Examensarbete i Informatik

Användarcentrerad Systemdesign

En praktisk studie i användarcentrerad systemdesign med fokus på användbarhet.

Alexander Hansson, Saman Rahbari
Göteborg, Sweden 2008

REPORT NO. 2007:123

Användarcentrerad systemdesign

En praktisk studie i användarcentrerad systemdesign med fokus på användbarhet.

ALEXANDER HANSSON

SAMAN RAHBARI

Department of Informatics

IT UNIVERSITY OF GÖTEBORG

GÖTEBORG UNIVERSITY AND CHALMERS UNIVERSITY OF TECHNOLOGY

Göteborg, Sweden 2007

Användarcentrerad Systemdesign
En praktisk studie i användarcentrerad systemdesign med fokus på användbarhet.

User centred system design
A case study in user centred system design with focus on usability.

Alexander Hansson, Saman Rahbari

Report no. 2007:123
ISSN: 1651-4769

Department of Informatics
IT University of Göteborg
Göteborg University and Chalmers University of Technology
P O Box 8718
SE – 402 75 Göteborg
Sweden
Telephone + 46 (0)31-772 4895

Göteborg, Sweden 2008

User centred system design

A case study in user centred system design with focus on usability.

Alexander Hansson, Saman Rahbari

Department of Informatics

IT University of Göteborg

Göteborg University and Chalmers University of Technology

Abstract

This essay focuses on the term usability while conducting a user centred system design. Usability means to what extent a user can use a product to achieve effectiveness, efficiency and satisfaction. Development using a user centered system design means that knowledge about the end-users and it's opinions becomes a crucial part of the development process. The question of this essay is if using user centred system design will lead to good usability.

We used methods from a user centered system design point of view while developing an application for a Swedish bookstore company called Bokia AB. This included user and task analysis following by a close relationship between the developers and end-users by using sketches and showing them prototypes of the system while collecting feedback.

After concluding our case study we determined that using methods from user centered system design could be used as a great tool to further improve the usability and to improve an applications success chances.

This report is written in Swedish.

Keywords: user centered system design, usability

Innehållsförteckning

1. Inledning

1.1 Bakgrund

System som utvecklas idag har ibland problem med användarvänligheten. Ifall ett system misslyckas interagera med användaren försvinner nyttan med systemet. Ett system som skapas där användare och utvecklare har lite eller ingen kommunikation alls under utvecklandets gång kan sluta i en produkt som för utvecklarna är logisk men för användarna det motsatta. Vad som uppfattas som god användbarhet är olika från person till person beroende på systemets användningsområde och användarnas bakgrund. Vi har valt att begagna oss av en användarcentrerad systemdesignmetod som kännetecknas av hög kommunikation mellan de som utvecklar och de tänkta användarna. Vi har fått i uppdrag att utveckla en applikation där god användbarhet står i fokus. Företaget som har beställt applikationen är Bokia. En av författarna arbetar sedan tidigare som konsult hos företaget.

1.1.1 Bokia

Bokia AB är verksam inom bokbranschen och använder MegaDisc som butiksdatasystem. Bokia har en handfull egenägda butiker och ungefär 90 stycken franchiseägda Bokiabutiker runt om i Sverige. I MegaDisc sköts försäljning, lagerhantering, statistik och orderläggning.

Bokia vill ha en applikation där en anställd, via ett grafiskt gränssnitt som är kopplat till en databas, kan kontrollera kassarapporter för kontroll. Kassarapporterna hämtas elektroniskt från butiksdatasystemet MegaDisc. Efter kontroll och redigering ska applikationen arkivera samt skicka vidare kassarapporterna till ett ekonomisystem.

1.2 Syfte

Syftet med den här uppsatsen är att för utvecklingen av en applikation åt Bokia kunna utnyttja en användarcentrerad systemdesign. Vi hoppas att detta i sin tur ska leda till att vi kan besvara vår frågeställning där vi fokuserar på begreppet användbarhet. Detta innebär att vi kommer koncentrera oss mest på processerna före, under och efter själva programkodandet och vi lämnar därmed kodningen utanför uppsatsen då denna del inte anses relevant för att kunna besvara på vår frågeställning.

1.3 Frågeställning

Vår frågeställning är som följer: ***Kan god användbarhet uppnås med hjälp av användarcentrerad systemdesign?***

1.4 Avgränsningar

Vi har valt att inte fokusera på att praktiskt beskriva själva kodandet och dess struktur då vi inte anser att detta är relevant i samband med vår frågeställning. Vi kommer istället att fokusera på metoder som anses viktiga när man utvecklar enligt en användarcentrerad systemdesign. För en lyckad integration mellan vår applikation och Bokias existerande system har vissa anpassningar krävts för deras del. I brist på tid har de inte lyckats utföra dessa anpassningar och därför har vi valt att enbart simulera indata för slutresultatet istället för att fullt integrera applikationen i Bokias nuvarande organisation.

1.5 Struktur

Efter inledningen följer det första riktiga kapitlet i uppsatsen, metodbeskrivningen. I detta kapitel kommer vi att beskriva de metoder vi haft som bakgrund för utvecklingsprocessen. Detta följs av teorikapitlet där de bakomliggande teorierna samt den teknik som använts beskrivs . Därefter följer resultatkapitlet där vi beskriver vårt tillvägagångssätt enligt metodkapitlet och presenterar det resultat vi fått fram. Efter resultatkapitlet följer diskussion och slutsats där vi försöker knyta ihop uppsatsen genom att besvara vår frågeställning samt diskutera resultatet.

2. Metod

I detta avsnitt kommer vi att presentera vilka metoder vi använt under utvecklingsarbetet. Det finns ett flertal olika metodvarianter att följa för att bedriva användarcentrerad systemdesign. Vi har för den här uppsatsen valt att använda de metoder som Jan Gulliksen och Bengt Göransson beskriver i sin bok Användarcentrerad Systemdesign, 2002, och använda dessa som våra riktlinjer. Samtliga metoder som beskrivs av Gulliksen och Göransson är dock inte applicerbara för oss och det systemet som utvecklas i samband med uppsatsen, därför har vi valt att göra vissa anpassningar och begränsningar.

2.1 Inledande analyser

Första steget i en användarcentrerad systemdesign går ut på att förstå uppgiften som ska lösas. Grunderna till detta läggs i någon form av systembeställare som bidrar med en grundläggande kravspecifikation om vad systemet ska utföra. Nästa steg är att systemutvecklaren får en bättre inblick i organisationen genom att förstå användarna, deras uppgifter, hur de utför dessa i dagsläget samt vilken funktion systemet ska ha i relation till användarna. Vidare måste utvecklarna ta reda på användarnas kunskapsnivå, utbildning, träning, användningsfrekvens och arbetsmiljö. Dessa uppgifter kommer sedan visa sig vara kritiska i utvecklingsarbetet. Denna kunskap kan för det mesta inte tillhandahållas via systembeställaren så det gäller att intervjua och studera användarna på sin arbetsplats för att på detta sätt få bättre översikt över organisationen än den information som systembeställaren kan erbjuda. (Gulliksen och Göransson, 2002)

2.1.1 Uppgiftsanalys

Uppgiftsanalysen används för att ta reda på vilka uppgifter användarna utför samt hur de genomför dessa uppgifter. Tanken är att man efter en genomförd uppgiftsanalys ska kunna besvara varför en användare utför en viss uppgift samt hur ofta och hur lång tid det tar vid varje tillfälle. Men en väl utförd uppgiftsanalys ställer även fler krav, man bör också ta reda på vilka steg och hjälpmedel som behövs för att utföra uppgiften samt om användaren samarbetar med någon under uppgiftens gång. Uppgiftsanalysen anses väldigt viktig för att öka användbarheten i systemet då det ofta existerar för mycket funktioner i olika system vilket försämrar användarnas effektivitet. En väl utförd uppgiftsanalys minimerar risken för detta. (Gulliksen och Göransson, 2002)

Själva analysen kan genomföras på flera olika sätt men den vanligaste metoden är en intervju i samband med en arbetsplatsobservation. Frågor som kan användas i uppgiftsanalysen är:

- Varför gör du det?
- Hur gör du det?
- Varför utförs inte uppgiften på ett annat sätt?
- Kan det uppstå fel och vad händer i så fall?

Man bör undvika allt för direkta frågeställningar om hur användarna anser att deras arbetssituation kan förbättras och istället fokusera på specifika och konkreta situationer som dagligen sker för användarna. (Gulliksen och Göransson, 2002)

2.1.2 Användaranalys

Uppgiftsanalysen används för att svara på frågor angående de uppgifter som användarna utför. Nästa steg är att svara på frågor angående användarna och detta görs med hjälp av en användaranalys. Kärnfrågorna som ska besvaras är vilka typer av användare som finns, vad deras egenskaper är samt vilka som ska använda systemet. Under egenskaper vill man få reda på bland annat vad för erfarenhet och utbildning användarna har. Nyanställda med liten erfarenhet ställer andra krav på systemet än mer erfarna användare. Andra egenskaper man bör ta reda på är hur god datorvana

användarna har samt hur mycket tid, i samband med implementeringen, som kommer att användas för utbildning. Även andra faktorer som ålder, kön eller användare med fysiska hinder kan visa sig vara viktiga för det fortsatta utvecklingsarbetet. (Gulliksen och Göransson, 2002)

Precis som med uppgiftsanalysen kan användaranalysen utföras med hjälp av intervjuer och observationer men här godtas även en enkätundersökning. Med hjälp av användaranalysen vill man bland annat få ut designrekommendationer som underlättar användandet av systemet. (Gulliksen och Göransson, 2002)

2.2 Visuell presentation

Det finns ett flertal olika sätt att presentera hur ett system är tänkt att fungera samt se ut för användarna innan själva kodandet börjar. Det är viktigt att det utförs någon form av systempresentation innan man kommer allt för långt i kodandet då det vid detta steg kan finnas eventuella missförstånd eller problem som rör vad användarna förväntar sig gentemot vad utvecklarna förväntar sig. (Gulliksen och Göransson, 2002)

En vanlig metod är att använda olika typer av textbaserade scenarier eller bildbaserade ”storyboards”. Ett scenario ska beskriva hur det är tänkt att systemet ska uppföra sig när användaren utför en specifik uppgift. Det är viktigt att varje scenario utvecklas specifikt för den användargrupp som utför en viss arbetsuppgift. En ”storyboard” fungerar på ett liknande sätt och kan även användas i kombination med textbaserade scenarier. En ”storyboard” har som uppdrag att visuellt beskriva hur en viss uppgift sköts av en viss användargrupp. Dessa görs vanligen med papper och penna och kan ses som väldigt tidiga protyper av systemet. Genom att använda sig av dessa tekniker får man tillgång till informationsrika möten med användarna där man kan diskutera och utvärdera olika lösningar och förslag. (Gulliksen och Göransson, 2002)

2.3 Prototyper

En prototyps syfte är att ge utvecklarna och användarna en översikt över hur det färdiga systemet kommer att se ut och fungera innan det är helt färdigutvecklat. Utnyttjandet av prototyper vid systemutveckling innebär en rad olika möjligheter så som att prova funktionalitet, hitta svagheter samt testa prestanda och utseende. Genom att kontinuerligt ha kontakt med användarna och utvärdera olika prototyper, databaserade eller pappersskisser, får man en bättre återkoppling och kan i ett tidigare utvecklingsstadium ta hand om förändringsförslag. (Gulliksen och Göransson, 2002)

När man utvecklar en prototyp finns det tre olika prototypsmodeller som man kan fokusera sig på beroende på vilken målsättning prototypen har. En vertikal prototyp innebär att fokus läggs på en viss del av systemet och på detta sätt kan man testa funktionalitet och gränssnitt för just det avsnitt som testas, detta är också den mest realistiska prototypen då allt är tänkt att fungera som i ”verkligheten”. Den andra prototypmodellen är en horisontell prototyp där man exempelvis visar hela systemets gränssnitt utan någon fungerade funktionalitet eller data. Med hjälp av denna prototyp får användaren en bra inblick i hur det färdiga programmet kommer att se ut och användbarheten kan utvärderas. Den sista typen är en scenariobaserad prototyp, denna prototyp liknar den vertikala till viss grad men istället för att fokusera på en del av systemet är fokus på en arbetsuppgift som användaren ska utföra. (Gulliksen och Göransson, 2002)

Maclean, Young, Bellotti och Moran (1991) har utvecklat ett tillvägagångssätt som kan användas antingen vid den visuella presentationen med skisser och scenarios eller vid prototypvisningarna. Deras modell kallas QOC och står för Question, Option and Criteria. Med hjälp av denna modell tar man reda på eventuella problem eller frågor som finns i gränssnittet för att sedan ta fram olika lösningar på dessa baserade på etablerade kriterier som t.ex. att systemet ska vara översiktligt eller att det ska vara lätt att använda för nybörjare. (Gulliksen och Göransson, 2002)

2.4 Utvärderingar

Utvärderingar har en viktig och central roll i användarcentrerad systemdesign och fokus läggs på god återkoppling mellan utvecklare och användare. Utvärderingar är också viktiga ur ett finansiellt perspektiv då eventuella fel eller förslag tidigare kan fångas upp av utvecklarna vilket innebär att det blir mindre kostsamt att göra korrigeringar. Ju senare i utvecklingsprocessen fel upptäcks, ju dyrare blir det oftast att göra korrigerar. Det är också viktigt att poängtera att utvärderingar bör ske i samarbete med verkliga användare och inte enbart med utomstående experter eller beställare. (Gulliksen och Göransson, 2002)

Det finns ett flertal olika metoder för att få användarna delaktiga i utvärderingsprocessen. De enklaste metoderna är användarobservationer, frågeformulär och intervjuer. Användarobservationer innebär att man samlar in information om användarnas beteende och prestationer då specifika uppgifter utförs. Här kan man exempelvis använda sig av scenariobaserade hjälpmedel under utvärderingen. Det innebär att användarna får diverse uppgifter att utföra med hjälp av systemet och medan dessa uppgifter utförs studerar man dem. Det kan också bli aktuellt att ställa frågor under arbetets gång, exempelvis om hur de upplever systemet. (Gulliksen och Göransson, 2002)

Frågeformulär- och intervjumetoden är ganska likartade, den enda egentliga skillnaden förutom att en intervju bör ske muntligt medan ett frågeformulär kan ske skriftligt är att man med en intervju också har större flexibilitet att bygga på med följdfrågor baserat på svaren som man får från användarna. (Gulliksen och Göransson, 2002)

Mer avancerade metoder är exempelvis deltagande utvärderingar där man blandar ett antal användare och experter för att tillsammans utvärdera ett system. Gruppen leds oftast av en expert på användbarhet och tillsammans går gruppen igenom ett antal scenarion som simulerar användarnas uppgifter. Här är det viktigt att man diskuterar varje steg som användaren måste utföra för att kunna identifiera eventuella problem. Denna metod bör användas tidigt i utvecklingsprocessen då man i ett tidigt skede kan få många olika synvinklar och synpunkter på systemet. Det är därför viktigt att man låter samtliga användare delta i diskussionen så mycket som möjligt. (Gulliksen och Göransson, 2002)

Det finns också en möjlighet att anlita en tredjepart, som agerar som en expert, som tar hand om utvärderingen. Denna person bör ha väldigt goda kunskaper om användbarhet samt design- och utvecklingserfarenhet. Personen får då i ansvar att gå igenom systemet och leta efter designproblem eller fel. En nackdel med denna utvärderingsmetod är dock att dessa experter ofta saknar kunskaper om användarna och dess uppgifter som står för den centrala delen i utvecklingsprocessen. Systemet bör vid en expertutvärdering också i en så stor grad som möjligt vara slutfört så att experten kan testa så mycket som möjligt vilket get problemet att eventuella problem som belyses kan visa sig väldigt kostsamma och tidskrävande ett korrigera. (Gulliksen och Göransson, 2002)

3. Teori

I denna del kommer vi att beskriva de allmänna teorier som vår uppsats bygger på.

3.1 Systemutveckling

För att bättre kunna förstå och se skillnaderna mellan olika systemutvecklingsmodeller kommer vi i detta avsnitt lite kort att ge en historisk bakgrund till olika systemutvecklingsmodeller. Själva begreppet systemutveckling inkluderar många olika delar. Det innefattar allt från kravhantering, design och konstruktion till verifiering och validering. Olika systemutvecklingsmodeller har som sitt primära syfte att beskriva de olika stadierna som ingår i utvecklingen men de ska även ange vilka kriterier som gäller innan man får gå vidare till nästa steg i utvecklingen. (Gulliksen och Göransson, 2002)

3.1.1 Tvåstegsmodellen

Innan några riktiga metoder hade tagits fram för systemutveckling användes en enkel tvåstegsmetod där krav, design, test och underhåll var utelämnat. Metoden såg ut på följande sätt:

1. Skriv kod.
2. Fixa till eventuella problem i koden.

Denna väldigt enkla metod stötte dock som väntat snabbt på flera problem:

- Programkod blev snabbt ostrukturerad.
- Koden blev dyr att underhålla.
- Användarnas behov blev inte tillfredsställda.

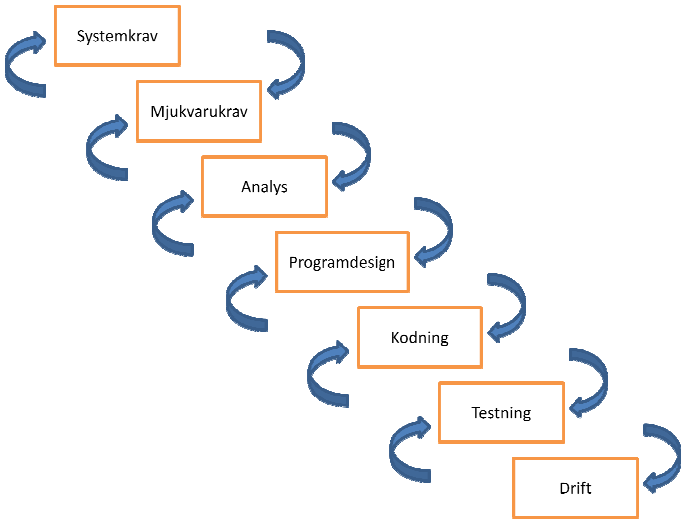
Dessa problem ledde till att det blev uppenbart att modellen behövde vidareutvecklas så att den innehöll någon form av planering, kravformulering, designfas och testning. (Gulliksen och Göransson, 2002)

3.1.2 Vattenfallsmodellen

Den överlägset mest förekommande modellen för dagens systemutveckling är vattenfallsmodellen som beskrevs av Winston Royce redan 1970. Modellen menar att mjukvara bör utvecklas i sekventiella faser där det finns etablerade milstolpar, dokument och granskningar i slutet av varje fas. Faserna ska inte heller överlappa varandra och det ska inte ges möjlighet att hoppa runt i faserna utan man definierar t.ex. att först göra klart system- och mjukvarukraven innan man går vidare med analysen. Som figur 3:1 illustrerar tillåter vattenfallsmodellen dock att man återgår till föregående steg. (Gulliksen och Göransson, 2002)

Ett av vattenfallsmodellens mest klara direktiv är att spendera mer och bättre strukturerad tid i början under de olika planeringsstadierna. Royce menar att man på detta sätt minskar risken för problem och buggar som kan uppstå senare i utvecklingsarbetet då det är mer tidskrävande och dyrare att korrigera dessa. Vattenfallsmodellen betonar också vikten av en god dokumentation före och efter varje steg, detta innebär att projekt ska kunnas läggas på is för att senare återupptas eller för att ett färdigt system ska kunna implementeras av en tredje part. (http://articles.techrepublic.com.com/5100-3513_11-6118423.html?part=rss&tag=feed&subj=tr#)

Kritikerna till vattenfallsmodellen menar dock att det läggs ner allt för mycket tid på oväsentlig dokumentation och att modellen inte är flexibel för korrigeringar och ändringar i efterhand. En eventuell ändring i systemet måste gå igenom ett flertal steg innan det kan genomföras vilket skapar problem för exempelvis de kunder som vill se färdiga prototyper innan de lämnar feedback. (http://articles.techrepublic.com.com/5100-3513_11-6118423.html?part=rss&tag=feed&subj=tr#)

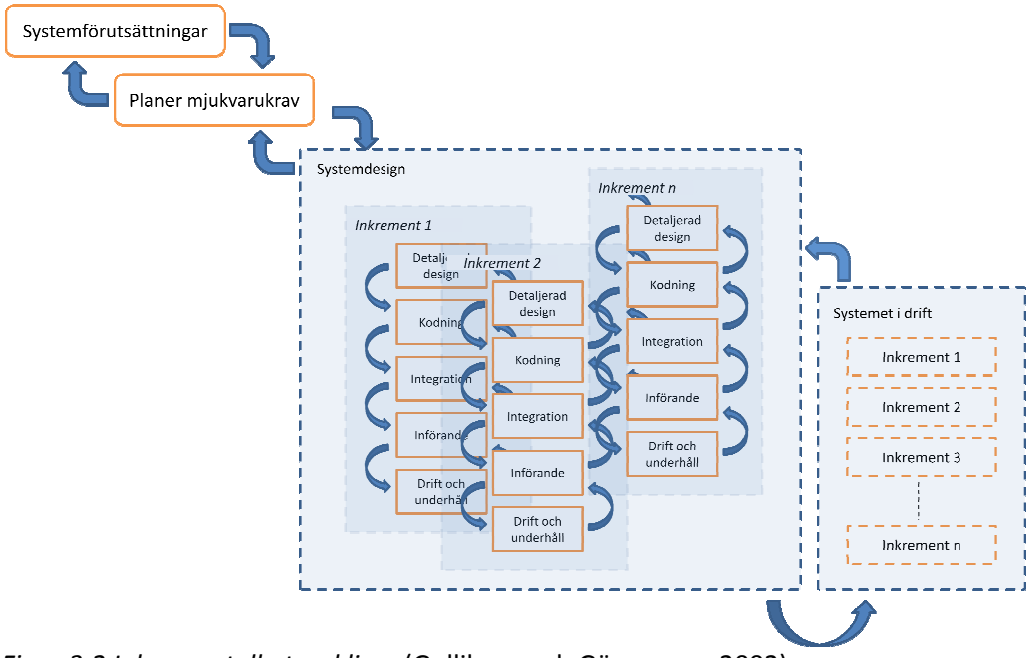


Figur 3:1 Vattenfallsmodellen. (Gulliksen och Göransson, 2002)

3.1.3 Inkrementell utveckling

Inkrementell utveckling innebär att man delar upp systemet i flera mindre delar som alla har en egen komplett livscykel liknande vattenfallsmodellens. Utvecklingsarbetet börjar med att man utvecklar en arkitektur för hela systemet och sedan planerar varje inkrement. Dessa inkrement kan sedan utvecklas både seriellt eller parallellt beroende på hur mycket tid och resurser som finns. Figur 3:2 illustrerar arbetssättet som används när man utför en inkrementell utveckling. (Gulliksen och Göransson, 2002)

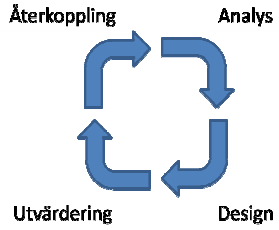
Genom att använda sig av inkrementell utveckling får man en bättre projektstyrning då varje delprojekt blir mindre. Det är enklare att testa olika mindre inkrement allt eftersom än att testa ett helt system. Inkrementell utveckling bidrar också till större flexibilitet då man kan utveckla initiala funktioner direkt och senare bygga vidare på dessa med mer funktionalitet. Detta bidrar till en större valfrihet för beställarna då de ständigt kan vara med och bestämma hur framtida inkrement ska fungera. (Gulliksen och Göransson, 2002)



Figur 3:2 Inkrementell utveckling. (Gulliksen och Göransson, 2002)

3.1.4 Iterativ utveckling

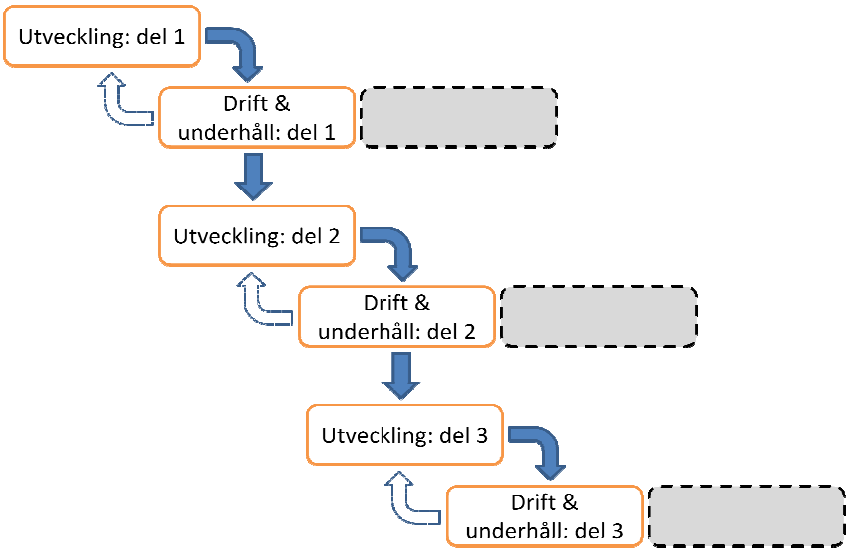
Grundprincipen bakom iterativ utveckling är att man inte kan göra allt rätt från första början då man inte vet hur problemet ser ut eller exakt vilka krav som finns. Iterativ utveckling innebär istället att man utvecklar och förfinar i efterhand, dvs. att samma moment utförs flera gånger som figur 3:3 visar. Efter flera iterationer lär man mer och mer om problemet och kan på detta sätt lättare komma fram till lösningar. Då en iterativ utveckling inte har något fördefinierat slut bör man i förhand bestämma vid vilka slutförda mål eller krav som man ska sluta iterera. (Gulliksen och Göransson, 2002)



Figur 3:3 Iterativ utveckling. (Gulliksen och Göransson, 2002)

3.1.4 Den evolutionära utvecklingsmodellen

Den evolutionära utvecklingsmodellen som visas i figur 3:4 kan ses som en kombination av vattenfallsmodellen, inkrementell samt iterativ utveckling. Grundtanken är att utvecklingen ska ske inkrementellt, men att varje inkrementell sedan ska utvecklas iterativt för att kunna ta del av de förändringskrav som uppstår under utvecklingstiden. Avslutningsvis sker en drift- och underhållsfas för varje inkrement där responsen från användarna tas omhand för eventuellt flera iterationer inom samma inkrement. När man väl avslutat ett inkrement återanvänder man de erfarenheter man fått för framtida inkrement. Den evolutionära utvecklingsmodellen är väldigt flexibel för användarkrav som tillkommer under utvecklingsarbetet. (Gulliksen och Göransson, 2002)



Figur 3:4 Evolutionära utvecklingsmodellen. (Gulliksen och Göransson, 2002)

3.2 Användarcentrerad systemdesign

Definitioner används ofta för att ge en kort och enkel beskrivelse över ett uttryck. I vårt fall saknas det dock en vedertagen definition för användarcentrerad systemdesign så det tolkas istället lite olika beroende på vem man frågar. Ett antal av dessa definitioner är:

”*A user centered design process is one that sets users or data generated by users as the criteria by which a design is evaluated or as the generative source of design ideas.”* (Dennis Wixon, 1996)

“*An approach which views knowledge about users and their involvement in the design process as a central concern*” (Preece, 1994)

“*For me, UCD is an iterative process whose goal is the development of usable systems, achieved through involvement of potential users of a system in system design.”* (John Karat, 1996)

Fokus läggs på två begrepp, **användare** och **centrerad**. ISO-standaren definierar begreppet användare som ”A *person/individual who interacts with the product/system”*. Det är viktigt att förstå skillnaden mellan beställare och slutanvändare. Dessa grupper är för det mesta helt olika och det är viktigt att som utvecklare involvera slutanvändaren så mycket som möjligt. Begreppet centrerad innebär att något placeras i mitten, i detta fall är det användaren och allt som görs ska utföras med fokus på användaren eller användaren. Vidare bör användarna vara aktiva under samtliga processer och inte enbart agera som passiva tyckare på den färdiga lösningen.
(http://www.acsd.se/acsd_kortom_m.php)

Gulliksen och Göransson (2002) har med hjälp av sina forskningsresultat och sin erfarenhet tagit fram en rad olika principer som de anser att man bör följa för att enligt dem utföra en verklig användarcentrerad systemdesign.

- Alla inblandande måste tidigt förstå verksamhetens mål samt användarnas situation. Med användarnas situation menas deras uppgifter samt varför och hur de utför sina uppgifter. Om man förstår sig på detta kan man lättare prioritera vad som är bra för användarna framför vad som är tekniskt möjligt. Intervjuer samt en uppgiftsanalys används vid det här stadiet för att enklare sätta sig in i verksamheten.
- Användare ska aktivt medverka tidigt och kontinuerligt genom hela systems livscykel. Här är det viktigt att förstå och se skillnaden mellan domänexperter samt verkliga användare. Domänexperterna är personer som är väl insatta i verksamheten men som inte agerar som slutanvändare av systemet.
- Systemet skall utvecklas iterativt och inkrementellt. Med detta menas att designlösningar bör itereras kontinuerligt med användarna. En iteration ska innehålla en analys av användarnas krav, en designfas och en dokumenterad utvärdering med konkreta förslag till förändringar. Inkrementell utveckling innebär att systemet stegvis utvecklas och att varje inkrement itereras tills dess uppsatta mål har uppnåtts.
- Designen skall dokumenteras så att samtliga parter enkelt kan förstå den. Man bör använda sig av konkreta presentationer så som olika prototyper och simuleringar för att på ett enkelt sätt visa den framtida användningssituationen för användarna. Prototyper är en viktig del och bör användas tidigt och ofta för att visualisera och utvärdera idéer och designlösningar med användarna.
- Specificera användbarhetsmål och utvärdera designen gentemot användbarhetsmålen tillsammans med användarna.
- Utvecklingen skall utföras av effektiva team med bred kompetens inom t.ex. systemarkitektur, databaser, programmering, informationsarkitektur och fältstudier.
- Användning av en erfaren användbarhetsförespråkare skall involveras tidigt och kontinuerligt under hela utvecklingsperioden. Denna person ska agera som en motor för den

användarcentrerade designprocessen och måste ges befogenhet att agera som en förespråkare under projektets gång.

- Alla delar i systemdesignen som påverkar användbarheten skall integreras med varandra så att de utvecklas parallellt, kontinuerligt och beroende av varandra.
- Anpassa den användarcentrerade processen för varje organisation. För detta krävs det att organisationen själv tar ansvar för hur detta bör utföras för att möta organisationen eller de enskilda projektens behov.
- En användarcentrerad attityd är viktig och bör upprättas bland utvecklingsteamets medlemmar. Detta görs genom att samtliga utvecklare träffar verkliga eller potentiella användare. Det är viktigt att samtliga personer inblandande är medvetna om vad användarna och vad användbarhet betyder

3.3 Användbarhet

I den användarcentrerade systemutvecklingsprocessen är användbarhet ett centralt begrepp, men vad menas egentligen med användbarhet? Den internationella standarden ISO definierar begreppet användbarhet på följande sätt: (http://www.acsd.se/anv_kortomanvandbarhet_m.php)

*”The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use” (*ISO 9241-11 Guidance on usability)

Med användbarhet menas därmed i vilken utsträckning en specifik användare kan använda en produkt för att uppnå specifika mål på ett ändamålsenligt, effektivt och för användaren tillfredsställande sätt, i ett givet sammanhang.

I dagens breda IT-marknad finns det nästan alltid flera alternativ till ett visst system och det är i slutändan alltid användaren som bestämmer om han eller hon vill använda produkten. Att då utveckla en produkt med fokus på att den ska vara lätt att använda bör ses som en prioritet om man inte vill tappa användare eller potentiella framtida kunder. Förr i tiden då marknaden var betydligt tunnare var ett vanligt synsätt att system agerade som en del av användarens arbetssysslor och att han eller hon därför var tvungen att lära sig systemet. Man la då inte lika stor vikt vid användbarhet och istället fick man hantera kostnaderna som ingick vid inlärning av nya system för användarna samt eventuellt andra problem som kunde uppstå vid dålig användbarhet. Exempel på detta är att en uppgift tar onödigt lång tid att utgöra, att onödiga fel uppstår samt att stress och otrivsel uppstår på arbetsplatsen. I värsta fall kan detta bidra till att man slutar använda produkten helt och istället ser det som en misslyckad investering. (Berndtsson och Ottersten, 2002)

God användbarhet ger en ökad produktivitet genom att systemen inte är för omständiga och användaren slipper ödsla tid på att fundera över hur produkten fungerar utan kan istället fokusera på uppgiften. I ett fall med dålig användbarhet kan produktivitet bli så lidande att användaren får spendera majoriteten av sin tid på att korrigera fel som systemet skapat. Vidare ger god användbarhet en minskad inlärningstid vilket ofta är livsviktigt för organisationer som har flera system som används sällan eller en organisation där det ständigt tillkommer nya anställda. Genom att använda, för användaren, bekanta begrepp och genom att utforma systemet så att den följer ett tydligt mönster kan man minska inlärningstiden. (Berndtsson och Ottersten, 2002)

Att utföra användbarhetsaktiviteter i utvecklingsprojekt kan också leda till minskade kostnader och kortare tid för utveckling. Mycket tid i diverse IT-projekt brukar läggas på att diskutera vad systemet egentligen har för syfte. Dessa diskussioner är viktiga men man kan istället använda konkreta tekniker för att formulera syftet för att sedan styra processen så att de förväntade effekterna

uppstår. Om det saknas kunskap om målgruppen och användningssituationerna riskerar man istället att projektet styrs för mycket av tyckande och detta ökar risken att det blir för många ändringar på vägen. Ökad förståelse för användarna och deras arbetssituation ger också bättre möjligheter att göra korrekta prioriteringar och man kan styra projekten så att deadlines och kostnadskrav följs. Detta hjälper i sin tur till att minimera risken för att projektet blir misslyckat då det är allmänt känt att många misslyckade IT-projekt beror på att man inte engagerat beställare och användare i tillräcklig omfattning. I ett projekt där man blandat in användaren under utvecklingsprocessen och där det sedan visar sig att systemet fungerar så som användaren förväntar sig bidrar till en ökad tillfredsställelse för användaren. (Berndtsson och Ottersten, 2002)

För att nå god användbarhet kan man titta på de fem punkter som Jakob Nielsen har tagit fram och som är hans konkreta syn på användbarhet.

- Lätt att lära: Användaren ska snabbt kunna komma igång med arbetet även om det handlar om ett nytt okänt system.
- Effektivt att använda: Efter att användaren har lärt sig systemet ska det vara lätt och effektivt att arbeta med.
- Lätt att komma ihåg: Det måste vara möjligt för användaren att efter en tids frånvaro fortfarande komma ihåg hur systemet fungerar.
- Få fel: Det är viktigt att minimera de fel som användaren kan göra, och utifall det blir fel måste det gå att kunna återgå till situationen innan felet uppstod.
- Subjektivt tilltalande: Användaren ska tycka om att jobba med systemet.
  (Nielsen, 1993)

3.4 Teknik

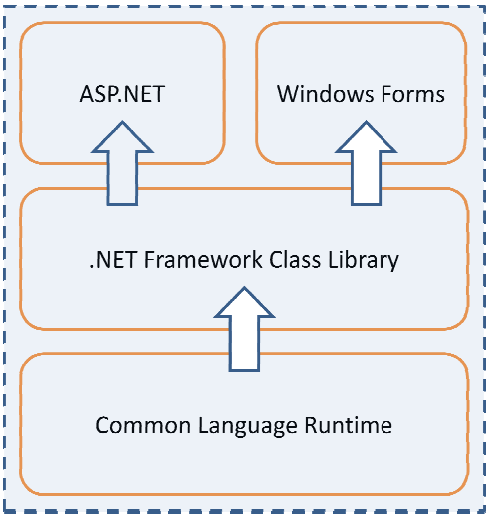
I följande teknikavsnitt kommer vi att beskriva de tekniska plattformar och standarder som vi använt oss av under utvecklingen av vår applikation. Vi har valt att göra en avgränsning genom att inte beskriva de tekniker som har en mindre betydelse för slutresultatet.

3.4.1 .NET

Microsoft har tagit fram ett ramverk för utveckling av applikationer kallat .NET. Ramverket kan liknas vid en plattform där utveckling kan ske i en inneslutande miljö oberoende av programmeringsspråk. .NET stödjer utveckling i exempelvis programmeringsspråken Visual Basic, C# och ASP.NET. (Aitken, 2002)

För att applikationer som är skrivna i .NET ska kunna köras krävs att ett .NET ramverk är installerat, för tillfället är ramverket endast officiellt tillgängligt på Microsofts egna operativsystem Windows vilket kan ses som en nackdel i jämförelse med andra utvecklingsplattformar, som exempelvis JAVA, som är plattformsoberoende.

Ett ramverk är inget nytt inom datorvärlden och Microsoft var inte först med att implementera en sådan lösning i sin .NET-plattform. Ramverket kan sägas hålla ihop alla delar som tillsammans utgör en applikation, ramverket pratar direkt med operativsystemet och utför de systemnära uppgifter som applikationen anropar. Dessa uppgifter är exempelvis filhantering, minnesallokering osv. .NET-ramverket består av flera mindre lager som i sin tur också innehåller egna lager och alltsammans är inbakat i ett ramverk. De fyra lager som visas figur 3:5 är de som vi kommer fokusera på och beskriva utförligare i detta avsnitt. (Ek och Overgaard, 2002)



Figur 3:5 Ramverk för .NET (Ek och Overgaard, 2002)

Strukturen är uppbyggd med Common Language Runtime längst ner i ramverket, sedan .NET Framework Class Library ovanpå CLR-lagret och tillsist Windows Forms och ASP.NET överst. (Ek och Overgaard, 2002)

Common Language Runtime, förkortat till CLR, är själva motorn i .NET och det är i CLR som de utvecklade applikationerna körs. CLR innehåller en mängd olika datatyper och gränssnitt som möjliggör att olika programmeringsspråk kan köras inom samma ramverk. CLR sköter även minnesallokering genom sin skräphantering vilket betyder att när en minnesallokering inte längre behövs så ges den tillbaka till operativsystemet. (Ek och Overgaard, 2002)

.NET FrameWork Class Library är ett klassbibliotek som är oberoende av något programmeringsspråk vilket gör att det kan nyttjas av såväl Visual Basic som C#. Klassbiblioteket innehåller alla klasser som anropas i en applikation. Klasserna används för filhantering, säkerhetshantering m.m. (Ek och Overgaard, 2002)

Windows Forms innehåller instruktioner för att skapa grafiska gränssnitt, Graphical User Interfaces, som exempelvis en knapp, ett formulär, en textbox eller en meddelanderuta. (Ek och Overgaard, 2002)

ASP.NET innehåller nätverksprotokoll och möjliggör nätverkskommunikation och databaskopplingar för applikationer. (Ek och Overgaard, 2002)

3.4.2 SIE

SIE är en standard för bokföringsfiler och är framtagen av SIE-föreningen. SIE-föreningen har som uppgift att tillhandahålla olika tekniska standarder för informationsutbyte mellan olika system och produkter. Med en standard menas ett format med fasta normer som följs för att det enkelt ska kunna anpassas till olika moduler och program. (www.sie.se)

SIE-formatet togs fram under 90-talet och syftet med standarden var att skapa en mall för att underlätta informationsutbyte mellan olika ekonomisystem. Den senaste versionen som används är SIE4, som togs fram 1998. Grunderna för SIE-formatet är att det ger möjlighet till:

- Utbyte av redovisningsdata mellan bokföringsprogram.
- Import av data från försystem såsom fakturerings och lönesystem.
- Export av bokföring till boksluts och inkomstdeklarationsprogram.
- Återföring av bokslutstransaktioner till bokföringsprogram.

(www.sie.se)

3.4.2.1 Beskrivning av SIE-formatet

En SIE-fil är en vanligt formaterad textfil, och i det här avsnittet kommer strukturen för en SIE-fil att förklaras. SIE-formatet inleds med ett såkallat huvud som beskriver filens innehåll:

```
#FLAGGA 0
#RAR 0 20060501 20070430
#SIETYP 4
#GEN 20071116
#KPTYP BAS2004
#PROGRAM MD
#FNAMN "BOKIA"
```

#SIETYP 4 talar om att filen är skapad i ett SIE4-format. #KPTYP beskriver kontoplanen, som i det här fallet är BAS2004. #FNAMN beskriver företagsnamnet och #PROGRAM är en indikator på vilket program som exporterat SIE-filen. (www.sie.se)

Varje verifikation i SIE-filen ska innehålla ett unikt verifikationsnummer. En verifikation innehåller konteringar som hör till en viss grupp. Huvudet på en verifikation ser ut som följer:

```
#VER "A" "910100673536" 20071001 "Dagsrapport 063100000013"
```

Det går att använda SIE-standarden på olika sätt och alla verifikationshuvuden behöver inte se ut på samma sätt. I exemplet ovan finns ett verifikationsnummer tillsammans med ett datum. Dessutom finns en beskrivning som talar om att verifikation rör en dagsrapport. (www.sie.se)

```
#VER "A" "910100673536" 20071001 "Dagsrapport 063100000013"
{
#TRANS 3010 {1 "0063" } -27841.90
#TRANS 2621 {1 "0063" } -1670.61
#TRANS 4010 {1 "0063" } 14763.63
#TRANS 1401 {1 "0063" } -14813.63
#TRANS 3110 {1 "0063" } -24217.81
#TRANS 2611 {1 "0063" } -6054.45
#TRANS 4110 {1 "0063" } 13666.69
#TRANS 1411 {1 "0063" } -13616.69
}
```

Exemplet visar en verifikation för en dagsrapport. En verifikation ska inledas och avslutas med en hakklammer. Varje transaktion inom verifikationen har en #TRANS-kod som innehåller information om belopp och vilket konto som ska debiteras eller krediteras. (www.sie.se)

3.4.3 SQL

SQL är en förkortning för Structured Query Language och är en standard för hantering av relationsdatabaser. År 1986 standardiserades det av ANSI (American National Standards Institute) och har sedan dess varit den mest vedertagna standarden för databashantering. (Groff, 2002)

SQL används för tre huvudsakliga mål och dessa är:

- Skapa databaser och strukturera dess innehåll, tabeller och kolumner.
- Skapa frågor mot databaser för att selektera data, infoga data och modifiera data.
- Kontrollera databasers säkerhet

(Wilton, 2005)

En SQL-fråga är en kort kod som skickas mot databasen som i sin tur ger tillbaka ett svar, detta sker genom ett subspråk kallat DCL (Data Control Language). När databasen får en fråga letar den upp svaret och skickar tillbaka ett svar i form av rader. Ifall frågan selekterar data från databasen skickas rader med information tillbaka till användaren och om frågan gäller bearbetning av data skickas ett svar tillbaka med antalet rader som hanterats. Med rader menas kolumnrader i databasens tabeller. (Wilton, 2005)

4. Resultat

I detta avsnitt kommer vi att presentera det resultat som framkommit med hjälp av vår metodbeskrivning.

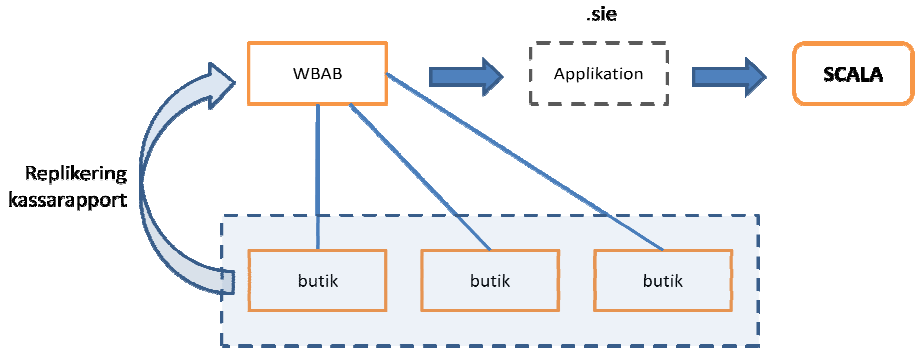
4.1 Inledande möten och analys

Det inledande analysarbetet påbörjades med en behovsanalys på Bokia tillsammans med två projektledare från företaget. Bokia förklarade att de har ett behov av att automatisera sin kassarapportshantering i flera led. I dagsläget sker samtliga steg manuellt av en enskild anställd och tanken är att en applikation istället ska sköta de flesta uppgifterna och bara lämna kvar enskilda kontrollmoment till användarna.

Under mötet förklarade projektledarna vilka de grundläggande kriterierna var samt hur de önskar att applikationen ska fungera. Tidigt förklarades att god användbarhet står i fokus för applikationen. Tillsammans arbetades en grundläggande kravspecifikation fram där Bokia formulerade de krav som ställs på applikationen.

- En självstående applikation som ska utvecklas i en plattform som är kompatibel med företagets nuvarande tekniska plattform.
- God användbarhet och ett bra användargränssnitt.
- Funktioner:
    - Applikationen ska kommunicera mot en SQL2005-databas.
    - Applikationen ska kunna hantera import och export av SIE4-filer.
    - Applikationen ska tillåta ändringar av kassarapporter.
    - Applikationen ska vara utrustad med en kvalificerad sökmodul för hämtning av kassarapporter.
    - Applikationen ska innehålla stöd för flera användare, inloggning m.m.
    - Applikationen ska kunna hantera utskrift av kassarapport på skrivare och skärm.

Vi blev också tilldelande en skiss över Bokias nuvarande system och dess subsystem samt den tänkta applikationens integration mellan systemen. Detta illustreras i figur 4:1.



Figur 4:1 Systemdiagram.

4.2 Uppgiftsanalys

För att öka vår förståelse över de uppgifter som applikationen är tänkt att utföra inleddes en uppgiftsanalys med en anställd på Bokia. Den anställde arbetar i dagsläget heltid med kassarapporter och mycket av den anställdes arbetsuppgifter sker manuellt.

Denna uppgiftsanalys påbörjades med en observation av den anställde samtidigt som vi ställde frågor som var relaterade till de uppgifter denne utförde. Den anställde fick gå igenom hela

utförandeprocessen för en kassarapport från det att den ankommer till henne tills den är rättad, godkänd och klar.

Genom observationen fick vi veta att den anställde dagligen får en kassarapport från varje butik som kontrolleras och denna rapport lämnas antagligen direkt i handen av en anställd från den berörda butiken eller så skickas den med posten. Den anställde visade också kassarapportens innehåll för oss samt förklarade hur hon kontrollerar att alla siffror är korrekta och stämmer överrens med det underlag hon får från butikens kassasystem. Kassarapportens innehåll består av en utskrift från butiksdatasystemet som visar dagens försäljning, kvitton som styrker olika uttag från kassan och ett kvitto som visar hur stor dagsinsättning i kontanter butiken gjort till deras eget bankkonto via en säkerhetsbox.

Det första steget i denna process är att i ett konstruerat Exceldokument markera den specifika kassarapporten som inkommen, dvs. att hon fått alla underlag för den dagens kassarapport. I Exceldokumentet har varje butik ett eget fält för varje dag och hon kan kontrollera att alla dagsrapporter har kommit henne tillhanda.

Den anställde använder sedan ett annat Exceldokument för att mata in de siffror hon får ifrån kassarapportens utskrift. Varje uppgift från rapporten har tilldelats en egen plats i Exceldokumentet och utifall någon siffra inte är korrekt korrigeras denna uppgift manuellt. Den anställde förklarade att ifall någon siffra inte stämmer så får hon ta kontakt med den person som i den aktuella Bokiabutiken har utfört kassarapporten så att de tillsammans kan reda ut de fel som finns. Hon förklarade att det ofta är så att dagsrapporten inte stämmer helt och hållet.

När kassarapporten sedan är helt rättad skriver den anställde ut Exceldokumentet i pappersform och placerar denna i en ”krokodil”, en såkallad mapp som har ett nummer för varje dag i månaden. Hon förklarade att när hon samlat ihop ett visst bestämt antal klara kassarapporter tar hon fram dem en och en och stansar in, dvs. matar in, i företagets ekonomisystem SCALA. I SCALA skapas transaktioner från olika konton och bokförs senare i företagets bokföring. Detta är det sista steget den anställde utför för varje kassarapport innan hon häftar ihop dem en och en tillsammans med ett försättsblad och placerar dem i en pärm som är daterad för varje månad och år.

Efter observationen var färdig utfördes en mer kvalitativ intervju för att mer ingående ta reda på uppgiftens syfte. Intervjun innehöll frågor om varför uppgiften utförs och hur ofta den utförs.

Den anställde svarade att uppgiften sker dagligen, då kassarapporter inkommer för varje berörd butik dagligen. Syftet med uppgiften är att kontrollera att dagsrapporterna stämmer. Hur lång tid varje dagsrapport tar att kontrollera och färdigställa hade hon svårt att svara på då det varierar beroende på hur korrekt utförda de är när de ankommer till henne men hon uppskattade att det ungefär tar 4-5 minuter i snitt för varje kassarapport exklusive inmatningen av siffrorna i Bokias ekonomisystem.

I dagsläget sker hela arbetsprocessen manuellt av en enskild person och det sker inte någon inblandning av någon annan anställd på företaget när hon utför uppgiften. De hjälpmedel som används utöver datorn är pärmar, ”krokodiler” och kassarapportsunderlag från butiksdatasystemet.

Kritiska faktorer och flaskhalsar som förekommer i uppgiften är ifall en kassarapport inte ankommer i tid, den anställde kan då inte utföra sin uppgift och samma sak gäller ifall kassarapporten innehåller handhavandefel, saknade poster eller felaktiga poster.

Vidare ansåg hon att en applikation skulle underlätta för henne då det skulle innebära att hon kan kringgå allt manuellt arbete som uppgiften består av idag. Effektiviteten skulle enligt henne öka och

många kontroller och handhavandefel i form av felinslag från den anställdes sida skulle minimeras. De fel som uppkommer ifall en kassarapport är felaktig är att fel summor eller konton blir bokförda i företagets ekonomisystem vilket i ett senare tillfälle skapar problem och som kräver ett stort merjobb för att korrigera.

4.3 Användaranalys

Vi träffade de användare på Bokia som var tänkta att använda den applikation som skulle skapas. Alla användare arbetar idag inom samma avdelning på företaget och har således snarlika arbetsuppgifter. Genom en enkät samlade vi in kvantitativ data för att lära känna de tänkta användarna. Enkäten skickades ut via e-post.

Enkäten var tänkt att bidra med mer information om användarna och deras bakgrund. Frågorna berörde både rena datorspecifika frågor och frågor om användarnas bakgrund som t.ex. deras företagshistoria och utbildning. Vi valde att även ta med frågor som svarar på vilka datorsysslor användarna utför för att få reda på komplexiteten i deras datoranvändande. Ett urval av enkätens frågor:

- Vilken är din nuvarande arbetssyssla?
- Vad har du för akademisk utbildning?
- Hur många timmar om dagen använder du datorn på jobbet?
- Hur stor andel (i procent) är surfande på internet (i arbetssyfte och fritidssyfte)?
- Hur stor andel (i procent) är arbete i övriga program och vilka är dessa i så fall?
- Hur mycket använder du datorn i hemmet?

Av de tillfrågade är majoriteten kvinnor och åldern på de svarande varierar från 35-46 år.

På den första frågan om vad deras nuvarande arbetssysslor är svarade samtliga relativt snarlikt då de alla arbetar inom ekonomiavdelningen på företaget. En tillfrågad arbetar som controller, två tillfrågade arbetar med kunder och leverantörer och den sista personen har kassarapporter som arbetssyssla.

De tillfrågade svarade varierat på frågan om deras akademiska bakgrund, en har högskoleexamen och de tre andra har en annan form av utbildning eller är självlärda.

På frågan om hur ofta de använder datorn på jobbet svarade samtliga att de använde datorn i princip alla arbetssysslor, en person svarade att den använder datorn ungefär 6 timmar (på en 8 timmars arbetsdag) medan de andra svarade att de använde datorn för samtliga sysslor.

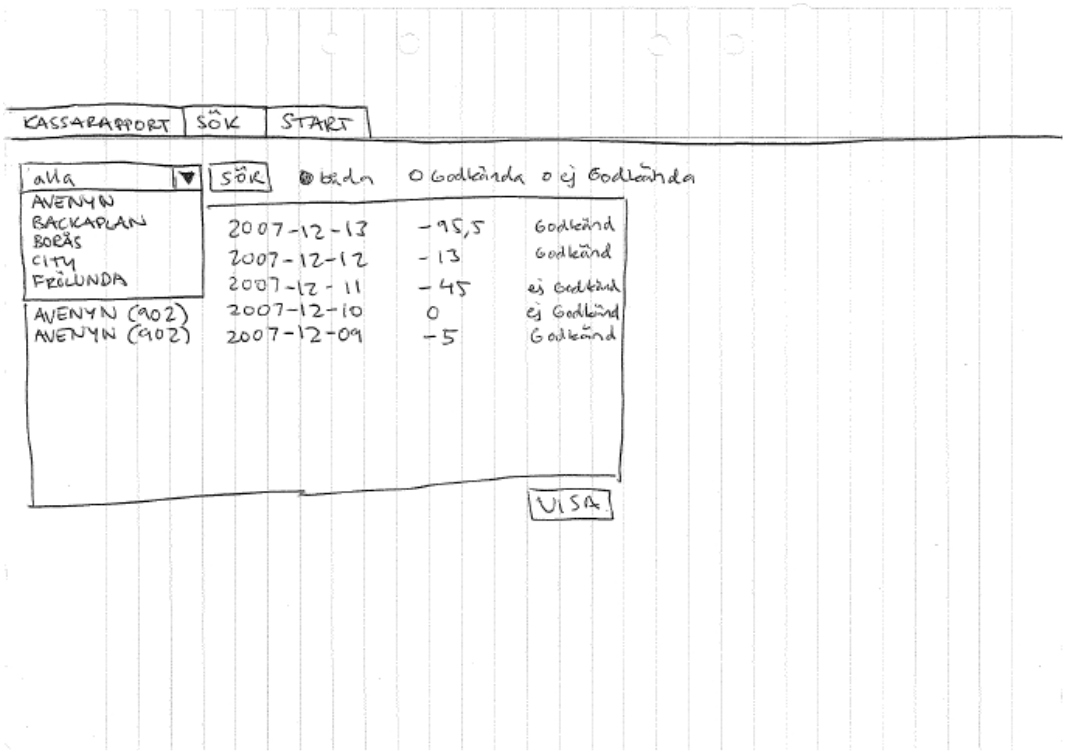
De tillfrågade svarade varierat på frågan om vilka sysslor de utför med datorn under arbetstid. Tre av de tillfrågade svarade att arbete i företagets ekonomisystem SCALA utgör deras största användningsområde för datorn och en svarade att kalkylprogrammet Excel är det mest använda verktyget. På frågan om hur mycket de använder Internet under arbetstid i arbetssysslor svarade alla att det är en liten eller mycket liten del av användningen (mindre än 10%).

På den sista frågan om hur ofta datorn används i hemmet gav de tillfrågade varierade svar. En svarade att datorn används mycket, medan två svarade att datorn används sällan eller mycket sällan. En tillfrågad svarade att datorn används 3-4 dagar i veckan. De sysslor som datorn används till i hemmet är varierande och består av allt från enkla bankärenden och shopping på Internet till att spela spel eller hantera digitalbilder.

4.4 Visuell presentation

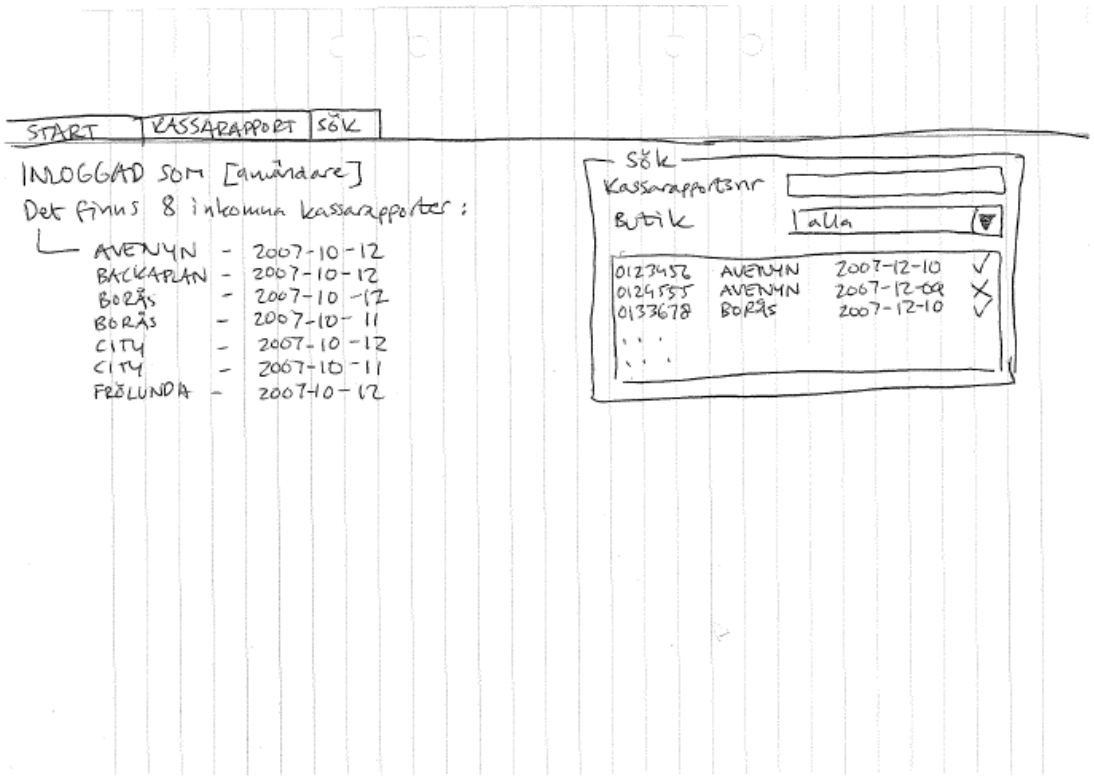
Tidigt i vårt arbete med applikationen utformade vi pappersskisser på applikationens tänkta utseende. Vi hade ett möte med den primära användaren av den tänkta applikationen för att användaren skulle få komma med förslag och idéer över utseendet.

På mötet visades enkla pappersskisser för användaren som representerade skärmdumpar på den tänkta applikationen, skisserna var utformade på ett sådant sätt att de enkelt skulle kunnas ändras efter behov. Mycket var också utlämnat så att vi och användaren tillsammans skulle kunna diskutera fram ett användargränssnitt. Figur 4:2 visar en skiss över den tänkta applikationens sökfunktion. Användaren hade önskemål om en mer utökad sökfunktion, där datumintervall för kassarapporterna kan anges. Dessutom ansåg användaren att en del information som visades vid sökträffarna var överflödig. Exempelvis ville användaren inte att differensen för varje kassarapport skulle visas.



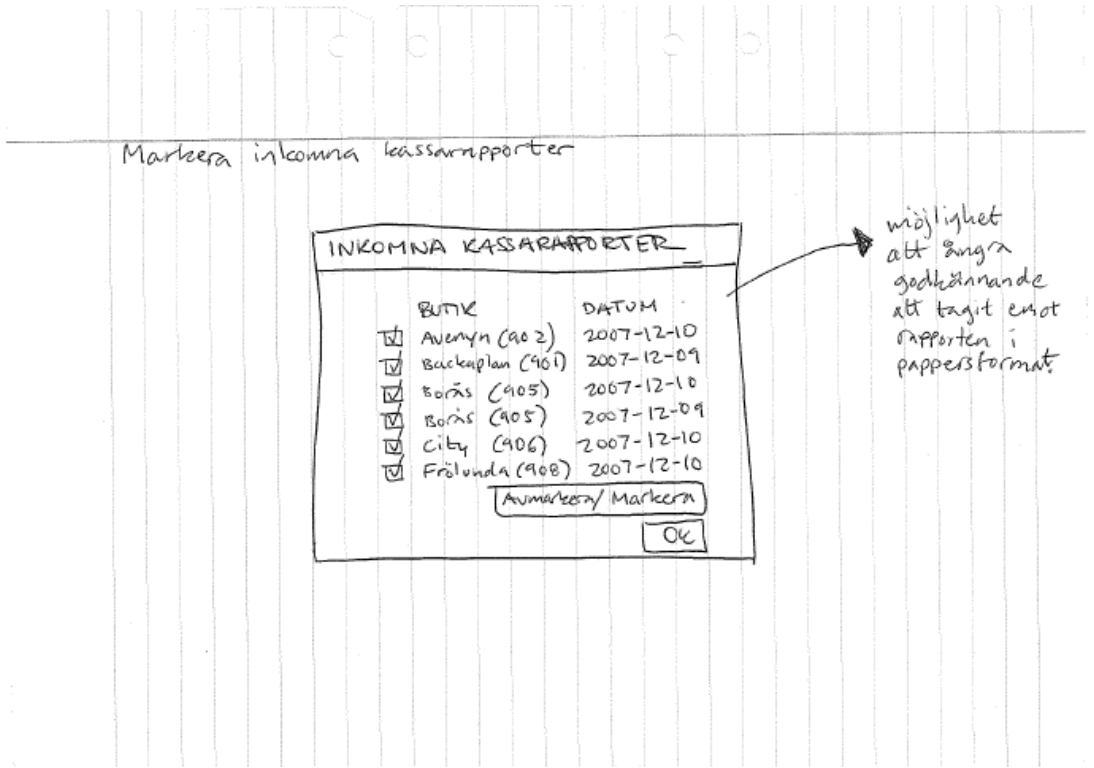
Figur 4:2 Skiss över sökfunktion.

Vi kom fram till att en startsida med information där alla grundläggande funktioner samlas är en bra lösning. Användaren ville ha en överskådlig sida som presenteras direkt vid programmets uppstart efter inloggning. Som figur 4:3 visar valde vi att under mötet skissa fram en prototyp på en startsida där information över händelser, snabbsökning och avancerad sökning presenteras. Vi valde att tillsammans implementera en navigering med ”flikar” där användaren kan leta sig fram mellan de olika menyvalen genom att klicka på ”flikarna”.



Figur 4:3 Skiss över startsida

Figur 4:4 visar en funktion där användaren får acceptera inkommande kassarapporter. Funktionen var ett förslag från vår sida för att ersätta den nuvarande rutinen där kassarapporter manuellt markeras som inkomna i ett Exceldokument.



Figur 4:4 Skiss över funktion där inkomna kassarapporter läses in i applikationen.

Innan mötet togs även en scenariobaserad prototyp fram och användaren fick tre scenarion att genomföra:

1. ”Ta emot” en inkommande kassarapport och markera en eller flera kassarapporter som inkomna i applikationen.
2. Kontrollera en kassarapport och godkänna den.
3. Söka upp en gammal kassarapport.

Under de tre olika scenariona fick användaren komma med egna synpunkter på hur uppgifterna bör kunna utföras och vi antecknade och ändrade i skisserna utefter användarens önskemål. Dessutom fick användaren komma med övriga önskemål om hur applikationen skulle utformas. QOC-metoden användes för att kunna sammanfatta hur funktionaliteten skulle utföras och resultatet av detta presenteras i figur 4:5.

QUESTION	CRITERIAS	OPTIONS
Hur ska kassarapporter presenteras?	• Överskådligt • Enkelt	• Flikar för varje kassarapport • Olika färgscheman. • Tydligt upplägg
Hur ska användaren ändra och godkänna en kassarrapport?	• Enkelt • Minimera fel	• Radiobuttons för att ändra status • Numeriska fält för ändring av siffror • Felkontroller
Hur ska användaren ”ta emot” en kassarapport?	• Enkelt • Minimera fel	• Tydlig lista med checkboxes för varje kassarapport • Felkontroller
Hur ska sökfunktionen fungera?	• Överskådligt • Enkelt • Snabbt	• Tydliga val för sökkriterier • Eget sökfält för snabbsökning

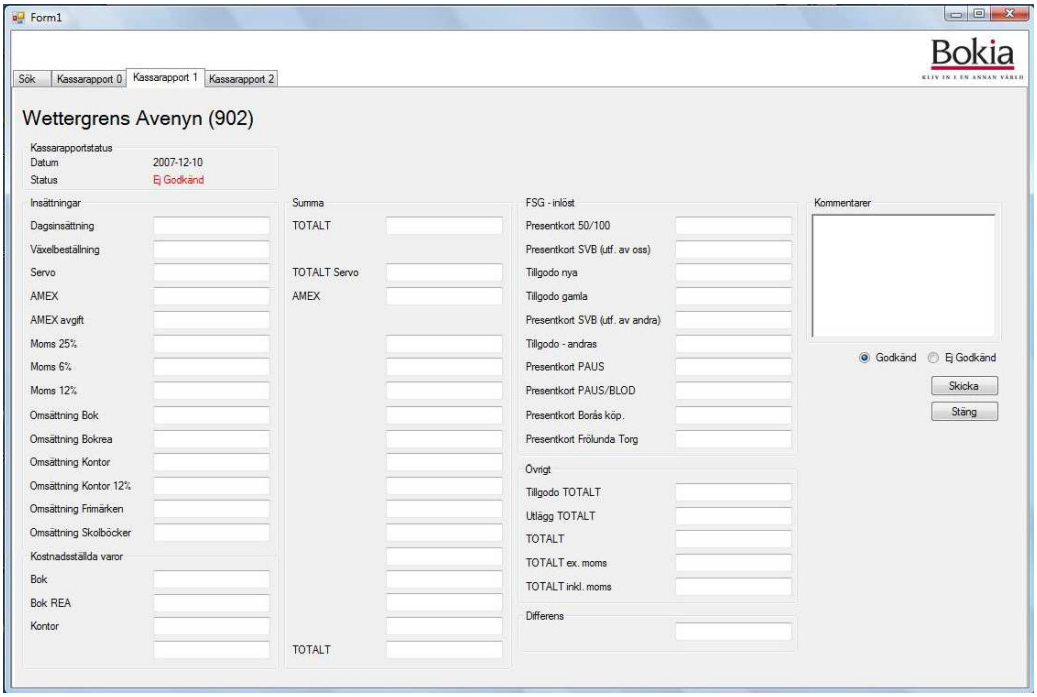
Figur 4:5 QOC-modell.

4.5 Prototyp

Till nästa möte med den primära användaren hade vi tagit fram en prototyp. Prototypen var gjord efter en horisontell modell vilket innebär att det var ett färdigt gränssnitt som saknade funktion. Under tiden användaren testade prototypen fick användaren komma med synpunkter på hur saker kunde förbättras.

Användaren hade som synpunkt att överskådligheten borde förbättras genom att olika siffror får olika färger beroende på om de är positiva eller negativa, detta för att öka överskådligheten. Dessutom ville användaren att vissa texter skulle fetmarkeras av samma anledning. Numeriska fält i applikationen som är låsta för redigering ville användaren skulle gråmarkeras för tydligare översikt. Tillsammans kom vi även överrens om att en menyrad överst i applikationen skulle förbättra navigationen i programmet.

Vi hade i prototypen tagit fram en lösning där kassarapporter visades i olika flikar som användaren kunde bläddra mellan för att kunna arbeta med flera kassarapporter samtidigt, och lösningen blev omtyckt hos användaren. Lösningen visas i figur 4:6. I kassarapportsbilden ville användaren lägga till vissa numeriska fält som saknades i nuläget.



Figur 4:6 Prototypbild över navigering med ”flikar”.

Användaren hade synpunkter på de olika statusvärden en kassarapport får när de bearbetas i applikationen. Tillsammans kom vi överrens om att en ”inkommen” kassarapport skulle få status ”Ej godkänd”, en godkänd kassarapport skulle få status ”Godkänd” och att en kassrapport som skickats iväg för export till Bokias ekonomisystem skulle få status ”Klar” och i det läget skulle kassarapporten vara låst för vidare redigering.

Figur 4:7 En kassarapports olika status.

En rapport kan ändra status från ej godkänd till godkänd och vice versa men när den väl är markerad som klar är den låst för ändring som synes i figur 4:7.

4.6 Applikationsutvärdering

I dagsläget är det endas aktuellt med en slutanvändare som och därför utfördes den slutgiltiga applikationsutvärderingen av det färdiga systemet enbart med denna person. Mötet inleddes med att starta applikationen för användaren som sedan fick testa och utforska på egen hand utan några fördefinierade användningsinstruktioner. Vid det här skedet framgick det ganska tidigt att användaren inte kunde hantera applikationen trots att denna person varit med under de tidigare designutvärderingarna. Användaren förklarade däremot att hon kände igen vissa saker från de tidigare mötena. Trots detta gissade användaren fel på vad ett antal grundläggande funktioner utförde i applikationen. Därför bestämde vi att erbjuda en kort genomgång av systemet, användaren fick dock vara kvar framför datorn som den person som kontrollerade tangentbordet och musen.

Efter 5-10 minuter började användaren att ta för sig mer. Istället för att ta emot våra instruktioner om hur programmet fungerar tog hon själv tag i händelserna och testade sig fram. Istället för att ge instruktioner fokuserade vi oss nu på att svara på frågor samt bekräfta händelser innan användaren

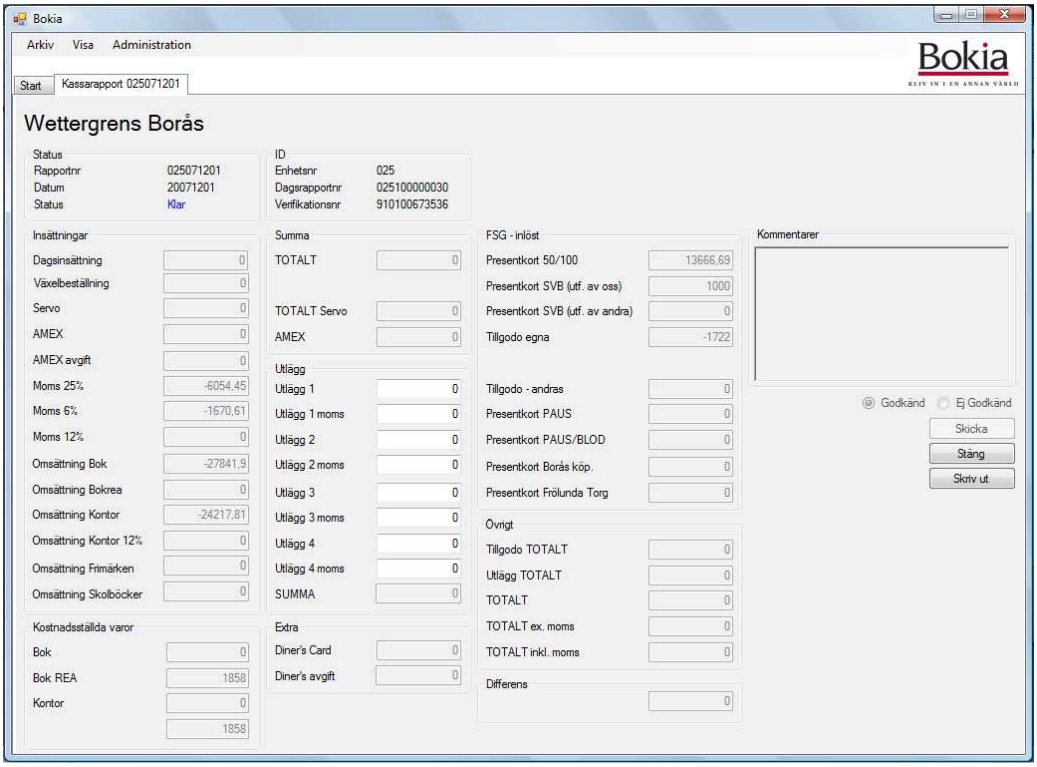
kände sig trygg nog att klickade vidare. Vid ett flertal tillfällen påpekade användaren om att hon är en novis på datorer och bad oss därmed att ha tålamod med henne.

Efter ytterligare 10 minuter började användaren komma med synpunkter istället för frågor. Hon nämnde bland annat att ett antal nya fält bör läggas till i systemet då vissa rutiner har tillkommit sedan vi utförde vår uppgiftsanalys. Hon hade också en del åsikter på estetiska korrigeringar som kunde göras i efterhand, exempelvis ändra namnet på en speciell knapp eller texten i en bekräftelseruta.

Användaren påpekade också om hur mycket effektivare hon nu skulle kunna utföra denna dagliga rutin. En annan detalj som hon gav mycket beröm över var hanteringen av kritiska operationer och felhanteringen i applikationen. Felhantering sker med hjälp av varningsrutor liknande den som visas i figur 4:8.

Figur 4:8 Varningstext.

Hon ansåg att det var mycket bra att det existerade bekräftelserutor för viktiga funktioner i applikationen samt att fält som inte ska redigeras var gråmarkerade. Denna funktion visas i figur 4:9. Slutligen uttryckte hon en stor förväntan över när applikationen skulle implementeras helt bland Bokias existerande system.



Figur 4:9 Skärmdump från den färdiga applikationen med gråmarkerade fält låsta för redigering.

5. Diskussion

Vi kommer i detta kapitel att diskutera vårt metodkapitel, teorin samt det resultat vi fått fram och försöka relatera detta till uppsatsens syfte och frågeställning. De två huvudbegreppen i detta kapitel är användarcentrerad systemdesign och användbarhet. Båda begreppen har långa och utvecklade beskrivningar men kortfattat så är användarcentrerad systemdesign en utvecklingsmodell där slutanvändaren sätts i fokus under utvecklingsprocessen. Användbarhet innebär hur väl en specifik användare kan använda ett system för att uppnå sina mål på ett effektivt och tillfredsställande sätt.

5.1 Metod och teori

Som det framgår av teoriavsnittet har systemutvecklingen inte alltid varit särskilt fokuserad eller intresserad av användaren och användbarhet. Vattenfallsmodellen som ses som den första riktiga metoden, och som kanske lite överraskande fortfarande är den mest populära modellen, erbjuder ingen vidare flexibilitet för utvecklingsmodellen utan har ett mer statiskt synsätt som bygger på att ett steg följs av ett annat. Vi tror att detta synsätt innebär att korrigeringar efter eller under utvecklingsprocessen kan bli onödigt tidskrävande och dyra att utföra. Detta kan i sin tur, inte helt ologiskt, leda till att uppdragsgivaren slopar de korrigeringar för att undgå den extra utgiften. Konsekvensen av detta kan då leda till att slutanvändaren inte anser att applikationen fyller den funktion denne önskar eller fungerar på det sättet som han skulle vilja.

Företaget som vi har utfört vår uppsats hos ville att applikation som vi utvecklade åt dem skulle ha fokus på god användbarhet. Vi ansåg därför att det skulle vara intressant att utföra ett utvecklingsprojekt med hjälp av en modell som är inriktad på just användaren och genom detta ville vi därmed ta reda på om användningen av denna modell skulle innebära att slutprodukten uppnådde kraven för användbarhet. Det visade sig dock att användbarhet är ett rätt svårt begrepp att mäta. Vi bestämde oss därför i ett tidigt skede att använda Jakob Nielsen 5 punkter för användbarhet som de grundläggande principerna för god användbarhet. Kortfattat innebär detta att systemet ska vara lätt att lära sig, effektivt att använda, lätt att komma ihåg även efter en tids frånvaro, begränsa antalet användarfel samt vara tilltalande. Dessa punkter beskrivs också mer utförligt i teorikapitlet.

Användar- och organisationskunskaper ses som mycket viktiga kunskaper för utveckling inom användarcentrerad systemdesign. Därför har det varit till god hjälpt att en av oss sedan tidigare arbetat som konsult på Bokia och därmed besitter en del förkunskaper rörande organisationen, dess anställda samt de tekniska plattformar och begränsningar som existerar. Detta har medfört att vi sparat en del tid och inte behövt införskaffa denna kunskap från början.

I vår metodbeskrivning har vi förklarat att vi inte kommer använda samtliga metoder som kan användas vid användarcentrerad systemdesign samt att vi kommer anpassa dem lite så att de är mer applicerbara för vårt projekt. Anledningen till att vi valt att göra detta är flera men främst beror det på tidsbrist samt projektets storlek och användningsområde. Den användarcentrerade systemdesignlitteraturen beskriver bland annat att man bör utnyttja flera olika sorters experter så som t.ex. databasexperter och systemarkitekter och detta har inte varit möjligt för oss. Vidare har också applikationens tilltänkta användningsområde begränsat oss något då det i dagsläget bara existerar en användare för systemet. Detta har i sin tur medfört att vi inte haft så många olika användarsynpunkter samt bakgrunder att väga in som varit önskvärt och även detta har inneburit att vi bestämt att utelämna vissa metoddelar.

5.2 Resultatdiskussion

Vid det första inledande mötet med Bokia blev vi informerade om vad för sorts system de var ute efter. Vi blev tilldelade en övergripande kravspecifikation som beskrev den tilltänkta applikationens funktion samt om vilka närliggande system som applikationen var tänkt att kommunicera med. Vi

anser att denna del inte är av allt för stort intresse för uppsatsens frågeställning utan utgör istället en teknisk grund för själva applikationen.

Den första intressanta delen i analysarbetet var uppgiftsanalysen. Genom att observera och samtidigt intervjua användaren som hanterar kassarapporterna manuellt i dagsläget fick vi i detalj veta vad hon gör dagligen. Vi spenderade ungefär två timmar genom att observera användaren. Detta gav oss först och främst detaljkunskaper om själva uppgiften men vi blev även tilldelade tekniskt underlag. Detta underlag bestod av mallar som idag används i det manuella arbetet. En av dessa mallar är det Exceldokument som vi tidigare beskrivit i resultatkapitlet. Detta dokument har agerat som en nyckelkomponent i utvecklingsprocessen då vi tidigt bestämde oss att återanvända Exceldokumentets struktur i själva applikationen så att användaren lättare skulle känna igen sig.

Användaranalysen gav oss ytterligare information om den primära användaren men även andra eventuella användare på samma avdelning. Enkätundersökningen gav oss information om på vilken nivå personalens datorkunskapsnivå var. Detta underlättade vårt gränssnittsarbete för applikationen då vi vid detta skede visste mer om vad för sorts förkunskaper användare besitter. De mer specifika datoranvändningsfrågorna gav oss mer detaljerad information om exakt vad för sorts kunskaper användarna har. En användare kan nämligen säga att han eller hon använder datorn 8 timmar per dag men om användaren enbart sitter i Word samtliga 8 timmar så borde deras kunskaper logiskt inte vara lika avancerade som en användare som hanterar flera applikationer men kanske lägger ner färre timmar per dag. Med enkätundersökningen som underlag ansåg vi att kunskapsnivån låg på en ganska låg nivå och att en enkel och tydlig applikation är att föredra framför en mer funktionsspäckad och avancerad.

Under den visuella presentationen visade vi upp skisser på det tänkta gränssnittet till applikationen och tillsammans med användaren och dennes synpunkter tog vi fram nya designförslag. Som tidigare beskrivet i diskussionen utformades skisserna så att de liknade det befintliga Exceldokument som användaren arbetar med idag. Detta för att minska inlärningskurvan samt göra det möjligt för användaren att efter en tids frånvaro fortfarande känna igen sig i gränssnittet. Vi valde också att kopiera användarens tillvägagångssätt i många av de funktioner som vi sedan implementerade. Exempelvis hur användaren godkänner kassarapporter som inkomna hanteras i dagsläget genom att användaren bockar av dem i ett speciellt Exceldokument. I våra skisser valde vi att låta programmet hålla reda på vilka kassarapporter som är inkomna och sedan låta användaren bocka av dem innan de läses in i applikationen.

QOC-modellen användes under mötet för att ta reda på hur olika uppgifter i applikationen skulle kunna lösas. Med hjälp av QOC samt den funktionslösa prototyp vi visade kunde vi få konkreta förslag på hur användaren ville ha gränssnittet. De uppgifter som skulle utföras ofta i applikationen krävde snabb och enkel hantering för att öka användbarheten. Dessa funktioner, med enkelhet och snabbhet som ”criteria”, försågs med snabbknappar och flera olika tillvägagångssätt för att utföra dem. De funktioner som ställde krav på överskådlighet fick istället andra lösningar, exempelvis i form av olika färger på text och förklarande rubriker. De uppgifter som är kritiska och utförs mer sällan i applikationen ställer krav på felhantering. Dessa uppgifter försågs med bekräftelserutor och varningstexter för att förhindra att användaren av misstag utförde dem. Genom att enbart använda felhantering på uppgifter som utförs sällan störs inte användaren av fördröjningar när denne utför mindre kritiska uppgifter.

Vi anser att applikationsutvärderingen utgjort den viktigaste informationen i uppsatsen för att besvara frågeställningen. Innan applikationsutvärderingen uppsatsen kunde vi bara diskutera och dra slutsatser över vad vi själva tror att vi lyckats. Frågeställningen lägger vikt på god användbarhet och frågan är om användarcentrerad systemdesign kan hjälpa till så att detta uppnås. Som vi tidigare angett har vi använt oss av Nielsen 5 punkter för att definiera vad god användbarhet är och med

dessa 5 punkter som grund tillsammans med applikationsutvärderingen som fakta ska vi försöka svara på frågeställningen.

- Lätt att lära.
  Kan vi bestämt säga att applikationen var lätt att använda för användaren? Utan några instruktioner eller någon hjälp klarade inte användaren av de enklaste operationerna, däremot gick det mycket bättre efter en kort vägledning. Men bör vi verkligen förvänta oss att en novis datoranvändare ska kunna hantera ett helt okänt program utan någon som helst hjälp i början? Det anser vi är lite för mycket att begära och vi tycker inte alls det var orealistiskt med en kort inlärningskurva. Tack vare att vi återanvänt vissa moment och designer som användaren tidigare använt i det manuella arbetet anser vi att programmet har gjorts enklare att använda. För att underlätta inlärningskurvan har också ett enkelt gränssnitt stått i fokus under utvecklingsarbetet och enbart de mest nödvändiga funktionerna existerar. Vi anser, med användarapplikationen som grund, att vi har uppfyllt detta krav.

- Effektivt att använda.
  Vi har haft det svårt att utvärdera denna punkt då vi bara haft en användare att testa applikationen på och denne har varit otillgänglig under långa stunder i slutet av utvecklingsfasen. Det har inte heller varit fullt möjlig att implementera applikationen fullt ut då Bokia först måste utföra vissa ändringar i sina existerande system.

- Lätt att komma ihåg.
  På grund av tidsbrist kan vi tyvärr inte svara på den här punkten. För att få ett bra underlag krävs fler tester och för att ta reda på om användaren tycker systemet är lätt att komma ihåg behöver vi en användare som under en tid använt systemet och sedan varit frånvarande från det för exempelvis en semester på några veckor. Vi kan dock dra lite paralleller till den första punkten om att systemet ska vara lätt att lära. Vi anser att applikationen uppfyller kraven om ett enkelt och lättförståligt system så det bör det inte vara allt för tidskrävande att återuppta arbetet med applikationen även efter en längre tids frånvaro.

- Få fel.
  Stor vikt lades ner på att minimera antalet fel som kan uppstå i applikationen. Bland annat genom att använda bekräftelserutor samt omöjliggöra ändringar i fält som inte ska kunna ändras. Dessa funktioner blev väldigt positivt mottaget av användaren som kände sig tryggare när vi visade detta. Vi anser därmed att även denna punkt är uppfyllt.

- Subjektivt tilltalande.
  Applikationen mottogs positivt av användaren vid utvärderingen och det fanns inte några direkta klagomål på dess användbarhet eller grafiska utseende.

5.3 Förslag för ytterligare forskning

Då denna uppsats är fokuserad på en applikation med få tilltänkta användare vore det helt klart intressant att göra en liknande studie för ett system som kan tänkas ha fler användare. Vi tror att det kan vara intressant att studera hur det kan tänkas fungera med en bred användargrupp med olika bakgrunder och åsikter om hur systemet bör se ut och fungera. Kommer detta då att resultera till att vissa användare anser systemet ha en bra användbarhet medan andra blir missnöjda eller att det kanske helt enkelt blir så att hamnar i någon sorts gråzon.

6. Slutsats

Kan god användbarhet uppnås med hjälp av användarcentrerad systemdesign?

Vår applikation, samt de analyser och utvärderingar vi utfört i den här rapporten, har varit baserad på en liten grupp användare som målgrupp och vi tror därför att detta hjälpt oss att i mycket större utsträckning kunna besvara vår frågeställning med ett ja. Med få skilda åsikter har det självklart varit lättare att utveckla ett system där användarna anser att vi uppfyller kraven på god användbarhet. Vi vill ändå belysa de analysfaser från den användarcentrerade systemdesignmodellen som vi använt i utvecklingsarbetet. Med hjälp av dessa har vi fått mycket bättre förståelse för vad användarna förväntar sig och vill ha. Om vi inte hade visat de utförliga presentationer och prototyper för användarna skulle vi med största sannolikhet fått göra ett flertal korrigeringar i efterhand. Nu fick vi istället mer konkret veta vad användarna tyckte och tänkte när de såg skärmdumpar och skisser på hur vi tänkt att applikationen ska se ut och fungera. Vi kan inte helt säkert svara på hur mycket mer tidskrävande det skulle bli att utföra korrigeringar i efterhand men med vattenfallsmodellen som bakgrund kan vi dra slutsatsen att det i de allra flesta fallen är bättre att minska antalet efterhandskorrigerar till så få som möjliga. Sådana korrigeringar som sker i efterhand löper också risk att bli en kompromiss mellan kvalité och tid/pris vilket medför att användaren hamnar i periferin och användbarheten blir lidande. Därför anser vi att användarcentrerad systemdesign är ett bra hjälpmedel för att uppnå god användbarhet.

Referenser

Aitken P. (2002). Visual Basic .Net Programming, with Peter Aitken. Paraglyph Press.

Contributor Melonfire. (2006). Understanding the pros and cons of the Waterfall Model of software development. [www]
< http://articles.techrepublic.com.com/5100-3513_11-6118423.html?part=rss&tag=feed&subj=tr#)>
(2007-12-28)

Ek J. & Overgaard J. (2002). Visual Basic.NET. Docendo Sverige AB.

Groff J. (2002) SQL: The Complete Reference, Second Edition. McGraw-Hill Professional

Guide Redina AB. (2004). Användarcentrerad Systemdesign [www]
<http://www.acsd.se> (2008-01-02)

Gulliksen J. & Göransson B. (2002). Användarcentrerad systemdesign. Studentlitteratur.

Göransson B. (2004). User-Centered Systems Design - Designing usable interactive systems in practice. Acta Universitatis Upsaliensis.

Karat J. (1996). User Centered Design: Quality or Quackery? The ACM/SIGCHI Magazine.

Nielsen J. (1993). Usability Engineering. Morgan Kaufmann Publishers.

Ottersten, I. (2002). Användbarhet i praktiken. Studentlitteratur.

Preece J. (1994). Human-Computer Interaction. Addison Wesley Publishing Company.

Shneiderman B. & Plaisant C. (2004). Designing the User Interface. Addison Wesley.

SIE-Gruppen (2008). SIE-Gruppen [www]
<http://www.sie.se> (2008-01-03)

Wilton P. (2005). Beginning SQL. John Wiley & Sons.

Bilagor

Bilaga 1 Enkätundersökning

Enkätsvar 1:

Kön:
Man.
Ålder?
35 år.
Hur länge har du jobbat på Bokia?
Sedan 1 maj 2007, i Wettergrens sedan 2003.
Vilken är din nuvarande arbetssyssla?
Controller med ansvar för redovisningen.
Hur länge har du jobbat med din nuvarande arbetssyssla?
Sedan 2003.
Vad har du för akademisk utbildning?
Ekonomie magister vid Umeå Universitet.
Hur många timmar om dagen använder du datorn på jobbet?
Åtta timmar.
Hur stor andel (i procent) är surfande på internet (i arbetssyfte och fritidssyfte)?
1%.
Hur stor andel (i procent) är arbete i övriga program och vilka är dessa isåfall?
Excel 85%.
Scala 11%.
Megadisc 2%.
Word 1%.
Vilka sysslor använder du datorn till i hemmet?
Internet.
Spela spel.
Tömma digitalkameran.
Musik.
Hur ofta använder du datorn hemma?
Mycket.
Hur stor andel (i procent) är surfande på internet?
50%.
Hur stor andel (i procent) är arbete i övriga program och vilka är dessa isåfall?
50% dataspel.

Enkätsvar 2:

Kön:

Kvinna.

Ålder?

46 år.

Hur länge har du jobbat på Bokia?

7 månader.

Vilken är din nuvarande arbetssyssla?

Sköter leverantörreskontra för Wettergrens Bokhandel.

Hur länge har du jobbat med din nuvarande arbetssyssla?

Sedan 1987 har jag arbetat blandat med kundreskontra och leverantörreskontra.

Vad har du för akademisk utbildning?

2-årig ekonomisk.

Hur många timmar om dagen använder du datorn på jobbet?

C:a 6 timmar.

Hur stor andel (i procent) är surfande på internet (i arbetssyfte och fritidssyfte)?

5 procent c:a i arbetssyfte.

Hur stor andel (i procent) är arbete i övriga program och vilka är dessa isåfall?

Scala 95 procent.

Word 5 procent.

Vilka sysslor använder du datorn till i hemmet?

Till bankärenden.

Enkätsvar 3:

Kön:

Kvinna.

Ålder?

46 år.

Hur länge har du jobbat på Bokia?

Sedan maj -07 (Wettergrens sedan augusti -00).

Vilken är din nuvarande arbetssyssla?

Kundreskontran (fakturering, upplägg nya kunder, kundbetalningar m.m.) gällande Wettergrens Bokhandel.

Hur länge har du jobbat med din nuvarande arbetssyssla?

Sedan augusti -00.

Vad har du för akademisk utbildning?

3-årig gymnasieutbildning.

Hur många timmar om dagen använder du datorn på jobbet?

Från det jag kommer kl 8.00 till dess jag går hem kl 17:00.

Hur stor andel (i procent) är surfande på internet (i arbetssyfte och fritidssyfte)?

C:a 10 %

Hur stor andel (i procent) är arbete i övriga program och vilka är dessa isåfall?

90% Megadisc + Scala + Boss.

Vilka sysslor använder du datorn till i hemmet?

E-mail + betala räkningar.

Hur ofta använder du datorn hemma?

3-4 dagar i veckan.

Hur stor andel (i procent) är surfande på internet?

20-30 %

Hur stor andel (i procent) är arbete i övriga program och vilka är dessa isåfall?

Inget svar.

Enkätsvar 4:

Kön:

Kvinna.

Ålder?

41 år.

Hur länge har du jobbat på Bokia?

6 månader (17 år på Wettergrens).

Vilken är din nuvarande arbetssyssla?

Kassarapporter, det mesta som har med redovisning att göra.

Hur länge har du jobbat med din nuvarande arbetssyssla?

4 år.

Vad har du för akademisk utbildning?

Ingen, är självlärd.

Hur många timmar om dagen använder du datorn på jobbet?

C:a 8 timmar.

Hur stor andel (i procent) är surfande på internet (i arbetssyfte och fritidssyfte)?

5% i arbetssyfte.

Hur stor andel (i procent) är arbete i övriga program och vilka är dessa isåfall?

95%.

Vilka sysslor använder du datorn till i hemmet?

Maila, lite internethandel.

Hur ofta använder du datorn hemma?

Väldigt sällan, knappt en gång i veckan.

Hur stor andel (i procent) är surfande på internet?

100%.

Hur stor andel (i procent) är arbete i övriga program och vilka är dessa isåfall?

0%.

Bilaga 2 Intervju i samband med uppgiftsanalys

1. Varför utförs uppgiften?
Uppgiften utförs för att kontrollera kassarapporter. Kassarapporterna kommer in dagligen från varje butik och måste kontrolleras då de till 99% innehåller fel som måste korrigeras. Utifrån kassarapporterna görs avstämningar.

2. Hur ofta utförs uppgiften?
Uppgiften utförs varje dag.

3. Hur lång tid tar uppgiften att utföra?
I snitt tar en kassarapport ungefär 4-5 minuter att kontrollera, justera och godkänna exklusive den manuella inmatningen av transaktionerna i ekonomisystemet. Men om det är något allvarligt fel med kassarapporten kan det ta mycket längre tid att slutföra den. Som längst har det tagit 2½ timme att göra klart en enda kassarapport.

4. Samarbetar du med någon annan när uppgiften utförs?
Hela uppgiften utförs själv.

5. Vad har du för några hjälpmedel när uppgiften utförs?
Exceldokument där siffror från kassarapporten matas in, ”krokodil” för att samla ihop kassarapporter och pärmar för att sortera in dem när de är klara.

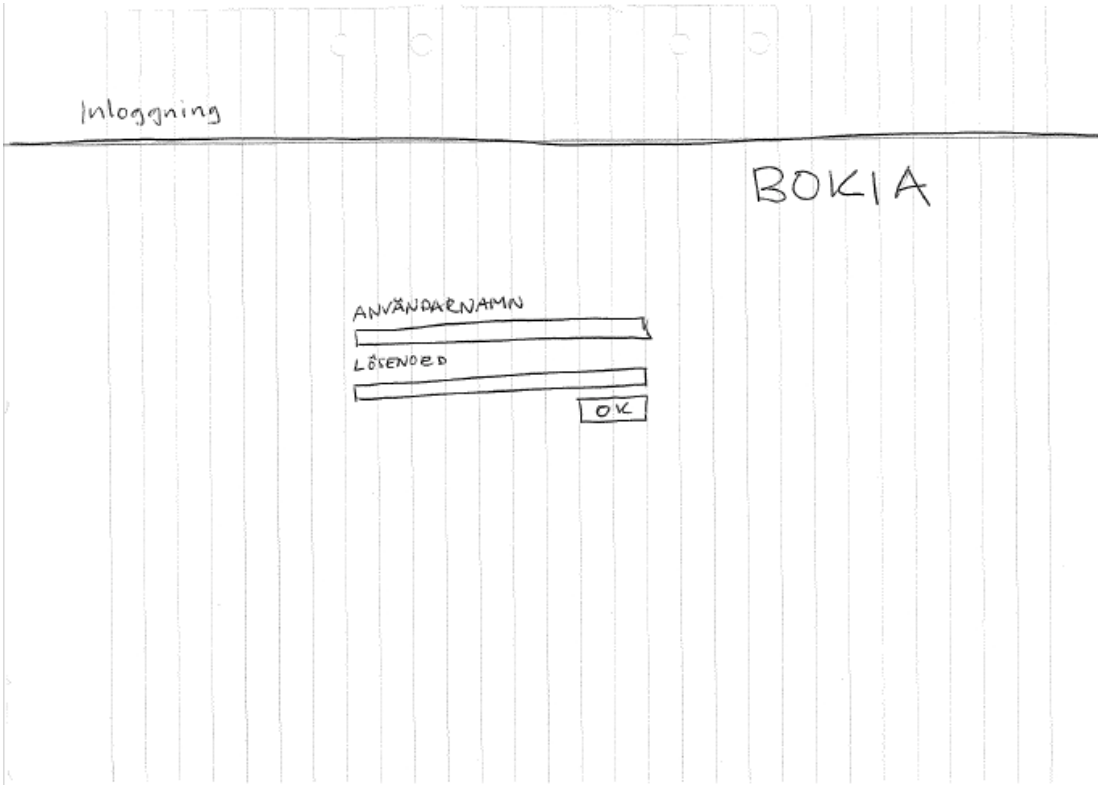
6. Vad finns det för några kritiska moment och ”flaskhalsar” med uppgiften?
Ifall kassarapporten inte kommer mig tillhanda så går det inte att slutföra den då den krävs för att kunna slutföra uppgiften. Samma sak gäller ifall kassarapporten är ofullständig och saknar nödvändiga uppgifter.

7. Hur kan situationen och informationsflödet förbättras?
Genom att det manuella arbetet som går att utföra automatiskt försvinner. I dagsläget måste jag stansa in information två gånger. Dessutom kan vissa kontroller som jag gör manuellt idag utföras per automatik.

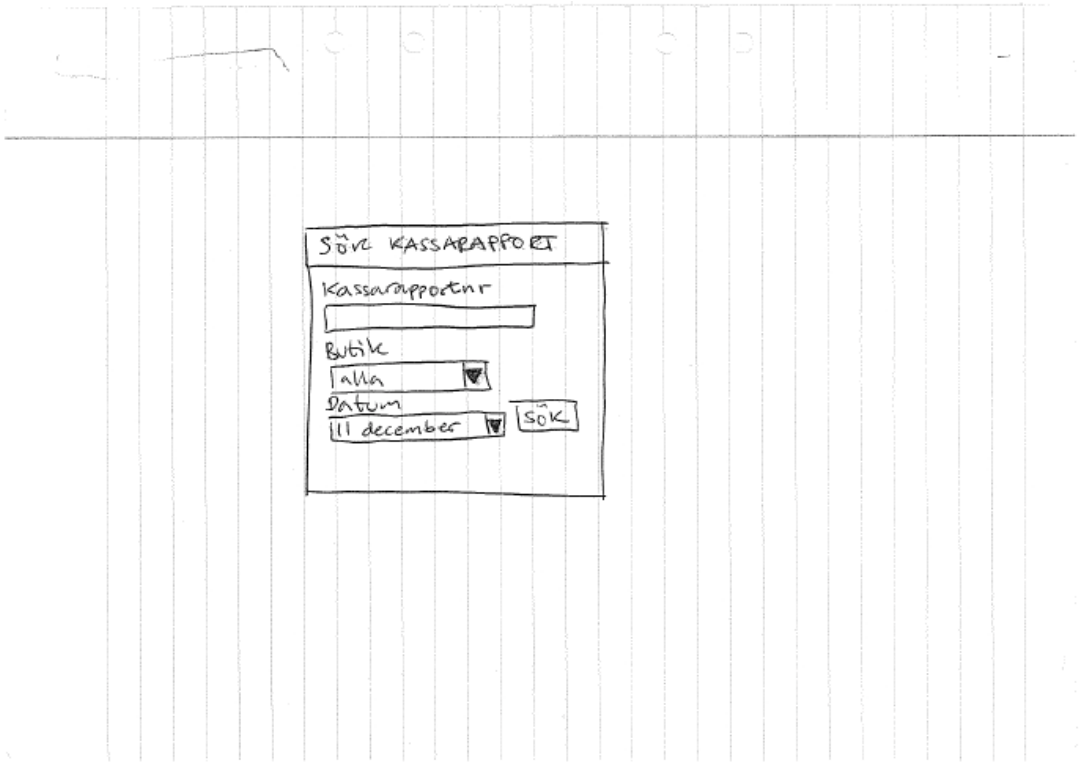
8. Blir det ibland fel när du utför uppgiften?
Det finns alltid en risk att jag matar in fel siffror i Exceldokumenten när jag manuellt läser av kassarapporterna.

9. Vad händer ifall det blir fel när uppgiften utförs?
Vi har en controller på Bokia som kontrollerar de uppgifter som jag bokför i vårt ekonomisystem. Ifall jag antingen matar in fel siffror eller fel konton så läggs fel belopp på fel ställen i bokföringen vilket i sin tur ger en felaktig bokföring.

Bilaga 3 Skisser

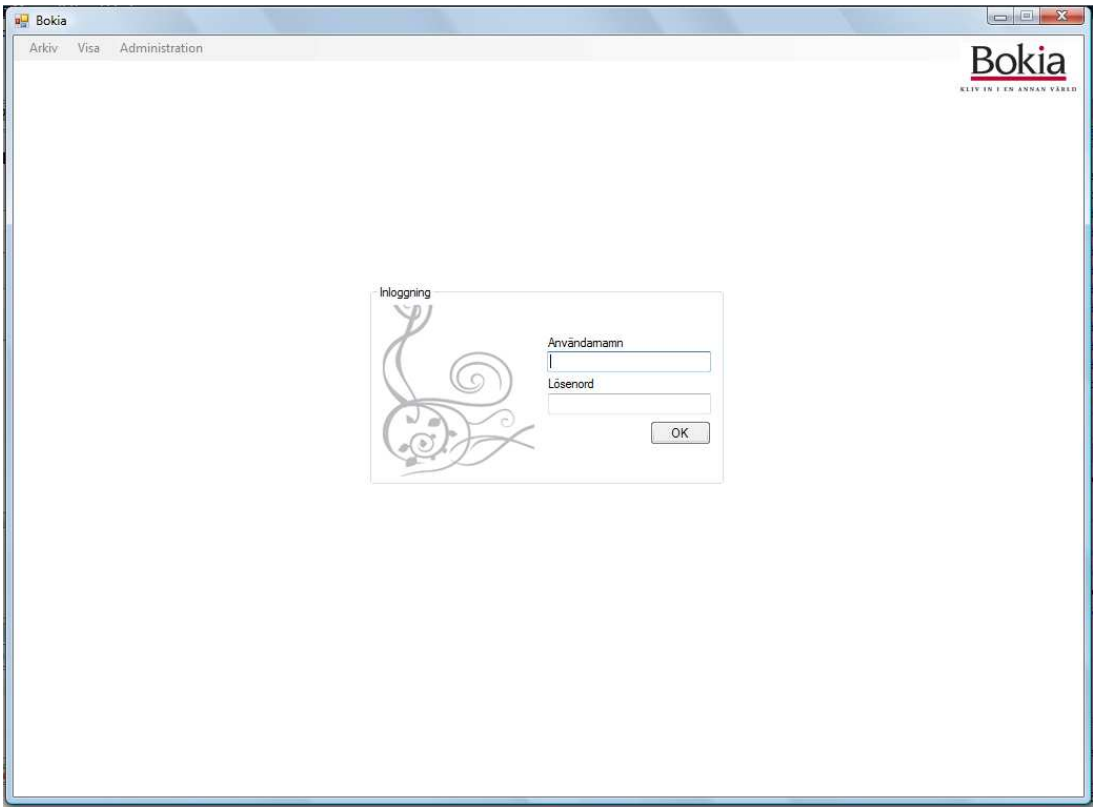


Skiss över inloggningsruta

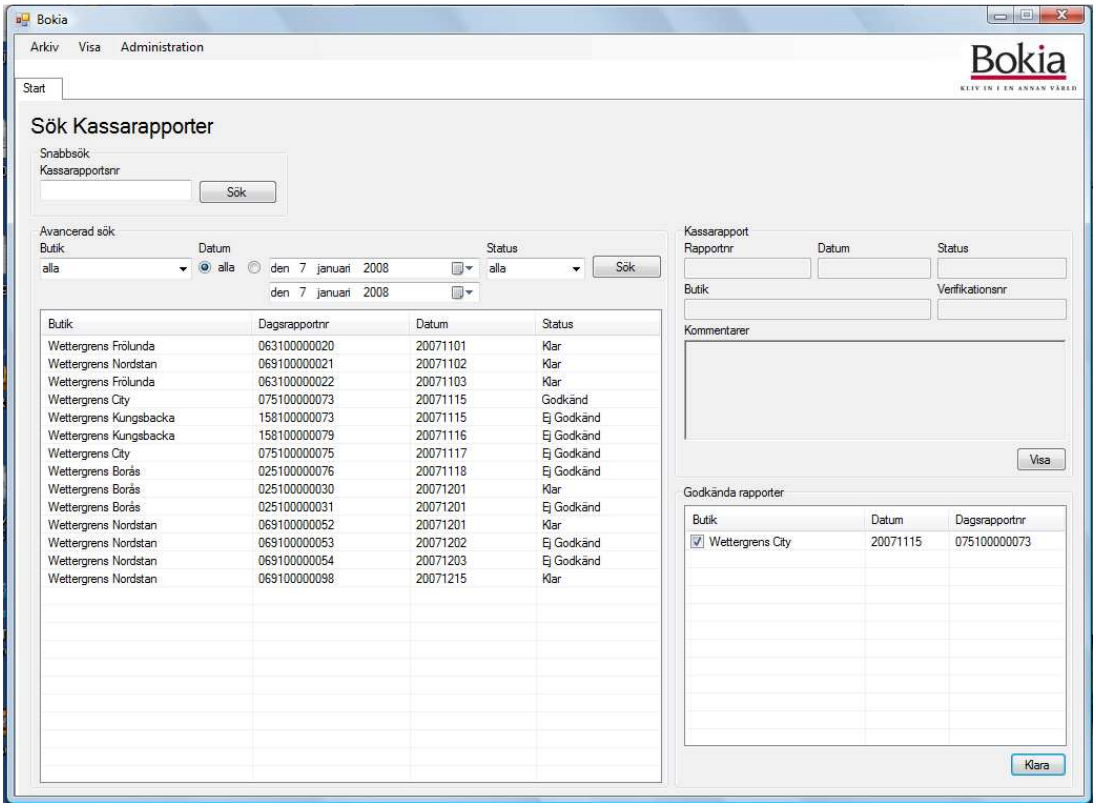


Skiss över sökning av kassarapport

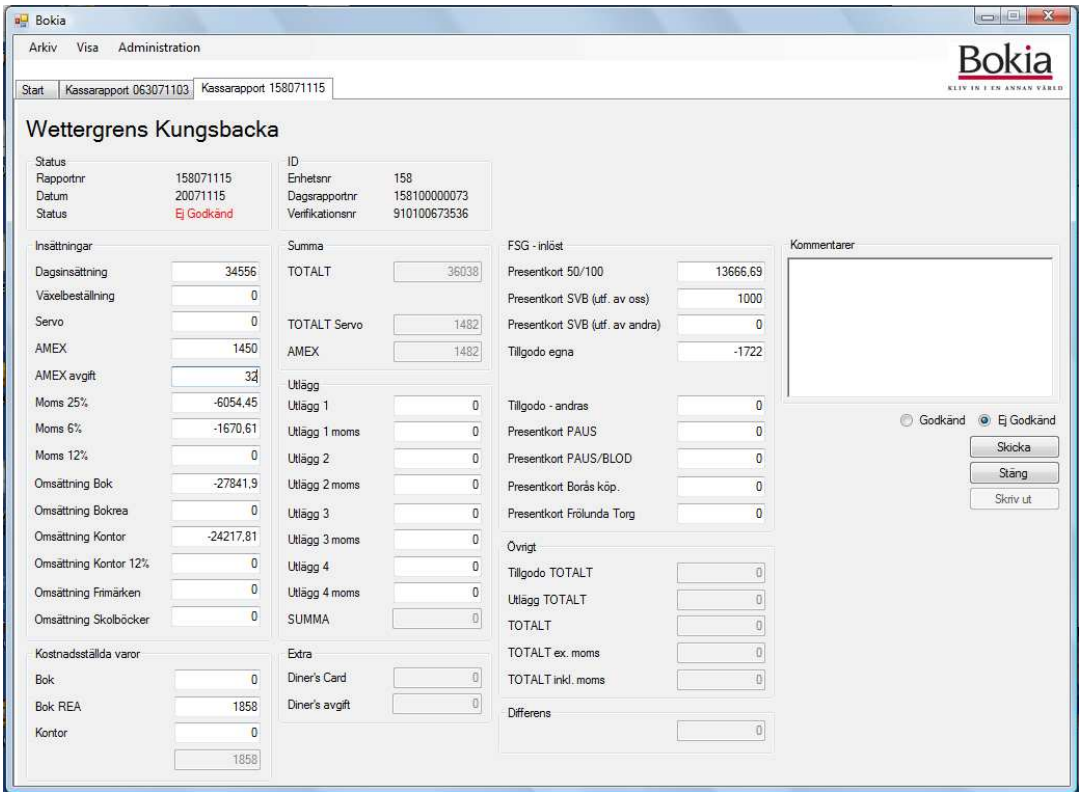
Bilaga 4 Skärmdumpar från den färdiga applikationen



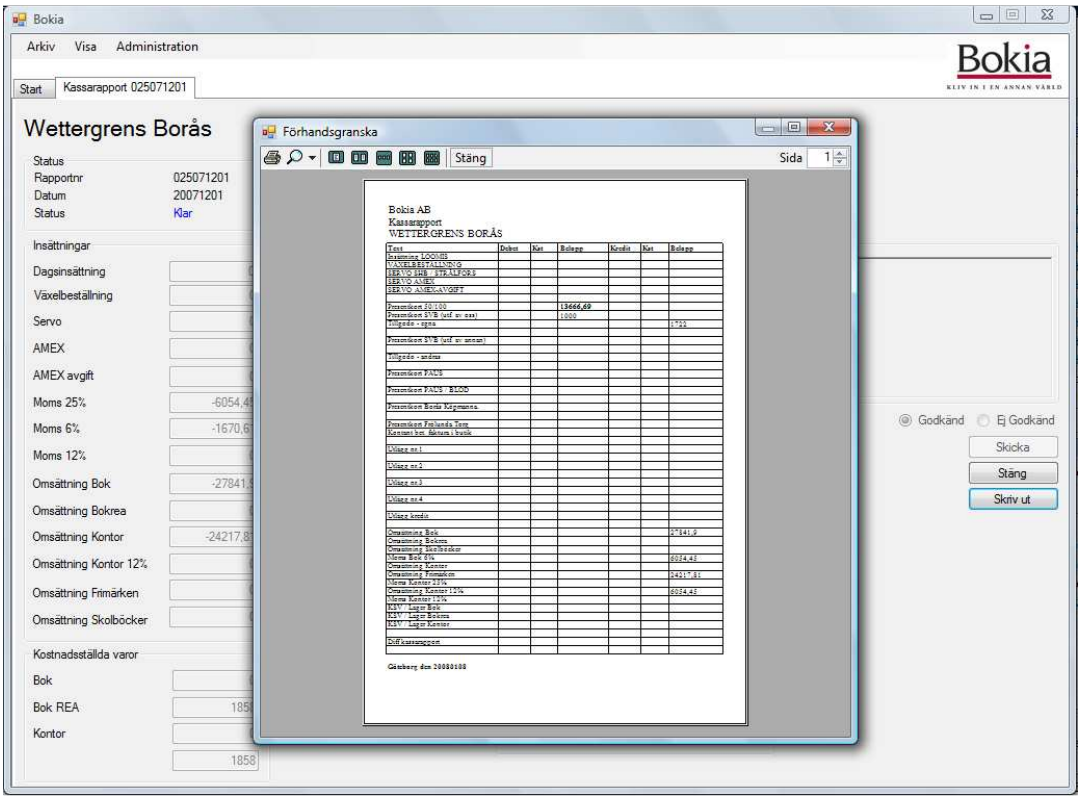
Inloggningsruta



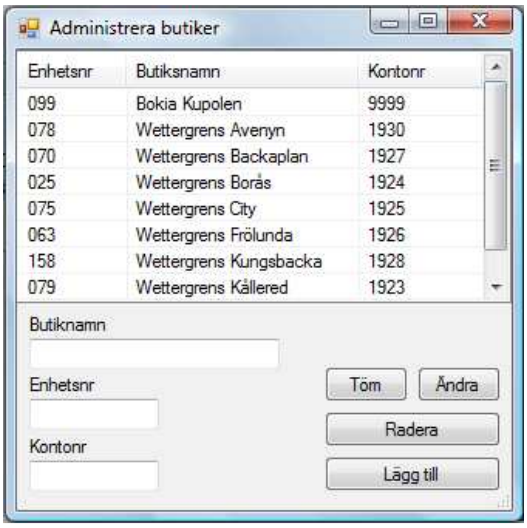
Startsida med sökning och navigering med ”flikar”



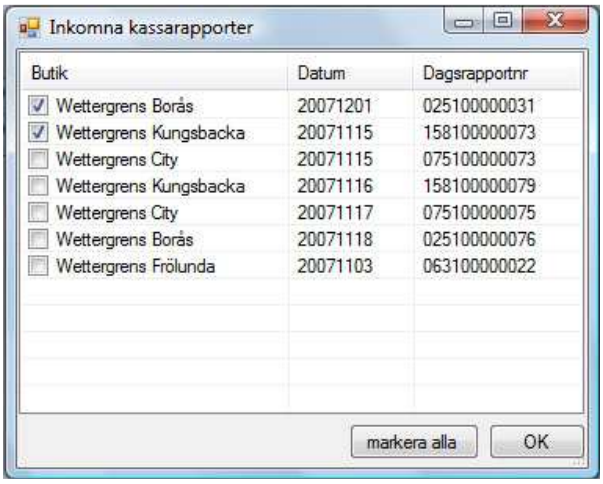
Kassarapportsbild för redigering och godkännande



Utskrift av klar kassarapport



Administrering av butiker



Godkännande av inkomna kassarapporter