UNIVERSITETET I OSLO
Institutt for informatikk

Interaksjonsdesign og diabetes: Den digitale loggboken

Masteroppgave
(60 studiepoeng)

Harald Haugerud

02.05 2006

Forord:

Dette er en masteroppgave (60 studiepoeng) skrevet i forbindelse med en mastergrad ved Institutt for Informatikk. Dette er også en avsluttende oppgave for proffesjenstudiet i informatikk. Arbeidet med oppgaven har pågått fra høsten 2005 til våren 2006.

Jeg vil takke min veileder, Anders Kluge, for meget bra veiledning og god støtte gjennom prosessen. Videre vil jeg takke Erik Richvoldsen, min medstudent, for gode dialoger i prosessen. Anna Guttormsgaard Engh trenger også en takk for hjelp med korrekturlesing og støtte i en krevende periode.

Harald Haugerud

Institutt for Informatikk, Universitetet i Oslo
Mai 2006

Innhold:

Figurer:

1 Introduksjon

1.1 ”Dingser” i hverdagen

Hver dag gjør vi oss avhengig av forskjellige ”dingser”, eller elektroniske håndholdte eksterne enheter, som det også kan kalles. Hovedformålet med disse dingsene kan være å gjøre hverdagen hyggeligere, bedre eller lettere for oss. Enten det er å snakke i telefon, notere avtaler eller høre på musikk så opplever vi disse elektroniske enhetene spennende, underholdene eller hjelpende.

Mobiltelefonen kan anses som en slik håndholdt enhet som har blitt allemannseie. Post- og teletilsynet (2005) sin rapport sier at det er litt flere mobilabonnementer enn det er innbyggere i Norge. Eksempler på andre håndholdte enheter er bærbare musikkspillere, digitale kameraer, GPSer, PDAer, bærbare spillmaskiner og andre elektroniske enheter som kan sees på som nyttige eller underholdende.

Siden en slik enhet skal være bærbar og fungere i brukerens dagligliv, bør enheten være så liten som mulig. Dette kan da få konsekvenser for funksjonaliteten til slike enheter. Som tidligere nevnt finnes det en rekke håndholdte musikkspillere, eller mp3-spillere som de også kan kalles. Mp3-spilleren har mye innbygget funksjonalitet. En kan for eksempel spille av en sang flere ganger på rad, spille alle sangene i vilkårlig rekkefølge eller for eksempel justere volumet opp eller ned. Brukeren kan da lytte til musikk ved hjelp av denne bærbare enheten. Det er allikevel en funksjonalitet brukeren ikke får tilgang til via brukergrensesnittet til den bærbare enheten. Det er å overføre nye sanger inn på enheten. Når en kjøper en mp3-spiller i dag ligger det ingen sanger inne på spilleren. Brukeren må overføre disse sangene selv. Problemet er at brukergrensesnittet på selve mp3-spilleren ikke gir brukeren mulighet til å overføre sanger. Dette må gjøres via en datamaskin gjennom for eksempel en egen kabel som følger med mp3-spilleren. Ved hjelp av programvare på datamaskinen og denne kabelen kan brukeren få overført sanger til mp3-spilleren. Datamaskinen og programvaren gir da mp3-spilleren en utvidet funksjonalitet. Det er for eksempel ikke mulig å legge inn sanger på en mp3-spiller uten å bruke en datamaskin. Når en kjøper musikk i butikken, får en dette i form av en cd eller et annet fysisk lagringsformat. Det er da ikke mulig å overføre denne musikken på cd-platen over på mp3-spillerne, siden spilleren mangler mulighet til å lese slike formater. Denne funksjonaliteten i brukergrensesnittet til mp3-spilleren er fjernet for å kunne lage spilleren så fysisk liten som mulig. Ønsker brukeren å overføre en musikk-cd til mp3-spilleren, må brukeren ta i bruk en datamaskin med cd-spiller. Datamaskinen gir mp3-spilleren utvidet funksjonalitet.

Det finnes flere eksempler på at datamaskinen gir brukeren utvidet funksjonalitet i forholdt til den elektroniske håndholdte enheten. Det er for eksempel færre muligheter til å redigere bildene på et digitalt fotoapparat enn de mulighetene en får ved å overføre bildene til en datamaskin. Et digitalt fotoapparat gir brukeren en del funksjonalitet, men en datamaskin med riktig programvare gir vesentlig flere. En kan si at programvaren på en datamaskin kan supplere og gi håndholdte enheter utvidet funksjonalitet.

1.2 En diabetikers brukssituasjon

De fleste eksterne enhetene i dag er laget for bruksituasjoner innenfor massemarkedet. Som tidligere nevnt er dette mobiltelefoner, PDAer, GPSer, musikkspillere og andre enheter som skal underholde oss eller gi oss mer eller mindre nyttige funksjoner. Ser en på andre brukssituasjoner kan en finne eksempler på at utviklingen ikke har kommet like langt med tanke på funksjonalitet og brukbarhet. Bruksituasjonen for en diabetiker er en slik situasjon.

En diabetiker har alltid med seg en slik ekstern enhet. Denne enheten kalles et blodsukkerapparat og har som hovedfunksjon å måle blodsukkeret. Det å kunne måle blodsukkeret er et viktig element av behandlingen for diabetikere. Et blodsukkerapparat har som regel noen innebygde funksjoner. De fleste apparatene kan i tillegg til å måle blodsukkeret huske tidligere målte blodsukkerverdier og regne ut noen gjennomsnittsverdier av tidligere målt blodsukkerverdier.

For en diabetiker er et slikt apparat et godt hjelpemiddel. En nyttig enhet. Allikevel vil en fort ønske seg utvidet funksjonalitet, spesielt med tanke på analyse av egne data. I dag finnes det noen apparater som har mulighet til å overføre data fra et blodsukkerapparat til en datamaskin. Tanken rundt dette er at brukeren kan laste sine målinger over på datamaskinen for å så kunne bruke et program for å analysere dataene. En analyse av egne data er nødvendig for mange diabetikere, siden det er en god måte å få kontroll over sin diabetes på. Datamaskinen kan her brukes til å gi brukeren en utvidet funksjonalitet som en ikke får på den eksterne enheten. En annen viktig fordel er at en får mulighet til å kombinere data fra et blodsukkerapparat med andre verdier, som er interessante å ha med i en analyse. Dette kan være verdier rundt matinntak, trening, alkohol, insulindoser eller andre verdier diabetikeren har behov for å loggføre.

For å gi blodsukkerapparatet en utvidet funksjonalitet er en avhengig av et program på datamaskinen. I dag finnes det noen slike programmer. Disse programmene lar brukeren overføre dataene til datamaskinen. Videre kan dataene fra blodsukkerapparatet kombineres med mange andre data for å få gjennomført en analyse. Dessverre er mange av disse programmene ikke veldig brukervennlige. De er ofte uoversiktlige og vanskelige å bruke. En bruker kan fort føle at en bruker mer tid på slike programmer enn en får igjen av utbytte. Dette medfører at denne løsningen blir for spesielt interesserte eller det vi kan kalle for ekspertbrukere, brukere som er datakyndige. Programmet blir da for tungvint i bruk slik det fremstilles i dag.

Innenfor diabetesdomenet vil en finne brukere som ikke er ekspertbrukere. Behovet for et slikt program kan være det samme for alle diabetikere uansett datakyndighet. De er avhengig av et godt hjelpeverktøy for å kunne ha en tilfredstillende kontroll over sin diabetes. Brukergrensesnittet til dagens hjelpemidler kan derfor ikke sees på som en fullgod løsning for en diabetiker.

1.3 Problemstilling

Brukergrensesnittet til et program er med på å synliggjøre funksjonaliteten som er tilgjenglig i programmet. Funksjonaliteten kan visualiseres på forskjellige måter. Norman (1998) bruker begrepet mapping om hvordan funksjonaliteten i

brukergrensesnittet er en relasjon til hvordan funksjonaliteten er visualisert. Funksjonaliteten i et program må da visualiseres gjennom et brukergrensesnitt for at brukeren skal få tilgang til funksjonaliteten. En kan si at visualiseringen må mappes med riktig funksjonalitet, slik at brukeren oppnår ønsket effekt eller respons fra systemet. En god mapping er når bruken av kontrollenheten gir det resultatet brukeren forventet. Det er da en avgjørende faktor hvordan kontrollobjektet blir visualisert, for hvilket resultat brukeren forventer å få tilbake (Norman 1998).

En diabetiker har en rekke daglige rutiner og er avhengig av hjelpemidler med forskjellig funksjonalitet. Denne funksjonaliteten kan visualiseres på forskjellige måter. En kommer da inn på selve hovedproblemstillingen i denne oppgaven som kan beskrives med to spørsmål:

- *Hvordan kan funksjonaliteten, som brukeren kan ha behov for, visualiseres og mappes i brukergrensesnittet, med tanke på bruksituasjonen til en diabetiker?*

- *Finnes det liknende bruksituasjoner og systemer en kan hente erfaring fra?*

Funksjonalitet kan sees på som en oppgave et program eller et hjelpemiddel kan utføre for diabetikeren. Funksjonaliteten brukeren kan ha bruk for, hentes i denne oppgaven fra en generell diabetikers daglige rutiner. Det vil da være naturlig å definere et sett med funksjonaliteter som en diabetiker kan ha bruk for. Videre må en se på hvordan disse kan best mulig visualiseres og mappes i brukergrensesnittet.

Det er ikke min oppgave å finne ut hvilke funksjoner som gir best medisinsk behandling eller hvilke funksjoner som vil være de beste for en diabetiker. Jeg vil se på hvordan et sett med funksjoner brukeren kan ha behov for kan representeres gjennom visualisering og hvordan det visualiserte objektet kan mappes, slik at brukeren oppnår å gjennomføre den handlingen brukeren ønsket.

Det er her allikevel sannsynlig at en forbedring i hjelpemiddelenes brukergrensesnitt vil føre til mer bruk av et slik program. En diabetiker må sette av en viss tid til behandling og oppfølging av sin egen diabetes i hverdagen. Ved bruk av en datamaskin kan diabetikeren få nye muligheter. De nye mulighetene kan gi bedre analyser og en mer effektiv behandling av dataene som en diabetiker er interessert i. Brukergrensesnittet gir diabetikeren tilgang til denne funksjonaliteten i datamaskinen. Hvis brukergrensesnittet er brukbart vil brukeren få lettere tilgang til funksjonaliteten enn om brukergrensesnittet var mindre brukbart. En god tilgang til de nye mulighetene en datamaskin vil gi diabetikeren, vil da kunne føre til mer bruk av programmet.

Et godt brukergrensesnitt i et program vil da kunne generere mer bruk siden funksjonaliteten til programmet blir mer og bedre synlig. Dette vil også øke sannsynligheten og muligheten til en bedre kontroll over behandlingen av sin diabetes. En bedre kontroll over sin diabetes vil føre til bedre livskvalitet, lenger levealder og minske sannsynligheten for andre komplikasjoner forbundet med diagnosen.

2 Metode

2.1 Empiriske studier og ekspertanalyser

I denne oppgaven vil jeg studere programmer som allerede eksisterer. Jeg vil se på loggføringsprogrammer som eksisterer for diabetikere og jeg vil se på andre programmer som har en liknende brukssituasjon. Disse programmene vil brukes som et grunnlag for å generere skisser som illustrer deler av et mulig brukergrensesnitt for et program som kan brukes av en diabetiker. Skissene vil være et resultat av en overføring av designløsninger fra programmer som eksisterer i andre bruksituasjoner til et mulig program for diabetikere.

2.1.1 Gjennomgang av eksisterende programmer

Jeg vil i denne skissen beskrive programmer som eksisterer i dag og hvordan disse programmene har visualisert sin funksjonalitet. Metoden jeg har brukt for å gå igjennom disse programmene kan kalles en ekspertanalyse. I en ekspertanalyse, brukes en person som kan anses som en ekspert innenfor det fagfeltet eller domenet som skal analyseres (Shneiderman 1998). Det som kjennetegner en ekspertanalyse er at eksperten får redegjort hvilke kriterier som legges til grunn for analysen. Videre får eksperten presentert skissen, prototypen eller produktet analysen skal utføres på. Resultatet av en slik analyse er ofte en rapport eller en tekstlig beskrivelse av hva eksperten fant i forhold til kriteriene av analysen. Det finnes flere forskjellige ekspertanalyser.

Ved gjennomgangen av de eksisterende programmene har jeg brukt en teknikk som kan minne om heuristisk evaluering, som er en ekspertanalyse. Heuristisk evaluering er en metode som er beskrevet av Nielsen (1994). I en heuristisk evaluering setter en opp en liste med kriterier. Disse kriteriene er da retningslinjer som eksperter bruker når en skal gjennomgå skisser, prototyper eller ferdige systemer. Jeg vil i kapittel 5.1 forme kriterier for gjennomgangen av programmene. Disse kriteriene har da som formål å finne ut hvordan de eksisterende programmene har visualisert forskjellige funksjonalitet.

2.1.2 Generering og Analyse av skissene

I analysen av de genererte skissene vil jeg også bruke en ekspertanalyse som kalles en retningslinjeanalyse (Shneiderman 1998). Den kjennetegnes ved at et brukergrensesnitt kontrolleres for samsvar etter en retningslinje som er gitt (Shneiderman 1998). Analysen må da gjennomføres av en ekspert som kjenner retningslinjene godt. I denne oppgaven vil skissene som vil bli generert kontrolleres for overensstemmelse med det teoretiske rammeverket i oppgaven. Gjennom mitt litteraturstudie i denne oppgaven, kan jeg selv anses som en ekspert innenfor de respektive teoriene. Det teoretiske rammeverket kan da anses som retningslinjene til selve retningslinjeanalysen. Resultatet av analysen vil bli en diskusjon over hvordan skissene samsvarer med retningslinjene og det vil da bli redegjort for hvilke deler av brukergrensesnittet som samsvarer med det teoretiske rammeverket.

I kapittel 3.3.2 beskrives tre forskjellige krav til brukbarhet til skissene. Kravene er: lærbarhet, fleksibilitet og robusthet. Disse kravene viser igjen til hvordan teorier og begreper i interaksjonsdesign kan være med på å tilnærme seg disse kravene. Videre blir det beskrevet i kapittel 3.3.1 behovet til funksjonalitet i et slikt program. Skissene som blir generert vil da vise hvordan denne funksjonaliteten kan visualiseres. Retningslinjeanalysen vil se på hvordan skissene er knyttet opp til teorier i interaksjonsdesign.

Resultatet av denne analysen vil da kun gi et svar på om hvordan skissene *kan* gi bedre brukbarhet, ikke om de *vil* gi bedre brukbarhet. For å kontrollere om de vil gi bedre brukbarhet, må skissene utvikles videre til et produkt som kan testes. Skissene må så videre testes på reelle brukere. Dette vil kunne gi svar på om skissene vil gi bedre brukbarhet.

Skissene må da sees på som en del av en større utviklingsprosess av et slikt program for diabetikere. Denne oppgaven vil bidra med forslag til løsninger til visualisering av funksjonaliteten. Skissene må da sees i sammenheng med en større utviklingsprosess, hvor utviklingen av forslag til mulige visualiseringer av funksjonalitet er et ledd i denne prosessen. Analysen av skissene vil da bidra til denne prosessen med løsninger på hvordan funksjonaliteten kan visualiseres. En teoretisk analyse av skissene vil beskrive hvordan skissene kan bidra til å øke brukbarheten av brukergrensesnittet i et slikt program.

2.1.3 Bruken av skisser i oppgaven

I denne oppgaven vil jeg ta i bruk skisser som vil videre bli analysert. Holmquist (2005) skiller mellom bruken av prototyper og skisser. Skisser er objekter som har utseende men ikke funksjonaliteten til et produkt (Holmquist 2005). Skisser baserer seg da på å visualisere hvordan produktet kan eller kommer til å se ut og ikke nødvendigvis hvordan funksjonene til produktet skal implementeres. I motsetning til dette beskriver Holmquist (2005) en prototype til å ha prinsippene om funksjonaliteten til et endelig produkt, men ikke nødvendigvis utseende. Forskjellen er da at skisser ikke trenger å ha en plan om hvordan funksjonaliteten til det endelige produktet skal fungere. En kan foreksempel ha en skisse som viser en ”evighetsmaskin”. Det vil si en maskin som går til evig tid uten å få tilført ny energi. Skissen kan da inneholde en rekke tegninger for hvordan denne maskinen skal se ut. Problemet med denne skissen er at det er bevist at det ikke er mulig å lage en evighetsmaskin. Allikevel er det mulig å lage en skisse av en slik maskin. Denne skissen fremstiller da et produkt som ikke kan realiseres i virkeligheten. Holmquist beskriver dette ved at det er allikevel fult mulig å lage en skisse over et produkt som løser et problem som det er bevist at ikke kan løses, siden skissen ikke beskriver hvordan de funksjonelle problemene skal løses.

En skisse beskriver ikke om den er realiserbar i virkeligheten. Allikevel kan en skisse være en del av en prosess som bidrar til å finne løsningen på et problem. En skisse kan for eksempel gi nye synsvinkler til en diskusjon. Holmquist (2005) beskriver slike skisser som generatorer og slike generatorer kan gi opphav til diskusjon. Skissen brukes da som en del av en større prosess og skissene kan da ikke anses som et

sluttprodukt, siden den ikke beskriver hvordan produktet implementerer funksjonaliteten.

Når en brukere en skisse er det da viktig å beskrive i tillegg skissens funksjonelle problematikk. En må gjøre det tydelig om en vet om skissen er faktisk realiserbar eller ikke. Eventuelt må en redegjøre for hvilke utfordringer en må løse for at skissen kan bli realiserbar.

I denne oppgaven vil det bli brukt en del skisser. Disse skissene vil bli laget for å vise hvordan funksjonalitet kan bli visualisert. Skissene er kun ment som det Holmquist (2005) beskriver som generatorer og vil være et innspill til en diskusjon om hvordan funksjonaliteten kan visualiseres. Det bli altså ikke beskrevet om det er mulig eller hvordan funksjonaliteten til programmene kan implementeres. Skissene beskriver også visualiseringsløsninger og det blir ikke beskrevet om eller hvordan disse visualiseringsløsningene kan bli implementert ved hjelp av dagens visualiseringsteknikker.

Skissene er da ment som generatorer og er ment som et innspill til diskusjon. Det er ikke avklart om skissene er realiserbare eller ikke.

2.2 Litteraturstudier

Denne oppgaven er basert på et litteraturstudie av relevante teorier i interaksjonsdesign. Interaksjonsdesign har også en relasjon med teoretiske begreper en finner menneske maskin interaksjonteorien. I litteraturstudiet har jeg fokusert på å sette meg inn i teorier som brukes for å visualisere funksjonalitet i brukergrensesnittet. Teoriene er valgt ut for å kunne bedre brukbarheten av visualiseringen. Jeg har valgt å se på teorier og begreper som eksisterer i dagens brukergrensesnitt. Videre har jeg sett på begreper som omhandler menneskets interaksjon med datamaskinen. Dette er teorier som omhandler selve interaksjonsprosessen og konkrete rammeverk for å generalisere interaksjonen i brukergrensesnittet. Jeg har også sett på hvordan funksjonalitet kan bli visualisert i brukergrensesnittet gjennom bruken av WIMP (vinduer, ikoner, menyer, pekere) og metaforer. Det utvalgte teoretiskerammeverket vil videre bli brukt i analyser av skisser som denne oppgaven genererer ut i fra empirien.

3 Diabetes og bruksituasjonen

I dette kapittelet vil jeg beskrive litt om hva diabetes er. Videre vil jeg beskrive hvordan en diabetikers bruksituasjon kan se ut i hverdagen og hvordan diabetikeren forholder seg til hjelpemidler. Jeg vil også beskrive hvilken generell funksjonalitet et program for diabetikere kan ha. Videre vil jeg beskrive hva som er viktig for brukbarheten til brukergrensesnittet til et slikt program.

3.1 Hva er diabetes?

Diabetes mellitus, ofte kun kalt diabetes, er en stoffskiftesykdom. Sykdommen strekker seg tilbake til år 1550 f. kr., da diabetes blir omtalt på papyrusruller funnet i egyptiske graver (Christophersen og Hanssen 2004). Selvet ordet diabetes mellitus betyr noe som ”renne igjennom” og ”søtt” (Hanås 2002). Dette stammer fra at sykdommen oppdages ved at en har veldig søt urin og hyppig vannlating.

Det finnes pr. i dag to typer diabetes, type 1 og type 2. Type 1 diabetes kjennetegnes ved at kroppen slutter å produsere hormonet insulin (Hanås 2002). Ved type 1 diabetes blir de insulinproduserende cellene i kroppen ødelagt. Uten insulin i kroppen, forblir sukkeret i blodet og blodsukkeret stiger. Dette er da skadelig og dødelig på lang sikt. Type 1 diabetes behandles ved å tilføre kroppen insulin gjennom injeksjoner.

Ved type 2 diabetes blir kroppen mer eller mindre resistent mot sin egen insulin (Hanås 2002). Kroppen produserer insulin, men den klarer ikke å bruke den effektivt. Diabetikere med type 2 diabetes bruker medisiner i tablettform for å øke insulinfølsomheten i kroppen, slik at en bruker sin egenproduserte insulin mer effektivt. Type 2 diabetikere kan oppleve å måtte tilføre insulin gjennom injeksjoner, hvis andre medisiner ikke fungerer effektivt nok.

Før en klarte å produsere insulin var diabetes en dødelig sykdom (Christophersen og Hanssen 2004). I dag er utfordringen å kontrollere blodsukkeret slik at det ligger på et normalt nivå. Insulin bidrar til å senke blodsukkeret. Studier viser at personer som går med for høyt blodsukker over lengre tid kan oppleve komplikasjoner (Hanås 2002). Komplikasjonene kan være alvorlige problemer med øynene, nyrer, føtter og nerver. En del diabetikere har fått et kortere liv grunnet nyreskader eller hjerte- og karsykdommer (Hanås 2002). Formålet med behandlingen er å begrense sannsynligheten for slike komplikasjoner.

Behandlingen for en diabetiker går da ut på å holde blodsukkeret så normalt som mulig. Det er fare for komplikasjoner hvis en har gjennomsnittlig for høyt blodsukker. Er det gjennomsnittlige blodsukkeret for lavt, er det større fare for insulinsjokk. Insulinsjokk, også kalt føling, oppstår når en diabetiker har fått i seg for mye insulin. Da vil blodsukkeret bli for lavt og hjernen vil ikke få nok sukker. Dette kan da utvikle seg til en akutt kritisk situasjon. Får ikke diabetikeren opp blodsukkeret, kan dette få fatale konsekvenser. Behandlingen mot insulinsjokk er i utgangspunktet å spise raske karbohydrater som går fort til blodet, som matvarer med mye sukker. Ved akutte

situasjoner kan det også brukes injeksjoner med glukagon som øker blodsukkeret meget raskt.

For å holde blodsukkeret så normalt som mulig er en diabetiker avhengig av å ha gode rutiner for de oppgavene en må gjøre i hverdagen. Dette vil jeg se nærmere på i neste delkapittel. Videre tar denne oppgaven utgangspunkt i hverdagen til en type 1 diabetiker, en som er avhengig av daglige insulininjeksjoner.

3.2 Diabetikerens hverdag og analoge og digitale hjelpemidler

Den viktigste oppgaven for en type 1 diabetiker er å bruke riktig mengde insulin. Det som skiller en diabetiker fra en frisk person er selve produksjonen av insulin. En diabetiker må derfor tilføre insulinet gjennom injeksjoner. Insulin er et hormon som senker blodsukkeret. Hovedoppgaven blir da å manuelt tilføre kroppen det insulinet kroppen ikke klarer å produsere selv. En må ta ansvar for en oppgave som kroppen tidligere klarte automatisk. Det å bruke riktig mengde insulin må da beregnes av diabetikere selv. For å kunne beregne riktig mengde må diabetikeren ha kontroll på en rekke faktorer som påvirker beregningen av doseringen av insulin.

3.2.1 Hva påvirker blodsukkeret?

En kan si at behandlingen avgjøres av mange forskjellige faktorer som en må ta hensyn til. Eksempler på faktorer som en diabetiker må ta hensyn til er blodsukkerverdier, matinntak, alkohol, trening, stress, feber og kroppsvekt (Hanås 2002). Dette er alle faktorer som påvirker hvordan en diabetiker gjennomfører sin egen behandling.

Alle disse faktorene har en form for en relasjon med hvordan blodsukkeret svinger og oppfører seg. En kan si at inntak av mat, derav karbohydrater, er med på å øke blodsukkeret (Hanås 2002). Generelt vil inntak av det som betegnes som ”raske” karbohydrater, som sukkerarter og stivelse, øke blodsukkeret raskere enn inntak av ”langsomme” karbohydrater, som kostfiber (Hanås 2002). Videre vil feber være med på å øke blodsukkeret. En infeksjon i kroppen vil frigjøre hormoner som igjen er med på å heve blodsukkeret i kroppen (Hanås 2002). Stress kan også ha en påvirkende faktor på blodsukkeret. Under en stressende situasjon vil en kunne frigjøre hormonet adrenalin som igjen kan heve blodsukkeret (Hanås 2002). Trening vil i hovedsak senke blodsukkeret. I en normal situasjon vil fysisk aktivitet senke blodsukkeret, siden musklene forbruker av sukkeret i kroppen (Hanås 2002). En forutsetting for at trening skal senke blodsukkeret er at det finnes insulin i kroppen. Er det ikke insulin i kroppen kan blodsukkeret øke under trening (Hanås 2002). Inntak av alkohol i kroppen er med på å senke blodsukkeret, siden leveren som skiller ut sukker er opptatt med å bryte ned alkoholen (Hanås 2002). Det vil da bli mindre sukker i blodet og blodsukkeret vil synke.

Det er en rekke faktorer som har betydning for blodsukkeret. Det er viktig å påpeke at disse faktorene påvirker hver person forskjellig. Dette gjør at det er vanskelig å utarbeide en standard dosering for alle diabetikere. En diabetikere må bygge seg opp

en del kunnskap angående disse faktorene som påvirker blodsukkeret for å kunne bestemme den riktige doseringen.

For å gjøre det enklere å bygge seg opp denne kunnskapen har mange diabetikere en del faste rutiner i hverdagen. Dette for å gjøre faktorene så oversiktelige som mulig. Dette betyr at en for eksempel kan prøve å spise måltidene til samme tider av døgnet og spise samme mengde mat til hvert måltid. I tillegg måles blodsukkeret før og etter hvert måltid. I dette eksempelet vil maten få blodsukkeret til å stige. Diabetikeren vil da prøve å dosere en riktig mengde insulin i forbindelse med måltidene, som igjen vil motvirke en blodsukkerstigning. Det er da et mål å få insulindosen til å passe sammen med måltidet.

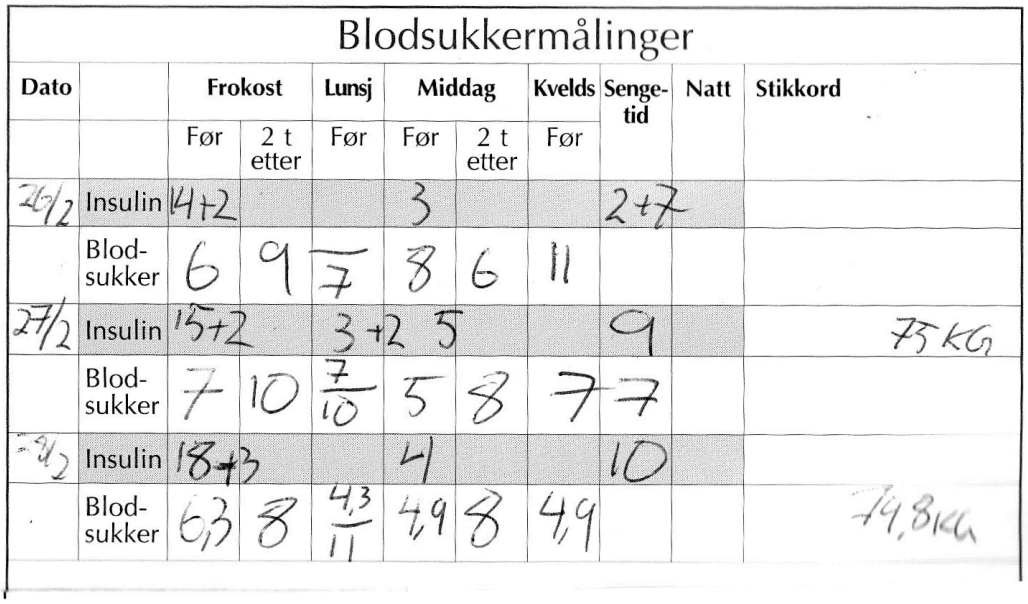
I dette tilfellet vil diabetikeren ha en fast rutine over matfaktoren. Diabetikeren har da mulighet til å sammenligne faktorene rundt måltidene mellom hver dag. Resultatet av blodsukkermålingene vil gi diabetikeren informasjon om hvordan insulindosen passer sammen med måltidene. Dette gir da diabetikeren mulighet til å bygge seg en erfaring over hvor mye insulin som skal benyttes til fremtidige måltid.

En diabetiker kan da ønske å kontrollere og observere flere slike faktorer, slik at behandlingen blir enklere å gjennomføre. Et hjelpemiddel for å gjennomføre dette vil være å bruke en loggbok.

3.2.2 Den manuelle loggboken

En logg er arbeidsverktøyet for en diabetiker. En diabetiker blir en ekspert på sin egen sykdom (Hanås 2002), og loggen er verktøyet for å lære mest mulig om sin egen diabetes. Det er vanlig å loggføre verdier rundt blodsukkeret, insulindoser, trening, alkohol og andre faktorer som kan ha påvirkning på blodsukkeret. Loggen brukes som et veiledningsverktøy og blir fort mye av kjernen i behandlingen. Hva en har gjort før og tidligere erfaringer er viktig i behandlingen av diabetes.

Loggen kan bli ført manuelt, ved hjelp av penn og papir, i en egen loggbok. Diabetikerne får ofte en egen loggbok som blir produsert av firmaer som selger diabetesprodukter (Figur 1). Det finnes forskjellige typer loggbøker som legger opp til litt forskjellige regimer. Noen er bygget opp rundt måling av blodsukker rundt måltider, andre må en skrive inn klokkeslett selv og noen er det plass til kommentarer rundt trening etc.. Dette gjør at loggboken ikke nødvendigvis passer til den enkelte diabetiker. En diabetiker som sitter på et kontor og har faste tider for måltider hver dag, kan ha et annet behov fra en loggbok en det en toppidrettsutøver kan ha. En toppidrettsutøver vil kunne ha stort behov for å notere verdier rundt treningsmengder. Da dette påvirker blodsukkeret i stor grad.



Blodsukkermålinger

Dato		Frokost		Lunsj	Middag		Kvelds	Senge-tid	Natt	Stikkord
		Før	2 t etter	Før	Før	2 t etter	Før			
26/2	Insulin	14+2			3			2+7		
	Blod-sukker	6	9	7	8	6	11			
27/2	Insulin	15+2		3+2	5			9		75 kg
	Blod-sukker	7	10	7/10	5	8	7	7		
28/2	Insulin	18+3			4			10		
	Blod-sukker	6,3	8	4,3/11	4,9	8	4,9			74,8 kg

Figur 1: Et eksempel på en loggbok laget av Aventis Pharma AS. En fører da opp dosen med insulin, som måles i egne enheter og blodsukkerverdiene. I denne loggboken føres ikke klokkeslett, men en skriver verdier i forhold til måltider.

En ulempe med en slik manuell loggbok er at dataene blir lagret i den formen de har blitt notert ned. Ønsker brukeren å gjennomføre statistiske analyser av sine egne data, begrenses denne muligheten siden loggboken blir ført i en form med penn og papir. En diabetiker vil kunne ønskere å analyser mange forskjellige faktorer i forbindelse med behandlingen. Brukeren vil med en loggbok kun ha mulighet til se igjennom dataene i den form de er skrevet ned. Ønsker en for eksempel å regne ut gjennomsnittet til alle blodsukkerverdier før frokost, må dette gjøres for hånd eller en kan digitalisere dataene og behandle de med et program på en datamaskin. En kan da si at ønsker brukeren å analysere dataene effektivt, bør dataene digitaliseres. I en manuell loggbok er ingen av dataene digitalisert.

3.2.3 Blodsukkerapparatet

En diabetiker vil ofte ha behov for å måle blodsukkeret. For å gjøre det benytter en diabetiker et blodsukkerapparat. Et blodsukkerapparat er en håndholt elektronisk enhet, som kan måle blodsukkeret og gi et resultat etter få sekunder. Et blodsukkerapparat kommer gjerne i et sett med flere deler (Figur 2). Delene består ofte av selve blodsukkerpratet, en penn med lansetter og en boks med teststrimler. En lansett brukes sammen med en penn for å stikke et lite hull i huden. Dette gjøres ofte i fingeren. Videre settes en teststrimmel inn i blodsukkerapparatet. På tuppen av denne teststrimmelen tilføres da en bloddråpe. Etter kort tid vil diabetikeren få opp et resultat i displayet som viser en blodsukkerverdi.

Figur 2: Et blodsukkerapparat fra produsenten Accu-Check®. En bruker pennen til høyre for å stikke hull på huden, fks. en finger. Pennen inneholder en lansett, som igjen er en liten nål. Ved hjelp av et fjærutløser stikker denne lansetten hull på huden. Videre tilføres da en liten bloddråpe på tuppen av teststrimmelen som stikker ut av blodsukkeraparatet. Brukeren får da opp blodsukkerverdien i displayet. Boksen i mitten av bildet inneholder flere teststrimler, da en kun kan bruke en strimmel en gang. Sylinderen ved siden av pennen inneholder lansetter.

De fleste blodsukkerapparat inneholder også et minne. Gjennom knappene på apparatet kan brukeren får frem tiden og blodsukkerverdier på tidligere målinger. En del blodsukkerapparater har også en mulighet for å overføre dataene i dette minnet til en datamaskin. En kobler da til kabelen mellom blodsukkerapparatet og en datamaskin. Dataene en kan overføre er selve blodsukkerverdien, klokkeslettet og datoen for målingen. På dagens blodsukkerapparater kan det kun overføres data fra blodsukkerapparatet. Det er ikke mulig å legge data inn på blodsukkerspratet fra et program.

Dette betyr at en kan overføre målinger digitalt fra et blodsukkerapparat. Dataene i minne i blodsukkerapparatet er da digitalisert samtidig som målingen ble utført. Dette åpner opp for andre muligheter rundt bruken av en loggbok. For ved bruken av en manuell loggbok vil dataene bli overført fra blodsukkerpratet til loggboken manuelt. I praksis vil da data som er digitalisert bli gjort om til et manuelt format. Hvis brukeren så må digitalisere den manuell loggboken igjen for å gjennomføre analyser, kan dette sees på som en unødvendig omvei. Dataene er allerede digitalisert i blodsukkerapparatet og det vil være fornuftig å behandle disse dataene videre digitalt. For å få til dette må en ha en digital loggbok.

3.2.4 En digital loggbok

I dag selger produsentene av blodsukkerapparatene også programvare som kan benyttes på en datamaskin. Brukeren kan da gjennom en egen kabel koble blodsukkerappartet til datamaskinen. Videre kan brukeren overføre dataene som ligger i minnet på blodsukkerapparatet over til datamaskinen og programmet.

Et slikt program kan kalles en digital loggbok. En har da overført den manuelle loggboken over til en digital representasjon. Dette medfører flere forandringer i forhold til den manuelle loggboken. Alle andre data enn blodsukkerverdiene brukeren ønsker å loggføre, må nå også digitaliseres. Dataene i blodsukkerapparatet vil bli automatisk digitalisert når brukeren måler blodsukkeret og kan da automatisk legges inn i den digitale loggboken. De andre dataene rund faktorer som matinntak, insulindoser, trening, alkohol etc. må legges inn manuelt i programmet for å få dataene digitalisert.

I forhold til en manuell loggbok vil en spare en del tid og arbeid siden en kan overføre alle blodsukkerverdiene automatisk til loggboken. En annen stor fordel ved en digital loggbok er at mulighetene for analyse forandres betraktelig. Når dataene er i et digitalt format, vil det være enklere å lage mer avanserte analyser av dataene. En kan for eksempel regne ut gjennomsnittsverdier av en rekke data og presentere data på forskjellige måter. Dette vil være en bedre og raskere analyse enn det en kan gjøre for hånd. Det kan også bli lettere for en diabetiker å skreddersy sin egen logg ut i fra egne behov. En digitalisering av loggboken vil kunne gi brukeren en rekke nye muligheter.

Det er allikevel en del utfordringer rundt en slik digital loggbok eller et slikt program. Programmet må støtte den funksjonaliteten en bruker kan ha behov for. Videre må denne funksjonaliteten visualiseres på en måte som gjør at programmet blir brukervennlig for brukeren. Dagens programmer kan kritiseres for å ikke være brukervennlige nok. Bruken av programmene kan i den daglige rutinen for bli vanskelig og tidkrevende å bruke. En er da avhengig av et program som har god brukbarhet. En kan også tro at et program med god brukbarhet vil oppmuntre brukeren til mer bruk av programmet. Som igjen vil føre til en bedre behandling av sin diabetes. Hvilken funksjonalitet og krav til brukbarhet beskrives i neste delkapittel.

3.3 Funksjonalitet for en digital loggbok

De generelle behovene til en digital loggbok for en diabetiker, kan kategoriseres i en liste. Behovene er ofte individuelt forskjellige. Personer med type 1 og type 2 diabetes kan ha forskjellige behov fra hjelpeapparatene både mellom typene og individuelt innad innenfor en type. En kan si at en blir sin egen ekspert på sin egen behandling og det er ikke uvanlig at en diabetiker har mer kunnskap om behandlingen enn en gjennomsnittelig lege (Hanås 2002). Allikevel kan en se at noen behov er generelle for alle diabetikere. En diabetiker trenger å lære hvordan sin egen behandling fungerer, og en logg er da en vesentlig del av denne behandlingen. Disse behovene kan da kalles generelle behov for en diabetiker.

Jeg vil videre se på de generelle behovene rundt bruken av en digital loggbok som hjelpemiddel i behandlingen av diabetes. Det er to viktige behov rundt programvaren. Hvilken funksjonalitet som blir tilbytt og hvor brukbart programmet er, er to viktige behov. Som beskrevet i problemstillingen skal jeg se på hvordan denne funksjonaliteten kan visualiseres i brukergrensesnittet for å oppnå best brukbarhet for bruksituasjonene for en diabetiker. Hvilke funksjoner jeg ser på og krav som defineres rundt brukbarhet, blir beskrevet videre i dette kapittelet.

3.3.1 Generelle behov av funksjonalitet til programvare

Det er viktig at programmet utvider funksjonaliteten i forhold til hva brukeren får ved bruk av en manuell loggbok. Brukeren må føle at programvaren blir et nyttig hjelpemiddel for brukeren. Dette gjøres ved å tilby brukeren funksjonalitet som er tidsbesparende og som oppleves effektivt og nyttig. Funksjonalitet som kan være viktig i et slikt program er loggføring, synkronisering, kategorisering, analyse og sikkerhet. Videre følger noen deler som beskriver disse grunnleggende funksjonalitetene som en digital loggbok bør ha.

3.3.1.1 Loggføring av verdier og notater

Hver dag kan en diabetiker ha behov for å loggføre relevante verdier rundt sin diabetes. Verdier som kan være fornuftige å loggføre er dato, tid, blodsukkeret, inntak av insulin eller annen diabetes medisin, inntak av mat, trening, inntak av alkohol, annen sykdom, feber, føling, trening eller langtidsblodsukkeret. Dette er alle verdier som kan hjelpe en diabetiker med å gjennomføre behandlingen. Allikevel er det individuelt hvilke verdier som en diabetiker har størst behov for å loggføre. Programmet må da tillate at brukeren fyller ut de dataene brukeren finner interessante, og ikke tvinger brukeren til å følge et spesielt mønster.

Dataene som loggføres må ha en fornuftig måleskala. Det er enklere å loggføre verdier som insulindoser og blodsukkernivåer, da disse verdiene er kvantifiserbare. Data rund sykdom, trenging eller matinntak er vanskeligere å måle. Programmet må bruke fornuftige verdier for å kunne loggføre disse dataene. Det vil si at programmet må for eksempel tilby brukeren en måte å måle trening på. En måte en kan gjøre dette på er å beskrive en trening med data for tid og intensitet. Det er viktig for brukeren at en får et forhold til skalaene, slik at dataene blir så konsekvente og korrekte som mulig. Brukeren må kunne beskrive en treningsøkt med mange nok verdier til å kunne skille mellom forskjellig type trening. Samtidig må det ikke være for mange verdier slik at det er vanskelig å skille mellom betydningen av de forskjellige verdien. Det er for eksempel vanskelig å skille mellom 100 forskjellige verdier for intensitet, samtidig som 2 verdier kanskje er litt for lite.

Brukeren må også kunne legge til egen kommentarer til målingene. Kommentarfeltet brukes til å beskrive ekstraordinære hendelser. Det er ikke alle datatyper som kan kategoriseres, slik at det må være mulig å fylle inn notater til loggen.

3.3.1.2 Synkronisering av data

Programvaren må kunne støtte synkroniseringen av data. Hvis programvaren ikke støtter dette må brukeren selv taste inn dataene fra blodsukkerapparatene. Noe som kan bli en tidkrevende og unødvendig prosess. En bruker gjerne et elektronisk blodsukkerapparat som har et minne. Det vil da være fornuftig at dataene i minne kan legges over i programmet gjennom en synkronisering med maskinen. Programmet må da ha funksjonalitet til å overføre dataene fra blodsukkerapparatet inn i programmets eget arkiv, slik at dataene kan brukes sammen med det som er loggført manuelt inn i programmet.

I en synkroniseringsprosess kan dataene overføres mellom enhetene. Skal en synkronisere en enhet med en annen, betyr dette at data blir overført mellom enhetene. En må da definere hvilken informasjon som skal overføres fra blodsukkerapparatet og hvilken informasjon som skal overføres til

blodsukkerapparatet. I dette tilfellet er det størst behov for at dataene blir overført fra blodsukkerapparatet til datamaskinen. Behovet for å overføre data fra datamaskinen til blodsukkerapparatet er ikke en nødvendig funksjonalitet for å gjennomføre bruken av en digital loggbok. Det er heller ingen av dagens programmer som støtter å overføre data fra datamaskinen til blodsukkerapparatet. Allikevel vil brukeren koble en enhet til datamaskinen, som igjen vil gi en teoretisk mulighet til å overføre data begge veier. Det er da viktig at programmet visualiserer hvilken vei dataene blir overført, slik at brukeren får en forståelse for denne prosessen.

Ved overføring av dataene må programmet kunne håndtere dette på en sikker måte. Dataene som blir overført er viktig for brukeren, da dette gir grunnlag for en medisinsk behandling. Det må også visualiseres hvordan programmet behandler dataene som ligger på blodsukkerapparatet. Blir disse dataene slettet fra blodsukkerapparatet, eller vil de bli liggende igjen etter en synkronisering? Hvis dataene blir liggende igjen, må programmet ha funksjonalitet som gjør at samme data fra blodsukkerapparatet ikke blir lagt inn to ganger.

Det finnes mange forskjellige blodsukkerapparater på markedet. Programvaren bør kunne støtte synkronisering av flere forskjellige blodsukkerapparater. I dag er det stort sett produsentene av blodsukkerapparatene som lager programvare som kan brukes som en digital loggbok. Disse programmene vil da kun fungere på produsentens blodsukkerapparater. Det vil ikke være mulig for brukeren å velge fritt blandt blodsukkerapparater og programmer. Det bør da være mulig i et slikt program at brukeren kan velge fritt hvilket blodsukkerapparat som skal synkroniseres med programmet.

3.3.1.3 Kategorisering av data

For å gjøre en analyse av dataene er en nødt til å kategorisere de etter de behovene en har. Typisk ønsker en diabetiker å se verdier rundt måltider, spesielle tider av døgnet, trening eller andre hendelser i livet. Disse kategoriene er viktig i forbindelse med analyse av dataene. En kan for eksempel kun ønske å se verdier en har hatt hver morgen den siste uken, eller hvilke verdier en har hatt 2 timer etter forkost den siste måneden. Programmet må da kunne tilby brukeren en enkel måte å kategorisere dataene etter forskjellige kategorier.

3.3.1.4 Analysering av data

For å kunne forbedre sin egen behandlinge er en avhengig av god analyse av sine egne data. En vil da få en oversikt over sin egen behandling og gjennom analysen kan en se hva som fungerer og hva som eventuelt ikke fungerer i behandlingen. En analyse må hjelpe diabetikeren til ”lese” sine egne data. En slik analyse kan for eksempel være oversikt eller grafer over blodsukkerverdier, insulin- eller medisinbruk, matinntak etc..

Dagens loggbøker tilbyr brukeren en begrenset analyse av dataene. Dataene kan kun vises på en form som er den formen de blir skrevet ned. Hvis brukeren ønsker å se dataene på en annen måte, må alle dataene skrives om. Dette fører til at en i praksis analyserer på grunnlag av den presentasjonen loggboken gir. Det er da viktig at programmet tilbyr bedre og mer avansert analysefunksjonalitet en det en får ved bruk av en manuell loggbok. Mye av fortjenesten ved å bruke et slikt program ligger i å

kunne behandle dataene på mange forskjellige måter, noe som ikke er mulig ved bruk av en vanlig loggbok.

Videre må analysene kunne tilpasses til den enkelte bruker. Programmet må kunne presentere et bredt spekter av analyser. Analysene må også kunne konfigureres etter brukerens egne behov, enten ved å kunne gi parametere til eksisterende analyser, eller ved å definere egne analyser. Kan ikke programmet brukes til den analysen en ønsker, mister programmet mye av sin effekt.

3.3.1.5 Sikkerhet og sikkerhetskopier

Dataene rundt behandlingen for en diabetiker er det viktigste hjelpemiddelet brukeren har. Er det feil i dataene kan dette få betydning for behandlingen og videre få betydning for diabetikerens helse. Dette gjør at dataene må beskyttes mot uønskelige forandringer. Dette betyr at programmet må tydelig visualisere for brukeren når data forandres eller eventuelt slettes. Ved bruken av en vanlig loggbok må en fysisk bruke viskelær eller korrekturlakk for å fjerne loggførte data. I en programvare er det større sannsynlighet for at data kan gå tapt uten at det er en bevisst handling fra brukeren. Programmet må da bevisstgjøre brukerens egen handling når data slettes eller forandres.

Videre må det også være funksjonalitet for å kunne ta sikkerhetskopier av dataene, slik at faren for å tape alle dataene begrenser seg. Sikkerhetskopien må kunne eksporteres til andre lagringsformer slik at en ikke mister alle dataene ved fysisk tap av datamaskinen. Det må også være funksjonalitet for å gjenopprette sikkerhetskopier, slik at alle data i en tidligere sikkerhetskopi kan legges tilbake i programmet.

3.3.2 Hjelpemiddelene og brukbarhet.

Programvaren som brukes har et brukergrensesnitt. Hva er så viktig for brukbarheten til disse brukergrensesnittene med tanke på at de skal brukes av diabetikere? Programmer som har blitt laget, reklamerer ofte med slike utsagn som ”Fast&Easy”, ”Self-learning” og ”Fast data entry”. Dette gjenspeiler at en er ute etter å gi brukerne et program som er lett, raskt og enkelt å bruke. Behovene for brukbarhet i et slikt program kan da kategoriseres etter lærbarhet, fleksibilitet og robusthet.

I de resterende delkapittlene vil disse kategoriene bli utdypet. Det vil også bli trukket paralleller til interaksjonsdesignteori. Teoriene som blir nevnt i de resterende kapitlene vil bli nærmere utdypet i kapittel 4.

3.3.2.1 Lærbarhet

Brukergrensesnittet i et slikt program må være lett å lære for brukerne. Datakunnskapen til en diabetiker varierer like mye som tverrsnittet av befolkningen, fra uerfaren til erfaren databruker. Brukeren må kunne få raskt en forståelse av hvilken funksjonalitet programmet tilbyr og hvordan denne funksjonaliteten fungerer i brukergrensesnittet. Dix (2004) beskriver læring som hvordan nye brukere kan komme i gang med en effektiv interaksjon og få maksimalt utbytte av programmet.

Forutsigbarhet og konsistens er begreper som kan knyttes til lærbarhet (Dix et al. 2004). Brukeren må kunne forutse hvilke handlinger som tilbys i brukergrensenittet, og tilbakemeldingene brukerne får må være konsistente. Hvis tilbakemeldingene er

konsistente medfører det at brukeren kan forutsi effekten av andre handlinger før handlingen er utført. Djajadiningrat et al. (2002) sitt begrep, operasjonsformål og innebygd respons[1], sier noe om informasjonen brukeren får før og etter selve handlingen eller operasjonene. Norman (1998) sitt frembydelses begrep forteller også noe om hva brukeren får av informasjon før selve handlingen gjennomføres.

Familiaritet kan også forbindes med lærbarhet. Det vil si at programmet viser et grensesnitt som brukeren kjenner igjen og er familiært for brukeren (Dix et al. 2004). En vil da kunne si at et familiært grensesnitt er lettere å lære. For å lage et familiært brukergrensesnitt kan en bygge på konvensjoner som allerede finnes i tidligere grafiske brukergrensesnitt eller bruke kunnskap brukeren har i den virkelige verden. Norman (1999) sitt oppfattede frembydelsesbegrep bygger på at brukeren vet hva en skal gjøre fordi en har gjort dette før i andre brukergrensesnitt som bygger på samme konvensjoner. En kan også bedre lærbarheten ved å bruke erfaringer brukeren har fra den virkelige verden. Dette kan en gjøre ved å bruke en metafor. Gjennom å lage et brukergrensesnitt som ligner på den virkelige verden, altså en metafor, er tanken at brukeren skal kunne overføre kunnskapen en har i den virkelige verden inn i bruken av brukergrensesnittet.

Det er da vesentlig for lærbarheten hvordan funksjonene programmet tilbyr, er visualisert. I dagens brukergrensesnitt brukes mye det som kalles WIMP (vinduer, ikoner, menyer og pekere) brukergrensesnitt. Dette kan sees på som en konvensjon i brukergrensesnittet som hjelper brukeren i å lære programmet. Et annet mye brukt begrep i dagens brukergrensesnitt er Shneidermann sitt direkte manipulasjonsbegrep, som gir mulighet for brukeren å operere direkte på virtuelle objekter. Dette er med på å visualisere funksjonene som er tilgjengelige, som igjen kan være med på å bedre lærbarheten av programmet. Holland og Oppenheim (1999) utvidet dette begrepet med sitt begrep som heter direkte kombinasjon. Instrumental interaksjon er også et begrep som er formet av Beaudouin-Lafon (2000) som kan knyttes til direkte manipulasjon.

3.3.2.2 Fleksibilitet

Personalisering er et prinsipp som kan knyttes til fleksibilitet (Dix et al. 2004). Enhver diabetiker har sine egne behov for hvordan hjelpemidlene skal brukes. Det er viktig at hjelpemidlene ikke låser brukeren i et bestemt mønster. Et eksempel på dette er ved kategorisering av data. En bruker kan spise frokost på veldig mange forskjellige tider av døgnet. Er frokost kl 06.00 eller 10.00 er forskjellig fra person til person. Det er derfor viktig at hjelpemidlene er fleksible og kan personaliseres til brukerens egen situasjon.

Et annet prinsipp med fleksibilitet er det som kan kalles dialoginitiativet (Dix et al. 2004). Dialoginitiativet beskriver om programmet ber brukeren om å gi input til programmet, eller om brukeren bestemmer selv når input skal gis til programmet. Når programmet ber brukeren om informasjon, er det naturlig å bruke et dialogvindu. Dette dialogvinduet kan låse brukeren til kun å kommunisere med selve dialogvinduet og brukeren låses ute fra resten av systemet. Når dialogen er ferdig gis brukeren tilgang til resten av systemet. For mange slike dialoger som er initiert av programmet, kan sees på som lite fleksibelt. Direkte kombinasjonsbegrepet til Holland og

[1] Engelsk: Feedforward og inherent feedback

Oppenheim (1999) definerer et brukergrensesnitt hvor mye av hensikten er å unngå bruken av dialogvinduer. Direkte manipulasjon er også med på å begrense behovet for dialogvinduer. Beaudouin-Lafon (2000) sitt begrep instrumental interaksjon er også med på å generalisere direkte manipulasjonsbegrepet. Brukeren opererer her på instrumenter i brukergrensesnittet som igjen kan minske behovet for dialogbokser.

3.3.2.3 Robusthet

Observerbarhet er et prinsipp som kan knyttes til robusthet (Dix et al. 2004). Brukeren må kunne observere eller utforske systemets status. Når brukeren har gjennomført en handling eller operasjon, må brukeren kunne observere forandringen i systemet. Djajadiningrat et al. (2002) sitt innebygdresponsbegrep, legger vekt på å styrke informasjonen og tilbakemeldingen etter at en har utført en operasjon. Et av kravene til direkte manipulasjon er å gi brukeren kontinuerlige tilbakemeldinger. Dette er da med på å øke observerbarheten til systemet.

Et annet prinsipp for robusthet er handlingsoverensstemmelse (Dix et al. 2004). Systemet tilbyr en rekke operasjoner som kan løse selve handlingen brukeren har lyst til å utføre. En kan spørre seg om systemet støtter alle operasjonene brukeren ønsker og om operasjonene utføres på den måten brukeren ønsker. Det er da viktig at operasjonene som er tilgjengelig stemmer overens med de oppgavene brukeren har lyst til å utføre. Robustheten til systemet er da avhengig av at alle oppgavene systemet tilbyr kan mappes til de handlingene brukeren har lyst til, eller ønsker å foreta seg. Mapping er et begrep som Norman (1998) først brukte, og referer til relasjonen mellom objekter. Er det en relasjon mellom alle brukerens handlingsmål for systemet og operasjoner som er tilgjengelig i systemet, er systemet robust.

Gjenopprettelsesmuligheten er også et prinsipp som kan knyttes til robusthet (Dix et al. 2004). Det skal være mulig for brukeren å angre sine handlinger. Har brukeren for eksempel slettet en fil, skal det være mulig å angre denne handlingen. Direkte manipulasjon har også et krav om at det skal være mulig å reversere handlinger brukeren har utført. Dette støtter opp under gjenopprettelsesmulighetsprinsippet. Er det mulig å angre sine handlinger, er systemet robust.

Tiden på respons er også et begrep som kan knyttes til robusthet. Systemet oppfattes som robust når brukeren får en respons umiddelbart etter at en handling eller operasjon er utført. Utfører brukeren en operasjon som bruker lang tid, må brukeren få en tilbakemelding om at operasjonen vil bruke en viss tid før resultatet foreligger. Trykker for eksempel brukeren på en nedtrekksmeny, forventer brukeren å se hele menyen umiddelbart. Direkte manipulasjon, direkte kombinasjon og instrumental interaksjon støtter opp under forutsetninger om umiddelbar respons.

4 Interaksjonsdesign

I dette kapittelet vil jeg utdype teorier innenfor interaksjon som ble beskrevet i forrige kapittel. Disse teoriene vil videre bli brukt i en analyse av skissene som vil bli generert i diskusjonskapittelet (kapittel 6).

4.1 Historie

Begrepet Menneske Maskin Interaksjon (MMI) eller Human Computer Interaction (HCI) ble først tatt i utstrakt bruk på 80-tallet (Dix et al. 2004). Begrepet stammer fra "man-machine interaction" som kom rett etter andre verdenskrig og hadde opphav i mer etablerte forskningsdisipliner. Forsvarsmiljøene hadde behov for å studere hvordan datasystemer påvirket menneskene. Behovet oppstod fordi forsvaret utviklet stadig mer avanserte systemer. Dette motiverte forsvaret til å se på hvordan systemer hadde en sammenheng med menneskets prestasjoner (Dix et al. 2004).

I denne oppgaven vil jeg beskrive teoretiske begreper som kan knyttes til interaksjonsdesign. Interaksjonsdesign har videre en relasjon til menneske maskin interaksjon.

4.2 Mapping

Norman beskriver begrepet mapping som en relasjon mellom to objekter (Norman 1998). Mapping brukes for å forklare relasjonen mellom objekter som representerer en form for styring, eller kontroll og resultatet en får ved bruk av dette styringsobjektet. Eksempelvis kan en se på det å styre en bil. Når en skal styre en bil må en identifisere to former for mapping. Det ene er hvilke av alle kontrollobjektene i bilen påvirker styringen av bilen. Et kontrollobjekt er noe brukeren fysisk kan ta og føle på som ratt, knapper, spaker, brytere, håndtak etc.. Det andre er hvilken vei skal en dra rattet for å styre bilen riktig vei. Brukeren vil da ha valget mellom å dra rattet til høyre eller venstre klokkeretning, siden dette er de eneste mulige valgene en har ved bruken av et ratt. Brukeren vil da dra rattet i den rettingen brukeren vil kjøre. Vil en kjøre til høyre drar en rattet i til høyre. Dette er et naturlig valg og resultatet av handlingen er synlig med en gang. En kan da si at mappingen mellom rattet og hjulene er lett å lære og lett å huske (Norman 1998).

4.2.1 Naturlig mapping

Norman (1998) beskriver begrepet naturlig mapping, som en mapping som tar i bruk fysiske analogier og kulturelle standarder (Norman 1998). En naturlig mapping leder også til en umiddelbar forståelse. Eksempelvis vil det være naturlig at en flytter et objekt opp når en dytter en kontroller opp.

Norman (1998) beskriver også det han kaller en kulturell mapping. En kulturell mapping bygger på kunnskap eller vedtatte sannheter i en kultur. Grunnet sannheter i vår kultur kan en for eksempel koble kraftig med mye og svakt mot lite. Dette gjør at en kan for eksempel bruke lyd for å beskrive en mengde. En kraftig lyd kan bety en stor mengde og en svak lyd kan bety en liten mengde. Norman beskriver dette som

additive dimensjoner. Andre eksempler på slike dimensjoner er volum, vekt, linjelengde eller lystyrke. De additive dimensjonen beskriver kun en inkrementell økning eller avtagning. En kan for eksempel ikke like lett koble en eksakt mengde mot en eksakt lystyrke. Men en økning av mengde er en additiv dimensjon til økning av lydstyrke.

Problemer med mapping er fremtredene og ofte er problemene ved mapping en grunnlegende vanskelighet i brukergrensesnittet (Norman 1998). Norman beskriver et eksempel ved bruk av ”recall” funksjonen ved et telefonsystem som eksempel for å beskrive mappingproblemer. Skal en få telefonen til å ringe tilbake, må en trykke på en recall knapp og taste tallkoden 60 og nummeret en prøver å ringe. Norman beskriver flere problemer ved dette eksempelet. For det første er beskrivelsen av funksjonen relativt kompleks og ikke helt komplett. Hva skjer hvis to personer setter opp samme funksjon samtidig? Hva skjer hvis personen ikke kommer tilbake før om en uke? Hvordan avbestiller man denne tjenesten? For det andre er handlingen du skal utføre ganske tilfeldig og uoversiktlig. Hvorfor skal en taste tallet 60? Kan en ikke taste tallet 22 eller 89? Hvordan klarer en å huske et slik tilfeldig nummer? For det tredje må en utføre en unødvendig handling. Hvorfor må en taste nummeret til den en ønsker å ringe tilbake? En har nettopp ringt nummeret. Hvorfor klarer ikke et slikt avansert telefonsystem å huske hvilket nummer en har ringt? For det fjerde mangler dette systemet en tilbakemelding til brukeren. Hvordan vet brukeren at en har gjort riktig handling? Alle disse problemene kan relateres til problemer med mapping. En enhet er enkel å bruke når en har visualisert de mulige handlingene til systemet, og når kontrollobjektene utnytter naturlig mapping (Norman 1998).

Norman bruker også et annet godt eksempel på styrken til naturlig mapping, det er hvordan en plasserer bryterne på en komfyr. De fleste komfyrene vi kjenner er bryterne plassert på rad og rekke på fremsiden av komfyren. Problemet oppstår når en skal si hvilken bryter som hører til hvilken plate. Rent teoretisk er det her 24 forskjellige muligheter for hvilken bryter som hører til hvilken plat (Norman 1998). For å løse dette har produsentene merket hver bryter, slik at en kan se hvilken bryter som hører til hvilken platen. Uten denne merkingen hadde det ikke vært noe naturlig løsning på dette problemet. Ved å organisere bryterne på en annen måte, illustrerer Norman hvordan naturlig mapping kunne løst denne problematikken,. Hadde en organisert bryterne på samme måte som platene var organisert, eller organisert platene etter samme måte som bryterne, ville problemet blitt løst. Det vil si at for eksempel platen øverst til venstre har bryteren som står øverst til venstre for de andre bryterne. Det ville da vært helt naturlig hvilken plate som hørte til hvilken bryter.

> *Hvis et design er avhengig av merking, kan designet være feil. Merking er ofte nødvendig, men et godt design kan minimere behovet for merking. Hver gang merking er nødvendig bør en vurdere et annet design. (Norman 1998)*

I eksemplet over er det ikke vanskelig å skjønne hva som er god mapping i systemet. Allikevel finner en ofte systemer hvor selv opplagte ting blir feil. Hvordan skal en så gå frem for å få mappingen riktig? Det er da viktig å spørre seg om brukeren kan forstå relasjonen mellom intensjonen og de mulige handlingene brukeren får presentert (Norman 1998). Brukeren må også forstå relasjonen mellom handlingen og dens effekt på systemet, relasjonen mellom systemets egentlige tilstand og hva

brukeren oppfatter, relasjonen mellom brukerens oppfattede systemtilstand og behovene, intensjonene og forventningene fra brukeren (Norman 1998). Dette betyr at bruken av kontrollere i et system bør vær analogt med forventet operasjon i systemet (Norman 1998). Brukerens evaluering av effekten er en kritisk del av en selve handlingen. For å få god mapping må effekten av handlingen svare til forventningene brukeren har. Det er da viktig hvordan tilbakemelding systemet gir til brukeren, da dette er grunnlaget for brukerens evaluering av effekten av handlingen brukeren nettopp har gjort. Responsen fra systemet må gi informasjon som svarer til brukerens intensjon, og må være lett å forstå for brukeren. Systemets respons er da avgjørende for hvordan brukeren oppfatter mappingen og spiller en viktig rolle i å få en så god mapping som mulig.

For god mapping er det viktig å finne ut relasjonen mellom en handling og resultatet, mellom kontrollere og deres effekt og mellom systemets tilstand og synligheten av systemets tilstand.

4.2.2 Mapping og respons

Respons er at systemet sender informasjon tilbake til brukeren om hvilken handlingen brukeren har utført, hva den har gjort og om handlingen er ferdig (Norman 1998). Vi er avhengig av respons når vi bruker et system. Fra vår virkelige verden er vi vant til å få respons på alt en foretar seg. Det vil være veldig rart å ikke få en respons når en for eksempel bruker en blyant. Blir det ikke noe merke etter blyanten, får en heller ingen respons fra blyanten og noe er feil. Et system uten respons ville vært meget vanskelig å bruke.

Norman beskriver det første telefonsystemet som et system med god respons. Når en tast ett tall fikk en automatisk en lyd om at tallet eller knappen var korrekt trykket på. Dette ga brukerne en taktil respons. Klikkelyder på linjen under en oppkobling ga brukeren tilbakemeldinger om progresjonen på oppkoblingen av samtalen. Brukerens stemme ble også ført tilbake til høytaleren i røret, slik at brukeren hadde kontroll på volumet på sin egen stemme. Disse nøye planlagte responsene gjorde det tidligere telefonsystemet veldig lett å bruke (Norman 1998).

I dagens telefonsystemer har en fått veldig mye mer funksjonalitet. En kan sette over samtaler, sette av og på telefonsvarer, besvare andres telefoner fra andre fysiske steder eller sette opp telefonkonferanser med flere brukere. Hvorfor er det da så vanskelig å bruke disse systemene? Problemet ligger i at de nyere telefonsystemene har fått mye mer funksjonalitet og gir mye mindre respons tilbake til brukeren (Norman 1998). Telefonesystemene har funksjonalitet til å gjøre det meste en måtte ønske med en telefon. Problemet ligger i at brukerne ikke forstår hvordan alle denne funksjonaliteten skal brukes, grunnet for lite respons på brukerens handlinger.

Systemet respons er da en viktig del av brukerens interaksjon med maskinen. En god mapping er også avhengig av en god respons fra systemet. En kan videre diskutere hvilken informasjon brukeren får før en utfører handlingen. Som tidligere skrevet er brukerens forventning til responsen avgjørende om handlingen kan anses som vellykket. Norman (1998) sitt frembydelsesbegrep beskriver denne informasjonen brukeren får før selve handlingen.

4.3 Frembydelse

Norman (1998) var den første til å introduserte begrepet frembydelse[2] i sammenheng med interaksjonsdesign. Begrepet hentet han fra psykologiens verden. Begrepet sier noe om de oppfattede egenskapene til en ting, eller et objekt. Frembydelse er de fundamentale egenskapene ved et objekt som sier oss hvordan dette objektet skal brukes (Norman 1998). Norman beskriver frembydelse ved en rekke eksempler. En stol frembyr en form for støtte eller hvile for mennesker, og derfor frembyr den sitting. Et dørhåndtak kan for eksempel utrykke om det trengs å bli trukket eller bli dratt, i avhengig av hvordan det er utformet. Er for eksempel dørhåndtaket utformet som en stor flate en kan legge hånden på, som en ikke kan gripe rundt, er det naturlig å dytte døren. Det motsatte skjer hvis for eksempel dørhåndtaket er utformet som et rør som en kan gripe rundt, da vil det være naturlig å dra i døren. Begge håndtakene frembyr at en skal ta tak i dem, men håndtakets utforming er med på å avgjøre om håndtaket frembyr trekking eller dytting (Norman 1998).

Objektets form og da dens frembydelse er med på å gi oss hint om hvordan objektet skal brukes (Norman 1998). Flater kan dyttes på, knotter er til å skru på, baller er til for å sparkes, sprettes eller kastes. Når en klarer å utnytte et objekts frembydelse, vil brukeren automatisk forstå hvordan objektet brukes. En trenger ingen merking eller små instruksjoner ved siden av objektet. Objektets frembydelse skal gi brukeren nok hint om hvordan objektet brukes. Norman (1998) mener at hvis enkle objekter trenger en forklaring eller en form for merking, er designet feil.

Norman bruker dette frembydelsesbegrepet om objekter som eksisterer i vår virkelige verden, objekter som en kan ta og føle på. Skjermbaserte brukergrensesnitt er virtuelle brukergrensesnitt. Kan dette begrepet brukes om de virtuelle brukergrensesnittene? I en artikkel utvider Norman frembydelsesbegrepet. Norman definerer to typer frembydelse, virkelig og oppfattet frembydelse (Norman 1999). Med virkelig frembydelse mener han den frembydelsen objekter som eksisterer i den virkelige verden har, objekter en kan ta å føle på. Oppfattet frembydelse finner en ofte i forbindelse med skjermbaserte brukergrensesnitt og kan kategoriseres som lærte konvensjoner. En bruker klikker for eksempel på et objekt i et virtuelt brukergrensesnitt, fordi brukeren har lært seg at disse objektene kan klikkes på. Norman mener at objekter i et virtuelt brukergrensesnitt har ingen virkelig frembydelse, bruken av objektene bygger på læring, konvensjoner og respons. Et virtuelt objekt kan ikke fremby klikking. En klikker på det fordi det er en konvensjon, en er lært opp til det (Norman 1999). Norman mener derimot at hvis en låste museknappen på musen fysisk, slik at den kun kunne klikkes på når den var over et objekt en kan klikke på, vil dette være en virkelig frembydelse.

Frembydelsesbegrepet sier oss noe om hvordan brukeren oppfatter visualiseringen av funksjonaliten. Ser en frembydelsen i lys av en hel interaksjon er dette noe som finner sted tidlig i en interaksjon med et system. For å beskrive en hel interaksjon, kan en bruke en interaksjonsmodell.

[2] Engelsk: Affordance

4.4 Interaksjonsmodellen – eksekvering og evaluering

En interaksjonsmodell hjelper oss i å forstå hva som foregår mellom systemet og brukeren. Tanken med en slike modell er at brukergrensesnittet oversetter brukerens behov til maskinen og maskinens tilbakemelding må oversettes slik at brukeren forstår den. Denne oversettelsen, mellom maskinen og brukeren, kan sees på som interaksjonen (Dix et al. 2004).

Normans (1998) modell har vært mye brukt i interaksjonsdesign. Modellen går ut på at brukeren formulerer en plan for sin handling, denne planen eksekveres i brukergrensesnittet. Når planen eller deler av planen har blitt eksekvert, observerer brukeren brukergrensesnittet og evaluerer resultatet av den gjennomførte planen. Videre bestemmer brukeren fremtidige handlinger (Dix et al. 2004). Denne modellen kan også kalles for en eksekvering-evalueringsløkke. Brukeren utfører en handling og evaluerer resultatet, før en utfører en ny handling.

Denne eksekvering-evalueringsløkken kan brukes til å forklare hvorfor brukeren gjør feil. Norman skiller mellom forskjeller ved eksekvering og forskjeller ved evaluering[3]. En kan tenke seg at brukeren og maskinen har hvert sitt språk. Når brukeren ønsker å gjøre en handling, må brukerens språk "oversettes" til maskinens språk. Forskjellen i språket og problemer med oversettelse kan gi brukeren problemer med bruken av grensesnittet. Brukeren formulerer en handling og når brukerens formulerte handling ikke er lovlig i systemet, kan den ikke oversettes. Dette vil da skape problemer for bruken av systemet og kalles forskjell ved eksekvering. Problemene i forskjellen ved evaluering, kan beskrives som forskjellen mellom det systemet presenterer til brukeren og hva brukeren forventet at skulle bli presentert.

Norman (1998) sin modell lager et ryddig bilde av interaksjonen mellom brukeren og maskinen. Denne modellen kan kanskje kritiseres for å bli for ryddig. Den bygger opp under maskinen som et verktøy tankegangen, i stedet for et mediet. Brukeren vet hva en vil gjøre og bruker maskinen til å nå målet. I dag brukes mye direkte manipulasjon i brukgrensesnittet. Direkte manipulasjon er med på å viske ut det klare skillet mellom når brukeren gir en kommando til systemet og når systemet gir en tilbakemelding. Brukeren får hele tiden tilbakemeldinger og brukeren gir mange forskjellige input til maskinen. Djajadiningrat et al. (2002) sitt innebygd respons- og operasjonsformålsbegrep beskriver en slik interaksjon.

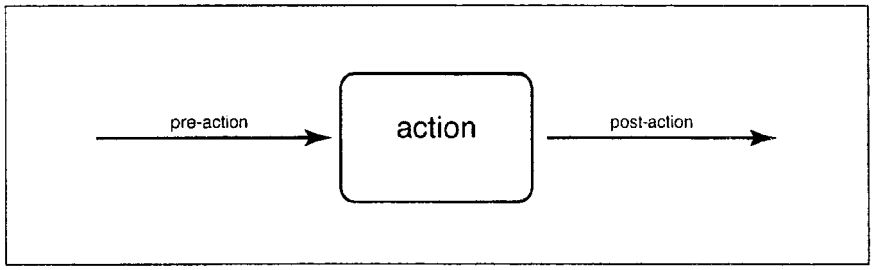
4.5 Operasjonsformål og innebygd respons

Djajadiningrat et al. (2002) formet begrepene operasjonsformål og innebygd respons[4]. Begrepene kommer egentlig som et svar på frembydelsesbegrepet[5] til Norman (1998). Djajadiningrat et al. (2002) mener at frembydelse egner seg godt til å designe fysiske objekter som har en bestemt funksjon, men egner seg mindre til å designe fysiske elektroniske objekter med mange forskjellig funksjoner. Knappene på en elektronisk enhet frembyr riktig nok trykking, men et elektronisk produkt har ofte mye forskjellig funksjonalitet som gjør produktet mer komplekst. En elektronisk enhet er ofte mer

[3] Engelsk: Gulfs of execution, gulfs of evaluation
[4] Engelsk: Feedforward og inherent feedback
[5] Engelsk: Affordance

abstrakt og knappene på elektroniske enheter er ofte like og kan utføre en mengde funksjoner.

Djajadiningrat et al. (2002) mener at frembydelsesbegrepet legger for stor vekt på å invitere brukeren til den rette operasjonen. I stedet må en kommunisere selve formålet med operasjonen for brukeren, som er betydningen av operasjonsformålsbegrepet. Djajadiningrat et al. (2002) ønsker å knytte designet strammere opp mot selve operasjonen. En skiller selve operasjonen opp i en post- og preoperasjonsfase (Figur 3). Det vesentlige er hvilken informasjon brukeren får i disse to forskjellige fasene. Brukerens operasjon med maskinen starter med en preoperasjonsfase, der brukeren får informasjon om meningen og formålet med den spesifikke operasjonen. Dette er det Djajadiningrat et al. (2002) kaller operasjonsformålsfasen. Operasjonsformålet gir brukeren informasjon om hva resultatet av operasjonen vil bli. Når brukeren er i postoperasjonsfasen vil brukeren få en respons. Denne responsen kan være informasjon om hva operasjonen har utført, en bekreftelse på navigasjon eller informasjon om en prosses som er under utføring. Innebygd respons er når brukeren føler denne responsen som en naturlig konsekvens av operasjonen brukeren har gjort. I likhet med Norman (1998) mener Djajadiningrat et al. (2002) at problemet oppstår når det kan synes at det ikke er noen sammenheng mellom operasjonen og responsen. Djajadiningrat et al. (2002) bruker et eksempel for å beskrive dette. En elektronisk enhet har ofte en lyd som en respons når brukeren trykker inn en knapp. Problemet med denne responsen er at den kunne tilhørt en hvilken som helst operasjon. Det er umulig å bruke responsen til å spore seg tilbake til hvilken handlig som ble utført. Det er ingen relasjon mellom operasjonen og responsen. I innebygd respons må det være en stram kobling mellom operasjonen og responsen.



Figur 3: Pre-operasjon og post-operasjon. Hentet fra Djajadiningrat et al. (2002)

Djajadiningrat et al. (2002) beskriver hvordan en interaksjon kan deles opp i flere sykluser eller operasjoner, ved å knytte interaksjonen strammere opp til selve designet. Dette kan sees på som en motsetning til Norman (1998) sin måte å se hele interaksjonen eller handlingen i en eksekverings-evalueringsløkke. Djajadiningrat et al. (2002) knytter da et operasjonsformål og en innebygd respons til hvert objekt i brukergrensesnittet, og hvert objekt kan ha en interaksjon i seg selv. Brukeren trenger da ikke på samme måte å bestemme seg for en handling tidlig i interaksjonen, for å så eksekvere denne handlingen. Objektene blir visualisert i brukergrensesnittet og brukeren kan operere på hvert objekt som har sitt eget operasjonsformål og innebygd respons. Denne interaksjonen med objektene blir også kalt for direkte manipulasjon.

4.6 Direkte Manipulasjon

En av de grunnleggende funksjonalitetene i det skrivebordsmiljøet vi kjenner i dag er direkte manipulasjon (skrivebordsmiljøet vil bli beskrevet i kapittel 4.9.2). Direkte manipulasjon beskriver brukerens mulighet til å manipulere objekter og navigere i den virtuelle verden (Preece 2002). For eksempel kan virtuelle objekter manipuleres ved å flytte, velge, åpne, lukke eller zoome inn og ut på objektene. Uten bruken av direkte manipulasjon ville det vært vanskelig å bruke skrivebordsmetaforen. En er her avhengig av å kunne manipulere på objekter en ser i brukergrensesnittet.

Ben Shneidermann (1998) formet begrepet direkte manipulasjon. En kan beskrive begrepet med følgende punkter (Dix et al. 2004):

- Objekter som er interessante skal være synlige
- Inkrementell handling i brukergrensesnittet med kontinuerlig tilbakemeldinger
- Mulighet for å reversere alle handlinger, slik at brukeren kan utforske uten å bli straffet for det
- Alle handlinger skal være syntaktisk riktige, alle brukerhandlinger er lovlige operasjoner
- Erstatte komplekst kommandospråk med mulighet for handlinger direkte på synlige objekter.

Direkte manipulasjon gir en rekke fordeler når en bruker det i brukergrensesnittet. Det hjelper nye brukere å raskt lære basis funksjonalitet som grensesnittet tilbyr. Brukeren vil over tid huske hvordan operasjonene blir utført, siden operasjonene har en visuell utforming. Det er i utgangspunktet veldig lite bruk for feilmeldinger, siden alle handlinger er syntaktisk riktige. Gjør brukeren noe som ikke er lov, vil ingen ting skje og brukeren vil oppfatte at denne handlingen var feil. Brukeren kan også raskt se om handlingen gjør det en ønsket å oppnå, hvis ikke vil en gjøre noe annet. Direkte manipulasjon vil gi brukeren en mindre følelse av usikkerhet rundt brukergrensesnittet. Brukeren oppnår heller følelsen av mestring og kontroll over brukergrensesnittet (Preece 2002).

Bruken av direkte manipulasjon i brukergrensesnittet har også noen ulemper. En kan oppleve at brukeren tar brukergrensesnittet med direkte manipulasjon for litterært (Preece 2002). Brukeren kan fort forvente at objektene i den virtuelle verden har nøyaktige samme egenskaper som i den virkelige verden. Et godt eksempel på dette er en versjon av søppelbøtten Apple har i sitt brukergrensesnitt. Når en skal løse ut en ekstern harddisk fra maskinen kan en gjøre dette ved å dra ikonet av disken ned i søppelbøtten. Det er mange brukere som kvier seg for dette, fordi de er redde for å slette hele disken (Preece 2002). Brukeren blir her forvirret fordi søppelbøtten brukes til å representere flere handlinger, det å slette filer og det å løse ut en disk (Preece 2002). En annen ulempe med dirkete manipulasjon er at det er ikke alle handlinger som kan beskrives ved hjelp av manipulerbare objekter. Noen handlinger utføres best ved hjelp av tekstlige instruksjoner. Hvis en for eksempel skal bytte ut en mengde like ord i en tekst med et annet, vil det være lettere å gjøre dette gjennom en kommando som sier ”bytt ut et ord med et annet”. Ved bruken av direkte manipulasjon må en for eksempel bla gjennom hele teksten og bytte ut ordene ved hjelp av ”dra og slipp” funksjonalitet. Det kan da virke mer tungvint å gjennomføre en slik handling gjennom direkte manipulasjon.

Når en bruker direkte manipulasjon blir skillet mellom inndata og utdata mindre. Eksekverings-evalueringshierarkiet, som er beskrevet av Norman (1998), er blitt annerledes. I et direkte manipulasjonssystem vil utdatauttrykkene være med på å forme et subsett av indatauttrykk (Dix et al. 2004). Dix et al. (2004) beskriver for eksempel visualiseringen av en fil vil være en utdata for et direkte manipulasjonssystem. Ved hjelp av denne utdataen kan brukeren ta tak i filen og dra den dit en vil bruke den. Systemet må svare med å vise brukeren en utdata om at han har tatt tak i filen og vil bruke denne til noe. Hva brukeren vil gjøre med filen er ennå ikke bestemt. Brukeren kan da for eksempel velge å flytte filen, slette filen eller kanskje åpne den i et program. Hvilken operasjon som skal utføres på filen vil ikke bli avgjort før brukeren er ferdig med sin handling. Systemet er da nødt til å gi en rekke utdata underveis før systemet har fått vite hvilken handling en skal gjøre. Denne typen interaksjon kan også knyttes til de tidligere beskrevede begrepene til Djajadiningrat et al. (2002), operasjonsformål og innebygd respons. I motsetning vil en i en vanlig inndata utdata tankegang som Norman (1998) sin eksekverings-evalueringsløkke, vil brukeren gi kommandoen ”slett denne filen”, ”flytt denne filen” eller ”åpne denne filen”. Brukeren vil da gi kommandoen som en helhet, og ikke ved hjelp av mange små interaksjonsdeler.

Direkte manipulasjon er mye brukt i dagens brukergrensesnitt på datamaskinen. Det har blitt beskrevet flere teorier som kan utvikle direkte manipulasjon. Direkte kombinasjon er et slikt begrep.

4.7 Direkte Kombinasjon

Holland og Oppenheim (1999) var de som først brukte begrepet direkte kombinasjon. Begrepet er en utvidelse av det direkte manipulasjonsbegrepet. Direkte kombinasjon er et navigasjonsrammeverk for å gi brukeren en bedre oversikt over hvilke operasjoner eller handlinger som er tilgjengelig i systemet. Holland og Oppenheim kategoriserer det direkte manipulasjonsbegrepet i en ”substativ-verb” kategori. Med dette mener de at i direkte manipulasjon velger brukeren et objekt eller et ”substantiv” for å så velge en operasjon eller et ”verb” fra en meny eller en knapp eller liknende, for å velge hvilken operasjon som skal utføres på det valgte objektet. Brukeren vil til slutt få mulighet til å gi eventuelle ekstra parametere som operasjonen trenger.

I direkte kombinasjon fokuserer en på par av interaksjonsobjekter (Holland og Oppenheim 1999). For hvert par av objekter eksisterer det en eller flere tilhørende operasjoner eller handlinger. En kan si at en velger to objekter eller ”substantiver” for og så velge hvilken handling eller ”verb” som skal utføres med denne parkombinasjonen. Dette har sin styrke i forhold til at brukeren ser med en gang hvilke operasjoner som er tilgjengelige. I et vanlig direkte manipulasjonsgrensesnitt må brukeren lete i gjennom mange forskjellige menyvalg som både kan være relevante og irrelevante. I programmer med mange mulige operasjoner kan det fort bli uoversiktlig for brukeren. Operasjonen kan ofte være navnet vanskelig eller være vanskelige å finne eller navnet kan være vanskelig å forstå. Direkte kombinasjon gjør det lettere for brukeren å søke rundt etter riktig operasjon ved å kombinere to relevante objekter (Holland og Oppenheim 1999).

Interaksjonsobjektene i direkte manipulasjon kan være mer enn ikoner. Tall, grafiske elementer, diagrammer, tekstvalg, lister, samling av celler, linker, filer, mapper eller

vinduer etc. kan være gyldige interaksjonsobjekter i direkte kombinasjon. Kombinerer en for eksempel en fil med tallet syv, kan for eksempel mulige handlinger være å skrive ut syv eksemplarer av denne filen eller forstørre denne filen syv ganger (Holland og Oppenheim 1999)

I direkte kombinasjon må alle objekter i systemet kunne kombineres og gi mulige operasjoner eller handlinger. Holland og Oppenheim (1999) definerer tre prinsipper for direkte kombinasjon: Alle objekter av interesse skal være synlige, alle objekter må kunne behandles som et interaksjonsobjekt og for hvert interaksjonsobjekt må det være mulig å gjøre en eller flere operasjoner ved kombinasjon av et annet objekt.

I direkte manipulasjon opplever en ofte at en har behov for dialogbokser for å få nødvendig informasjon fra brukeren (Holland og Oppenheim 1999). Dialogboksene låser brukeren slik at brukeren blir tvunget til å gi den informasjonen systemet trenger. Brukeren må da gi informasjonen eller avbryte hele handlingen. Disse dialogboksene mener Holland og Oppenheim (1999) bryter med grunnprinsipper for direkte manipulasjon, for objekter skal være tilgjengelige og mulige å manipulere til en hver tid. Dialogboksene kan også være irriterende for brukerne. Holland og Oppenheim (1999) beskriver dette med et eksempel. Hvis en får opp en dialogboks hvor en må skrive inn en filbane, kan en oppleve at filen ligger ved siden av boksen, men den er ikke tilgjengelig. En vil kunne ønske å dra filen over i boksen i stedet for å skrive inn selve filbanen til filen. Holland og Oppenheim (1999)mener at direkte kombinasjon vil løse noen slike problemstillinger med dialogbokser. En er ikke bundet opp til ”substantiv-verb” tankegangen. Hvor en velger et objekt, en handling og eventuelt får opp en dialogboks over hvilke argumenter en trenger for å gjennomføre handlingen. I direkte kombinasjon får en frem de mulige handlingene ved å kombinere to objekter. Brukeren får nå mulighet til å utforske mulige valg ved å kombinere forskjellige objekter, uten å måtte utføre valgene.

Direkte kombinasjon trenger ikke nødvendigvis være knyttet opp til kun to og to interaksjonsobjekter. Holland og Oppenheim (1999) beskriver scenarioer hvor en kombinerer flere objekter. En vil da få opp alle mulige operasjoner eller handlinger som kan gjennomføres på alle objektene.

Direkte manipulasjon er da en utvidelse av direkte manipulasjonsbegrepet. Et annet begrep som bygger på direkte manipulasjon er instrumental interaksjon.

4.8 Instrumental interaksjon

Beaudouin-Lafon (2000) formet begrepet instrumental interaksjon. Instrumental interaksjon har som formål å utvide og generalisere prinsippene til direkte manipulasjon. Beaudouin-Lafon (2000) definerer instrumental interaksjon som en interaksjonsmodell. En interaksjonsmodell er et sett av prinsipper, regler og egenskaper som skal hjelpe designeren til å lage brukergrensesnitt (Beaudouin-Lafon 2000). Tanken er at denne modellen kan brukes til å evaluere og kategorisere andre interaksjonsteknikker.

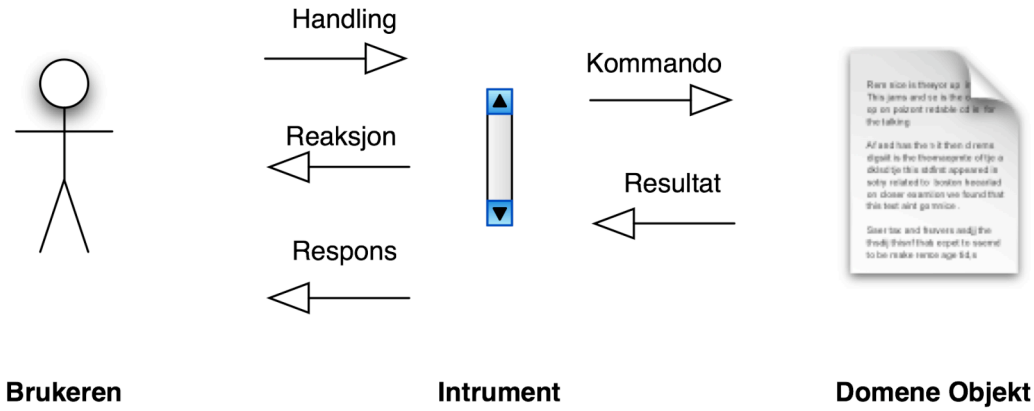
Det er to viktige begreper for å beskrive grafiske brukergrensesnitt i instrumental interaksjon. Det er domeneobjekter og interaksjonsinstrumenter. Beaudouin-Lafon (2000) beskriver domeneobjekter som et sett av data som representerer objekter

brukeren er interessert i, i et brukergrensesnitt. Objektene har forskjellige atributter som beskriver deres karakteristikk (Beaudouin-Lafon 2000). Brukeren opererer på objektene ved å forandre deres atributter, eller ved å behandle de i en helhet ved å flytte, lage nye, eller slette dem. Beaudouin-Lafon (2000) beskriver for eksempel at teksten i en teksteditor kan sees på som et domeneobjekt. En kan da forandre teksten ved å skifte farge, skrifttype, skriftstørrelse etc. eller en kan slette hele teksten eller flytte på den. Disse dataene kan da sees på som atributter til selve teksten. En bruker da instrumenter for å forandre på atributtene til teksten eller domeneobjektet. I dette tilfellet vil da instrumentene gi brukeren mulighet til å forandre farge, skrifttype, skriftstørrelse etc., som igjen er atributtene til teksten. Det er viktig å merke seg at atributter kan gå over til å bli domeneobjekter, objekter av interesse for brukeren. Samles atributtene i forskjellige tekststiler som beskriver forskjellige formateringer av teksten, vil disse tekststilene kunne bli domeneobjekter i seg selv (Beaudouin-Lafon 2000).

Interaksjonsinstrumenter kan sees på som et mellomledd mellom domeneobjektene og brukeren (Beaudouin-Lafon 2000). En bruker instrumentet for å betjene domeneobjektet. Instrumentet oversetter brukerens handling over til en kommando som blir sendt videre til objektet. Brukeren får så en reaksjon fra instrumentet for å vise brukeren at instrumentet er i bruk. Videre får brukeren en respons som forteller at kommandoen har blitt utført.

Beaudouin-Lafon (2000) deler interaksjonsinstrumenter opp i to lag, fysisk handling og reaksjonen fra instrumentet (Figur 4). Fysisk handling er den handlingen som foregår mellom brukeren og selve instrumentet, og reaksjonen fra instrumentet er den interaksjonen som skjer mellom instrumentet og objektet det blir brukt på. Reaksjonen deles opp i resultat og kommando. Kommando er den beskjeden instrumentet sender videre til objektet, resultatet er det instrumentet får tilbake fra objektet, som kan bli omdannet til respons tilbake til brukeren. Beaudouin-Lafon (2000) bruker rullefeltet[6] som et eksempel på et slikt instrument. Rullefeltet brukes til forandre hvilke deler av et dokument som er synlig for brukeren. Når en bruker klikker på rullefeltet, sendes en kommando til dokumentet. Reaksjonen fra rullefeltet er at pilen brukeren har trykket på blir opplyst, eller markert. Videre får brukeren en respons om at andre deler av dokumentet blir synlig for brukeren. Brukeren får også en respons fra instrumentet om den nye posisjonen rullefeltet har registrert på dokumentet.

[6] Engelsk: Scrollbar



Figur 4: Interaksjonen mellom brukeren, interaksjonsinstrumentet og domeneobjektet. Hentet fra Beaudouin-Lafon (2000)

Bruken av instrumenttankegangen forklarer Beaudouin-Lafon (2000) med en sammenligning med den virkelige verden. Vi er allerede vant til å bruke redskaper eller instrumenter i den virkelig verden. Skal vi skru på et lys i et rom bruker vi redskapet en lysbryter. I denne sammenligningen blir lyset i rommet domeneobjektet som er av interesse for brukeren, og lysbryteren blir instrumentet.

Beaudouin-Lafon (2000) deler aktiveringsprosessen av instrumenter i to forskjellige kategorier. I dagens WIMP-grensesnitt finner en eksempler som Beaudouin-Lafon (2000) bruker for å forklare dette. Ser en på det tidligere nevnte rullefelteeksempelet, blir instrumentet aktivert når musen er over rullefeltet. Brukeren trenger kun å klikke med musen for å bruke instrumentet. Ser en på et eksempel fra tegneprogrammer, har en ofte en meny i programmet hvor en kan velge verktøy eller instrumenter. For eksempel kan en velge et instrument som lager en trekant for deg. For å aktivere dette instrumentet må en føre musen over riktig ikon, for å så klikke på dette ikonet. Da vil en skifte over til en modus som gjør at musepekeren vil tegne en trekant, neste gang en plasserer musepekeren et egnet sted og klikker. Denne modusen forsvinner kun når en velger å gå over i en annen modus. Beaudouin-Lafon (2000) beskriver dette som to forskjellige aktiveringsprinsipper og vurderer de etter hvor plasskrevende og tidkrevende de er å bruke. Rullefelteksempelet er mer plasskrevende og mindre tidkrevende enn tegneprogrameksempelet. Det krever mer plass fordi rullefeltet må være synlig til enhver tid for å kunne bli brukt. Siden det er synlig til en hver tid, trenger en ikke å bytte modus for å benytte dette instrumentet. Dette gjør rullefeltet mindre tidkrevende å bruke. Instrumenter som trenger å bli aktivert er mer tidkrevende å bruke (Beaudouin-Lafon 2000). Design av slike instrumenter vil da være en avveiing mellom tidsbruk og plassbehov. Beaudouin-Lafon (2000) viser til at en utvidelse av det fysiske brukergrensesnittet vil kunne forandre på dette forholdet. Ved å legge til en ekstra funksjonalitet i innputtenhetene til brukeren, kan en koble den fysiske enheten direkte med et instrument. En mus med rulleknapp er et slikt eksempel. Her kobles rulleknappen direkte med et instrument i brukergrensesnittet. Dette vil redusere både plassbehovet og aktiveringstiden for et instrument (Beaudouin-Lafon 2000).

Beaudouin-Lafon (2000) beskriver flere egenskaper ved instrumenter. Egenskapene kan brukes til å beskrive og klassifisere instrumentene. Blant disse egenskapene finner en graden av indireksjon og graden av kompatibilitet (Beaudouin-Lafon 2000).

Graden av indireksjon defineres videre inn i et avstand- og tidsaspekt[7] (Beaudouin-Lafon 2000). Avstandsaspektet er avstanden fra instrumentet til objektet det kontrollerer. Et instrument for å velge objekter i et tegneprogram, har typisk et veldig lite avstandaspekt. Objektet en kontrollerer er under eller helt ved siden av instrumentet. Dialogbokser kan for eksempel ha et stort avstandsaspekt, siden dialogboksen kan være grafisk et helt annet sted på skjermen enn det objektet dialogboksen kontrollerer. Det er ikke nødvendigvis negativt med et stort avstandsaspekt. Beaudouin-Lafon (2000) beskriver at det er ofte en god avstand fra selve lysbryteren til selve lampen en kontrollerer. Instrumentet er da plassert slik at det er lett tilgjengelig. Tidsaspektet beskriver hvor lang tid det tar fra en har brukt et instrument til brukeren får en respons fra objektet (Beaudouin-Lafon 2000). Av og til skjer dette i sanntid, som ved bruken av et rullefelt. En får en respons fra objektet med en gang. Ved bruken av for eksempel en dialogboks vil en ikke få en respons før slutten av hele handlingen.

Graden av kompatibilitet beskrives av Beaudouin-Lafon (2000) som forholdet mellom den fysiske handlingen brukeren gjør og responsen en får fra objektet en kontrollerer. Beaudouin-Lafon (2000) beskriver begrepet med noen eksempler. Drar en for eksempel et objekt med musen i brukergrensesnittet, er dette en høy grad av kompatibilitet. Objektet på skjermen følger håndens og musens bevegelse. Drar en for eksempel i et rullefelt, er dette en lav grad av kompatibilitet. Når en drar hånden nedover, vil objektet gå oppover. Bruker en et tekstfelt for å beskrive størrelsen på en skriftstørrelse, er dette en lav grad av kompatibilitet. Inndatatypen er forskjellig fra datatypen til resultatet (Beaudouin-Lafon 2000). Et tall beskriver dårlig hvor stor teksten kommer til å bli.

Instrumenter er da også en måte å visualisere funksjonalitet til programmet. Gjennom instrumentene kan en editere og redigere parametere i programmet. I neste kapittel vil jeg presentere flere måter å visualisere funksjonalitet på.

4.9 Visualisering

Når en bruker en enhet er det viktig for brukeren at mulige handlinger er synlig og at det er synlig hvilken tilstand enheten har (Norman 1998). Norman bruker et telefonsystem som eksempel for å vise hvor viktig synlighet av systemets funksjoner er. Noen telefonsystemer har mulighet for å besvare telefoner som ringer i andre steder, i for eksempel et kontorlandskap. Ønsker en å gjøre dette må en for eksempel taste et bestemt nummer for og så vente på en tone for og så taste et annet nummer. Problemet med dette er at funksjonaliteten som ligger i telefonen ikke er spesielt synlig for brukeren. Ønsker brukeren å benytte seg av denne funksjonen må brukeren huske en rekke tallkombinasjoner og rekkefølgen disse tallene skal tastes for å gjennomføre det en ønsker. Funksjonen er ikke synlig og dårlig visualisert for brukeren.

Telefonsystemet i dette eksempelet har masse funksjonalitet. Er det noe sammenheng mellom hvor komplekst et system er opp mot hvor vanskelig det er å bruke? Norman tar utgangspunkt i en bil for å belyse denne problematikken. En bil har en mengde

[7] Beskrevet som ”spatial offset” og ”temporal offset”.

kontroller og masse funksjonalitet. En kan kontrollere lyset, motoren, radioen, vinduene, vindusviskerne, setene, bensinlokket osv.. Allikevel oppfattes en bil enklere å bruke en et slikt telefonsystem (Norman 1998). En bil og et telefonsystem kan da sees på som komplekse systemer. Hva er det da som gjør at bilen er enklere å bruke? Mye av problemet ligger i visualiseringen av funksjonene. I en bil er det stort sett en knapp for hver funksjon. I telefonsystemet skjuler all funksjonalitet seg bak en rekke tilfeldige tall med kryptiske koder og en må en bruke tallknappene for å få brukt funksjonaliteten. Tallknappene er heller ikke merket med hvilken funksjonalitet de tilbyr. Når antall mulige handlinger overskrider antallet på kontrollere, vil det kunne bli vanskeligheter (Norman 1998). Videre er det vanskeligere å huske kontrollere som har mer enn en funksjon. Funksjoner er generelt usynlige for brukeren, de må visualiseres. I bileksempelet er hver funksjon visualisert med hver sin kontroller. Synligheten av hver kontroller minner brukeren på hvilke funksjoner som er tilgjengelige og brukeren vil få mindre å memorere (Norman 1998).

Hvordan visualiserer vi så handlinger som er tilgjengelige i et program på en datamaskin? Gjennom historien har det blitt brukt og kommet frem en rekke forskjellige konsepter eller begreper. Jeg vil nå beskrive et lite utvalg av disse.

4.9.1 Kommandolinjen

Kommandolinjen var et av de første interaktive brukergrensesnittet som ble allment brukt (Dix et al. 2004). Brukergrensesnittet er fortsatt i bruk, med sine styrker og svakheter. Det fungerer på den måten at brukeren gir en kommando i form av et ord eller utrykk og maskinen utfører denne kommandoen. Brukergrensesnittet har sitt opphav fra teleprinteren og ble tatt i bruk på 50-tallet. Denne metoden for å gi kommandoer til maskinen ble fort å foretrekke fremfor hullkortene, men forutsatte bruk av en skjerm.

Kommandolinjen har sin største fordel ved at en får direkte adgang til systemet. Det finnes i dag flere systemet som bruker denne typen grensesnitt. Den er fortsatt i flittig bruk blant brukerne av operativsystemer som Unix og Linux. Kommandolinjen er veldig fleksibel da kommandoene kan spesifiseres med forskjellige valg. Dette gjør at brukeren får tilgang til en mengde av muligheter, noe som kan være vanskelig å fremstille i form av menyvalg. Kommandolinjen åpner også for en enkel mulighet for det en kaller ”skripting”. En lager ferdigdefinerte oppskrifter av kommandoer som blir utført etter hverandre, dette gir systemet en mulighet til å lage avanserte skript som kan gjøre mye ved kun en kommando. Dette er også en funksjon som ikke er så lett å få til i form av menyvalg eller objekter.

Det største brukbarhetsproblemet med kommandolinjen, er at brukeren er nødt til å huske hvilke kommandoer som kan gis. Dette gjør at synligheten til systemets funksjonalitet er veldig liten, i motsetning til hvordan synligheten av funksjoner er i skrivebordsmiljøet.

4.9.2 Skrivebordsmiljøet og WIMP

Et mye brukt konsept i dagens brukergrensesnitt er det vindusbaserte brukergrensesnitte. Dette brukgrensesnittet har sitt opphav fra forskningen på ”Stanford Research Institute” og “Massachusetts Institute of Technology” (Brad

1998). Det var arbeide hos Xerox PARC som først presenterte denne vindubaserte måten å tenke på. De presenterte sitt ”Star” system i 1981 (Preece 2002). Brukergrensesnittet var designet som et kontormiljø og var designet for brukere som ikke var interessert i det bakenforliggende systemet. Designerne prøvde å gjøre datamaskinen så ”usynlig” som mulig for brukeren. Ved å designe det grafiske brukergrensesnittet som et kontormiljø, ved hjelp av vinduer, ikoner, menyer og pekere (Smith og Irby 1998). Dette systemet gjorde derimot aldri noen kommersiell suksess. Apple adopterte dette designet og Apple ble da den første til å bruke denne vindusbaserte skrivebordsmetaforen på sine datamaskiner. Ideen ble senere fanget opp av Microsoft som presenterte sitt skriverbordsmiljø i operativsystemet Windows.

En generalisering av disse typene grensesnitt er å kalle de WIMP-grensesnitt. WIMP står for vinduer, ikoner, menyer og pekere (Dix et al. 2004). Vinduene er et uavhengig område på skjermen som kan inneholde tekst eller grafikk. De har en generell funksjonalitet felles. En kan for eksempel flytte vinduene rundt på skjermen, gjøre om størrelsen på vinduene og minimere vinduene. Ikonene er små bilder som kan representere forskjellige typer objekter eller funksjoner. Menyene gir brukeren mulighet til å velge mellom kommandoer maskinene kan tilby. Menyene organiseres i forskjellige nedtrekksmenyer for å kunne lettere navigere rundt i kommandoene som er mulige. Pekeren bruker en til å velge elementer på skjermen. Dette er brukerens mulighet til å gi input til systemet. Pekeren fremstår på forskjellige måter avhengig av hvilken funksjon den har. Dette er en vanlig brukergrensesnittstil for de vanligste interaktive datasystemene vi kjenner i dag.

Ikonene er da med på å visualisere objekter og funksjonalitet i brukergrensesnittet. I neste kapittel vil jeg se på bruken av ikoner i brukergrensesnittet.

4.9.3 Ikoner

Ikoner er en viktig del av det skrivebordsbrukergrensesnittet vi kjenner i dag. Ikonene brukes til å representere objekter eller mulige handlinger i brukergrensesnittet. Ved hjelp av små bilder eller tegninger prøver ikonet å beskrive objektet eller handlingen den tilbyr. I Macintosh Human Interface Guideline (1992) blir ikoner beskrevet som en grafisk representasjon av objekter som dokumenter, lagringsmedier, mapper, programmer og søppelbøtten. Videre skal ikonene se ut som deres referanse i virkeligheten, så lenge det er mulig. Brukeren kan velge å åpne, flytte, kopiere og kaste vekk ikoner.

Ikonene er populære fordi de kan utrykke en handling eller et objekt på en relativ effektiv og lite plasskrevende måte. Skulle en utrykke det samme med ord, ville dette krevd mer plass på skjermen. En kan si at ”et bilde kan utrykke mer enn tusen ord”. Ikoner har den fordelen at de raskt kan læres og gjenkjennes (Baecker et al. 1991). Brukeren trenger egentlig ikke å huske hvordan ikonet ser ut, men brukeren må kunne gjenkjenne det når en ser ikonet. Ikonet har også den fordel at brukerne liker å se på bilder og ikoner fremfor å lese tekstlige beskrivelser (Haramundanis 1996). Det er raskere for brukeren å kjenne igjen ikoner eller bilder, så lenge en har sett bildet eller ikonet før. Ikonene har også den fordelen at de kan utrykke universale utrykk som ikke trenger oversettelse. I trafikken brukes mange ikoner. Veldig mange trafikkskilt består av små bilder eller ikoner. Forskning har her vist at symboler er lettere gjenkjennelig fra avstand enn tekstlige utrykk (Apple Computer 1992).

Det er allikevel en del utfordringer forbundet med bruken av ikoner. Utrykket ”et bilde kan bety mer en tusen ord”, gir utfordringer når en ønsker at et ikon kun skal bety et ord, eller en handling. En kan spørre seg hvilke av disse ”tusen ordene” er det ikonet egentlig skal representere. En kan også snu på det og si at, ”en trenger tusen bilder for å utrykke et ord”. Da fort assosiere et ikon med flere enn en handling eller et objekt. Hvordan brukeren velger å tolke ikonet, er avhengig av brukerens tidligere erfaringer eller kunnskap. Det er også tydelig at kulturen brukeren kommer fra også har betydning for tolkningen av ikonet. En må da ta hensyn til kulturelle og kontekstspesifikke spørsmål når en skal designe et godt ikon (Preece 2002). Da ikoner først ble brukt i england for å representere dame og herre toalettet, var det turister som ikke forsto hva som var dame og herre toalett ut ifra ikonene som hang på dørene. Grunnen var at de ikke forstod forskjellen på en person med bukser eller skjørt (Preece 2002). En kan da spørre seg om kanskje ikonene sin viktigste oppgave er å skille mellom funksjonene, ikke nødvendigvis representer funksjonene.

Det er enklere å representer subjekter enn handlinger med ikoner (Apple Computer 1992). Med subjekter menes mennesker, ting, steder etc.. Hvordan skal en for eksempel utrykke en lagrefunksjon ved hjelp av et ikon? En god løsning på dette er å inkludere tekst sammen med ikonet. Det å bruke tekst sammen med ikonene løser også en del andre problemer forbundet med ikoner. Ikoner er lette å kjenne igjen, men de kan være vanskelige å lære første gang en ser ikonet (Haramundanis 1996). En god metodikk for å lære brukeren hva ikonene betyr, kan være å vise ikoner med en beskrivende tekst. Når brukeren har lært seg betydningen av selve ikonet er ikke den hjelpende teksten nødvendig lenger. En må se på ikonet som et hint for hvilken handling eller objekt den representerer (Haramundanis 1996).

En kan si at ikoner er en del av skrivebordsmiljøet og blir brukt til å visualisere objekter og funksjonaliteter og er da en del av en større sammenheng i skrivebordsmiljøet. Hvilke teknikker kan brukes for å gi en visualiserende sammenhengen mellom ikonene? En kan da se på det som kalles en konseptuel modell og bruken av metaforer i brukergrensesnittet.

4.9.4 Metaforer og den konseptuelle modellen

4.9.4.1 Konseptuelle modeller

En konseptuel modell er en beskrivelse av det foreliggende systemet i form av integrerte ideer og konsepter (Preece 2002). Konseptene og ideene beskriver hva systemet skal gjøre og hvordan systemet skal oppføre seg. Hensikten er at de integrerte ideer og konsepter skal gjøre systemet forståelig av brukeren slik systemet var tilsiktet. En konseptuel modell lar oss forutsi effekten av våre handlinger (Norman 1998). Norman (1998) mener at uten en god modell vil brukeren ikke ha noen formening om hvilken respons eller effekt en får fra systemet. Hvis noe går galt eller noe rart skjer, trenger brukeren en dypere forståelse av systemet.

Den konseptuelle modellen er da en beskrivelse av hvordan en skal forstå systemet. Hvordan kan en beskrive den konseptuelle modellen for brukeren gjennom brukergrensesnittet? En metafor kan være en mulig løsning på dette.

4.9.4.2 Metaforer

En måte å få presentert konseptuelle modeller, er å bruke metaforer. Metaforene skal hjelpe oss til å lettere forstå brukergrensesnittet. De brukes ofte til å forklare noe som er vanskelig på en enkel måte, gjerne med relasjoner til noe en kjenner fra før (Preece 2002). Metaforene skaper noe som er kjent for brukeren og dette hjelper brukeren å lære og forstå programmet lettere.

Metaforer er ikke noe nytt som kun har blitt tatt i bruk i interaksjonsdesign. I den daglige talen bruker vi masse metaforer som er en integrert del av vårt språk (Preece 2002). Utrykk som ”vann over hodet”, ”medaljens bakside” og ”luke vekk feil” er metaforer vi bruker i vår daglige tale. I datamaskinverden ble historisk sett metaforer tatt tidlig i bruk. Det engelske ordet ”computer” var tidligere tittelen til en person som satt og gjorde beregninger (Marcus 1998). Dette ordet ble brukt om datamaskinen siden den ble opprinnelig laget for å gjøre de beregningene som en ”computer” gjorde. I dagens brukergrensesnitt brukes en rekke metaforer. Det bokstavbaserte brukergrensesnittet, som kommandogrensesnitt, kan trekke referanser tilbake til fjernskriveren. Det grafiske brukergrensesnitt vi kjenner i dag, bruker metaforer til skrivebordet og kontormiljøet representert med mapper, filer, skrivebordsbakgrunn, søppelbøtter etc.. I brukergrensesnittet kan en ofte ta i bruk metaforer for å beskrive avanserte logikk eller funksjoner på en enkel måte. Det kan for eksempel sees på som en vanskelig oppgave å forklare hvordan Internett fungerer uten å bruke metaforer (Preece 2002).

En kan klassifisere metaforer i to nivåer, overordnet metaforer og individuelle metaforer (Marcus 1998). Overordnede metaforer gir en basis for hvordan forstå et system eller et konsept. En skrivebordsmetafor kan sees på som overordnet metafor. Den overordnede metaforen kan da hjelpe brukeren i å forstå en konseptuel modell. Det Marcus (1998) kaller individuelle metaforer bruker referanser til spesielle kontrollfunksjoner eller datafremstillinger, som kan minne om det Beaudouin-Lafon (2000) beskriver som instrumenter i instrumental interaksjon (kapittel 4.8). En radioknapp i et brukergrensesnitt kan sees på som en individuell metafor. Radioknappen referer til en logisk ”og/eller” funksjon, hvor en kun kan ha ”inne” en knapp av gangen (Marcus 1998). Dette ligner da på knappene til en gammel radio, hvor bare en knapp kan være inne av gangen.

De fleste metaforer fokuserer på å lage en likhet mellom det som er kjent for brukeren og det brukergrensesnittet brukeren blir presentert. Mye av suksessen til dagens skrivebordsmetafor er at en kan se klare likheter mellom brukergrensesnittet og kjente objekter i virkeligheten, objekter som mapper, dokumenter og filer. En kan si at metaforene i skrivebordmiljøet oppfører seg mye likt deres originale objekter, men aldri helt likt. Dette gjør at en kan snakke om ”magien” til en metafor (Kim Halskov 2000). En kan ofte ønske at en metafor gjør mer enn hva den gjør i virkeligheten. For eksempel mappemetaforen i et skrivebordsmiljø ønsker vi at skal være mer effektiv, enn en mappe i virkeligheten. Hvis en ønsker å sortere innholdet i en mappe etter dato er dette enkelt i en mappe på en datamaskin, mens i virkeligheten blir dette en mer tidkrevende oppgave hvis mappen i den virkelige verden er stor. Vi ønsker å dra nytte av fordelene ved å bruke en datamaskin. Dette gjør at metaforene ikke må ekskludere denne ”magien” en får ved å bruke en datamaskin. Hovedpoenget er at det er mange likheter og ulikheter. Men en kan dra nytte av likhetene så lenge en også vet om

ulikhetene. Vi prøver å lage brukergrensesnitt som skal virke kjente for brukerne, men vi må også bruke de nye mulighetene som datamaskinen tilbyr.

4.9.4.3 Fordeler, ulemper og problemene med metaforer

Hvis en ser på fordelene ved bruk av metaforer, finner en to klare fordeler. En fordel ved å bruke konsepter som er kjent for brukeren, kan en oppnå å lage brukergrensesnitt hvor behovet for opplæring blir mindre. Kjennskapet til et annet domene kan overføres og brukes slik at det skapes en gjenkjennelseseffekt for brukeren (Marcus 1998). I praksis betyr dette at ved hjelp av metaforer klarer en å skape objekter i brukergrensesnittet som brukeren har en relasjon til fra før. Blir brukeren presentert for eksempel en mappe, vil de brukerne som kjenner en mappe fra før forstå at denne kan brukes til å ha filer i. Brukeren får da hjelp av metaforen til å forstå en konseptuel modell. Den andre store fordelen ved bruk av metaforer er at en får konsepter som passer sammen i utgangspunktet. Skrivebordet passer sammen med mapper og mapper passer sammen med filer (Laurel 1993).

Disse to fordelene har også sine ulemper. Når en bruker metaforer, skaper en forventninger til at det som blir representert fungerer på samme måte som i virkeligheten. Ideen med metaforer bygger på å representere virkelige objekter slik at brukeren vet hva en skal gjøre med dem (Laurel 1993). Problemet med dette er at metaforene har forskjeller fra den virkelige verden. Brukeren får problemer når en ikke nøyaktig vet hva som er forskjellen (Laurel 1993). En metafor ligner på det vi har i den virkelige verden, når noe ligner har det også forskjeller. For eksempel finnes det ikke et skrivebord i den virkelige verden som er likt det skrivebordet en ser i skrivebordsmetaforen. Videre har for eksempel ikke mapper i skrivebordsmetaforen noen vekt, de gir deg heller ikke papirkutt i fingeren (Laurel 1993). Problemet med dette er at metaforene er like virkeligheten, men de er også forskjellige.

Et annet problem med metaforene er at en fort kan ”dette av” metaforen. En metafor har som tidligere nevnt en del fordeler med at objekter passer sammen i utgangspunktet (Laurel 1993). Problemet oppstår når et objekt ikke passer sammen med resten. Et eksempel på dette finner en i den tidligere beskrevet søppelbøtten i Apple sin skrivebordsmetafor. Apple bruker her søppelbøtten til to ting. Det ene er å kaste filer en ikke vil ha. Det andre er at når drar en disk eller cd-plate i søppelbøtten løser den heller ut cd-platen fremfor å slette den. I dette tilfellet passer ikke den virtuelle søppelbøtten med brukerens forestilling av hvilken funksjonalitet søppelbøtten har. Den virtuelle søppelbøtten har mer funksjonalitet enn den i virkeligheten har (Laurel 1993). En kan registrere at Apple forandret brukergrensesnittet sitt på dette området. I dag vil søppelbøtteikonet bli forandret til en pil, for å illustrere at en ”løser ut” mediet i stedet for å slette det.

Dette er med på å forandre fokuset for hva en vil oppnå ved bruken av en metafor. En kan se på metaforen som et hjelpemiddel til brukeren, for å lage en modell for forståelsen av datamaskinen. Metaforen brukes ikke nødvendigvis til å forklare datamaskinens spesifikke funksjonalitet. Metaforen kan da benyttes til å lage konsepter som gjør at objekter passer sammen og ikke nødvendigvis til å forklare all funksjonaliteten som tilgjengelig i brukergrensesnittet.

4.10 Oppsummering av interaksjonsdesign

I dette kapittelet har jeg beskrevet en rekke teorier innenfor interaksjonsdesign. Jeg har beskrevet hvordan Norman (1998) bruker begrepet mapping for å forklare relasjonen mellom objekter som representerer en form for styring eller kontroll, og resultatet en får ved bruk av dette styringsobjektet. Norman (1998) bruker dette begrepet for å binde sammen objektet en utfører en handling med, og resultatet en får fra selve handlingen. Videre har jeg forklart Norman (1998) sitt frembydelses begrep. Dette begrepet brukes for å beskrive den informasjonen brukeren får gjennom visualiseringen av objekter en bruker for å gjennomføre en handling. Frembydelsen er da med på å gi oss hint om hvordan objektet skal brukes (Norman 1998). Norman (1999) beskriver videre at ved bruken av dette begrepet i brukergrensesnittet på datamaskiner, må en se på det Norman (1999) kaller oppfatet frembydelse. Oppfattet frembydelse bygger på lærte konvensjoner, i stedet for en intuitiv oppfattelse av et objekts utforming og funksjon.

En interaksjonsmodell kan brukes til å beskrive brukerens interaksjon med et dataprogram. Jeg har beskrevet Norman (1998) sin eksekverings-evalueringsløkke, som kan sees på som en interaksjonsmodell. I denne modellen bestemmer brukeren seg for å utføre en handling, for å så finne ut hvordan denne kan gjennomføres i brukergrensesnittet. Brukeren gjennomfører så handlingen og mottar en tilbakemelding fra systemet. Denne modellen beskriver en interaksjon som en stor syklus. Som en motsetning til denne måten å se en interaksjon på, har Djajadiningrat et al. (2002) sine begreper operasjonsformål og innebygd respons. Disse begrepene knytter interaksjonen nærmere visualiseringen av objektene. Hvert objekt har en interaksjon i seg selv. Dette gjør at en hel handling kan deles opp i flere sykluser, eller små interaksjoner.

En slik interaksjon i flere små sykluser brukes i Shneidermann (1998) sitt direkte manipulasjons begrep. Dette begrepet beskriver en interaksjon hvor brukeren kan manipulere direkte på objekter i brukergrensesnittet. Brukeren vil da få en kontinuerlig tilbakemelding fra brukergrensesnittet. Direkte manipulasjon erstatter et mer komplekst kommandospråk og kan bidra til at brukergrensesnittet er lettere å lære og forstå.

Holland og Oppenheim (1999) sitt direkte kombinasjonsbegrep er en utvidelse av direkte manipulasjonsbegrepet. Holland og Oppenheim (1999) beskriver at en av ulempene ved direkte kombinasjon, er at en ofte må bruke dialogbokser. En dialogboks låser brukeren til en interaksjon og kan da sees som lite fleksibelt. Holland og Oppenheim (1999) beskriver da direkte kombinasjon som en løsning på dette problemet. I direkte kombinasjon knytter en sammen to eller flere objekter og tilfører en operasjon til disse objektene. Brukeren vil se hvilke operasjoner som er tilgjengelig for denne kombinasjonen og slipper da å lete seg frem i brukergrensesnittet for å finne riktig operasjon.

Beaudouin-Lafon (2000) bruker prinsippet instrumental interaksjon for å utvide og generalisere direkte manipulasjonsbegrepet. I instrumental interaksjon opererer brukeren på instrumenter som igjen opererer på objektene sine parametere. Dette prinsippet er da nyttig til å kategorisere og analysere direkte manipulasjonsgrensesnitt.

Jeg har så beskrevet noen teknikker som brukes for å visualisere funksjonalitet i brukergrensesnittet. Jeg har beskrevet hvordan kommandolinjen mangler en måte å visualisere funksjonalitet for brukeren. Videre har jeg beskrevet hvordan skrivebordsmiljøet og WIMP- brukergrensesnittene visualiserer funksjonaliteten for brukeren. Jeg har også sett på problematikk rundt bruken av ikoner i brukergrensesnittet. Haramundanis (1996) mener da at et ikon må stå sammen med en tekst for å ha full betydning. Ikonenes beste egenskap er da kanskje å skille mellom funksjoner, ikke nødvendigvis å forklare dem. Jeg har også beskrevet hvordan en kan bruke en metafor til å beskrive en konseptuel modell. Utfordringene med en metafor er å beskrive for brukeren hva som er forskjellig fra objektet i den virtuelle verden med objektet i den virkelige verden. En metafor egner seg da kanskje best til å beskrive et konsept og ikke nødvendigvis funksjonalitet.

Dette teoretiske rammeverket skal jeg da bruke til å analysere skisser som er et forslag til et mulig brukergrensesnitt til et digitalt loggbokprogram. Denne diskusjonen blir beskrevet i kapittel 6.

5 Eksisterende programmer

I dette kapittelet vil jeg beskrive hvordan andre programmer har visualisert sin funksjonalitet. Alle disse programmene er programmer som kan kjennetegnes ved at de kan synkroniseres med eksterne enheter og behandle og lagre dataene som vil bli synkronisert. Disse programmene vil bli brukt som et grunnlag for generering av skissene i kapittel 6. Programmene vil bli gjennomgått etter kriteriene som er beskrevet i neste kapittel.

5.1 Kriteriene for gjennomgang av programmene

I gjennomgangen av programmene er det 5 spørsmål som er kriteriene for gjennomgangen. Spørsmålene har tatt utgangspunkt i hvilke funksjonaliter en ønsker fra et digitalt loggbokprogram. Disse spørsmålene har da en tilknytning til de generelle behovene en kan ha til en digital loggbok, som er beskrevet kapittel 3.3.1.

Kriteriene i form av 5 spørsmål, er som følger:

- Hvordan er synkroniseringen med en ekstern enhet visualisert i brukergrensesnittet?

- Når brukeren legger inn nye data i brukergrensesnittet, hvordan er dette visualisert?

- Hvis data som er loggført i programmet kan kategoriseres, hvordan er dette visualisert?

- Ved en analyse av data i programmet, hvordan er analysene organisert og hvilke valgmuligheter har brukeren for å spesifisere analysene?

- Hvor lett er det for brukeren å forandre eksisterende data, hvordan foregår gjennomføringen av en sikkerhetskopiering av dataene i programmet?

Resultatet av denne gjennomgangen av programmene med disse kriteriene, vil da være en beskrivelse av hvordan programmene har visualisert funksjonaliteten. Disse beskrivelsene vil da være grunnlaget for hvordan skissene blir generert.

5.2 Diabetes programmer

I dag finnes det noen programmer som gir en diabetiker mulighete til å loggføre sine data digitalt. Produsentene av blodsukkerapparatene lager egne programmer som er tilpasset deres egne produkter. Disse programmene vil da kun fungere på produsentens egne blodsukkerapparater. For å overføre dataene må en kjøpe en spesiell kabel og programvare i tillegg til blodsukkerapparatet. Det finnes også frittstående produsenter av programvare som ikke produserer blodsukkerapparater selv. Noen av disse programmene støtter synkronisering med en rekke blodsukkerapparater uavhengig av produsent, men det er også noen som ikke støtter

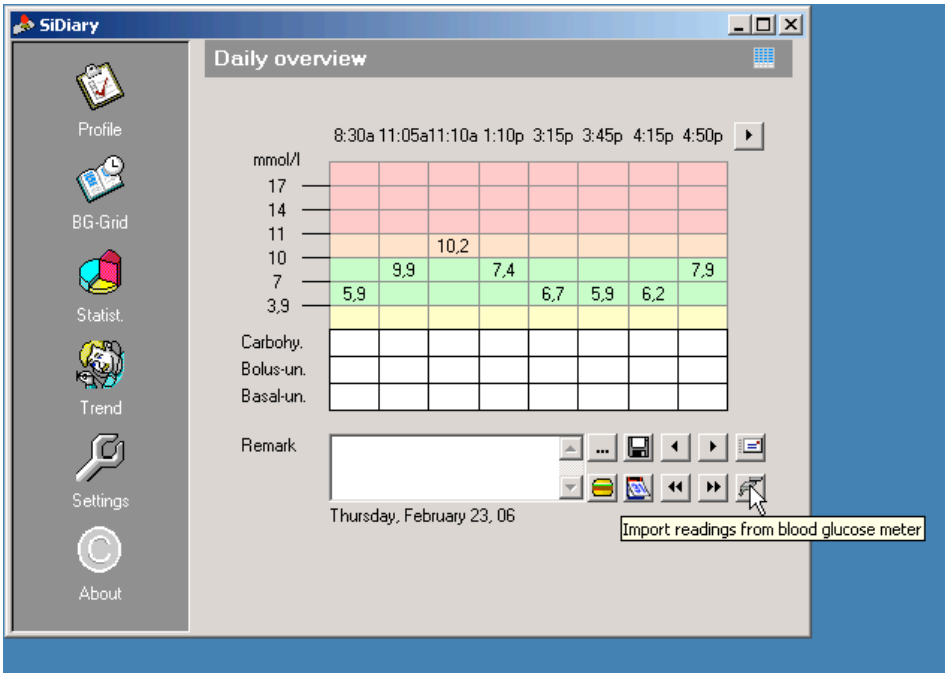
noen form for synkronisering med et blodsukkerapparat. En som bruker disse programmene må da legge inn alle data manuelt.

Jeg har sett på et lite utvalg av slike programmer som brukes i denne oppgaven. Jeg har sett på programmer fra blodsukkerapparatprodusenter og ett frittstående program. Programmene heter ”OneTouch™ Diabetes Management Software”, ”Accu-Check® Compass” og det frittstående programmet ”SiDiary - Diabetes Management-Software”. Det frittstående programmet har mulighet til å synkronisere med en rekke blodsukkerapparater.

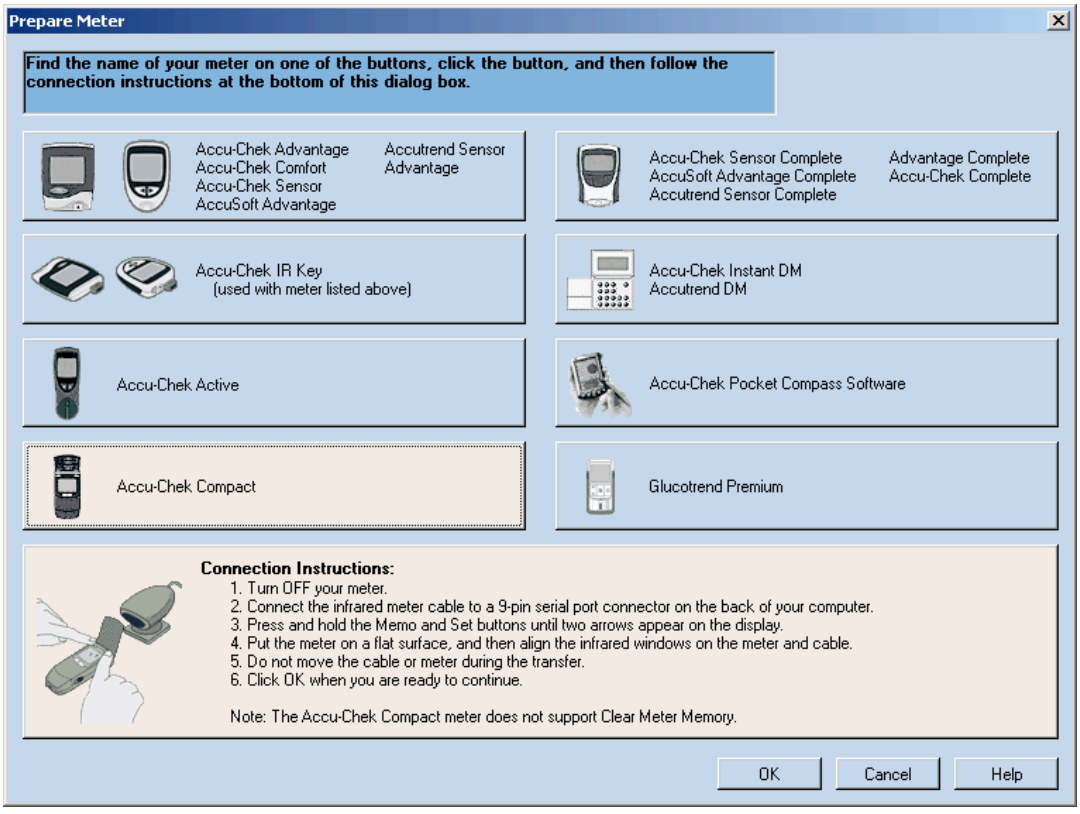
Programmene ”OneTouch™ Diabetes Management Software” og ”Accu-Check® Compass” har blitt valgt siden de er produsert av to av de største produsentene av blodsukkerapparater på markedet i Norge. Accu-Check® sin programvare blir solgt på de fleste apoteker i Norge mens OneTouch™ sin programvare blir solgt direkte fra importøren. Programvaren ”SiDiary - Diabetes Management-Software” er produsert av en produsent som ikke selger blodsukkerapparater. Dette er da en programvare som er blitt spesialisert for å fungere mot en rekke forskjellige blodsukkerapparater. Programmet er tatt med i analysen siden produsenten av programmet kun produserer programvare for blodsukkerapparater og ikke apparatet selv. Produsenten kan da sees som en uavhengig leverandør i forhold til blodsukkerapparatene.

5.2.1 Synkronisering

Programmene kan synkroniseres med en egen kabel som er tilleggutstyr til blodsukkerapparatet. Formålet med synkroniseringen er å overføre data til programmet på datamaskinen, slik at det videre kan analyseres. For å synkronisere må brukeren trykke inn et ikon eller velge et menyvalg (Figur 5). Da vil brukeren få opp en egen dialogboks som hjelper brukeren videre i prosessen. Denne dialogboksen gir brukerne mulighet til å velge hvilket blodsukkerapparat som skal synkroniseres og andre innstillinger i forbindelse med synkroniseringen. Videre er det ofte en tekstlig beskrivelse av hva brukeren skal foreta seg før selve prosessen starter (Figur 6). Denne beskrivelsen forteller brukeren at kabelen skal kobles til og eventuelle tastetrykk en må foreta seg på periferienheten, eller blodsukkerapparatet. Spesielt i Accu-Check® sin programvare er det viktig å lese den tekstlige beskrivelsen, da denne er forskjellig fra apparat til apparat. OneTouch™ sin programvare skiller mellom blå og grå apparater. Eksempelvis skal de grå apparatene være på under synkroniseringen, i motsetning til de blå som skal være skrudd av under synkroniseringen.



Figur 5: For å komme til synkroniseringsdialogen må en klikke inn ikonet for importering. En hjelpetekst kommer opp for å beskrive ikonet ytterligere.

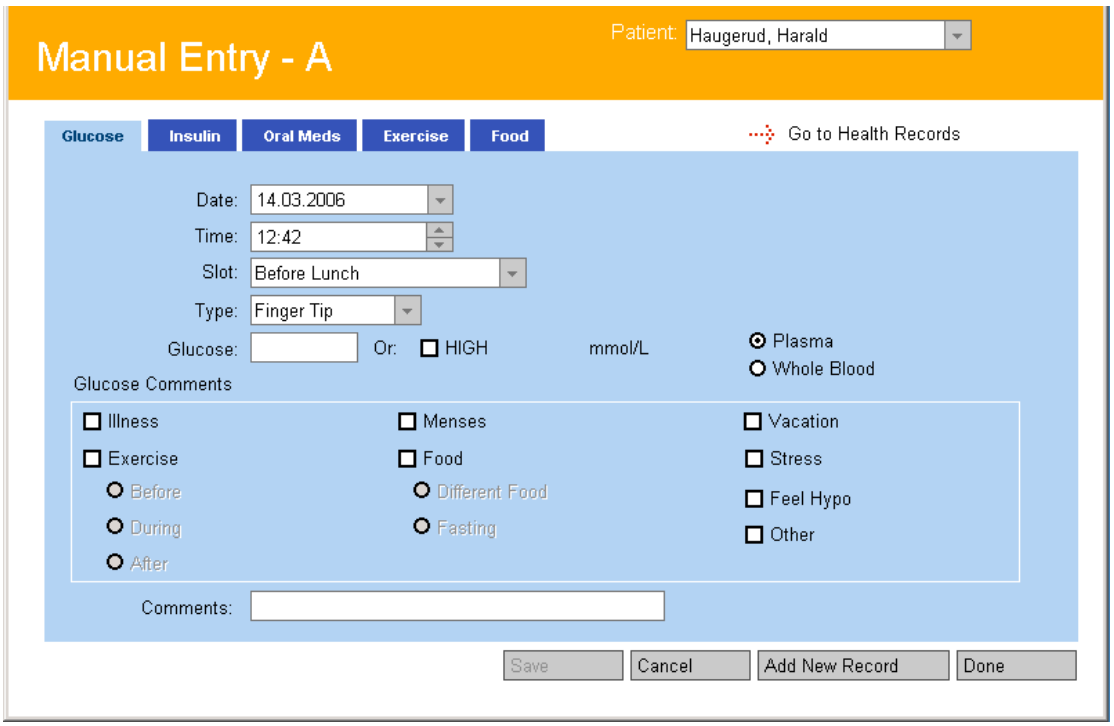


Figur 6: Dialogboksen og en tekstlig instruksjon for hva en skal gjøre i forbindelse med blodsukkerapparatet. Hentet fra Accu-Check® sin programvare.

5.2.2 Loggføring av data

Programmene støtter kun importering av data fra blodsukkerapparatene. Dette betyr at andre relevante data må legges manuelt inn i programmet. Alle programmene jeg har sett på tilbyr dette. Det er samtidig en forskjell mellom Accu-Check® og OneTouch™ sin programvare og SiDiary sin programvare med tanke på hvordan den grafiske fremstillingen av denne funksjonaliteten er visualisert.

Accu-Check og OneTouch har løst dette ved å lage et eget dialogvindu for inntastingen av data. Brukeren får opp dialogvinduet ved å klikke på en "legg inn data manuelt" knapp. I dialogvinduet kan brukeren legge inn data som blodsukkerverdier, insulindoser, medisiner, trening og måltider. Videre kan disse verdiene spesifiseres mer detaljert. En kan for eksempel i OneTouch™ sin programvare knytte egne standardiserte eller tekstlige kommentarer til en blodsukker måling. Eksempel på slike kommentarer er sykdom, trening før eller etter måling, spist annen mat enn det en er vant til, føling, stress eller for eksempel vaksine. Det er viktig å merke seg at her legger en kun kommentarer til en blodsukkermålig. I OneTouch™ er det en egen fane for å legge inn verdier om mat. Her kan en da legge inn verdier rundt selve måltidet.



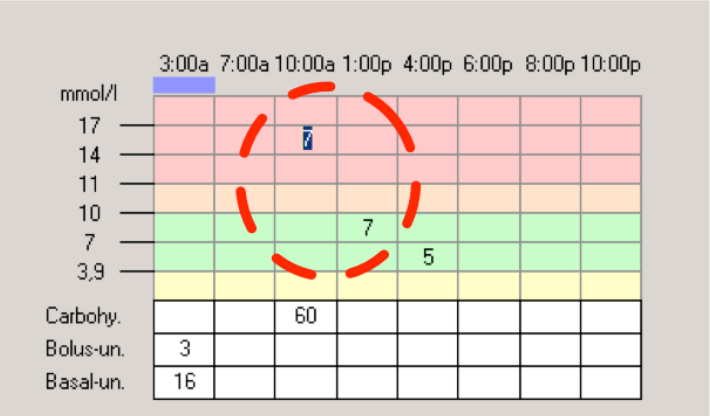
Figur 7: Et dialogvindu for å legge inn data manuelt i OneTouch™ sin programvare. Under de forskjellige fanene kan en legge inn data mer spesifisert. Her er et eksempel på kommentarer til en blodsukkermåling.

Accu-Check® og OneTouch™ sin programvare kan også tilordne data til en spesiell "time block" eller en egen kategori. I en egen nedtrekks meny kan en velge tidsintervaller, som før frokost, etter forkost, før lunsj etter lunsj, etc..

SiDiary skiller seg litt ut i fra de andre når det gjelder å legge inn data. SiDiary viser ikke en egen dialogboks for å legge inn data. Brukeren kan i stedet manipulere direkte på en tabell. Dette er samme tabellen som kan brukes til å analysere eller få oversikt over sine egne verdier for en dag. Denne tabellen har en x- og y-akse. X-aksen representerer tiden av døgnet og deler av y-aksen representerer en blodsukkerverdi.

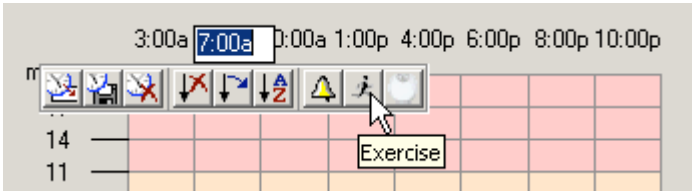
Nederst på y-aksen er det plass til å notere verdier rundt karbohydrater og insulininntak. Det deles da opp i to insulin typer. Videre er det fargekoder i noen av cellene, med henholdsvis fargene rødt, grønt og gult. Disse er med på å symbolisere hvilke blodsukkerverdier som er innenfor målsettingen en har satt opp i preferansene. For å skrive inn en blodsukkerverdi eller en av de andre verdiene, klikker en på riktig celle og taster inn en verdi.

En skal være klar over at hvis en taster inn en blodsukkerverdi i feil celle i forhold til y-aksen, vil programmet automatisk flytte på denne verdien (Figur 8). Hvis en for eksempel taster inn verdien 7 i intervallet 11 til 14 vil tallet bli flyttet ned til riktig intervall etter at brukeren er ferdig og klikker et annet sted i programmet.



Figur 8: Hvis en bruker feil celle til å taste inn en verdi, vil programmet automatisk flytte tallet ned til riktig celle i forhold til y-aksen. Firkanten under ”3:00a” markerer at en har lagt inn data ang. trening. Hentet fra programmet SiDiary.

Ønsker en å legge til data utenom de tidskolonnene som er tilgjengelige, må en legge til en ny kolonne med riktig tidsverdi. SiDiary skiller seg litt ut fra de andre programmene ved at det ikke direkte støtter å legge inn data med vilkårlig tid. Ønsker en å legge inn data ang. trening må en gjøre dette gjennom et skjult menyvalg som kommer frem når en klikker på klokkeslettallene ovenfor kolonnene. Når en klikker på tallene kommer det da frem en egen meny bestående av ikoner. Denne menyen brukes til å legge inn data som trening og den brukes til å organisere tidskolonnene. En kan for eksempel legge til, slette og sortere kolonner.

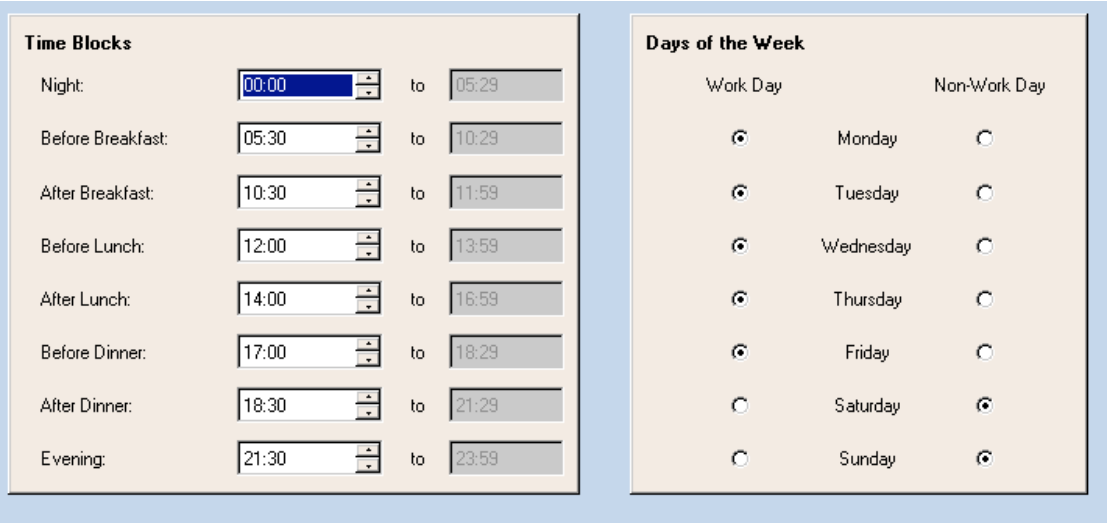


Figur 9: En skjult meny med ikoner kommer frem når en klikker på klokkeslettallene. Hentet fra programmet SiDiary.

5.2.3 Kategorisering av data

Kategorisering av data er noen en trenger å gjøre for at det skal bli lettere å analysere dataene. En kan ønske å for eksempel kategorisere dataene etter før og etter måltidene

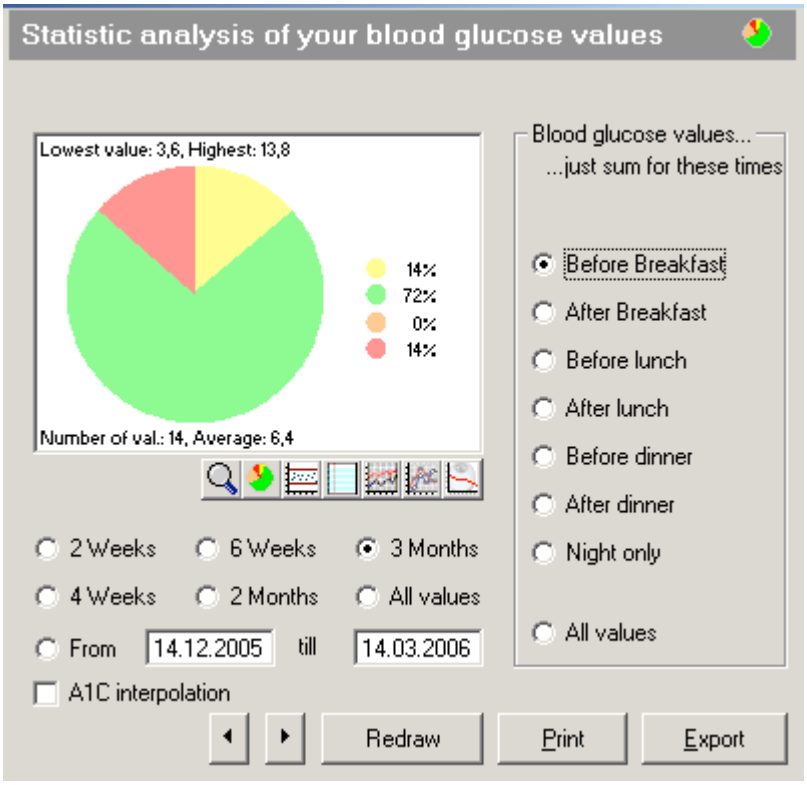
en har i løpet av et døgn. Alle programmene tilbyr en form for kategorisering av dataene. Accu-Check® og OneTouch™ er ganske like ved å tilby dette ved registreringen av data. En kan da velge i en nedtrekksmeny hvilken kategori dataene skal tilhøre. I alle programmene er kategoriene bestemt i en egen preferansedel av programmet (Figur 10). Der stiller en inn hvilke klokkeslett som er innenfor kategoriene. En kan da for eksempel sette at "før frokost" er fra kl 06.00 til klokken 09.00 og "før middag" er fra klokken 14.00 til 17.00. Videre tilbyr Accu-Check® og OneTouch™ programmene at en kan skille mellom arbeidsdager og fridager. En kan da sette andre tider for kategoriene i helger og andre fridager. Dette gjør at programmene kan tilpasses til en viss grad den enkeltes behov. Data som blir importert direkte fra blodsukkerapparatet vil da bli kategorisert etter det ferdige oppsettet.



Figur 10: Kategoriseringen av tider for måltider i programmet til Accu-Check®. En kan også velge hvilke dager som ikke er arbeidsdager, for å kunne personalisere programmet etter egne behov.

5.2.4 Analysering av data

Alle programmene har en form for analysering av data (Figur 11). Disse vises gjerne i form av forskjellige typer grafer som kakediagrammer, histogrammer, kurvediagrammer etc.. Det er også vanlig at en kan vise de forskjellige grafene med utgangspunkt i et bestemt tidsrom eller egne kategorier. Det er da mulig å for eksempel få et kakediagram over blodsukkermålinger de 4 siste ukene. Videre kan en også begrense seg til å for eksempel kun se målinger foretatt før frokost.

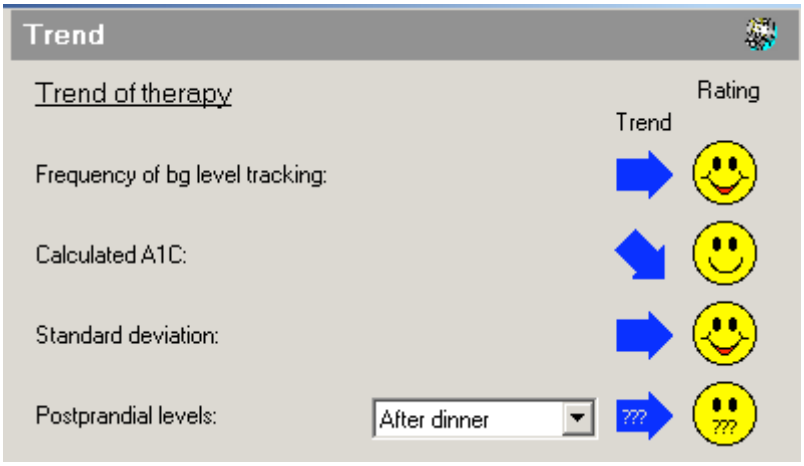


Figur 11: Et skjermbilde fra SiDiary hvor en får presentert en analyse av dataene. Fra ikonmenyen kan en velge forskjellige presentasjonsformer som punktgraf, linjegraf etc.. En kan også sette premisser for analysen ved hjelp av radioknappene.

Poenget med disse målingene er å se trender i sin egen behandling. Tar en for eksempel for mye insulin til frokost, vil slike grafer være et nyttig verktøy for å se dette. Programmene tilbyr da mange forskjellige statistikker for å fortelle deg hvilke målinger eller verdier som er innenfor eller utenfor en målsetting en selv har satt opp. En målsetting kan typisk være at blodsukkermålinger skal være innenfor verdien 4 (mmol/L) til 8-10 (mmol/L). Programmene utfører mange forskjellige statistiske tester over målsettingene over dataene som er loggført og viser resultatene enten som grafer eller rent tekstlig. En typisk måte å vise dette på er å vise prosentvis hvor mange målinger som er innenfor eller utenfor målsettingen.

Programmet til SiDiary tilbyr også en egen trendoversikt (Figur 12). Her får en oversikt over verdier som frekvensen av blodsukkermålinger, kalkulert langtidsblodsukker (HbA1c), standardavvik eller verdier etter måltider. Videre visualiseres disse verdiene med piler og sure eller smilende ansikter. Pilene og ansiktene er med på å generalisere verdiene slik at en ser trendene tydeligere. Pilen forteller brukeren om behandlingen har en positiv eller negativ trend. Eksempelvis peker pilen nedover hvis blodsukkeret er på vei til å bli dårligere. Ønsker brukeren en mer detaljert beskrivelse kan brukeren holde musepekeren over pilen, og det kommer opp en tekst som beskriver situasjonen nærmere. Ansiktene er med på å si noe om situasjonen på det nåværende tidspunktet. For eksempel betyr en smilende munn at situasjonen er bra og en smilende åpen munn betyr at behandlingen er utmerket. Brukeren kan selv velge hvilke verdier som skal kobles med de forskjellige utrykkene.

En kan si at statistiske tallverdier her er mappet med en grafisk representasjon. Forandringen i verdien er mappet med en pil, og selve verdien er mappet med et ansikt.



Figur 12: En oversikt over trender i behandlingen. Pilene representerer forandringer i behandlingen og ansiktene representerer status i behandlingen.

5.2.5 Sikkerhet og sikkerhetskopier

Sikkerheten i programmet beskriver hvor enkelt det er for brukeren å forandre data uønsket. Generelt kan en si at dataene som er loggført ikke skal kunne forandres, da dette er et bilde av virkeligheten. Hvis brukergrensesnittet tilbyr brukeren å lett forandre sine egne data vil dette kunne føre til at brukeren kan trekke feil konklusjoner. En annen faktor er at brukeren må kunne ta sikkerhetskopier av sine egne data, da det kan være kritisk for behandlingen å miste slike data.

Programmene oppfører seg forskjellig i forbindelse med hvor lett det er å forandre eksisterende data. I Accu-Check® sin programvare det vanskelig å forandre data som allerede er loggført. En har ikke mulighet til å forandre en verdi i data som allerede er loggført. Skal en forandre for eksempel en blodsukkerverdi må en slette hele oppføringen, for å så legge inn en ny oppføring med den forandrede blodsukkerverdien. En kan altså ikke gå inn i en oppføring for å kun endre en blodsukkerverdi. Da må en slette alle dataene i hele oppføringen som dato, kommentarer og andre verdier en har loggført sammen med blodsukkerverdien. I OneTouch™ sin programvare finnes det en egen ”rediger”-knapp som lar brukeren forandre verdien i en dataoppføring. Brukeren slipper da å slette en hel oppføring for så å lage en ny som nesten er identisk.

I SiDiary sin programvare er det enklere for brukeren å forandre allerede loggførte verdier. Der kan brukeren direkte klikke på en verdi i den totale oversikten, for å så forandre verdien til noe annet. Brukeren slipper da å gå via en redigeringsdialog eller andre funksjoner for å forandre verdiene i en oppføring.

Accu-Check® har et eget menyvalg for å lage en sikkerhetskopi av alle dataene som er loggført. En får tilgang til denne funksjonen ved hjelp av et valg i en nedtrekksmeny. Brukeren får da opp en egen dialogboks hvor en velger hvor en egen fil skal lagres med alle dataene. For å opprette den sikkerhetskopierte databasen igjen,

kan brukeren velge denne filen i et annet menyvalg i programmet. Sikkerhetskopien består da av en fil hvor alle dataene er lagret.

OneTouch™ sitt program har en liknende funksjon, men her kan brukeren velge hvilke deler av databasen som skal lagres. Brukeren velger da en ”fra dato” og en ”til dato”. Programmet lagrer da alle dataene innenfor disse datoene til en egen fil. Denne filen kan brukes til å importere data tilbake til databasen. SiDiary tilbyr ikke en sikkerhetskopifunksjon til brukeren.

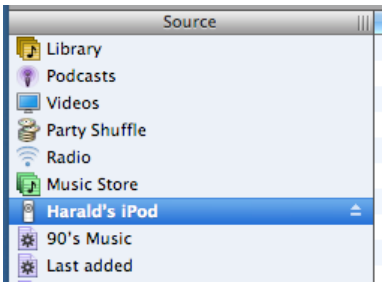
5.3 Programmer i andre brukssituasjoner

I dette kapittlet vil jeg vise hvordan andre programmer i liknende bruksitasjoner visualiserer liknende funksjonalitet. Programmene som vil bli beskrevet er iTunes, iCal og Aperture. Alle programmene er produsert av Apple Computer. Disse programmene er valgt for de reiser flest interessante spørsmål i forbindelse ved visualiseringen av liknende funksjonaliteten.

5.3.1 Synkronisering av iPod i iTunes

Apple Computer har laget et program som brukes i forbindelse med den bærbare musikkavspilleren iPod. ITunes er et program som holder orden på alle musikkfilene en har på maskinen sin. Programmet er også mer enn en et vanlig musikkavspillingsprogram. ITunes kan sees på som et bibliotek over alle musikkfiler en har på sin datamaskin. I tillegg til å være et bibliotek tilbyr iTunes mye forskjellig funksjonalitet. Eksempler på slik funksjonalitet er synkronisering med en iPod, importering av musikk fra cd, brenning av musikk til cd, kjøpe musikk over internett, lytte til nettradioer, høre på ”podcasts”, søking etter sanger eller for eksempel lage egendefinerte spillelister.

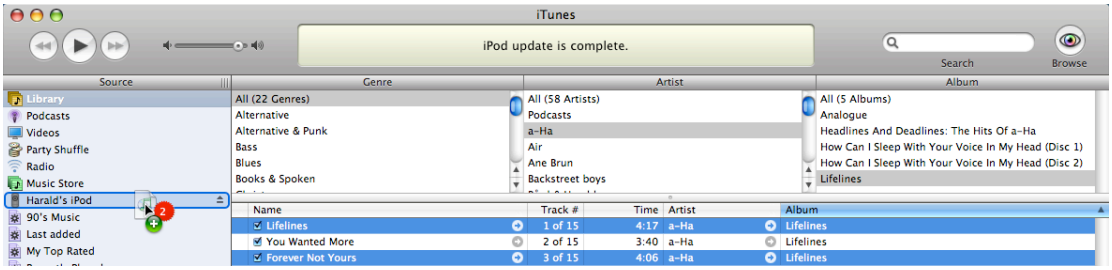
Når en kobler en iPod til datamaskinen, visualiserer iTunes dette ved å vise brukeren et ikon i kildelisten i iTunes. Kildelisten er en liste som gir brukeren mulighet til å velge hvilke kilde en skal utforske (Figur 13). Kilder kan være biblioteket som består av musikkfiler, podcasts, musikkbutikk eller radiokanaler etc.. I denne kildelisten blir et ikon av selve iPoden presentert når en iPod kobles til maskinen. Ikonet som brukes er avhengig av hvilken iPod en kobler til maskinen. Det finnes mange forskjellige typer iPoder på markedet med forskjellig utseende. Ikonet viser da et utsende som stemmer med den iPoden en har i virkeligheten. Kobles for eksemepel en sort iPod til maskinen, vises en sort iPod som ikon. ITunes representerer da iPoden gjennom et ikon som ligner på den fysiske iPoden og en tekstlig beskrivelse.



Figur 13: Kildelisten i iTunes. Her velger brukeren hvilken kilde en vil utforske. Pilen ved siden av iPod-ikonet, løser iPoden ut slik at den trykt kan fjernes.

Synkroniseringen med iPoden kan foregå på forskjellige måter med iTunes. Dette er litt avhengig av hvilken type iPod en har og hva brukeren har satt som preferanser i oppsettet i iTunes. En kan skille mellom to grunnprinsipper ved synkronisering. IPoden oppdaterer seg slik at innholdet på iPoden er identisk med innholdet i iTunes eller brukeren bestemmer selv hvilke sanger som skal ligge på iPoden. Velger brukeren at alt innhold i iTunes skal overføres til iPoden, kan en sette iTunes til å automatisk overføre eller synkronisere alle data når iPoden kobles til maskinen.

Hvis ikke brukeren ønsker at iPoden skal oppdateres automatisk må en gjøre velge selv hvilke filer som skal overføres. Brukeren må da velge hvilke sanger brukeren ønsker og overføre til iPoden. Dette gjøres ved at brukeren velger biblioteket eller andre kilder. Brukeren vil da kunne utforske den valgt kilden i et eget vindu i programmet. Der vil sangene bli representert gjennom forskjellige lister. Brukeren belyser og drar da sangene over på iPodikonet for å overføre sanger til iPoden (Figur 14). Det kommer da et pluss tegn og et tall opp ved siden av musepekeren. Dette illustrerer at sangene blir kopiert og ikke flyttet, tallet illustrerer hvor mange sanger som blir overført.

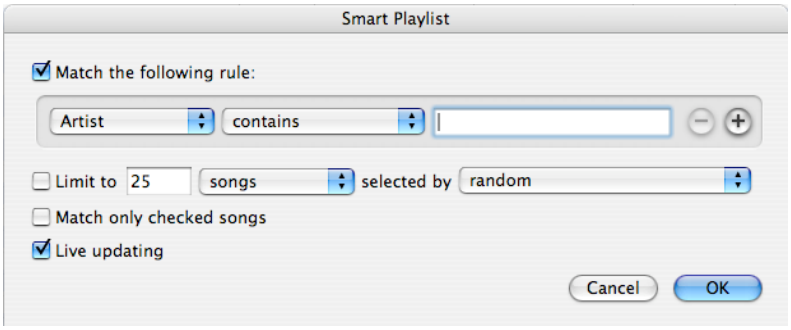


Figur 14: Overføring av sanger til iPod manuelt. Brukeren drar sangene en vil ha over selve iPod-ikonet.

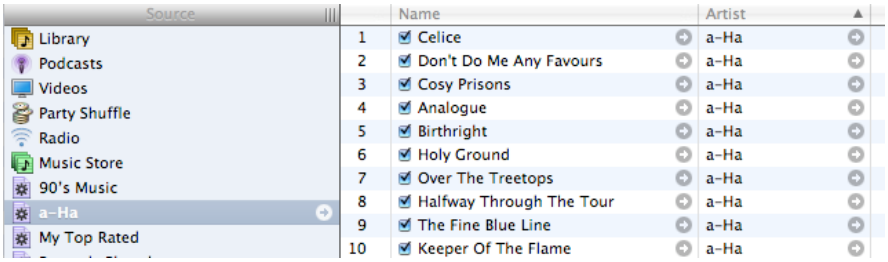
5.3.2 Smarte spillelister i iTunes

ITunes tilbyr en funksjonalitet som heter smarte spillelister. En vanlig spilleliste er en samling av sanger fra biblioteket. En spilleliste kopierer ikke sangene, men referer til de. Brukeren kan da bestemme i hvilken rekkefølge sangene skal spilles, uten å måtte flytte selve filene i riktig rekkefølge. I iTunes er det tilføyd en ny funksjonalitet til spillelistene, som da har fått navnet smarte spillelister. En smart spilleliste settes opp til å utføre et søk hver gang en viser spillelisten. Gjennom å definere forskjellige regler for den smarte spillelisten, defineres hvilke sanger som skal vises. Et dialogvindu vises når en lager en ny smart spilleliste, der kan brukeren taste inn forskjellige premisser som må stemme (Figur 15). Ønsker for eksempel en bruker å lage en liste som automatisk viser alle a-ha sangene som ligger i biblioteket i iTunes,

lages en spilleliste med en regel hvor teksten ”a-ha” må være en del av artistnavnet. Brukeren kan da klikke på denne spillelisten fra det tidligere nevnte kildevinduet og brukeren vil få opp alle a-ha sangene i bilioteket i iTunes (Figur 16).



Figur 15: Dialogvinduet hvor brukeren kan lage egne regler for hvilke sanger som skal vises i den smarte spillelisten.



Figur 16: En smart spilleliste i aksjon. Brukeren har da laget en spilleliste som viser alle a-ha sangene i biblioteket i iTunes.

Denne typen smarte spillelister kan kobles med en analyse av data. En smart spilleliste utfører et søk over alle dataene i programmet og presenterer de for brukeren. En analyse i et program for diabetikere kan også sees på som et søk i dataene hvor resultatet visualiseres som en graf eller liknende.

I iTunes må brukeren editere parametrene for det smarte albumet i et dialogvindu. Programmet Aperture, som blir beskrevet i neste delkapittel, har også en slik smarte mapper funksjonalitet, men i Aperture heter det smarte album. I Aperture har en også et dialogliknende vindu, men dette er allikevel visualisert på en annen måte.

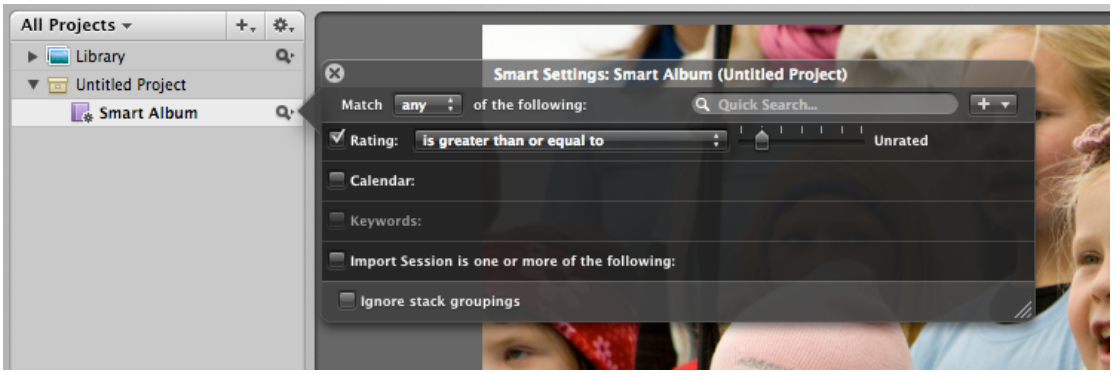
5.3.2.1 Visualisering av parameter i Aperture

Aperture er et bildeprogram laget av Apple Computer. Programmet importerer bildefiler som er tatt med digitalkamera. Videre kan brukeren redigere, behandle, kategoriserie, sortere, sammenligne bilder etc.. Aperture er et program som holder orden på alle bildene til brukeren samtidig som det kan brukes til bildebehandling.

I programmet Aperture har brukeren mulighet til å gi parametere til smarte album uten å bruke en vanlig dialogboks. Smarte album gjør mye av det samme som smarte spillelister i iTunes. Det interessante med smarte albumer i Aperture er hvordan parametrene visuelt er knyttet til et smart albumobjekt.

Når brukeren har laget et smart album, blir albumet representert som et objekt i en liste. Brukeren kan da klikke på dette objektet for å få visualisert innholdet til høyere i skjermbildet. På dette smarte albumobjektet er det i tillegg et ikon, hvor brukeren kan

klikke for å få se parameterne til objektet. Når brukeren klikker på dette ikonet blir parameterne visualisert i et eget vindu. Dette vinduet er et transparent vindu som legger seg oppå det eksisterende brukergrensesnittet. Vinduet har i tillegg en liten pil som peker på det smarte albumobjektet som parameterne er gjeldene for. Brukeren kan nå editere parameterne i dette transparente vinduet. Det er ingen avslutningsknapp i dette vinduet som betyr at brukeren når som helst kan forlate vinduet og fortsette en interaksjon andre steder i programmet. Når brukeren editerer parameterne vil brukeren få en umiddelbar respons fra brukergrensesnittet som ligger ”under” det gjennomsiktige parametervinduet. Brukeren vil altså alltid se det gjeldende resultatet som kombinasjonen av de satte parameterne vil gi. Brukeren kan da få en oppfattelse av at en editerer direkte på det smarte albumobjektet.



Figur 17: Parameterne til objektet "smart album" visualiseres gjennom et gjennomsiktig vindu som legger seg oppå det eksisterende vinduet. En liten pil på vinduet forteller brukeren hvilket objekt parameterne gjelder for. Dette er ikke en tradisjonell dialogboks hvor brukeren må avslutte for å se resultatet.

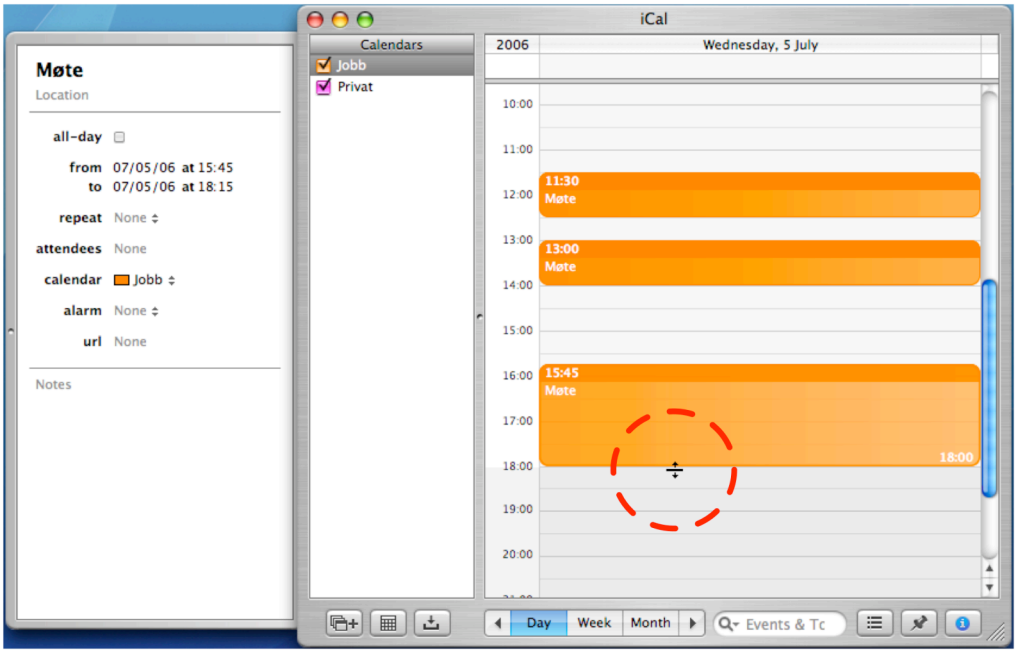
5.3.3 Nye avtaler og editering i iCal

ICal er et kalenderprogram som også er laget av Apple Computer. Programmet lar brukeren holde orden på avtaler og gjøremål en har i hverdagen. I likhet med diabetesprogrammene holder iCal oversikt over en rekke data, som avtaler med tid og sted, notater, påminnelsesfunksjonalitet etc.. Brukeren trenger da funksjoner for å legge inn nye data. Samtidig er brukeren avhengig av at iCal visualiserer dataene på en god og oversiktelig måte tilbake til brukeren. ICal har da funksjonalitet som ligner på den funksjonaliteten en finner i diabetesprogrammene.

ICal består hovedsakelig av et kalendervindu. Kalendervinduet kan vise en hel måned, uke eller dag. Videre er det et eget vindu hvor en kan velge forskjellige kalendere, som en kan definere selv. Brukeren kan da definere kalendere som for eksempel ”jobb” og ”privat”, disse vil da få forskjellige farger i det store kalendervinduet som vil vise alle kalenderne samlet. Brukeren kan også skru av kalenderne slik at de ikke vises i det store vinduet. På denne måten kan brukeren selv bestemme hvilke data som skal visualiseres.

Når brukeren skal legge inn en ny avtale kan dette gjøres ved å klikke og dra musepekeren rett inn i det store kalendervinduet. Avtalen som blir lagt inn begynner der musepekeren var og slutter der en slipper knappen på musen. Avtalen blir da visualisert som en farget boks. Ønsker brukeren å forandre på avtalen, er det et eget vindu hvor alle de tilhørende dataene til avtalen kan editeres. Ønsker brukeren å forandre på tidspunktene til avtalen, kan brukeren ta tak i kanten på boksen og dra

denne oppover eller nedover (Figur 18). Skal en forandre eller flytte hele avtalen, er det bare å ta tak i boksen og flytte den til en annen dag eller en annen tid. Ved å dobbeltklikke på boksen kan brukeren skrive et eget navn på avtalen. ICal trenger da ikke å bruke dialogbokser for å få data fra brukeren. Brukeren kan enten editere dataene i et eget vindu ved siden av hovedvinduet, eller direkte manipulere på selve avtalen.

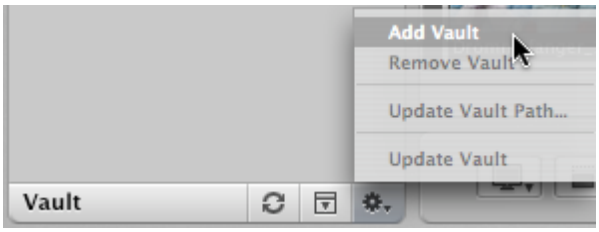


Figur 18: Skjermbilde fra iCal. Hovedvinduet viser en daglig oversikt. I vinduet til venstre kan brukeren editere data i forbindelse med den registrerte avtalen. Musepekeren skifter form når en plasserer den på øverste eller nederste kant av avtaleboksen. Den stiplede ringen, som er lagt til i denne skissen, markerer musepekeren når brukeren drar i avtalen for å forandre tidspunktet.

Brukeren kan da i iCal loggføre nye data ved å klikke rett i kalendervinduet. Ønsker brukeren å editere loggførte data, kan dette gjøres ved å direkte manipulere på dataobjektene.

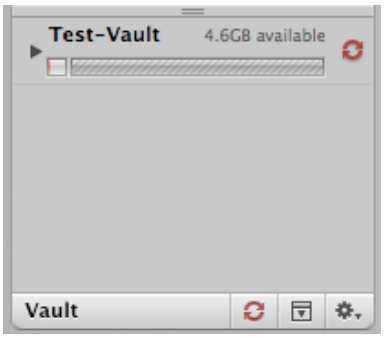
5.3.4 Filsafen i Aperture

I Aperture har det blitt lagt inn en funksjonalitet for å ta sikkerhetskopi av alle dataene eller bildene som ligger i programmet. Denne funksjonaliteten blir kalt ”File vault” eller en filsafe. Denne filsafene representeres i et eget lite vindu i programmet. I dette vinduet kan brukeren legge til flere filsafer. Ved å velge en ny filsafe fra en egen meny får brukeren opp en dialogboks hvor brukeren må bestemme navnet og hvor denne filsafen skal ligge (Figur 19). Filsafen blir da en fil i operativsystemet som inneholder alle bildene.



Figur 19: Brukeren kan legge til en elle flere ny filsafe. Brukeren blir da bedt om i en dialogboks om hvor en ønsker å lagre filsafen, som da blir en egen fil som inneholder alle bildene som ligger i programmet.

Når filsafen er laget blir filsafen visualisert i brukergrensesnittet i Aperture. Filsafen blir visualisert med navnet på filsafen, en tekstelig beskrivelse og en progresjonsbar over hvor mye plass det er i filsafen og et ikon for å oppdatere filsafen. Ikonet for filsafen bytter farge når filsafen er forskjellig fra dataene en finner i programmet. Fargekoden rødt betyr at det finnes bilder som ikke er i filsafen, og fargekoden gul betyr at det finnes metadata om bildene som ikke er lagret i filsafen.



Figur 20: Visualiseringen av en filsafe i programmet Aperture. Progresjonsbaren forteller hvor mye plass som er brukt og hvor mye plass som er igjen i filsafen. Ikonet med pilene gir brukeren mulighet til å oppdatere filsafen.

5.4 Oppsummering av eksisterende programmer

Jeg har i dette kapitlet vist hvordan eksisterende programmer har løst visualiseringen av kriteriene som beskrevet i delkapittel 5.1. Hvordan programmene har visualisert funksjonaliteten har blitt beskrevet med tekst og skjermbilder. Disse beskrivelsene og skjermbildene vil bli videre bli diskutert i neste kapittel. Jeg vil da se på hvordan en kan overføre visualiseringen av funksjonalitet mellom diabetesprogrammene og iTunes, iCal og Aperture.

6 Diskusjon

I dette kapittelet vil jeg diskutere mulige løsninger for brukergrensesnittdesignet til et digitalt loggbokprogram som kan hjelpe en diabetiker med behandlingen. Jeg vil koble funksjonaliteten som er beskrevet i kapittel 3.3 med brukergrensesnittløsninger i diabetesprogrammene og andre typer programmer, som er beskrevet i kapittel 5.3. Jeg vil altså lage skisser som viser hvordan loggførings-, synkroniserings-, kategoriseringsfunksjonaliteten, analyse- og sikkerhetskopieringsfunksjonaliteten kan visualiseres i brukergrensesnittet.

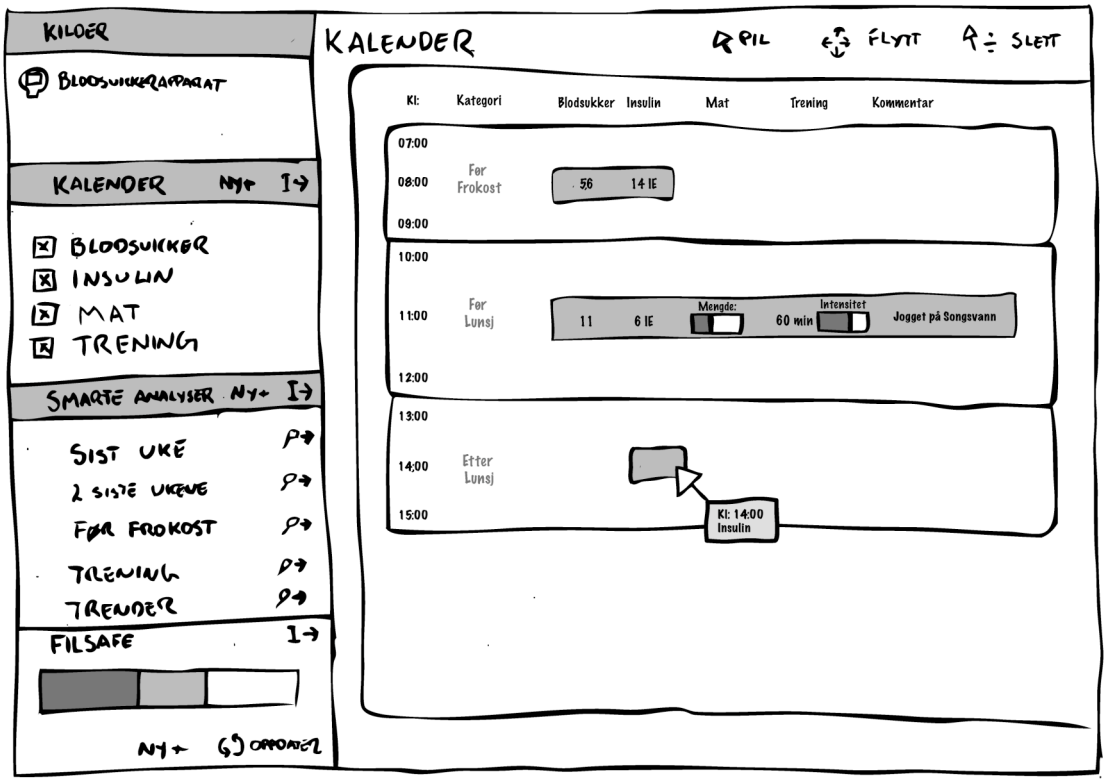
I de påfølgende kapitlene vil jeg presentere skisser som illustrerer hvordan funksjonaliteten kan visualiseres. Skissene er da generert ut i fra en sammenkobling mellom diabetesprogrammene og andre typer programmer. For hver funksjonalitet vil jeg beskrive hva som kan være et problem i brukergrensesnittet til diabetesprogrammene. Videre vil jeg beskrive hvordan en liknende funksjonalitet har vært visualisert i andre programmer. Skissene vil da bli et forslag på hvordan designløsninger fra andre programmer kan overføres til et diabetesprogram.

Skissene som blir presentert vil så bli analysert opp mot det teoretiske rammeverket som er beskrevet i kapittel 4. Disse analysene vil bli kalt en teoretisk analyse. Jeg vil da se på hvordan skissene kan relateres til de teoretiske begrepene som er beskrevet. Begrepene beskriver teorier som kan bidra til å gjøre visualiseringen av funksjoner mer brukbare. I kapittel 4 er det presentert teorier som kan bedre lærbarheten, fleksibiliteten og robustheten. Analysen vil beskrive hvordan skissene kan knyttes til teorier som igjen kan knyttes til bedre lærbarhet, fleksibilitet og robusthet. Jeg vil også diskutere eventuelle teoretiske problemer ved skissene som kan ha påvirkning på brukbarheten. Den teoretiske analysen vil da se på skissene opp mot et teoretisk rammeverk om brukbarhet.

Skissen i de neste kapitlene vil bli beskrevet hver for seg. I det neste kapitlet vil jeg beskrive hvordan skissene skal sees på i en sammenheng. Skissene tar for seg hver funksjonalitet, men er også tenkt i en sammenheng som et helhetlig program.

6.1 Sammenhengen mellom skissene

I Figur 21 beskrives en skisse som viser hvordan hovedbrukergrensesnittet til et digitalt loggbokprogram for diabetikere kan se ut. Til venstre i skissen kan en se fire forskjellige vinduer. Dette er vinduer som visualiserer hver sin funksjonalitet. Kildevinduet vil visualisere synkroniseringsfunksjonaliteten. Kalendervinduet vil vise loggførings- og synkroniseringsfunksjonaliteten. Det vinduet som heter ”smarte analyser” vil visualisere analysefunksjonaliteten og filsafen vil visualisere sikkerhetskopifunksjonaliteten.



Figur 21: En skisse over brukergrensesnittet til programmet. Denne skissen viser hvordan programmet ser ut som en helhet. I vinduene til venstre kan brukeren navigere mellom forskjellig funksjonalitet som kilder, kalender, smarte analyser og filsafe. Innholdet i vinduet til høyre vil forandre seg avhengig av hva en foretar seg i vinduene til venstre. I dette eksempelet viser skissen kalenderfunksjonalitet.

Vinduet til høyre er hovedvinduet. Det som vil bli visualisert i dette vinduet er avhengig av hva en foretar seg i vinduene til venstre. I denne skissen (Figur 21) vises en kalender. Dette vinduet er da visualisert når en opererer på det vinduet til venstre som heter ”kalender”.

6.2 Loggføring av data

Loggføring av data er en vesentlig del av et digitalt loggbokprogram som skal støtte diabetikere. I dag har brukeren kun mulighet til å overføre data fra blodsukkerapparatet. Dette betyr at andre data en ønsker å loggføre må en skrive manuelt inn i programmet for at dataene skal ta del i analysen. Det er viktig at loggføringen ikke blir en tidkrevende prosess for brukeren. Brukeren må enkelt og effektivt få loggført de dataene en måtte ønske som beskrevet i kapittel 3.3.1.1 som omhandler funksjonaliteten til et digitalt loggbokprogram.

En kan kategorisere data som skal loggføres i to kategorier. Den ene kategorien er de direkte kvantifiserbare dataene og den andre kategorien er de dataene som ikke er direkte kvantifiserbare. Blodsukkerverdier og insulindoser er direkte kvantifiserbare. En blodsukkermåling gir et resultat som er et tall som kan vises i en egen måleenhet (mmol/l). Insulindoser måles i antall enheter (IE, internasjonale enheter). Dataene som ikke er direkte kvantifiserbare kan være data for mat, trening, alkohol, sykdom etc.. Når en da skal loggføre verdier rundt disse dataene er det viktig å lage skalaer og verdier som brukeren kan forholde seg til. Det må være enkelt for brukeren å koble hendelser i den virkelige verden med de mulige verdiene for loggføring av hendelsen.

Loggføring av mat kan knyttes opp mot inntak av karbohydrater, fett og proteiner. For en diabetiker er det inntak av karbohydrater som er mest interessant, da det er inntak av karbohydrater som hever blodsukkeret. En vanlig måte å beskrive inntaket av karbohydrater å skrive hvor mange gram en har spist med karbohydrater. Problemet med dette er at det er vanskelig å si hvor mye karbohydrater det er i maten en spiser. En løsning på dette kan være at programmet kan inneholde en stor liste over matvarer. Hvor brukeren kan velge matvarer fra denne listen. Brukeren registrerer hvilke matvarer en har spist og programmet regner da ut mengden karbohydrater. Problemet med dette er at en må bruke tid på å legge inn all maten en har spist i programmet, slik at dette fort kan bli en tidkrevende og omfattende prosess.

En annen metode for å måle inntaket av mat er glykemisk indeks. Glykemisk indeks er en tabell som forteller hvor stor effekt en viss mengde og type mat har på blodsukkeret (Hanås 2002). Denne metoden må da gjennomføres på samme måte som registrering av matvarer. Dette medfører at en vil få samme problematikken med at det fort kan bli en tidkrevende og omfattende prosess og registrere alt en spiser. Et annet problem med denne metoden er at den glykemisk tabellen sier kun noe om hvordan hver enkelt matvare påvirker blodsukkeret. En vil da få problemer med å loggføre et måltid sammensatt av flere matvarer. Den glykemisk indeksen forteller kun påvirkningen av blodsukkeret for en matvare isolert sett. Blir matvaren sammensatt med andre matvarer vil den nye sammensatte matvaren få en egen glykemisk indeks (Christophersen og Hanssen 2004). Det er da ikke mulig å finne ut den nye glykemisk indeksen ved å kombinere indeksen for hver matvare.

Mye av den samme problemstillingen vil en møte for loggføring av fysisk aktivitet. En må finne kvantifiserbare verdier i treningen. En mulig løsning på dette kan være å loggføre lengden på trengingsøkten. Videre kan en sette et tall på hvor intens selve treningsøkten har vært. En kan da bruke lengde og intensitet for å kvantifisere fysisk aktivitet.

Det å loggføre ikke direkte kvantifiserbare data på en effektiv måte, er en vanskelig problemstilling. Skal dataene brukes i mer avanserte analyser er en avhengig av at de er så eksakte som mulig. For å få data eksakte må de kvantifiseres i en eller annen form, som igjen betyr at en må få satt en tallverdi på dataene. En må da finne en måte å konvertere ikke direkte kvantifiserbare data til kvantifiserbare. En må for eksempel kunne konvertere et måltid til å bli representert ved tall. I denne konverteringen er det da viktig å finne en metode som ikke blir for tidkrevende og omfattende for brukeren. Hvis dataene blir for tidkrevende å loggføre er det fare for at brukeren ikke bruker tid på dette i sine daglige rutiner. Det kan også diskuteres hvor mye disse dataene skal brukes i avanserte analyser. Det viktigste med en slik loggføring er at brukeren kan ta med seg erfaringer videre. Har brukeren et forhold til verdiene som er loggført, er det sannsynlig at dataene kan hjelpe brukeren i etterkant. Brukeren kan da bli avhengig av et enkelt system for loggføring som er visualisert på en brukervennlig måte, samtidig som verdiene som loggføres må gjennomføres på en tidseffektiv måte. Visualiseringen av loggføringen bør også ha en ryddig overordnet struktur. Dataene som blir loggført i de forskjellige kategoriene, som blodsukkerverdier, insulindoser, mat, trening etc. må passe sammen i en helhet. Slik at en kan danne seg et ryddig bilde av hvordan dataene er organisert i programmet i forhold til hverandre.

6.2.1 Forskjellen mellom diabetesprogrammene og iCal

Jeg vil nå beskrive hvordan diabetesprogrammene har løst problematikken rundt loggføring av dataene. Videre vil jeg beskrive hvordan dataene har blitt visualisert i brukergrensesnittet i diabetesprogrammene. Denne beskrivelsen bygger videre på skjermbildene som har blitt presentert i kapittel 5.2.1. Jeg vil videre se på hvordan Apple programmet iCal har løst liknende problemstillinger.

Diabetesprogrammene lar brukeren legge inn de kvantifiserbare dataene som blodsukkerverdier og insulindoser som deres opprinnelige enheter. Det vil si at blodsukkerverdiene legges inn som det tallet som en finner på blodsukkerapparatet (mmol/l)og insulindosene legges inn som en internasjonal enhet (IE). Ved loggføring av måltider lar diabetesprogrammene brukeren registrere måltidet i form av karbohydrater. Brukeren må altså beskrive måltidet i form av mengden karbohydrater måltidet bestod av. Programmet SiDiary gir brukeren i tillegg muligheten til å registrer hvilke matvarer måltidet bestod av. Brukeren kan da søke i en stor liste over en mengde matvarer. Ved å sette sammen forskjellige matvarer kan en beskrive selve måltidet. Programmet regner da ut innholdet av karbohydrater til måltidet. Data som beskriver fysisk aktivitet blir beskrevet ved hjelp av lengden i tid og en egen intensitetskala. Brukeren registrere da lengden på selve treningsøkten og hvor intens en oppfattet den i form av for eksempel lett, middels eller hard.

Når en bruker skal legge inn data i diabetesprogrammene må en i OneTouch™ og Accu-Check® gjennomføre en egen registreringsprosess. Brukeren må benytte et eget dialogvindu for å registrere dataene. I dette dialogvinduet taster brukeren inn alle dataene en ønsker å registrere for å så avslutte ved hjelp av en ”ok”-knapp. Dataene blir da sendt videre i programmet eller til en form for database. Når dialogvinduet avsluttes må brukeren navigere seg til en annen del av programmet for å kunne se dataene som har blitt lagt inn. Dataene har da fått en annen visuell fremstilling enn den visuelle fremstillingen brukeren så under innleggelsen av dataene.

Programmet SiDiary har en annen fremstilling av denne loggføringsprosessen. SiDiary brukeren egen tabell, hvor brukeren kan legge dataene rett inn i en tabellen. Brukeren trenger da ikke å starte et eget dialogvindu for å registrere og loggføre data. Dette gir brukeren mulighet til å editere på dataene mer direkte. Tabellen begrenser allikevel brukeren til å kun legge inn data innenfor tidsrammene som tabellen er laget etter. Tabellen har for eksempel en rad for verdier som ble registrert klokken 07.00 og en rad for de verdiene som ble registrert klokken 10.00. Hele tabellen må da forandres vis brukeren ønsker å registrer data mellom disse to klokkeslettene. Dette kan gjøre registrering av nye data unødvendig tungvint, siden en må lage en helt ny kolonne før en kan registrer dataene på det ønskede tidspunktet. Tabellen har også en form for det en kan kalle en logisk brist. Tabellen har, som tidligere beskrevet i kapittel 5.2.2, en x- og y-akse. Y-aksen viser blodsukkerverdier, og x-aksen viser tiden av døgnet. Ønsker brukeren å registrere for eksempel en blodsukkerverdi på 5 mmol/l klokken 17.00, klikker brukeren på de representative verdien på x- og y-aksen. Brukeren får da opp eksempelvis verdien 6.9 mmol/l i vinduet. Denne verdien er da feil plassert i forhold til x- og y-aksen. Brukeren må da taste inn ønsket verdi, som i dette tilfellet er 5 mmol/l. I dette tilfellet må en her taste inn verdien som allerede er angitt i forhold til y-asken. Taster brukeren inn en annen verdi, vill tallet flytte seg automatisk til riktig celle i forhold til y-asken. Brukeren får presentert et brukergrensesnitt som er

tilsynelatende manipulerbart. Problemet oppstår når brukergrensesnittet forandrer automatisk på brukerens handlinger.

Diabetesprogrammene lar brukeren loggføre matvarer etter karbohydrater og trening etter tid og intensitet. Jeg har tidligere argumentert for at karbohydrater kanskje kan bli for vanskelig og tidkrevende å loggføre. Brukeren må vite hvor mye karbohydrater det er i måltidet en spiser, noe som kanskje ikke alltid er like lett å vite. Finnes det da andre måter å gjøre dette på? I skissen som blir presentert i neste kapittel vil en se en annen måte å visualisere loggføringen av matvarer. OneTouch™ og Accu-Check® brukere et dialogvindu for å registrere nye data. I kapittel 3.3.2.2 beskrives det at et slikt program bør være fleksibelt for brukeren. Holland og Oppenheim (1999) beskriver videre at bruken av dialogbokser er lite fleksibelt for brukeren. En dialogboks låser brukeren ute av programmet som igjen kan begrense brukerens interaksjon med programmet. Dialogboksen som brukes i registreringen av dataene kan da anses som lite fleksibelt. Programmet SiDiary bruker ikke en slik dialogboks for å registrere data fra brukeren. I stedet bruker de en tabell, hvor brukeren kan legge inn data direkte. Dette kan da anses som mindre fleksibelt for brukeren. Allikevel har tabellen noen ulemper. Som tidligere vist må brukeren lage nye kolonner hvis tidspunktet for de eksisterende kolonnene ikke passer med de dataene en ønsker å legge inn. Tabellen har også det som ble kalt for en logisk brist. Tabellen forandrer automatisk på data som brukeren legger inn. En kan da si at SiDiary har klart å lage en visualisering av loggføringsprosessen uten å bruke et dialogvindu, men noen av egenskapene til tabellen kan gjøre den tungvint i bruk. Hvordan kan en så visualisere dette på en annen måte?

Kalender programmet iCal presenterer loggføringsprosessen av dataene på en annen måte. I kapittel 5.3.3 kan en se at skjermbilder i ICal presenterer dataene gjennom en kalenderoversikt. Kalenderoversikten ligner på de kalendrene vi finner i papirutgave i form av almanakker eller kalendere til å henge på veggen. En kan si at iCal brukere en metafor og metaforen er da en kalender. Kalenderen gir brukeren mulighet til å loggføre dataene direkte i kalenderen. På denne måten unngår iCal å bruke et dialogvindu. Gjennom bruken av det Shneiderman (1998) kaller direkte manipulasjon, kan brukeren legge nye data rett inn i kalendermetaforen. Brukeren klikker med musepekere der hvor en ønsker at dataene skal legges, da vil en få opp en egen boks som visualiserer dataene. På denne måten lager iCal en struktur i programmet som gir dataene en sammenheng med hverandre. Alle dataene ligger i en kalender noe som vil gi en naturlig sammenheng mellom boksene som representerer dataene.

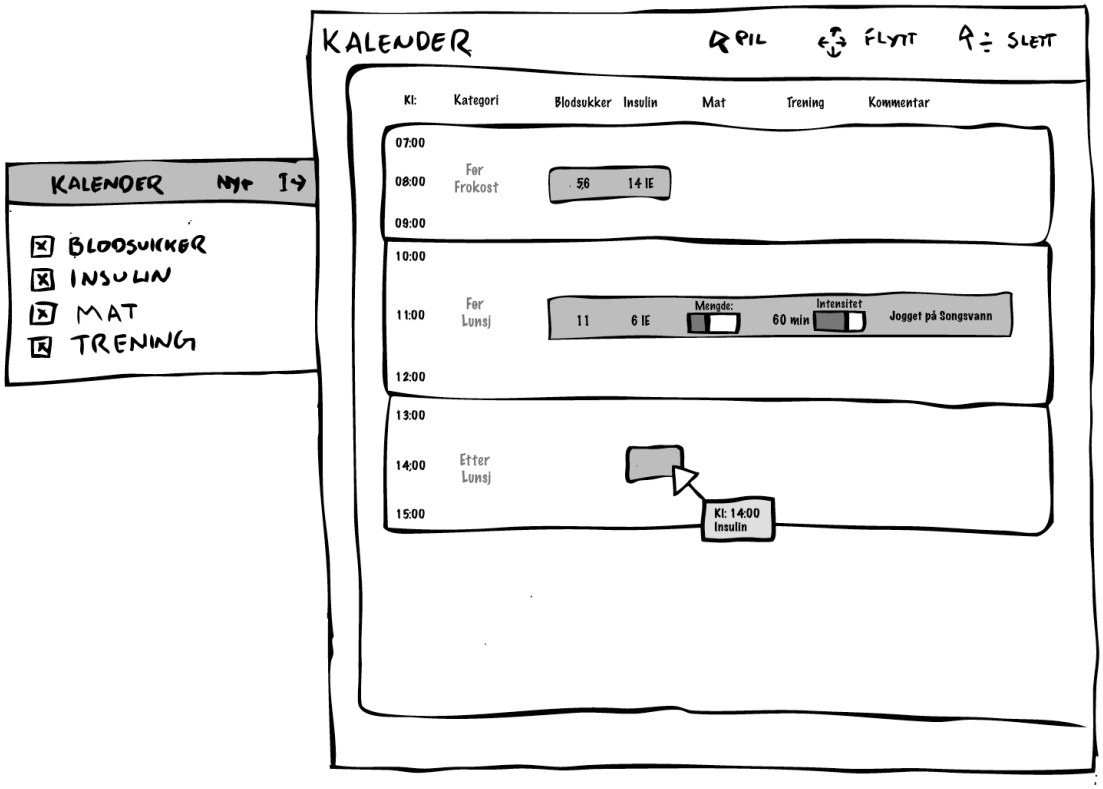
Det kan da virke som en god ide å prøve å overføre denne kalendermetaforen til et program for diabetikere. I neste kapittel vil en se en skisse på en mulig løsning på en slik overføring.

6.2.2 En skisse av et brukergrensesnitt for loggføring av data

Figur 22 viser et eksempel på hvordan loggføring av data kan gjennomføres. Til høyre i eksempelet er det en kalender, som viser en dag. Y-aksen i denne kalenderen er klokkeslettene i løpet av en dag og x-aksen er kolonner hvor data kan loggføres i kalenderen. Tanken er at brukeren kan legge til og fjerne kolonner i x-aksen etter eget ønske. I kolonnene i dette eksempelet (Figur 22) kan brukeren loggføre data rundt blodsukkeret, insulin, mat og trening. Det er også mulig å skrive en kommentar til

selve målingen. Ovenfor selve kalender er det laget eksempler på noen verktøy. Tanken er at verktøyene kan brukes til å manipulere på selve kalenderen. Velger brukeren for eksempel ”pil”, kan brukeren klikke i selve kalendervinduet og det blir opprettet en ny måling der brukeren klikker. Brukeren vil se en hjelpetekst ved siden av pilen, som gir brukeren utfyllende informasjon om hvilke klokkeslett og kategori som vil bli lagret hvis brukeren klikker på dette stedet. Videre finnes det instrumenter for å flytte og slette data.

Til venstre i eksempelet er det et eget kalender vindu. I dette kalendervinduet kan brukeren legge inn egne underkalendere over ønskelige verdier som blodsukker, insulin, trening, mat etc. Brukeren kan lage slike kalendere etter eget ønske, ved å klikke på ”Ny +” knappen. Det er da en sammenheng mellom underkalendrene og de forskjellige kolonnene i kalendervinduet. Ønsker brukeren å skru av visualiseringen av en underkalender i kalendervinduet, klikker en på krysset ved siden av underkalenderen.



Figur 22: Skissen viser hvordan en kan loggføre nye data. Brukeren kan klikke rett i kalendervinduet for å lage en ny måling. I en egen boks ved siden av musepekere vil brukeren se detaljert informasjon om hvilke data som vil bli registret hvis brukeren klikker der musepekeren befinner seg.

Når brukeren har klikket for å lage en ny boks i brukergrensesnittet kan brukeren taste inn verdien på dataene som brukeren vil lagre. De direkte kvantifiserbare dataene, som blodsukker og insulin doser, må brukeren taste inn som tallverdier i brukergrensesnittet. I denne skissen er det to typer data som ikke er direkte kvantifiserbare, det er mat og trening. Når brukeren skal angi en verdi i forhold til mat, drar brukeren eller klikker brukeren direkte i en egen boks. Denne boksen kan minne om en progresjonsbar som vanligvis brukes til å gi informasjon om fremdriften for en igangsatt operasjon i et program. Brukeren drar den delen som vanligvis viser

progresjonen, til høyre for å øke mengden. Brukeren angir da her en visuell mengde i brukergrensesnittet ved hjelp av det en kan kalle en manipulerbar progresjonsbar. Dette gjør at brukeren forholder seg til en visuell mengde. En kan da bestemme selv hva som skal representere minste og største verdi. Denne verdien kan enkelt oversettes av programmet til en eksakt verdi, for eksempel et prosent tall. En kan da gjennomføre analyser med den visuelle verdien. Når brukeren skal loggføre en treningsmengde kan brukeren angi tidslengden på trengingen i minutter og bruker en manipulerbar progresjonsbar for å angi intensiteten.

I neste kapittel vil jeg analysere hvordan denne skissen benytter teorier som kan knyttes til bedre brukbarhet.

6.2.3 Teoretisk analyse av skissen

6.2.3.1 Interaksjon

Skissen som beskriver loggføringsprosessen gir brukeren mulighet til å loggføre verdier direkte i kalendervinduet. Dette gjør at skissen ikke tar i bruk et eget dialogvindu for å registrere ny data. Dette er med på å forandre hvordan brukerens interaksjon med programmet. Ved bruken av et dialogvindu ville brukerens interaksjon ligne på Normans (1998) eksekverings evalueringsløkke. Hvor brukeren bestemmer seg for å utføre en handling, eksekverer handlingen i brukergrensesnittet og til slutt evaluerer om resultatet gjorde den handlingen en ønsket. Et dialogvindu kan da brukes for å gjennomføre en eksekvering-evaluerings løkke. Brukeren bestemmer seg for å utføre en handling når en starter dialogen, ved hjelp av et menyvalg eller et ikon. Videre får brukeren ikke noen tilbakemeldinger fra brukergrensesnittet før dialogen avsluttes og utføres. Dette betyr at valg som brukeren gjør i selve dialogvinduet, vil en ikke få en tilbakemelding på før en evaluerer resultatet fra hele dialogen.

I skissen unngår en da denne dialogproblematikken ved å ikke bruke et dialogvindu. Brukeren kan manipulere direkte på en kalender. Dette gjør at en kan si at brukerens interaksjon i programmet utføres i mange små sykluser. Ønsker brukeren å loggføre for eksempel en insulindose, navigerer brukeren musen til det stedet hvor dataene skal lagres ved hjelp av musepekeren og den ekstra informasjonsboksen under selve musepekere. Brukeren klikker så i kalenderen og vil få opp en tom firkant i brukergrensesnittet. I denne firkanten legger en så inn ønsket verdi. Med denne interaksjonen får brukeren mange flere tilbakemeldinger under selve handlingen enn ved bruken av en dialogboks. En kan si at brukeren får kontinuerlig tilbakemeldinger under handlingen. Musepekeren, informasjonsboksen under musepekeren, firkanten i kalenderen og tallverdien som tastes inn er med på å gi kontinuerlig respons tilbake til brukeren. En kan da si at handlingen blir utført av flere små operasjoner. Hver operasjon i selve handlingen gir brukeren tilbakemeldinger.

Interaksjonen med firkantene eller objektene kan forklares med Djajadiningrat et al. (2002) sine begreper: operasjonsformål og innebygd respons. Operasjonsformålet kan sees på den informasjonen brukeren får før selve handlingen, i preoperasjonsfasen. Innebygd respons er resultatet av selve handlingen eller informasjonen brukeren får i en postoperasjonsfase. En kan si at hvert element i skissen har et operasjonsformål og en innebygd respons. Ser en for eksempel på musepekeren i skissen vil

operasjonsformålet bli visualisert av boksen som vises under musepekeren. Informasjonen i denne boksen vil fortelle brukeren hva resultatet av en eventuell operasjonen vil bli. Når brukeren klikker i selve kalenderen vil den innebygde responsen være en opprettelse av en firkant der musepekeren befant seg. Djajadiningrat et al. (2002) vektlegger at informasjonen en får i postoperasjonsfasen må kunne knyttes nært til selve operasjonen. I dette tilfellet vil firkanten bli visualisert der brukeren klikket, og vil ha en nær tilknytning til selve klikkeoperasjonen. Elementene i brukergrensesnittet vil da ha hver sin pre- og postoperasjonsfase. Dette får som tidligere beskrevet, betydning for brukerens interaksjon med selve programmet. En vil løsrive seg fra Normans eksekverings-evaluerings løkke og få en interaksjon som består av små sykluser.

De små syklusene gjør at brukeren letter kan rette opp feil. Normans (1998) begrep som kalles forskjell ved evaluering, beskriver feil som oppstår fordi resultatet av handlingen ble forskjellig fra hva brukeren forventet, vil reduseres når interaksjonen forekommer i små sykler. Operasjonsformålet til pilen med informasjonsboksen under forteller brukeren hva resultatet vil bli, som igjen er med på å minske sannsynligheten for en feil ved evaluering av operasjonen. I Normans eksekverings-evalueringsløkke må brukeren gjennomføre hele løkken eller handlingen før en har mulighet til å oppdage feil. Når en deler handlingen opp i flere operasjoner har en da mulighet til å angre sine operasjoner før selve handlingen er utført. Klikker og lager brukeren en firkant i brukergrensesnittet som en finner ut at var feil plassert, kan brukeren slette boksen og utføre operasjonen på nytt. Brukeren slipper å gjennomføre hele handlingen før en kan korrigere sine egne feil. Ved bruk av en dialogboks og Normans eksekverings-evalueringsløkke må brukeren taste inn alle dataene og få en respons for hele handlingen, før en har mulighet til å oppdage feil. Brukeren har da liten eller ingen mulighet til å rette opp feil under selve interaksjonen, siden en ikke får tilbakemelding før hele handlingen er ferdig.

6.2.3.2 Metaforer og den konseptuelle modellen

I skissen brukes en form for metafor. Kalendervinduet kan kalles en metafor for en almanakk. I en almanakk finnes det forskjellig typer dagsoversikter. En av disse typene er da brukt i denne skissen. Hver dag finnes det en rad med klokkesletter slik at en kan notere avtaler eller gjøremål ved siden av klokkeslettet. Muligheten til å notere avtalere er da byttet ut med mulighet for å loggføre data i skissen.

Preece (2002) beskriver en metafor som et virkemiddel for å presentere en konseptuel modell. I denne skissen kan en si at metaforen er med på å presentere en konseptuel modell. Kalenderen gjør at brukeren får et forhold til hvordan dataene er lagret i forhold til hverandre i dagsoversikten. Loggføring av data er knyttet til hendelser i hverdagen og hverdagen kan beskrives ved hjelp av en kalender eller almanakk. Laurel (1993) beskriver at en av fordelene ved bruken av en metafor er at en får et konsept som passer sammen. Dette blir da et konsept som passer sammen, en bruker kalenderen til å loggføre data, og kalenderen vil da inneholde de loggførte dataene.

En finner også eksempler i skissen som bryter med en almanakk metafor. I en virkelig almanakk kan en ikke ta direkte tak i en avtale og dra den rundt i kalenderen. En har heller ikke subkalendere (egne kalendere for blodsukker, insulin, mat, etc.), slik som en har i skissen. En har heller ikke muligheten for å bruke små instrumenter for å angi data i en almanakk, slik som en har med den manipulerbare progresjonsbaren. Dette

er med på å bryte med almanakk metaforen. En kommer da inn på fordelene og ulempene ved å bruke metaforer. En metafor kan ikke være lik virkeligheten, da ville det ikke vært en metafor heller en avbildning. Problemet blir da å beskrive for brukeren hva som er forskjellig fra virkeligheten. Madsen (2000) beskriver det som skiller seg fra den virkelige verden i metaforen, som ”magien” i en metafor. Hadde skissen vært en direkte avbildning av en almanakk, ville ikke brukeren kunne slettet og flytte data like enkelt. En vil da miste mye av fortjenesten ved at brukeren loggfører dataene sine på en datamaskin. Mye av argumentene for at en skal bruke en datamaskin for slik loggføring er nettopp at loggføringen skal være enklere og raskere.

Almanakk metaforen kan i denne skissen være med på å hjelpe brukeren i å forstå en konseptuel modell. Den hjelper ikke nødvendigvis brukeren til å forstå hvordan elementene i brukergrensesnittet skal benyttes, siden flere av de bryter med metaforen. For å beskrive bruken av disse elementene hjelper ikke nødvendigvis almanakkmetaforen brukeren til denne forståelsen. Allikevel ser en at almanakkmetaforen er med på å lage et konsept som passer sammen. Som tidligere beskrevet passer de loggførte dataene sammen med en kalender. En kalender inneholder data som er knyttet til tid og dato, slik som vi kjenner en almanakk fra den virkelige verden.

6.2.3.3 Direkte manipulasjon og instrumenter

Skissen over loggføringsprosessen tar også i bruk direkte manipulasjon. Dataene brukeren logger blir representer som virtuelle objekter. Objektene er i form av en firkant, som kapsler inn tallverdiene og de manipulerbare progresjonsbarene. Videre er også selve kalendervinduet også direkte manipulerbart, da brukeren kan klikke i kalenderen for å loggføre nye data. Shneidermans (1998) direkte manipulasjons prinsipper beskriver blant annet at objekter skal være synlige og brukeren skal få kontinuerlig tilbakemelding.

Ut i fra denne skissen kan en si at interessante objekter er synlige. Kalenderen er synlig og dataobjektene som er plassert inni kalenderen er synlige for brukeren. Disse objektene gir også en kontinuerlig tilbakemelding. Det er beskrevet tidligere i analysen at under interaksjonen får brukeren kontinuerlig tilbakemeldinger. Som en utvidelse til direkte manipulasjon er det i skissen beskrevet det som Beaudouin-Lafon (2000) beskriver som instrumenter. Pilene som kan brukes til å operere på objektene i skissen kan betraktes som instrumenter. Den manipulerbare prograsjonsbaren kan også kategoriseres som et instrument. Disse instrumentene kan da brukes til å forandre på parametrene til dataobjektene i kalenderen eller behandle dataobjektet som en helhet. Dataobjektene i dette tilfellet er det Beaudouin-Lafon (2000) kaller domeneobjekter. Et domeneobjekt kjennetegnes da ved at intrumenter kan forandre på domeneobjektets parametere.

Beaudouin-Lafon (2000) beskriver to forskjellige instrumenttyper avhengig av aktiviseringsprinsipper og plassbehov. En kan definere pilene og den manipulerte progresjonsbaren til hvert sitt prinsipp. Pilene kjennetegner at de må aktiveres før en bruker dem. En må aktivere ønsket pil for å så bruke den på objektet. Dette medfører at pilene er mer tidkrevende å bruke, siden en må alltid aktivere riktig instrument (Beaudouin-Lafon 2000). Ved bruken av progresjonsbarinstrumentet trenger en ikke på samme måte å aktivisere riktig instrument. Instrumentet er hele tiden visualisert i

brukergrensesnittet og kan benyttes direkte. Dette gjør at instrumentet har et større plassbehov enn pilene.

I dette eksempelet kan en diskutere hvordan disse prinsippene aktivt hinder brukeren i å gjøre uønskede feil. En kan tenke seg et brukergrensesnitt i kalendervinduet hvor brukeren kunne brukte samme musepeker til å flytte data, slette data og registrere nye data. Objektene må da kunne utføre forskjellige handlinger avhengig hvor på objektet musepekeren befinner seg. Et kryss i høyre hjørne kan for eksempel bety at en ville slette dataobjektet, og plassere brukeren musepekere midt på data objektet vil en kunne flytte selve objektet. En slik løsning vil kreve mer plassbehov i brukergrensesnittet, siden alle funksjonene må visualiseres på selve dataobjektet. I skissen har en gitt denne funksjonaliteten til instrumenter. Dette medfører at dataobjektene kan være mindre i selve brukergrensesnittet. Brukeren må i stedet velge et instrument som utfører den ønskelige operasjon. En får da lenger aktiveringstid for selve operasjonen. Denne forlengede aktiveringstiden kan bidra til at det er vanskeligere for brukeren å gjennomføre en uønsket handling.

Ser en på det Beaudouin-Lafon (2000) kaller graden av indireksjon på disse instrumentene er instrumentene ganske like. Graden av indireksjon deles opp i et avstand- og tidsaspekt (Beaudouin-Lafon 2000). Pilene og progresjonsbarinstrumentet har begge et lite avstands- og tidsaspekt. Dette fordi avstanden fra instrumentet og det objektet en styrer er liten. Pilene brukes direkte på objektene, slik at avstanden er minimal. Når brukeren drar i progresjonsbarinstrumentet vil også reaksjonen fra instrumentet være ved en minimal avstand mellom instrumentet og responsen.

Progresjonsbarinstrumentet har også det Beaudouin-Lafon (2000) kaller stor grad av kompatibilitet. Kompatibilitetsgraden beskriver hvordan inndata og utdata er kompatible i forholdt til hverandre. Siden brukeren drar deler av instrumentet mot høyre eller venstre er dette identisk med håndbevegelsen brukeren fysisk utfører ved hjelp av musen. Det er også stor kompatibilitet i responsen brukeren får fra intrumentet. Brukeren vil se at progresjonsbaren vil bli større eller mindre ettersom brukeren drar den mot venstre eller høyre.

6.2.3.4 Frembydelsen til progresjonsbarinstrumentet

Preece (2002) beskriver at mye av fordelen ved å bruke direkte manipulasjon, er at dette hjelper brukeren i å lære seg basisfunksjonaliteten som et grensesnitt tilbyr. Shnidermans (1998) beskriver dette med at direkte manipulasjon skal erstatte et komplekst språk. Allikevel er en avhengig av at objektene visualiseres på en slik måte at brukeren forstår hva de skal brukes til. En kan bruke Norman sitt frembydelses begrep for å beskrive den intuitive oppfattelsen brukeren har av et virkelig objekt. Videre mener Norman at digitale objekter ikke kan ha frembydelse på samme måte som i virkeligheten, en bygger heller erfaring om bruk av virtuelle objekter på lærte konvensjoner.

I skissen presenteres progresjonsbarinstrumentet. Dette er et instrument som brukeren ikke nødvendigvis kjenner fra lærte konvensjoner. Brukeren vil da ha en mer eller mindre intuitiv oppfattelse av hvordan dette instrumentet fungerer uansett om denne intuisjonen er riktig eller ikke. Denne intuisjonen kan sammenlignes med Norman sitt frembydelsesbegrep. Det vil her da være naturlig å tro at hvordan instrumentet er visualisert er vesentlig for hvordan brukeren oppfatter frembydelsen til instrumentet.

Instrumentet i skissen har en form for tekstlig merking. Denne merkingen beskriver kun verdien som instrumentet representerer og gir ingen beskrivelse av hvordan instrumentet fungerer. Haramundanis (1996) hevder at ikoner, som også er objekter i brukergrensesnittet, ikke kan stå alene uten tekst for å gi en fullverdig beskrivelse av objektets betydning. En kunne da gitt en tekstlig beskrivelse av instrumentets funksjon, for å gi objektet en bedre frembydelse til brukeren. Ulempen med dette er at det ville tatt større plass i brukergrensesnittet og Norman (1998) mener at hvis et objekt trenger merking kan dette indikere at designet er feil. Behovet for en tekstelig beskrivelse blir også minimal når brukeren har lært seg bruken av dette instrumentet, det har da blitt en del av det Norman (1998) beskriver som opplærte konvensjoner. En annen fremgangsmåte kan være å bruke en egen tekstboks som kommer opp ved siden av musepekeren når musen er over instrumentet. Teksten i denne boksen kan da gi brukeren en kort beskrivelse av instrumentet. En slik boks kan også minne om direkte kombinasjon. Ved kombinasjonen av et instrument og musepekeren, vil en tekstboks bli visualisert i brukergrensesnittet. En utfordring ved bruken av slike bokser er at instrumentet ikke gir brukeren noen indikasjon på at denne boksen eksisterer i utgangspunktet. Brukeren kan ikke se på instrumentet at en slik boks vil visualiseres når musepekeren er over boksen. Bruken av en slik boks, bygger da på en konvensjon, som brukeren må ha lært tidligere.

Progresjonsbarinstrumentet er også et objekt som er direkte manipulerbart. Et av Shneidermans (1998) direkte manipulasjonsprinsipper sier noe om at brukeren skal ha mulighet til å angre sine handlinger. Denne skissen sier ingen ting om brukeren kan angrer sine handlinger, men hvis brukeren har mulighet for dette, kan dette ha innvirkning på hvordan brukeren utforsker instrumentet. Brukeren har da mulighet til å teste hvordan dette instrumentet fungerer, for å så angre sine handlinger for å komme tilbake til utganspunktet. Dette er med på at brukeren kan lære seg selv opp til å oppnå forståelse for bruken av dette instrumentet. Gjennom en slik prosess kan da brukeren lære seg opp til en konvensjon, som en kan bruke videre i brukergrensesnittet.

6.2.3.5 Sammendrag

I denne delanalysen har jeg analysert en skisse som beskriver en visualisering av loggføringsfunksjonaliteten. Jeg har beskrevet hvordan skissen benytter seg av en interaksjon som er delt opp i flere små sykler. Denne interaksjonen har jeg knyttet til Djajadiningrat et al. (2002) sitt opperasjonsformål og innbygget respons. Dette er i motsetning til et dialogvindu hvor en har en stor sykl for hele interaksjonen, som kan knyttes til Normans (1998) eksekverings evalueringsløkke. Videre har jeg vist hvordan skissen bruker en almanakkmetafor for å beskrive et helhetlig konsept i skissen. Dette gjør at en får et konsept hvor dataene en loggfører passer inn i kalenderen som er visualisert. Jeg har også beskrevet hvordan skissen bruker direkte manipulasjon og instrumenter. Det er to typer instrumenter i skissen, de med aktiveringsbehov og de som ikke har det. De med aktiveringsbehov kan være med på å gjøre det vanskeligere å gjøre uønskete feil, siden det tar lenger tid å aktivere denne typen instrumenter. Videre har jeg vist at den manipulerbare progresjonsbaren er et instrument uten aktiveringsbehov og har en stor grad av kompatibilitet siden inndata er på samme formen som utdataene. Videre har jeg diskutert hvordan det nye manipulerbare progresjonsbarinstrumentet kan læres av brukeren. Instrumentet bygger ikke nødvendigvis på det Norman (1998) kaller lærte konvensjoner. Shneidermans

(1998) krav til at i direkte manipulasjon skal det være mulig å angre sine handlinger, kan bli et viktig element i hvordan brukeren lærer dette nye instrumentet.

6.3 Synkroniseringsprosessen

Brukeren har et behov for å overføre data fra periferiutstyr til programmet som behandler dataene for diabetikeren. Dette gjøres gjennom en synkroniseringsprosess. Brukeren slipper da å legge dataene inn manuelt i programmet, noe som sparer brukeren for mye tid.

En kan da spørre seg hva skjer egentlig i en synkroniseringsprosess? Sier en at enhet A skal synkroniseres med enhet B, har en flere forskjellige muligheter for hva resultatet blir. A kan kopiere alle data over til B, A og B kan kopiere sine data over til hverandre eller B legger sine data over på A. I tillegg kan en ha flere forskjellige regler for hva som skal skje i selve prosessen. Hva skjer med data som er like? Skal disse dataene legges inn dobbelt, eller skal en av to duplikater slettes? Hvis den ene enheten overfører data til den andre, hva skjer med dataene som ligger på den originale enheten? Skal disse dataene slettes eller blir de liggende til neste synkronisering. En kan da si at det finnes mange forskjellige resultater av en synkroniseringsprosess.

En utfordring ved en slik synkroniseringsprosess er å gi brukeren en forståelse av hvilke av flere mulige prosesser som kommer til å bli gjennomført når en setter i gang en synkroniseringsprosessen. En kan si at visualisering av synkroniseringsfunksjonaliteten og selve bruksituasjonen har betydning for hvordan brukeren oppfatter funksjonaliteten. Når brukeren kobler til en iPod til en datamaskin for første gang tilsier bruksituasjonen at det er naturlig å tro at brukeren har som mål å få sanger overført til sin iPod, siden brukeren vet at det ikke eksisterer noen sanger på den fra før. Problemet oppstår når brukeren kobler til iPoden for andre gang, hva skjer da i synkroniseringsprosessen? Brukeren har da de tidligere overførte sangene på iPoden og har kanskje som mål å slette noen sanger, eller å legge til noen nye sanger og beholde resten av sangene på iPoden. Hvordan funksjonaliteten visualiseres er da vesentlig.

Hvordan synkroniseringen visualiseres er med på å gi brukeren en forståelse av den konseptuelle modellen til programmet og spesielt den konseptuelle modellen til synkroniseringsprosessen. Synkroniseringsprosessen kan fort bli en skjult prosess hvor brukeren ikke har forståelse eller kontroll over hva prosessen utfører.

6.3.1 Forskjellen i synkroniseringsprosessen mellom diabetesprogrammene og iTunes

I kapittel 5.2.1 beskrives det at diabetesprogrammene visualiserer synkroniseringen med blodsukkerapparatet via et dialogvindu. Dialogvinduet veileder brukeren gjennom synkroniseringsprosessen. Dette vinduet brukes til å forklare for brukeren hva brukeren skal foreta seg i forbindelse med synkroniseringsprosessen og til å få informasjon fra brukeren. Hva som vil skje i selve prosessen, hvilke data som blir overført og hvor disse vil bli lagret er ikke visualisert for brukeren.

Dette gjør at brukergrensesnittene i diabetesprogrammene gir liten indikasjon på hvordan dataene blir behandlet, med tanke på regler under overføringsprosessen. SiDiary er det eneste programmet som har en egen preferansedel hvor brukeren kan bestemme om bare nye data skal overføres eller om alle data skal overføres. Det er da ikke visualisert for brukeren hvordan regler om hvordan duplikater blir behandlet eller om dataene på blodsukkerapparatet blir slettet eller ikke etter overføringen. Mangel på slik visualisering er med på å usynliggjøre deler av prosessen. Programmet må da ha et standardoppsett med regler for hva som vil skje med behandlingen av dataene. Dette kan føre til at brukeren kan få en dårligere forståelse av den konseptuelle modellen til synkroniseringsprosessen.

I iTunes er synkroniseringen av iPoden visualisert på litt forskjellige måter. En kan skille mellom to metoder. Den ene er hvor brukeren velger at iPoden skal oppdatere seg automatisk med en gang en plugger iPoden til datamaskinene, og den andre brukes direkte manipulasjon for å overføre data. Ikonet som visualiserer iPoden i brukergrensesnittet er med på å vise koblingen mellom enheten og programmet. Ikonet representerer et objekt i brukergrensesnittet som kan brukes til direkte manipulasjon. Videre visualiseres innholdet på selve iPoden gjennom en liste over alle sangene som ligger på iPoden. Brukeren får da en annen kontroll over synkroniseringsprosessen, enn gjennom dialogvinduet som diabetesprogrammene tilbyr. Gjennom direkte manipulasjon kan brukeren dra sangene til og fra selve iPoden. Brukeren har da full kontroll over om dataene blir overført fra iPoden til datamaskinen eller omvendt. Når synkroniseringsprosessen visualiseres og kontrollesers gjennom direkte manipulasjon løses også problematikken rundt håndteringen av duplikater. Hvis brukeren ønsker at det skal ligge to eksemplarer av samme sang på iPoden, kan brukeren dra samme sang to ganger over på iPod ikonet.

Visualiseringen av synkroniseringsprosessen fremstår på en annen måte når iPoden oppdateres automatisk når brukeren kobler inn iPoden. Brukeren får da kun informasjon om synkroniseringsprosessen. Brukeren må da ha satt opp egne preferanser i en egen del av programmet. Disse preferansen bestemmer hvilke regler som skal brukes under prosessen og hvordan prosessen gjennomføres. Denne måten å visualisere synkroniseringen av den eksterne enheten på, kan sammenlignes med det som eksisterer i diabetesprogrammene.

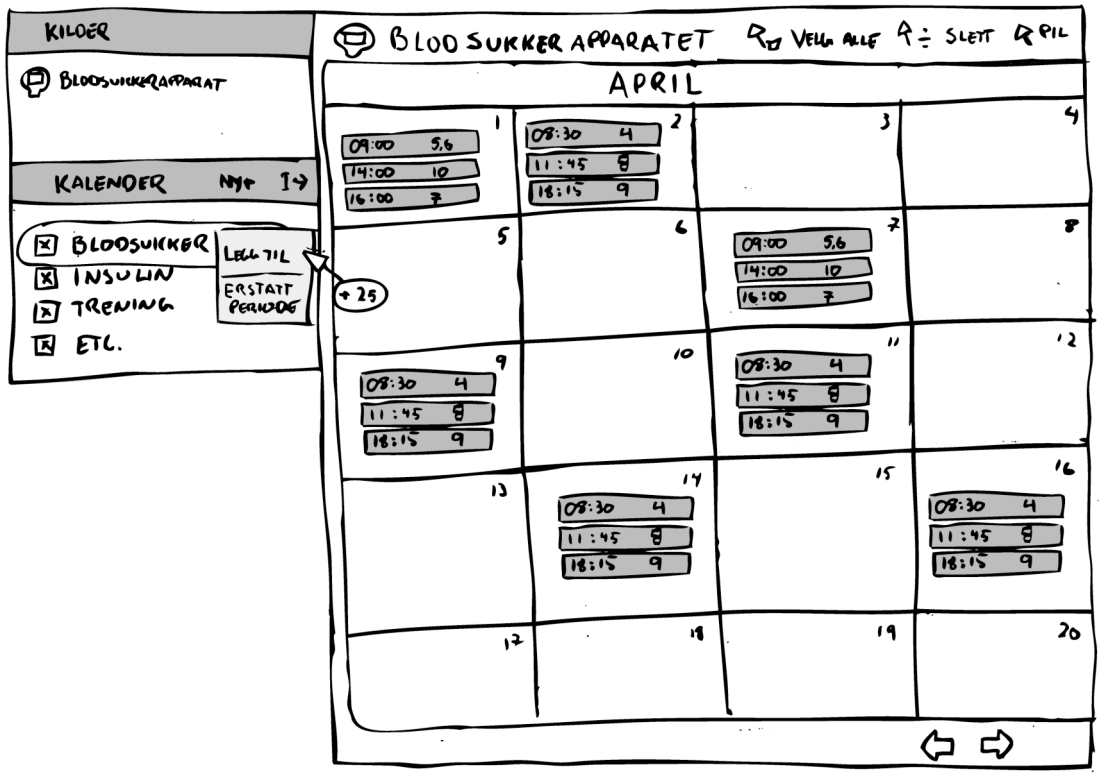
I neste kapittel beskrivers en skisse av en mulig løsning på hvordan synkroniseringsprosessen kan gjennomføres. Skissen overfører da måten iTunes synkroniserer data med en ekstern enhet til et diabetesprogram. Dette kan løse ulempene med bruken av et dialogvindu til synkroniseringsprosessen i diabetesprogrammene.

6.3.2 En skisse av en synkroniseringsprosess

Figur 23 viser en skisse på hvordan en overføringsprosess kan visualiseres. I likhet med iTunes er det en liste til venstre i brukergrensesnittet, som viser hvilke kilder som er koblet til datamaskinen. Under kildelisten er det en egen kalender. Denne kalenderen er den samme kalenderen som en bruker til å loggføre data. Vinduet til høyere representerer de data som finnes på selve blodsukkerapparatet. Dataene er visualisert som egne bokser i en kalenderoversikt med tallverdier for tid og blodsukkerverdi. Øverst til høyre i brukergrensesnittet er det plass til instrumenter

som en kan bruke på dataene som ligger i blodsukkerapparatet. I dette eksempelet er det tegnet opp tre instrumenter ”velg alle”, ”slett” og ”pil”. Dette gir brukeren mulighet til å slette eller velge ett eller flere objekter.

Dataene på blodsukkerapparatet blir da visualisert som objekter. En kan overføre data til kalenderen ved å bruke musepekere til å dra ett eller flere objekter over kalenderobjektet som representerer blodsukkerverdiene. Når brukeren drar en eller flere objekter fra blodsukkerapparatet over på blodsukkerkalenderen, kommer det opp en liten meny. Brukeren kan da velge hvilken operasjon som skal utføres. I dette eksempelet (Figur 23) er det to eksempler på handlinger ”legg til” og ”erstatt periode”. ”Legg til” legger de valgte dataene inn i databasen, ”erstatt periode” legger inn og erstatter alle data for eksempel perioden april. Ved siden av pilen vil det også bli vist et tall. Dette tallet forteller hvor mange blodsukkerverdier som vil bli overført.



Figur 23: En løsning hvor en bruker direkte manipulasjon og direkte kombinasjon for å overføre data fra blodsukkerapparatet til programmet. I dette brukergrensesnittet kan en da dra målingene over objektet som representerer blodsukkermålingene i kalenderen. Ved kombinasjonen av blodsukkermålinger og kalenderobjektet vil brukeren få opp en egen meny over valg av mulige handlinger. Ved siden av pilen vil en få opp et tall, som viser hvor mange målinger som vil bli overført.

Denne skissen viser da hvordan en kan overføre synkroniseringsvisualiseringen en finner i iTunes over i et diabetesprogram. Skissen bruker da ikke et dialogvindu for å visualisere prosessen. Det blir brukt en metode som bygger på direkte manipulasjon.

I neste kapittel vil jeg også analysere hvordan denne skissen benytter teorier som kan knyttes til bedre brukbarhet.

6.3.3 Teoretisk analyse av skissen

6.3.3.1 Blodsukkerapparatet som et virtuelt objekt

I denne skissen blir blodsukkerapparatet visualisert som et virtuelt blodsukkerapparat. En kan si at et objektet som eksisterer i den virkelige verden, blir visualisert i brukergrensesnittet som et virtuelt objekt. En kan kategorisere det virtuelle blodsukkerapparatobjektet som en type metafor. Metaforen kan da kalles en blodsukkerapparat metafor. Brukeren ser da et objekt i brukergrensesnittet som minner om blodsukkerapparatet. En får da her, som en også får ved almanakkmetaforen i loggføringsfunksjonaliteten, et konsept som passer sammen. Konseptet er at et blodsukkerapparat inneholder blodsukkermålinger, slik det også kan beskrives i den virkelige verden.

En forskjell fra denne typen metaforer og andre metaforer er at denne blodsukkerapparatmetaforen er koblet direkte med det virkelige objektet det er en metafor for. Andre type metaforer, som skrivebordsmetaforen og almanakkmetaforen i loggføringsprosessen, har ikke en slik kobling. Det er for eksempel ikke en kobling mellom skrivebordsmetaforen og skrivebordet en for eksempel fysisk har datamaskinen på i den virkelige verden. En slik kobling i en skrivebordsmetafor ville vært meget unaturlig. Flytter en for eksempel en mappe i det virtuelle skrivebordet, ville også mappen blir flyttet på det virkelige skrivebordet i den virkelige verden. En slik kobling i skrivebordsmetaforen er da ikke reel og kan sies å være unaturlig. Når blodsukkerpratet blir beskrevet som en metafor er det en fysisk kobling mellom datamaskinen og blodsukkerapparatet, gjennom en egen kabel. Dette gjør at forandringer brukeren gjør på det virtuelle blodsukkerapparatobjektet, vil også kunne kobles til det virkelige blodsukkerapparatet. Sletter for eksempel brukeren en blodsukkermåling på det virtuelle blodsukkerapparatet kan også blodsukkermålingen bli slettet fra det virkelige blodsukkerapparatet. Dette er mulig siden datamaskinen og blodsukkerapparatet er koblet sammen med en kabel. Hvordan brukeren vil forstå denne koblingen er ikke så lett å forutsi.

For å belyse denne koblingen mellom det virtuelle objekt og det virkelige objektet, kan en se på det virtuelle objektet sin frembydelse. På samme måte som brukeren må lære seg et nytt instrument som den manipulerbare progresjonsbaren i loggføringsprosessen, kan en argumentere for at brukeren må lære seg at det er en kobling mellom det virtuelle objektet og det virkelige objektet. En må da også her prøve å bygge opp det Norman (1998) beskriver som opplærte konvensjon. Et argument for at brukeren kunne lære seg den manipulerbare progresjonsbaren er at en i direkte kombinasjon kan angre sine handlinger. Brukeren kan da utføre en handling for så å angre den rett etterpå. Dette gjør at brukeren kan utforske brukergrensesnittet. Dette kan da brukes som et argument for at brukeren kan lære seg denne koblingen mellom det virtuelle objektet og det virkelige. På en annen side skal ikke brukeren i dette tilfellet lære seg et instrument, men forstå en konseptuel modell. En kan se for seg at brukeren ved en feiltagelse sletter data på blodsukkerapparatet og oppdager ikke dette før nestegang en kobler til blodsukkerapparatet. Det er da ikke funksjonelt vanlig at et slikt program kan angre handlinger langt tilbake i tid. Det kan ta tid før brukeren oppdager at en har gjort en feil, siden feilen ikke nødvendigvis oppdages før neste gang en kobler til blodsukkerapparatet. Brukeren skal lære seg at det er kobling mellom det virtuelle og det virkelige objektet.

I forbindelse med den manipulerbare progresjonsbaren i loggføringsprosessen er det ikke plass til en beskrivende tekst ved siden av instrumentet. I denne skissen blir blodsukkerapparatet beskrevet og visualisert ved hjelp av to tekster og et ikon. Dette følger da Haramundanis (1996) råd om at ikoner ikke bør stå alene. Den ene teksten beskriver hele blodsukkerapparatobjektet ved hjelp av ordet ”kilder”. Videre blir selve objektet beskrevet med ordet ”blodsukkerapparatet”. Ikonet som er en del av objektet har i skissen samme utseende som det virkelige blodsukkerapparatet. Dette kan være med på å gi brukeren en forståelse av at det virtuelle blodsukkerapparatet er koblet med det virkelige blodsukkerapparatet. Ikonet er da med på visuelt koble det virtuelle blodsukkerappartet med det virkelige, siden de har samme utseende. Hvis ikonet var utformet som et generelt blodsukkerapparat kan denne koblingen bli mindre, noe som ikke er ønskelig. En kan da si at visualiseringen av det virtuelle blodsukkerapparatet er med på å øke frembydelsen. Gjennom teksten som utrykker kildevinduet, blodsukkerapparatet i entall og ikonet som er likt som blodsukkerapparatet, er dette med på visualisere koblingen mellom det virtuelle og det virkelige objektet. Dette kan også bidra til en forståelse av denne koblingen. Blodsukkerapparatikonet og teksten vil også kun være synlig i brukergrensesnittet når blodsukkerapparatet er koblet til. Dette kan være med på å bidra til en forståelse for at det virtuelle objektet representerer det virkelige blodsukkerapparatet.

6.3.3.2 Overføring ved direkte kombinasjon

Visualiseringen synkroniseringsfunksjonaliteten kan minne om det Holland og Oppenheim (1999) kaller direkte kombinasjon. Ønsker brukeren å overføre målinger kombinerer brukeren blodsukkermålinger med blodsukkerkalenderen. Ved kombinasjonen av disse to elementene vil brukeren få opp en listen over mulige operasjoner. Holland og Oppenheim (1999) beskriver at ved direkte kombinasjon kombineres to eller flere substantiver med et verb. I denne skissen er substantivene blodsukkermålingene og blodsukkerkalenderen. Brukeren får mulighet til å velge et verb fra listen over mulige operasjoner. Gjennom denne kombinasjonen får brukeren kontroll over hvilke objekter som blir overført til kalenderen.

Holland og Oppenheim (1999) bruker tre prinsipper om direkte kombinasjon. De beskriver at objekter av interesse skal være synlige, alle objekter må kunne behandles som interaksjonsobjekter og det må være mulig å gjøre en eller flere operasjoner ved kombinasjonene av et annet objekt. Skissen (Figur 23) utnytter ikke alle prinsippene fullt ut. Ut i fra skissen vil en kun få mulighetene til å velge en operasjon ved å kombinere blodsukkerverdier med blodsukkerkalenderen. Det vil for eksempel ikke visualiseres noen operasjoner når en kombinerer et blodsukkerverdiobjekt med en annen. En kan diskutere om dette burde vært mulig for brukeren. En må da se på relevansen ved en slik kombinasjon. For å gjøre en slik kombinasjon mulig, må programmet tilby en funksjon som kan utføres på kombineringen av to eller flere blodsukkerverdiobjekter. Dette gjør at en må i ytterste konsekvens tilføye programmet funksjonalitet som ikke relevant i forhold til bruken av programmet.

Holland og Oppenheim (1999) beskriver at det direkte kombinasjonsbegrepet er med på å bedre oversikten og tilgjengeligheten over mulige opperasjoner i brukergrensesnittet. I dette tilfellet kan begrepet tvinge frem opperasjoner som ikke nødvendigvis er relevante for brukeren, hvis prinsippene skal følges. På en annen side er begrepet med på å synliggjøre de tilgjengelige operasjonene ved en overføring av data fra blodsukkerapparatet til programmet. Avhengig av hvilken operasjon brukeren

velger fra menyen, kan brukeren styre hvordan dataene skal overføres, med hensyn til for eksempel behandlingen av duplikater. Teksten som står i operasjonslisten blir visualisert for brukeren slik overføringen kommer til å bli utført. Det er da mulig å gi brukeren en mengde forskjellige muligheter og valg gjennom å utvide operasjonslisten. Skulle brukeren fått samme mulighetene uten bruken av direkte kombinasjon, hadde bruken av dialogvinduer vært en mulig løsning. Holland og Oppenheim (1999) beskriver at en av fordelene ved direkte kombinasjon er at en slipper bruken av dialogvinduer.

6.3.3.3 Interaksjon

Dialogvinduet som diabetesprogrammene bruker gjennom synkroniseringsprosessen, kan minne om Normans (1998) sin eksekverings-evaluerings-løkke. En vil da møte på samme diskusjonen som ved bruken av et dialogvindu i loggføringsprosessen. Et dialogvindu er med på å låse brukeren i en interaksjon, og brukeren får ingen tilbakemelding før dialogvinduet er avsluttet.

I denne skissen er alle elementene i synkroniseringen visualisert som objekter og de kan videre manipuleres og kombineres ved hjelp av direkte kombinasjon. Dette gjør at en også her kan bruke Djajadiningrat et al. (2002) sine opperasjonsformål og en innbygd respons begreper på å beskrive denne interaksjonen. Som i loggføringsprosessen kan hvert objekt ha et opperasjonsformål og en innbygd respons. Hver blodsukkerverdi kan bli overført hver for seg til kalenderen. En kan da si at hvert blodsukkerverdiobjekt har et operasjonsformål om at objektet kan bli flyttet over til kalenderen. Blodsukkerverdiobjektene sin innebygde respons er da at objektet blir visuelt og fysisk kopiert over til kalenderen, noe som er en naturlig respons av en slik operasjon. Interaksjonen med synkroniseringsprosessen blir da delt opp i flere ledd, slik som i loggføringsprosessen. En bryter opp Norman sin eksekverings-evalueringsløkke i flere mindre deler. Det vil også i dette tilfellet, i likhet med loggføringsprosessen, være med på å minske sjansen for det Normans (1998) kaller feil ved evaluering. Brukeren vil på samme måte som i loggføringsprosessen få kontinuerlig tilbakemelding ved interaksjon i denne skissen. Tar for eksempel brukeren tak i en blodsukkerverdi og drar den over blodsukkerobjektet, vil brukeren få en respons på det Djajadiningrat et al. (2002) kaller bekreftelse på navigasjon. Brukeren vil få visualisert blodsukkermålingsobjektet under selve musepeker, som indikerer at brukeren har tatt tak i selve objektet. Som igjen gjør at brukeren får en kontinuerlig respons under interaksjonen. Dette gjør at det er en relasjon mellom operasjonen og responsen. Som beskrevet i loggføringsprosessen vil dette være med på å minske sansynligheten for feil ved evaluering. Brukeren får responsen kontinuerlig og kan oppdage en feil handling tidlig i interaksjonen. Dette er da i motsetning til en dialogboks hvor brukeren ikke kan oppdage feil før dialogboksen er avsluttet og brukeren får en respons på handlingen.

6.3.3.4 Sammendrag

I denne delanalysen har analysert en skisse som beskriver en synkroniseringsprosess. Jeg har beskrevet at blodsukkerapparatet kan sees på som en metafor, i likhet med almanakkmetaforen. Det er allikevel en forskjell, blodsukkerapparatmetaforen representerer også det virkelige blodsukkerapparatet. Dette er en kobling som ikke er vanlig for metaforene en kjenner fra tidligere. En kan da diskutere hvordan denne koblingen skal synliggjøres for brukeren. Jeg har da argumentert for at bruken av ikoner og tekst er med på å visualisere dette for brukeren. Samtidig som det virtuelle

blodsukkerapparatobjektet kun er synlig i brukergrensesnittet når blodsukkerapparatet fysisk er koblet til maskinen. Dette vil da være med på å visualisere koblingen mellom disse to objektene.

Videre har jeg beskrevet at overføringen mellom blodsukkerapparatet og programmet skjer ved hjelp av direkte kombinasjon. Når brukeren drar blodsukkerverdiobjekter over til kalenderen får brukeren opp en meny hvor brukeren kan velge en opperasjon. Dette er videre med på at en unngår bruken av et dialogvindu. Som i loggføringsprosessen er interaksjonen delt opp i flere små sykler. Dette er med på at en minsker risikoen for å få det Normans (1998) kaller feil ved evaluering

6.4 Kategorisering

Kategorisering av dataene i et program for diabetiker er med på å klargjøre dataene for en senere analyse. Kategoriene basere seg på tidsrom som ofte er knyttet til måltidene i løpet av døgnet. Dataene kan da tilknyttes et slikt tidsrom, for å gjøre en senere analyse mer inntresant. Eksempler på slike tidsrom kan være før og etter frokost, før og etter lunsj etc..

Utfordringen ved en slik kategorisering er å tilrettelegge programmet slik at brukeren kan kategorisere sine data på en rask og effektiv måte. Dagens diabetesprogrammer lar brukeren sette opp en standard mal med tider som definerer tidsrommene eller kategoriene (Figur 10 side: 42). Problemet med en slik standard mal er veldig lite fleksibel. Brukere som for eksempel ikke jobber til samme tid hver dag, har liten mulighet til å tilpasse programmet til egne spesielle behov.

Tilpassing av kategorier kan også ha en ulempe, det kan fort bli tidkrevende for brukeren. Må brukeren editere hver enkel loggføring til en kategori, kan dette fort bli tidkrevende. En har da to motsetninger hvor den ene er muligheten for å tilpasse kategoriene og den andre er tiden det tar å tilpasse en kategori. Utfordringen er å finne en visualisering som balanserer tidsbruk og tilpassning.

6.4.1 Regulering av avtaler iCal

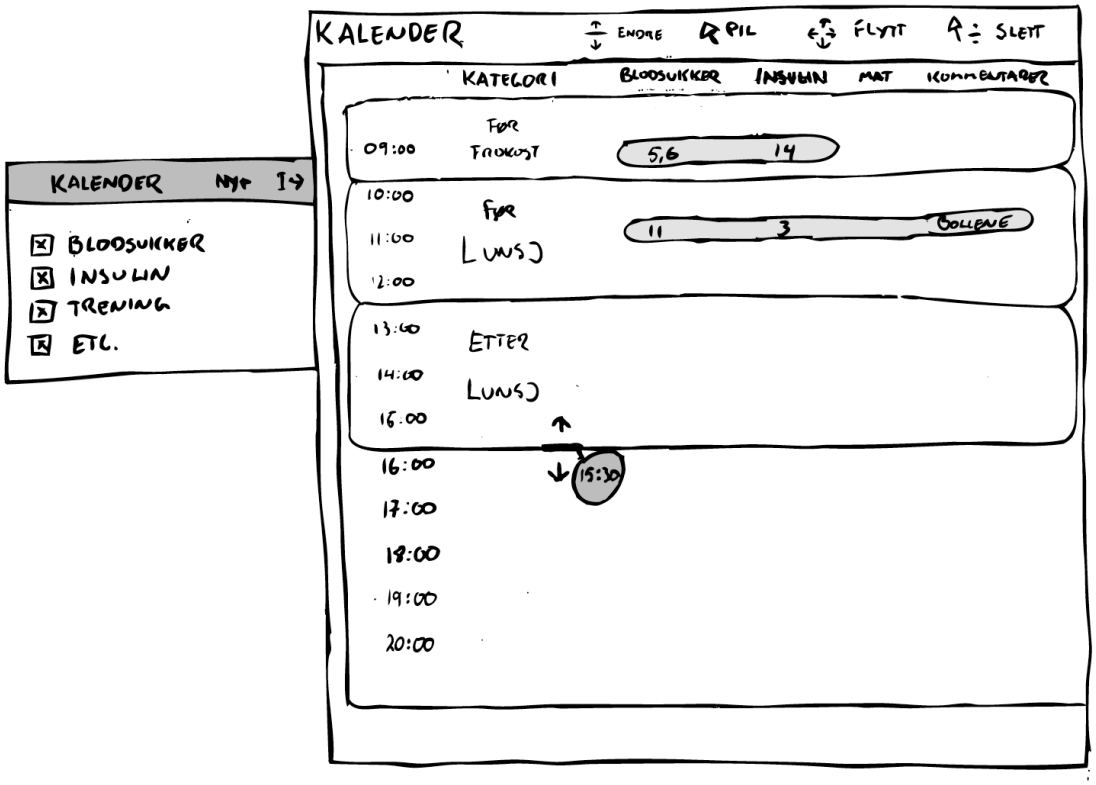
I iCal har en mulighet til å registrere avtaler som varer over et visst tidsrom. Det vil si at en avtale har et start og et slutt tidspunkt, i likhet med kategoriene en bruker i diabetesprogrammene. ICal visualiserer en avtale gjennom å vise et firkantet objekt i brukergrensesnittet. Objektet er da plassert i forhold til en y-akse som indikerer tidsrommet. Videre kan objektet manipuleres ved hjelp av musepekeren. Brukeren kan dra i kantene på objektet, ved hjelp av musepekeren, for å forlenge eller forkorte en avtale. En kan også flytte i hele objektet for å flytte en hel avtale til en annen tid.

Brukeren får i iCal visualisert avtalen på en annen måte en gjennom tall. Dette er i motsetning til diabetesprogrammene, som bruker tall eller klokkeslett for å visualisere kategoriene. I diabetesprogrammene må brukeren sette opp start og stopptidspunkt for kategoriene, lengden på avtalen blir da utrykt ved forskjellen i klokkeslettene. I iCal får brukeren se virtuelle objekter hvor størrelsen på objektet utrykker lengden til avtalen. Et objekt som er høyere enn et annet objekt, utrykker en forskjell i lengde på avtalen.

I skissen i neste kapittel har en prøvd å overføre iCal sin måte å visualisere tidsrom som objekter til et diabetesprogram. Videre vil også denne skissen analyseres i en delanalyse.

6.4.2 En skisse for editering av kategorier

Figur 24 viser en skisse på hvordan kategoriene kan representeres og manipuleres. Brukeren kan da editere og manipulere kategoriene i kalendervinduet. Kategoriene har i likhet med avtalene i iCal blitt visualisert som firkantede objekter. Brukeren kan da ta tak i øverste og nederste kant for å editere start- og slutt-tidspunkt for kategoriene. For å kunne editere kategoriene må brukeren velge et instrument som i dette eksempelet heter ”endre” og har en strek med to piler som ikon. Når brukeren endrer tidspunktet ved hjelp av ”endre”-instrumentet, vil klokkeslettet for det aktuelle start- eller slutt-tidspunktet vises ved siden av musepekeren.

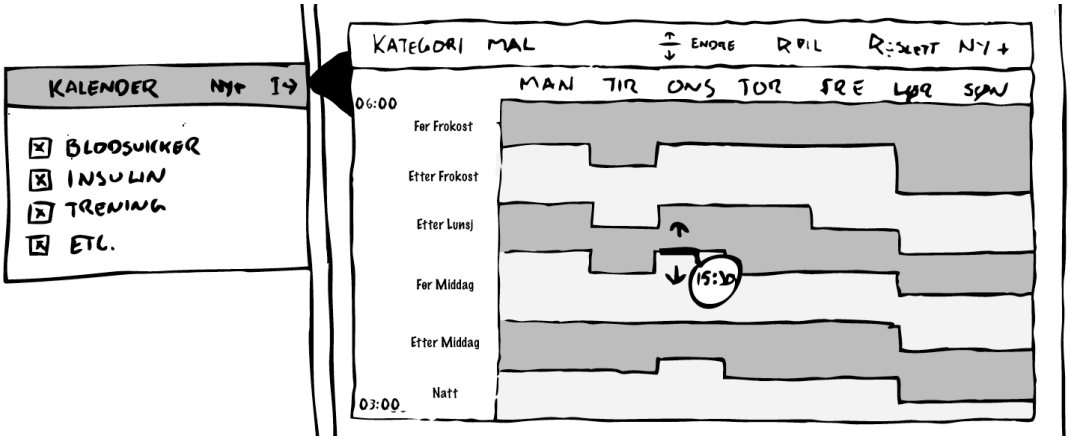


Figur 24: Brukeren kan editere kategoriene ved hjelp av et instrument og direkte manipulasjon.

Editeringen i Figur 24 viser hvordan brukeren kan tilpasse kategoriene i løpet av en dag. En kan allikevel være avhengig av å kunne sette opp en standard mal for kategoriene i løpet av en uke. Figur 25 viser en skisse av hvordan en standard mal kan visualiseres. Dette eksempelet viser et eget tilleggsvindu i programmet hvor brukeren kan editere den generelle malen for kategoriene. Vinduet vil bli visualisert når brukeren klikker på ikonet som er merket med en ”I” og en pil til høyre. Vinduet vil da legge seg oppå det eksisterende brukergrensesnittet med en gjennomsiktig effekt. Vinduet har ingen avslutningsknapp da brukeren når som helst kan klikke andre steder i brukergrensesnittet. Dette vinduet vil fungere på samme måte som visualiseringen av parametere i programmet Aperture, som er beskrevet i kapittel 5.3.2.1. Når brukeren

editerer i dette gjennomsiktige vinduet, vil eventuelle forandringer visualiseres umiddelbart i vinduet som ligger ”bak” dette egne gjennomsiktige vinduet.

I dette gjennomsiktige vinduet er kategoriene også visualisert som objekter. I overgangen mellom objektene kan brukeren editere start og slutt tidspunktet for kategorien, i likhet med editeringen av kategoriene i kalenderoversikten. Brukeren kan da sette opp forskjellige tider for hver kategori med hensyn til hvilken ukedag det er. Øverste i vinduet er det satt av plass til instrumenter, som kan brukes til å editere, slette og lage nye kategorier. En kan tenke seg at denne malen bestemmer hvordan kategoriene skal defineres i utgangspunktet. Hvis brukeren endrer tidene for en bestemt dag, vil denne typen endringer få presidens ovenfor den generelle malen.



Figur 25: Et standard oppsett av kategorier. Dette er utgangspunktet for kategoriene til hver dag, hvis ikke en dag har blitt spesifikt behandlet i kalenderen.

Denne skissen har da laget en visualisering av kategoriene som kan enkelt tilpasses. Det som var problemet med diabetesprogrammene var at en ikke kunne tilpasse kategoriene til hver dag. I denne skissen kan en kan dra i kantene på firkantene for å tilpasse tidsrommene til kategoriene. I tillegg er det mulig å sette opp en mal som gjelder for alle dager, som igjen kan sees på som en tidsbesparende funksjonalitet. Denne skissen har da en balanse mellom tilpassningsmuligheter og tidsbruk.

6.4.3 Teoretisk analyse av skissen

6.4.3.1 Representasjon gjennom objekter, ikke tall

I disse eksemplene visualisere kategoriene ved hjelp av objekter i stedet for tall. I motsetning til diabetesprogrammene hvor brukeren kan definere kategoriene ved hjelp av tall som klokkeslett. Norman (1998) bruker et begrep som kalles naturlig mapping, som er kunnskap om vedtatte sannheter i en kultur. Dette begrepet kan si noe om forskjellen mellom bruken av tall og objekter for i visualisere en kategori. Når kategoriene visualiseres ved hjelp av tall, får brukeren kun start og sluttidspunktet for kategorien. Varigheten av kategorien blir ikke visualisert. Ved bruken av objekter, vil brukeren kunne se start og sluttidspunkt, samtidig som varigheten til kategorien blir visualisert gjennom i dette tilfellet høyden på objektet. Dette kan en si er en form for kulturell mapping. Den kulturelle mappingen har sammenheng mellom høyden på objektet og varigheten til kategorien. Varigheten og høyden blir det Norman (1998) kaller additive dimensjoner. En økning i høyden kan mappes med en økning i

varigheten til kategorien. Brukeren kan da mappe høyden på objektet til varigheten til kategorien gjennom kulturell mapping.

Et annet begrep som kan brukes til å belyse dette er det Beaudouin-Lafon (2000) kaller graden av kompatibilitet. Beaudouin-Lafon (2000) bruker grad kompatibilitet for å beskrive egenskaper ved instrumenter. Graden av kompatibilitet beskriver forskjellen på inndata og utdata. Et eksempel kan brukes for å beskrive dette nærmere. En kan tenke seg et brukergrensesnitt hvor brukeren må taste inn klokkeslettene for kategoriene i et eget vindu eller i en egen del av programmet. Videre brukes disse verdiene til å vise kategoriene som objekter, slik som det er gjort i skissen (Figur 24). En kan da kategorisere klokkeslettene som inndata til programmet og objektene i kalenderen som utdata. Det vil da være en liten grad av kompatibilitet mellom disse dataene, siden de ikke er like i sin form. Hvis graden av kompatibilitet skulle vært stor, må inndata være på samme form som utdataene. I skissen (Figur 24) er kategoriene beskrevet som objekter både som inndata og utdata. Dette betyr at det er stor grad av kompatibilitet.

En kan da forklare bruken av objekter og ikke tall med Norman (1998) sin additive dimensjoner. Siden varigheten er en additiv dimensjon med lengden på objektet, kan en bruke et objekt for å representere lengden. Objektet vil også ha en stor grad av kompatibilitet siden brukeren angir tidsrommet på samme måte som tidsrommet er visualisert i brukergrensesnittet.

6.4.3.2 *Editering med instrumenter*

I skissen brukes det Beaudouin-Lafon (2000) kaller instrumenter for å manipulerer objektene. Disse instrumentene er av samme type som de som blir brukt i loggførings og synkroniseringsprosessen. Instrumentene brukes til å opere på kategoriobjektet, som Beaudouin-Lafon (2000) kategoriserer som domeneobjektet. Instrumentene i denne skissen er da av den typen som har lang aktiveringstid og lite plassbehov, siden brukeren må klikke på instrumentet for å aktivere det. Plassbehovet er lite siden instrumentet ikke er visualisert på kategoriobjektet.

En kan da diskutere hvordan brukeren får visualisert manipulasjonen som kan utføres på kategoriobjektet. Det vil si hvordan får brukeren visualisert at det er mulig å dra i objektene for å endre tidspunktet. Shneiderman (1998) beskriver hvordan direkte manipulasjon skal synliggjøre for brukeren hvilke handlinger som er mulige å gjennomføre med objektet. I dette tilfellet blir denne mulige handling kun synliggjort av instrumentet som heter ”endre”. Norman (1998) sitt frembydelses begrep sier noe om hvilke handlinger vi oppfatter når vi ser et objekt. I dette tilfellet mener Norman (1999) at frembydelsen bygger på tidligere lærte konvensjoner. Det betyr at brukeren vil forstå at det er mulig å ta tak i kanten på objektet, siden brukeren har lært dette i andre liknende situasjoner.

For å øke visualiseringen av funksjonaliteten til objektet, kunne en ha brukt et annet instrument. En kunne brukt et instrument som ikke har behov for å aktiveres. Det betyr at instrumentet måtte vært synlig i kantene på kategoriobjektene. Brukeren kunne da klikket eller manipulert direkte på instrumentet som er visualisert på kategoriobjektet. Dette medfører at plassbehovet til instrumentet er større, som igjen betyr at kategoriobjektet også har et større plassbehov. Ved at instrumentet er synlig på kategoriobjektet eller domeneobjektet til en hver tid, kan en si at en øker

visualiseringen av funksjonaliteten. Hvordan dette spiller inn på frembydelsen av objektet er avhengig av brukerens lærte konvensjoner (Norman 1999). Uansett vil avstanden mellom instrumentet og kategoriobjektet bli mindre, enn ved bruken av piler og aktiveringsprinsippet.

I skissen knyttes en boble med et klokkeslett inni med instrumentet som er merket ”endre”. Dette gjøres for å gi brukeren en tilbakemelding over det eksakte tidspunktet instrumentet befinner seg på. Uten denne boblen ville dette vært vanskeligere å bestemme nøyaktig tidspunkt, siden tidsskalaen befinner seg helt til venstre i bildet. En kan bruke Beaudouin-Lafon (2000) sin grad av indireksjon for å utdype dette, som også har blitt brukt til å beskrive den direkte manipulerbare progresjonsbaren i loggføringsprosessen. Som tidligere nevnt er graden av indireksjon avhengig av to aspekter: avstandsaspektet og tidsaspektet. Tidsaspektet beskriver hvor raskt instrumentet responderer ved handling. I dette tilfellet er det umiddelbart siden brukeren ser forandringen direkte fra objektet. Avstandsaspektet forteller noe om hvor lang avstand det er fra instrumentet til reaksjonen fra objektet. I dette tilfellet er riktignok reaksjonen fra avstanden til selve objektet meget kort i dette tilfellet, men en av faktorene brukeren justeres etter har en betydelig lenger avstand fra objektet. Brukeren må uten boblen justere etter skalaen til venstre i skjermbildet, dette er med på å øke graden av indireksjon. Siden boblen plasseres like ved siden av instrumentet vil avstanden synke betraktelig og graden av indireksjon vil bli vesentlig høyere. Brukeren får da ved hjelp av boblen den nødvendige informasjonen i kort avstand fra instrumentet.

6.4.3.3 Interaksjonen, dialogvindu kontra instrumenter

I denne skissen kan brukeren gjennom instrumenter og direkte manipulasjon forandre på kategoriens starttidspunkt, sluttidspunkt. Dette kan brukeren gjøre uten å forlate kalendervinduet, når kategoriene skal editeres kun for en dag. Ønsker brukeren å editere den generelle malen (Figur 25), må brukeren klikke på et ikon for å få synliggjort et vindu hvor disse parametrene kan editeres. Dette vinduet kan minne om et dialogvindu og kan forandre eller gi en annen interaksjonen med programmet. Et dialogvindu vil som tidligere beskrevet gi brukeren en type eksekverings-evaluerings løkke for interaksjonen. Det er allikevel flere faktorer ved visualiseringen av dette vinduet som gjør at det ikke nødvendigvis må forbindes med et dialogvindu. Vinduet i skissen har ingen avslutningsknapp, dette betyr at brukeren ikke blir låst til å kun operere med dette vinduet. Brukeren kan når som helst klikke andre steder på skjermen for å operere på andre objekter. Et typisk kjennetegn for dialogvinduer er at brukeren må avslutte dialogen for at endringene skal tre i kraft. I denne skissen vil endringene visualiseres umiddelbart, selv i vinduet som ligger ”bak” det gjennomsiktige vinduet.

Den sorte pilen i kanten på vinduet, peker på det objektet som vinduet er gjeldene for. I skissen peker da pilen på kalender objektet, noe som indikerer at parametrene en kan stille på i dette vinduet gjelder for kalender objektet. Beaudouin-Lafon (2000) beskriver instrumenter som et verktøy for å operere på et domeneobjekts atributter. I dette tilfellet er parametrene, den generelle malen for kategoriene, i vinduet atributter for kalenderobjektet. Kalenderobjektet kan sees på som et domeneobjekt. Vinduet kan da sees på som et stort instrument, som kan brukes til å kontrollere domeneobjektet. Det er tidligere forklart at Beaudouin-Lafon (2000) bruker faktorene aktiveringsbehov og plassbehov for å beskrive et instrument. Dette instrumentet eller vinduet har et

aktiveringsbehov, da en må klikke på et ikon for å få tilgang til instrumentet. Samtidig er plassbehovet relativt. Når instrumentet ikke er i bruk er plassbehovet lite, men når instrumentet er i bruk er plassbehovet vesentlig større. I skissen er da dette plassbehovet løst ved å gjøre vinduet gjennomsiktig. Videre kan en se på det Beaudouin-Lafon (2000) kaller graden av indireksjon, avstands- og tidsaspektet. Avstands aspektet for dette instrumentet er relativt liten. Vinduet visualiseres like ved objektet det kontrollerer, men selve kalenderobjektet gir ingen visuell forandring når eventuelt brukeren forandrer på parametrene. Disse forandringene vil finne sted ”under” og ved siden av vinduet, siden dette er gjennomsiktig. Dette er med på å skille mellom avstanden til domeneobjektet og avstanden til responsen fra domeneobjektet. Avstanden til selve domeneobjektet er liten, og avstanden til responsen fra domeneobjektet er større. Tidsaspektet er liten ved bruken av dette instrumentet, forandringene i domeneobjektet vil finne sted umiddelbart etter en forandring i instrumentet. Denne umiddelbare responsen fra domeneobjektet eller kalenderobjektet kan beskrives med Djajadiningrat et al. (2002) sitt innbygd respons begrep. Dette er da med på å dele interaksjonen med dette objektet opp i flere små deler, noe som igjen gjør at det har andre egenskaper enn et dialogvindu. Ved bruken av dette instrumentet vil brukeren få tilbakemeldinger umiddelbart ved forandringen av parametere i vinduet. Dette er vesentlig forskjelle fra bruken av et dialogvindu, hvor ikke brukeren få noe tilbakemelding før dialogvinduet avsluttes.

Når brukeren får en umiddelbar tilbakemelding vil dette også redusere muligheten for det som er tidligere beskrevet som feil ved evaluering. Normans (1998) beskriver at denne feilen oppstår når brukeren får et annet resultat enn forventet. Hvis for eksempel brukeren forandrer en kategori i instrumentet, vil brukeren kunne se en forandring umiddelbart i vinduet som ligger ”under”, som viser en dagsoversikt over loggførte data. Dette kan være med på å forminske sannsynligheten for en feil ved evaluering, siden brukeren får responsen på sin handling umiddelbart.

6.4.3.4 Ikonet til oppsettet av malen

Det gjennomsiktige instrumentet blir aktivisert ved at en klikker på et eget ikon. Dette ikonet har ingen beskrivende tekst. Som beskrevet i loggføringsprosessen og synkroniseringsprosessen mener Haramundanis (1996) at et ikon ikke bør stå alene uten en tekst for å ha full betydning. I dette eksempelet er det liten plass til en tekst. Ikonet er plassert på et annet objekt, kalenderobjektet. En kan da argumentere for at en tekstelig beskrivelse vill bli for plasskrevene. Ikonet er bilde av bokstaven ”I”. Bokstaven gir utrykk for ordet ”innstillinger”. Videre er det bilde av en pil som gir utrykk for at det vil visualiseres et vindu der denne pilen peker. Preece (2002) beskriver at en må ta hensyn til kulturelle og kontekstspesifikke spørsmål når en lager ikoner. I dette tilfellet er bokstaven ”I” en kulturell betegnelse, bokstaven er en forkortelse for et norsk ord. I et land med et annet språk ville det da vært naturlig med andre bokstaver i dette ikonet.

Dette ikonet blir brukt som et generelt ikon for å få frem et instrument som kan regulere preferanser eller atributter til domeneobjektet. Det betyr at det blir brukt flere steder i skissen. En av de kanskje viktigste oppgaven til utseendet til et ikon, er å skille ut ikonet fra andre ikoner (Preece 2002). Samtidig mener Haramundanis (1996) at ikoner er vanskelige å lære men lette å kjenne igjen. En mulig løsning for å lære brukeren betydningen av ikonet kan være å vise en tekstelig beskrivelse av ikonet når brukeren holder musepekeren over ikonet. Da vil nye brukere kunne lese betydningen,

samtidig som at den beskrivende teksten ikke tar opp plass i brukergrensesnittet. En ulempe ved dette er at det bygger på det Norman (1999) kaller innlærte konvensjoner. For en som ikke kjenner denne konvensjonen vil ikke kunne ha nytte av hjelpen ved å holde musepekere over ikonet. En annen løsning på denne problematikken vil være å inkludere et eget instrument i brukergrensesnittet, som har til hensikt å vise slike hjelpetekster. Dette instrumentet kunne vært et aktiveringsinstrument. Brukeren aktiverer da instrumentet for å så bruke instrumentet på et objekt i brukergrensesnittet. Brukeren vil da få visualisert hjelpeteksten. Fordelen ved dette fremfor å ha en skjult hjelpetekst, er at brukeren får visualisert et instrument som er synlig i brukergrensesnittet hele tiden. Hjelpeteksten er da ikke en ”skjult” funksjonalitet.

En kan da skissere tre forskjellige løsninger på å gi brukeren en tekstlig beskrivelse av ikonet. En kan plassere teksten ved siden av ikonet, teksten kan komme opp som en boks ved siden av musepekeren eller en kan lage et eget instrument som kan visualisere teksten. Hvilken løsning som egner seg best er avhengig hver enkelt situasjon. I denne skissen er det lite plass ved siden av ikonet og en løsning med musepekeren eller ved hjelp av et verktøy, kan være en fornuftig løsning. Uansett mangler denne skissen en tekst som beskriver ikonet. Dette kan da medføre at ikonet i følge Haramundanis (1996) blir vanskeligere å lære.

6.4.3.5 Sammendrag

I denne delanalysen har jeg analysert en skisse som visualiserer kategoriseringsprosessen. Kategoriene i skissen blir vist ved hjelp av firkantete objekter. Objektenes størrelse kan knyttes til varigheten av kategorien, som igjen kan forklares ved det som Norman (1998) beskriver som kulturell mapping og additive dimensjoner. De firkantede objektene blir også brukt til å angi lengden på kategoriene. En har da en stor grad av kompatibilitet siden inndata og utdata visualiseres på samme form.

Videre har jeg beskrevet hvordan denne skissen bruker instrumenter i brukergrensesnittet. Deretter har jeg diskutert hvordan instrumentet frembyr sin funksjonalitet. Aktiveringsprinsippet og plassbehovet til instrumentene kan da knyttes til hvordan instrumentets frembydelse er ovenfor brukeren. Ved at instrumentet ikke har et aktiveringsprinsipp og et større plassbehov, kan instrumentet bruke mer plass på å visualisere instrumentets funksjonalitet for brukeren.

I denne skissen blir det presentert et eget vindu for hvor brukeren kan legge inn en standard mal. Dette vinduet kan minne om et dialogvindu. Jeg har argumentert for at dette vinduet skal sees på som et instrument. Det er flere aspekter ved vinduet som tilsier at det ikke er et dialogvindu. En av hovedaspektene er at det ikke finnes en avslutningsknapp slik at brukeren kan forlate interaksjonen med vinduet når brukeren selv måtte ønske det. Alle parameterforandringer i vinduet vil også gi brukeren umiddelbar respons fra vinduet som ligger under selve programmet. Dette gjør at interaksjonen blir delt opp i flere mindre sykler.

Jeg har også beskrevet hvordan skissen ikke har en tekstlig beskrivelse av ikonet som brukes til å aktivisere det gjennomsiktige instrumentet. Jeg har beskrevet noen løsninger på hvordan dette kan gjøres. Siden ikonet mangler en beskrivende tekst kan det i følge Haramundanis (1996) blir vanskeligere å lære.

6.5 Analyse av data

I kapittel 3.3.1.4 argumenteres det for at analysefunksjonaliteten i et program for diabetikere må være fleksibel. Brukeren bør få tilgang til en analyse som hjelper brukeren til en bedre behandling. En er da avhengig av at programmet har en analysefunksjonalitet som brukeren kan få utbytte av. En fremgangsmåte for dette er at programmet lager en rekke forskjellige ferdigdefinerte analyser, med et utvalg av forskjellige data og visninger av disse. En kan for eksempel vise et histogram over blodsukkerverdier før frokost de to siste ukene eller en kan vise et kakediagram over hvor mange verdier som var over f.eks.10 mmol/l den siste måneden. Det finnes en rekke kombinasjoner av data og presentasjoner av resultatet. Et problem kan være at når programmene kun har ferdigdefinerte analyser får brukeren et begrenset utvalg av analyser. Det er ikke sikkert brukeren finner disse interessante nok til sitt eget behov. Programmet kan da eventuelt ha en stor mengde med forskjellige ferdigdefinerte analyser, men dette kan igjen fort bli uoversiktlig. En løsning på dette kan være å la brukeren definere preferansene for analysene selv. Hvis brukeren kan bestemme preferansene selv, er det stor sannsynlighet for at brukeren får en analyse som er tilpassets brukerens eget behov. Utfordringen ved at brukeren kan bestemme preferansene selv, er at kompleksiteten til programmet kan øke. En er da avhengig av å få visualisert dette på en god og brukervennlig måte.

6.5.1 Analysen i diabetesprogrammene og smarte album i iTunes

Det er forskjellig hvordan diabetesprogrammene lar brukeren bestemme preferansene rundt analysefunksjonaliteten. I noen av programmene kan ikke brukeren tilpasse analysen på noen som helt måte, brukeren må da finne en av de ferdigdefinerte analysene som passer best til brukerens behov. Andre programmer lar deg velge noen forskjellige preferanser. Programmet SiDiary skiller seg mest ut av de tre ved å la brukeren bestemme tidsrom for hvilke data som skal analyseres, hvilken kategori som skal analyseres og hvilken visningsform. Brukeren får her en del forskjellige muligheter til å lage en egne tilpassede analyser. Problemet med SiDiary er at brukeren må sette alle preferansene hver gang en vil se på analysen. Brukeren har ingen mulighet til å lagre preferansene for analysen, slik at brukeren kan enkelt gå tilbake til en egendefinert analyse.

SiDiary har også en egen type analyse som heter trender. Trender vil i dette tilfellet være forandringer i statistiske data. En slik oversikt kan være en nyttig funksjonalitet for brukeren. En vanlig analyse vil ta utgangspunktet i de lagrede dataene og presentere en fremstilling av denne utførte analysen. En trendanalyse vil ikke ta utgangspunktet i dataene som brukeren har lagt inn, men vil ta utgangspunkt i resultatet fra analysene. En kan si at en analyse av trender abstraherer seg over en vanlig analyse av dataene. Fordelen med dette er at brukeren veldig lett kan se forandringer i sin egen situasjon. Brukeren slipper for eksempel å notere ned gjennomsnittet for blodsukkeret for å kunne sammenligne det med den samme analysen neste måned. En slik trend analyse kan være nyttig funksjonalitet for et program for diabetikere.

Problemet med diabetesprogrammene er da at brukeren ikke kan bestemme og lagre preferansene for analysene selv. Brukeren må velge blant mange forskjellige

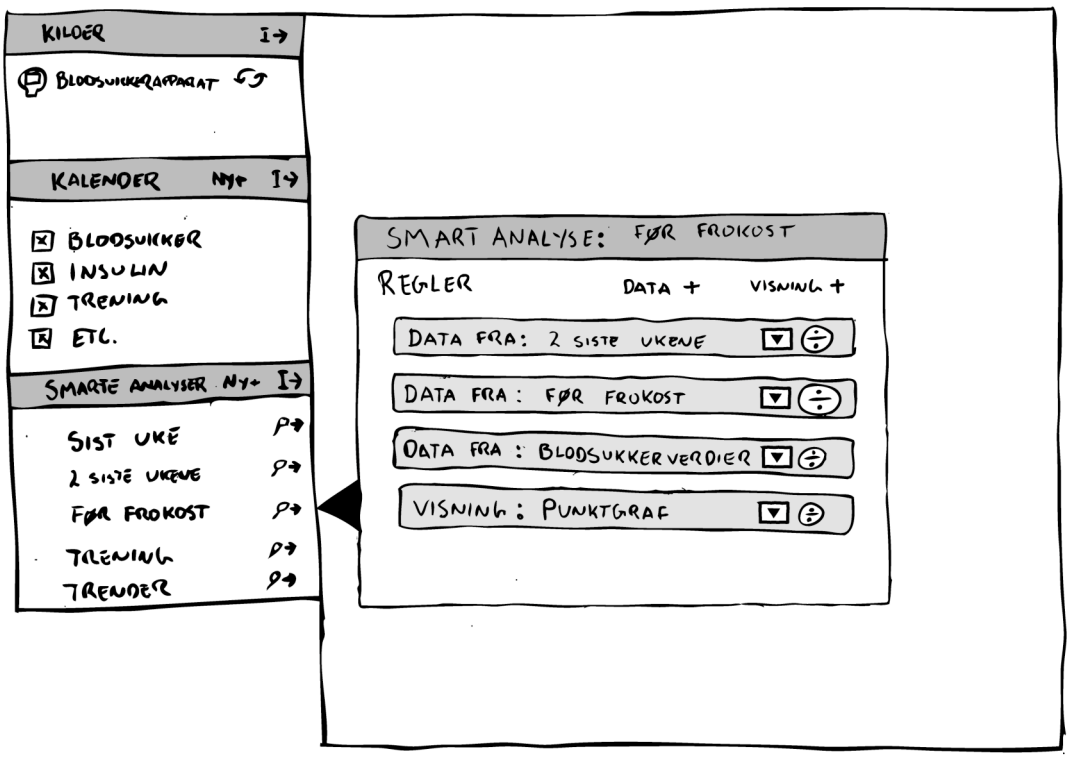
ferdigdefinerte analyser. Videre har SiDiary et trendanalyse konsept som kan virke fornuftig i et slikt analyseprogram. Hvordan kan en så få visualisert mulighetene for å bestemme preferansene og en trendanalyse i et diabetesprogrammene? ITunes har et konsept som kan virke interessant i denne sammenhengen.

ITunes sin funksjonalitet ”smarte album”, kan kobles til organiseringen og visualiseringen av slike analyser. I ITunes har brukeren mulighet til å definere et søk som kobles til et mappeobjekt i brukergrensesnittet. Denne funksjonaliteten kalles en smart mappe og har blitt beskrevet i kapittel 5.3.2. En smart mappe er et standardisert søk, hvor brukeren selv bestemmer preferansene for søket. Det standardiserte søket blir så visualisert gjennom brukergrensesnittet som en mappe. Dette gjør at brukeren får et objekt å forholde seg til i brukergrensesnittet. Innholdet i dette objektet er da resultatet av det standardiserte søket. Brukeren kan bestemme preferanser for dette objektet, som igjen er regler for hvordan søket skal gjennomføres. Selve søket blir skjult for brukeren gjennom mappeobjektet. Denne måten å lage objekter av søk, kan overføres til diabetesprogrammene. En kan definere det en kan kalle for ”smarte analyser”, som tillater brukeren å standardisere en analyse.

6.5.2 En skisse for generering av analyser

Figur 26 viser et eksempel på hvordan smarte spillelister kan kombineres med et program for diabetikere. I dette eksempelet har funksjonen fått navnet smarte analyser. Brukeren kan lage nye smarte analyser i et vindu til venstre. Når brukeren klikker på ”ny+” knappen vil det vises et nytt smart analyseobjekt i listen. Brukeren kan så editere reglene for det ferdigdefinerte søket ved å klikke på ikonet til høyre for navnet til den smarte analysen. Det vil da få opp et preferansevindu som vil legge seg oppå det eksisterende vinduet med en gjennomsiktig farge. Dette vinduet vil fungere på samme måte som visualiseringen av parametere i programmet Aperture, som er beskrevet i kapittel 5.3.2.1, og i kategoriseringsprosessen. Vinduet vil også her ha en liten pil i kanten av vinduet som peker på hvilket smart analyseobjekt dette vinduet er gjelden for.

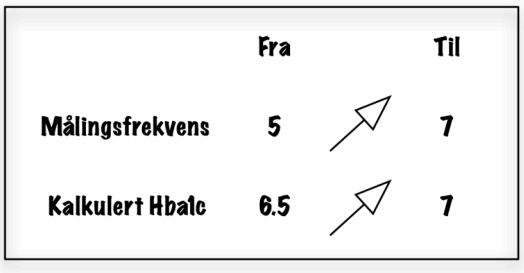
I vinduet kan brukeren bestemme hvilke regler som skal gjelde for den smarte analysen. Ved å legge til en ny dataregel (”data +” knappen) kan brukeren bestemme hvilke data som skal brukes, og ved en ny visningsregel (”visning +” knappen) bestemmer brukeren hvordan analysen skal visualiseres i brukergrensesnittet. Det dannes da nye regelobjekter som vises i preferansevinduet. Brukeren kan så videre velge parametere fra en nedtrekksmeny for å gi parametere til et regelobjektene. Brukeren får tilgang til nedtrekksmenyen ved å klikke på firkanten med en pil som peker nedover. Videre kan brukeren slette en regel ved å klikke på minusknappen.



Figur 26: En mulig måte å sette preferansene til en smart analyse. Vinduet vil legge seg oppå det eksisterende vinduet med en gjennomsiktig farge.

Når brukeren klikker på det smarte analyseobjekt vil resultatet av analyse vises i hovedvinduet til høyre i skjermbildet. Brukeren får da presentert den analysen som er satt opp i preferansevinduet til det tilhørende smart analyseobjektet.

I denne skissen kan en også lage en analyse av trender som en kan gjøre i SiDiary. Ønsker brukeren å lage en smart analyse som viser trender i behandlingen, kan brukeren for eksempel velge å hente data fra hele databasen og vise det som trender. En kan da bruke piler for å visualisere en oppgang eller nedgang i en tallverdi (Figur 27).



Figur 27: Et eksempel på hvordan trendene kan vises. Pilene kan her mappes med en oppgang eller nedgang i tallverdien slik at en unngår problematikken rundt mapping slik som i SiDiary programmet.

6.5.3 *Teoretisk analyse av skissen*

I denne skissen kan brukeren selv definere hvordan analysene skal utføres og vises. Brukeren får mange muligheter til å skreddersy analysedelen av programmet til sitt

eget behov. På samme måte som de smarte mappene i iTunes har gjort objekter av et standardiserte søk, har skissen gjort objekter av en standardisert analyse. En større analysefunksjonalitet er her skjult for brukeren og brukeren får da tilgang til denne funksjonaliteten gjennom objektene.

6.5.3.1 Instrumenter som domeneobjekt og visualisering

Det smarte analyse objektet kan kategoriseres som det Beaudouin-Lafon (2000) kaller et domeneobjekt. Dette domeneobjektet kan en så bruke instrumenter på, siden Beaudouin-Lafon (2000) beskriver instrumenter som opererer på et objekt ved å forandre deres atributter. Vinduet som kommer opp ved siden av forandrer reglene for analysen og kan derfor anses som et instrument. Dette vinduet og instrumentet er helt av samme typen som det instrumentet som ble beskrevet i analysen av kategorisering i kapittel 6.4.3.3. Vinduet kan da anses som et instrument og ikke som et dialogvindu. Som beskrevet i kategoriseringsprosessen i kapittel 6.4.3.3 er det ingen avslutningsknapp i dette vinduet. Vinduet operer også direkte på det smarte analyseobjektet. Brukeren vil da få kontinuerlig tilbakemelding når en forandrer på parametrene i instrumentet. I praksis betyr dette at brukeren kan til enhver tid se resultatet av de gjeldene parametrene i vinduet ”bak” det gjennomsiktige instrumentet. Brukeren kan da se hvilke resultater gjeldene kombinasjon av parametrene vil gi.

Det gjennomsiktige instrumentet vil inneholde egne instrumenter. Dette beskriver Beaudouin-Lafon (2000) som at instrumenter i seg selv kan bli domeneobjekter for andre instrumenter. Det gjennomsiktige vinduet inneholder egne regelobjekter. Disse objektene kan da videre manipuleres ved hjelp av andre instrumenter. Regelobjektene blir da en form for domeneobjekter. På selve regelobjektet finner en to forskjellige instrumenter, en nedtrekksmeny og en knapp med et minus tegn. Nedtrekksmenyen forandrer på regelobjektet sine preferanser, og minusknappen opererer på hele objektet ved å slette selve regelobjektet. Ser en på de prinsippene Beaudouin-Lafon (2000) kaller plassbehov og aktiveringsbehov. Har begge disse instrumentene et plassbehov og en trenger ikke å aktivere instrumentet før det benyttes. Instrumentene er plassert på selve objektet og må være synlig hele tiden. Ser en på det Beaudouin-Lafon (2000) kaller graden av indireksjon, som igjen deles opp i avstands- og tidsaspektet. Har begge instrumentene et lite avstandsaspekt og et lite tidsaspekt. Avstandsaspektet er lite siden instrumentene er plassert på objektet det kontrollerer. Tidsaspektet er lite siden brukeren vil få en reaksjon fra objektet umiddelbart.

En kan diskutere om hvordan brukeren vil forstå funksjonaliteten til disse to instrumentene som er plassert på regelobjektet. En kan si at minusknappen bygger på det Norman (1998) kaller kulturell mapping. Som beskrevet i kategoriseringsprosessen bygger kulturell mapping på kulturelle standarder. Minustegnet forteller oss at noe skal trekkes fra. I denne skissen brukes minustegnet til å fjerne et eksisterende regelobjekt. Frembydelsen til dette tegnet vil da gjennom kulturell mapping ha en betydning om at noe skal fjernes. Nedtrekksmenyen bygger på det Norman (1999) kaller oppfattet frembydelse, brukeren forstår et brukergrensesnitt gjennom opplærte konvensjoner. Nettrekksmenyer er et standard element i et vanlig WIMP og skrivebordsbrukergrensesnitt. En kan da anta at brukeren har lært bruken av en slik nedtrekksmeny fra andre liknende situasjoner.

Når brukeren skal lage nye regelobjekter må brukeren benytte seg av ”data +”-knappen og ”visning +”-knappen, som beskrevet i skissen. Når brukeren klikker på denne knappen vil det da automatisk komme opp et nytt objekt under disse knappene. En kan her diskutere hvilken sammenheng denne knappen har med objektene den lager. Er det en naturlig konsekvens at et nytt objekt blir laget når en trykker på en knapp?

Disse to knappene kan sees på som instrumenter. En kan bruke det Beaudouin-Lafon (2000) kaller graden av kompatibilitet for å belyse problematikk ved disse instrumentene. Som tidligere beskrevet, sier graden av kompatibilitet noe om hvordan inndata og utdata er kompatible i forhold til hverandre (Beaudouin-Lafon 2000). Inndata i denne situasjonen er et klikk på en knapp med et ikon. Utdata i denne situasjonen er et helt nytt regelobjekt. Dette kan kategoriseres som en lav grad av kompatibilitet. En knapp og et ikon er ikke det samme som et regelobjekt. Dette beskriver at det er lite sammenheng mellom det å klikke på en knapp og at det blir skapt et nytt objekt. Dette kan få betydning for det Norman (1998) kaller frembydelse. Visualiseringen og frembydelsen av instrumentet, har liten visuell sammenheng med responsen. Som tidligere beskrevet bruker Djajadiningrat et al. (2002) begrepet operasjonsformål om den informasjonen brukeren får før selve operasjonen og innebygd respons om informasjonen brukeren får etter en operasjon. Beaudouin-Lafon (2000) sitt kompatibilitetsbegrep viser da at det er en forskjell i formen operasjonsformålet blir visualisert og hvordan den innbygde responsen blir visualisert. Visualiseringen av operasjonsformålet er et ikon og en knapp mens visualiseringen av responsen er et eget regelobjekt, med tilhørende instrumenter. Løsningen i skissen bygger da på det Norman (1999) kaller oppfattet frembydelse. Brukeren må da i en tidligere sammenheng ha lært at bruken av et ikon kan lage et nytt objekt et annet sted i brukergrensesnittet.

En mulig løsning på å forbedre kompatibiliteten mellom operasjonsformålet og den innebygde responsen, kan være å bruke en form for direkte kombinasjon. Knappen og ikonet som lager et nytt objekt, kan være visualisert som et tomt regelobjekt. Brukeren må da dra det tomme regelobjektet ned til listen hvor de regelobjektene som er i bruk blir visualisert. Ved en kombinasjon av et tomt regelobjekt og listen, vil et nytt regelobjekt bli lagt til listen. Dette vil gjøre visualiseringen av operasjonsformålet mer kompatibelt med den innebygde responsen. Denne løsningen bygger også på det Norman (1999) kaller oppfattet frembydelse. Men kompatibiliteten er bedret. Brukeren må i dette tilfellet ha lært en konvensjon om at en kan ta tak i objekter i brukergrensesnittet for å så kunne dra visuelt til en annen del av programmet.

6.5.3.2 Direkte manipulasjon erstatter komplekst språk

Brukeren kan i skissen legge inn regler i den smarte analysen. Regelobjektene danner grunnlaget for selve analysen. Slike regler kan minne om for eksempel spørringer en kan gjøre opp mot en database, som igjen er en form for programmeringsspråk. En slik regellogikk kan fort bli meget avansert og kreve at brukeren forstår en avansert konseptuel modell.

Shneiderman (1998) beskriver at direkte manipulasjon skal erstatte et slikt komplekst regelspråk, og erstattes med synlige objekter. I skissen er et slikt språk erstattet med synlige objekter. Men en kan diskutere om disse objektene gir brukeren nok forståelse av den underliggende konseptuelle modellen. Et eksempel på dette er hvilken

betydning rekkefølgen av regelobjektene har i skissen (Figur 26). En kan da diskutere hvilken logikk det er mellom regelobjektene. Er det for eksempel en logisk ”og” eller en logisk ”eller”? Hva skjer hvis brukeren lager en regelkombinasjon som ikke er logisk riktig? En kan si at i dette eksempelet har direkte manipulasjon laget objekter av regler, men direkte manipulasjon har ikke laget objekter av avansert logikk. Dette stiller krav til programmets underforeliggende funksjoner. Disse funksjonene er da ikke visualisert i denne skissen.

En mulig fremgangsmåte for å visualisere logikken er å kun vise de valg som er gyldige. En kan da for eksempel gjøre det umulig for brukeren å lage en visningsregel før en har laget en regel om hvilke data som skal brukes. En kan altså ikke klikke på ”visnings +” knappen før en har trykket på ”data +” og lage en regel for hvilke data som skal velges. En kan da tenke seg at ”visnings +” knappen er gjort diffus.

En annen mulighet er å la brukeren lage de objektene en måtte ønske og at det ligger en logikk i programmet som velger rekkefølgen på reglene. En kan tenke seg at brukeren lager en visningsregel før en lager en regel over hvilke data en skal bruke. Programmet må da ha en logikk som utfører dataregelen før visningsregelen. Denne løsningen kan være mer knyttet til et av prinsippene rundt direkte manipulasjon enn den andre. I direkte manipulasjon skal alle handlinger være syntaktisk riktige (Shneiderman 1998). En bruker skal ha mulighet til å gjøre alle mulige valg i brukergrensesnittet, men en trenger ikke nødvendigvis å få et resultat. I dette tilfellet må alle mulige regler være mulig å sette opp, men hvis ikke programmets bakenforliggende logikk klarer å tolke reglene vil ikke brukeren få et resultat. Siden hele dette vinduet er et instrument med et lite tidsaspekt, vil brukeren umiddelbart se resultatet av forandringer i instrumentet. Dette bygger igjen på prinsippene rundt direkte manipulasjon, om at handlinger skal gi en kontinuerlig tilbakemelding (Shneiderman 1998).

6.5.3.3 Pilene i trendanalysen

Figur 27 viser hvordan skissen kan vise en analyse av trender. Denne skissen bruker piler for å visualisere en oppgang eller nedgang i tallverdien. SiDiary har visualisert denne trendanalysen på en litt annen måte (Figur 12 s.44). Pilene i SiDiary er på samme måte som i skissen koblet med en tallverdi. Pilen peker oppover hvis tallverdien øker og peker nedover hvis tallverdien avtar. Forskjellen er at ved siden av pilen er det ansiktstrykk som visualiserer status for selve behandlingen. Et smilende ansikt er godt og et surt ansikt er dårlig. Disse ansiktene kan gi opphav til forskjellig tolkning av pilene.

En kan bruke Normans (1998) kulturelle mappingbegrep for å forklare dette nærmere. En kan da koble smilende ansikt med bra og surt ansikt med dårlig. Dette er hva Normans (1998) kaller additive dimensjoner. Videre kan det være naturlig å koble en pil som peker oppover med de samme additive dimensjonenes. En pil som peker oppover kan bety godt og en pil som peker nedover kan bety dårlig. Problemet her er at pilene skal knyttes til en annen additiv dimensjon, en pil som peker oppover skal kobles med at tallverdien øker, og en pil som peker nedover skal kobles med en tallverdi som avtar. Det kan da være uklart hvilke additive dimensjoner som skal kobles med hverandre. En pil som pekere oppover skal ikke nødvendigvis kobles med positivt.

I skissen bruker en ikke samme løsning som i SiDiary programmet. Pilene kan kun kobles til en additiv dimensjon. Tallverdien øker når pilen peker oppover og tallverdien avtar når pilen peker nedover.

6.5.3.4 Sammendrag

I denne delanalysen har jeg analsert en skisse som visualiserer analyseprosessen til et diabetesprogram. Vinduet som brukes til å visualisere reglene til en smart analyse er av samme typen gjennomsiktig vindu som blir brukt i kategoriseringsprosessen. Dette vinduet kan sees på som et instrument og ikke et dialogvindu. Instrumentet er koblet til et smart analyseobjekt og bestemmer parametrene for dette objektet. Det gjennomsiktige instrumentet kan også sees på som et domeneobjekt i seg selv. Dette domeneobjektet inneholder da regelobjekter som igjen manipuleres av instrumenter. Jeg har diskutert hvordan disse instrumentene blir visualisert ovenfor brukeren. Minusknappen bygger på kulturell mapping. Nedtrekksmenyen bygger på opplærte konvensjoner, brukeren kjenner denne typen instrumenter fra liknende situasjoner. De to knappene som lager nye regler har et stort kompatibilitetsaspekt. Det er en liten sammenheng mellom visualiseringen av knappen og resultatet av operasjonen denne knappen gjennomfører. En mulig løsning på dette er å la brukeren dra et tomt regelobjekt til listen av regelobjekter for å generere et nytt objekt.

I skissen erstatter direkte manipulasjon et mer komplekst kommandospråk. Reglene i instrumentet som bestemmer preferansene for den smarte analysen, erstatter et mer avansert språk. Problemet er at disse regelobjektene visualiserer ikke kompleksiteten til et avansert søk. En må lage funksjonaliteten i programmet slik at gyldige kombinasjoner av regelobjekter gir resultater tilbake til brukeren. Kombinasjoner som ikke gir et gyldig resultat vil da ikke gi et resultat.

I skissen brukes en liknende visualisering av en trendanalyse som en finner i programmet SiDiary. I skissen er de smilende ansiktene fjernet og det er kun piler som visualisere forskjellen i tallverdien. Problemet med de smilende ansiktene er at de bidrar til at en kan tolke pilenes retning på to forskjellige måter. En kan si at de kan knyttes til hver sin additive dimensjon.

6.6 Sikkerhetskopi av data

I et program for diabetikere trengs en funksjonalitet for å lagre sikkerhetskopier av alle dataene sine. Dataene som brukeren legger inn i programmet har stor verdi for brukeren og et eventuelt tap av disse vil ha betydning for behandlingen av brukerens diabetes.

For å gjøre en slik funksjonalitet brukervennlig for brukeren, er visualiseringen av funksjonaliteten en viktig faktor. Visualiseringen må kunne gi brukeren en forståelse av den konseptuelle modellen av selve sikkerhetskopiprosessen. Hvis brukeren kommer i den situasjonen at data må gjenopprettes, er det viktig at brukeren har en forståelse for hvordan denne prosessen utføres.

6.6.1 Forskjellen mellom sikkerhetskopi i diabetesprogrammene og Aperture

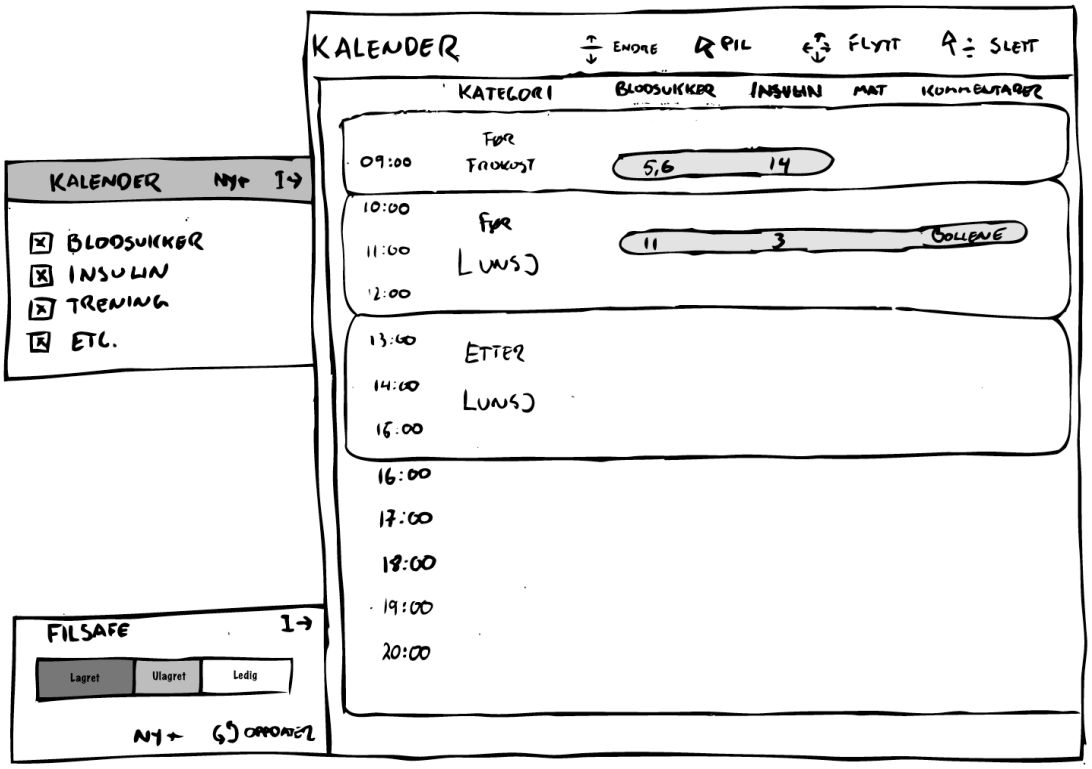
I diabetesprogrammene som støtter sikkerhetskopiering, blir sikkerhetskopifunksjonaliteten visualisert gjennom et menyvalg. Gjennom menyen får brukeren tilgang til en dialogboks, hvor brukeren kan bestemme hvor i filsystemet til operativsystemet sikkerhetskopien skal lagres. Sikkerhetskopien lagres når dialogboksen avsluttes. Sikkerhetskopien ligger nå som en fil i operativsystemet uten noen form for kobling med brukergrensesnittet til programmet. Brukeren møter også i dette tilfellet en dialogboks. Tidligere har jeg beskrevet bruken av dialogvinduer som lite fleksibelt for brukeren. Brukeren bli låst til et vindu og en interaksjon. Det kan da sees på som en ulempe for brukbarheten at programmene bruker en dialogboks for å gjennomføre sikkerhetskopifunksjonaliteten.

Aperture har visualisert bruken av sikkerhetskopier i brukergrensesnittet på en annen måte, uten å bruke et dialogvindu. De lar brukeren lage et eget objekt i brukergrensesnittet som kalles en filsafe[8]. I en likhet med en virkelig safe, kan en legge filene i safen og der ligger de trygt oppbevart. Brukeren opererer på selve filsafen gjennom ikoner som visualiserer de nødvendige funksjonene brukeren trenger i forbindelse med filsafen. Denne filsafen kan en overføre til et program for diabetikere, for å at brukeren kan lage sikkerhetskopier av sine data. I neste kapittel viser en skisse hvordan en slik filsafe kan implementeres.

6.6.2 En mulig løsning

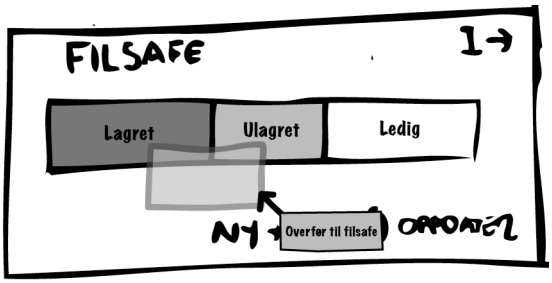
Figur 28 viser en skisse på hvordan en filsafe kan implementeres i et brukergrensesnitt. Denne filsafen likner veldig på den en finner i Aperture. Brukeren kan lage en eller flere filsafer ved å klikke på ”ny +” knappen. Brukeren vil da måtte velge en filbane hvor filsafen skal lagres, og kan da få mulighet til å lagre filsafen på en hardisk, en ekstern minnebrikke, eller en cd/dvd som har den egenskapen at de kan skrives til flere ganger. Når brukeren har laget en ny filsafe, vil denne safen bli visualisert som et eget objekt i brukergrensesnittet. Brukeren vil da kunne se ved hjelp av en type progresjonsbar hvordan forholdet er mellom data som ligger i filsafen, data som ikke ligger i filsafen og ledig plass. Det er også knyttet et ikon med to buede piler til filsafen. Ved hjelp av dette ikonet kan brukeren oppdatere filsafen automatisk.

[8] I det engelskspråklige programmet heter objektet ”file vault”



Figur 28: Et eksempel på hvordan en filsafe kan visualiseres i sammenheng med kalenderen i brukergrensesnittet. Brukeren kan se en oversikt over hvor stor andel av dataene som er lagret i filsafen, og hvor stor plass det er igjen i selve safen. Videre kan kalenderobjektet og filsafeobjektet gi opphav til direkte manipulasjon.

Figur 29 viser et eksempel hvor brukeren kan dra delen som visualiserer de dataene som ikke ligger i filsafen (ulagret) over på delen som viser de lagrete dataene (lagret). Brukeren får her opp en boks ved siden av musepekeren som forteller hvilken handling som blir gjennomført.



Figur 29: Brukeren kan dra den delen som ikke er lagret i filsafen over den delen som er lagret i filsafen.

Brukeren kan også dra hele kalenderobjektet fra kalenderen over på filsafen, for å ta sikkerhetskopi av dataene. Ønsker brukeren å opprette dataene fra filsafen kan brukeren dra drar den lagrede delen i brukergrensesnittet opp til kalendervinduet eller kalenderen for å opprett data fra en filsafe. Brukeren får da en tekst ved siden av musepekeren som forklarer hvilken operasjon som vil bli gjennomført ved denne kombinasjonen. Brukeren kan også ta tak i kanten mellom den lagrede og ulagrede delen. Denne kanten kan draes mot høyre slik at den dekker hele den ulagrede delen. Dataene i den ulagrede delen vil da bli lagt til den lagrede delen. Filsafeobjektet kan manipuleres på forskjellige måter.

6.6.3 *Teoretisk analyse av skissen*

6.6.3.1 Bruken av metaforer

Bruken av ordet ”safe” i denne skissen kan kalles en metafor, slik som den andre metaforen som almanakkmetaforen i loggføringsprosessen. I likhet med almanakkmetaforen er safemetaforen med på å presentere en konseptuel modell. Hva er så fordelene og ulempene med bruken av denne safe metaforen i denne skissen?

Det er fornuftig å tro at brukeren får en assosiasjon til en virkelig safe i den virkelige verden, når en ser ordet filsafe. Samtidig kan en si at denne filsafen ikke ligner så mye på en virkelig safe. I den virkelige verden kan en si at en safe er et skap som er vanskelig å bryte seg inn i. Videre kan en få en oppfattelse av at gjenstander som ligger inne i denne safen ligger da trygt mot omgivelsene. Det trengs også en nøkkel eller en kode for å komme seg inn i en virkelig safe. Denne beskrivelsen likner ikke mye på den safen en ser i skissen. I safemetaforen brukes verken en nøkkel eller en kode for å få tilgang til dataene. Visualiseringen av filsafen likner heller ikke på det en forbinder med utseende til en vanlig safe. Dette bryter med selve safemetaforen. En kan si at det metaforen og en virkelig safe har felles er at den beskriver et konsept som kan knyttes til sikkerhet. Det er få fellestrekk med funksjonaliteten til filsafen og en virkelig safe. En kan da si at metaforen kan hjelpe brukeren i å forstå et konsept og ikke funksjonalitet. Laurel (1993) forklarer at ulempen med å bruke en metafor oppstår hvis brukeren tar metaforen bokstavelig. Siden en kan assosiere filsafen med en virkelig safe, kan det skapes forventninger til filsafens funksjonalitet. Laurel (1993) beskriver dette som at en kan ”dette av” metaforen når funksjonaliteten mellom metaforen og det virkelige objektet har vesentlig forskjellig funksjonalitet. I dette tilfellet kan det være fare for at brukeren kan ta metaforen bokstavelig, siden funksjonaliteten er forskjellig.

Filsafens funksjon i skissen, kan sies å ha en tilknytning til å gi brukeren en forståelse av konseptet, eller en konseptuel modell. Marcus (1998) beskriver en overordnet metafor som en metafor som gir en basis for hvordan en skal forstå et system. Filsafen kan da kategoriseres som en overordnet metafor, siden den forklarer et konsept. Det er allikevel en fare for at brukeren kan ta safe metaforen bokstavelig, noe som kan sees på som en ulempe ved bruken av en slik metafor.

6.6.3.2 Interaksjon

Ved bruken av filsafen legger skissen opp til en annen interaksjon med brukeren, enn ved bruken av et dialogvindu. Som tidligere skrevet vil et dialogvindu kunne sammenlignes med Normans (1998) eksekverings-evalueringsløkke. I diabetesprogrammene brukes for eksempel et slikt dialogvindu. I diabetesprogrammene vil brukeren oppleve en slik interaksjon gjennom dialogvinduet. Brukeren bestemmer seg for å ta en sikkerhetskopi, for å så finne ut hvilket menyvalg som vil lage en sikkerhetskopi. Videre må brukeren via dialogen gi den nødvendige informasjonen som trengs for å opprette en sikkerhetskopi. Når brukeren er ferdig med dialogvinduet vil brukeren få den hele og første tilbakemelding for selve handlingen. Ut i fra tilbakemeldingen kan brukeren evaluere resultatet. Ble ikke resultatet det brukeren hadde forventet vil en ha en det Normans (1998) kaller feil ved evaluering. Hvis brukeren for eksempel ikke finner hvilket riktig menyvalg eller forstår deler av dialogvinduet vil en ha det Normans (1998)

kaller feil ved eksekvering. Hvordan har da denne skissen løst interaksjonen med brukeren i sikkerhetskopiprosessen uten å bruke et dialogvindu?

I skissen er interaksjonen rundt sikkerhetskopien annerledes. Det er ikke like naturlig å knytte interaksjonen med filsafen til Normans (1998) eksekverings-evalueringsløkke. Interaksjonen med filsafen blir delt opp i en mengde små deler, som vil gi brukeren en egen respons for hver del. Denne interaksjonen vil, som tidligere beskrevet, minne mer om det Djajadiningrat et al. (2002) kaller operasjonsformål og innebygd respons. Brukeren kan i skissen lage en ny filsafe som en egen operasjon med en egen respons. Videre kan brukeren oppdatere filsafen som en egen operasjon og for eksempel velge å overføre en og en blodsukkermåling til filsafen. Alle disse deloperasjonene vil ha en egen innebygd respons. I Figur 29 kan brukeren dra en del av filsafen over på en annen, for å få overført data som ikke ligger i filsafen til filsafen. I dette eksempelet vil da den ulagrede delen ha et operasjonsformål om at den kan bli dratt over den andre delen. Når brukeren drar delen over til den andre vil brukeren få en tilbakemelding i form av at den ulagrede firkanten vil bli visualisert sammen med musepekeren. Brukeren får her det Djajadiningrat et al. (2002) kaller en respons på navigasjon. Når brukeren slipper den ulagrede firkanten over den lagrede, vil brukeren få en respons ved at den ulagrede firkanten vil bli erstatte med den lagrede firkanten. Dette vil igjen være en respons på selve operasjonen, om at dataene er lagret i filsafen.

Djajadiningrat et al. (2002) deler en slik interaksjon med objekter også opp i en pre- og postoperasjonsfase. Det er den informasjonen brukeren får før og etter selve handlingen. I dette tilfellet er visualiseringen av navigasjon og den ulagrede delen av filsafen informasjon brukeren får i preoperasjonen. Når den lagrede delen av filsafen vokser over den ulagrede delen, er dette informasjon brukeren får i en postoperasjonsfase. Djajadiningrat et al. (2002) beskriver at det er viktig at det er sammenheng mellom pre- og postoperasjonsfasen. Siden brukeren kan se at den ulagrede delen av filsafen, forsvinner kan en si at pre- og postoperasjonsfasene er nært knyttet til hverandre. Dette er med på å knytte operasjonsformålet og den innebygde responsen nær til hverandre. Ved bruken av et dialogvindu kan en si at operasjonsformålet og den innebygde responsen ikke er knyttet like nært til hverandre. Siden responsen kommer helt til slutt når dialogen er ferdig.

Brukeren kan også i skissen klikke på et ikon med to buede piler for å oppdatere filsafen. Ved bruken av denne typen piler vil mye av prosessen bli autmoatisert. Dette gjør at det kan bli lenger avstand mellom pre- og postoperasjonsfasen. Det er her viktig at brukeren får det Djajadiningrat et al. (2002) definerer som en prosess som er under utføring. En kan si at en naturlig visualisering av en slik prosess, er en animering av at den lagrede delen av filsafen beveger seg over den ulagrede, slik som en progresjonsbar. Denne responsen vil bygge på samme respons som brukeren får ved å dra delene over hverandre. Det vil være med på å knytte sammen pre- og postoperasjonsfasene. Brukeren får tilsynelatende samme respons på samme operasjon, men operasjonen har blitt startet på forskjellige måter.

Jeg har vist hvordan interaksjonen med denne filsafen er delt opp i flere sykluser. Dette gjør at programmet har en annen visualisering av sikkerhetskopiprosessen enn en løsning ved bruk av dialogvindu.

6.6.3.3 *Direkte kombinasjon, instrumenter og direkte manipulasjon*

I denne skissen er filsafen grunnlag for direkte manipulasjon. Siden brukeren kan manipulere med musepekeren på selve filsafeobjektet. Filsafeobjektet kan også sees på som et instrument. Videre bruker dette instrumentet prinsipper som kan beskrives som direkte kombinasjon. Dette vil bli begrunnet nærmere.

Når brukeren drar enten den ulagrede delen av filsafen, målinger eller kalenderobjektet over filsafen, kan dette beskrives som direkte kombinasjon. En ser i skissen at brukeren vil få opp en egen tekst ved siden av musepekeren. Denne teksten forteller brukeren hvilken handling som vil bli utført hvis objektet ”slippes” her. Dette kan minne om hvordan blodsukkerapparatet benytter direkte kombinasjon i synkroniseringsprosessen. Som beskrevet i synkroniseringsprosessen forklarer blant annet at Holland og Oppenheim (1999) direkte kombinasjonsbegrepet som at to eller flere substantiv kombineres med et verb. I skissen knytter en da substantivene ”lagrete” og ”ulagrete” og verbet blir ”overføring”, som er teksten som kommer opp i boksen under musepekeren. Siden det er kun mulig å utføre en kombinasjon med disse to objektene, får ikke brukeren mulighet til å velge handling ut i fra en listen. Brukeren får da i praksis informasjon om hvilken handling som kan bli utført. Holland og Oppenheim (1999) beskriver da en slik operasjon som en motsetning til en dialogboks. Brukeren kan manipulere filsafeobjektet uten å måtte avbryte handlingen for å gi informasjon i en dialogboks. Bruken av direkte kombinasjon kan da gi bedre brukbarhet, enn ved bruken av et dialogvindu.

I denne skissen over sikkerhetskopiprosessen kan en i tillegg til å beskrive filsafen med direkte kombinasjon, også definere filsafen som et instrument. Filsafen i seg selv kan beskrives som det Beaudouin-Lafon (2000) kaller et domeneobjekt og instrumentene behandler da atributtene på selve domeneobjektet. I skissen kan en si at instrumentet er visualisert på selve domeneobjektet. Når brukeren tar tak i kanten mellom de lagrede og ulagrede dataene for å så dra denne kanten over de ulagrede dataene, er dette en interaksjon som kan kobles med instrumenter. Selve progresjonsbaren på filsafeobjektet kan da sees på som et instrument. Dette instrumentet har ikke det Beaudouin-Lafon (2000) kaller et aktiveringsbehov. Brukeren kan manipulere direkte på dette instrumentet ved hjelp av musepekeren, uten å aktivere selve instrumentet først. Instrumentet har videre det Beaudouin-Lafon (2000) kaller et plassbehov. Instrumentet er visualisert på selve domeneobjektet, og har et plassbehov slik at brukeren kan manipulere på instrumentet. Beaudouin-Lafon (2000) bruker graden av indireksjon for å beskrive instrumentets avstand- og tidsaspekt. I denne skissen er graden av indireksjon liten. Avstanden fra instrumentet til domeneobjektet er liten og tidsaspektet fra brukeren utfører en operasjon til brukeren får en respons er også liten.

Filsafeinstrumentet setter rammene på hvordan interaksjonen med brukeren blir gjennomført, som beskrevet tidligere i denne analysen. Instrumentene og den direkte kombinasjonen er med på å gjennomføre interaksjonen i små sykluser. Beaudouin-Lafon (2000) beskriver interaksjonen med et instrument med at brukeren utfører en handling på selve instrumentet, deretter får brukeren får en reaksjon fra instrumentet. Videre sender instrumentet en kommando til domeneobjektet som igjen gir et resultat til instrumentet. Deretter oversettes resultatet til en respons til brukeren gjennom instrumentet. Et slikt interaksjonsmønster kan knyttes til det Djajadiningrat et al. (2002) kaller operasjonsformål og innebygd respons. Hver handling blir delt opp i

små operasjoner. En kan si at bruken av instrumenter filsafen støtter opp under den interaksjonen som er beskrevet i interaksjonsdelen av denne analysen (kapittel 6.6.3.2).

6.6.3.4 Sammendrag

I denne delanalysen har jeg analysert visualiseringen av sikkerhetskopiprosessen til skissen. Jeg har beskrevet filsafen som en metafor. Bruken av en metafor i dette tilfellet har fordeler og ulemper. Metaforen hjelper brukeren til å forstå et helhetlig konsept, men ikke til å forstå funksjonaliteten til filsafemetaforen. Filsafemetaforen sin funksjonalitet er ikke lik en virkelig safe. Dette gjør at funksjonaliteten til filsafen må beskrives med andre begreper. Jeg har også vist hvordan denne skissen unngår å bruke dialogvinduer for å gjennomføre sikkerhetskopiprosessen. Ved bruken av direkte manipulasjon på selve filsafe metaforen har interaksjonen blitt delt opp i flere små sykler. Videre har jeg har beskrevet hvordan denne skissen bruker instrumenter og direkte kombinasjon. Når en kombinerer den lagrede delen og den ulagrededelen, bygger dette på prinsippene rundt direkte kombinasjon. I tillegg kan hele filsafeobjektet sees på som et instrument. Bruken av direkte kombinasjon og instrumenter er som beskrevet med på å dele opp interaksjonen i små sykluser, som igjen kan gi brukeren mer fleksibilitet.

6.7 Skissene og brukbarhet

I diskusjonen rundt hver enkel skisse har jeg knyttet skissene opp mot begreper innenfor interaksjonsdesign. Disse teoretiske begrepene beskrev jeg i kapittel 4. I kapittel 3.3.2 beskrev jeg hvordan disse teoriene vil kunne knyttes til brukbarhet. I samme kapittel ble også brukbarhetsbegrepet delt opp i lærbarhet, fleksibilitet og robusthet. Jeg vil nå gå igjennom hvert punkt og beskrive hvordan skissen har belyst dette.

6.7.1 Lærbarheten i skissene

Lærbarheten sier noe om hvor lett brukergrensesnittet er å lære. Det er da viktig hvordan funksjonene programmet tilbyr er visualisert. I skissene er funksjonen visualisert ved hjelp av objekter og ikoner. Blodsukkerapparatet, kalenderne, de smarte analysene, dataobjektene, kategoriobjektene, regelobjektene og filsafen er alle objekter i brukergrensesnittet. Alle disse objektene er med på å visualisere funksjonaliteten. En får da en synlighet av funksjonaliteten ovenfor brukeren. I motsetning til for eksempel et kommandogrensesnitt vil alle funksjonene være tilgjengelige gjennom visuelle objekter i brukergrensesnittet i skissene. Dette kan gjøre funksjonene lettere å lære.

Disse objektene er igjen direkte manipulerbare. Brukeren kan for eksempel ta tak i objekter i brukergrensesnittet for å forandre dem. I skissen kan en også bruke instrumenter og direkte kombinasjon i brukergrensesnittet. Dette er teknikker som igjen er direkte manipulerbare. Skissen har da instrumenter som intrumentpilene, filsafeinstrumentet, progresjonsbarinstrumentet og det gjennomsiktige vinduinstrumentet. Ved synkroniseringsprosessen og ved filsafen brukes det da direkte kombinasjon. Dette er alle elementer av direkte manipulasjon. Preece (2002) beskriver, som tidligere beskrevet, at direkte manipulasjon vil gi brukeren en mindre følelse av usikkerhet rundt brukergrensesnittet. Brukeren oppnår heller følelsen av

mestring og kontroll over brukergrensesnittet (2002). Noe som igjen kan være med på å øke lærbarheten.

I skissene brukes det også metaforer. I skissene finner en almanakk-, blodsukkerapparat- og en filsafemetafor. Metaforer kan være med på gjøre brukergrensesnittet familiært for brukeren og kan hjelpe brukeren til å forstå en konseptuel modell. Preece (2002) beskriver at metaforene skaper noe som er kjent for brukeren og dette hjelper brukeren å lære og forstå programmet lettere. Det har også i skissene blitt brukt additive dimensjoner. Disse dimensjonene bygger da på kunnskap brukeren har fra før og kan oppfattes som familiært. At elementer i brukergrensesnittet kan forbindes med familiært, kan bidra til å bedre lærbarheten i brukergrensesnittet.

Forutsigbarhet og konsistens er også blitt knyttet til lærbarheten i kapittel 3.3.2. Som tidligere beskrevet er skissene bygget opp av objekter og direkte manipulasjon. Objektene har en interaksjon som er knyttet til Djajadiningrat et al. (2002) sitt operasjonsformål og innebygde respons. Hvert objekt har et operasjonsformål og en innbygget respons. Dette er med på å gi en forutsigbarhet og konsistens i brukergrensesnittet.

Gjennom visualisering av funksjonalitet gjennom objekter med et operasjonsformål og en innebygd respons og et brukergrensesnitt som spiller på forutsigbarhet, konsistens og familiaritet, kan disse skissene oppnå lærbarhet ovenfor brukeren.

6.7.2 Fleksibilitet i skissene

Et viktig begrep rundt fleksibilitet er det som heter dialoginitiativet. Brukeren skal føle at en har kontroll over systemet og må da være den som styrer interaksjonen med systemet. Som tidligere beskrevet er dialogvinduer med på å begrense fleksibiliteten til brukeren. I skissene er objektene i brukergrensesnittet knyttet til Djajadiningrat et al. (2002) sitt operasjonsformål og innebygde respons. Dette er med på å gjøre programmet fleksibelt. Det brukes ikke dialoger i skissene og brukeren gir alle inndata uten bruk av dialoger. Dialogvinduene som diabetesprogrammene brukte i synkroniseringsprosessen, har da blitt erstattet med løsninger som benytter direktemanipulasjon og direkte kombinasjon. Som igjen medfører at dialoginitiativet består hos brukeren, siden brukeren bestemmer selv når inndata skal gis til programmet.

Personalisering er også knyttet til fleksibilitet i kapittel 3.3.2. I skissene kan brukeren personalisere hvilke datatyper som skal loggføres i egne subbkalendere, kategoriene og ikke minst kan brukeren definere og personalisere analysene selv. Brukeren har da en mulighet til å personalisere programmet etter egne ønsker.

Gjennom å unnlate å bruke dialogvinduer og gjøre programmet personaliserbart for brukeren, kan da denne skissen gi brukeren fleksibilitet.

6.7.3 Robusthet i skissene

Observerbarhet og respons er begreper som er knyttes til robusthet i kapittel 3.3.2. Brukeren må kunne observere tilstanden til brukergrensesnittet. Som tidligere

beskrevet er funksjonaliteten i skissene bygget opp av objekter som representerer funksjonalitet. Disse objektene er direkte manipulerbare og vil da gi en kontinuerlig tilbakemelding til brukeren. Det er da lagt til rette for at en kan observere tilstanden i programmet også under gjennomføring av funksjoner. Når en for eksempel drar i et objekt vil dette objektet gi en respons tilbake til brukeren både under gjennomførelsen og ved at objektet viser en forandring. Dette har blitt kalt av Djajadiningrat et al. (2002) som en bekreftelse på navigasjon. Det er da mulig å observere gjennomførelsen av utføringen av en handling i brukergrensesnittet. Dette er med på at en hele tiden har muligheten til å observere tilstanden til programmet.

Gjenopprettelsesmuligheten er også et begrep som ble knyttet robusthet i kapittel 3.3.2 Ved bruken av direkte manpiluasjonsteknikker skal det også være mulig å angre sine handlinger. Dette vil gi brukeren en gjenopprettelsesmulighet. I skissene er det også beskrevet at dette kan være et hjelpemiddel for brukeren til å utforske brukergrensesnittet. Brukeren kan da gjennomføre en handling for så å angre den etterpå. En har da utforsket et objekts funksjonalitet ved å gjennomføre en operasjon for å se på responsen som er knyttet til objektet. En kan da si at det i skissene er det mulig å angre sine handlinger.

Gjennom observerbarhet, respons og gjennopprettelsesmuligheten kan en si at disse skissene kan gi en form for robusthet.

7 Konklusjon

I innledningen til denne oppgaven beskrev jeg en problemstilling for oppgaven. Problemstillingen ble formulert som to spørsmål og ble beskrevet slik:

- *Hvordan kan funksjonaliteten, som brukeren kan ha behov for, visualiseres og mappes i brukergrensesnittet, med tanke på bruksituasjonen til en diabetiker?*

- *Finnes det liknende bruksituasjoner og systemer en kan hente erfaring fra?*

Jeg vil nå beskrive hvordan jeg har besvart disse spørsmålene i oppgaven i den rekkefølgen de står beskrevet.

Skissene i diskusjonskapittelet har tatt for seg hver av funksjonalitetsbehovene som er beskrevet i kapittel 3.3.1. Til hvert funksjonalitetsbehov er det blitt beskrevet en skisse som viser hvordan denne funksjonaliteten kan visualiseres. Videre har jeg redegjort for teoretiske begreper innenfor interaksjonsdesign og knyttet skissene til disse teoretiske begrepene. Jeg har redegjort for hvordan begrepene kan beskrive skissene, for å få en grundigere analyse og forståelse av visualiseringselementer i skissene. De teoretiske begrepene innenfor interaksjonsdesign kan igjen knyttes til behovene for brukbarhet som er beskrevet i kapittel 3.3.2. Jeg har da knyttet skissene til teori som kan gi bedre brukbarhet. Gjennom skissene har jeg vist hvordan funksjonaliteten, som en bruker kan ha behov for kan, visualiseres og mappes i brukergrensesnittet.

I oppgaven har jeg tatt utgangspunkt i programmer som eksisterer for diabetikere og programmer som blir brukt i liknende brukssituasjoner. Skissene som er blitt laget har blitt generert ut i fra dagens diabetesprogrammer og andre programmer i liknende bruksituasjoner. I kapittel 5 har jeg redegjort for hvordan funksjonaliteten har blitt visualisert i dagens diabetesprogrammer. Videre har jeg vist løsninger på hvordan andre programmer i liknende bruksituasjoner har visualisert funksjonalitet, som kan ha likhetstrekk med de funksjonelle behovene et diabetesprogram kan ha. En kan da si at skissene har hentet visualiseringselementer fra andre programmer i andre bruksituasjoner. Jeg har i kapittel 5 beskrevet hvilke erfaringer på visualisering av funksjonalitet en kan hente fra liknende bruksituasjoner, som kan sees som en utdypning av det andre spørsmålet i problemstillingen. Det finnes da andre bruksituasjoner en kan hente erfaring fra og skissene viser hvordan disse erfaringen kan overføres til en diabetikers brukssituasjon.

Skissene har blitt analysert i forhold til et teoretisk grunnlag. Det kan videre argumenteres for at dette teoretiske grunnlaget kan gi bedre brukbarhet. Dette er gjort i kapittel 3.3.2. Det er viktig å merke seg, som det er beskrevet i metodekapittelet, at det er kun argumentert for at skissen *kan* gi bedre brukbarhet og ikke *vil* gi bedre brukbarhet. Skissene må sees på som et ledd av en større utvikling og som en diskusjon rundt visualiseringen av disse funksjonene. For å undersøke om disse skissene vil gi bedre brukbarhet, må det gjennomføres ytterligere studier. Dette vil være en del av denne oppgavens videre arbeid.

7.1 Videre arbeid

For å beskrive videre arbeid vil jeg skille mellom videre arbeid for den prosessen denne oppgaven er en del av og videre arbeid som kan være interessant i forbindelse med hele bruksitasjonen. For å beskrive videre arbeid i bruksituasjonen er en nødt til å se arbeidet i denne oppgaven i en større sammenheng.

7.1.1 Videre arbeid knyttet til denne oppgaven

Denne oppgaven gir ikke svar på hvordan funksjonaliteten skal visualiseres for å gjøre den brukbar. Den bidrar kun i en diskusjon om hvordan funksjonalitetene kan gjøres brukbare og dette er argumentert med teoretiske begreper. Det vil være vanskelig å konkludere hvor vidt disse visualiseringsløsningene er brukbare uten å ha testet disse på reelle brukere.

Videre arbeid i forhold til brukbarhet vil da være å utvikle skissene videre til et funksjonelt produkt som kan testes i en reell bruksituasjon. En bør lage et produkt som kan testes i virkelige situasjoner med diabetikere. Videre kan det da studeres om designløsningene er mer brukbare enn andre løsninger. Det er da etter et slikt studie en kan ha mulighet til å trekke konklusjoner om brukbarheten til visualiseringene av funksjonalitet i skissene.

En annen del av det videre arbeidet med skissene vil da være å fullføre den utviklingsprosessen denne oppgaven kan sees som en del av. Da må en få klargjort noen av denne oppgavens avgrensninger. Som beskrevet i innledningen i oppgaven er det tatt stilling til hvordan skissene skal implementeres. Det vil da være et videre arbeid å se på hvordan visualiseringselementene i skissen kan implementeres og hvordan funksjonaliteten som skissene beskriver kan implementeres.

Det kan være interessant å studere annen funksjonalitet fra hjelpemiddelene for denne brukergruppen. I denne oppgaven er det kun tatt utgangspunkt i det en kan kalle generell funksjonalitet som et slikt program bør tilby. For å belyse denne typen videre arbeid må en se funksjonaliteten og skissene inn i en større sammenheng i bruksituasjonen.

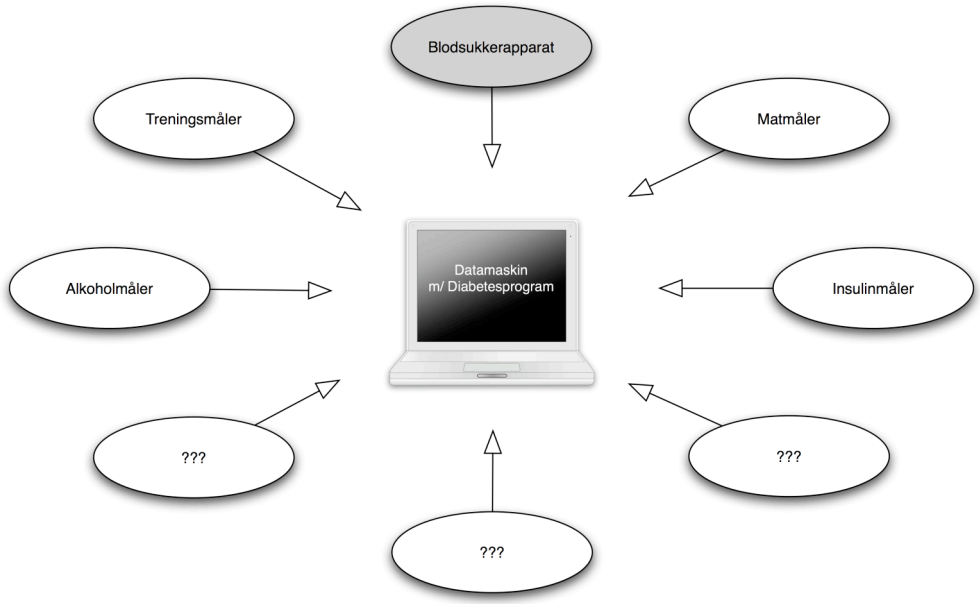
7.1.2 Videre arbeid i bruksituasjonen

For at dataene skal bli overført til et program må de som beskrevet i oppgaven digitaliseres. Ved bruken av et blodsukkerapparat vil digitaliseringsprosessen finne sted når brukeren måler blodsukkeret. Dette medfører at brukeren ikke trenger å memorere dataene for så å få lagt de inn i programmet. Brukeren kan som beskrevet i oppgaven overføre disse dataene digitalt. De andre data vil bli digitalisert når brukeren manuelt registrerer dataene i brukergrensesnittet til et program for diabetikere. Brukeren er da avhengig av å memorere disse dataene helt til brukeren er ved datamaskinen for å få lagt de inn manuelt.

Figur 30 beskriver hvordan en digital loggbok kan være kjernen blant mange forskjellige enheter som måler forskjellige datagrupper. I denne skissen er det kun blodsukkerapparatet (merket med grått) som i dag har mulighet til å digitalisere

dataene og overføre de til en datamaskin. De andre dataene må digitaliseres gjennom brukergrensesnittet til programmet.

En kan i Figur 30 se for seg en mengde forskjellige målere som kan digitalisere dataene for brukeren i den daglige brukssituasjonen. Målerne må da ha en mulighet til å loggføre de aktuelle dataen i den daglige situasjonen, for så å overføre de til et program digitalt. Brukeren trenger da ikke å memorere dataene for så å digitalisere de senere i et program, siden digitaliseringen vil finne sted i de daglige situasjonene. Programmet vil da samle inn dataene fra de forskjellige målerne.



Figur 30: Et program samler inn alle data fra forskjellige enheter. I de tilfellene hvor det ikke finnes en teknisk enhet, må brukeren loggføre disse dataene manuelt i systemet.

Et videre arbeid kan da være å se på hvordan dataene utenom blodsukkerapparatet kan digitaliseres på andre måter enn i brukergrensesnittet til programmet. En kan tenke seg at disse forskjellige målerne samles i en eller flere håndholdte enheter. Hvordan disse eller denne enheten skal designes og visualisere sin funksjonalitet for å bli mest mulig brukbare, er da en interessant problemstilling.

Det kan også være interessant å lage håndholdte produkter som mulig kan erstatte en programvare som brukes på en datamaskin. Det vil da i være interessant å se hvilke fordeler og ulemper bruken av håndholdte enheter vil ha i denne bruksituasjonen.

Dette er interessante problemstillinger som bør drøftes for å belyse flere av aspektene i en diabetikers bruksituasjon. Denne oppgaven har tatt utgangspunktet i den bruksituasjonen som eksisterer i dag og hvordan denne kan videreutvikles innenfor de tekniske hjelpemiddelene som allerede eksisterer. Jeg har da bygget videre på modellen om at brukeren har et blodsukkerapparat og en datamaskin med programvare som tekniske hjelpemidler i sin bruksituasjon. Skissene som er generert i oppgaven belyser da kun en av flere forskjellige aspekter ved bruksituasjonen til diabetikere.

Kilder:

APPLE COMPUTER (1992) *Macintosh Human Interface Guidelines*, Apple Computer, Inc.

BAECKER, R., SMALL, I. & MANDER, R. (1991) Bringing Icons To Life.

BEAUDOUIN-LAFON, M. (2000) Instrumental interaction: an interaction model for designing post-WIMP user interfaces. *Proceedings of the SIGCHI conference on Human factors in computing systems.* The Hague, The Netherlands, ACM Press.

BRAD, A. M. (1998) *A brief history of human-computer interaction technology*, ACM Press Journal Article

CHRISTOPHERSEN, Y. & HANSSEN, K. F. (2004) *Diabetes for livet : aldri fred å få?,* [Gjettum], Y. Christophersen.

DIX, A., FINLAY, J., ABOWD, G. D. & BEALE, R. (2004) *Human-computer interaction,* Upper Saddle River, NJ, Pearson.

DJAJADININGRAT, T., OVERBEEKE, K. & WENSVEEN, S. (2002) But how, Donald, tell us how?: on the creation of meaning in interaction design through feedforward and inherent feedback. *Proceedings of the conference on Designing interactive systems: processes, practices, methods, and techniques.* London, England, ACM Press.

HANÅS, R. (2002) *Type 1 diabetes, hos barn, ungdom og unge voksne*, NKS forlaget.

HARAMUNDANIS, K. (1996) *Why icons cannot stand alone*, ACM Press.

HOLLAND, S. & OPPENHEIM, D. (1999) Direct combination. *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit.* Pittsburgh, Pennsylvania, United States, ACM Press.

HOLMQUIST, L. E. (2005) Prototyping: generating ideas or cargo cult designs? Methods & tools: design methodology and design practice.

KIM HALSKOV, M. (2000) *Magic by metaphors,* Elsinore, Denmark ACM Press.

LAUREL, B. (1993) *Computers as theatre,* Reading, Mass., Addison-Wesley.

MADSEN, K. H. (2000) Magic by metaphors. *Proceedings of DARE 2000 on Designing augmented reality environments,* 20 167-169

MARCUS, A. (1998) Metaphor design in user interfaces. *ACM SIGDOC Asterisk Journal of Computer Documentati*.

NIELSEN, J. & MACK, R. L. (1994) Usability inspection methods.

NORMAN, D. A. (1998) *The design of everyday things,* London, MIT Press.

NORMAN, D. A. (1999) Affordance, conventions, and design. *interactions,* 6**,** 38-43.

POST- OG TELETILSYNET, N. (2005) Det norske telemarkedet 2004.

PREECE, J. (2002) Interaction design : beyond human-computer interaction.

SHNEIDERMAN, B. (1998) *Designing the user interface : strategies for effective human-computer interaction,* Reading, Mass., Addison-Wesley.

SMITH, D. C. & IRBY, C., H. (1998) Xerox Star live demonstration. *CHI 98 conference summary on Human factors in computing systems.* Los Angeles, California, United States, ACM Press.