationen är enligt Karlsson (1996) viktigt av tre anledningar:

- Den fungerar som underlag för alla kommande utvecklingsaktiviteter.
- Den kan fungera som ett formellt kontrakt mellan leverantör och kund.
- Den används ofta vid testning för att verifiera att det nya informationssystemet verkligen uppfyller de önskvärda egenskaperna.

Även Loucopoulos och Karakostas (1995) beskriver tre anledningar till varför det är viktigt att upprätta en kravspecifikation.

- Kravspecifikationen fungerar som kommunikationsmedel.
- Kravspecifikationen fungerar som ett kontrakt.
- Kravspecifikationen kan användas för att utvärdera slutprodukten och för att testa kundens acceptans av informationssystemet.

Loucopoulos och Karakostas (1995) menar även att kravspecifikationens kvalitet och därmed också informationssystemets kvalitet till stor del beror på hur väl systemutvecklarna kan utvinna och förstå kunskap om den domän för vilken informationssystemet utvecklas. Lauesen (2000) beskriver olika kvalitetskriterier för en kravspecifikation, enligt IEEE-standard (IEEE-830, 1993), en kravspecifikation ska vara:

Korrekt – Varje krav reflekterar ett behov

Komplett – Alla nödvändiga krav ska vara inkluderade

Entydig – Alla intressenter ska vara överens om betydelsen av kraven

Konsistent – Olika delar ska inte motsäga varandra

Ordnad efter betydelse och stabilitet – Prioritering och förväntade förändringar för varje krav

Förändringsbar – Enkel att förändra och att upprätthålla konsistens

Verifierbar – Möjligt att se om varje krav har uppfyllts

Spårbar – Spåra krav till mål och design

Enligt Saiedian och Dale (2000) är en välskriven kravspecifikation en nödvändighet för att kunna utveckla ett informationssystem. Utan kravspecifikationen vet inte systemutvecklarna vad de ska utveckla, användarna vet inte vad de ska förvänta sig och det finns ingen möjlighet att validera att det färdiga informationssystemet möter de behov som fanns hos användarna från början. Kravspecifikationen har många användningsområden och är ett viktigt kommunikationsverktyg mellan de olika intressenter som är engagerade i utvecklingen. Kravspecifikationen skapar också en gemensam ram för hur informationssystemet ska fungera och alla inblandade utvecklare ska använda samma version av kravspecifikationen för detta syfte. Ofta fungerar också kravspecifikationen som ett kontrakt mellan kund och leverantör varför det är mycket viktigt att denna blir så korrekt som möjligt, om tveksamheter uppstår ska det vara möjligt att kontrollera i kravspecifikationen vad som överenskommits.

Hur kan då de krav som ska dokumenteras i kravspecifikationen samlas in på ett effektivt sätt? För att förenkla denna process kan ett modelleringsspråk vara till god hjälp. Grafiska bilder kan göra kraven lättare att hantera och de utgör samtidigt en hjälp i kommunikationen mellan användare och utvecklare. Ett exempel på ett modelleringsspråk som används allt mer i systemutvecklingskretsar är The Unified Modeling Language (UML), som beskrivs närmare i nästa avsnitt.

2.4 The Unified Modeling Language

Enligt Booch, *et al*. (1999) så finns det många anledningar till att ett systemutvecklingsprojekt misslyckas, men de projekt som verkligen lyckas har oftast modellerat på ett effektivt sätt. En modell är en förenkling av verkligheten och modeller kan användas för att möjliggöra en bättre förståelse för det informationssystem som ska utvecklas utifrån systemutvecklarnas synvinkel såväl som för andra intressenter, främst användarna av informationssystemet (Booch, *et al.* 1999). För systemutvecklarna hjälper modeller till att skapa kunskap om aktuell verksamhet samt tydliggöra vad det är som ska byggas. För användare är modeller ett stöd för att ta fram krav på det tänkta informationssystemet. The Unified Modeling Language (UML) är ett grafiskt språk för visualisering, specificering, konstruering och dokumentering i systemutvecklingsprocessen (Booch, *et al.* 1999; Fowler & Scott, 2000).

UML utvecklades i syfte att framställa en standard för modellering med objektorienterad inriktning (Fowler & Scott, 2000). Karlsson (1996) beskriver att den grundläggande principen i objektorientering är att skapa modeller av det informationssystem som ska utvecklas, baserat på klasser och objekt. Ett objekt är något som kan identifieras, har ett tillstånd och ett beteende och flera objekt som har gemensamma egenskaper kan grupperas i klasser (Andersen, 1994).

UML är ett modelleringsspråk och inte en metod enligt Fowler och Scott (2000). En metod ger detaljer om olika steg som ska genomföras i systemutvecklingsprocessen samt vilka tekniker, till exempel modelleringsspråk som ska användas i de olika stegen. Fowler och Scott (2000) menar vidare att modelleringsspråket är den notation som en metod använder för att beskriva en design. Med notation menas de grafiska bilder som finns i olika modeller (Fowler & Scott, 2000). Även Eriksson och Penker påpekar att UML innefattar en standardiserad notation för olika modeller men inte en standardiserad process, det vill säga de steg som en metod bidrar med, för hur modellerna ska framställas. UML används ofta tillsammans med The Rational Unified Process (RUP)[4] och RUP bidrar med processen och UML bidrar med modelleringsspråket.

De modeller av informationssystemet som används i UML skapar förståelse och fungerar som ett hjälpmedel i utformandet av både problem och lösningar i informationssystemet (Kruchten 2000). Enligt Eriksson & Penker (2000) består UML av nio olika diagramtyper och varje diagram visar en specifik del av informationssystemet, det kan vara antingen en statisk eller en dynamisk aspekt av informationssystemet. Nedan presenteras de nio diagrammen kortfattat.

Klassdiagram beskriver strukturen i ett informationssystem och bygger på klasser och relationer.

Objektdiagram uttrycker möjliga kombinationer av objekt tillhörande ett specifikt klassdiagram. Används ofta för att exemplifiera ett klassdiagram.

Tillståndsdiagram visar möjliga tillstånd för en klass eller för ett informationssystem.

Aktivitetsdiagram beskriver aktiviteter och händelser som sker i ett informationssystem.

Sekvensdiagram visar en av flera sekvenser av meddelanden som skickas mellan en mängd objekt.

Samarbetsdiagram beskriver samarbete mellan en mängd objekt.

Komponentdiagram är en specialversion av klassdiagram som används för att beskriva komponenter i ett informationssystem.

Deploymentdiagram är en specialversion av klassdiagram som används för att beskriva hårdvaran i ett informationssystem.

Use case-diagram visar relationer mellan olika use case. Varje use case beskriver en del av informationssystemets funktionalitet.

[4] För närmare beskrivning av RUP se till exempel Kruchten (2000) *The Rational Unified Process: An Introduction*

Eriksson och Penker (2000) menar att UML har haft, och fortsätter att ha, ett stort inflytande och ökat användande inom systemutveckling och många företag använder UML som enda modelleringsspråk. Booch, *et al*. (1999) anser att UML passar för att modellera informationssystem i alla storlekar och komplexitet och att UML trots sin bredd är lätthanterligt och lättförståeligt. Vidare menar Fowler och Scott (2000) att det huvudsakliga skälet till att använda UML är att underlätta kommunikation mellan olika intressenter i systemutvecklingsprocessen. När UML används bör kommunikationen och utbytet mellan utvecklare och användare kunna ske med större fördel än till exempel vid naturligt språk, som inte ger tillräckligt tydliga beskrivningar av informationssystemet, och kod som är alltför detaljerat och kan vara svårt för användare att förstå och ta till sig.

Dock benämner Simons och Graham (1998) flera problem med UML-modellering. Bland de problem, för systemutvecklare som använder UML, de påträffat finns tvetydig semantik i notationen, lätt att missuppfatta notationen, bristfällig beskrivning av viktiga egenskaper i informationssystemet, brist på verktygsstöd samt brist på erfarenhet av UML hos utvecklare. Simons och Graham (1998) påpekar dock att vissa av dessa problem kan lösas genom att utbilda systemutvecklare för att ökar kunskapen om UML och hur UML kan tolkas. Övriga problem kräver dock en översyn av UML i sig och de verktyg som idag finns tillgängliga för stöd i användandet av UML. UML har alltså, som nämnts, fått ökad uppmärksamhet den sista tiden och har även fått mycket god kritik av många personer i systemutvecklingskretsar. Det finns dock en del problem med UML, dessa problem har nu också börjat uppmärksammas.

Enligt Eriksson och Penker (2000) förespråkar användare av UML att systemutvecklingsprocessen bör inledas med use case-modellering för att definiera funktionella krav på informationssystemet. Use case och use case-diagram är den teknik inom UML som används för att försöka identifiera och dokumentera olika sorters krav. I nästa avsnitt ges en mer grundlig beskrivning av use case.

2.5 Use case

Use case är en teknik som används för att hantera krav i systemutvecklingsprocessen. Use case kan ses som en steg-för-steg-beskrivning över vad ett informationssystem ska kunna utföra. Cockburn (2001) beskriver use case som en beskrivning av informationssystemets beteende och olika villkor som ska uppfyllas då systemet interagerar med olika intressenter. Intresset för use case-modellering i systemutvecklingsprocessen har ökat, särskilt i RE-processen. Grundidén till use case är att identifiera och dokumentera krav genom diskussioner med olika intressenter till det tänkta informationssystemet (Regnell, 1999).

Det finns olika varianter av use case. Den variant som kommer att beskrivas och undersökas i detta examensarbete är den som finns presenterad som en del av UML (se till exempel Booch, *et al*. 1999; Fowler & Scott, 2000). Kruchten (2000) använder begreppet metod för att beskriva use case, Avison och Fitzgerald (1995) däremot benämner olika modelleringstyper som tekniker. Både Maciaszek (2001) samt Fowler och Scott (2000) benämner use case som en teknik. En teknik är enligt Andersen (1994) ett arbetssätt eller ett sätt att göra en beskrivning. Med hänsyn till ovanstående resonemang kommer begreppet teknik att användas fortsättningsvis i detta arbete i samband med use case.

Ett use case kan ses som en ögonblicksbild av ett specifikt skeende i ett informationssystem, till exempel ’registrera ny kund’ och en samling av samtliga use case ger den externa bilden av informationssystemet (Fowler & Scott, 2000). Det finns tre centrala begrepp i användandet av use case vilka kräver viss beskrivning.

Use case är en sekvens av händelser som informationssystemet utför och som ger ett resultat som har ett värde för någon aktör (Booch, *et al*. 1999, Kruchten, 2000). Ett use case kan också ses som ett visst sätt som informationssystemet kan användas på (Karlsson, 1996). Ett exempel på ett use case kan vara ’kontrollera stavning’.

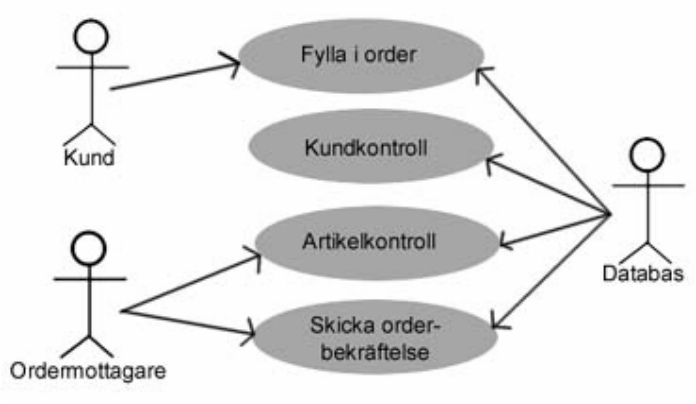
En *Aktör* är någon eller något utanför informationssystemet som interagerar med informationssystemet (Kruchten, 2000). En aktör kan vara en användare men kan också vara till exempel ett annat informationssystem (Karlsson, 1996). En och samma person kan också utgöra flera olika aktörer, till exempel slutanvändare och systemadministratör (Lee & Xue, 1999). Aktören utför ett use case (Fowler & Scott, 2000) och är den/det som initierar alla interaktioner med informationssystemet (Kulak & Guiney, 2000).

En *händelse* är någon form av procedur som anropas då en aktör ger en signal till informationssystemet. En sekvens av händelser beskriver ett flöde av händelser i informationssystemet (Kruchten, 2000).

Utöver ovanstående begrepp finns, enligt Fowler & Scott (2000) också fyra typer av relationer som kan förekomma mellan olika use case. Dessa relationer beskrivs nedan med de engelska begreppen då det inte förekommer någon riktig svensk översättning på dessa begrepp:

Association beskriver kommunikationsvägen mellan en aktör och ett use case (Maciaszek, 2001). *Include*-relationen uppkommer då det existerar en samling av beteenden som liknar varandra och överlappar flera use case, man behöver då inte kopiera beskrivningen av detta beteende. *Generalization* används då ett use case liknar ett annat use case men varierar i beteende. *Extend*-relationen liknar generalization men har fler regler kopplade till sig och är mer formellt beskriven (Fowler & Scott, 2000). De olika relationerna kommer ej att undersökas eller beskrivas närmare då de inte är i fokus för problemställningen i detta examensarbete.

Ett use case kan vara både textbaserat och i form av ett diagram eller en kombination av båda. Use case-diagrammet introducerades för att visualisera use case (Jacobson, 1994 i Fowler & Scott, 2000). Ett use case-diagram visar en mängd use case med aktörer och relationer. Use case-diagram är mycket användbara för att modellera informationssystemets beteende. Vanligtvis används use case-diagram på följande två sätt. Antingen för att modellera vad ett informationssystem ska innefatta eller för att modellera krav på ett informationssystem (Booch, et al. 1999). Det är viktigt att poängtera att ett use case-diagram endast är en illustration för att hjälpa till att förstå det textbaserade use caset (Cockburn, 2001). Även Fowler och Scott (2000) påpekar att Use case-diagram inte är nödvändiga, textbaserade use case fungerar väl som enda use case-form. I ett use case-diagram representeras ett use case av en ellips. En sådan ellips representerar en till två sidor text i ett textbaserat use case (Kulak & Guiney, 2000). En aktör representeras med en streckgubbe i ett use case-diagram där aktörens namn också skrivs ut och varje händelse i diagrammet representeras med en pil. Figur 4 nedan visar ett enkelt use case-diagram som illustrerar inköp av varor via Internet. En kund fyller i en order och en ordermottagare hanterar denna order. Databasen är iblandad i samtliga händelser då dessa lagras eller hanteras i databasen.



Figur 4. Use case-diagram. Diagrammet visar tre aktörer och de use case som dessa kan utföra. Pilarna representerar olika händelser som kan utföras på de olika use casen.

Ett textbaserat use case beskriver ett händelseflöde, det vill säga ett antal steg som utförs av en aktör (Schneider & Winters, 2001). Figur 5 nedan visar ett kortfattat exempel på ett textbaserat use case som motsvarar use case-diagrammet i figur 4. Normalt sett kan ett use case motsvara flera sidor text.

Inköp via Internet

Kund fyller i order i form av kundvagn

1. Kund går till kassan
2. Kund fyller i leveransinformation (adress, leveranstid)
3. Databasen kontrollerar om kunden är ny eller redan finns lagrad
4. Databasen släpper order till ordermottagare
5. Ordermottagare kontrollerar om beställda artiklar finns inne
6. Ordermottagare sänder orderbekräftelse per e-post till kunden

Alternativ: Fel i kundkontroll

Databasen misslyckas i steg 4 att kontrollera kundens uppgifter

Tillåt kunden att åter fylla i kunduppgifter och försöka igen

Figur 5. Textbaserat use case

Sutcliffe, *et al*. (1998) beskriver use case som ett nätverk av händelser, sammankopplade för att uppnå ett mål. Målet beskriver syftet med use caset. Use case har användardefinierade egenskaper som beskriver vad use caset ska göra, till exempel transaktion eller kontroll (Sutcliffe *et al*. 1998). Use case används för att beskriva ett informationssystems tänkta beteende utan att behöva beskriva hur detta beteende ska implementeras. Ett use case beskriver *vad* ett informationssystem utför men inte *hur* det utförs. Use case ger en möjlighet för utvecklare och användare att skapa en gemensam förståelse för informationssystemet och vad det ska åstadkomma (Booch, *et al*. 1992). Även om use case ofta förknippas med objektorientering så är det en generell teknik som går att använda även för icke objektorienterade projekt (Regnell, 1999).

Ett use case beskrivs med domänspecifika begrepp som har betydelse för användarna istället för i dataspecifika termer som kan vara svårt för användare att förstå (Wiegers, 1997). Use case utvecklades för att åstadkomma en mer formell användning och dokumentation av scenarios än vad som tidigare funnits (Fowler & Scott, 2000). För att ge ytterligare förståelse för vad ett use case är kan begreppet scenario användas.

Ett scenario kan enligt Loucopoulos och Karakostas (1995) beskrivas som en framställning av hur ett tänkt informationssystem kommer att uppfylla användarens behov. De beskriver vidare att ett scenario visar en detaljerad beskrivning av en specifik interaktion mellan människa och informationssystem. Även Fowler och Scott (2000) beskriver scenarios som en sekvens av steg som beskriver en interaktion mellan en användare och ett informationssystem. Scenarios används enligt Kruchten (2000) för att gestalta en sekvens av händelser eller för att upprätthålla en röd tråd i ett use case. Han menar vidare att ett enskilt use case kan ses som ett scenario och flera sammankopplade use case beskriver ett flöde i informationssystemet. Fowler och Scott (2000) menar istället att ett scenario är *en* händelse som kan ske i informationssystemet och ett use case är en mängd sammankopplade scenarios som har ett gemensamt mål för användaren. Ett use case kan liksom scenario beskrivas som en typisk interaktion som en användare har med systemet för att uppnå ett mål (Fowler & Scott, 2000).

Hur sker då identifiering av de use case som är aktuella för informationssystemet? I nästa avsnitt beskrivs hur identifiering av use case kan gå till.

2.5.1 Identifiering av use case

För att identifiera use case är det lämpligt att interagera och diskutera med användare av det tänkta informationssystemet hävdar Fowler & Scott (2000). De rekommenderar att identifiering av aktörer sker som första åtgärd. Detta kan underlättas genom att använda ett antal frågor som grund (Schneider & Winters, 2001). Det kan påpekas att dessa frågor endast är riktlinjer och att tillvägagångssättet är relativt fritt för systemutvecklaren.

- Vem använder informationssystemet?
- Vem installerar informationssystemet?
- Vem startar upp informationssystemet?
- Vem underhåller informationssystemet?
- Vem stänger av informationssystemet?
- Vilka andra system använder informationssystemet?
- Vem erhåller information från informationssystemet?
- Vem förser information till informationssystemet?
- Finns det något som händer automatiskt i informationssystemet?

Som ytterligare stöd vid intervjuerna kan fyra frågor ställas om aktörerna (Jacobson, 1992, i Maciaszek, 2001).

- Vilka är huvuduppgifterna som utförs av varje aktör?
- Kommer en aktör ha tillgång till eller förändra information i informationssystemet?
- Kommer en aktör informera informationssystemet om förändringar i andra system?
- Ska en aktör informeras om oväntade förändringar i informationssystemet?

På liknande sätt som för att identifiera aktörer kan identifiering av use case förenklas genom att använda ett antal frågor (Schneider & Winters, 2001).

- Vilka funktioner vill aktören att informationssystemet ska utföra?
- Lagrar informationssystemet någon information? Vilka aktörer kommer att skapa, läsa, uppdatera eller radera denna information?
- Kommer informationssystemet behöva informera någon aktör om förändringar i dess tillstånd?
- Finns det några interna händelser som informationssystemet måste känna till? Vilka aktörer informerar informationssystemet om dessa händelser?

Varje use case måste innehålla detaljer om vilka villkor som måste uppfyllas för att dess funktionalitet ska uppnås. Hänsyn måste tas till villkor som ska uppfyllas före respektive efter use caset utförs. Exempel på sådana villkor kan vara att en berättigad användare måste ha loggat in innan use case funktionen utförs och om funktionen utförts utan avbrott så ska en uppdatering ske. När de olika use casen framställts kategoriseras de utifrån värde för verksamheten, utvecklingsrisk och tidslängd för utvecklandet av use caset (Fowler & Scott, 2000). Det kan vara svårt att avgöra om interaktioner mellan användare och informationssystem ska beskrivas som ett eller flera use case. Enligt Kruchten (2000) identifieras use case enklast genom att fokusera på det önskade resultat som informationssystemet ska tillhandahålla en specifik aktör. Ett use case uppfyller ett mål som en specifik aktör har och det ska genomföras av informationssystemet. Karlsson (1996) beskriver också omfattningen av ett use case som en beskrivning av hur en aktör använder en av informationssystemets tjänster för att försöka uppnå ett mål.

Use case är en teknik som kan användas för att identifiera och dokumentera krav (RE-processen), detta beskrivs närmare i nästa avsnitt.

2.5.2 Möjligheter och begränsningar med use case i RE-processen

Ett use case kan användas för att identifiera och samla ihop användarnas krav till meningsfulla mängder (Fowler & Scott, 2000). Kruchten (2000) beskriver use case som ett hjälpmedel för att uttrycka funktionella krav som ställs på informationssystemet. Use case kan ses som ett resultat av kravhanteringsprocessen och används för att beskriva vad informationssystemet ska göra, ur användarens synvinkel.

Maciaszek (2001) menar att ett funktionellt krav i många fall går att översätta direkt till ett use case medan Fowler och Scott (2000) vänder på det och hävdar att varje use case är ett potentiellt krav. De betonar att use case är centralt för att kunna förstå vad användarna vill ha. De menar vidare att use case är mycket väl lämpade för att hantera krav och att use case sedan driver på utvecklingsprocessen och är grundläggande för att planera utvecklingen. Även Booch, *et al.* (1999) anser att de flesta, om inte alla krav kan beskrivas med hjälp av use case. Det finns många fördelar med att använda use case som teknik för att identifiera och dokumentera krav.

Fowler och Scott (2000) menar att det inte finns någon situation där use case inte kan användas. De menar vidare att use case är en utmärkt teknik för att identifiera krav. Regnell (1999) beskriver att användandet av use case underlättar hanteringen av komplexitet i systemutvecklingsprocessen eftersom use case fokuserar på en sak i taget. Han beskriver också use case som en bra teknik för att engagera användarna i systemutvecklingsprocessen då tekniken är enkel och kan baseras på verksamhetsspecifika begrepp. Även Lee & Xue (1999) beskriver hanteringen av komplexitet som en viktig fördel med use case. De beskriver vidare hur use case utgår från synvinkeln att ett informationssystem byggs för dess användare. Tuok och Logrippo (1998) anser att use case är mycket användbart för att beskriva krav på ett informationssystem tack vara att de innehar en mänsklig förståelse och bidrar till en kreativitet som krävs för att identifiera och analysera krav.

Karlsson (1996) menar att use case är ett effektivt sätt att hantera funktionella krav (se avsnitt 2.3.1 kategorisering av krav) och Kulak och Guiney (2000) anser att use case passar utmärkt även för icke-funktionella krav (se avsnitt 2.3.1 kategorisering av krav). De menar att då use case diskuteras med användarna så framkommer även ickefunktionella krav och dessa bör då dokumenteras direkt istället för att göra identifiering av ickefunktionella krav till en separat process. Då ett specifikt use case diskuteras bör systemutvecklarna fråga användarna om de ickefunktionella krav som kan kopplas till use caset, till exempel svarstider (Kulak & Guiney, 2000). Kruchten (2000) menar att det ofta förekommer svårigheter att upprätthålla en röd tråd genom informationssystemet och dess beteende. Han menar vidare att use case tillhandahåller denna röda tråd genom att de definierar informationssystemets beteende i olika situationer samt genom att samma use case används i analys, design, implementation och testning. Use case fungerar som ett hjälpmedel för en fungerande kommunikation mellan användare och systemutvecklare men förenar också kraven med design av informationssystemet (Kruchten, 2000). Vidare beskriver Kruchten (2000) att use case via designen används i implementationen och att use case används för att identifiera testfall då informationssystemet är implementerat. Han anser också att det är viktigt att använda en effektiv teknik för att förstå och modellera problem i utvecklingsprocessen och han menar att use case är en sådan teknik. Use case är möjligt att förstå för såväl användare som utvecklare av informationssystemet (Kruchten 2000). Karlsson (1996) beskriver use case som ett mycket tilltalande sätt att hantera krav på grund av dess enkelhet och kraftfullhet. Även Allen och Frost (1998) anser att styrkan med use case är dess enkelhet som ger användare och utvecklare möjlighet att förstå både varandra och kraven utifrån ett gemensamt synsätt. De menar vidare att även om use case diagram kan verka obetydliga är de mycket användbara i diskussioner kring vilka aktörer som är ansvariga i interaktion med informationssystemet. Eftersom use case-tekniken fokuserar på användarnas mål med informationssystemet så möjliggör detta ett sätt att avgöra vilka funktioner som är nödvändiga för informationssystemet, i motsats till att utveckla specifika funktioner och hoppas på att de passar användarna (Wiegers, 1997).

Use case är en relativt enkel teknik, en del anser den vara alltför enkel. Lee & Xue (1999) anser att use case brister i formalitet, struktur och hantering av stora use case-modeller. Regnell (1999) nämner bristen på sammanhang som den största nackdelen med use case. Han menar att det finns risk att use casen ses som en mängd 'lösa delar' och att kopplingen mellan olika use case går förlorad. Ett annat problem som Regnell (1999) berör är att om den verksamhet som modelleras är mycket komplex så leder det till en mycket stor mängd use case och att det då kan förekomma inkonsistens mellan olika use casen. Även Tuok & Logrippo (1998) beskriver att det finns en del problem i användandet av use case, främst vad gäller att säkerställa konsistens genom hela systemutvecklingsprocessen samt att kraven blir kompletta. Cockburn (2001) betonar att use case endast representerar en del av alla krav som ställs på ett informationssystem, även om use case är en central del i RE-processen så beskriver use case endast olika beteenden och innefattar till exempel inte prestanda, affärsregler eller användargränssnitt. Karlsson (1996) påpekar att use case inte är lika effektiva i hanteringen av ickefunktionella krav som i hanteringen av funktionella krav. Use case borde vara mer effektivt att använda i hanteringen av funktionella krav än ickefunktionella krav, då use case främst beskriver vad ett informationssystem ska göra och inte i samma grad beskriver till exempel kvalitet och effektivitet.

För en sammanfattning av möjligheter och begränsningar med use case-tekniken se tabell 1, nedan.

Tabell 1. Sammanställning av möjligheter och begränsningar med use case-tekniken

	Möjligheter	**Begränsningar**
Funktionella krav	• Fångar upp funktionella krav (Kruchten, 2000; Maciaszek, 2001; Fowler & Scott, 2000; Karlsson, 1996)	
Ickefunktionella krav	• Fångar upp ickefunktionella krav (Kulak & Guiney, 2000)	• Fångar ej upp ickefunktionella krav (Kruchten, 2000; Maciaszek, 2001, Karlsson, 1996; Cockburn, 2001)
Kommunikation	• Stöder kommunikation mellan användare och systemutvecklare (Kruchten, 2000; Karlsson, 1996; Allen & Frost, 1998)	
Enkelhet	• Enkelheten som styrka (Karlsson, 1996; Allen & Frost, 1998)	• Enkelheten som svaghet (Lee & Xue, 1999; Regnell, 1999)
Komplexitet	• Hanterar komplexitet väl, till en gräns (Regnell, 1999; Lee & Xue, 1999)	• Hanterar ej mycket komplexa verksamheter (Regnell, 1999; Lee & Xue, 1999)

Åsikterna går isär vad gäller om use case-tekniken fångar upp ickefunktionella krav. Dock anser en majoritet inom litteraturen att use case-tekniken inte fångar upp ickefunktionella krav, vilket tyder på att så är fallet. Enkelheten med use case-tekniken anses vara både en möjlighet och en begränsning. Det bör vara en klar möjlighet då det gäller att lära in tekniken och använda men kanske enkelheten gör att tekniken inte fungerar i stora, komplexa miljöer.

Use case är en teknik som är på framfart i systemutvecklingskretsar. Större delen av den litteratur som författaren kommit i kontakt med vid bakgrundsstudierna till detta examensarbete är övervägande positiv till användandet av use case-tekniken vid kravhantering. Dock finns det både möjligheter och begränsningar med att använda denna teknik. Eftersom use case har en sådan ökad användning är det viktigt att belysa, inte bara möjligheter, utan även begränsningar med tekniken. Problembeskrivningen i nästa kapitel kommer att ytterligare poängtera detta.

3 Problembeskrivning

I detta kapitel presenteras en beskrivning av projektets problemområde samt den problemställning som detta examensarbete har som utgångspunkt. Fortsättningsvis i kapitlet kommer även de avgränsningar som gjorts för området att presenteras och därefter framställs förväntat resultat av undersökningen.

3.1 Problemområde

Objektorienterad systemutveckling används idag i allt större utsträckning (Karlsson, 1996). Detta faktum tillsammans med det ökande kravet på informationssystem vad gäller ständigt ökande storlek, omfattning och komplexitet har lett till utvecklandet av olika modelleringsspråk som ger möjlighet att modellera funktionella krav på en konceptuell nivå (Loucopoulos & Karakostas, 1995). Fowler och Scott (2000) påstår att en av de största utmaningarna i systemutveckling idag är att bygga *rätt* informationssystem, det vill säga ett informationssystem som uppfyller användarnas behov till en rimlig kostnad. De menar att detta är komplicerat då det kräver en god kommunikation mellan utvecklare och användare. Krav som inte hanteras på ett riktigt sätt är en stor anledning till misslyckanden inom systemutveckling. Saiedian och Dale (2000) påpekar att utan en välskriven kravspecifikation vet inte systemutvecklarna vad de ska utveckla, användarna vet inte vad de ska förvänta sig och det finns ingen möjlighet att validera att det färdiga informationssystemet möter de behov som fanns hos användarna från början. Dataindustrin har länge försökt att hitta ett sätt att representera funktionella krav på ett sätt som användare kan ta till sig (Kulak och Guiney, 2000). Det är viktigt att systemutvecklare förstår de behov som användarna uttrycker för informationssystemet och det är tvärtom också viktigt att användare förstår de krav som systemutvecklarna sedan framställer. Loucopoulos och Karakostas (1995) menar att användandet av grafiska modelleringsspråk ökar förståelsen för problemdomänen och möjliggör en väl fungerande kommunikation mellan systemutvecklare och användarna. Det finns många olika intressenter till ett informationssystem och dessa olika intressenter har olika sätt att uttrycka krav på. Vissa kanske använder naturligt språk, ritar bilder eller använder annan typ av grafik (Pohl, 1996). Att finna en teknik för representation av krav som kan användas och förstås av alla typer av intressenter vore värdefullt för systemutveckling i framtiden.

Ett modelleringsspråk som utvecklats av ovanstående anledningar är UML. UML har blivit modelleringsstandard för objektorienterad systemutveckling. En av de tekniker inom UML som kan användas för att modellera funktionella krav är use case. Booch, *et al*. (1999) beskriver hur ett informationssystem interagerar med olika aktörer, vilka använder informationssystemet för ett visst syfte och har vissa förväntningar på hur informationssystemet ska agera. De anser vidare att use case är en lämplig teknik för att specificera hur systemet agerar, systemets beteende. Karlsson (1996) förutspår att användandet av use case för att samla in, beskriva och validera krav kommer att öka framöver. Då use case är en teknik som anses vara lätt att använda och förstå kanske denna teknik kan användas av alla inblandade intressenter som en gemensam grund för förståelse. Ovanstående förutsättningar leder fram till den problemprecisering som presenteras i nästa avsnitt.

3.2 Problemprecisering

Då kravhantering får allt större betydelse i systemutvecklingen finns ett behov av att hitta tekniker som stödjer denna process. Användandet av UML och därmed också av use case har spridits till många företag och det förväntas att spridas ytterligare framöver. Det finns ett intresse av att se om denna teknik är tillräcklig för att hantera olika sorters krav i RE-processen. Om use case inte är en tillräcklig teknik för att hantera dessa olika sorters krav, så kompletteras eventuellt use case med andra tekniker. Utifrån detta är frågeställningen för detta examensarbete:

> *Är use case tillräcklig som enda teknik för att identifiera och dokumentera krav. Om inte; hur kompletterar systemutvecklare de former av krav som use case-tekniken inte fångar in?*

Avsikten är att undersöka om use case kan användas som enda teknik för att identifiera krav eller om det finns områden där use case-tekniken behöver kompletteras med någon annan teknik. Eftersom use case används i allt större utsträckning är det viktigt att belysa hur tekniken fungerar för att identifiera och dokumentera olika sorters krav under RE-processen.

3.3 Avgränsning

I detta examensarbete kommer endast use case som teknik för att identifiera och dokumentera krav att undersökas, use case som stöd för övriga aktiviteter i RE-processen kommer inte att behandlas. Andra användningsområden för use case-tekniken än ovan nämnda, såsom design och implementation, kommer inte att beröras i detta arbete.

3.4 Förväntat resultat

Förväntat resultat av undersökningen i detta examensarbete är att belysa möjligheter och begränsningar med att använda use case som enda teknik för att identifiera och dokumentera krav. Framför allt kommer eventuella generella begränsningar med use case-tekniken att belysas. Med stöd av den litteratur som presenterats i bakgrunden förväntas att begränsningar med use case-tekniken kommer att framkomma i undersökningen samt att use case-tekniken inte är tillräcklig som enda teknik för att identifiera och dokumentera krav. Det förväntas också framgå hur systemutvecklare löser detta och vilka eventuella tekniker som kan komplettera use case-tekniken.

4 Tillvägagångssätt

I detta kapitel presenteras möjliga tillvägagångssätt för att undersöka den frågeställning som ställs i detta examensarbete. Tre metoder har ansetts som möjliga att använda var för sig eller i kombination med varandra och dessa metoder presenteras närmare i nästa avsnitt.

4.1 Möjliga ansatser

Det finns flera olika sätt att samla information som behövs för att få svar på de frågeställningar som tas upp i ett examensarbete. Enligt Patel och Davidson (1994) beror val av teknik att använda i undersökningen på vad som kan ge ett optimalt svar på frågeställningen. De påpekar vidare att det också är viktigt att ta hänsyn till den tid och de medel som finns tillgängligt för undersökningen, till exempel finansiella och personella medel. Viktigt att ta hänsyn till vid val av metod är också undersökarens egna forskningserfarenheter, olika metoder kan vara lämpliga beroende på om undersökaren är erfaren eller ej. Nedan presenteras tre metoder som anses möjliga att använda för att undersöka detta examensarbetes problemställning:

> *Är use case tillräcklig som enda teknik för att identifiera och dokumentera krav. Om inte; hur kompletterar systemutvecklare de former av krav som use case-tekniken inte fångar in?*

Eilertsson (1996) menar att två datainsamlingsmetoder kan särskiljas då inskaffande av information sker genom att människor själva aktivt besvarar frågor, dessa är intervju och enkät. Av dessa två metoder har enkäter valts bort då enkäter kräver korta och precisa frågor och det inte anses att denna form av frågor kan ge tillfredsställande svar för problempreciseringen. Istället anses att redogörande frågor krävs för att få svar på problempreciseringen vilket är svårt att åstadkomma i en enkät.

De metoder som är möjliga att använda i detta arbete för att undersöka problemställningen kommer att presenteras i kommande avsnitt och är följande:

- Intervjuer
- Fallstudier
- Litteraturstudier

4.1.1 Intervju

En intervju är en metod för att samla information. Enligt Patel och Davidson (1994) bygger denna metod på frågor. En intervju kan vara personlig, det vill säga att intervjuaren träffar respondenten, men den kan också utföras via ett telefonsamtal. Båda dessa former av intervju kan anses lämpliga för detta arbete. Detta på grund av att redogörande frågor kan ställas och en diskussion kring frågorna kan föras på ett lämpligt sätt. Respondenten ges också möjlighet att ställa frågor till intervjuaren om något är oklart och intervjuaren får möjlighet att ställa följdfrågor som kommer upp under intervjuns gång.

Patel och Davidson (1994) beskriver att det finns två aspekter att beakta vid insamlandet av information med hjälp av intervju, dessa är *standardisering* och *strukturering*. Graden av standardisering beror på hur mycket ansvar intervjuaren har vid utformning av frågor. Vid låg grad av standardisering formuleras frågorna under själva intervjun och ställs i den ordning som är lämplig för den enskilda respondenten. Vid hög grad av standardisering ställs samma frågor i exakt samma ordning till varje respondent. Graden av strukturering beror på det svarsutrymme och tolkningsmöjlighet som respondenten ges menar Patel och Davidson (1994). En strukturerad intervju lämnar mycket litet svarsutrymme och det går att förutsäga vilka alternativa svar som är möjliga. En helt ostrukturerad intervju lämnar ett maximalt utrymme för respondenten att svara inom, det vill säga respondenten kan svara helt fritt och styrs inte av några svarsalternativ.

I detta examensarbete bör det underlätta för en oerfaren intervjuare om det finns en viss standardisering av intervjufrågorna så att intervjuaren får en ram att hålla intervjun inom. Standardisering av intervjufrågorna bör också ge struktur i analyserandet av svaren. I och med att varje respondent fått frågor kring samma ram blir också analys av svaren enklare att genomföra. Dock kan det krävas en viss domänkunskap för att formulera rätt frågor, varför det bör vara möjligt att tillåta att nya frågor framkommer under intervjuernas gång och att flexibilitet i ordning av frågorna också kan finnas. För att erhålla så utförliga svar som möjlig bör ett stort svarsutrymme lämnas för respondenterna så att de kan tala fritt kring frågorna

Enligt Ejlertsson (1996) kan respondenten ställa frågor till intervjuaren om denne känner osäkerhet kring vissa frågor. Missuppfattningar kan härmed undvikas. En intervjuundersökning ger också möjlighet till att mer komplicerade frågor kan ställas och frågorna i detta examensarbete är av sådan karaktär. Ejlertsson (1996) menar vidare att vid intervjuer kan en öppen diskussion hållas där svaren inte kan förutsägas, detta ansågs lämpligt för arbetets problemställning. I och med att en öppen diskussion hålls kan också följdfrågor ställas vilket talade för att intervjuer skulle väljas. Följdfrågor kan vara nödvändiga för att klargöra respondentens svar eller för att få fram mer information under intervjuns gång. Intervjuer ansågs även lämpliga i och med att redogörande frågor avsågs att ställas och intervjuer ger ett stort svarsutrymme vilket krävs vid denna form av frågor för att erhålla så utförliga svar som möjligt.

4.1.2 Fallstudie

En fallstudie innebär enligt Patel och Davidson (1994) att en undersökning genomförs exempelvis i form av intervjuer eller observationer, på en mindre avgränsad grupp och inte på ett stort antal personer. Det kan röra sig om en individ, en grupp individer, en organisation eller en situation. Fallstudier är lämpliga att använda för att studera processer och förändringar (Patel & Davidson, 1994). Dock påpekar Shaughnessy och Zechmeister (1997) att fallstudier kan se mycket annorlunda ut och ge material som är ett par sidor långt eller som kan fylla en bok.

En fallstudie kan göras direkt eller indirekt. Vid en direkt fallstudie tas hjälp av till exempel intervjuer eller observationer och vid indirekt fallstudie studeras ett företags dokumentation (Dawson, 2000). För att genomföra en fallstudie krävs en viss domänkunskap för att kunna tillgodogöra sig den information som framkommer. Det kan vara svårt inom ramen för examensarbetet att erhålla sådan domänkunskap. Shaughnessy och Zechmeister (1997) menar också att det är svårt att generalisera information som framkommit ur en enda fallstudie. Detta bör innebära att analysarbetet med den data som framkommer i en fallstudie kan vara svårare att genomföra för att det ska tillföra ny värdefull information till utredningen då det kräver mer kunskap hos undersökaren.

Att utföra en fallstudie för detta examensarbete vore berikande eftersom det skulle ge en mer praktisk kunskap och erfarenhet om hur arbetsprocessen vid kravhanteringsinsamlandet med use case utförs. Att observera hur ett företag arbetar praktiskt kan ge en annan form av information än genom att endast använda intervjuer i och med att undersökaren får ”first hand experience”.

4.1.3 Litteraturstudie

Enligt Patel och Davidson (1994) är de vanligaste källorna till kunskap böcker, vetenskapliga artiklar samt rapporter. I böcker finns utarbetade teorier och modeller och i artiklar och rapporter finns de allra senaste rönen. I kapitel 2, bakgrund har en litteraturstudie gjorts för att skapa en förståelse för det aktuella problemområdet. Denna litteraturstudie kan dock utökas under arbetets gång för att till exempel undersöka trender och nya rön som kan framkomma.

Allt eftersom arbetet med examensarbetet fortskrider erhålls mer kunskap och kanske väcks också fler frågor och funderingar kring problemområdet. Av dessa anledningar finns det anledning att fortlöpande underhålla litteraturstudierna för att utöka den kunskap och få svar på de frågor som arbetet med examensarbetet leder till.

4.1.4 Summering möjliga ansatser

Tre metoder har ansetts vara lämpliga att användas i kombination med varandra i undersökningen av examensarbetets problemställning. Kombinationen anses fördelaktig för att kunna få fram så fullständig information som möjligt. Intervjuer kan användas för att få information om hur systemutvecklare arbetar med use case i samband med kravhantering och vad deras åsikter kring detta är, en fallstudie kan bidra med en mer praktisk kunskap om hur use case används och en litteraturstudie kan ge en ökad förståelse kring problemställningen och bidra med nya rön och trender som framkommit.

4.2 Arbetsprocessen

Nedan presenteras den ansats som använts vid genomförandet av undersökningen till detta examensarbete.

Eftersom denna undersökning förväntades resultera i att kunna sammanställa och i viss mån jämföra resultaten planerades intervjuerna att ha en viss grad av standardisering. Det planerades att färdiga frågor skulle finnas förberedda inför intervjuerna men att det under intervjuerna skulle finnas utrymme för eventuellt nya frågor och följdfrågor. Det skulle också vara möjligt att under intervjuernas gång ändra ordningen på frågorna utefter hur respondenterna svarar. Intervjufrågorna formulerades relativt ostrukturerat, det vill säga att respondenterna skulle få så stort svarsutrymme som möjligt för att ej begränsa deras svar. För att få svar på problemställningen krävdes utförliga frågor med stort svarsutrymme, endast ja- och nej-frågor ansågs inte vara tillräckligt för att få ett tillfredsställande resultat. Intervjuerna i detta projekt avsågs vara personliga, det vill säga intervjuaren träffar respondenten och genomför intervjun istället för att de skulle utföras per telefon. Detta för att det ansågs ge bättre och mer utförliga svar vid ett direkt möte med respondenterna då diskussionerna kan föras mer öppet vid en personlig kontakt. Det kan också vara så att respondenterna tar sig mer tid att svara på frågorna vid en personlig intervju, en telefonintervju kan upplevas ansträngande och respondenterna kanske inte tar sig tid att svara på frågorna lika utförligt. Det kan vara lätt att känna ett mindre engagemang i en telefonintervju än vid en personlig intervju.

För att anförskaffa lämpliga intervjupersoner planerades att kontakta ett antal företag i Skövderegionen. Att inskränka sig till en sådan begränsad region kan vara en nackdel för undersökningen på grund av att detta leder till ett begränsat urval av respondenter och kanske inte de mest lämpliga respondenterna kan väljas. För att möjliggöra den önskade personliga kontakten och för att kunna utföra intervjuerna på plats fattades ändå beslutet att avgränsa det geografiska området. Det planerades att mellan 5 och 10 intervjupersoner skulle vara ett lämpligt antal för detta arbete eftersom intervjuerna avsågs vara av djupintervjukaraktär och därmed begränsades det möjliga antalet intervjuer, på grund av tids- och resursramen för examensarbetet.

Initialt kontaktades fyra olika systemutvecklingsföretag i Skövderegionen. Dessa företag kontaktades via e-post samt med telefonsamtal. Av de fyra företagen var det endast två som arbetar regelbundet med use case och som var intresserade av att ställa upp med intervjudeltagare. Av de två företag som var intresserade erhölls sammanlagt sju namn på personer att kontakta vidare för att boka in intervjuer. Ett par veckor innan intervjuerna planerats att genomföras började kontakter tas via telefon med de personer som namngetts som möjliga intervjupersoner. Fyra intervjuer kunde bokas in relativt snart per telefon, men två personer kontaktades, efter upprepade kontaktförsök per telefon, istället per e-post. Även dessa två kunde sedan bokas in via e-post.

En av de tänkta intervjupersonerna hade långsemester och kunde därför inte intervjuas inom den aktuella tidsperioden. Detta innebar att det totala antalet respondenter slutligen blev sex stycken, vilket utefter de tids- och resursramar som var aktuella för examensarbetet samt av intervjuernas karaktär, ansågs som ett lämpligt antal. En presentation av dessa sex intervjudeltagare ges i kapitel 4.3 Intervjudeltagare.

Problempreciseringen, om use case kan användas som enda teknik för att identifiera och dokumentera krav, var utgångspunkt för utformandet av intervjufrågorna, (se bilaga 1). Intervjufrågorna formulerades att inte vara ledande men att ändå ge en viss vägledning i frågeställandet. De första tre frågorna fungerar som inledande frågor, detta för att kartlägga varje respondents erfarenhet, både av kravhantering och användande av use case, men också för att skapa en bra stämning. Trost (1993) menar att de första frågorna som ställs kan vara avgörande för hur resten av intervjun kommer att fortlöpa. På grund av detta kan det vara lämpligt att ställa allmänna frågor som inte direkt har med respondentens kunskap att göra. Frågorna fyra till sju samt tio och elva är huvudfrågor, vilka formulerats för att få svar på problemställningen. Detta innefattade både att få reda på hur respondenterna ser på kravhantering och på use case-tekniken, vad de anser vara styrkor och svagheter med use case-tekniken samt hur denna används. Frågorna 13 och 14 formulerades för att få ytterligare information om processen i arbetet med use case samt för att få en bild av hur mycket samarbete som förekommer med användarna. Frågorna åtta, nio och tolv är följdfrågor som skall ställas om visst svar ges på föregående fråga, (se bilaga 1) för detaljer. Avslutningsvis formulerades fråga 15 för att ge respondenten en möjlighet att tala fritt kring sin egen åsikt om kravhantering och use case på ett sätt som inte täckts upp av redan ställda frågor. Detta ansågs lämpligt eftersom alla människor har olika erfarenheter och upplevelser och det är svårt att formulera frågor som täcker alla aspekter för de olika respondenterna.

Innan intervjuerna genomfördes lästes intervjufrågorna igenom av en utomstående person med domänkunskap i området för att försöka säkerställa att frågeformuleringarna var tydliga och att inte några uppenbara missuppfattningar av frågorna skulle kunna uppstå. Det gjordes dock inte någon formell pilotintervju dels på grund av tidsramarna för examensarbetet men också på grund av att intervjuerna avsågs vara relativt ostrukturerade och det då beror mycket på varje enskild respondent hur intervjun blir. Om det under intervjuernas gång skulle framkomma nödvändiga förändringar av intervjufrågorna var detta möjligt att genomföra innan nästa intervjutillfälle.

Det beslutades att intervjuerna skulle bandas om inte någon respondent motsatte sig detta, till exempel på grund av att de upplever obehag av att bandas eller liknande. Trost (1993) påpekar att många människor inte vill bli inspelade på band. Oftast accepterar de att bli inspelade men kan uppleva det besvärande och hämmande. Han menar vidare dock att de flesta vänjer sig vid bandspelaren och glömmer att de blir inspelade. Anledningen till att använda bandspelare var att inte missa värdefull information som framkom under intervjuerna. Att ta anteckningar vid de redogörande frågor som avsågs att ställas ansågs svårt att genomföra med tanke på den omfattning de förväntade svaren skulle inneha. Dessutom ansågs intervjuerna bli mer tidskrävande på grund av att antecknandet skulle förlänga tiden mellan frågorna och därmed kan flödet i kommunikationen förloras.

Diskussionerna med respondenterna ansågs kunna utföras på ett mer avslappnat sätt med hjälp av bandspelare än om anteckningar skulle tas.

Innan intervjuerna genomfördes tillfrågades båda de företag där intervjuerna skulle utföras om det fanns någon möjlighet att få utföra en fallstudie eller liknande för att erhålla mer praktisk kunskap kring användandet av use case. Dock fanns inga lämpliga projekt för tillfället varför ingen fallstudie kunde utföras. Detta innebar att en viss information förlorades som skulle ha kunnat ge en stor insikt i hur det praktiska arbetet fungerar i en verksamhet.

Fortsatta litteraturstudier planerades att utföras under hela det fortsatta arbetet med detta examensarbete. Främst avsågs artiklar kring nya rön och trender att studeras men även en del litteratur för att ytterligare bygga på den befintliga bakgrunden. Detta har genomförts genom sökningar i olika databaser, på bibliotek samt i olika tidskrifter, både av forskningskaraktär samt mer kommersiella tidskrifter.

4.3 Intervjudeltagare

Nedan ges en kort beskrivning av de två företag vars anställda medverkat i undersökningen samt de respondenter som deltagit i undersökningen. Detta för att ge en bild av vilken form av verksamhet som medverkat i undersökningen samt den erfarenhet de olika respondenterna har av kravhantering med use case. Företagen är stora, internationella företag med verksamhet runt om i hela världen. Dock har endast anställda vid respektive Skövde-kontor intervjuats. Informationen om de båda företagen har inhämtats vid genomförda intervjuer med anställda på företagen.

Företag A har 30-tal anställda på Skövde-kontoret och företaget har funnits i Skövde sedan 1998. Företaget arbetar uteslutande med programvaruutveckling i olika stadier för radarer (markradar, nosradar och flygradar) och simulatorer, främst mot Försvaret. Företaget har en högteknologisk inriktning på sin verksamhet och sina produkter. Det finns två enheter på företaget. Den ena enheten arbetar med kravhantering, projektledning och systemutveckling, medan den andra enheten arbetar med konstruktion och programmering av olika system och programvaror. Företaget fungerar ofta som resurs internt för huvudkontoret. Företaget arbetar främst i projektform, ofta ingår lokala projektgrupper i stora projekt på över 100 personer.

Företag B har 130 anställda i Skövde och har funnits i nuvarande form sedan 1999[5]. Företaget arbetar med datordrift, data- och telekommunikation, utveckling av metoder, system och teknik inom en rad specialistområden. Företag B utvecklar och förvaltar olika typer av system, främst fabriksnära produktions- och materialsystem men även ekonomisystem och andra administrativa system, vilka tillverkas för verkstadsindustrimiljö. Företaget erbjuder dock även andra tjänster av supporttyp till sina kunder såsom projektledning av IT-projekt, webbdesign och utveckling av verksamhetsstöd. Företaget ansvarar också för projektering, uppbyggnad och övervakning av sina kunders IT-infrastruktur såsom nätverk, PC, terminaler, skrivare och telefonväxel. Till avdelningen hör också en helpdesk dit kunderna kan vända sig vid problem. Företaget arbetar främst i projektform där projektgrupperna består av 8-10 personer, men ibland kan ett litet deluppdrag innebära att man arbetar ensam. Tabell 2 nedan, visar en sammanställning av de båda företagen.

Tabell 2. Sammanställning de aktuella företagen

Företag	**A**	**B**
I Skövde sedan	1998	1999[5]
Antal anställda	30	130
Arbetsuppgifter	Programvaruutveckling Kravhantering Programmering mm	System-utveckling Projektledning Helpdesk mm

De båda företagen är relativt unga i Skövde, men är stora internationella företag. Antalet anställda skiljer ganska mycket på de båda företagen och dess huvudarbetsuppgifter skiljer sig åt till viss del. Det ena fokuserar på utveckling av programvaror medan det andra stödjer processen vid en verkstadsindustri på olika sätt. Det är dock två mycket framgångsrika och välkända företag som arbetat en längre tid med use case-tekniken varför dessa företag anses lämpliga för examensarbetets undersökning.

De respondenter som medverkat i undersökningen har tilldelats av respektive företag. De kriterier som utgivits var endast erfarenhet av use case-tekniken och därefter har förtroende getts till respektive företag att välja ut lämpliga personer. Förhoppningen var att företagen skulle välja ut personer utefter deras kunskap och erfarenhet av use case-tekniken och inte utefter deras tillgänglighet för att intervjuas. Kanske hade detta kunnat säkerställas på ett bättre sätt om undersökaren själv hade haft möjlighet att välja respondenter. Dock uppfattades urvalet av respondenter som ett väl genomfört urval. En kortfattad presentation av respondenternas erfarenhet samt arbetsuppgifter ges nedan.

[5] Verksamheten har funnits i över tio år men har först 1999 blivit ett eget bolag

Respondent 1 har varit anställd på företag A sedan juni 1999 och har erfarenhet av systemutveckling under samma tidsperiod, dock ser respondenten sina arbetsuppgifter som programvaruutveckling då arbetet endast gäller delar av system och inte hela system eftersom projekten är mycket stora. Under ett års tid har respondenten arbetat med kravhantering. Respondenten har sedan 1999 arbetat med use case i någon form men först på senare tid mer formellt.

Respondent 2 har varit anställd på företag A sedan 1999 och har erfarenhet av systemutveckling under samma tidsperiod. Respondenten har arbetat med use case sedan tre år tillbaka men har modellerat själv de två senaste åren. Respondenten arbetar med kravhantering i MDI-sammanhang (människa-dator-interaktion), i detta fall gäller det främst gränssnittsfrågor.

Respondent 3 har varit anställd på företag B sedan 1989 och har erfarenhet av systemutveckling under samma tidsperiod. Respondenten har arbetat med use case sedan 1999. Respondentens arbetsuppgifter är att vara metodstöd och metodutvecklare vilket bland annat innebär att anpassa metoder så att de passar lokalt samt att samordna metodfrågor ”uppåt”. Respondenten ansvarar även för kompetensutveckling inom metoder.

Respondent 4 har varit anställd på företag B sedan 1990 och har arbetat med systemutveckling under samma tidsperiod. Respondenten är systemutvecklare och arbetar med allt från utredning till programmering och testning samt en del projektledning. Respondenten har arbetat med use case sedan två år tillbaka.

Respondent 5 har varit anställd på företag B sedan augusti 1999. Respondenten arbetar som systemutvecklare. Arbetsuppgifterna innebär främst förvaltning och vidareutveckling av system. Respondenten har erfarenhet av systemutveckling sedan 1999. Respondenten jobbar för tillfället inte med use case men har praktisk erfarenhet av att arbeta med use case från ett några månader långt projekt, utöver den teoretiska kunskapen.

Respondent 6 har arbetat på företag B i snart sex år. Respondenten är systemutvecklare och jobbar främst som systemanalytiker och inte så mycket med programmering och design utan mer i de tidiga faserna med analys. I vissa projekt har respondenten rollen som verksamhetsutvecklare. Respondenten har haft erfarenhet av systemutveckling i elva år, med skolan inräknad är det 14 år. Respondenten har arbetat med use case i drygt 4 år. I tabell 3 nedan, presenteras en sammanställning av respondenterna.

Tabell 3. Sammanställning av respondenter

Respondent	1	2	3	4	5	6
Företag	A	A	B	B	B	B
Anställd sedan	1999	1999	1989	1990	1999	1993
Erfarenhet inom system-utveckling	3 år	3 år	13 år	12 år	3 år	11 år
Erfarenhet av use case	3 år	3 år	3 år	2 år	Ca 3 mån	4 år

Tre av respondenterna har lång erfarenhet av systemutveckling och de övriga tre är relativt nya inom yrket. Detta var en lämplig blandning av intervjupersoner eftersom det är intressant att få åsikter både från erfarna systemutvecklare såväl som mindre erfarna systemutvecklare. Erfarna systemutvecklare har under sina yrkesverksamma år lärt av erfarenheten och kan därmed bidra med rutin och kunskap. Det kan dock finnas en viss risk att de är ”fast i gamla vanor” och därmed inte är lika öppna som oerfarna systemutvecklare. Oerfarna systemutvecklare kan vara öppna för nya idéer och tekniker men har kanske inte den rutin som kanske krävs för att bedöma det nya. De flesta av respondenterna har arbetat ungefär lika länge med use case, detta beror troligtvis på att use case är en relativt ny teknik som blivit vanlig som standard först på senare år.

4.4 Genomförande samt reflektioner på arbetsprocessen

Under undersökningen av examensarbetets frågeställning har sex djupintervjuer genomförts med systemutvecklare från skilda projekt eller olika företag och med varierande lång erfarenhet av såväl systemutveckling som användande av use case-tekniken. Samtliga intervjuer har genomförts som personliga intervjuer.

De sex intervjuerna utfördes under cirka en och en halv veckas tid. Att intervjua upplevdes som en svår och ovan situation och eftersom det var en form av djupintervjuer som utfördes ställde detta höga krav på intervjuaren, i detta fall författaren. Rollen som intervjuare var ovan men de flesta respondenterna var dock intresserade av frågeställningen och det gick lätt att föra en bra diskussion med dem. Två av respondenterna var mycket kortfattade i sina svar och det var då svårt att avgöra hur mycket påverkan och ledning som var lämpligt utan att ”lägga orden i munnen” på respondenten.

Intervjuerna gav dock mycket värdefull information, både för examensarbetet och för författaren i sin blivande yrkesroll. De gav inte bara en inblick i hur use case-tekniken används utan även hur systemutveckling i allmänhet bedrivs rent praktiskt.

Intervjuerna dokumenterades med hjälp av bandspelare. Samtliga respondenter var tillfrågade innan intervjun om de gav tillåtelse till att bandspelare skulle användas och ingen respondent motsatte sig detta. Intervjuerna var enkla att genomföra med bandspelare då full uppmärksamhet kan riktas på respondenten och diskussionen med denne istället för att behöva lägga uppmärksamhet på att anteckna under hela intervjun. Det blir en situation som främjar en fri diskussion. Ingen av respondenterna verkade störas eller hämmas av bandspelaren. Detta är dock enligt författarens uppfattning och kan naturligtvis inte tas för givet. Det finns en möjlighet att vissa respondenter kan ha upplevt bandspelaren som besvärande.

Intervjufrågorna visade sig vara väl formulerade eftersom de upplevdes att ge de svar som efterfrågats. Respondenterna hade inga problem med att förstå frågorna och det var inte heller några problem att ställa följdfrågor som passade in i sammanhanget. I en intervju glömdes en fråga av att ställas men intervjun gav ändå tillräckligt bra underlag att bearbeta och i gengäld uppkom en del nya frågor som gav ytterligare information. Det insamlade intervjumaterialet har inte validerats med intervjupersonerna, detta på grund av att renskrivning av de bandade intervjuerna gjorts i princip ordagrant. Ingen av respondenterna har heller efterfrågat validering av vad de yttrat.

Bemötandet hos de olika respondenterna upplevdes i stort mycket väl. Respondenterna var intresserade av problemställningen och delade gärna med sig av sin kunskap och erfarenhet inom området. Detta medförde att rollen som intervjuaren blev lättare att hantera och intervjuerna blev mer diskussion än utfrågning. En av respondenterna var mycket kort i sina svar och upplevdes inte ha en positiv inställning till att bli intervjuad. Vad detta beror på är svårt att avgöra, kanske respondenten inte trivs i intervjusituationer eller kanske inte intresse för frågeställningen fanns. Eventuellt kan respondenten ha upplevt sig personligen utfrågad angående sin kunskap vilket kan ha upplevts negativt. Kanske en mer utförlig beskrivning av intervjuns syfte och hur intervjusvaren skulle användas kunde ha hjälpt denna situation. Dock kan det också vara så att respondenten helt enkelt ”hade en dålig dag”.

Intervjuerna anses av författaren att har gett svar på problemställningen, se vidare i kapitel 5, resultat och analys. Intervjusvaren kan anses som pålitliga eftersom berörda respondenter svarat utifrån egen erfarenhet i praktiska situationer. Dock kan sex intervjuer ses som ett för litet underlag för att generalisera svaren att gälla i alla situationer. Eftersom intervjuerna utfördes som djupintervjuer och med hänsyn till tidsramarna för examensarbetet anses dock sex intervjuer som ett fullgott underlag för denna undersökning.

I och med att inte någon fallstudie kunde utföras blev den kunskap som erhölls på ett mer teoretiskt plan än om det hade varit möjligt att under en tid följa hela arbetsprocessen och användningen av use case-tekniken och erhålla ”first hand experience”. Den data som samlats in är respondenternas erfarenhet och inte författarens egen erfarenhet. Detta har inneburit att en del av den metodkombination som planerats har gått förlorad, dock anser jag att intervjuerna lett till ett tillfredsställande svar på problempreciseringen men att en fallstudie hade kunnat bidra till helheten på ett annat sätt. En av respondenterna erbjöd sig dock att visa hur denne arbetade med use case och hur de datorstödda verktygen som användes fungerade, vilket ändå gav lite mer praktisk inblick i arbetet med use case. Detta gav en större förståelse för hur arbetet med use case faktiskt sker och detta upplevdes som mycket positivt av författaren.

Under arbetets gång har ny litteratur införskaffats genom bland annat sökning via databaser och bibliotek efter artiklar av olika slag. Även en del litteratur som tidigare inte varit tillgänglig har införskaffats. Även samtal med olika personer på Högskolan i Skövde med kunskap om aktuellt område har genomförts vilket ytterligare berikat arbetet och givit författaren en större förståelse för problemområdets komplexitet

.

5 Resultat och analys

I detta kapitel presenteras resultatet av de intervjuer som utförts samt en analys av de svar som framkommit på examensarbetets problemställning;

> *Är use case tillräcklig som enda teknik för att identifiera och dokumentera krav. Om inte; hur kompletterar systemutvecklare de former av krav som use case-tekniken inte fångar in?*

De bandade intervjuerna har transkriberats i det närmaste ordagrant, dock har upprepningar samt ”hummanden” och liknande utelämnats. Transkriberingarna kommer ej att bifogas som bilagor till examensarbetet men kan lämnas ut på förfrågan. Efter transkriberingen av intervjuerna har intervjumaterialet kodats för att möjliggöra en sammanställning. Likheter och skillnader mellan olika respondenters svar har också kunnat lyftas fram på detta sätt och detta har möjliggjort analys av orsaker därtill.

Resultaten presenteras inte i intervjufrågornas ordningsföljd (se bilaga 1), istället presenteras resultaten genom att vissa frågor slagits samman med hänsyn till deras innehåll och samband. Anledningen till detta är att ge läsaren en röd tråd genom kapitlet och hålla samman information som hör ihop. Att intervjufrågorna ställdes i den ordning som gjordes var på grund av att inte leda respondenterna i någon viss riktning i början av intervjun. Även det som framkommit av de frågor som ställts, som inte har direkt koppling till problempreciseringen, kommer att presenteras då dessa frågor skapar en förståelse för arbetsprocessen med use case-tekniken. Eftersom intervjuerna genomförts på ett ostrukturerat sätt med enbart viss standardisering är ordningsföljden av intervjufrågorna inte av avgörande betydelse. De aktuella frågorna kommer också att redogöras för att förtydliga presentationen.

De första tre frågorna fungerar som inledande frågor som syftar till att kartlägga varje respondets erfarenhet, både av kravhantering och användande av use case, och inte i första hand att bidra med material till undersökningen. Dessa frågor kommer på grund av detta ej att presenteras här då en närmare presentation av de inledande frågorna finns i kapitel 4.2.2 Respondenter.

5.2 Representation av use case

Nedanstående kapitel innefattar en redogörelse för de frågor som är relaterade till representation av use case.

- *”Föredrar du att representera kraven i textform eller i use case-diagram? Varför?”*

Alla respondenter utom respondent 2 använde en kombination av use case i textform och i diagramform. Respondent 2 använde endast textform på grund av att det var detta sätt som hade bestämts i projektet. Dock ansåg samtliga respondenter att en kombination av de båda presentationsformerna är optimalt. Skälet till detta är att diagrammen anses visa för få detaljer och textformen kan i vissa sammanhang vara för detaljerad och i dessa fall kan ett diagram ge en bättre överblick. Respondent 6 menade:

> *”För att få en blick över funktionerna och det som krävs av systemet så är det ju bättre att ha en bild över det … när man dokumenterar exakt vad use casen ska utföra för användarna då är det ju lite jobbigare att beskriva och då blir det ju text.”*

Use case i textform och i diagramform kompletterar varandra och use case-tekniken anses fungera på bästa sätt då båda dessa representationsformer används. För att få en översikt över vad systemet ska åstadkomma och för att föra diskussioner med användare passar diagrammen bättre, men för systemutvecklarna krävs mer detaljer och textbaserade use case blir nödvändiga. Cockburn (2001) och Fowler och Scott (2000) menar att ett use case-diagram inte är en nödvändighet utan endast en illustration för att hjälpa till att förstå det textbaserade use caset. Detta kan nog stämma teoretiskt men utefter de intervjusvar som erhållits verkar det mest praktiskt att även använda diagramformen då denna förbättrar överblicken över systemet och ökar förståelsen, kanske främst hos användarna.

- *”Hur går du till väga för att ta fram use case?”*

För att ta fram use case verkar det mest förekommande tillvägagångssättet vara att föra en diskussion med användarna av det tänkta informationssystemet, dock skiljer sig tillvägagångssättet lite åt mellan olika respondenter. Respondenterna 1, 2 och 5 har inte varit involverade i projekt där direktkontakt med kunden förekommit varför de heller inte tagit fram use case från ”scratch”. De har istället fått tilldelat sig färdiga use case-modeller som de sedan utvecklat och kompletterat. Det vanligaste tillvägagångssättet hos de övriga respondenterna tycks vara att först beskriva use case-tekniken och dess begrepp för användarna och ge någon form av exempel. Sedan får användarna ge förslag på vad som kan vara ett möjligt use case i det tänkta systemet. Därefter diskuteras aktörerna fram. Respondent 3 exemplifierade detta på följande vis:

> *”… sedan frågar jag ’har ni något exempel på det vi ska bygga nu som kan vara ett use case?’ Då brukar de oftast hitta på någonting. Då frågar man ’Vilka är det som har någon nytta av detta?’ och då får man upp aktörerna. Det brukar funka.”*

Respondent 6 däremot vänder på detta tillvägagångssätt och börjar med att identifiera aktörerna för att därefter undersöka vad de olika aktörerna ska kunna utföra i informationssystemet. Det förefaller vara svårt att få fram aktörerna till en början då de flesta användare ofta tänker sig personer istället för roller. Det är kanske lättare att föreställa sig specifika personer som har vissa arbetsuppgifter än att se en roll som kan innehas av vem som helst, även systemet. Respondent 5 påpekade:

> *”Det är viktigt att man har med användaren i den roll den har och inte den titel den har, utan i den rollen som de använder systemet”*

Att få fram korrekta use case verkar ha mycket att göra med kommunikationen med användarna och inte enbart i vilken form use casen representeras. Use case-diagrammen är lättare för oerfarna att förstå men text krävs för att få en tillräckligt detaljerad beskrivning av use caset. Det har framkommit att en nära relation med användarna är nödvändigt för att use case-tekniken ska fungera på ett tillfredsställande sätt. Även respondent 6 uttryckte detta: *”Det är en ständig diskussion med användarna”*. Use case-tekniken anses också som mycket lämplig för att kommunicera med användarna.

- *”Anser du att use case påverkar kommunikationen med användarna? På vilket sätt?”*

Samtliga respondenter som har använt use case ihop med användare anser att tekniken förbättrar kommunikationen med användarna. Anledningen till detta tycks vara att use case-tekniken har en nivå som passar både systemutvecklare och användare. Respondent 5 kommenterade detta på följande sätt:

> *”Det känns som en mellanhand. Användarna får lyfta sig från sin värld och systemutvecklarna får lyfta sig från sin systemvärld och så kan man enas om något.”*

Även respondenterna 2 och 6 anser att use case är en mycket god teknik för att komma överens med användarna, det är det man ”skakar hand” på, eller som respondent 2 kommenterade: *”Det är ingen som köper grisen i säcken om man gör en use case-modellering”*.

Use case-teknikens enkelhet gör att intressenter på olika nivåer kan förstå och kommunicera med tekniken. Detta är något som starkt talar för att use case-tekniken är lämplig att använda då diskussioner förs med användare och andra intressenter.

Respondent 6 betonar att det inte nödvändigtvis beror på den teknik man använder om alla krav täcks in under kravhanteringsprocessen, det beror också på kommunikationen med användarna:

> *”Jag tror det handlar mycket om delaktighet och ansvar. Är man delaktig så får man med alla krav. Use case-modellering hjälper inte om bara vi [systemutvecklarna] gör det, då kommer kraven att ramla förbi i alla fall.”*

Att tillåta och engagera användarna till att medverka i systemutvecklingsprocessen och då kanske främst vid kravidentifiering borde vara viktigt för att få fram ett väl fungerande informationssystem. Det är ju trots allt användarna som vet vad informationssystemet ska användas till, även om de naturligtvis inte kan sitta inne med alla svar.

Respondent 6 påpekade även:

> *”… det är väl en känsla som man får med sig under resans gång, att här kanske det ska vara två use case, är det inte två aktörer här. Ena stunden är han truckförare och i den andra ska han köra en feltrans men är han verkligen truckförare då, kan inte någon annan göra den feltransen. Det är bara en känsla man får av erfarenheten.”*

I diskussionen med användarna är det viktigt att förklara aktörsbegreppet och skillnaden mellan aktörer och personer i verksamheten. En erfaren systemutvecklare kan hjälpa användarna med denna åtskillnad.

För att möjliggöra ett väl fungerande framtagande av use case bör det finnas en god kommunikation med användarna. I detta hänseende är use case-diagram ett värdefullt hjälpmedel. Use case-diagram utvecklades just för att visualisera ett use case (Jacobson, 1994, i Fowler & Scott, 2000). Det bör vara lättare för användarna att förstå den visuella bilden än ett textbaserat use case. Dock är text nödvändigt för att få en fullkomlig beskrivning av ett use case.

5.3 Möjligheter och begränsningar med use case-tekniken

Nedanstående avsnitt innefattar en redogörelse för de frågor som berör möjligheter och begränsningar med use case.

- *”Vilka fördelar ser du med att använda use case för att identifiera och dokumentera krav?”*

Use case anses vara en enkel teknik som är lätt att lära in och lätt att förstå. Tekniken ger också en standard för att identifiera och dokumentera krav och skapar en gemensam bild av kraven för samtliga inblandade. Styrkan med use case som en enkel teknik är en återkommande åsikt även i litteraturen. Regnell (1999) nämner till exempel use case-teknikens enkelhet och att den är ett bra sätt att engagera användarna i utvecklingen. Även Karlsson (1996) såväl som Allen och Frost (1998) betonar enkelheten och det gemensamma synsätt mellan användare och systemutvecklare som use case kan åstadkomma som den största styrkan med use case-tekniken. Sammantaget tyder detta på att use case är en enkel teknik att lära och förstå för alla inblandade intressenter. Use case befrämjar en bra kommunikation med användarna och på så sätt underlättas arbetet med kravidentifiering. Use case-tekniken ger en tydlig gemensam standard för hur kraven ska hanteras vilket också minskar missförstånd och skillnader både mellan användare och utvecklare men också mellan olika utvecklare i projektgrupper som arbetar med att identifiera krav.

Nedan presenteras de möjligheter som framkommit under intervjuerna med att använda use case för att identifiera och dokumentera krav mer specifikt.

- Use case ger en tydlig överblick över hur ett system fungerar då det åskådliggör interaktionen mellan användare och system dvs ”vem som gör vad” i systemet. Det är också ett enkelt och snabbt sätt att få upp en bild av systemet, dess storlek och struktur och hur systemet hänger ihop
- Lätt att ta till sig och lära sig tekniken både för systemutvecklare och användare vilket bland annat beror på en enkel notationsteknik
- Ger en bra grund för att bygga systemet, skriva manualer och utforma tester
- Ger en grafisk bild som olika intressenter kan diskutera kring och fungerar som ett bra sätt att ”prata med användare”
- Samlar upp informationen på ”rätt” ställe ur användarens synvinkel
- Ger en gemensam standard för hantering av krav inom en verksamhet

Respondent 5 illustrerade ovanstående på följande sätt:

> *”Det är enkelt och snabbt att få upp en bild, storleken, strukturen, vad hänger ihop, man får ordning på sina aktörer, vem som gör vad i systemet. Man får en bild av systemet, att det är någonting som används i en roll, att det inte bara är en massa, inte bara ett system utan man får fram funktionaliteten.”*

Use case anses vara en väl fungerande teknik för att identifiera och dokumentera krav. Den anses också vara lätt att lära och att använda och detta är också något som bland annat Regnell (1999) samt Allen och Frost (1998) poängterar.

- *”Vilka nackdelar ser du med att använda use case för att identifiera och dokumentera krav?”*

Den främsta svagheten med use case-tekniken är att den inte täcker alla former av krav. Samtliga respondenter är överens om att use case inte täcker in ickefunktionella krav och detta är den vanligaste åsikten även i litteraturen. Kulak och Guiney (2000) anser dock att use case fungerar utmärkt även för ickefunktionella krav. (För vidare diskussion kring olika sorter av krav se nästa avsnitt.) Övriga svagheter som framkommit med use case-tekniken är att det är svårt att se processen/flödet i systemet. Use casen kan uppfattas som splittrade enheter utan kopplingar, vilket gör det svårt att se hur systemet egentligen ska fungera och i vilken ordning olika use case ska utföras. Det har under intervjuerna framkommit att respondenterna saknar ett sätt att tydliggöra processen/flödet då de arbetar med use case-tekniken. Ytterligare ett problem som nämnts av flera respondenter är att det är svårt att avgöra vilken detaljnivå ett use case ska ha. För mycket detaljer blir svårt att överblicka men vid för få detaljer blir det svårt att bygga systemet utifrån use casen. Det har även framkommit åsikter om att det är svårt att avgöra vad som är kravställande i ett use case, det vill säga vad som är absoluta krav och vad som bara är steg för att föra flödet framåt. Det kan vara svårt att avgöra om ordningen i sekvensen är viktig eller om alternativa ordningar kan finnas. Här är det viktigt att gemensamma regler skapas kring hur ett use case ska skrivas och tolkas. På ett av de medverkande företagen har detta lösts med vad som kallas för kravatomer vilket är minsta möjliga kravenhet. Dessa kravatomer kopplas till varje use case och är det som är kravställande. Kopplingen sker i ett datoriserat verktyg och det är då möjligt att för varje use case se vilka krav som är direkt kopplade till use caset.

Att skapa en konsensus om hur ett use case ska se ut och hanteras är mycket viktigt för att arbetet med use case ska fungera på ett tillfredsställande sätt. Ytterligare en svaghet med use case som tagits upp både av respondenter och i litteraturen är dess enkelhet. Enkelheten med use case är både en styrka och en svaghet, risken är att tekniken är för enkel och att den därför inte tas på allvar eller att den helt enkelt inte fungerar i komplexa sammanhang. En respondent kommenterade att många användare ansåg att streckgubbarna i use case-diagrammen var barnsliga. Vid sådana åsikter kan det naturligtvis vara svårt att få en bra kommunikation och svårt att bli ”tagen på allvar” som systemutvecklare. Detta kan kanske i viss mån också ha med företagskulturen att göra. Det var endast en respondent som kommenterade detta och det kanske kan bero på vilken inställning som just de användare respondenten arbetat med hade i det projektet.

Nedan presenteras de begränsningar som framkommit under intervjuerna med att använda use case för att identifiera och dokumentera krav mer specifikt.

- Stora system är svåra att överblicka med use case bland annat på grund av att det är svårt att se processen/flödet i systemet, dvs i vilken ordning användarna utför sina arbetsuppgifter
- Use case täcker inte alla krav, t ex prestanda, design och svarstider. Use case täcker inte heller in olika villkor
- Svårt att avgöra vad som är kravställande i ett use case, text är alltid öppet för tolkning
- Aktörsbegreppet kan vara svårt för användarna att förstå
- Kanske lite för enkel teknik, tas inte riktigt på allvar
- Svårt att avgöra vilken detaljnivå ett use case ska ha

Respondent 3 påpekade angående aktörsbegreppet:

> *”När man pratar med användarna vill de gärna ha en annan ikon för system och en för personer. De vill ha två olika sorters aktörer. Samtidigt tycker jag att det är rätt som det är för det som är en person idag kan vara ett system i morgon.”*

Ovanstående kan vara en viktig anledning till att aktörsbegreppet inte bör förändras, trots att det kan vara svårt att förstå till en början. En aktör kan vara både en människa och ett system och det ena kan komma att ersätta det andra. (se även respondent 6:s kommentar om aktörer under frågan ’Anser du att use case påverkar kommunikationen med användarna?’)

En viktig begränsning med use case-tekniken är också svårigheten att se processen/flödet i beskrivningen. Responden 6 påpekar:

> *”Flöden är något man av traditionen använt länge inom systemutveckling på olika sätt och det finns ju inte i den här modellen som vi använder.”*

Use case anses inte fungera för ickefunktionella krav vilket är en betydande begränsning med tekniken. Att det är svårt att se flödet i systemet ger också en begränsning i representationen av informationssystemet då sammanhanget mellan olika use case går förlorat. Svårigheten med att avgöra vilken detaljnivå som use case-modellen ska ha är troligtvis ett problem i början av användandet av use case-tekniken. Med ökad erfarenhet minskar troligen detta problem.

5.4 Use case som enda teknik för att representera krav

Nedanstående kapitel presenterar de frågor som är direkt kopplade till examensarbetets problemprecisering. Fråga åtta och nio, (se bilaga 1) är följdfrågor till fråga sju. Eftersom samtliga respondenter svarade nej på fråga sju har fråga åtta och nio ställts till samtliga respondenter. Fråga tolv är en följdfråga för de respondenter som svarat nej på fråga 11. Några respondenter har också ombetts att sätta sig in i situationen att inga kompletterande tekniker används och vad de tror att konsekvenserna av detta blir.

- *”Anser du att use case kan användas som enda teknik för att identifiera och dokumentera krav? Motivera varför/varför inte och ge gärna exempel?”*

Samtliga respondenter är överens om att use case inte fungerar som enda teknik för att identifiera och dokumentera krav. Detta tyder således på att det finns brister med use case-tekniken som behöver kompletteras med andra tekniker. Det är viktigt att inse, när ett företag anammar use case som standard, att det inte fungerar ensamt utan att det måste finnas kompletterande tekniker i de fall där use case inte fungerar. Bland respondenterna har det varit stor variation mellan vilka tekniker som används för att komplettera use case och det har framkommit att det egentligen inte finns några gemensamma riktlinjer för vad som ska användas. Det finns mycket att vinna, både i tid och pengar, om det finns en gemensam syn på vilka kompletterande tekniker som ska användas tillsammans med use case-tekniken.

Nedan presenteras de anledningar som angavs till att use case inte fungerar som enda teknik för identifiering och dokumentering av krav mer specifikt.

- Fungerar inte i stora system, bla på grund av brist på överblick
- Måste använda kompletterande kravdokument för övergripande/ generella krav som ej går att specificera i varje use case (tex Supplementary Specification, se nedan)
- Use case täcker inte olika villkor vilka måste dokumenteras någon annanstans
- Gränssnittet förloras, bör använda tex storyboarding eller gränssnittsprototyper för gränssnittet då det kan vara svårt att få en uppfattning om informationssystemets utseende annars. Dessutom krävs en kravlista för de ickefunktionella kraven.
- Processen saknas, det krävs t ex aktivitetsdiagram[6] för att få mer detaljer i kravbeskrivningen

Respondent 6 om use case som enda teknik:

> *”Man saknar processen, så någon typ av processmodellering tycker jag ska ligga till grund för det. Beskrivningarna av use case i sig behöver man komplettera i något dokument.”*

Det ena av de deltagande företagen använder sig av RUP där det finns anvisningar om vilka dokument som ska användas. För krav som ej fångas upp av use case dokumenteras dessa i Supplementary Specification, det vill säga ett mer generellt kravdokument.

Sammanfattningsvis anses use case inte fungera som enda teknik för att identifiera och dokumentera krav och nedan presenteras en närmare beskrivning över när use case-tekniken anses fungera respektive inte fungera.

[6] Ingår i UML

- *”Vid vilka typer av krav har use case fungerat tillfredsställande?”*

Nedan presenteras de typer av krav där respondenterna anser att use case-tekniken fungerar tillfredsställande.

- Samtliga respondenter är överens om att use case fungerar utmärkt vid funktionella krav
- Vid användarinteraktioner med systemet, vad en aktör ska kunna åstadkomma i systemet och vilken respons som ska ges, även här är samtliga respondenter överens om att use case fungerar

Respondent 1 menade att use case fungerar ”outstanding” vad gäller funktionella krav och detta tycks vara en åsikt som delas även av de övriga respondenterna.

I princip kan sägas att use case-tekniken anses fungera utmärkt för funktionella krav. I den litteratur som beskriver hur use case ska användas betonas också att det främst är en teknik för att hantera just funktionella krav (se till exempel Fowler & Scott, 2000; Kruchten, 2000; Karlsson, 1996). Detta är en åsikt som delas av samtliga respondenter och många respondenter har påpekat att de inte kan se någon annan teknik som skulle fungera bättre än use case-tekniken i dagsläget.

- *”Vid vilka typer av krav har use case inte fungerat tillfredsställande?”*

Det har framkommit att use case inte fungerar vid ickefunktionella krav. Det finns många krav på ett informationssystem som inte kan kategoriseras som funktionella krav och samtliga dessa faller utanför use case-tekniken. Respondenterna är överens om att det finns brister med use case-tekniken i fråga om ickefunktionella krav. Som nämnts tidigare finns det stöd för detta påstående även i litteraturen även om Kulak och Guiney (2000) anser att use case fungerar utmärkt även för ickefunktionella krav. I undersökningen har dock framkommit att hantera ickefunktionella krav med use case-tekniken torde bli svårt, om inte omöjligt. Jag kan inte se att use case är skapat för att hantera ickefunktionella krav från första början varför det borde vara svårt att använda tekniken på detta sätt. Dock kan andra tekniker eller dokument kopplas till use case-modellen och på så sätt specificeras även de ickefunktionella kraven. Många respondenter använder sig av något sådant dokument, som är direkt kopplat till use case-modellen, oftast i ett datoriserat verktyg. Nedan presenteras de typer av krav där respondenterna anser att use case-tekniken inte fungerar tillfredsställande mer detaljerat.

- Samtliga respondenter är överens om att use case-tekniken inte är tillräcklig vad gäller ickefunktionella krav. Exempel på sådana krav som getts under intervjuerna är
  - Egenskapskrav
  - Prestandakrav
  - Gränssnittshantering
  - ”Flummiga krav” som t ex användbarhet
- Två respondenter påpekar att use case inte ger stöd för processen/flödet i systemet

Respondent 6 påpekar också att use case-beskrivningen inte går att programmera utefter utan att det i dessa fall krävs andra former av diagram som till exempel klassdiagram och sekvensdiagram för att ge en mer detaljerad beskrivning att programmera efter.

- *”Anser du att use case bör kompletteras med någon annan teknik för att representera krav? Varför/varför inte?”*
  - ✓ *”Har du använt någon kompletterande teknik för att representera krav? Vilken och varför samt på vilket sätt har denna teknik använts?”*

Respondent 1 och 2 har inte använt någon kompletterande teknik till use case-tekniken och detta på grund av att det inom företaget bestämts att use case ska vara den enda tekniken som används. Dock ”fuskas” det med hur use case-tekniken används även ickefunktionella krav läggs till i use case-modellen. Respondent 2 påpekade:

> *”Det går inte att få med allting i ett use case, vi fuskar med den biten, vi lägger våra prestandakrav i use casen ... Vi har tre sorters krav, funktionella krav, prestandakrav och design restrictions som vi knyter på use case. Egentligen ingår ju de inte i use case men de kraven måste man ju också få ner.”*

Övriga fyra respondenter har använt kompletterande tekniker till use case-tekniken. De tekniker som används varierar och några exempel är storyboarding, kravlista, processmodellering, aktivitets-, sekvens- och klassdiagram. Respondent 3 menar att det faller sig naturligt att använda andra tekniker, *”behöver man det så tar man till det”*. Respondent 5 menar att teknikerna får skapas för att lösa ett aktuellt problem, därför kan det bli många olika tekniker beroende på problemet. Här saknas ett gemensamt sätt att lösa dessa problem. I den litteratur som undersökts finns heller inte några fungerande lösningar på dessa problem. Över huvud taget nämns inte mycket i litteraturen om svagheterna med use case och därmed inte heller hur detta kan lösas och vilka tekniker som är lämpliga komplement till use case-tekniken.

Det har framkommit att det finns vissa problem med att använda kompletterande tekniker till use case-tekniken. Främst beror dessa problem på att det inte finns några gemensamma bestämmelser om vilka tekniker som ska användas utan systemutvecklarna får ofta ”hitta på” vilken teknik som är lämplig i olika situationer. Det är viktigt att medvetenhet finns om use case-teknikens begränsningar och att det finns ett gemensamt sätt att hantera dessa begränsningar. Kan en formalisering av hantering av ickefunktionella krav ske så skulle det underlätta arbetet med kravhantering ytterligare. Kanske behövs det en motsvarande use case-teknik för ickefunktionella krav.

- *”Om du anser att use case bör kompletteras med annan teknik men ändå inte använt någon kompletterande teknik, vad tror du det har haft för konsekvenser för kravrepresentationen?”*

Respondent 2, som inte använder några kompletterande tekniker, anser att många av de felrapporter som kommer in på felaktiga eller saknade krav beror på att use case-tekniken inte räcker till; *”Man har mer i huvudet men ingenstans att skriva ner det”*. Detta leder till fler felrapporter att åtgärda än vad som skulle vara nödvändigt och en kompletterande teknik kunde fånga in det som use case inte fångar in.

Ytterligare kommentarer kring följden av att inte använda någon kompletterande teknik är att det blir svårare att hitta alla krav vilket i sin tur leder till problem i design och konstruktion av systemet. Att åtgärda dessa problem kan bli betydligt dyrare än om problemen åtgärdas tidigare i systemutvecklingsprocessen.

Att inte använda någon kompletterande teknik till use case-tekniken borde leda till att inte alla krav identifieras och att det därmed blir omöjligt att konstruera ett väl fungerande system. Som nämnts tidigare är det av vikt att det finns en medvetenhet om detta för att kunna uppväga de krav som use case-tekniken inte täcker in.

5.5 Övriga kommentarer kring use case och kravhantering

I detta kapitel presenteras sådan information som framkommit under intervjuerna och inte täcks in av ovanstående frågor.

Användarnas medverkan i systemutvecklingsprocessen och systemutvecklarnas erfarenhet spelar en avgörande roll för hur väl use case-tekniken fungerar. Ju längre erfarenhet systemutvecklare har av en teknik desto bättre behärskar de den och det är ju också så att misstagen så småningom blir erfarenheter. Mycket av den kunskap som systemutvecklarna får med erfarenheten är dock så kallad ”tyst kunskap” (Waern, 1993). Detta innebär att det är kunskap som är svår att beskriva och sätta ord på, den ”bara finns där”. Denna kunskap är dock mycket värdefull för en väl fungerande systemutveckling.

Respondent 1 anser att use case-tekniken fungerar bra och att det behövs en teknik för kravhantering som är genomgående i ett företag så att alla talar samma språk, på samma sätt som syntaxen är densamma i ett programmeringsspråk. Att det inom en verksamhet finns just en gemensam syn på hur use case-tekniken ska användas bör vara en förutsättning för att tekniken ska fungera på optimalt sätt. Use case-tekniken kan vara öppen för tolkning vilket kan innebära att, även om use case används som standard inom en verksamhet, missförstånd ändå uppstår, både mellan användare och systemutvecklare men också mellan olika systemutvecklare.

Under intervjuerna har frågan om verktygsstöd för use case-tekniken dykt upp vid ett flertal tillfällen. Åsikterna kring detta går isär då vissa respondenter anser att det inte finns tillräckligt verktygsstöd för tekniken medan andra anser att det finns tillräckligt verktygsstöd, men det gäller att lära in dem för att kunna använda dem optimalt. Även Simons och Graham (1998) tar upp bristande verktygsstöd för use case-tekniken som ett problem. Särskilt viktigt med ett väl fungerande verktygsstöd verkar det vara då kopplingar mellan use case-modellen och andra dokument ska ske. Det måste finnas en koppling mellan use case-modellen och de krav som denna inte täcker in för att inte några krav ska glömmas av.

5.6 Reflektioner

Sammantaget kan sägas att de intervjuer som genomförts i examensarbetet givit ett gott underlag för att ge svar på arbetets problemställning. De olika intervjudeltagarna har i stort varit mycket överens i sina uppfattningar kring de olika intervjufrågorna, vilket tyder på att det som framkommit också kan vara en korrekt bild av use case-tekniken. Dock är underlaget på sex intervjudeltagare litet och resultatet kan därför inte generaliseras att gälla i alla kravhanteringssituationer och för samtliga systemutvecklare.

De intervjufrågor som inte hade någon direkt koppling till problempreciseringen har visat sig vara mycket värdefulla. De har hjälpt till att skapa en större förståelse för processen med att arbeta med use case-tekniken och resultatet hade troligtvis inte på samma sätt kunnat ge de svar som nu framkommit på problemställningen utan dessa frågor.

I nästa avsnitt presenteras de slutsatser som dragits som svar på examensarbetets problemprecisering samt en diskussion kring detta resultat.

6 Slutsatser och diskussion

I följande avsnitt presenteras slutsatser utifrån hur resultatet givit svar på examensarbetets problemprecisering:

> *Är use case tillräcklig som enda teknik för att identifiera och dokumentera krav. Om inte; hur kompletterar systemutvecklare de former av krav som use case-tekniken inte fångar in?*

Dessutom förs en diskussion kring detta resultat och vad det kan bidra till.

I nedanstående kapitel presenteras också en diskussion kring den inblick om use case-tekniken som erhållits samt kring arbetsprocessen med examensarbetet. Slutligen presenteras också förslag på framtida arbete.

6.1 Slutsatser

Utifrån den undersökning som gjorts i examensarbetet är use case-tekniken inte tillräcklig som enda teknik för att identifiera och dokumentera krav. Use case-tekniken fångar in funktionella krav väl men vad gäller ickefunktionella krav fungerar use case-tekniken inte lika väl. Detta stämmer också väl överens med vad som framkommit i litteraturen. Systemutvecklare använder flera olika kompletterande tekniker för att uppväga use case-teknikens svagheter. Exempel på sådana tekniker är:

- *Gränssnittsprototyper*, för att på ett mer illustrativt sätt beskriva hur informationssystemet kommer att se ut.
- *Storyboarding och processmodellering*, för att fånga upp processen/flödet, dvs användarnas arbetsordning.
- *Sekvensdiagram*, för en mer sammanhängande bild av systemet.
- *Klassdiagram*, att programmera utefter.
- *Kravdokument*, för att dokumentera generella krav och ickefunktionella krav såsom prestanda, tidskrav samt andra övergripande systemkrav.
- *Kravatomer*, för att beskriva vad i ett use case som är kravställande

Det har dock framkommit under intervjuerna att det saknas en formell beskrivning över vilka tekniker som ska användas och hur dessa ska användas då use case-tekniken inte räcker till. Det har inte heller framkommit något i den litteratur som undersökts kring detta. Det verkar inte finnas så många förslag på lämpliga tekniker för att komplettera use case-tekniken i litteraturen. Detta kan bero på att det är svårt att uppnå samma form av standardisering för ickefunktionella krav som för funktionella. Ickefunktionella krav är ofta ”luddiga” vilket gör det svårt att hantera dem på ett gemensamt, strukturerat sätt. Det är viktigt för en verksamhet som beslutar att använda use case som standard att medvetenhet finns för de begränsningar som tekniken innehar samt att en lösning för dessa begränsningar utarbetas. Risken är annars att varje enskild systemutvecklare konstruerar olika lösningar under utvecklingsprocessens gång och därmed finns otydligheter och risk för missuppfattningar kvar.

Resultatet i examensarbetet tyder på att en komplettering till use case-tekniken bör göras eller att en eller flera andra tekniker bör utvecklas, vilka kan motsvara use case-tekniken, för att lösa de begränsningar som use case-tekniken har idag och ge samma struktur och standardisering som use case ger. Kan en formalisering av hantering av ickefunktionella krav ske så skulle det underlätta arbetet med kravhantering ytterligare. Det bör finnas mycket att vinna, både i tid och pengar, om det i en verksamhet finns en gemensam syn på vilka kompletterande tekniker som ska användas tillsammans med use case-tekniken. Det kan dock vara viktigt att poängtera att use case-tekniken endast är en beskrivningsteknik och att denna inte kan förväntas att lösa alla problem i kravhanteringsprocessen. Det bör finnas tillräckligt stöd för kravhanteringsprocessen i den systemutvecklingsmetod som används så att det är möjligt att förlita sig på metoden då tekniken inte räcker till.

Ytterligare diskussion kring den information som erhållits om use case-tekniken under undersökningen i examensarbetet presenteras i nästa avsnitt.

6.2 Diskussion kring use case-tekniken

Kravhantering är en mycket viktig del i systemutvecklingsprocessen vilket har framkommit tydligt under arbetet med examensarbetet. Om det finns ett flertal väl fungerande tekniker som kan stödja processen med att identifiera och dokumentera krav så bör detta vara oerhört värdefullt för kvaliteten i kravhanteringsprocessen och därmed kvaliteten på det färdiga informationssystemet. Mycket talar för att use case-tekniken är en god början för att identifiera och dokumentera krav och då främst funktionella krav. Use case-tekniken är enkel att förstå och ta till sig vilket innebär att tekniken också fungerar som ett bra kommunikationsmedel mellan olika intressenter till ett informationssystem. Om det finns en väl fungerande kommunikation med användarna kan rätt krav identifieras från början och detta leder till ett bättre informationssystem. Use case beskriver den funktionalitet som användare och systemutvecklare diskuterat fram, det informationssystem som ska byggas.

Det har framkommit att use case-tekniken inte alltid verkar ha fått den status som den kanske förtjänar. På ett av de medverkande företagen förvaltas inte use case-modellerna utan används enbart i analysfasen för att sedan läggas undan och inte användas mer. Om use case-tekniken kan få en högre status inom systemutvecklingskretsar och kanske främst i verksamhetsledningar skulle detta troligen innebära att tekniken också skulle fungera bättre. Om en teknik inte tas på allvar kan den heller inte fungera optimalt. Det är en kombination av teknikens kvalitet och kunskap om tekniken hos de som är inblandade i utvecklingen av ett informationssystem som avgör om en teknik fungerar på ett optimalt sätt eller ej. Det kan också vara viktigt att väga in den företagskultur som existerar då även denna kan påverka vilken status use case-tekniken får då den introduceras på ett företag.

Under den litteraturstudie som utförts under examensarbetets gång har framkommit ett problem med use case som inte framkommit under intervjuerna. Enligt Sindre och Opdahl (2001) stödjer inte use case-tekniken säkerhetskrav av formen ’vad får *inte* hända i informationssystemet’ och ’vem får inte använda systemet’. Med ökad Internetanvändning blir säkerhetskrav av denna sort allt viktigare och det krävs stöd även för hantering av dessa krav. Varför dessa former av krav inte framkom under intervjuerna är svårt att avgöra. Kanske anses dessa krav vara så ”långt ifrån” use case att de inte ens reflekteras över.

Enbart use case-tekniken garanterar inte en väl fungerande kravhanteringsprocess, utan det är också av stor betydelse vilken erfarenhet de systemutvecklare har som arbetar med kravhantering. Det märktes mycket tydligt under intervjuerna med systemutvecklare med skiftande erfarenhet, att de oerfarna utvecklarna litade mer till tekniker och metoder än de erfarna. De erfarna systemutvecklarna har insett att tekniker och metoder inte alltid har svar på allt utan att det kan finnas variationer på hur processen kan gå till och att man inte bör vara rädd för att frångå metod och teknik om man anser att det behövs.

Det är naturligtvis också så att use case-tekniken som sådan inte nödvändigtvis måste vara den enda rätta för kravhantering. Use case är en relativt ny teknik och anses för tillfället som mycket väl fungerande för kravhantering men om tre år har kanske ytterligare en ny teknik har utvecklats som anses ännu bättre. Kanske kan det ibland vara ”nyhetens behag” med nya tekniker som påverkar systemutvecklare att använda dem. Kanske är dock use case-tekniken en god grund för vidareutveckling som kan leda till allt bättre tekniker framöver.

6.3 Erfarenheter av arbetsprocessen

Under arbetsprocessens gång med examensarbetet har jag fått flera nya erfarenheter och en del bekräftade teoretiska kunskaper. Att utföra intervjuer upplevdes svårt och ovant. Intervjuernas kvalitet beror mycket både på hur jag fungerat i rollen som intervjuare men också på hur öppna respondenterna varit. Att bemöta människor på ett sätt som får dem att känna sig avslappnade och positiva till intervjusituationen är en viktig men också en svår uppgift. Detta är något som kommer med erfarenhet och ju fler intervjuer som genomförts desto lättare blev bemötandet av andra människor. Att hålla sig objektiv och inte leda in respondenterna på något särskilt spår är även det en svår uppgift. Det förekom under intervjuerna att några respondenter bad mig att ge några exempel på vad andra svarat för att de lättare skulle kunna svara på frågan. Detta kändes naturligtvis inte rätt och det blev en svår uppgift att ”parera” sådana önskemål.

Att kravhantering är en mycket viktig del i systemutvecklingen var något jag var medveten om, om än på ett teoretiskt plan, men detta har bekräftats gång på gång under undersökningen. Att spendera tid och resurser på att förbättra kravhanteringen kan vara väl värt då det i förlängningen bör leda till bättre och billigare informationssystem. Det verkar vara så att detta är något som systemutvecklare är väl medvetna om men att förståelse för detta inte alltid finns i ledningen där finansieringsbeslut tas. Det kan vara svårt att se att något som kostar mycket nu kommer att ge stora besparingar långt fram i tiden.

Arbetet med examensarbetet har gett mig mycket som jag kan ha nytta av i min framtida yrkesroll som systemutvecklare. Att komma ut i ”verkligheten” och intervjua systemutvecklare har varit mycket givande, på ett bredare plan än att enbart att få svar på problemställningen. Trots en lång och krävande process känns detta dock mycket givande liksom att jag anser mig fått svar på min problemställning på ett tillfredsställande sätt.

6.4 Förslag på fortsatt arbete

Eftersom underlaget gällande intervjudeltagare varit relativt litet, två olika företag inom ett begränsat geografiskt område och totalt sex stycken intervjudeltagare, kan inte någon generalisering av resultatet göras. På grund av detta vore en utökning av underlaget värdefullt för att försöka åstadkomma någon form av generaliserbarhet av ett resultat att gälla för systemutvecklare i allmänhet. Denna utökning skulle kunna vara i form av en eller flera av följande förslag; fler företag, fler deltagare och ett större geografiskt område, kanske även utanför Sverige för att få en internationell synvinkel. Use case-tekniken används över stora delar av världen varför det vore intressant att undersöka om åsikterna kring tekniken skiljer sig åt i Sverige mot utomlands.

Ett annat intressant fortsatt arbete, som dock kräver mer tid och resurser än vad som finns inom ramen för ett examensarbete, är att utveckla en eller flera tekniker som kan komplettera use case-tekniken. Troligtvis krävs flera olika tekniker för att fånga in till exempel ickefunktionella krav, flödesbeskrivningar och övergripande systemkrav. Då dessa tekniker utvecklats bör de också testas och utvärderas för att se hur väl teknikerna löser de begränsningar med use case-tekniken som framkommit under denna undersökning.

Med tanke på att use case trots allt endast är en beskrivningsteknik och kanske inte kan förväntas att lösa samtliga problem som kan uppkomma i kravhanteringsprocessen kan det vara intressant att undersöka hur olika systemutvecklingsmetoder stöder kravhanteringsprocessen. Om en beskrivningsteknik inte räcker till bör den systemutvecklingsmetod som används ha alternativ till hur andra tekniker kan användas. Frågan är om för mycket ”ansvar” läggs på hur teknikerna fungerar istället för att lägga detta ansvar på metoden, vilket kanske borde vara mer korrekt.

Eftersom frågan kring verktygsstöd för use case-tekniken kommit upp vid ett flertal tillfällen under intervjuerna vore en undersökning av befintliga verktygsstöd värdefullt. Dels för att ta reda på vilka typer av verktygsstöd som finns på marknaden idag och dels för att undersöka hur väl dessa stödjer arbetet med use case-tekniken. Kanske finns även här ett behov för vidare utveckling. I den litteratur som beskriver hur UML och use case ska användas finns inte mycket information om vilka verktyg som kan eller bör användas. Då detta är en viktig del i att use case-tekniken ska kunna fungera på bästa sätt bör verktygsstöd undersökas och framhållas till systemutvecklare som arbetar med use case.

I examensarbetet har endast systemutvecklares syn på att arbeta med use case-tekniken undersökts. En intressant fortsättning på detta arbete vore att undersöka hur användarna av ett blivande informationssystem ställer sig till att arbeta med use case-tekniken. Anser användarna att use case-tekniken är lätt att lära och förstå? Ökar användandet av use case-tekniken intresset hos användarna för medverkan i systemutvecklingsprocessen? Har användarna liknande åsikter som systemutvecklarna eller skiljer åsikterna sig åt och i så fall varför?

Referenser

Allen, P. & Frost, S. (1998) *Component-Based Development for Enterprise Systems.* Cambridge: Cambridge University Press.

Andersen, E.S. (1994) *Systemutveckling – principer, metoder och tekniker* (2:a upplagan). Lund: Studentlitteratur.

Avison, D.E. & Fitzgerald, G. (1995) *Information Systems Development: Methodologies, Techniques and Tools* (2:a upplagan). Berkshire: McGraw-Hill Book Company Europe.

Booch, G., Rumbaugh, J. & Jacobson, I. (1999) *The Unified Modeling Language User Guide*. Reading: Addison Wesley Longman, Inc.

Cockburn, A. (2001) *Writing Effective Use Cases*. Boston: Addison Wesley.

Dahlstedt, Å.G. (2001) *Requirements Managements from a Life Cycle Perspective – Overview and Research Areas.* MSc dissertation. Datainstitutionen, Högskolan i Skövde.

Dawson, C.W. (2000) *The Essence of Computing Projects: A Student´s Guide.* Essex: Pearson Education Limited.

Ejlertsson, G. (1996) *Enkäten i praktiken –en handbok i enkätmetodik*. Lund: Studentlitteratur.

Eriksson, H.E. & Penker, M. (2000) *Business Modeling with UML – Business Patterns at Work*. New York: John Wiley & Sons, Inc.

Fowler, M. & Scott, K. (2000) *UML Distilled: A Brief Guide To The Standard Object Modeling Language* (2:a upplagan). Reading: Addison Wesley Longman, Inc.

Gause, D.C. & Weinberg, G.M. (1989) *Exploring Requirements: Quality Before Design*- New York: Dorset House Publishing Co.

Harker, S.D.P., Eason, K.D. & Dobson, J.E. (1993) *The Change and Evolution of Requirements as a Challenge to the Practice of Software Engineering*. Proc IEEE Symposium on Requirements Engineering, San Diego, California, USA.

Karlsson, J. (1996) *Framgångsrik kravhantering – vid utveckling av programvarusystem*. Sveriges Verkstadsindustrier.

Kruchten, P. (2000) *The Rational Unified Process: An Introduction* (2:a upplagan). Reading: Addison Wesley Longman, Inc.

Kulak, D. & Guiney, E. (2000) *Use Cases – Requirements in Context*. Boston: Addison-Wesley.

Lauesen, S. (2000) *Software Requirements – Styles and Techniques* (2:a upplagan). Frederiksberg: Forlaget Samfundslitteratur.

Lee, J. & Xue, NL. (1999) Analyzing User Requirements by Use Cases: A Goal-Driven Approach. *IEEE Software July/August*, 92-101.

Loucopoulos, P. & Karakostas, V. (1995) *System Requirements Engineering*. Berkshire: McGraw-Hill Book Company Europe.

Maciaszek, L. (2001) *Requirements Analysis and System Design – Developing Information Systems with UML*. Essex: Pearsons Education Limited.

Patel, R. & Davidson, B. (1994) *Forskningsmetodikens grunder, att planera, genomföra och rapportera en undersökning*. Lund: Studentlitteratur.

Pohl, K. (1996) *Process-Centered Requirements Engineering*. Somerset: Research Studiees Press LTD.

Regnell, B. (1999) *Requirements Engineering with Use Cases – a Basis for Software Development*. Teknisk rapport nr 132. Lund: Lunds Universitet

Saiedian, H. & Dale, R. (2000) Requirements engineering: making the connection between the software developer and customer. *Information and Software Technology 42*, 419-428.

Schneider, G. & Winters, J.P. (2001) *Applying Use Cases, Second Edition – A Practical Guide* (2:a upplagan). Boston: Addison-Wesley.

Shaughnessy, J.J & Zechmeister, E.B. (1997) *Research Methods in Psychology* (4:e upplagan). Singapore: McGraw-Hill Companies Inc.

Simons, A. & Graham, I. (1998) *37 Things that Don´t Work in Object-Oriented Modelling with UML*. University of Sheffield & Chase Manhattan Bank.

Sindre, G. & Opdahl, A.L. (2001) *Templates for Misuse Case Description*. In Proceedings of the Seventh International Workshop on Requirements Engineering: Foundation for Software Quality, REFSQ'01, Interlaken, Switzerland, 4-5 June, 2001, pp.125-136.

Sommerville, I. & Sawyer, P. (1997) *Requirements Engineering – A Good Practice Guide*. Chichester: John Wiley & Sons.

Sutcliffe, A.G., Maiden, N.A.M., Minocha, S. & Manuel, D. (1998) Supporting Scenario-Based Requirements Engineering. *IEEE Transactions on Software Engineering 24: (12)*, 1072-1088.

Trost, J. (1993) *Kvalitativa intervjuer*. Lund: Studentlitteratur.

Tuok, R. & Logrippo, L. (1998) Formal specification and use case generation for a mobile telephony system. *Computer Networks and ISDN Systems 30*, 1045-1063.

Waern, Y. (1993) *Från människa till datoranvändare*, i L. Lennerlöf (red.) (1993) *Människor – Datateknik – Arbetsliv*. Stockholm: Fritzes AB.

Wiegers, K.E. (1997) Listening to the Customer´s Voice. *Software Development, March*. [Elektronisk version]. Tillgänglig på Internet: http://www.processimpact.com [Hämtad 02.01.28]

Intervjufrågor

Inledande frågor

1. Hur länge har du varit anställd på företaget?
2. Hur lång erfarenhet har du inom systemutveckling?
3. Hur lång erfarenhet har du av att arbeta med use case i samband med kravhantering?

Huvudfrågor

4. Föredrar du att representera kraven i textform eller i use case-diagram? Varför?
5. Vilka fördelar ser du med att använda use case för att identifiera och dokumentera krav?
6. Vilka nackdelar ser du med att använda use case för att identifiera och dokumentera krav?
7. Anser du att use case kan användas som enda teknik för att identifiera och dokumentera krav? Motivera varför/varför inte och ge gärna exempel?

Om nej på fråga 7 ställs fråga 8 och 9

8. Vid vilka typer av krav har use case fungerat tillfredsställande?
9. Vid vilka typer av krav har use case inte fungerat tillfredsställande?

10. Anser du att use case bör kompletteras med någon annan teknik för att representera krav? Varför/varför inte?
11. Har du använt någon kompletterande teknik för att representera krav? Vilken och varför samt på vilket sätt har denna teknik använts?

Om nej på fråga 11 ställs fråga 12

12. Om du anser att use case bör kompletteras med annan teknik men ändå inte använt någon kompletterande teknik, vad tror du det har haft för konsekvenser för kravrepresentationen?
13. Hur går du till väga för att ta fram use case?
14. Anser du att use case påverkar kommunikationen med användarna? På vilket sätt?

15. Övriga kommentarer kring kravhantering och use case.