INSTITUTIONEN FÖR INFORMATIK
Handelshögskolan vid Göteborgsuniversitet

SYSTEMUTVECKLING

- en jämförelse mellan teoretiska modeller och ett praktikfall

Detta examensarbete behandlade ämnet systemutveckling. Ett systemutvecklingsprojekt där ett bokningssystem konstruerades för ett språkreseföretag har studerats och jämförts med ett urval av olika teoretiska modeller. Ingen konkret systemutvecklingsmodell användes men utvecklingsarbetet som bedrivits kan liknas vid prototyping. I analysen har jag kommit fram till att utvecklingsarbetet trots allt gav ett bra slutresultat men att processen kunde effektiviserats genom användandet av en kombination av datamodellering och prototyping. Detta skulle ge goda förutsättningar för ett bra informationssystem genom att utvecklingen av informationssystemet sker strukturerat i dialog mellan utvecklare och användare.

EXAMENSARBETE 10 p
ingående i ADB-programmet 80p
Victoria Falkenström
Vårterminen 1998
Handledare: Roy Corneliusson

Innehållsförteckning

1 Introduktion

1.1 Bakgrund

Mitt examensarbete grundar sig på ett arbete jag utfört på Funktionell Organisation. Funktionell Organisation är ett företag som utvecklar och säljer det administrativa standardsystemet WinBas. Genom att konstruera ett specifikt program som kopplas ihop med WinBas kan informationssystemet anpassas efter användarnas krav.

Min uppgift var att utveckla ett bokningssystem åt International Meetings, ett företag som säljer språkresor. De hade redan beställt WinBas och för att det skulle passa in i verksamheten behövde de också ett bokningssystem.

När ett informationssystem skall utvecklas finns det en uppsjö olika modeller och metoder att tillgå. Arbetet med att utveckla bokningssystemet stämde inte överens med någon av de teoretiska modeller som jag tidigare kommit i kontakt med. Bokningssystemet utvecklades successivt utan någon direkt plan för hur det skulle gå till, vilket medförde att vissa problem uppstod och att arbetet komplicerades. Det var intressant att se hur utvecklingsarbetet skilde sig från vad jag tidigare lärt mig och jag började fundera på om arbetet hade kunnat underlättas genom att använda en traditionell systemutvecklingsmodell.

1.2 Syfte

Syftet med min rapport är att beskriva mina erfarenheter beträffande systemutveckling i den miljö jag arbetat i. Jag redogör först för hur systemutveckling bör gå till enligt några systemutvecklings-modeller. Därefter beskrivs mitt arbete med att skapa bokningssystemet och utifrån detta görs en jämförelse mellan det aktuella systemutvecklingsarbetet och de traditionella systemutvecklings-modellerna. Jag ställer mig frågan om arbetet med att utveckla bokningssystemet hade blivit effektivare om jag använt mig av någon traditionell utvecklingsmodell. Dessutom analyseras vilken systemutvecklingsmodell som skulle varit lämplig att använda i den aktuella situationen.

1.3 Frågeställning

- Hur bör systemutveckling gå till enligt ett urval av olika systemutvecklingsmodeller?

- Vilka problem kan man stöta på under utvecklingen av ett informationssystem?

- Beskrivning av systemutvecklingsprocessen vid konstruerandet av bokningssystem för International Meetings.

- Jämförelse mellan mitt tillvägagångssätt och de olika systemutvecklingsmodellerna
  -Likheter
  -Skillnader
  -Fördelar och nackdelar
- Hade arbetet med att utveckla bokningssystemet underlättats om jag använt mig av någon av de traditionella systemutvecklingsmodellerna?

- Vilken systemutvecklingsmodell skulle vara lämplig att använda i den aktuella utvecklingsmiljön?

1.4 Metod

De olika systemutvecklingsmodellerna har jag studerat i tillgänglig facklitteratur. Dessutom har jag använt mig av de erfarenheter jag fått när jag var med och utvecklade ett bokningssystem. Därefter har jag jämfört mina erfarenheter med de olika systemutvecklingsmodeller jag studerat.

1.5 Disposition

I kapitel två beskrivs systemutvecklingens historia. I kapitlen tre till sju följer en beskrivning över ett antal olika traditionella systemutvecklingsmodeller. I det efterföljande kapitlet redogör jag för hur systemutvecklingsprocessen för att utveckla ett bokningssystem för företaget International Meetings utfördes, vilka problem vi stötte på samt hur resultatet blev. I kapitel nio görs en bedömning av hur utvecklingsarbetet kunde ha förbättrats och effektiviserats om en traditionell systemutvecklingsmodell använts.

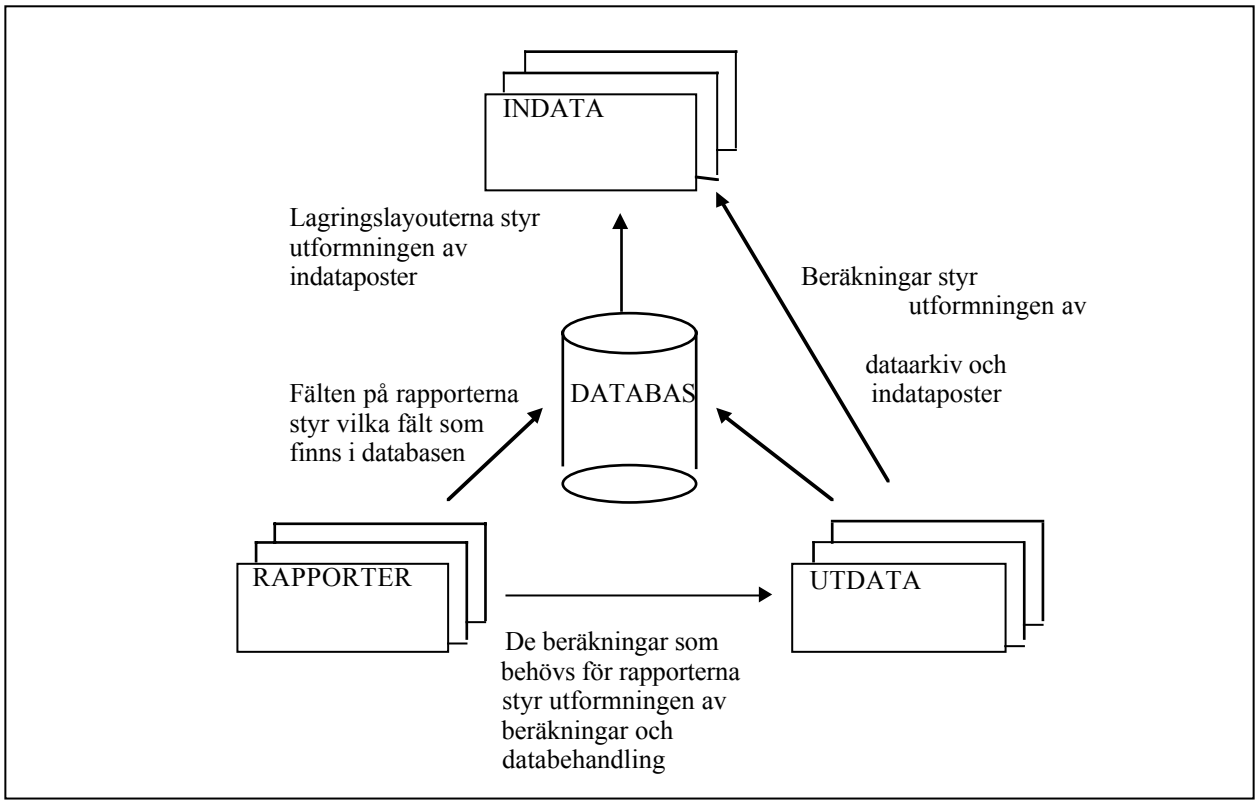
2 Systemutvecklingens historia

Med begreppet systemutveckling avser jag utvecklingen av ett informationssystem, det skulle alltså kunna kallas informationssystemutveckling. Detta blir dock krångligt i längden, därför har jag valt att kalla det systemutveckling.

Sedan 1960-talet har systemutveckling som begrepp existerat. Det finns en mängd olika åsikter och synsätt på systemutveckling, och det är svårt att ge någon ”rätt” beskrivning över det bästa tillvägagångssättet när ett informationssystem skall utvecklas. Många förespråkar någon strikt systemutvecklingsmodell, medan andra hävdar att man snarare begränsar arbetet än effektiviserar det genom att strikt följa systemutvecklingsmodellerna.

2.1 1960-talet

Under 1960-talet var systemutvecklingsarbetet till stor del inriktat på vad som skulle komma ut ur systemet, t ex fakturor och lönelistor. Utifrån detta beslutade man därefter vilken indata som skulle in i informationssystemet. Utvecklingsprocessen gick så att säga baklänges. (Brown, s. 36 ff)



Figur 2.1 visar en överblick över en” utdata-orienterad” modell (Brown, s.39)

2.2 1970-talet

Under 1970-talet började processorienterade och funktionsorienterade metoder användas (Brown, s36 ff). I det processorienterade synesättet ses systemutvecklingen som någonting mer än att bara skapa ett informationssystem. Inriktningen sker inte enbart på informationssystemet, utan även på den övriga verksamheten, t ex genom att ge användarna mer kunskap om den egna verksamheten. Det funktionsorienterade synesättet utgår från de funktioner (aktiviteter) som utförs i verksamheten. Genom att studera de olika funktionerna identifieras informationsbehovet informationssystemet skall täcka.

Lösningen på problemen ansågs under 1970-talet vara att utforma detaljerade specifikationer för hur systemet skulle se ut och fungera. Utifrån det skulle systemerarna och programmerarna sedan realisera informationssystemet. Det ges då mycket lite utrymme för förändringar av informationssystemet under utvecklingens gång. En typisk funktionsorienterad modell från 1970-talet är ISAC-modellen (en beskrivning av ISAC görs i kapitel fyra).

2.3 1980-talet

1980-talet innebar stora förändringar i synen på hur systemutvecklingsarbete bör bedrivas. Nu började man utgå från ett dataorienterat synesätt på systemutvecklingen (Brown, s.36 ff). Utgångspunkten är då vilka förhållanden i och utanför verksamheten som det är viktigt för medarbetarna i verksamheten att ha information om.

Datamodellering beskriver det dataorienterade synesättet på systemutveckling men är ingen komplett systemutvecklingsmodell, utan täcker endast en del av systemeringen. (En beskrivning av Datamodellering följer i kapitel fem)

2.4 1990-talet

Informationssystemen har blivit alltmer komplexa. På 1960-talet bestod ett stort informationssystem av ca 10000 - 15000 rader kod. Idag innehåller de stora projekten miljoner rader kod och dessutom måste mjukvaran interagera, kommunicera och samarbeta med kanske hundra andra mjukvaruprodukter (Brown, s.18). Detta har resulterat i att nya tankegångar angående systemutveckling och programmering tagit fart. Under 1990-talet har den objektorienterade systemutvecklingen blivit alltmer populär. Objektorientering härstammar från början från programmeringsområdet men har på senare tid även fått spridning inom andra faser av systemutvecklingen. (En beskrivning av objektorientering följer i kapitel sju)

3 Systemutvecklingens livscykel

I samband med systemutveckling talas det ofta om systemutvecklingens livscykel. (Brown, s. 162 ff samt Andersen, s.39 ff) Utvecklingen består av sex olika faser som varje informationssystem går igenom. Dessa faser är:

3.1 Analys

I denna fas studeras användarnas verksamhet för att komma fram till *vad* systemet behöver göra. Analysfasen kan delas upp i två delområden, dels verksamhetsanalys, dels informationssystem-analys.

I verksamhetsanalysen analyseras verksamheten och därefter avgörs på vilket sätt informations-systemet kan förbättra verksamheten. I informationssystemanalysen bestäms vilket innehåll informationssystemet skall ha.

3.2 Utformning

En plan skapas över *hur* informationssystemet skall kunna utföra de funktioner som i analysfasen identifierades. Det beslutas vilken mjukvara och hårdvara som skall användas. Gränssnittet, rapporterna och databaserna formges. Utifrån detta kan sedan programmeraren konstruera informationssystemet.

Även utformningsfasen kan delas upp i två delområden. Dessa är Principiell utformning av teknisk lösning och Utformning av utrustningsanpassad teknisk lösning.

3.3 Realisering

Det är här själva programmeringsarbetet tar vid. Programmeraren skriver programkoden och databaser konstrueras. När detta är färdigt testas systemet och eventuella fel korrigeras.

3.4 Implementering

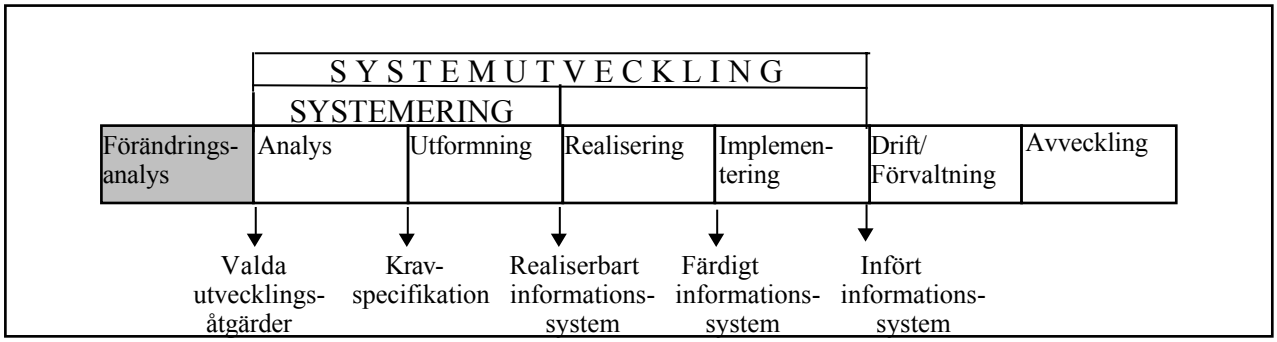
Nu installeras systemet och användarna utbildas i hur systemet fungerar och skall användas. Det är inte ovanligt att användarna under en övergångsperiod använder både det nya och det gamla systemet. Detta gör att verksamheten inte helt stannar upp om något fel skulle uppstå.

3.5 Drift och förvaltning

Under drifts- och förvaltningsfasen kan man fortfarande räkna med att en hel del korrektioner och förändringar kommer att behöva göras. Det kan vara allt från programfel till funktioner som användarna kommer på att informationssystemet behöver utökas med. Ofta sker förbättringar i informationssystemet under hela dess livslängd

3.6 Avveckling

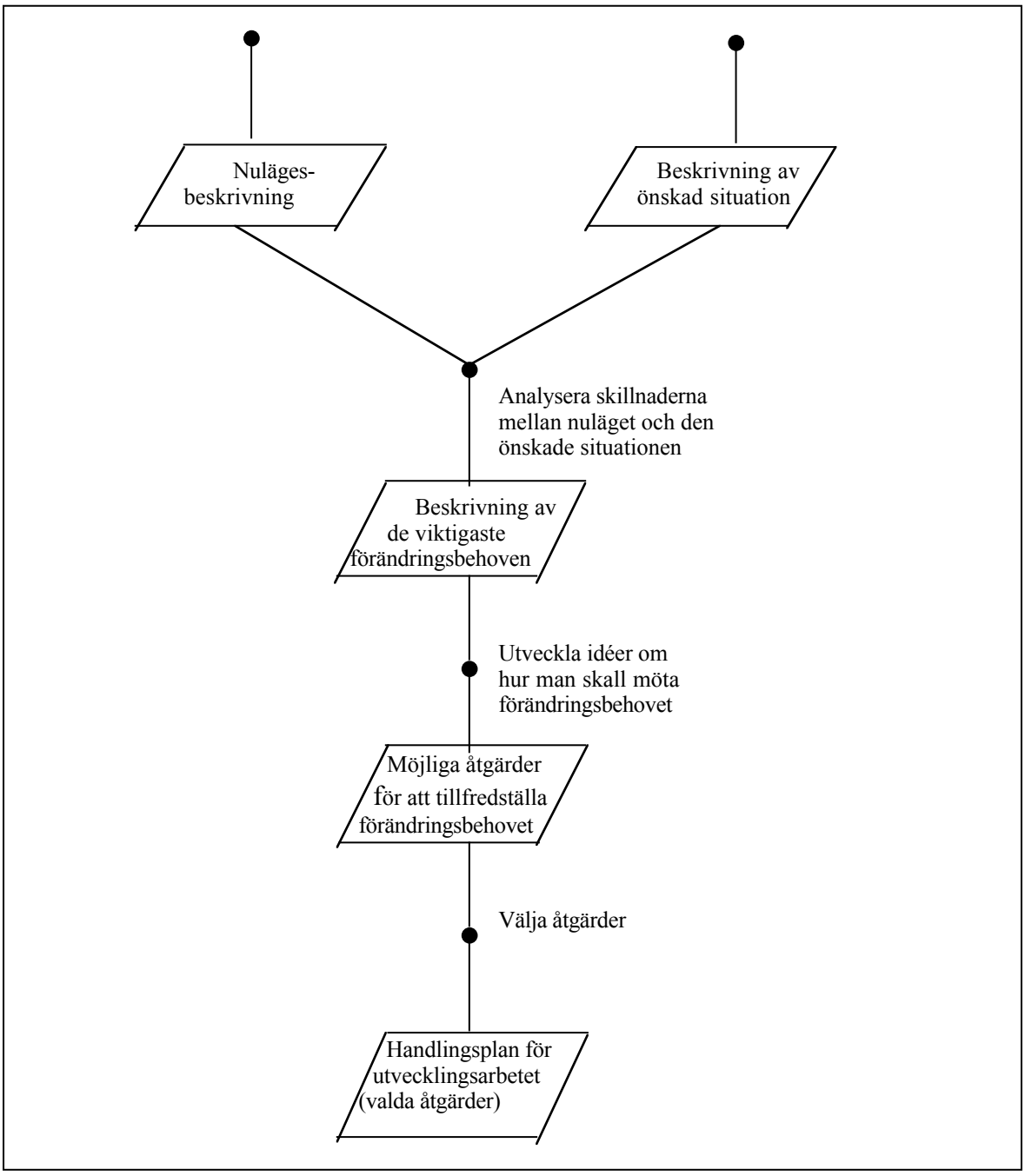
Informationssystemet kommer troligtvis någon gång att avvecklas, antingen på grund av att det har tjänat ut sin roll och byts ut mot ett nytt informationssystem, eller också kanske hela verksamheten läggs ner. Det är då viktigt att all information som informationssystemet tillhandahåller behandlas på rätt sätt, så att till exempel känslig information inte kommer i orätta händer.



Figur 3.1 visar systemutvecklingens livscykel, förändringsanalys bör göras innan. (Andersen, s.48)

3.7 Förändringsanalys

Innan varje systemutvecklingsprocess bör en förändringsanalys göras (Andersen, s.57).
I förändringsanalysen beskrivs dels den nuvarande situationen i verksamheten och dels den önskade situationen. En bra modell för att beskriva detta är Y-modellen, som visas på nästa sida.



Figur 3.2 visar Y-modellen (Andersen, s. 59)

4 ISAC-modellen

(Information SystemsWork and Analyses of Changes). På 1970-talet utformade en grupp forskare vid Stockholms Universitet en systemutvecklingsmodell som fick stor uppmärksamhet.
Den kom att kallas ISAC. Inspirationskällan till ISAC-modellen är professor Börje Langefors. ISAC-modellen är en relativt strikt form av systemutveckling, den är mycket grundlig och det finns detaljerade beskrivningar för varje fas. Erling S Andersen skriver följande i sin bok Systemutveckling – Principer, metoder och tekniker:

> ”Just det faktum att ISAC-modellen är så grundlig, förklarar både den framgång och det motstånd man då och då upplevt. Många tilltalas av en modell som ger detaljerade upplysningar om vad som ska göras från början till slut. Den typen av modell kan vara särskilt fördelaktig i en undervisningssituation. Men en stor detaljrikedom kan också skapa motstånd hos dem som har stor erfarenhet på området och själva vill bestämma arbetsgången.” (Andersen, 1994)

ISAC är vedertagen inte bara i Sverige, utan också internationellt. Modellen får ofta representera det skandinaviska synesättet på systemutveckling. Många högskolor och universitet använder sig av ISAC för att lära ut systemutveckling. Anledningen till detta är att den är välstrukturerad och relativt lättförståelig.

ISAC lägger tonvikten vid själva systemeringsdelen, men har utökats till att täcka även realisering, implementering och förvaltning av systemet. Enligt ISAC-modellen är det mycket viktigt att göra en förändringsanalys innan systemeringsarbetet börjar. Modellen förespråkar användarmedverkan. Det är viktigt att användarna medverkar under hela utvecklingsprocessen för att öka kvaliteten och få användarna att förstå systemet bättre.

Utifrån en beskrivning av verksamheten undersöks vilket informationsbehov som finns. För att komma fram till detta tar man hjälp av användarna, som medverkar med sin kunskap om de olika delarna i verksamheten. En s k informationsanalys görs.

4.1 Beskrivningstekniker i ISAC

En rad olika beskrivningstekniker används i ISAC, de används för att beskriva de olika delområdena i utvecklingen (Andersen, s.146 ff). De grafiska beskrivningarna är hierarkiskt uppbyggda och har alla olika symboler. Beskrivningsteknikerna delas in i tre olika grupper, efter vad de beskriver:

- **Beskrivningstekniker för verksamheten**

Verksamhetsbeskrivningen kallas SVB-teknik (Systematisk VerksamhetsBeskrivningsteknik). Man gör dels en strukturell beskrivning, där V-grafen är ett centralt begrepp och dels har man en egenskapsbeskrivande del, där man använder sig av egenskapstabeller.

V-grafen (verksamhetsgrafen) beskriver verksamhetens struktur. Den beskriver vilka inmängder verksamheten tar emot och vilka utmängder som levereras. V-grafen visar även vilka meddelanden och fysiska objekt som skapas. Här används begreppet meddelandemängd. En meddelandemängd består av ett eller flera meddelanden, och är något som ger ifrån sig information. Till den grafiska bilden finns en textsida som beskriver vad de olika meddelandemängderna innehåller.

Egenskapstabellen kompletterar V-grafen för att ge en så bra bild som möjligt av verksamheten. Egenskapstabellen visar hur lång tid det tar för en delverksamhet att omvandla en eller flera inmängder till en eller flera utmängder.

- **Beskrivningstekniker för informationssystemet**

För att beskriva informationssystemet analyserar man vilka informationsmängder systemet behandlar. Man letar sambanden mellan de olika informationsmängderna och hur de bearbetas. Det finns tre olika beskrivningstekniker som man använder sig av här.

I-grafer beskriver informationsmängderna och deras relationer. I-grafen är mer detaljerad än V-grafen och beskriver informationssystemet. Den visar vilken information som behövs för att få annan information, den visar även vilken information som skapas utifrån en annan information. I denna graf används begreppet informationsmängd i stället för meddelandemängd, men begreppen har egentligen ingen större skillnad.

K-graferna beskriver vad den enskilda informationsmängden innehåller. Den ger en detaljerad beskrivning över hur informationsmängderna är uppbyggda

Processtabellerna beskriver de olika informationsprocesserna. I processtabellerna beskrivs relationerna mellan olika informationsmängder. Här visas vilka meddelanden som måste finnas för att en informationsbearbetning skall kunna äga rum. Processtabellen visar även vilka beräkningar som skall göras på meddelandena.

- **Beskrivningstekniker för ADB-systemet**

D-grafen (Datasystemuttformningsgraf) visar hur informationsbehandlingen skall gå till i praktiken. Beslut har ännu inte fattats om exakt vilken utrustning som skall användas.

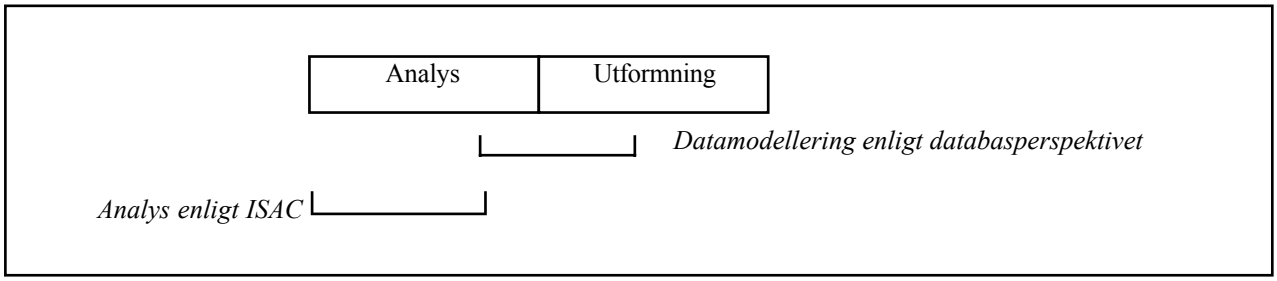
U-grafen (Utrustningsanpassad D-graf) beskriver hur utformningen skall ske när utrustningen är vald.

4.2 Var passar ISAC-modellen bäst?

ISAC-modellen lämpar sig bäst i relativt stabila verksamheter. Detta eftersom man lägger så stor vikt vid analys. Det är meningslöst att lägga ner så mycket tid med analysarbetet om verksamheten ofta genomgår stora förändringar.

5 Datamodellering

Datamodellering bygger på att användarna har en viktig roll i utvecklingsarbetet. Utifrån de erfarenheter användarna har om sin verksamhet kan den information som behövs för att skapa informationssystemet fås fram. Dessutom är det viktigt att studera dokument och anteckningar som är viktiga för verksamheten. Eftersom datamodelleringen inte är en heltäckande systemutvecklingsmodell kan arbetet kompletteras med en annan modell, till exempel ISAC. Då analyseras först verksamheten enligt ISAC, därefter används de utarbetade beskrivningarna för att göra en konceptuell datamodell. (Andersen, s.268 ff)



Figur 5.1 visar hur systemutvecklingsmodellerna kan kombineras (Andersen, s.307)

5.1 Entiteter, Attribut och Relationer

Viktiga begrepp i datamodelleringen är
- entitet
- attribut
- relation

En entitet är en passiv enhet som är intressant för att den ger information om någonting. I Andersens bok beskrivs entiteter på följande sätt:

> ”En entitet är något vi vill ha information om inom vårt intresseområde. Det kan till exempel vara en viss person, en viss sak (ett tillstånd), ett visst ställe, en viss tilldragelse eller en viss begreppsmässig konstruktion. En entitet kan vara något som faktiskt existerar, har existerat eller något som kommer att existera.” (Andersen, s.277)

Entiterna grupperas i *entitetstyper*, gemensamt för alla entiteter i entitetstypen är att de uppfyller defintitionskriterierna för entitetstypen.

Ett *attribut* kan sägas vara en egenskap hos en entitet. Attributet kan delas upp i två delar. Identifierande attribut som namnger entiteten, och beskrivande attribut som beskriver entitetens särdrag.

Entiterna i de olika entitetstyperna sammanbinds genom *relationer*, dels relationer mellan entiteterna i entitetstypen och dels mellan entiteter i olika entitetstyper. Relationerna mellan entiteter i olika entitetstyper kan delas upp i olika typer:

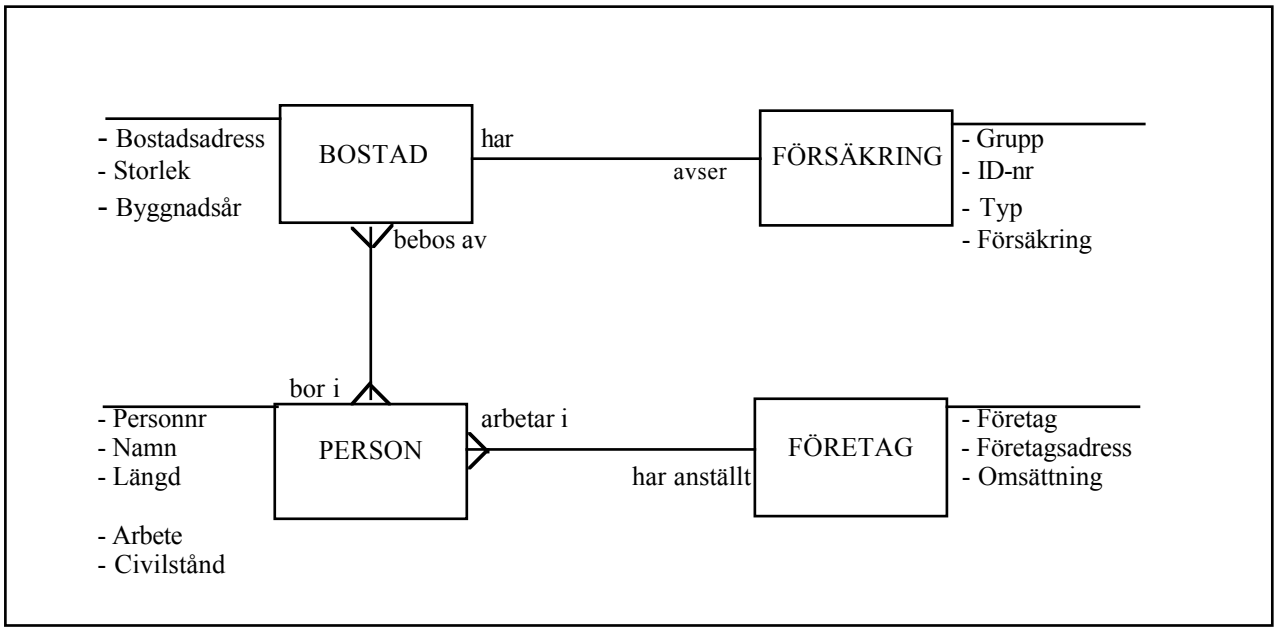
- 1:1 *En* entitet i den ena entitetstypen är relaterad till *en* entitet i den andra, och tvärtom.

- 1:N *En* entitet i den ena entitetstypen är relaterad till *en* eller *flera* entiteter i den andra entitetstypen. *En* entitet i den andra entitetstypen är dock endast relaterad till *en* entitetstyp i den första entitetstypen.

- N:N *En* entitet i den ena entitetstypen är relaterad till *flera* entiteter i den andra entitetstypen, och *en* entitet i den andra entitetstypen är relaterad till *flera* entiteter i den första entitetstypen.

5.2 Konceptuell datamodell och Logisk datamodell

Först görs en konceptuell modell av verksamheten, den visar de olika entitetstyperna, deras attribut samt relationerna mellan dem. Figuren nedan visar hur en konceptuell datamodell kan se ut.



Figur 5.2 visar exempel på konceptuell datamodell. Rektanglarna betecknar entitetstyper, beteckningarna vid sidan om är attributen för entitetstypen och linjerna mellan fyrkanterna visar relationer, 1:1, 1:N samt N:N (Andersen, s.291)

Utifrån den konceptuella datamodellen görs sedan en logisk databasmodell. I denna fas väljs vilken typ av databas som skall användas (hierarkisk, nätverk eller relationsdatabas). Först görs en normalisering, det vill säga den redundans som kan finnas i den konceptuella datamodellen tas bort.

Den konceptuella datamodellen görs om till ett antal tabeller, som sedan bildar databasen. Varje entitetstyp blir en tabell och spalterna i tabellen är entitetens attribut. Tabellerna karakteriseras genom att man säger att de är i första normalform, andra normalform eller tredje normalform (jag går inte närmare in på normalisering i denna rapport).

I realiseringsfasen skapas databasen utifrån den logiska databasmodellen. Register skapas för entitetstyperna och tillhörande attribut. Dessutom konstrueras själva programmet med gränssnitt och olika funktioner. Därefter kan informationssystemet implementeras.

6 Objektorienterad systemutveckling

Den objektorienterade systemutvecklingen är ett relativt nytt begrepp. Det härstammar från den objektorienterade programmeringen. I programmeringen kan man dra stora fördelar av att återvinna kod. Samma kod kan användas i en mängd olika sammanhang. Objektorienteringen har utökats till att även innefatta analys- och designfasen (Brown, s.162 ff, och Andersen, s.327 ff.)

Objektorienterad systemutveckling innehåller bland annat dessa två aktiviteter:

- Objektorienterad Analys (OOA)
- Objektorienterad Design (OOD)

Centrala begrepp inom objektorienterad systemutveckling är:

- Objekt
- Klass
- Arv
- Informationsgömning
- Inkapsling
- Polymorfism

Nedan följer en kort beskrivning av dem:

Ett *objekt* är en företeelse som har en identitet, ett tillstånd och ett beteende. Företeelser i verksamheten identifieras. De modelleras som objekt, deras egenskaper och beteenden beskrivs på ett så korrekt sätt som möjligt (Apelkrans och Åbom, 1990). Objekten är aktiva, de kommunicerar med varandra genom meddelandeskickning. Objektets *beteende* talar om vad det skall göra, objektets *metod* talar om hur det skall göras

De objekt i verksamheten som har gemensamma egenskaper delas in i en *klass*. En klas*s* är alltså en grupp objekt med samma beteenden och attribut.

Mellan de olika klasserna finns relationer. Det finns Superklasser och Subklasser. Superklassen är den mest generella klassen, den innehåller ett visst antal egenskaper. De objekt som finns Subklasserna ärver sina egenskaper från Superklassen och kan därtill få egenskaper som är speciella för denna klass.

Varje objekt har både yttre och inre egenskaper, objektet gömmer de inre egenskaperna och endast de yttre blir kända för de andra objekten. Objekten känner till varandras beteenden men inte metoderna. Detta kallas *informationsgömning* Ett annat begrepp inom objektorientering är *inkapsling.* Detta liknar till stor del informationsgömning och är ett sätt att gömma informationen om objektet så att det ser ut som en helhet.

Polymorfism innebär att ett objekt kan sända samma meddelande till objekt i flera olika klasser och att de har förmåga att tolka meddelandet på sitt speciella sätt. Detta är en stor fördel eftersom objekten då inte behöver specialanpassa meddelandet till varje mottagare.

6.1 Objektorienterad Analys

I den objektorienterade analysen tas de olika objekten fram och utifrån objekten kan informationssystemet skapas. Varje objekt blir en del som skall tas om hand av informationssystemet. Objekten delas in i klasser och sedan identifieras relationerna mellan klasserna.

6.2 Objektorienterad Design

Här bestäms hur informationssystemet skall se ut och fungera och utifrån det identifieras ytterligare objekt. Dessa skapas för att hjälpa programmet att utföra operationer som till exempel att peka på en knapp för att spara information i databasen.

7 Prototyping

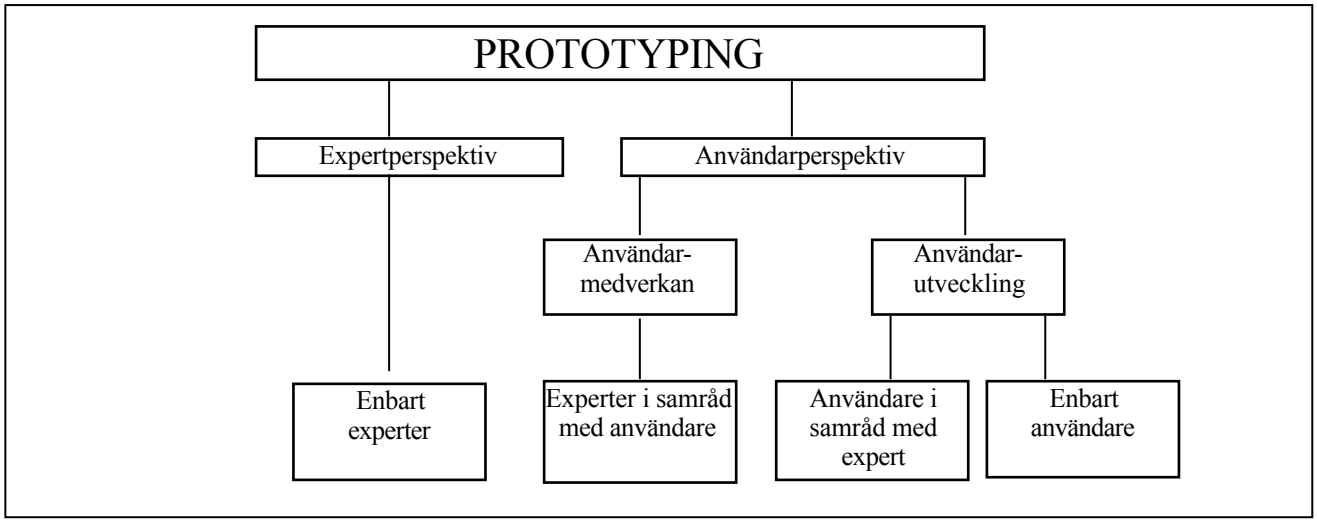
En metod för att utföra systemutvecklingsarbete är prototyping, det vill säga man utvecklar en prototyp som så mycket som möjligt liknar det tänkta färdiga systemet (Andersen, s.405 ff). Prototypen visar t ex användargränssnittet och vissa funktioner. Prototypen behöver inte klara de rent tekniska belastningarna, som t ex stora mängder data och många användare. Det viktiga är att användarna får chans att pröva prototypen och komma med synpunkter. Utifrån användarnas synpunkter kan man sedan förbättra prototypen och sedan visa användarna prototypen ännu en gång och så vidare. Genom att använda prototyping kan man bättre komma fram till hur det färdiga systemet skall se ut och fungera. Kravspecifikationen är alltså inte fastställd från början utan ändras successivt under systemutvecklingens gång.

Prototyping bygger på en utvecklingsmodell med flera faser, men trots att arbetet är strukturerat är det meningen att iterationer skall göras.

7.1 Grundtankarna bakom Prototyping

En av grundtankarna bakom prototyping är att få en bättre kravspecifikation. Kravspecifikationen fryses inte innan realiseringen börjar, utan förändras gradvis. En annan anledning till att använda Prototyping är att många anser att systemutvecklingsprocessen går fortare och mer effektivt, vilket dessutom leder till att arbetet blir billigare.

Det finns olika typer av prototyping (Friis, s.16 ff). Det finns expertprototyping eller så kallad rapid prototyping, dessutom finns så kallad användarprototyping. Vid expertprototyping är det enbart experten som konstruerar prototypen. Den första prototypen byggs snabbt, användarna testar prototypen och kommer med önskemål om förändringar varvid experten vidareutvecklar prototypen. På detta sätt arbetas prototypen stegvis fram mot en prototyp som stämmer överens med användarnas krav.



Figur 7.1 visar olika slag av prototyping och graden av användarnas medverkan (Friis, s.16)

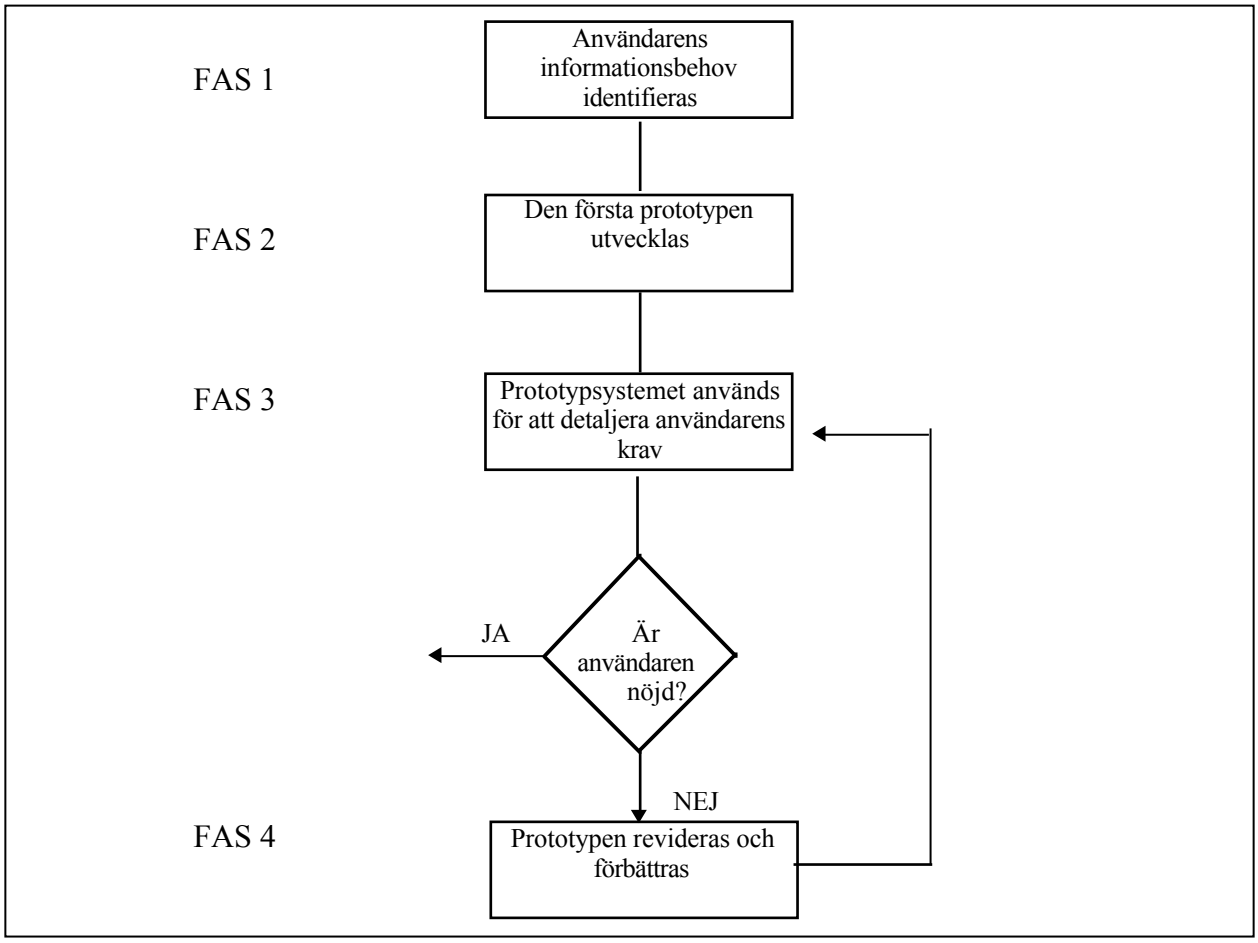
7.2 Expertprototyping enligt Jenkins modell

Professor Milton Jenkins, Indiana University beskriver hur expertprototyping bör gå till (Friis s. 283). I sin idealmodell finns endast en användare - ”designer” och en expert - ”byggare”. Fler användare och experter leder enligt Jenkins endast till att prototyputvecklingen tar längre tid, och till och med försämras.

Användaren bör ha tagit initiativet till att utveckla ett system, och sökt hjälp från experten. Användaren bör vara kompetent inom sitt område. Experten bör behärska alla tillgängliga verktyg för utveckling av prototypen, och bör dessutom vara insatt i den aktuella verksamhetens datastruktur.

Jenkins modell rymmer fyra faser:

1 Identifiering av användarens behov
2 Utveckling av en första prototyp
3 Testning av prototypen
4 Revidering av prototypen



Figur 7.2 Jenkins modell för prototyping (Friis, s. 29)

Informationsbehovet som användaren identifierar inför den första prototypen skall enbart utgöra de mest grundläggande kraven och önskemålen. Detta för att det skall gå fort att utveckla den första prototypen.

När användaren får testa den första prototypen är det förmodligen lättare att se vad användaren har för önskemål beträffande informationssystemet. Jenkins anser att det är lättare att kritisera och komma med synpunkter om det finns något konkret att utgå från.

Nu ändrar experten prototypen efter användarens krav. Vid ändring av informationssystemets omfattning är det viktigt att inkludera detta i kostnadsberäkningen.

Under arbetets gång pågår hela tiden ett utbyte av kunskaper mellan användare och expert. Efter varje test av systemet får användaren mer insikt i hur systemet byggts upp och fungerar. Dessutom lär sig experten mer och mer om verksamheten allt eftersom användaren kommer med nya synpunkter på prototypen. Ju mer iteration mellan de olika faserna, desto bättre blir informationssystemet och kommer bättre att motsvara användarens krav och önskemål. Iterationen mellan testning och revidering pågår till dess att användaren är nöjd med prototypen.

7.3 Användarprototyping

Vid användarprototyping är användarna mer eller mindre aktiva under prototyputvecklingens gång. Användarprototyping kan delas upp i ytterligare delar, användarmedverkan och användarutveckling. Användarmedverkan bygger på att användarna till viss del hjälper till med konstruerandet av prototypen. Vid användarutveckling utvecklar användarna själva prototypsystemet, experten finns vid behov tillgänglig som stöd. Vid användarutveckling är det viktigt att användarna är motiverade till att lägga ner mycket tid vid att arbeta med att utveckla sitt informationssystem.

> ”Frågan är då inte om användarna kan utveckla ett informationssystem själva, utan snarare om de vill. (---) Om experten kan utveckla ett för användaren bra informationssystem har användaren inte lust att lägga ner tid på att utveckla detta själv.” (Friis, s.47)

Det är dock en fördel om användarna är engagerade i utvecklingsprocessen, även om det är experten som utarbetar prototypen.

8 Beskrivning av företag och undersökningsmiljö

Min undersökning har gjorts på Funktionell Organisation, ett företag med sex anställda. De utvecklar och säljer bl a ett administrativt system som de kallar WinBas. Systemet innehåller bland annat funktioner för lagerhantering, kundregister, leverantörsregister, kund- och leverantörsreskontra och redovisning. Till detta kan sedan kunden få specialanpassade systemfunktioner. WinBas används så långt det går, och sedan skapas anpassningar, det vill säga egna program som samarbetar med WinBas.

Grundtanken bakom WinBas är att gränssnittet skall se ut som övriga Windowsprogram. WinBas har dessutom öppningar mot Excel, Access och Word. WinBas passar inom de flesta branscher, många av Funktionell Organisations kunder är grossist-, transport- och tjänsteföretag. Företagen som använder WinBas är medelstora med upp till 30 användare.

Funktionell Organisation har en supportavdelning som användarna dagligen kan nå via telefon, telefax eller e-post. Supportavdelningen hjälper användarna om problem eller svårigheter uppstår. Varje problem, fel eller önskemål som tas emot lagras i en kunskapsdatabas. Det är sedan enkelt att söka i databasen om samma fel har uppstått tidigare hos andra användare. I kunskapsdatabasen är det dessutom lätt att se om många användare har haft liknande önskemål om förändringar och förbättringar av WinBas. Utifrån detta kan WinBas kontinuerligt förbättras och nya funktioner läggs till eller tas bort. Ett par gånger om året skickas en ny version av WinBas ut till användarna med förbättringar och tillägg av funktioner.

8.1 Undersökningsmiljö

Den uppgift jag fick var att konstruera ett bokningssystem för ett företag som säljer språkresor. Bokningssystemet skulle ingå som ett försystem till WinBas.

Verktygen jag hade för att skapa systemet var Visual Basic 3.0 för att konstruera själva informationssystemet och Access 2.0 för att konstruera databasen. Kravet på bokningssystemet var att man skulle kunna utföra bokningar, lagra information om skolor, kurser, logiorganisationer, försäkringar och kunder. Utifrån bokningssystemet skall sedan ett fakturaunderlag skapas och därefter skall en order automatiskt registreras i WinBas.

8.2 Bokningssystemet

Databasen består av ett antal grundregister. I grundregistren uppdateras informationen om skolor, kurser, logiorganisationer, försäkringar och kunder. Dessa grundregister med tillhörande gränssnitt färdigställdes först så att användarna kunde få tillgång till dessa delar innan hela bokningssystemet var färdigt.

På Funktionell Organisation är det brukligt att gå till väga på detta sätt för att driften av informationssystemet skall kunna komma igång så fort som möjligt. När användarna får den

första delen av informationssystemet är avsikten att de skall börja registrera data med en gång. Detta därför att detta ofta är en tidskrävande uppgift. Användarna får också chans att testa informationssystemet för att upptäcka eventuella fel och brister. Under tiden fortsätter arbetet med resterande delar av informationssystemet.

8.3 Analysarbete

När mitt arbete började hade det redan bestämts tillsammans med användarna hur systemet skulle se ut. En rad möten hade hållits då anteckningar gjordes över användarnas önskemål och utifrån det kunde en slags kravspecifikation utarbetas. Kravspecifikationen bestod dels av en specifikation över hur databasen skulle se ut, dels vilka skärmbilder som skulle finnas och dessutom förslag till hur utskrifterna skulle kunna se ut.

Under hela processen har användarna följt arbetet. Systemet har konstruerats successivt i dialog med användarna. Detta har medfört att programmet har ändrat form och funktion betydligt under utvecklingens gång. Under arbetets gång har användarna kommit med nya synpunkter och nya idéer, vilket har resulterat i att kravspecifikationen ständigt förändrats.

8.4 Utarbetning av startprototyp

Det systemutvecklingsarbete som bedrivits för att konstruera det aktuella bokningssystemet kan närmast liknas vid någon form av prototyping. Jag har inte följt någon speciell metod för att utveckla bokningssystemet och några egentliga direktiv från Funktionell Organisation över hur arbetet skulle gå till gavs inte heller.

När konstruktionen av den första prototypen skulle börja hade alltså kravspecifikationen redan utarbetats fram. Utifrån den utvecklades en första prototyp, bestående av databasen och av gränssnittet, samt ett fåtal funktioner, såsom att uppdatera poster i databasen. Den första prototypen var tämligen ofullständig, den var endast till för att användarna skulle få en första bild av hur systemet var tänkt att se ut och fungera. Prototypen visades upp för användarna, och korrigerades utifrån deras synpunkter.

8.5 Installation av bokningssystemet första prototyp

Utifrån de anteckningar som gjordes kunde prototypen vidareutvecklas. Efter ett par veckors arbete med prototypen gjordes en ny avstämning med användaren och ytterligare justeringar gjordes.

Nu var grundregistren och de tillhörande gränssnitten färdiga och lämnades till användarna för installation. Avsikten var alltså att användarna skulle börja registrera uppgifter om sina skolor och kurser. Detta misslyckades dock till viss del. Anledningen var att användarna nu kom fram till att de måste förändra sitt sätt att ta fram koder till skolorna och kurserna. Resultatet blev att uppdateringsarbetet försenades till viss del. Detta var förstås ett litet misslyckande, kanske borde vi tänkt på detta i ett tidigare skede, innan vi kommit så här långt i arbetet.

8.6 Vidare prototyputveckling

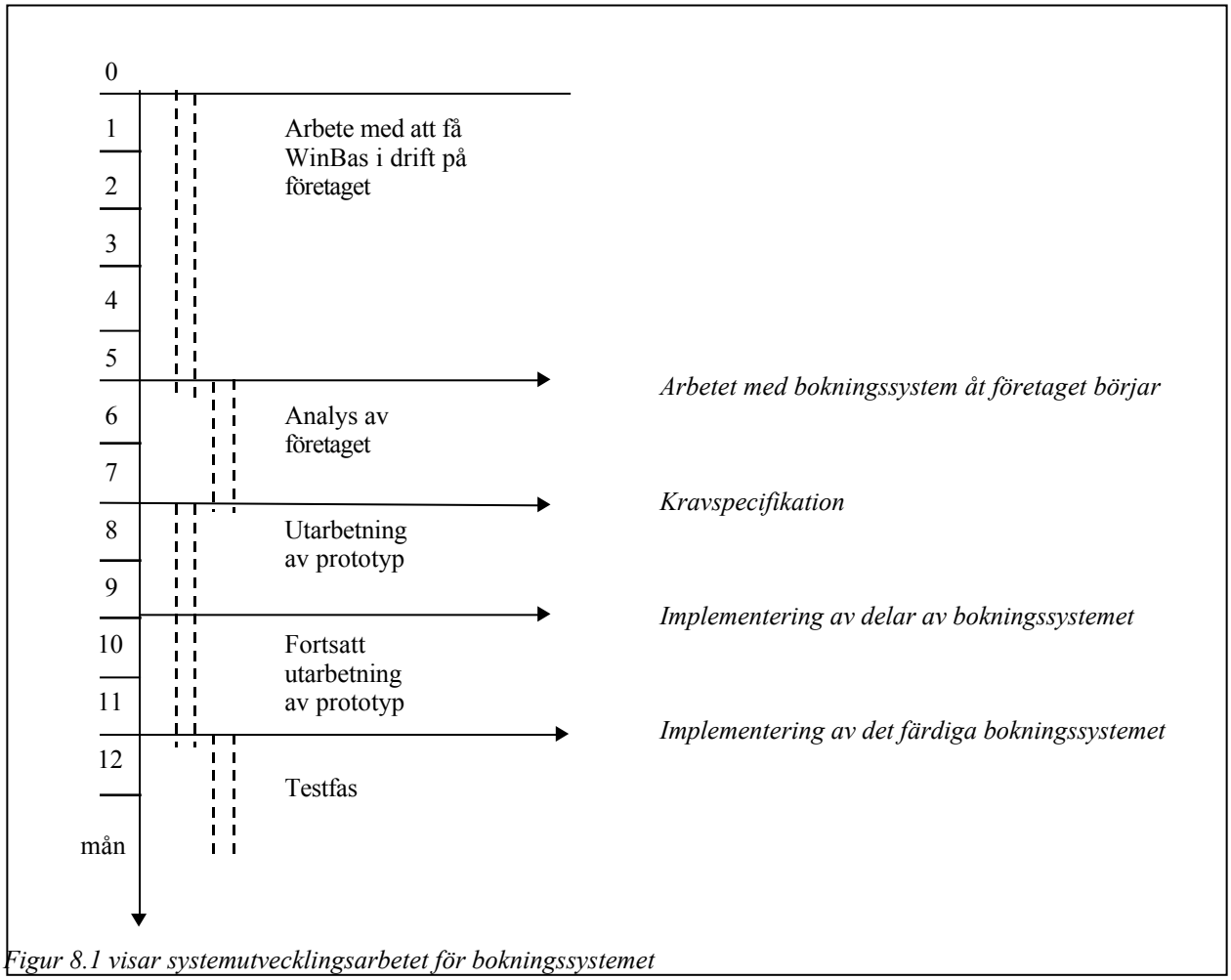
Under tiden fortsatte arbetet med prototypen och efter ytterligare ett antal veckors arbete och tester hos användarna var de till stor del nöjda med systemet. Nu började de dock komma med förslag till ganska stora förändringar. De önskade genomföra förändringar såväl i den egna verksamheten som i informationssystemet. Arbetet med informationssystemet började dra ut på tiden. Deadlinen för systemets installation var faktiskt redan överskriden. Det är då viktigt att klargöra för alla parter vad de önskade förändringarna kommer innebära för systemutvecklingsarbetet. Kommer förändringar och tillägg göra att arbetet drar ut mycket på tiden? Skulle de önskade förändringarna göra att kravspecifikationen skulle behöva ändras alltför mycket? Skulle det påverka den beräknade kostnaden av systemet?

Slutsatsen drogs att det var värt att utföra dessa förändringar. Systemet skulle förbättras avsevärt och det skulle inte innebära alltför stora kostnads- och tidsförändringar. Användarna ville att vi hellre skulle lägga ner extra tid på systemet och göra de förändringar som önskades. Förändringarna skulle leda till tillägg och förändringar i databasen samt nya gränssnitt med olika funktioner i bokningssystemet.

Det började nu märkas tydligt att användarna blev allt mer engagerade i sitt nya informationssystem. De blev väldigt entusiastiska och lade ner mycket tid och möda på att komma med nya idéer. Detta var förstås roligt och det gjorde att många bra synpunkter kom fram. Ett problem var att jag hade svårt att sätta stopp för deras önskemål om förändringar och förbättringar, det var väldigt lätt att dras med i deras entusiasm och genomföra alla önskemål. Detta var dock inte helt lyckat, arbetet måste trots allt någonstans avslutas, annars skulle vi kunnat hålla på i evigheter. De ledande på Funktionell Organisation började nu också inse att de kanske givit oss lite för fria händer vad det gäller utvecklingsarbetets gång.

8.7 Installation av systemet hos användarna

Tilläggen och förändringarna gjordes i alla fall och detta ledde till att systemutvecklingsarbetet dröjde ytterligare några veckor. Kopplingar till WinBas gjordes, och de olika utskrifterna konstruerades. Därefter visades prototypen för användarna som nu var nöjda med informationssystemet. Nu var det dags för installation av hela systemet så att det kunde börjas tas i drift. Under testfasen då systemet används parallellt med användarnas tidigare bokningssystem kommer antagligen användarna att komma med nya synpunkter och förslag till förändringar.



Figur 8.1 visar systemutvecklingsarbetet för bokningssystemet

9 Slutsatser

Ja, detta är alltså det systemutvecklingsarbete som har bedrivits för att skapa det beskrivna bokningssystemet. Vad kan man då säga om systemutvecklingsarbetet? Trots att bokningssystemet är relativt litet med ganska få riktigt avancerade funktioner och trots att systemet kommer att ha ganska få användare har systemutvecklingsarbetet varit tämligen omfattande och tidskrävande.

Hela utvecklingsarbetet från starten till implementeringen har tagit ca sex månader. Dessutom har det tagit ytterligare fem månader för att få WinBas i drift hos företaget, sammanlagt alltså drygt ett år för hela processen. Det är då lätt att förstå att detta varit en stor förändring för det aktuella företaget. Utvecklingsprocessen har tagit mycket tid i anspråk och kräver också god planering. Därför är det viktigt att användarna är motiverade till att lämna sitt gamla informationssystem för att skaffa ett nytt redan innan arbetet börjar. Viktigt är också att tänka på att det är en krävande process för företaget att gå igenom, både kostnadsmässigt och dessutom för personalen i verksamheten. Man får räkna med att detta blir en stökig och på många sätt jobbig period att gå igenom. Förhoppningsvis kommer dock detta att vägas upp av de effektiviseringar och förbättringar för verksamheten som informationssystemet är tänkt att innebära.

9.1 Problem som uppstått

Under utvecklingsarbetets gång har vi varit fyra personer som kontinuerligt har arbetat med systemet. Dels två av användarna på språkreseföretaget och dels två personer från Funktionell Organisation. Utvecklingsprocessen har varit ganska komplicerad, då och då har det uppstått en del oklarheter mellan de inblandade parterna. Något speciellt schema för hur arbetet skall bedrivas har inte följts, vilket har resulterat i delvis onödig tidsfördröjning. Kravspecifikationen har varit ganska ”luddig” och lite svår att tyda. Det har inte stått klart riktigt vad som skall och inte skall göras. Det är förmodligen till viss del detta som gjorde att utvecklingsarbetet drevs så långt som det gjorde. Hade vi haft klara mallar att gå efter hade det varit betydligt lättare att sätta stopp. Då hade alla varit på det klara med vad som ingick och inte.

Dessutom har kommunikationen mellan de inblandade ibland varit bristande, även detta har gjort att arbetet dragit ut på tiden och onödiga problem uppstått. Systemet är dock färdigt och användarna är nöjda med slutprodukten, med undantag för en del förändringar som behöver utföras.

Hade då dessa problem gått att undvika om en annan systemutvecklingsmodell använts? Jag har valt att jämföra med tre av systemutvecklingsmodellerna som beskrivits i tidigare kapitel.

9.2 Jämförelse med ISAC-modellen

Det systemutvecklingsarbete som utfördes kan inte på något sätt liknas vid ISAC-modellen. ISAC förespråkar en mycket detaljerad beskrivning av verksamheten, informationssystemet och adb-systemet.

Hade det varit bättre att använda ISAC-modellen? Nej, förmodligen inte. Eftersom ISAC-modellen är komplex och mycket detaljerad krävs det stora kunskaper i hur den skall användas, vilket jag inte har. Dessutom är ISAC-modellen alltför detaljerad för att det skall löna sig att lägga ner tid på att använda sig av den vid ett så här förhållandevis litet projekt. Troligen lämpar sig ISAC-modellen bättre för större systemutvecklingsarbeten. Vid riktigt stora projekt är det troligtvis betydligt viktigare att ha klara direktiv för vad som skall utföras. Dokumentationen över vad som har gjorts blir då också viktigare så att det är enkelt för alla inblandade att följa arbetets gång. Kanske är det till och med så att denna modell är något förlegad i dagens läge? I dag har man kanske inte tid att så detaljerat analysera och beskriva arbetet under systemutvecklingens gång.

9.3 Jämförelse med Datamodellering

Det systemutvecklingsarbete som bedrevs kan inte heller liknas med datamodellering. Ingen konceptuell datamodell gjordes, ej heller någon logisk databasmodell. Det som skulle kunna liknas vid datamodellering är de delar av kravspecifikationen som innehöll mallar för hur databasen skulle se ut med de olika registren.

Systemutvecklingsarbetet kunde mycket väl ha bedrivits enligt datamodelleringen principer. Dock kunde kanske en annan kompletterande modell än ISAC i så fall väljas. Datamodelleringen verkar vara en bra metod för att konstruera en databas.

9.4 Jämförelse med objektorienterad systemutveckling

Eftersom Visual Basic har använts för att programmera, lämpar sig inte objektorientering för den aktuella miljön, i alla fall inte i programmeringsfasen (C++ är till exempel ett objektorienterat programmeringsspråk). OOA och OOD skulle kunna ha använts. Dock skulle detta inte vara till någon nytta om inte OOP utnyttjas.

9.5 Jämförelse med Jenkins modell för expertprototyping

Om en jämförelse görs mellan Jenkins modell för expertprototyping och mitt arbete, ser man relativt många likheter.

- Bokningssystemet konstruerades i nära samarbete med användarna.
- Den första prototypen utarbetades ganska snabbt för att testas hos användarna.
- Därefter utarbetades och förbättrades prototypen successivt i dialog med användarna.

En skillnad mellan Jenkins modell och vårt arbete är att vi i vårt arbete var två användare och två experter inblandade i stället för en av varje som Jenkins förespråkar. Jag tror inte att detta har haft någon större negativ effekt, tvärtom har det gjort att många idéer har kommit fram som annars kanske inte hade uppmärksammats.

9.6 Slutkommentar

När vi arbetade med bokningssystemet kändes det ibland hopplöst, ingenting fungerade som det skulle och det mesta kändes misslyckat. Trots allt löste sig problemen efter hand och resultatet blev till slut bra. Så här i efterhand känns det som att systemutvecklingsarbetet ändå gick ganska bra, fast att det var lite väl ostrukturerat ibland.

Det ideala tillvägagångssättet för det systemutvecklingsarbetet som bedrivits tror jag skulle kunna vara en slags kombination av prototyping och datamodellering. Det jag saknade under utvecklingsprocessen var någon beskrivning att utgå från. Det hade varit bra att göra en konceptuell datamodell och en logisk datamodell, för att underlätta arbetet med att skapa databasen. ISAC-modellen hade varit alltför komplex för att använda. Den objektorienterade systemutvecklingen behöver man nog också studera ganska noggrant för att kunna använda sig av dess olika tillvägagångssätt vid ett systemutvecklingsarbete.

En stor fördel med prototyping tycker jag är att användarna blir aktivt deltagande i systemutvecklingsprocessen. De har stor chans att komma med synpunkter och önskemål under utvecklingens gång och kan se hur informationssystemet successivt fram och färdigställs enligt deras egna önskemål. Detta leder dessutom till att användarna blir entusiastiska och engagerade i sitt nya informationssystem.

Källförteckning

Andersen, E S, 1994, *Systemutveckling – principer, metoder och tekniker*, Studentlitteratur, Lund

Bansler, J, 1990, *Systemutveckling teori och historia i skandinaviskt perspektiv*, Studentlitteratur, Lund

Brown, D, 1997, *An introduction to Object-Oriented Analysis objects in plain English*, JohnWiley & Sons

Friis, S (projektledare), 1988, *Prototyping för högre grad av användarstyrning vid systemutveckling*, Lunds Universitet (Sex examensarbeten)