Prototyping för effektiv kravutvinning.

(HS-IDA-ES-98-325)

Anders Winbladh (b95andwi@ida.his.se)
Institutionen för datavetenskap
Högskolan i Skövde, Box 408
S-54128 Skövde, SWEDEN

Examensarbete på det systemvetenskapliga programmet under vårterminen 1998.
Handledare: Anders Eklund

Prototyping för effektiv kravutvinning

Examensrapport inlämnad av Anders Winbladh till Högskolan i Skövde, för Kandidatsexamen (BSc) vid Institutionen för Datavetenskap.

1998-06-08

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: _____

Prototyping för effektiv kravutvinning.

Anders Winbladh (b95andwi@ida.his.se)

Key words: Prototyping, requirements elicitation, information system

Abstract

To success in developing an information system, the most important thing is to understand the needs and requirements of its users. There are several methods to elicitate user requirements and needs. I have in this report critically examined one of these methods, prototyping. Prototyping is a method which using working modells, prototypes, to elicitate requirements and to improve the understanding about the system among the users. But unfortunately there are several boundary conditions. The issues in this report are; what are the boundary conditions for the use of prototyping, what kind of effects have the type of system to be developed on the boundary conditions of prototyping, What influence do the boundary conditions have on the phases within the system lifecycle? I have also made a guideline to deside when prototyping is an appropriate method or not.

Innehållsförteckning

SAMMANFATTNING **1**

1 INTRODUKTION **2**

2 BAKGRUND **3**

2.1 INTRODUKTION TILL PROBLEMOMRÅDET 3
2.1.1 Kravhantering *4*
2.1.2 Kravutvinningsmetoder *5*
2.1.3 Prototyping *6*
2.2 CENTRALA BEGREPP 7
2.2.1 Krav *7*
2.2.2 Informationssystem *9*
2.2.3 4-GL verktyg *9*
2.2.4 Kvalitet *10*

3 PROBLEMOMRÅDES BESKRIVNING **11**

3.1 AVGRÄNSNING 11
3.2 PROBLEMDEFINITION 11
3.3 FÖRVÄNTAT RESULTAT 12

4 METOD **13**

4.1 MÖJLIGA METODER 13
4.1.1 Fallstudie *13*
4.1.2 Litteraturstudie *13*
4.1.3 Intervjuer *14*
4.1.4 Enkäter *14*
4.2 VAL AV METOD 15
4.3 KÄLLOR 15

5 GENOMFÖRANDE **17**

5.1 SÖKMETODER 17
5.2 KÄLLINFORMATION 17
5.2.1 Presentation *17*
5.2.2 Undersöka vilka egenskaper som är begränsande för prototyping. *17*
5.2.3 Undersöka hur typen av informationssystem påverkar lämpligheten av prototyping *19*
5.2.4 Ta reda på vilka faser under utvecklingen som påverkas tillföljd av dessa brister. *19*
5.2.5 Utföra en bedömning när det kan vara lämpligt att använda prototyping *21*
5.3 ANALYS 22
5.3.1 Presentation *22*
5.3.2 Undersöka vilka egenskaper som är begränsande för prototyping. *22*
Kostnad *22*
Användarmedverkan *22*
Verktyg *23*
Icke funktionella krav *23*
Flexibilitet *24*
5.3.3 Undersöka hur typen av informationssystem påverkar lämpligheten av prototyping *24*

Tekniska system *24*
Interaktion *24*
Storlek *25*
5.3.4 Ta reda på vilka faser under utvecklingen som påverkas tillföljd av dessa brister. *25*
5.3.5 Utföra en bedömning när det kan vara lämpligt att använda prototyping *25*

6 SLUTSATSER **28**

6.1 UNDERSÖKA VILKA EGENSKAPER SOM ÄR BEGRÄNSANDE FÖR PROTOTYPING 28
6.2 UNDERSÖKA HUR TYPEN AV INFORMATIONSSYSTEM PÅVERKAR LÄMPLIGHETEN 29
6.3 TA REDA PÅ VILKA UTVECKLINGSFASER SOM PÅVERKAS TILLFÖLJD AV BRISTERNA 29
6.4 UTFÖRA BEDÖMNING AV LÄMPLIGHETEN FÖR ANVÄNDNING UTAV PROTOTYPING 29

7 DISKUSSION **31**

7.1 PRESENTATION 31
7.2 UNDERSÖKA VILKA EGENSKAPER SOM ÄR BEGRÄNSANDE FÖR PROTOTYPING 31
7.3 UNDERSÖKA HUR TYPEN AV INFORMATIONSSYSTEM PÅVERKAR LÄMPLIGHETEN 31
7.4 TA REDA PÅ VILKA UTVECKLINGSFASER SOM PÅVERKAS TILLFÖLJD AV BRISTERNA 31
7.5 UTFÖRA BEDÖMNING AV LÄMPLIGHETEN FÖR ANVÄNDNING UTAV PROTOTYPING 31
7.6 ÖVRIGT 32
7.7 UPPSLAG TILL FORTSATT ARBETE 32

REFERENSER **33**

INDEX **34**

Sammanfattning

Vid utveckling av datoriserade informationssystem är kvaliteten och därmed kostnaden på slutprodukten i hög grad beroende av hur väl man har lyckats att realisera kundens krav och önskemål på det tänkta systemet. För att kunna realisera kundens krav måste först de krav som kunden har på systemet komma till kännedom för utvecklaren. Kunden är inte heller alltid förmögen att ställa samtliga krav då han eller hon inte vet vad som är möjligt att kräva av systemet. Som stöd för utvinningen av kundkrav finns ett antal olika metoder och ansatser. Samtliga av dessa metoder har dock brister och begränsande egenskaper som påverkar deras effektivitet för att utvinna samtliga kundkrav för ett tänkt informationssystem.

Den metod som kritiskt analyseras i denna studie är prototyping. Närmare bestämt throw-away prototyping. Prototyping är en kravutvinningsmetod som utnyttjar prototyper av det tänkta systemet som grund för kravutvinningen. Detta förfarande främjar dialogen mellan användare och utvecklare då man har något konkret att tala omkring samt att användarna får en förståelse vilka krav som är möjliga att ställa på systemet. Dock har metoden vissa egenskaper som begränsar dess lämplighet vid utveckling av vissa informationssystem.

Jag har utifrån en litteraturstudie behandlat frågeställningarna; vilka begränsande egenskaper har prototyping och när uppkommer dessa egenskaper, hur påverkar typen av informationssystem, som skall utvecklas, lämpligheten för prototyping samt vilka faser i systemutvecklingsarbetet påverkas tillföljd av dessa begränsningar? Jag har dessutom gjort en bedömning när det kan vara lämpligt att använda sig utav prototyping i avseende på totalkostnaden för ett projekt. Det är inte en jämförande studie mellan olika metoder utan en studie som enbart behandlar prototyping.

Jag har kommit fram till att prototyping har flera begränsande egenskaper vilka man måste beakta vid valet av prototyping som kravutvinningsmetod. Prototyping ger en förändrad kostnadsbild med en ökad kostnad för analysfasen, prototyping kräver en intensiv användarmedverkan, prototyping skapar behov av utvecklingsverktyg och utvecklare som behärskar dessa, prototyping ger ett svagt stöd för icke funktionella krav samt att prototyping hämmar flexibiliteten vid fluktuerande krav.

Informationssystemets typ och beskaffning spelar också en avgörande roll för lämpligheten av prototyping. Jag har kommit fram till att tekniska system samt system som har liten eller ingen interaktion mellan användare och system inte är lämpade för prototyping som kravutvinningsmetod. Även storleken på informationssystemet är en faktor som påverkar lämpligheten av metoden.

Systemutvecklingsfaserna som påverkas av de begränsande egenskaperna som är knutna till prototypingmetoden är samtliga faser, enligt min studie. De kostnader som genereras om man inte beaktar de begränsande egenskaperna är dock främst knutna till underhållsfasen.

Med utgångspunkt från de begränsande egenskaperna jag har analyserat har jag gjort en bedömning då prototyping är lämpligt utifrån vissa förutsättningar. Bedömningen kan även användas som en guideline för att säkerställa att prototyping är en lämplig metod för kravutvinning i ett specifikt systemutvecklingsprojekt.

1 Introduktion

Vid utveckling av informationssystem är kvaliteten på både utvecklingsarbetet i sig och framför allt slutprodukten starkt beroende av hur väl man lyckas med arbetet i de tidiga faserna av utvecklingsarbetet. Den grund av information som byggs upp i initieringsfaserna av behov och krav, som kunden ställer på informationssystemet, ligger till grund för det fortsatta utvecklingsarbetet. Det innebär att kundens samtliga krav och behov måste vara korrekt tolkade och fullständiga för att ett kvalitativt och effektivt och därmed lyckat systemutvecklingsprojekt skall bli realiserat.

Hög kvalitet innebär att kostnaderna minskar, framför allt i underhållet av systemet, och att leveranstiderna blir kortare. En undersökning som IBM har genomfört som visar att upptill 60 % av de totala kostnaderna under hela systemets livscykel står underhållet av systemet för (Vonk, 1990). Andra studier visar (VI, 1995) att det kostar upp till 100 ggr mer att ändra fel i krav efter leverans eller i de senare testfaserna. Om kvaliteten kan säkras på ett tidigt stadium, genom att kundens samtliga behov och krav kan fångas och dokumenteras kan underhållskostnaderna, och därigenom den totala utvecklingskostnaden minskas till ett minimum (Vonk, 1990).

Svårigheterna med kravhanteringen är att utvecklaren inte förstår kundens krav, kravens konsekvenser och kundens oförmåga att förstå vilka krav som kan ställas (Loucopoulos & Karakostas, 1995). Dessutom är kommunikationsproblem vanliga, då utvecklare och kund har olika referensramar samt språk.
I de traditionella systemutvecklingsmodellerna finns det flera olika metoder att komma åt dessa svårigheter. Genom bl a strukturerade, ostrukturerade intervjuer och observationer (Karlsson, 1996). En metod som blir allt vanligare, då 4-GL verktygen blir billigare och mer raffinerade , är prototyping.
Genom prototyping och användningen av prototyper av det tänkta systemet får kunden på ett tidigt stadie en bild av systemet och kan ställa mer korrekta krav och utvecklaren får direkt feedback på de ändringar och tillägg som görs gentemot de krav kunden ställt.

Prototyping med stöd av 4-GL verktyg är en relativt ny och teknikdriven metod. Det innebär att den förutom sina positiva egenskaper för kravutvinning, även kan ha mindre bra egenskaper och brister som i vissa fall begränsar utvecklingsarbeten och tvingar utvecklare att använda alternativa metoder för kravutvinning.

Med detta arbete skall jag ge en övergripande bild över vilka brister som prototyping har för kravutvinning och hur typen av informationssystem påverkar prototypingmetodens lämplighet för kravutvinning samt analysera vilka faser i utvecklingsarbetet som påverkas av dessa. Dessutom skall jag göra en bedömning under vilka förutsättningar prototyping är ett lämpligt val för ett systemutvecklingsprojekt, vilket också kan fungera som guide vid val av kravutvinningsmetod.

2 Bakgrund

För att läsaren skall kunna sätta sig in i problemområdet följer en bakgrund till systemutvecklingsarbetets problem och möjligheter.

2.1 Introduktion till problemområdet

Vid utveckling av informationssystem är det brukligt att använda en utvecklingsmodell. Modellen har till uppgift att ge en bild av de arbetsuppgifter som utvecklingen kräver och skall fungera som ett ramverk och stöd i arbetet. Modellen har normalt ett antal arbetssteg eller faser som täcker systemets hela livslopp.

Den modell jag har valt att refererar till i texten är livscykelmodellen (Anderssen, 1991). Jag valde livscykelmodellen som referensmodell för att den är enkel att förstå, tack vare sin sekventiella natur, och att den är en välbeprövad modell. Livscykelmodellen har fyra systemutvecklings faser; Analys, utformning, realisation och implementering. Modellen har dessutom en initial fas som kallas förändringsanalys och två avslutande faser; Förvaltning/drift även kallad underhållsfasen och avveckling. Dessa tre sistnämnda faser ingår dock inte i systemutvecklingsområdet enl. ovan nämnda källa, men jag tar ändå upp underhållsfasen då denna har stor betydelse för totalkostnaden för ett system.

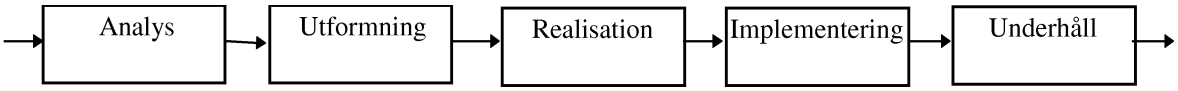


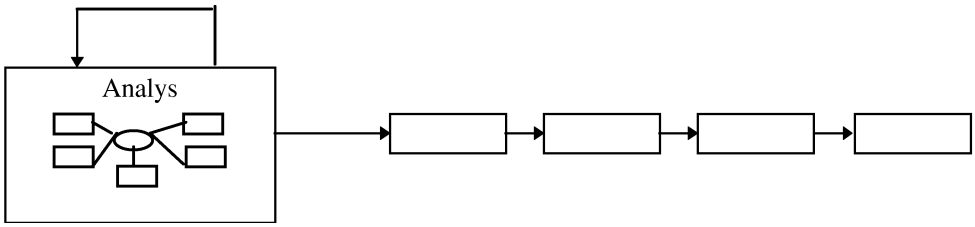
Fig 1: Livscykelmodellens systemutvecklingsfaser samt underhållsfasen.

Historiskt har analysfasen varit i skuggan av de övriga faserna av systemutvecklingen. I början av 1970- talet började dock utvecklare att få upp ögonen för analysen och förstå vikten av en grundlig analys för att säkra kvaliteten på det fortsatta systemutvecklingsarbetet samt på slutprodukten (Vonk, 1990). Studier som har gjorts visar att det kostar upp till 100 ggr mer att ändra fel i kraven efter leverans av systemet (VI, 1995). Enligt min mening indikerar detta vikten av en grundlig och fullständig analys för att hålla nere kostnader och projekttider.
Det skall understykas att även arbetet under utformningsfasen är viktig för att förändringar i systemet, efter implementeringen, skall vara enkla att genomföra och därigenom mindre kostsamma. Men genom en grundlig analys kan antalet förändringar minimeras.

Under den viktiga analysfasen, vilket ligger som grund för hela utvecklingsarbetet, analyseras verksamheten och vilka uppgifter det tänkta informationssystemet skall utföra. Resultatet av analysfasen är en kravspecifikation där kundens samtliga krav på det tänkta informationssystemet är dokumenterade (Anderssen, 1991). Kravspecifikationen kan ses som länken mellan analysfasen och utformningsfasen. Under analysfasen sker en kravhanteringsprocess där krav som kunden har på systemet insamlas, analyseras, dokumenteras, valideras, förvaltas och prioriteras (Karlsson, 1996).

2.1.1 Kravhantering

Kravhantering är en process som syftar till att identifiera och definiera egenskaper som ett informationssystem skall uppfylla. Denna process, som är en del i analysfasen, är iterativ, dvs den upprepas tills man har uppnått ett förväntat resultat. Analysfasen i den sekventiella livscykelmodellen är således av iterativ natur.

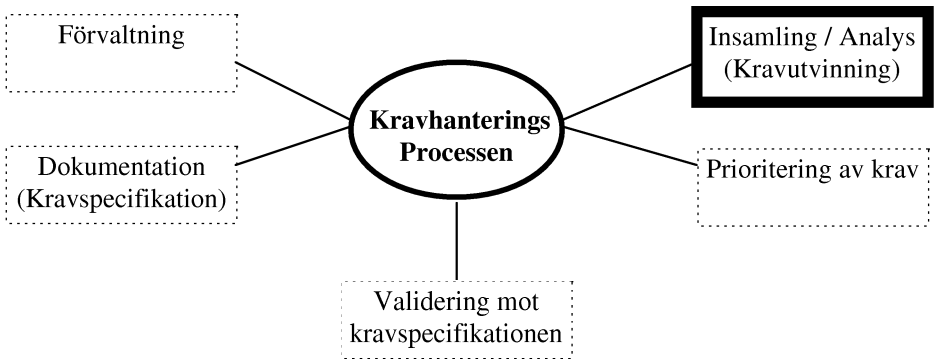


Figur 2: Analysfasen

Kravhanteringsprocessen innehåller ett antal olika aktiviteter för att på ett effektivt sätt hantera de krav som kunden kan tänkas ha på ett system (Karlsson, 1996).

Den primära och initiala aktiviteten i kravhanteringsprocessen är insamling av kundens krav (se fig 3). Denna aktivitet kan lätt missuppfattas då insamling för tanken till en trivial och snabbt genomförd aktivitet. Någon egentlig insamling, i ordets rätta bemärkelse, sker aldrig då kunden ofta inte kan uttrycka alla sina krav och behov samt att kommunikationsproblem lätt uppstår mellan kund och utvecklare pga olika referens- och kunskapsramar, anser jag. Vidare förändras ofta kundens krav under systemutvecklingens gång, vilket omöjliggör en direkt insamling av de dynamiska kraven. Kravutvinning är enligt mig en bättre tolkning av aktiviteten då kraven mognar och utvinns under processens gång dvs under en längre tid. Det är denna aktivitet som är central för detta arbete och är därför inramad i fig 3.

Priotitering av krav innebär att man rangordnar kraven i förhållande till hur viktigt det är för användaren eller kunden att kravet på det tänkta systemet uppfylls (Karlsson, 1996). Detta är speciellt viktigt då det finns tydliga tidskrav för systemleveransen och man inte tidsmässigt hinner med att uppfylla samtliga krav på systemet. Man säkerställer då genom prioriteringen att de primäraste kraven som användaren eller kunden har ställt på systemet uppfylls.



Figur 3: Kravhanteringsprocessen med fokusering på insamling / analys. (Karlsson, 1996)

När ett krav är funnet och prioriterat måste det dokumenteras. Detta görs normalt i kravspecifikationen. Syftet med att man dokumenterar samtliga krav och dess prioritet är i första hand att dokumentationen ligger till grund för den fortsatta utvecklingen. Dokumentationen fungerar också som ett kontrakt mellan kunden och utvecklaren om vilka krav som skall uppfyllas och realiseras i systemet. Kravspecifikationen används också vid valideringen av krav.

Validering är en aktivitet som i likhet med de övriga aktiviteterna sker fortlöpande under kravhanteringsprocessen. Validering innebär i korthet att man kontrollerar att man utvecklar rätt produkt utifrån kravspecifikationen, dvs att systemet uppfyller de krav som har dokumenterats i kravspecifikationen. Med valideringen säkerställer man att systemet motsvarar de förväntningar och krav kunden har på systemet (Karlsson, 1996).

Förvaltning innebär bl a att man gör förändrings- och påverkansanalyser (Karlsson, 1996) . Man analyserar hur förändringar i kraven kan och kommer att påverka systemet och det pågående arbetet. Man analyser också vilken tidsåtgång förändringen i kravet kommer att kräva och till vilken kostnad. Dessa aspekter vägs sedan emot nyttan av förändringen för systemet.

2.1.2 Kravutvinningsmetoder

En kravutvinningsmetod har alltid, ur utvecklarens synvinkel, ett antal önskvärda egenskaper. En sådan grundläggande egenskap är att en kravutvinningsmetod, på ett effektivt sätt, skall stödja utvinningen av krav som olika intressenter kan tänkas ha på det tänkta systemet. Att kravutvinningsmetoden är tidsbesparande, kostnadseffektiv, personalbesparande och enkel i användningen är ytterligare egenskaper som är önskvärda för metoden. Alla önskvärda egenskaper som inte uppnås genom metoden tolkar jag som brister i den aktuella metoden.

Det finns många olika metoder, mer eller mindre effektiva, för att utveckla krav. I litteraturen behandlas flera olika metoder. Bland de vanligaste förekommande finner man observation, ostrukturerade- och strukturerade intervjuer, brainstorming, protokollanalys och prototyping.

Observation innebär kort att utvecklaren observerar när användaren, i dess egna verksamhet, utför arbetsuppgifter och rutiner (Karlsson, 1996). Denna metod är enkel och användaren kan ägna sig åt sitt ordinarie arbete. Risken är dock att utvecklaren, om denne inte har kunskap om verksamheten, observerar och dokumenterar irrelevant information. Det är även stor risk att man inte får möjlighet att observera alla tänkbara arbetsuppgifter och situationer användaren har och kan hamna i. Ofta observeras bara de primära rutinerna i användarens arbete, vilket leder till en ofullständig kravbild för det tänkta systemet.

Intervjuer är mycket vanligt förekommande för kravutveckling och innebär att utvecklaren ställer, strukturerade eller ostrukturerade, frågor till användaren för att skapa sig en uppfattning av vad som är viktigt för användaren och vilka grundläggande krav som finns på det tänkta systemet. Svårigheten med intervjumetoder är, enligt mig, att ställa relevanta frågor och risken är stor att mycket tid spenderas på oviktiga entiteter och att viktig information som inte

efterfrågas försummas. Kommunikationsproblem mellan användare och utvecklare bidrar också till svårigheter pga de olika referens och kunskapsramar parterna har. Följden är att kunden inte kan uttrycka vad som krävs eller inte vet vad som går att kräva av systemet och utvecklaren förstår inte betydelsen av kraven som ställs eller tolkar dem på ett eget, erfarenhetsmässigt, sätt.

Brainstorming innebär att man samlar ihop ett antal användare och utvecklare för gemensam *idékläckning* (Karlsson, 1996). Metoden är effektiv för informationsinsamling men det krävs mycket engagemang av deltagarna för att uppnå ett fullgott resultat. Metoden kräver mycket efterarbete för sammanställning av resultatet som ofta är mycket ostrukturerat.

Protokollanalys är en metod där användaren utför en speciell arbetsuppgift och berättar högt hur hon eller han utför den. På detta sätt får utvecklaren kunskap om hur användaren går tillväga och vilken ordning uppgifter löses (Karlsson, 1996). Risken med metoden är enligt mig att användaren inte nämner alla arbetssteg som utförs eftersom de kan verka självklara och obetydliga för utvecklingsarbetet. Även med denna metod blir efterarbetet tidsödande då resultatet av analysen är ostrukturerat.

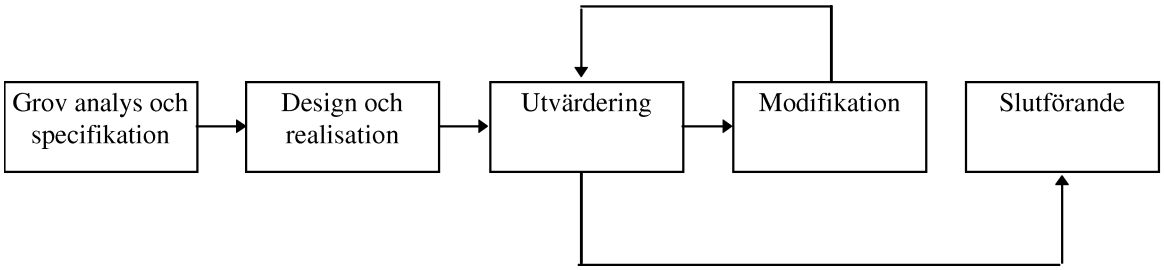
2.1.3 Prototyping

Den största bristen med de ovan nämnda metoderna är att användaren saknar något konkret att ta på och tala omkring vilket hämmar en genuin dialog mellan de båda parterna (Vonk, 1990). Prototyping är en metod som utnyttjar prototyper av det tänkta systemet för kravutvinning, vilket medför att användaren redan på ett tidigt stadie kan få se hur det framtida systemet, både funktionellt och designmässigt, kan fungera. Prototyping främjar på detta sätt både dialogen mellan kund och utvecklare och kundens förståelsen för systemet, lärande process, och dess möjligheter vilket leder till att kraven på systemet blir lättare att utvinna (Vonk, 1990).

Prototyping är inte någon ny metod. Prototyper har används i systemutvecklingsprojekt under lång tid men då enbart till för att utvärdera gränssnitt och designsfrågor. Genom 4-GL verktygens introduktion fick prototyping en ny innebörd vilket innebär att man på ett enkelt sätt kan göra prototyper för att både utvärdera gränsnittet och funktionaliteten på ett tänkt system.

Det finns två olika tillvägagångssätt vid användning av prototyping (Anderssen, 1991). Det första är att endast utnyttja prototyping för kravutvinning. När kundens samtliga krav är dokumenterade förkastas prototypen, vilken kan vara utvecklad med hjälp av ett separat designprogram, penna och papper eller genom ett 4-GL vertyg. Utvecklingen av det egentliga systemet påbörjas i en separat utvecklingsmiljö, med hänsyn till de krav som finns i kravspecifikationen. Detta tillvägagångssätt kallas throw-away prototyping (Vonk, 1990).

I det andra tillvägagångssättet, evolutionär prototyping, förkastas inte prototypen utan den fortsätts att utvecklas tills systemet är komplett och leveransfärdigt. Evolutonär prototyping är ett tillvägagångssätt som har blivit populärt vid utveckling av enklare informationssystem, eftersom 4-GL verktygen har blivit mer sofistikerade och kraftfullare.



Figur 4: Arbetsfaser vid prototyping (Vonk, 1990).

Oberoende av prototypingmetod görs en grov analys initialt, se figur 4, med tillhörande specifikation för att fånga de grundläggande kraven kunden och organisationen har på systemet. Därefter sker designen och realisationen av prototypsystemet med utgångspunkt från den specifikation som erhölls i det tidigare steget. Kunden gör sedan, tillsammans med utvecklaren, en utvärdering av prototypen (Vonk, 1990). Prototyping är en iterativ metod, vilket innebär att prototypen modifieras och utvärderas upprepade gånger tills ett tillfredsställande resultat är uppnått, dvs när samtliga, funktionella och icke funktionella krav på systemet är dokumenterade.

Med prototyping har kunden möjlighet att, vilket saknas i de övriga metoderna, utvärdera gränsnittet, och om stöd finns även funktionaliteten, på det tänkta systemet (Vonk, 1990). Ur ett kognitivt perspektiv är kraven på gränssnittet minst lika viktiga som de övriga kraven på systemet.

Den problematik som är knuten till prototyping är att användaren i stor utsträckning inte förstår att det endast är en prototyp av det tänkta systemet som utvärderas (Pressman, 1997). Förväntningarna kan därför bli höga på att leverans av systemet skall ske kort efter utvecklingsstarten trots att man endast är i utvecklingsarbetets början. Ytterligare ett problem är, enligt mig, att få tillgång till användare för att utvärdera prototypen under flera tillfällen. Deras ordinarie arbete kan göra det svårt att medverka vid utvecklingsarbetet. Det är således viktigt att alla parter förstår syftet med prototyper och prototyping, dvs ett stöd i kravutvinningen vilket minskar kostnaderna och ökar kvaliteten på slutprodukten.

2.2 Centrala begrepp

För att få en mer nyanserad och fördjupad kunskap om de som jag anser vara centrala begrepp i texten följer en beskrivning av dessa.

2.2.1 Krav

Det finns många olika definitioner på kravbegreppet men den som, enligt mig, är den mest kompletta och med det bredaste synsättet är; ”Ett krav är en önskvärd egenskap som har ett ursprung, ett motiv och ett realiseringsobjekt” (VI, 1995). Krav har enl definitionen tre beståndsdelar. Ursprung, motiv, realiseringsobjekt. Ett krav har alltid ett ursprung, dvs en kund, användare eller utvecklare som ställer kravet. Ett krav har även alltid ett motiv. Motivet kan vara att en användare har ett behov eller att konkurrenterna har de efterfrågade

egenskaperna i sina system. Slutligen har ett krav även ett eller flera realiseringsobjekt. Vid utveckling av informationssystem är det naturligt att systemet är detta objekt.

Krav kan delas in i två huvudtyper, funktionella- samt icke funktionella krav (Meldrum, 1993). Funktionella krav är krav på datainnehållet och krav på beteende hos systemet.
De icke funktionella kraven är mer egenskapsbetonade krav. Exempel på icke funktionella krav är (VI, 1995):

Gränssnittskrav, dvs hur kontroller, knappar och färger skall vara designade och placerade på skärmen. Prototyping ger ett mycket bra stöd för utvinning av gränssnittskrav. Det gäller både vid prototyping med sparat designverktyg och vid användning av 4-GL verktyg.

Prestandakrav, behandlar bl a hur långa svarstiderna får vara, krav på systemets säkerhet och systemets beräkningsnoggrannhet. Genom att använda 4-GL verktyg med kraftfulla programmeringsspråk kan, enligt mig, prototyping vara ett stöd för utvinning av prestandakrav, då användaren kan få en uppfattning om vad som är möjligt att kräva av systemet.

Livcykelkrav, vilket är krav på bl a systemets underhållbarhet och flyttbarhet. Det finns inget stöd i literaturen för att prototyping skulle vara en effektivare metod än de övriga metoderna för utvinning av dessa krav. Min åsikt är att livcykelkrav är svåra att utvinna oberoende av metod. Det gäller med andra ord också för prototyping.

Användbarhetskrav, exempelvis krav på att systemet skall vara lätt att lära och lätt att använda. Prototyping är ett bra stöd för utvinning av denna typ krav. Genom att användaren utvärderar prototypen iterativt blir prototypingen i sig en lärande process om systemet.

Komersiella krav, är krav på utvecklingskostaden , leveranstiden av systemet och systemets livslängd. Dessa krav är mycket svåra att utvinna med hjälp av prototyping pga prototypingmetodens iterativa natur. Eftersom antalet iterationer, dvs modifikationer av prototypen, är svåra att uppskatta är krav på leveranstid och kostnad mycket vanskliga att uppfylla.

Vid kravställande, vilket är ett problem, fokuseras vikten ofta på de funktionella kraven. Skälet är att de ofta är tydligare och mer mätbara än de icke funktionella kraven (VI, 1995). Genom prototyping får man fram funktionella krav på ett effektivt sätt genom att prototypen utvecklas och utvärderas kontinuerlig. Även vissa icke funktionella krav kan utvinnas genom prototyping då dessa växer fram under den iterativa processens gång (Vonk, 1990).

2.2.2 Informationssystem

Ett informationssystem är ett system för insamling, bearbetning, lagring, överföring och presentation av information (Anderssen, 1991). Vidare betonas att ett informationssystem kan både vara manuella och datoriserade. Även en kombination av dessa kan förekomma i en verksamhet. När informationssystem nämns fortsättningsvis i texten är det de datoriserade systemen som åberopas. Informationssystem kan delas in i två olika kategorier; affärs- eller administrativa informationssystem och tekniska system (Vonk, 1990). Administrativa system, vilka även kallas dataorienterade, kännetecknas av stora informationsmängder och krav på enkla processer och uträkningar. De tekniska systemen har det motsatta förhållandet, dvs relativt små datamängder men komplexa processer som kan har tidskrav, då realtidsaspekten ofta är viktig för denna typ av informationssystem. De kallas därför processorienterade system.

Jag tycker att det är viktigt att betona att det skall vara verksamheten som påverkar informationssystemet och inte tvärtom, dvs att en verksamhet får anpassa sig efter ett informationssystem. Detta är en risk då de behov och krav en verksamhet har på ett system inte uppfylls till fullo.

2.2.3 4-GL verktyg

Prototyping är ett teknikdrivet arbetsätt vilket innebär att det är viktigt att förstå den bakomliggande teknologin för att förstå prototyping (Vonk, 1990).
Den första generationen av programmeringsspråk som användes var i maskinkods form. Det innebar att man kodade på bit-nivå genom ettor och nollor i olika sekvenser. Detta var ett mycket tidskrävande och expertdominerat arbetssätt (Beekman, 1994).

På 1950-talet utvecklades den andra generationens programmeringsspråk; assembler. Logiskt sett fungerar assembler och maskinkod på samma sätt men det var lättare att förstå, läsa och skriva kod i assembler då den inte var i binär form utan i alfabetisk form. Vidare användes symboliska adresser i stället för maskinkodens fysiska hårdvaruadresser vilket förenklade arbetet ytterligare (Beekman, 1994).

På 1960-talet introducerades ett antal språk som med ett samlat namn kallas för den tredje generationens programspråk. FORTRAN och COBOL var pionjeerna för den tredje generationens språk och därefter kom Pascal, C och Basic som fortfarande används i näringslivet och i utbildningssyfte. Det som förändrade arbetet med den tredje generationens språk var kompilatorn. Kompilatorn översätter den aktuella koden till maskinkod vilket medför att man kan ha en mer logisk terminologi och notation på det *högre språket*. Genom detta blev arbetet med att utveckla kod mindre tidskrävande och felsökning blev lättare. Dessutom blev programmen i viss mån flyttbara mellan olika maskiner vilket inte hade varit möjligt i de tidigare generationernas språk (Beekman, 1994).

Den fjärde generationens programspråk är snarare ett verktyg än ett programspråk. Verktyget kan ses som ett skal som har byggts runt ett befintligt programspråk och verktyget fungerar därför som ett gränssnitt mellan användaren och språket. Verktyget ger stöd för gränssnitts utformning samt en dialog mellan användare och system. Detta underlättar arbetet avsevärt då man skall utveckla enklare applikationer då mycket av de tidigare tidskrävande uppgifterna är automatiserade i verktyget. Om man skall utveckla dynamiska och mer funktionellt avancerade prototyper eller applikationer i den fjärde generationens utvecklingsverktyg krävs att man kodar manuellt i det språk som utvecklingsverktyget har i botten.

Genom att använda fjärde generationens utvecklingsverktyg får man dock stora tidsvinster vid utveckling av system, vilket reducerar kostnaderna, samt att man på kort tid kan bygga prototyper som kan utnyttjas för kravutvinning (Anderssen, 1991).

2.2.4 Kvalitet

En definition av begreppet kvalitet är svår att ge, då kvalitetsbegreppet är subjektivt och kan således tolkas på flera sätt. En produkt kan uppfattas ha hög kvalitet i en verksamhet medan en annan verksamhet har en helt annan uppfattning om produkten (Perry, 1991).
Dock har flera försök gjorts för att definiera kvalitetsbegreppet inom systemutvecklingsområdet. Under min utbildning har jag kommit i kontakt med och noterat följande definitioner på kvalitetsbegreppet:

- ”Kvalitet är detsamma som samstämmigheten mellan det färdiga informationssytemet och kravspecifikationen.”

- ”Överenstämmelsen mellan ett färdigt system och de förväntningar kunden har på systemet.”
  Ju högre sammstämmighet ju högre kvalitet har systemet

- ”Tekniskkvalitet * acceptans hos kunden = system kvalitet”

Kunden har också ett mått på vad kvalitet är bl a:

- Överensstämmelse med verksamhetens mål, arbetssätt och information.
- Enkelhet i användningen av systemet.
- Support till kunden.

Genom att använda en effektiv kravutvinningsmetod, exempelvis prototyping, kan fler krav på det tänkta systemet realiseras, vilket resulterar i högre kvalitet och nöjdare kunder.

3 Problemområdes beskrivning

När prototyping utnyttjas för kravutvinning vid utveckling av informationssystem är syftet att höja kvaliteten på slutprodukten samt på utvecklingsarbetet och därigenom sänka kostnaderna på utvecklingsarbetet, vilka framförallt är knutna till dyra förändringar av systemet efter implementationen. Kostnaderna som är knutna till själva prototypingmetoden är dock högre än till andra, mindre teknikdrivna, kravutvinningsmetoder (Vonk, 1990). Det är även viktigt att betona att prototyping inte är någon garanti för att man lyckas utvinna kundens samtliga implicita och explicita krav på det tänkta systemet, och där igenom höja kvaliteten på systemet.

En kravutvinningsmetod har alltid, ur utvecklarens syn, ett antal önskvärda egenskaper. En primär och önskvärd egenskap är att metoden, på ett effektivt sätt, skall stödja utvinningen av samtliga krav som olika intressenter har på systemet. Att metoden är tidsbesparande, enkel att använda, billig, dvs genererar inga merkostnader samt personalsbesparande är ytterligare egenskaper som är önskvärda för en metod. Alla önskvärda egenskaper som inte uppnås genom metoden tolkar jag som brister i den aktuella metoden.

Att alla önskvärda egenskaper inte uppnås, dvs att det finns brister i en metod, kan enligt mig bero på flera faktorer. För det första kan bristerna ha sin grund i metodens naturliga beskaffenhet. En metod som i sin natur är personalintensiv kan inte uppfylla den önskade egenskapen att vara personalbesparande. För det andra kan typen av informationssystem som skall utvecklas och vissa speciella situationer medföra att metoden får brister och inte är ett lämpligt metodval.

3.1 Avgränsning

Min avsikt är inte att göra en jämförande studie mellan olika kravutvinningsmetoder som existerar idag, utan istället avgränsa min analys till prototypingmetoden och undersöka dess begränsande egenskaper samt undersöka i vilka situationer dessa begränsningar uppkommer och hur typen av informationssystem påverkar uppkomsten. Vidare skall jag undersöka vilka faser under systemutvecklingen som påverkas tillföljd av brister i prototypingmetoden och tillsist utföra en bedömning när det är lämpligt att utnyttja prototyping för kravutvinning med avseende på totalkostnaden för systemutvecklingsprojektet, dvs inklusive underhållskostnader. Med prototyping avser jag throw-away prototyping som stöds med hjälp av 4-GL verktyg.

3.2 Problemdefinition

De tre första punkterna i min problemdefinition skall utmynna i den fjärde punkten som är en frågeställning som bygger på resultaten från de övriga frågeställningarna.

- Undersöka vilka egenskaper som är begränsande för prototyping, dvs ta reda på vilka brister som finns med prototyping.

- Undersöka hur typen av informationssystem påverkar lämpligheten av prototyping.

- Ta reda på vilka faser under systemutvecklingen som påverkas tillföljd av dessa brister.

- Utföra en bedömning när det kan vara lämpligt att använda sig utav prototyping med avseende på totalkostnaden för ett projekt.

3.3 Förväntat resultat

Bland de egenskaper som är begränsande för användningen av prototyping tror jag mig finna den återkommande utvärderingen kunden eller användaren gör under prototyping. Detta förfarande kräver att användaren alternativt kunden under ett flertal tillfällen måste *släppa* sitt ordinarie arbete vilket inte är möjligt alla gånger, då den primära verksamheten ofta prioriteras. Vidare återfinns med stor sannolikhet även den ökade kostnaden för kravutvinning med stöd av prototyping som en begränsande egenskap. Prototyping är en förhållandevis kostnadsintensiv process, speciellt då prototyping endast används för kravutvinning. Detta kan troligen, särskilt i mindre projekt, vara en faktor som avgör om prototyping används som kravutvinningsmetod eller ej.

De faser under systemutvecklingsarbetet som kan tänkas påverkas av bristerna i prototyping är troligen, mer eller mindre, samtliga faser. En ofullständig kravbild genererar troligen följdfel genom hela utvecklingscykeln och utmynnar i en förhöjd underhållskostnad pga dyra förändringar i det färdiga systemet.

Resultatet av kostnadsbedömningen blir troligen att prototyping är lämplig att använda i de flesta fall då osäkerheten i kravbilden är hög. Dock tror jag att typen av informationssystem som skall utvecklas spelar en viss roll för effektiviteten och därmed för lämpligheten av prototyping. Om ett administrativt informationssystem skall utvecklas och tillgängligheten till användare är god tror jag att prototyping är ett bra val i förhållande till totalkostnaderna för projektet.

4 Metod

4.1 Möjliga metoder

Det finns flera möjliga metoder för att få fram ett adekvat resultat för den problemställning jag har definierat tidigare. De metoder jag har övervägt att använda för fakta- och informationsinsamling är följande:

- Fallstudie
- Litteraturstudie
- Intervjuer
- Enkäter

För att förstå mitt val av metod följer en förklaring av samtliga metoder samt deras, enligt mig, för- och nackdelar som stöd för informationsinsamling under arbetet.

4.1.1 Fallstudie

En fallstudie innebär att man studerar ett eller flera enskilda fall av den företeelse man är intresserad av. I detta fall skulle ett eller flera systemutvecklingsprojekt som har använt sig utav prototyping för kravutvinning studeras.

Jag anser att en nackdel med fallstudier utifrån min problemställning är att den kräver en studie av många fall för att kunna fungera som underlag i arbetet och ge svar på min problemställning. En punkt i min problemställning är att undersöka vilka brister prototyping har, detta kan man inte få en korrekt uppfattning om ifall inte dessa brister har uppmärksammats i ett flertal fall. Det kan också vara svårt att få tillgång till material som grund för en fallstudie. Företag är försiktiga med att lämna ut uppgifter för att skydda sina klienter samt egna intressen.

Fördelarna med fallstudier utifrån min problemställning är att man får en onyanserad bild av verkliga systemutvecklingsprojekt och de brister som har uppkommit. Dvs när man gör en fallstudie är det man själv som tolkar vad som har skett och ingen annan.

4.1.2 Litteraturstudie

Litteraturstudier är vanligt förekommande i de fall då det finns mycket dokumenterad information om ämnet, både i rapport, artikel och bokform.

Nackdelarna med litteraturstudier är att det krävs mycket sökande för att hitta relevant information med hänsyn till problemdefinitionen. Prototyping är ett mycket stort ämne och finns representerat i många vetenskapliga sammanhang och mitt arbete behandlar enbart en mycket liten del av detta. Dock har det på senare tid förenklats genom möjligheten till sökning i databaser och via Internet. Inom det område mitt arbete behandlar finns det både litteratur och dokumentation som stödjer arbetet vilket är en förutsättning för att denna metod skall vara ett alternativ. I litteraturstudier måste man också tänka på, vilket är en nackdel, att författaren av dokumentationen kan ha nyanserat bilden då hans eller hennes tolkningar och egna erfarenheter speglar innehållet. Ytterligare en tänkbar nackdel med litteraturstudier är att den litteratur som är tillgänglig kan vara några år gammal. Utvecklingen inom det område jag

behandlar är som bekant ett mycket expansivt och dynamiskt område som ständigt är i förändring. Därför måste man troligen sålla i den tänkbara litteraturen för att få en bild av prototyping som är relevant för nuläget.

Fördelen med litteraturstudier är, enligt mig, att man kan få svar på sin problemställning ur olika perspektiv och synsätt. Prototyping för kravutvinning återfinns både i litteratur som behandlar requeriments engineering, vilket behandlar de aktiviteter som krävs för att förstå och specificera användarnas krav, och software engineering, vilket har sin tyngdpunkt på de senare faserna i utvecklingscykeln. Detta bidrar till att man får en bred och vidsynt uppfattning om brister och fördelar med användningen av prototyping. Man kan dessutom få ta del av undersökningar som författaren redan har genomfört eller refererar till på ett strukturerat och sammanfattat vis, vilket underlättar arbetet.

4.1.3 Intervjuer

Det finns två olika huvudtyper av intervjuer. Strukturerade och ostrukturerade intervjuer. Strukturerade intervjuer innebär att man som intervjuledare i förväg har skapat de frågor man skall ställa till den person man intervjuar. En ostrukturerad intervju är en intervju som inte har några fördefinierade frågor utan mer liknar ett samtal som intervjuaren kan styra mot de frågeställningar han eller hon har.

Problemet med metoden, speciellt i detta fall, är svårigheten att få kontakt med rätt personer för intervju. Med rätt person i detta fall är de som har erfarenheter och medverkat vid systemutvecklingsprojekt med prototyping som kravutvinningsmetod. Dessutom är risken, enligt mig, att den intervjuade svarar på ett nyanserat sätt pga deras egna värderingar eller tidigare erfarenheter. Därför kräver denna metod att man genomför många intervjuer för att få en övergripande bild över bristerna och dess verkningar under prototyping. Om man dessutom använder sig utav ostrukturerad intervjuteknik är det mycket information att sammanställa innan man kan dra några slutsatser när prototyping är lämpligt eller ej.

Fördelen med att använda sig utav intervjutekniken är att man under en interjuv kan göra förtydliganden i sina frågeställningar samt få en diskussion runt frågeställningarna vilket inte är möjligt i de övriga metoderna. Om man dessutom gör en strukturerad intervju, vilket är fullt möjligt då jag har väl definierad problemställning, blir det relativt enkelt att sammanställa resultatet, då samtliga intervjuade svarar på samma frågor.

4.1.4 Enkäter

En enkätundersökning baseras på att man skapar en eller flera blanketter på vilka man definierar frågor som man vill ha svar på. Dessa blanketter skickas sedan ut till en viss målgrupp som svarar på frågorna. När man har fått tillbaka de ifyllda blanketterna sammanställs resultatet av undersökningen.

Det finns dock några nackdelar med valet av enkätundersökningar. För att få fram ett säkert resultat ur statistiksynpunkt på de frågeställningar man har i enkäten måste man ha en mängd enkäter för att sammanställa information utifrån. Problemet är dock att svarsfrekvensen är låg, dvs man får inte tillbaka samma antal enkäter som man delar ut. Svarstiden blir därför lång då man med stor säkerhet måste skicka påminnelser för att få in flera enkätsvar. Ytterligare en nackdel med enkäter utifrån min problemställning är att skapa frågorna. I enkäter skall frågorna

i största mån vara av typen ”välj alternativ”. Att ha frågor som kräver mycket långa svar är därför inte lämpligt. En punkt i min problemställning är att undersöka bristerna med prototyping. För att kunna använda enkäter för att få svar på detta måste jag skapa en fråga som har ett antal brister som alternativ att välja på. Detta är praktiskt omöjligt då jag i förväg inte vet vilka brister som finns eftersom det är det jag skall undersöka.

Fördelen med enkäter är att man får mycket konkreta svar som därför inte kan misstolkas i resultatsammanställningen. Dessutom behöver man inte vara engagerad i enkäten mellan utskick och sammanställning vilket ger möjlighet till att komplettera undersökningen genom att använda andra metoder under den tidsrymden.

4.2 Val av metod

Utifrån den analys jag har gjort av de möjliga metoder som går att använda för att få svar på mina problemställningar har jag kommit fram till att jag skall använda metoden litteraturstudie för fakta och informationsinsamling. Valet av litteraturstudie grundar sig dels av att det har skrivits mycket om prototyping för kravutvinning vilket gör det möjlig att utföra denna typ av studie samt att det är extra intressant då det finns, som jag nämnde tidigare, olika skolor och därmed infallsvinklar till ämnet. Litteraturstudier är, enligt mig, också ett relevant val för informationsinsamling under arbetet då jag inte gör anspråk på att skildra några absoluta sanningar utan finna ett mönster eller en genomgående trend när prototyping är lämplig för kravutvinning i ett systemutvecklingsprojekt eller ej.

Jag har valt att inte använda fallstudier som metod då jag inte vill knyta min undersökning till något eller några specifika företag. Även att metoden kräver att man undersöker ett antal projekt för att få fram ett relevant resultat påverkade mitt val eftersom det, vilket jag nämnde i min metodanalys, kan vara problematiskt.

Intervjuer var den metod som jag in i det sista övervägde att använda. Dock vägde de nackdelar jag redogjorde för i metodanalysen något tyngre än de fördelar metoden bevisligen har. Det var framför allt svårigheten i att få tag i rätt människor som avgjorde att jag inte valde att använda mig utav intervjuer.

Huvudorsaken till att jag inte har valt att använda enkäter till min studie är att jag, som jag nämnde tidigare, inte har underlag för att skapa korrekta och entydiga frågor.

4.3 Källor

De källor jag har valt att använda då de är relevanta i avseende på min problemställning är följande:

Vonk, R. (1990), Prototyping
Pressman, R. S. (1997), Software engineering
Anderssen, E. S. (1991), Systemutveckling–principer, metoder, tekniker
Martin, J. (1991), Rapid application development
Karlsson, J. (1996), Framgångsrik kravhantering - nr 2
Ould, M. A. (1990), Strategies for software engineering

Jag anser att Vonk (1990) är en relevant källa i avseende på min problemställning då den uteslutande behandlar ämnet prototyping för kravutvinning under systemutvecklingsprojekt. Den beskriver de olika typerna av prototyping som kan används och vilka för och nackdelar användning av prototyping medför ett systemutvecklingsprojekt. Även jämförelser mellan prototyping och andra beprövade ansatser samt beskrivning av case-verktygens betydelse för prototyping ingår i boken.

Jag har valt Pressman (1997) som källa i mitt arbete då han i ett kapitel i sin bok tar upp prototyping som en metod för kravutvinning under analysfasen. Pressman (1997) tar upp frågeställningen när prototyping är lämplig att användas samt, i likhet med Vonk (1990), vilka olika typer av prototyping och verktyg som finns att tillgå. Boken tar även upp ämnen som projektledning, kvalitetssäkring, objektorienterad programvaruutveckling samt olika design- och testmetoder.

Boken Andersen (1991) är en grundläggande bok i ämnet systemutveckling. Boken är ändå relevant i avseende på min problemställning då den tar upp kriterier när prototyping är en lämplig och mindre lämplig metod för kravutvinning, i likhet med Pressman (1997), samt grundtankar och metodsteg för prototyping.
Boken ger en mycket bra helhetsbild av systemutveckling och ger förståelse för varför analysfasen och kravhantering är avgörande för resultatet på slutprodukten.

I rapporten Karlsson (1996) beskrivs kravhanteringsprocessen, kravbegreppet samt beprövade ansatser för kravhantering. Jag anser att rapporten är relevant som källa då den tar upp begreppet prototyping samt beskriver dess brister och tillgångar vid kravutvinning för olika användargrupper och projekt.

I Martin (1991) beskriver författaren en livcykelmodell, RAD, som kan förkorta ned utvecklingstiden och där igenom sänka kostnaderna för systemutveckling. Boken behandlar även prototyping och tar bl a upp vilka faror det kan innebära att använda prototyping och vilka informationssystem som är lämpliga alternativt olämpliga att göra prototyper av för stöd i kravhanteringen.

I likhet med Pressman (1997) är Ould (1990) en bok som behandlar projektledning och kvalitetssäkring under systemutveckling samt olika typer av utvecklings strategier. Jag har valt att ta med boken som källa för dess beskrivning av de brister prototyping har och hur prototyping påverkar de inblandade aktörerna under ett projekt.

I bland de källor som jag har redovisat noterar den uppmärksamme läsaren att det endast är en av dessa som tar upp prototyping som en isolerad företeelse. Litteratur som enbart behandlar prototyping för kravutvinning under systemutveckling är inte speciellt utbredd. Det finns dock en stor mängd litteratur som i samband med beskrivningar av olika systemutvecklingsmetoder och kravhantering som behandlar prototyping för kravutvinning. Intressant att notera är att samtliga av de nämnda källorna har en positiv attityd för användning av prototyping för stöd i kravutvinningen.

5 Genomförande

5.1 Sökmetoder

Under min litteraturstudie har jag använt olika metoder för att hitta litteratur, information och fakta som kan ses relevant med hänsyn till min problemställning. Metoderna har varit sökning i databasen LIBRIS, sökning på Internet samt genomgång av befintlig och relevant litteratur på Högskolan i Skövde.

Information och faktasökning på Internet gav ett dåligt resultat. Den mesta informationen om företeelsen prototyping behandlade produktkonstruktion och produktdesign och inte som kravutvinningsmetod för utveckling av informationssystem. Detta är ett återkommande problem som även omfattar de övriga sökmetoderna. De träffar i sökningen som behandlade prototyping under systemutveckling var av mycket grundläggande natur, vilket inte gav något stöd i arbetet.

LIBRIS är en databas som innehåller samtliga publicerade artiklar, rapporter och verk som finns representerade på de Svenska forskningsbiblioteken. Jag utnyttjade denna databas flitigt för att hitta lämplig och relevant litteratur för arbetet. Antalet träffar vid sökningen var av överskådlig karaktär. Den litteratur som jag har använt under arbetet har jag till stor del funnit genom detta förfarande.

Ett ytterligare problem som jag uppmärksammade under genomförandet var att det fanns mycket litteratur som behandlade systemutveckling på ett mer övergripande sätt men också innehöll information om prototyping som stöd under kravutvinningen. Dessa böcker gav inga träffar i LIBRIS. I stället letade jag information om prototyping i böckerna under ämneskategorin systemutveckling på studentbiblioteket, på grundval av dess titel.

5.2 Källinformation

5.2.1 Presentation

Jag har grupperat källornas olika ståndpunkter och kommentarer efter respektive problemställning för att skapa en överskådlig bild.

5.2.2 Undersöka vilka egenskaper som är begränsande för prototyping.

En av punkterna i min problemdefinition var att undersöka vilka brister som finns med prototyping. I den litteratur som jag har redovisat finns en mängd uppfattningar om vilka brister som prototyping har och som man bör ha i åtanke innan man använder metoden i ett systemutvecklingsprojekt. För att få en strukturerad bild över de brister som prototyping har enligt litteraturen följer nu en beskrivning av källornas synpunkter och där efter en uppdelning av bristerna kategorivis, vilket jag har gjort med utgångspunkt från källornas gemensamma uppfattningar.

I Martin (1991) betonar författaren att det finns brister och faror att använda prototyping vilket man måste vara väl medveten om innan man påbörjar ett systemutvecklingsprojekt. En av dessa är att prototyping ofta inte täcker kravutvinning för vissa icke funktionella krav och aspekter. De krav och aspekter som åsyftas är återhämtning efter systemet har gått ned,

säkerhetsaspekter, lätthet i underhåll av systemet, fleranvändarstöd, volymkörningar med responstidskrav, nätverksfrågor, flyttbarhet mellan olika plattformar.
En fara med prototyping är enligt författaren att användarnas förväntningar kan bli för höga. Användarna tror ofta att systemet skall vara färdigt omedelbart eftersom prototypen fungerar som det framtida systemet utan att förstå att det mesta av arbetet återstår för att få ett stabilt system. Vidare menar författaren att prototyping stimulerar användarna att ständigt förändra sina krav samt finna nya så att prototypen tar mycket lång tid att slutföra. Prototyping är även ett redskap för utbildning. De användare som är med under utvecklingsarbetet får samtidigt färdigheter och träning på det framtida systemet innan det är levererat.

I Ould (1990) skriver författaren att prototyping är svårt att använda om användarna som skall utvärdera prototypen inte har någon tidigare datorvana. Vidare menar författaren att en stor nackdel med prototyping är att det är svårt att göra en kostnads- och tidsbedömning på projektet. Detta beror på att antalet iterationer är svåra att avgöra på ett tidigt skede i projektet. Författaren skriver också att användarna i ett projekt med prototyping som kravutvinningsmetod måste vara mycket engagerade för att man skall lyckas med projektet.

I boken Pressman (1997) betonar författaren att användarna måste var tillgängliga under hela processen för att kunna utvärdera prototypen kontinuerligt så att inte några avbrott i processen uppkommer tillföljd av detta. Vidare betonar Pressman (1997) att det krävs att verktyg är tillgängliga för att kunna utveckla prototyper samt att utvecklaren måste ha stor kunskap i verktygen som används vid prototyping för ett lyckat resultat.

Prototyping kommer särskilt ifråga om man står inför användargrupper som har liten erfarenhet av adb anser Andersen (1991). Adb står i detta fall för automatisk data behandling. Risken med prototyping menar författaren är att användarna ägnar för mycket tid åt detaljer istället för att se på de riktiga problemen som existerar.

I Vonk (1990) menar författaren att användarmedverkan under prototypingen är mycket intensiv och kräver mycket av användarens tid i anspråk. Prototyping kräver mycket mer användarmedverkan än i de traditionella metoderna för kravutvinning.
Vidare menar författaren att kostnaden för prototyping beror på antalet iterationer som behövs för att nå en av användarna accepterad prototyp. Antalet iterationer är i sin tur relaterade till osäkerhetsgraden i kravbilden vid början av projektet. Det som påverkar osäkerhetsgraden är bland annat användarnas kunskapsnivå och deras tidigare erfarenhet, risken för kommunikationsproblem samt härledningsgraden av krav från uppgiften som skall stödjas. Även om man använder avancerade verktyg menar författaren att kostnaderna vid prototyping blir högre än vid de övriga metoderna för kravutvinning. Dock kan totalkostnaden för projektet bli lägre om osäkerheten i kravbilden är hög. Detta pga att underhållskostnaderna blir reducerade då fler av användarens krav kan utvinnas.
Det finns ett direkt samband mellan osäkerheten i kravbilden och underhållskostnaden. Den största vinsten med användning av prototyping blir vid en hög osäkerhet i kraven.
Det är dock lätt att fastna i detaljer vid prototypingförfarandet och att iterationerna då blir orimligt många. Om man skall använda prototyping så får det inte finnas några konflikter om vilken funktionalitet systemet skall ha eller om dess externa design mellan olika intressenter. Dock är konflikter mellan användarna under kravutvinningen normalt.
Prototyping är inte heller ett alternativ för kravutvinning om man skall skapa ett system i en mycket dynamisk miljö där det sker snabba förändringar, då kraven och tillika kravspecifikationen är frysta efter analysfasen.

5.2.3 Undersöka hur typen av informationssystem påverkar lämpligheten av prototyping

När jag undersökte hur typen av informationssystem som skall utvecklas påverkar lämpligheten av att använda prototyping eller ej fann jag att litteraturen hade en mycket samstämmig mening om detta.

I Vonk (1990) säger författaren att det är en missuppfattning att det inte är lämpligt att använda prototyping för kravutvinning vid utveckling av stora och komplexa system. Prototyping är lika lämpligt för stora som för små system, enl författaren. Informationssystem kan enligt författaren delas in i två olika kategorier; affärs- eller administrativa informationssystem och tekniska system. Administrativa system, vilka även kallas dataorienterade, kännetecknas av stora informations mängder och krav på enkla processer och uträkningar. De tekniska systemen har det motsatta förhållandet, dvs relativt små datamängder men komplexa processer som kan har tidskrav, då realtidsaspekten ofta är viktig för denna typ av informationssystem. De kallas därför processorienterade system. Han skriver att tekniska system inte är lämpliga för prototyping då de tillgängliga verktygen inte stödjer dessa system. Verktygen är bättre anpassade för de dataorienterade systemen som lägger stor vikt vid användargränssnittet och har on-line karaktär.

I Anderssen (1991) skriver författaren att prototyping rekommenderas särskilt vid transaktionsorienterade system vilka präglas av en omfattande dialog mellan människa och maskin. Adb-system med liten interaktion mellan människa och maskin och där databehandlingen består av fastlagda räkneregler är ej lämpliga för prototyping. Dessa system hör ofta hemma i vetenskapliga och akademiska världen.

Pressman (1997) hävdar att om systemet som skall utvecklas är komplext och kräver tusentals rader av kod för att skapa en protoyp är prototyping inte ett alternativ. Möjligen för vissa valda delar av systemet som har hög osäkerhet i kraven. Författaren hävdar också att prototyping skall användas då kraven är instabila på det tänkta systemet.

I Martin (1991) betonar författaren att prototyping inte är lämpligt för logiskt intensiva system utan skall användas vid vid kravutvinning för interaktiva system. Författaren nämner också att prototyping är lämpligt vid osäkerhet i kraven.

5.2.4 Ta reda på vilka faser under utvecklingen som påverkas tillföljd av dessa brister.

Gemensamt för samtliga brister som källorna har redovisat är att de primärt påverkar arbetet i analysfasen. Det beror på att prototyping av throw-away typ är en metod för att effektivt finna de krav användare kan tänkas ha på ett informationssystem som skall utvecklas. Om man som utvecklare inte tar hänsyn till de brister eller de begränsande egenskaper som prototyping har påverkas i första hand således analysfasen i negativ bemärkelse. Se fig 1, sidan 3.

I rapporten Karlsson (1996) betonar författaren att det krävs omfattande omarbeten vid upptäckt av fel i kraven. De omarbeten som måste göras är bl a att dokumentationen måste uppdateras, designen ändras och programkod korrigeras.

Det är framför allt är underhållskostnaden som minskar vid användning av prototyping hävdar Vonk (1990). Detta beror på, enligt författaren, att fler av användarnas krav kan tillmötesgås med prototyping än med de övriga metoderna. Om kravbilden är osäker blir minskningen av underhållskostnaden ännu större och differensen mellan den ökade kostnad prototyping

genererar under kravutvinningen i analysfasen och kostnadsminskningen i underhållsfasen ökar. Kostnadsbilden för prototyping skiljer sig därför från de övriga metoderna. Se fig 4

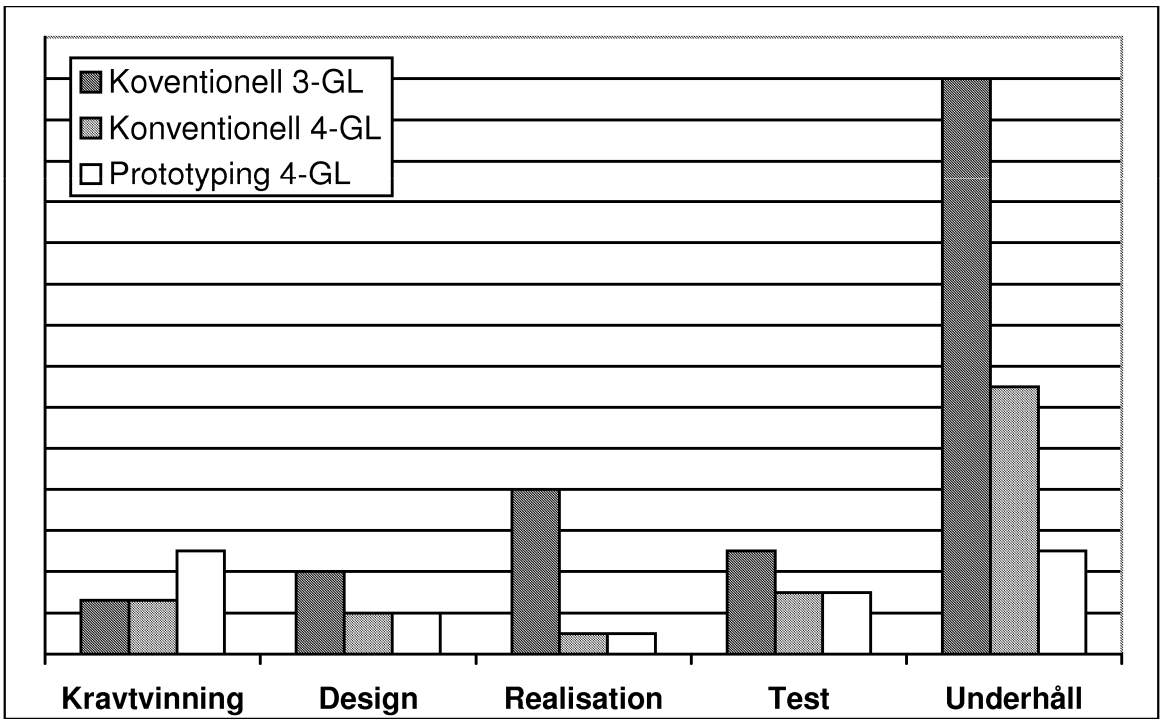


Fig. 4: Förändring i kostnadmönstret vid prototyping, Vonk (1990)

Vonk (1990) skriver vidare att om man inte tar hänsyn till de brister som prototyping har kan följden bli att kravutvinningen med stöd av prototyping inte blir effektivare än någon av de övriga metoderna.
Genom att analysfasen påverkas i negativ riktning blir följden en ofullständig kravspecifikation för informationssystemet. Den ofullständiga kravspecifikationen som ligger till grund för designen och realisationen av systemet bidrar med att ofullständigheten i kravbilden fortplantar sig vidare i dessa faser. Detta resulterar i ett system som inte användarna är nöjda med, eftersom deras krav inte är realiserade i systemet. För att tillmötesgå användarnas krav måste ändringar och korrektioner göras i programkoden och designen samt uppdateringar av dokumentationen. Dessa korrektioner i underhållsfasen är mycket kostsamma och tidskrävande. Följden av detta är att underhållskostnaderna inte blir lägre än vad de skulle ha blivit med någon av de övriga metoderna och detta resulterar i att totalkostnaden för projektet rusar i höjden.

Min uppfattning är att bristerna eller de begränsande egenskaperna för prototyping påverkar samtliga faser i utvecklingscykeln då kravutvinningen och dess slutprodukt ligger till grund för hela det fortsatta utvecklingsarbetet.

5.2.5 Utföra en bedömning när det kan vara lämpligt att använda prototyping

För att kunna göra en helhetsbedömning när det kan vara lämpligt att använda sig utav prototyping måste man ta hänsyn till samtliga faktorer som kan påverka denna. En ytterligare faktor som är avgörande för lämpligheten att använda prototyping för kravutvinning är användarnas kunskapsstruktur (Karlsson, 1996).

Kunskapen som användarna besitter kan enligt författaren delas in i fyra typer:

- Framtida kunskap
- Tillgänglig kunskap
- Igenkänd kunskap
- Dold kunskap

Framtida kunskap är sådan kunskap som användarna ännu inte besitter. Detta är fallet vid utveckling av helt nya informationssystem då användaren inte vet vilka krav som är relevanta att ställa samt att det är svårt att förmedla kraven till utvecklaren. För att få fram denna typ av kunskap och krav är prototyping mycket lämpligt. Genom prototyping kan man simulera och visualisera det framtida systemet vilket bidrar till att användaren kan få en uppfattning om hur systemet både designmässigt samt funktionellt fungera och lättare kan förmedla sina krav.

Tillgänglig kunskap är kunskap om de krav som användaren har och som han eller hon lätt kan förmedla till utvecklaren. Prototyping är ett mindre bra alternativ för att få fram den tillgängliga användarkunskapen. Orsaken är, som jag nämnde tidigare, att prototyping är en mer kostnadsintensiv metod för kravutvinning. När kraven är lätta att utvinna blir kostnadsminskningen av underhållskostnaderna minimala och prototypingmetoden blir inte kostnadseffektivt.

Kunskap som användaren har men som han eller hon inte kan förmedla på ett effektivt sätt kallas igenkänd kunskap. Exempel på denna företeelse är när användare skall beskriva funktionerna i det system som de dagligen arbetar med. De kan då ofta inte direkt beskriva alla funktioner, men om de har programmet framför sig kan de med stöd av detta lättare förmedla kunskapen om de olika funktionerna. Användning av prototyping under dessa premisser är mycket lämpligt. Visualisering av det tänkta systemet med hjälp av prototyper får på ett effektivt sätt fram dessa kunskaper och krav.

Dold kunskap är av sådan art att användaren har den men ofta inte är medveten om den. Detta resulterar till att de har mycket svårt att förmedla denna kunskap. Ett exempel ur litteraturen (Karlsson, 1996) på dold kunskap är att användaren kan ha kunskap om hur musen skall användas vid kommunikation med en programvara, men har svår att förmedla denna kunskap till andra. Att använda prototyping för att utvinna dolda kunskaper eller krav är ett alternativ, dock inte den effektivaste kravutvinningsmetoden.

Genom de olika kunskapstyperna kan vi se att om kunskaperna är tydliga och tillgängliga är prototyping inte en lämplig metod för kravutvinningen.

5.3 Analys

5.3.1 Presentation

Jag har grupperat analysen av litteraturen på respektive problemställning. Varje problemställning har i sin tur flera underrubriker som är sammanställningar av gemensamma ståndpunkter som litteraturen har presenterat.

5.3.2 Undersöka vilka egenskaper som är begränsande för prototyping.

Kostnad

Att prototyping genererar, i de flesta fall, en högre kostnader under kravutvinningen och därigenom också under analysfasen är en uppfattning som är generell bland författarna. Att prototyping dessutom orsakar problem under initieringsfasen av ett projekt då det är svårt att uppskatta tidsåtgång och kostnader av systemutvecklingsprojektet är en uppfattning flera av källorna har. Dessa problem har sin grund i att prototyping är av iterativ natur. Det är antalet iterationer, dvs antalet utvärderingar och modifikationer, som är avgörande för kostnads- och tidsaspekten. Det är mycket svårt att på ett planeringsstadium av ett projekt att avgöra hur många iterationer som behövs innan man har uppnått ett av användarna godtagbart resultat.

Användarmedverkan

För att få prototyping att fungera krävs en intensiv användarmedverkan. Detta är enligt mig givet då utvärderingsförfarandet vid prototyping står och faller med detta. Denna intensiva användarmedverkan kräver dock att användarna är tillgängliga och förstår sin centrala roll under projektet för att inga kostsamma väntetider skall uppkomma. Jag tycker man skall betona att det också kräver att deras ordinarie arbetsuppgifter inte kräver deras fulla uppmärksamhet och att deras arbetsledare är införstådda med detta dilemma.

För att prototyping skall fungera får det inte heller finnas några konflikter mellan olika intressenter och aktörer om vilken funktionalitet applikationen skall ha eller hur den externa designen skall vara. Man måste i detta fall skilja på de konflikter och meningsskiljaktigheter som kan uppstå under processens gång mellan användare, vilket är naturligt, och de intressekonflikter som kan råda mellan beställare och användare. Användarna måste enligt litteraturen också vara motiverade för att man skall uppnå ett fullgott resultat under kravutvinningen.

I litteraturen finner man även två motstridiga påståenden. Dels att en begränsande faktor för användningen av prototyping är att metoden kräver att användarna har datorvana (Ould, 1990) och dels att prototyping kommer särskilt ifråga om man användarna har liten erfarenhet av adb (Andersen, 1991).
Jag tycker att påståendet om att datorvana skulle vara ett krav för att använda sig utav prototyping har en liten förankring till verkligheten. Prototyping är enligt min mening ett utmärkt hjälpmedel för användare utan datorvana att kunna ställa rätt krav på ett system vilket är näst intill omöjlig uppgift utan prototypingmetodens visualisering av det tänkta systemet. Detta är enligt mig det fundamentala målet med användningen av prototyping för kravutvinning. Den iterativa processen som är karaktäristisk för prototyping är dessutom en lärandeprocess som gagnar användare med mediokra datakunskaper.

Verktyg

I litteraturen finner man att ytterligare begränsande faktorer med prototypingmetoden under kravutvinningsprocessen är att den kräver att man har tillgång till utvecklingsverktyg, i detta fall fjärde generationens, samt att det krävs att utvecklaren inte bara har kunskap om systemutveckling och dess arbetsformer utan också om verktygen som prototyping kräver. Detta ställer, enligt mig, mycket höga krav på både projektledare och utvecklare för att man skall kunna dra fördel av prototypingmetoden för kravutvinning.

Icke funktionella krav

Prototyping ger enligt litteraturen också ofta ett svagt stöd under kravutvinningen för vissa aspekter på systemet som man ofta måste tillgodose för att slutresultatet skall bli tillfredsställande. Aspekterna som nämns i litteraturen utvecklas dock inte vidare. Jag har därför gjort en beskrivning av dessa utifrån egna reflektioner.

- *Säkerhetskrav och tillika säkerhetsaspekter.* Med säkerhet innefattas i detta sammanhang både inre och yttre säkerhet. Med inre säkerhet menas hur säkert och stabilt systemet är och därigenom hur säker informationen är i systemet. Den yttre säkerheten skall skydda informationssystemet och dess information från intrång och åtkomst utifrån.

- *Krav på enkelhet i underhållet*, dvs att informationssystemet är flexibelt och lätt att ändra vid förändrade rutiner och krav.

- *Nätverksaspekter.* Nätverksaspekterna kan röra sig om hur man skall uppdatera informationen i systemet och enkelheten i detta. Det kan också handla om hur länkning av tabeller, om det är en databas man utvecklar, skall fungera praktiskt.

- *Plattformsfrågor.* Med plattform i detta sammanhang menas vilken typ av system eller maskintyp som informationssystemet skall vara kompatibelt på. Dessa krav är svåra att få fram då prototypen ofta bara skapas i en systemmiljö.

- *Volymkörningar och krav på responstider.* För att kunna provköra stora mängder av information och göra responstidstester på prototypen krävs för det första att mycket stora mängder information matas in i systemet. Detta är en tidsmässigt orimlig uppgift då det kan röra sig om tiotusentals poster. För det andra är prototypen just en prototyp och har inte samma prestanda och mätvärden som det riktiga systemet.

- *Krav på återhämtning efter systemnedgång,* dvs de krav som ställs på att systemet skall kunna återhämta sig utan att information går förlorad och extra arbete tillföljd samt samt tidskrav på återhämtningen efter en nedgång.

Den gemensamma nämnaren för dessa svaga punkter är att samtliga sorteras in under ickefunktionella krav. Det är således mycket viktigt att ha detta i åtanke när prototyping används i ett systemutvecklingsprojekt då de ickefunktionella kraven kan spela en avgörande betydelse för ett informationssystem.

Flexibilitet

Vid utveckling av system i mycket dynamiska miljöer med snabba förändringar är prototyping en mindre lämpad metod enligt litteraturen. Detta beror på att kravutvinning med prototyping av throw-away typ utvinner samtliga användarkrav på ett tidigt stadium och därigenom fryser man kravspecifikatonen på ett tidigt stadie i utvecklingscykeln.

Detta anser jag vara en nackdel med prototyping. Kravspecifikationen skall frysas så sent som möjligt för att skapa flexibilitet i systemet och underlätta vid förändrade krav. Jag anser att det är förvånansvärt att inte fler av källorna tar upp denna problematik.

5.3.3 Undersöka hur typen av informationssystem påverkar lämpligheten av prototyping

Tekniska system

En allmän ståndpunkt bland källorna är att prototyping inte är lämpligt för utveckling av tekniska system. De tekniska systemen, även kallade processorienterade system, karaktäriseras av att de behandlar relativt små datamängder men utför komplexa processer som kan har tidskrav. Motsatsen till de tekniska system är de administrativa systemen som behandlar stora datamängder men utför bara enkla uträkningar och transaktioner. Vid utveckling av sådana system ger prototyping ett bra stöd för kravutvinning. Även system som är logiskt intensiva och som bygger på databehandling med fastställda räkneregler, sk algoritmer, är enligt flera källor mindre lämpade för metoden. Detta beror enligt litteraturen på att det inte finns några verktyg som stödjer detta samt att det är allt för tidskrävande att göra prototyper av system som har en komplex och avancerad funktionalitet och sedan kassera denna för att sedan börja om på nytt.

Jag anser att litteraturens samstämmighet i att tekniska system inte är lämpade för prototyping, och i synnerhet realtidssystem var väntad. Som jag har påpekat tidigare är inte 4-GL verktygen ämnade för realtidssystem utan för design av gränssnitt och viss funktionalitet, vilket gör prototyping lämpligare för de administrativa systemen.

Interaktion

Det finns också stöd på flera håll i litteraturen att prototyping är ett mindre lämpligt val vid utveckling av system som har liten eller ingen interaktion mellan människa och maskin. Följaktligen är prototyping inte ett lämpligt val vid utveckling av inbyggda system eller system som uteslutande interagerar med andra system. Detta grundar sig på prototypingmetodens natur. Den är en metod som har en stark användarmedverkan under processens gång och är givetvis därför mer lämpad för människa-maskin dialogen.

Jag anser att på grund av den centrala roll användarmedverkan har vid prototyping är metoden inte heller ett fullgott alternativ vid utveckling av generellasystem eller standardsystem då fler användare och kunder måste få sina röster hörda.

Storlek

Det finns motsägelser i litteraturen angående frågan om storleken på systemet som skall utvecklas har någon betydelse för valet av prototyping eller ej. En källa (Pressman, 1997) anser att mycket stora och komplexa system som kräver tusentals rader kod för att få en användbar prototyp är inte lämpliga i sin helhet för prototyping metoden. Den andra källan (Vonk, 1990) anser att det är en missuppfattning att prototyping inte skulle vara lämpligt för kravutvinning vid utveckling av stora och komplexa system. Vidare hävdar författaren att prototyping är lika lämpligt för stora som för små system. Dock inlägger den mer kritiska författaren att man kan använda prototyping för kravutvinning av valda delar av stora system där kravbilden är mer osäker.

Jag anser att den mer kritiska författaren har en mer korrekt uppfattning. Kostnaderna, på grund av tidsåtgången, kan bli orimligt höga om man skall utveckla kompletta prototyper av mycket stora system. Jag anser även att små och mycket triviala system inte är lämpliga att utveckla med prototyping. Detta grundar jag på att kostnaderna för prototyping är högre än andra metoder och för att kompensera detta måste underhållskostnaden för systemet, dvs kostnader för fluktuerande krav, bli lägre med prototyping än med någon alternativ metod. Detta är inte fallet då kravbilden är tydlig och klar, vilket den i större utsträckning är i små och triviala system. Det innebär att resultatet av användningen utav prototyping motsvarar inte de ökade kostnaderna.

5.3.4 Ta reda på vilka faser under utvecklingen som påverkas tillföljd av dessa brister.

Bristerna eller de begränsande egenskaperna som inte beaktas vid användning av prototyping påverkar enligt en samstämmig litteratur samtliga faser i systemutvecklingen. Detta beror på att de begränsande egenskaperna prototyping har, och som inte beaktas, leder till att samtliga krav användaren har inte realiseras och därför måste åtgärdas på ett senare stadie i utvecklingscykeln; underhållsfasen. För att kunna åtgärda felet måste således dokumentationen uppdateras, designen ändras och programkod korrigeras vilket gör att samtliga faser i utvecklingen berörs.

Jag vill även betona att kostnaderna som bristerna genererar har i stor utsträckning sin uppkomst i underhållsfasen vilket bidrar till att totalkostnaden för systemutvecklingsprojektet stiger kraftigt. Se fig 4, sidan 20.

5.3.5 Utföra en bedömning när det kan vara lämpligt att använda prototyping

Bedömningen av när prototyping kan vara en lämplig metod har jag grundat på den analys av litteraturen jag har gjort angående de olika brister och begränsande faktorer prototyping har och medför. Bedömningen har till syfte att ge en övergripande bild över de förutsättningar som krävs för att man med större sannolikhet skall lyckas i ett projekt med prototyping som kravutvinningsmetod.

Jag bedömer utifrån nämnda förutsättningar att prototyping är en lämplig metod för kravutvinning om följande förutsättningar uppfylls för projektet.

1. Användarnas krav på det tänkta systemet är osäkra, dvs att de inte har en tydlig kravbild vid projektets början. Osäkerheten i kravbilden behöver dock inte innefatta hela systemet utan det kan röra sig om osäkerhet i kravbilden på vissa delar av det. Jag stödjer detta

antagande på den samstämmighet som litteraturen hade kring kravbildens betydelse för prototyping. (Vonk, 1990), (Pressman, 1997), (Martin, 1991).

2. Användarna besitter kunskap av typen framtida kunskap, dvs användaren vet inte vilka krav som är relevanta att ställa och hon eller han har svårt att förmedla kraven till utvecklaren, eller igenkänd kunskap som kännetecknas av att användaren har kunskapen om kraven men kan inte förmedla dess på ett effektivt sätt. Flera av källorna stödjer detta antagande om betydelsen av användarnas kunskapsstruktur. (Vonk, 1990), (Karlsson, 1996).

3. Systemet som skall utvecklas är ett administrativt system som har en intensiv dialog mellan användaren och systemet. Systemet skall också vara skräddarsytt för *en* organisation eller kund. Detta krav stödjer samtlig litteratur som jag har använt under studien.

4. Den intensiva användarmedverkan är inte något hinder, dvs att användarna är tillgängliga under hela kravhanteringsprocessen och deras överordnade på deras ordinarie arbetsplatser är fullt införstådda med detta. Detta krav grundar sig på prototypingmetodens intensiva användarmedverkan som flera källor betonar. (Vonk, 1990), (Pressman, 1997)

5. Man har tillgång till utvecklare som kan hantera de verktyg som används under processen samt att verktygen är tillgängliga. Genom att 4-GL verktyg används för att skapa prototypen ställer detta höga krav på utvecklaren och tillgängligheten på verktyget (Pressman, 1997).

6. Kritiska icke funktionella krav har redan specificerats innan man påbörjat prototypingen eller att det inte existerar några kritiska icke funktionella krav. Med kritiska icke funktionella krav menar jag de krav som beskrivs i Martin (1991), se sidan 17.

7. Det finns inte några konflikter mellan olika intressenter på funktionaliteten och den externa designen på systemet som skall utvecklas. Som jag nämnde tidigare måste man i detta fall skilja på de konflikter och meningsskiljaktigheter som kan uppstå under processens gång mellan användare, vilket är naturligt.

8. Organisationen som systemet skall stödja är stabil och har en relativt låg förändringstakt. Detta krav grundar sig på att kravspecifikationen fryses på ett tidigt stadium när prototyping används (Vonk, 1990). Om organisationen är mycket dynamisk hinner kraven att förändras ett flertal gånger innan systemet är klart och följden är dyra förändringar i det färdiga systemet.

9. Projektet har ingen skarpt dragen deadline då systemet skall vara leverans klart, utan leverans tiden är tänjbar under systemutvecklingens gång.

10. Projektet har inte en fast budget utan kostnaden för projektet kan i viss mån förändras under projektets gång. Både detta och föregående krav beror på att prototyping är en iterativ metod. Antalet iterationer är mycket svårt att avgöra på ett planeringsstadie vilket leder till osäkerhet i tids- och kostnadsberäkningen.

De tio punkter som jag har redovisat kan ses som en guideline för att säkerställa att man inte använder prototyping under ett projekt som inte är lämpat för detta. Bristerna med prototyping

kan på detta sätt undvikas eller minimeras vilket säkerställer att totalkostnaden för systemutvecklingsprojektet hålls nere.

6 Slutsatser

Jag har med utgångspunkt från min problemställning kommit fram till, efter att ha genomfört min litteraturstudie, följande slutsatser angående prototypingmetodens brister och möjligheter för kravutvinning. Jag redovisar mina slutsatser grupperade efter respektive problemställning.

6.1 Undersöka vilka egenskaper som är begränsande för prototyping.

Genom min litteraturstudie har jag kommit fram till att prototyping som stöd för kravutvinning har vissa begränsande egenskaper som man måste ta hänsyn till vid ett systemutvecklingsprojekt. Dessa begränsningar kan undvikas eller elimineras då man vet vilka de är samt hur de påverkar utvecklingsarbetet. De begränsande egenskaperna som jag har funnit vara primära för prototyping presenteras nedan.

- Kostnaden är hög för kravutvinningen och därigenom också för analysfasen då prototyping används som stöd. Kostnaden är relaterad till tidsåtgången tillföljd av antalet iterationer som måste göras för att få en av användarna accepterad prototyp. När prototyping används är risken hög att användarna finner kontinuerligt nya krav som inte är av relevant betydelse för systemets funktionalitet men som ändå skall realiseras för att tillfredsställa kundens önskemål. Följden blir då, som jag har redovisat tidigare, att iterationerna blir många och projekttiden blir utdragen med följden att kostnaderna blir höga för analysfasen.

- Den intensiva användarmedverkan som prototyping kräver kan lätt bli en begränsande egenskap för användning av metoden. Om användarna är låsta vid sitt ordinarie arbete och inte är tillgängliga vid utvärderingstillfällena blir följden att utvecklingsprojektet blir stillastående och kostnaderna för projektet blir höga. Dessutom krävs det att användarna är motiverade och engagerade för att prototyping skall vara framgångsrikt.

- Behovet av utvecklingsverktyg och utvecklare som behärskar dessa verktyg är ytterligare en begränsande faktor för prototyping. Som jag har redogjort för tidigare, se sidan 23, ställer verktygen höga krav på både projektledare och utvecklare vid prototyping med stöd av 4-GL verktyg.

- Prototyping ger ett svagt stöd vid utvinning av vissa ickefunktionella krav. De ickefunktionella kraven som jag hänvisar till är de som jag tidigare har redovisat i genomförandedelen, se sidan 23.

- Prototyping hämmar flexibiliteten för att kunna hantera fluktuerande krav under den fortsatta utvecklingen av informationssystemet. Detta beror på att man med throw-away prototyping utvinner kraven på ett tidigt stadie i utvecklingen och därigenom fryser kravspecifikationen på ett tidigt stadie i utvecklingsarbetet. Detta medför att prototyping inte är lämpligt att använda vid utveckling av informationssystem i dynamiska och snabbt föränderliga miljöer.

- Kostnads- och tidskalkylering för utvecklingsarbetet är mycket svårt vid prototyping. Detta är en mycket begränsande egenskap för prototyping i de fall då man har skarpa ramar för budget och tidsåtgång för projektet.

6.2 Undersöka hur typen av informationssystem påverkar lämpligheten

Informationssystemets typ och beskaffning har också betydelse för hur prototyping lämpar sig som kravutvinningsmetod. Jag har under min undersökning kommit fram till, vilket jag har redovisat i genomförande delen, att det finns några egenskaper att ta hänsyn till då prototyping skall utnyttjas i ett systemutvecklingsprojekt.

Tidigare har jag beskrivit de två huvudtyper av informationssystem som existerar idag. Tekniska system och administrativa system, se sidan 9. De tekniska systemen, även kallade processorienterade system, karaktäriseras av att de behandlar relativt små datamängder men utför komplexa processer som kan har tidskrav vid realtidsprocesser. Dessa system är enligt min undersökning, se sidan 24, inte lämpade för prototyping metoden. Detta har sin förklaring i att det idag inte finns några några effektiva verktyg för utveckling av prototyper av dessa system samt att denna system typ inte har någon större dialog mellan människa och maskin.

Jag har även kommit fram till att prototyping är ett mindre lämpligt val vid utveckling av system som har liten eller ingen interaktion mellan människa och maskin, se sidan 24. Följaktligen är prototyping inte ett lämpligt val vid utveckling av inbyggda system eller system som interagerar uteslutande med andra system. Prototyping är en metod som har en stark användarmedverkan under processens gång och är därför mer lämpad för människa-maskin dialogen och system som utnyttjar denna.

Storleken påverkar också valet av prototyping som metod för kravutvinning. Kostnaderna, på grund av tidsåtgången, kan bli orimligt höga om man skall utveckla kompletta prototyper av mycket stora system. Även små och mycket triviala system är inte lämpliga att utveckla med prototyping med avseende på deras inverkan på totalkostnaden för projektet.

6.3 Ta reda på vilka utvecklingsfaser som påverkas tillföljd av bristerna

Bristerna och tillika begränsande egenskaper som jag har redovisat påverkar enligt min studie samtliga faser i utvecklingsarbetet. Som jag har betonat tidigare påverkas i första hand analysfasen då prototyping av throw-away typ är en metod för kravutvinning, se sidan 19.
Dock påverkar en bristfällig kravutvinning och där med en bristfällig kravspecifikation även de efterföljande faserna då kraven inte blir riktigt realiserade. För att tillmötesgå användarnas krav måste ändringar och korrektioner göras i programkoden och designen samt uppdateringar av dokumentationen. Detta genererar en kostnadsökning speciellt i underhållsfasen vilket resulterar i, se sidan 20, en kraftig ökning av totalkostnaderna för projektet.

6.4 Utföra bedömning av lämpligheten för användning utav prototyping

Bedömningen som följer har jag gjort med utgångspunkt från de uppgifter som jag har erhållit från mina källor och som jag har redovisat i kapitel 5. Bedömningen har till syfte att ge en övergripande bild över de förutsättningar som bör vara uppfyllda innan man beslutar att använda prototyping som stöd för kravutvinningen i ett utvecklingsprojekt. Bedömningen kan också fungera som en guideline vid projektstarten för att säkerställa att prototyping endast används då det är lämpligt för det aktuella projektet.

6 Slutsatser

Förutsättningarna som bör vara uppfyllda för att prototyping skall vara en effektiv kravutvinningsmetod är följande:

- Användarnas krav på det tänkta systemet är osäkra, dvs att de inte har en tydlig kravbild vid projektets början. Osäkerheten i kravbilden behöver dock inte innefatta hela systemet utan det kan röra sig om osäkerhet i kravbilden på vissa delar av det.

- Användarna besitter kunskap av typen framtida kunskap, dvs användaren vet inte vilka krav som är relevanta att ställa och hon eller han har svårt att förmedla kraven till utvecklaren, eller igenkänd kunskap som kännetecknas av att användaren har kunskapen om kraven men kan inte förmedla dess på ett effektivt sätt.

- Systemet som skall utvecklas är ett administrativt system som har en intensiv dialog mellan användaren och systemet. Systemet skall också vara skräddarsytt för *en* organisation eller kund.

- Den intensiva användarmedverkan är inte något hinder, dvs att användarna är tillgängliga under hela kravhanteringsprocessen och deras överordnade på deras ordinarie arbetsplatser är fullt införstådda med detta.

- Man har tillgång till utvecklare som kan hantera de utvecklingsverktyg som används under processen samt att verktygen är tillgängliga.

- Kritiska ickefunktionella krav har specificerats redan innan man påbörjat prototypingen eller att det inte existerar några kritiska ickefunktionella krav på det tänkta informationssystemet. Med kritiska ickefunktionella krav menar jag de krav som återfinns på sidan 17.

- Det finns inte några konflikter mellan olika intressenter på funktionaliteten och den externa designen på systemet som skall utvecklas. Man måste dock i detta fall skilja på de konflikter och meningsskiljaktigheter i kravbilden som kan uppstå under processens gång mellan olika användare, vilket är naturligt.

- Organisationen som systemet skall stödja är stabil och har en relativt låg förändringstakt.

- Projektet har ingen skarpt dragen deadline då systemet skall vara leverans klart, utan leverans tiden är tänjbar under systemutvecklingens gång.

- Projektet har inte en fast budget utan kostnaden för projektet kan i viss mån förändras under projektets gång.

7 Diskussion

7.1 Presentation

Jag har valt att diskutera varje problemställning var för sig och grupperar därför detta kapitel efter respektive problemställning och avslutar med en allmän diskussion samt ger förslag och idéer till fortsatta arbeten inom ämnet.

7.2 Undersöka vilka egenskaper som är begränsande för prototyping

Enligt min mening har jag genom min litteraturstudie fått fram ett adekvat resultat på denna problemställning. Jag anser att antalet brister och begränsande egenskaper som återfinns i resultatet och som är förknippade med prototyping motsvarar de förväntningar jag hade. I mitt förväntade resultat, se sidan 12, har jag tagit upp de återkommande utvärderingarna som kunden eller användaren måste göra som en trolig egenskap som är begränsande, vilket jag sedermera fick bekräftat av resultatet. Även att den ökade kostnaden för kravutvinning med stöd av prototyping samt storleken på det tänkta informationssystemet kan vara begränsande faktorer återfinns i det förväntade resultatet.

Att prototyping ger ett svagt stöd för utvinning av icke funktionella krav har jag tagit upp som en begränsande egenskap för metoden. Jag vill betona att denna begränsande egenskap inte på något sätt är unik för prototyping utan är generell för samtliga kravutvinningsmetoder. Dock har jag valt att inkludera denna begränsande egenskap i resultatet, då jag *inte* gör en jämförande studie mellan olika metoder utan enbart undersöker prototyping, vilket jag har betonat tidigare.

7.3 Undersöka hur typen av informationssystem påverkar lämpligheten

Genom att litteraturen hade en mycket samstämmig uppfattning om hur informationssystemets typ och beskaffenhet påverkar lämpligheten av prototyping som stöd för kravutvinning anser jag att svaret på denna frågeställning är fullgott. I mitt förväntade resultat har jag tagit upp informationssystemets betydelse för prototyping och att storleken på systemet har en avgörande betydelse för lämpligheten vilket även resultatet visar.

7.4 Ta reda på vilka utvecklingsfaser som påverkas tillföljd av bristerna

I mitt förväntade resultat bedömde jag att samtliga faser under utvecklingsarbetet påverkas av de brister som prototypingmetoden har. I det slutgiltiga resultatet är uppfattningen den samma. Med tanke på att prototyping är en metod för kravutvinning och att användarnas kraven är grunden för hela utvecklingsarbetet tycker jag att resultatet är relevant och trovärdigt.

7.5 Utföra bedömning av lämpligheten för användning utav prototyping

Eftersom jag har baserat bedömningen på de resultat jag har erhållit i de tidigare problemställningarna anser jag att resultatet och trovärdigheten i bedömningen är hög. I mitt förväntade resultat var jag, enligt min åsikt, lite väl försiktig då det reella resultatet är betydligt mer utförligt.

7.6 Övrigt

I min litteraturstudie har jag gjort en relativt snäv avgränsning då jag har haft till avsikt att enbart behandla prototyping av typen throw-away med stöd av 4-GL verktyg. Ofta förknippas dock 4-GL med evolutionär prototyping. Dessa två olika typer av prototyping har många likheter men även olikheter. Resultatet jag har redovisat för bristerna med throw-away prototyping skulle, med stor sannolikhet, varit identiska med resultatet från en undersökning av evolutionär prototyping i avseende på brister som kan härledas till användandet av 4-GL verktygen. Dock skulle de övriga bristerna, som är knuten till själva arbetsmetodiken, skilja sig i flera avseenden. Under min studie har jag använt metoden litteraturstudier för att svara på frågeställningarna. Det är möjlig att andra faktorer och värderingar skulle ha kommit fram om jag hade använt mig av intervjuer som kompletterande metod. Jag tror att undersökningen skulle kunnat bli mer nyanserad och något mer verklighetsförankrad om jag hade inkluderat någon eller några intervjuer i undersökningen. Jag finner dock att litteraturen har varit ett fullgott stöd för att få svar på mina frågeställningar och väcka nya frågor i ämnet.

Jag tycker att det är viktigt att betona, med tanke på resultatet, hur viktigt det är att beakta de begränsande egenskaperna prototyping har och vilka förutsättningar som krävs för att metoden skall vara kostnads- och tidsbesparande för ett utvecklingsprojekt.

7.7 Uppslag till fortsatt arbete

En möjlig fortsättning till denna studie skulle vara att göra en jämförande studie mellan olika typer av kravutvinningsmetoder. Varje kravutvinningsmetod har sina brister och tillgångar och är mer eller mindre lämpade för olika utvecklingsprojekt. Det skulle därför vara intressant att göra en sammanställning över samtliga metoder och deras lämplighet utifrån olika förutsättningar som kan förekomma vid utvecklingsarbeten.

Ytterligare ett uppslag till fortsatt arbete är att genomföra en motsvarande studie på evolutionär prototyping. Som jag nämnde tidigare finns det likheter mellan throw-away prototyping och evolutionär prototyping men även flera skillnader som är intressanta att identifiera och jämföra.

Referenser

Andersson, E. S. (1991), Systemutveckling–principer, metoder, tekniker, Studentlitteratur, Lund.

Beekman, G. (1994), Computer currents, The Benjamin / Cummings Publishing Company, Redwood city, California.

Karlsson, J. (1996), Framgångsrik kravhantering - nr 2, Sveriges Verkstadsindustrier, Stockholm.

Loucopoulos, P. och Karakostas, V. (1995), System requirements engineering, McGraw-Hill, New York.

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

Meldrum, M. och Lejk, M. och Guy, P. (1993), SSADM Techniques, Studentlitteratur, Lund

Ould, M. A. (1990), Strategies for software engineering, John Wiley & Sons, England

Perry, W. E. (1991), Quality assurance, John Wiley & Sons, New York.

Pressman, R. S. (1997), Software engineering - Fourth edition, McGraw-Hill, New York.

VI (1995), Krav på krav - nr 2, Sveriges Verkstadsindustrier, Stockholm.

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

Index

4

4-GL verktyg · 10

A

administrativa informationssystem · 10
analysfasen · 3
Användarmedverkan · 24

B

begränsande egenskaper · 29
Brainstorming · 7

E

Enkäter · 15
evolutionär prototyping · 7

F

Fallstudie · 14
Flexibilitet · 26
funktionella krav · 9

I

icke funktionella krav · 9
informationssystem · 10
Interaktion · 26
Intervjuer · 6
iterativ · 8

K

Kostnad · 24
kravbegreppet · 8
Kravhantering · 4
Kravhanteringsprocessen · 4
kravspecifikation · 3
Kravutvinning · 4
Kravutvinningsmetoder · 6
kunskapsstruktur · 23
Kvalitet · 11

L

LIBRIS · 19
Litteraturstudie · 14
Litteraturstudier · 14
livscykelmodellen · 3

O

Observation · 6

P

Priotitering · 4
processorienterade system · 10
programmeringsspråk · 10
Protokollanalys · 7
Prototyping · 7

S

Storlek · 27
systemutvecklingsfaser · 3
Sökmetoder · 19

T

tekniska system · 10
throw-away · 7

U

Utvecklingsfaserna · 27
utvecklingsmodell · 3

V

Validering · 6
Verktyg · 25