Menneske-Maskine Interaktion

Obligatorisk opgave

Carsten V. Munk 20030576
Søren E. Andersen 20061647
Nicki T. Hansen 20030605

Indledning

Vi har i denne aflevering inkluderet to del-opgaver fra faget menneske-maskine interaktion.

Den første opgave omhandler en analyse af Microsoft OneNote, som belyser, hvordan en bruger benytter dette computeriserede noteværktøj. Vi indsamler empiri fra en testbruger, således at vi kan benytte empirisk analyse, og arbejder med direkte analytiske metoder for at fremvise de designmæssige fordele og ulemper i programmet.

Den anden opgave omhandler WIMP paradigmet og de designmetoder, der eksisterede før og efter dette. Vi ser nærmere på udviklingen fra præ-WIMP til WIMP og videre til hvad der kommer efter WIMP. Intentionen med opgaven er at belyse de forskellige paradigmers syn på menneske og maskine, da dette ændrer sig med den historiske udvikling. Derudover ser vi nærmere på de analyseværktøjer og design-ideer, som vi har fået fremvist i vores undervisning og sætter disse ind i en historisk kontekst.

Microsoft OneNote

MMI analyse

Carsten V. Munk 20030576
Søren E. Andersen 20061647
Nicki T. Hansen 20030605

Indholdsfortegnelse

Introduktion

Vi har valgt at skrive om det interaktive noteværktøj Microsoft OneNote, som en afløser for almindelige håndskrevne noter. I den teknologiske højalder vi lever i er det fordelagtigt at udnytte de maskiner vi i forvejen bruger, specielt til forelæsninger og øvelsestimer. Almindelige skriveprogrammer såsom Word og lignende har flere begrænsninger, specielt med formatering og organisering af information. OneNote tilbyder et bedre alternativ til de håndskrevne noter, da man hurtigt kan lave tegninger, figurer, punkter og tabeller. Hvordan en bruger interagerer med OneNote er målet for denne opgave.

Metode

Vi analyserer i denne opgave Microsoft OneNote. Vi starter med at beskrive programmets fremtræden for brugeren, og hvad programmets intenderede og aktuelle brug er. Derefter gennemgår vi programmet med de analytiske værktøjer vi har til rådighed, Cognitive Walkthrough, Activity Walkthrough og GOMS. Følgende benytter vi vores empiriske værktøjer, think aloud og fokusskiftanalyse, til at bearbejde data fra en konkret testbruger. Vi konkluderer ud fra disse metoder eventuelle forbedringer og optimeringer der kan indføres i programmet.

Endeligt reflekterer vi over opgaven og de komplikationer vi stødte på undervejs, samt de overvejelser vi har gjort os om opgaven.

Vi optog vores empiri med en enkelt testbruger, og optog med kamera, noter og skærmoptagelser. Desværre skete der det, at vores skærmoptager brød ned efter lang tids optagelse, hvorfor vores empiri har lidt under dette.

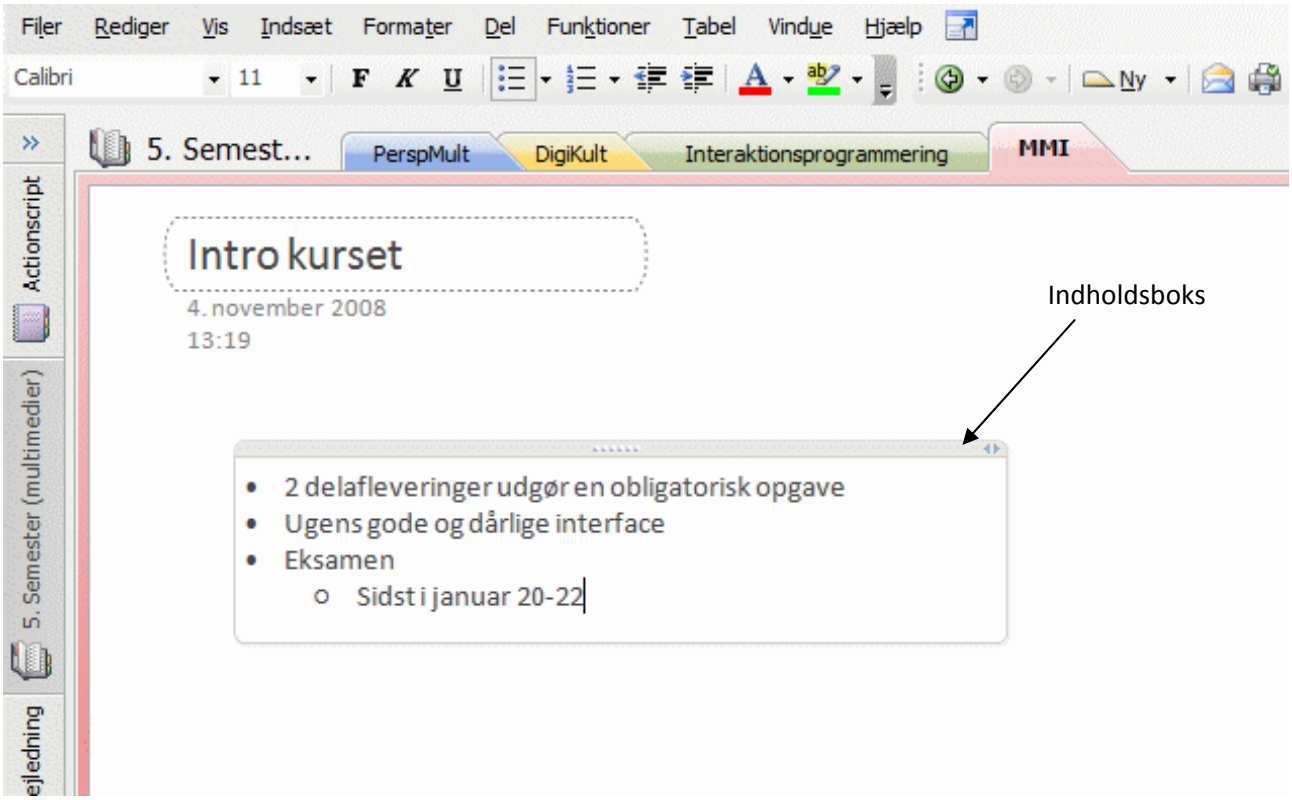
Applikationen – Microsoft OneNote

Fremtræden

Programmet bruger genkendelige former fra papir alternativet. Overordnet inddeler programmet dine filer i ”notesbøger”, under det er der sektioner i notesbogen og under dette endnu er der sider.

Programmet arbejder altså med metaforer fra det værktøj som det søger at afløse. Dette gøres formentligt for at en given bruger skal kunne bruger værktøjet intuitivt. De overordnede inddelinger kan gøre det svært for at bruger at ”bare sætte sig ned og notere”, dog har det nogle fordele i det lange løb, såsom at noter ikke forsvinder og at der er en fast organisering af noter. Sektionerne farves automatisk med en baggrundsfarve, som ligger omkring det hvide ”papir”. Denne farve kan man ændre og hjælper en bruger til at hurtigt identificere en given sektion.

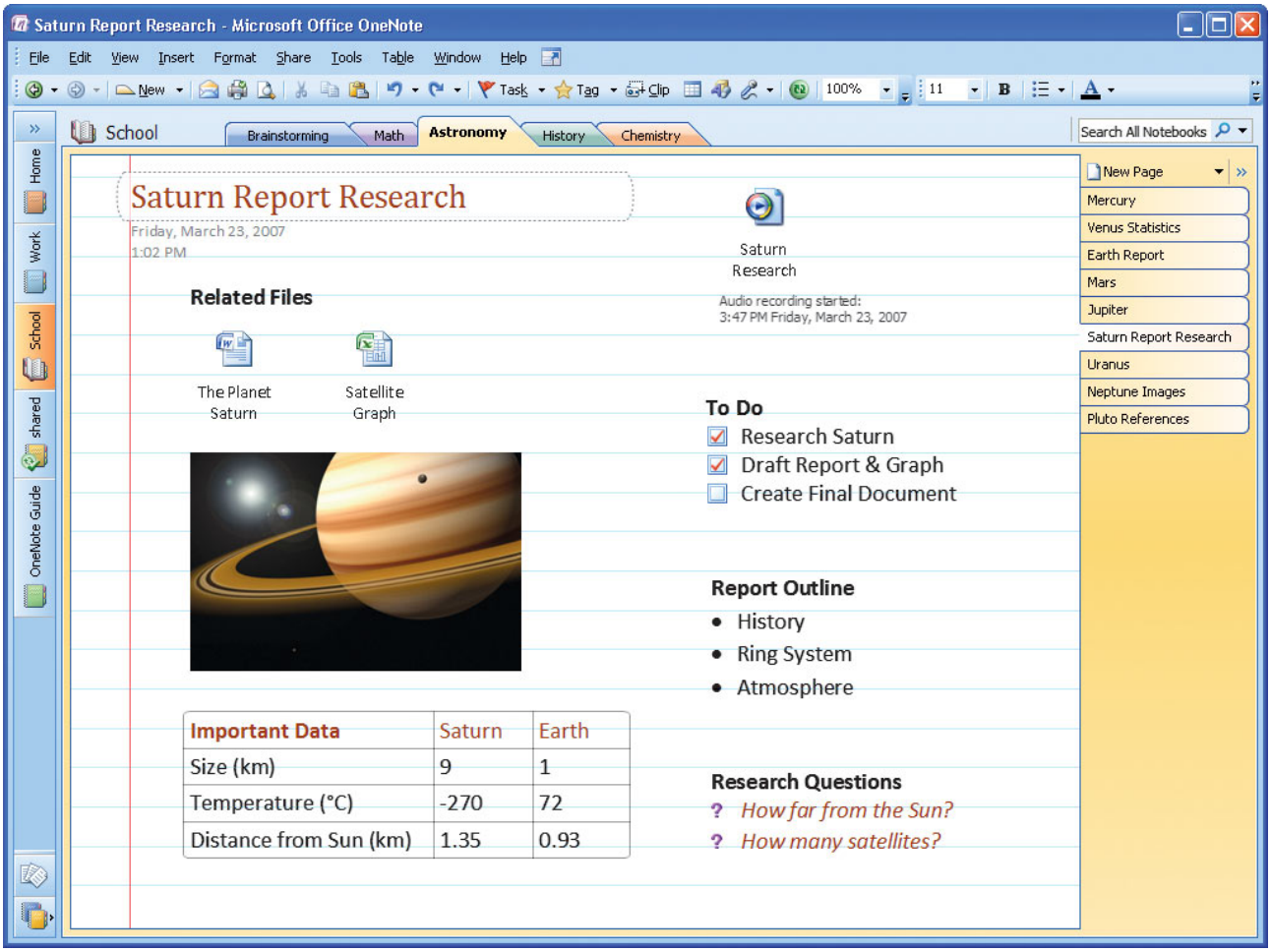
Når vi kigger nærmere på opbygningen af en side, ser vi at elementer er inddelt i bokse, som kan flyttes på frit. Disse bokse fungerer som en slags ”post-its” som kan flyttes rundt på det virtuelle papir som man vil. Det er således meget nemmere at omrokere sin information end det er i papirnoter. Når man vil lave en ny indholdsboks handler det blot om at placere cursoren et sted på den tomme baggrund og begynde at skrive. Dette er meget mere frit end f.eks. Word, som kun kan lave tekst i fast linjer.



Indholdsboksene kan indeholde flere former for indhold: tekst, billeder, lydoptagelser og figurer. Derudover kan en indholdsboks markeres, sådan at man nemt kan finde den igen igennem OneNotes indbyggede søgefunktion. Denne søgefunktion er også en stor forbedring over papirnoter, da der her kun er manuel søgning tilgængelig.

En side kan udvides, så den bliver arbitrært stor, hvilket også er en klar fordel fra papirnoter, desuden kan man også lave arbitrært mange sider i et hæfte uden at det bliver uhåndterbart. I det hele taget er det, at en side kan indeholde andet en tekst og figurer en omfattende forbedring ift. papirnoter.

Nedenstående ses et eksempel, hvor flere typer indhold er indsat på en side.



OneNotes intenderede brug

Microsoft forklarer deres intention med programmet i den medfølgende manual, som er skrevet direkte i programmet selv:

"OneNote er en idebehandler, en notesbog og en informationsorganisator – nogle kalder endda programmet en "udvidelse af din hjerne". Mange synes, at OneNote er uundværligt, når de først er begyndt at bruge det, og det håber vi også, at du gør!"

De forsætter med at understrege områder hvor OneNote kan hjælpe en bruger:

- *Sørge for, at du ikke mister oplysninger, som du synes er vigtige*
- *Organisere oplysningsstumper, der ikke passer ind i e-mail-, kalender- eller formelle dokumenter*
- *Samle og henvise til noter fra møder eller forelæsninger*
- *Hente oplysninger fra internettet eller andre kilder og forsyne dem med noter til dig selv eller andre*
- *Holde styr på dine opgaver og ikke glemme noget*

Dette er alle punkter der kan være gjort i papir tidligere, men som Microsoft nu intenderer at OneNote skal håndtere.

Endvidere er intentionen også at fylde et behov for noter direkte i forbindelse med computeren, dvs. når man arbejder med et andet program eller viser noget i en browser, så kan man nemt og hurtigt kopiere skærmbilleder over til OneNote, i stedet for at skulle referere til det i en håndskreven note. I denne forstand er OneNote en ny form for noteværktøj i stedet for blot et erstatning for noget andet.

OneNotes Aktuelle brug

Vi har som studerende et unikt perspektiv, hvilket betyder at vi som norm ser studerende som brugere af programmet. Studerende har i hvert fald distinkte fordele ved at bruge det, som vi også har noteret tidligere. Til en forelæsning, kan det være bedre at tage noterne i OneNote, hvis man kan gøre det hurtigere og mere præcist. Desuden er der mulighed for at integrere forelæserens slides direkte i OneNote, således at der spares tid, ved at omgå at skulle skrive det samme igen. Derudover kan en studerende bruge OneNote i både øvelser og grupperarbejdssituationer, hvor der skal tages noter, som måske senere skal udsendes til andre (resten af gruppen).

Kravet for at kunne bruge OneNote er en computer eller en håndholdt PC, hvilket betyder, at ikke alle og enhver har mulighed for at bruge programmet. Denne naturlige begrænsning afgrænser brugen af programmet til studerende, excecutives og lignende high-tech miljøer, hvor der er mulighed for at have en computer kørende. Endvidere kan det bruges i hjemmet til at tage noter fra hjemmesider til en given opgave.

Konkrete brugsscenarier

Vi har kigget nærmere på flere konkrete brugsopgaver, som vi analytisk gennemgår og nogle som vi afprøver empirisk med en testperson. Disse brugsscenarier er ret simple, men vi vil her liste de konkrete opgaver, hvorefter vi vil lave hhv. cognitive og activity walkthrough på nogle af dem. Alle tre scenarier testede vi også på vores testbruger, som gennemgik dem og løste opgaverne på den måde, der var mest logisk for ham.

1) Tabeller

Der skal oprettes en 2x2 tabel i OneNote. Denne tabel skal manipuleres, således at der fyldes tekst i alle felter.

2) Tekst og indholdsbokse

Der skal laves flere individuelle indholdsbokse med tekst, som skal flyttes rundt og omorganiseres. Dette er et spring ift. papirnoter, men en vigtig feature i OneNote, hvorfor vi føler at det er vigtigt at få gjort.

3) Indsæt billede og tilpas

Der skal indsættes et billede i forbindelse med research på et konkret emne. Dette billede skal tilpasses siden, således at det er af "passende" størrelse ift. teksten.

Analytiske metoder

Analytiske metoder, forstået overfor empiriske metoder, er en gruppe teorier indenfor HCI, der ikke gør brug af empiriske data, men i stedet søger at sige noget om applikationer ud fra et mere teoretisk perspektiv.

Cognitive walkthrough

Cognitive Walkthrough, altså "kognitiv gennemgang" ("CW"), er en fremgangsmåde indenfor HCI, til relativt let at analysere et interface, både i den teoretiske fase, og med det færdige produkt. Den kræver ikke brugerinteraktion eller et færdigbygget interface. Den består kort sagt i at kortlægge trinvise metoder til udførsel af bestemte opgaver i et givent interface. Når man har klargjort hvilke trin der er nødvendige, vurderes det af forskeren, om hvert enkelt trin er åbenlyst for brugeren, og ligeledes om fuldførelsen af et trin er åbenlyst.

For at udføre en CW-analyse, skal forskeren først vælge en opgave, der skal testes. Denne skal helst være en af de mest almindelige interaktioner med artefaktet, eller dennes intenderede formål. opgaven deles så op i en serie handlinger, brugeren må udføre for at fuldføre opgaven (det er vigtigt at påpege at der kan være flere ruter til samme mål, så CW fungerer som en mål-orienteret kortlægning af artefaktets funktionalitet) – alle mulige veje til målet noteres. Forskeren gennemgår til sidst disse handlinger, og vurderer for hvert trin, om det næste trin er tydeligt for en hypotetisk bruger, og også om det er tydeligt, at et trin er blevet fuldført.

CW er således en teoretisk undersøgelse, og ikke en bruger-orienteret test. Man får indsamlet forskerens vurdering af artefaktet på en rigid facon, men ikke empiriske data om faktiske brugere. Ydermere indsamles der kun information om nogle snævert definerede opgaver, og der kan ikke siges noget om artefaktet som helhed. Hensigten med Cognitive Walkthrough er at spotte problemer i de handlingsserier, brugeren må udføre for at bruge artefaktet korrekt.

Der er nogle praktiske begrænsninger ved teorien, som man må holde sig for øje: den rigide kortlægning af alle handlinger, der skal til for at udføre en opgave, gør teoriens anvendelse på større opgaver næsten uoverkommelig. Man må begrænse sig til relativt simple funktioner indenfor programmet. et andet problem er at teorien ikke rummer nogen metodiske værktøjer til at hjælpe forskeren med at forstå den hypotetiske bruger, andet end sin egen uformelle forestillingsevne, og egne subjektive erfaringer. Ej heller er det let at sige noget om brugerens baggrund for sin opførsel, der sandsynligvis har stor betydning for, om test-opgavens trin er let forståelige (tidligere erfaring med lignende programmer, fx.). Endelig er teoriens trinopdeling determineret af artefaktets konstruktion, der kan være arbitrær. Dette fokus gør det svært at identificere andre trin der måtte indgå i en virkelig brugers interaktion med interfacet. Med disse begrænsninger for øje vil vi analysere et udvalg af opgaver i Microsoft OneNote gennem en Cognitive Walkthrough.

Cognitive Walkthrough på OneNote

Ifølge CW-teorien skal man vælge sin test-opgave udfra artefaktets mest normale brug, eller intenderede formal. Vi går ud fra, at OneNotes egen hjælp-fil kan betragtes som programmets intenderede formål[1]. De opgaver, vi har valgt at teste med CW er de samme som vi har brugt i de empiriske brugertests. Det er en fordel, for så kan vi sammenligne vores analytiske formodninger med de empiriske data.

Til denne analyse vil vi analysere opgaven "at indsætte et billede fra et website". Dette er en normal aktivitet i forbindelse med one-note, og den har indbyggede features til at understøtte brugen af billeder i

[1] Alternativt kunne vi have baseret det på fx. salgslitteraturen, et interview med udviklerne, eller en grundig brugsundersøgelse.

notetagnings-arbejdet. For eksempel indsættes automatisk en billedtekst med URL, timestamp og filinformation, som gør noten indeholdende billedet mere brugbar i det større studie/research-arbejde.

Der er følgende udgangspunkter når man vil indsætte et billede fra en browser (vi vil kun overfladiske beskrive brugeraktivitet udenfor OneNote-vinduet).

1.	Højreklik på billedet i browseren, og gem det på computeren	Hvis browservinduet er maksimeret, gør det mindre
2.	Gå ind i OneNote-vinduet	Med markøren over billedet, hold venstre museknap nede
3.	Vælg ”indsæt” i hovedmenuen	Med museknappen nede, træk markør og billede over til OneNote-vinduet
4.	Vælg ”billeder”	Slip museknappen
5.	Vælg ”fra fil”	
6.	Vælg filen i dialogboksen, dobbeltklik	

Der er altså to mulige måder man kan få et billede ind i programmet på. I nogle tilfælde vil CW-kortlægningen af en opgave afdække et overlap af forskellige løsninger, der deler en eller flere handlinger. I dette tilfælde er der dog ikke noget overlap, men to helt forskellige måder, vi kan kalde A og B, som vi nu kan lave selve Walkthroughen på. A har 6 trin; det første trin er at aktivere kontekst-menuen over billedet i browseren, og klikke ”save image as”. kontekstmenuen er standard for alle windows-programmer, så en hypotetisk bruger vil sandsynligvis have let ved dette trin. At aktivere OneNote-vinduet er ligeledes en triviel hverdagshandling for brugeren. Resten af processen (”indsæt” -> ”billeder” -> ”fra fil” -> [fil]) har OneNote til fælles med resten af office-programmerne, så dette vil sandsynligvis ikke volde nogen problemer for brugeren. Dog kan det indvendes at deres lethed stammer fra velkendthed på tværs af programmer, og at de ikke nødvendigvis er intuitive for en helt ny bruger. Vi må dog formode at en bruger sjældent vil blive konfronteret med OneNote som det allerførste Microsoft-program nogensinde, da OneNotes arbejdsområde ikke hører til de opgaver, der oftest udføres på computere.

Løsningsmuligheden B er til gengæld mere intuitiv, da den gør brug af Drag & Drop-funktionaliteten i stedet for kontekstmenuen og office-menuen. Det første trin er at sikre sig at både browser-vinduet og OneNote-vinduet er synlige på skærmen. Vindues-metaforen er i høj grad intuitiv, ligesom Drag & Drop er det. Det andet trin er at holde musemarkøren over billedet og trykke venstre museknap ned. Ligheden mellem at gribe fat i en fysisk ting, og at holde musetasten nede, er let forståelig for de fleste. Dernæst trækkes billedet over i OneNote-vinduet. Igen, temmelig let at forstå. Det kan yderligere bemærkes, at hvis browseren der er tale om er Firefox, følger et omrids af billedet med når der bruges Drag & Drop, hvilket gør handlingens konsekvenser endnu tydeligere for brugeren (men det er et andet program, så det er kun en sidebemærkning).

Alt i alt kan vi ikke med CW identificere nogle specifikke trin der er mere problematiske end andre, men til gengæld har vi identificeret to mulige måder at udføre denne opgave på, hvor den ene sandsynligvis vil være at foretrække for garvede Windows-brugere, og den anden sandsynligvis vil blive brugt af

førstegangsbrugere af officeprogrammer, der vælger at prøve OneNote som det første, hvor sjældne de end må være.

Activity walkthrough

Activity Walkthrough (AW) er en virksomhedsteoretisk baseret teori, som er skabt som en udvidelse af CW, som tager højde for brugerens kontekst. Brugerens baggrund tages således i betragtning, hvor CW var meget mere fokuseret på de mere banale opgaver. AWs fokus kan siges at være bredere end CW.

AW har 6 faser, som skal gennemgås for at kunne lave en fornuftig analyse, som vi vil gennemgå og bruge nedenfor.

Første fase – preparation

Vi skal her identificere de typiske opgaver, der er nødvendige for en given bruger. Vi vil i denne analyse tage udgangspunkt i en enkelt analyse, da vi er begrænset med plads. Vi kigger på ovenstående eksempel med ”Tabeller”, hvor der skal indsættes en 2x2 tabel, og den skal udvides til 2x4.

Anden fase – kontextualisation

Vi skal nu placere den konkrete opgave i de aktiviteter, hvor den typisk vil blive brugt.

En studerende skal indsætte en tabel, for at konkretisere nogle pointer som en forelæser har vist ved en forelæsning. Informationen skal sorteres i en 2x2 tabel.

Tredje fase – task verification

Vi skal her verificere om de tasks vi har opstillet stemmer overens med den kontekstualisering vi har fundet frem til, og fordi vi arbejder med et simpelt eksempel kan vi hurtigt sige at de stemmer overens.

Fjerde fase – task analysis

Task – Opret tabel	Metode 1	Metode 2
1	Tryk på menuen ”tabel” med mus	Skriv første ord
2	Vælg ”Indsæt tabel”	Tryk tab (2x1 tabel oprettes)
3	Vælg tabel størrelse (2x2)	Skriv andet ord
4	Fyld tekst i	Tryk enter (tabellen udvides til 2x2)
5	DONE	Fyld ord i de sidste celler
6		DONE

Selvom det her ser ud til at det tager færre skridt at klare opgaven med metode 1, er metode 2 tidsmæssigt meget overlegen, da der er store mængder spildtid så snart brugeren skal bruge musen, da hænderne skal flyttes til musen fra tastatur og tilbage igen. Desuden kræver metode 1 at vedkommende husker hvad der skal skrives indtil efter tabellen er oprettet, hvor metode 2 aflaster hukommelsen undervejs.

Femte fase – walkthrough

I dette afsnit skal vi gøre klart, hvorvidt brugeren nemt kan genkende de rigtige metoder. Er metode 1 og 2 tydelige? Metode 1 er klar, når man har genkendt at ”tabel” menuen eksisterer i top menuen, så giver den sig selv. Metode 2 er nærmest skjult for brugeren, hvorfor vedkommende skal have instruktion for at kunne bruger programmet optimalt. Metode 2 er endvidere ikke en standardiseret metode i andre lignende programmer såsom Word, hvorfor en bruger ikke kan drage på tidligere erfaringer for at ”finde” metoden.

Sjette fase – Task analysis verification

Metode 2 er ikke tydelig for en begynderbruger, hvorfor man ikke kan forvente at en begynderbruger nogensinde vil bruge den i den givne situation. Dette er en mangel fra programmørernes side, da brugeren skal lære hvordan programmet virker igennem en tutor eller tutorial. Metode 1 er den metode man kan forvente at de fleste brugere vil benytte, da den logisk giver mere mening, selvom der er lidt tvetydighed mellem ”Tabel” og ”Indsæt” menuen, da de er på samme niveau, og ”indsæt” har inkluderet tabeller i andre Office programmer.

GOMS

GOMS er en forkortelse af Goals, Operators, Methods og Selection rules, altså mål, operatorer, metoder og selektionsregler. Disse fire begreber er de ting, der har indflydelse på GOMS-analysens konklusion. Selve analysen er en meget detaljeret, trinvis gennemgang af en begrænset del af en applikation, for eksempel hvad der skal til for at gøre en bid tekst i en teksteditor fed[2]. Det ville i GOMS-sammenhæng være ”goal”, og alle de enkelte handlinger, brugeren skal udføre for at opnå målet, er ”operators”. Disse kan sammenfattes i ”methods”, bestemte sekvenser af operationer, til opfyldelse af bestemte mål. Et goal kan eventuelt opfyldes af mere end en method. Hvilken method der vælges i en given situation, styres af den sidste bestanddel, selection rules. Denne stringente, serielle gennemgang af operationer minder meget om et computerprogram, hvor nogle af de samme udtryk også går igen.

Der er visse svagheder ved teorien. Den kan ikke håndtere brugere der afviger fra ”programmet”, altså den slaviske gennemgang af operationer i bestemt rækkefølge og med evt. adskillige i hinanden indlejrede målsætninger, for eksempel på grund af fejl. Brugeren formodes at være afklaret mht sine mål, og desuden fuldstændig bekendt med applikationen, så ingen af testens tidsophold skyldes usikkerhed eller fejl.

GOMS-analyse på Onenote

Vi vil bruge GOMS til at analysere en relativt simpel opgave i OneNote: At oprette en tekstboks, skrive en lille tekst (”tortur”), og derefter flytte den. Siden den trinvise gennemgang minder så meget om et programmeringssprog, har vi valgt at skrive GOMS-gennemgangen i pseudokode:

[2] Newman, Lamming (1995), p. 170

Goal: **Skrive ”tortur” i en tekstboks og flytte den** {

Method: **Lav en tekstboks** {

Operator: **Klik med mus et sted i tekstområdet**

}

Method: **Skriv ”tortur”** {

Operator: **tryk på ”T”-tasten**

Operator: **tryk på ”O”-tasten**

Operator: **tryk på ”R”-tasten**

Operator: **tryk på ”T”-tasten**

Operator: **tryk på ”U”-tasten**

Operator: **tryk på ”R”-tasten**

}

Method: **Flyt tekstboks** {

Operator: **Bevæg musen hen til tekstboksens titelbar**

Operator: **Klik på venstre musetast**

Operator: **Flyt musen med knappen holdt nede**

Operator: **Slip venstre musetast**

}

}

Via denne detaljerede gennemgang har vi nu defineret nogle metoder, der kan indgå i andre arbejdsopgaver. Siden vi valgte en meget lille opgave, var der ikke noget tidspunkt hvor vi havde flere valgmuligheder, og vi fik derfor ikke brug for Selection Rules, den sidste del af teorien. Hvis de metoder vi her har defineret skulle genbruges i en større GOMS-analyse af OneNote, kan Selection rules dog komme på tale. Man kunne også tage tid på gennemgangen af ovenstående arbejdsopgave, og sammenligne den med lignende opgaver i andre programmer.

Empiriske metoder

Testsituation

Brugeren blev sat i et kontor med to laptops, en til at bruge applikationen på, og en anden hvor en podcast af en forelæsning kunne køres på. På applikations-laptoppen kørte der et screen recorder program (Wink) indstillet til at optage 4 frames per sekund og ved flytning af cursor og klik af mus. I rummet kørte der en laptop med webcam der optog lyd og billede af brugeren fra siden. Der blev også taget sporadiske notater af brugerens handlinger og observationer. Det var håbet at dette ville indsamle så meget data som muligt til brug i analyse.

Brugeren blev sat ved computeren og bedt om at udføre forskellige brugsscenarier, hvor vi brugte højt-tænking i alle brugsscenarier med undtagelse af to scenarier, der omhandlede notetagning ved "forlæsninger", grundet lyden fra forelæsningen ville forstyrre højttænkningsprocessen.

Situering

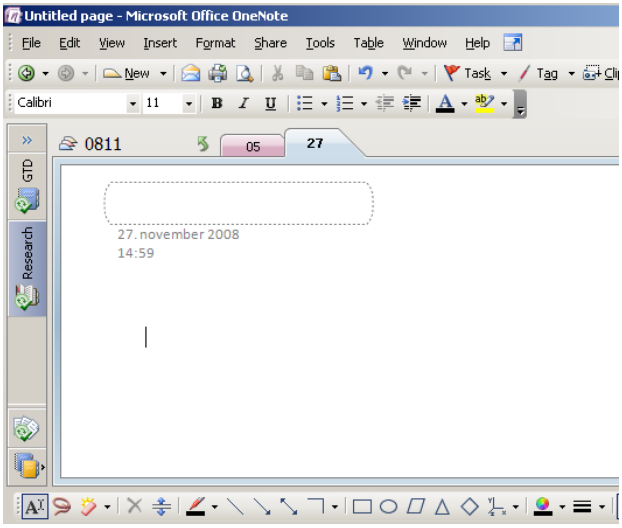
Programmet OneNote er til brug til notetagning. Notetagning er en generel moderne aktivitet indenfor generel virksomhed, til møder, forelæsninger, etc., og denne aktivitet er forklaret tidligere.

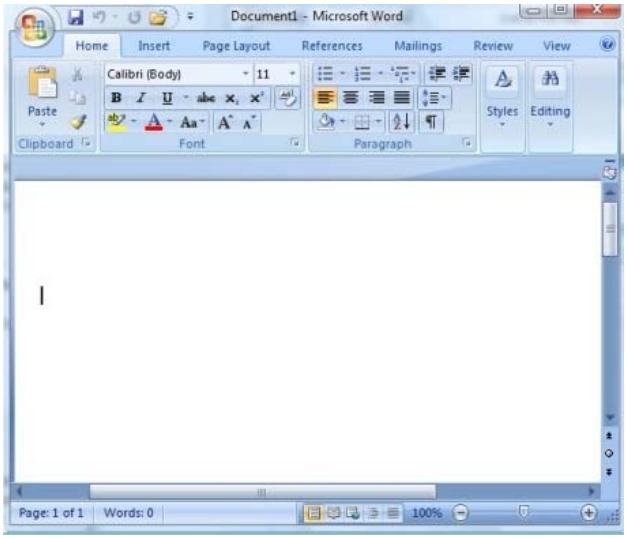
Brugere af OneNote er typisk mennesker, der medbringer deres mobile enheder (tablets, laptops, etc.) til situationer, hvor der er brug for at kunne kigge tilbage på aktiviteterne til mødet/forelæsningen/etc. Disse aktiviteter er de samme som hvor OneNote bruges, men OneNote er ikke et krav for at aktiviteterne kan gennemføres. Brugere af OneNote er også generelt folk som researcher ved hjælp af deres computer, og har brug for at kunne arkivere web-materiale eller andre kilder hurtigt, hvor en digital tilgang til notetagning vil gøre det lettere at opsamle data. Objekterne der er involveret i brugen af OneNote er alt fra browsere / Office-pakken / mobile enheder med notetagning, mikrofoner, etc., som kan bruges til at indsamle data til notesbøgerne i OneNote.

Brugen af OneNote bruges meget traditionelt som med scrapbøger / notetagning. Brugen virker som en interaktiv scrapbog, som udbyder forskellige værktøjer, som man traditionelt bruger i scrapbøger (screen snapshot/saks, cut-and-paste, blyant, tegninger, marker, etc.). Som sådan er OneNotes feature sæt meget lig de virkelige værktøjer og muligheden for at interagere med disse virtuelle værktøjer er, hvad vi kigger på i denne opgave.

Understøttelsen af OneNote i det omliggende miljø som det bruges i, det virtuelle, ligger i brugen af meta-dataen i cut-and-paste/clipboardet i Windows, som indeholder formateringsdata fra rich text editorer, som udnyttes til at kunne udveksle mange forskellige slags formater. I browseren indeholdes der også plugins til at kunne give korrekt kildeangivelse på citater kopieret fra websider. OneNote passer ind i det ikke-virtuelle miljø som en scrapbog ville gøre liggende foran en.

OneNote baserer sig historisk set på det typiske Microsoft Office interface, og har derfor arvet mange af de samme elementer som Word har (formateringsbar, genveje, etc.), men er specialiseret i forhold til nem notetagning. I modsætning til Word 2007 (se følgende figur) har OneNote 2007 ikke det nye design som Office 2007 er blevet profileret på (ribbon).





Vi har valgt at opdele objekterne i OneNote efter devisen omkring en scrapbog - hvor notesbogen er repræsenteret, med de eksterne værktøjer (OneNote), det ydre miljø (Forelæsning, Internet Explorer), brugerinterfacet til operativsystemet/Widgets (text boxes, menu, dialogbokse/wizards).

Det ville have været muligt at inddele i flere objekter, hvis vores skærmoptager ikke havde givet op efter 45 minutters intens optagelse, og selv ved vores kraftige indgriben, ødelagt vores optagede data. Derfor er vores data mangelfuld, men der er forsøgt at rekonstruere det så vidt muligt ud fra vores noter og optagede video og lyd (hvor skærmen ikke klart kunne ses).

Fokus mapping

[nummer] markerer et betydeligt fokusskift, som vi i det følgende diskuterer og dokumenterer.

Transcript	Forelæsning	OneNote	Notesbog	Textbox	Menu	Dialog box
Start tilstand		X				
[1]00:58 Opretter notesbog, spørger efter navn					X	
[2]Skriver actionscript				X		
[3] Vælge som tom notesbog. Ikke prefabrikeret template						X
En computer, ikke flere						X
Gemme i mappe						X
Først vælge denne computer						X
Gå tilbage, vælge skrivebord						X
Og mappe						X
[4] Opretter.		X				
01:53 Selve interfacet		X				
Opdager sektioner og sider		X				
[5] Ny side, actionscript				X		
[6] Har fået oprettet notesbog og sider ved naturlig brug af sektioner (klik for at skrive på stedet)		X				
[7] 02:46: Begyndelse af forelæsning, ikke	X					

højt-tænkning						
[8] Klikke for at oprette skrive felt, skrive. Brug af at lave layoutless noter. Anderledes fra word			X			
Brug af indentering vha keyboardgenveje. Kendt fra word.			X			
Transcript	Forelæsning	OneNote	Notesbog	Textbox	Menu	Dialog box
[9] Brug af bold, CTRL-B vs CTRL-F (fejl)						X
Tilbage til onenote vha anullering			X			
[A] Markering af sætning med mus, markerer for meget / rammer tastatur når man vil lave genvej og tekst slettes. Hurtigt ctrl-z			X			
10:02 færdig med forelæsning uden stop	X					
Anden side til test		X				
Brug af faneblade/hvordan?		X				
10:59 Research et emne: Star Wars, på wiki	X					
19:46 Lave ny sektion (Filer->Ny)					X	
Vælge sektion og kalde Star Wars				X		
Researche	X					
Overfører billeder og noter. Trækker billeder ind. Skriver kilde i Notesbog	X					
Kan ikke skrive ovenpå billede			X			
[10] 22:15: Kan godt resize et billede, kan ikke andet			X			
Holder en størrelse, (canvas?)			X			
Nå, den holder den størrelse			X			
[11] 25:00 Knap med tabel, std icon		X				
Søge efter række under, for at lave 6 rums tabel i stedet for 5 rums					X	
Tryk for at skrive			X			
Fjerne reference, slet. Fed funktion, hvor vi har elementet fra, men kan også let slippes af med			X			
Ukendt handling						
[12] Fik flyttet meget drastisk på billede, fået resizet ham		X				
Nummerering af items		X				
Transcript	Forelæsning	OneNote	Notesbog	Textbox	Menu	Dialog box

Det første fokus brugeren foretager sig efter den oprindelige tilstand (OneNote hoved display), er som del af handlingen han er blevet bedt om at udføre ("Opret notesbog omkring ActionScript"). Brugeren fokuserer på menuen, i OneNote, for at finde instrumentet (File->New->Notebook). Han flytter fokus fra

OneNote displayet til dets menu [1], som del af en intentionel handling, - og skiftet sker grundet et subjektivt ønske om at teste det han bedes om.

Det næste fokusskifte [2], sker grundet at applikationen prøver at opfylde den handling, som den er blevet forespurgt "Opret notesbog", hvor der vises en text box, hvor brugeren spørges om navnet på en notesbog. I fokusskifte [3] ser brugeren muligheden for en template til en notesbog. Brugeren vælger en tom notesbog, i fokusskifte [4], brug af en ”liste over muligheder”-instrumentet, som brugeren fokuserer specifikt på (i vores tabel, noteret som dialog box). Gennem denne proces sker der som del af at opfylde brugerens intention, spørgsmål og lagring og deling, indtil der i [5] sker et intentionelt fokusskifte for at returnere til den oprettede notesbog (målet er opfyldt).

For så at navngive en sektion, sker der i [6] et fokusskift, intentionelt, for at kunne navngive. Instrument der bruges er en tekst boks (faktisk et faneblad med en label der kan editeres). I [7] sker der et fokusskift til forelæsningen, for at kunne lære noget. Fokusskiftet [8], hvor brugeren vil notere noget fra forelæsningen, klikker det sted i noten han vil skrive. Instrumentet der bruges er Notesbogen, og sker som del af et mål om at ville notere.

Der sker et breakdown, i [8]. Brugeren skriver på teksten, og kendt fra anden brug, bruger han ctrl-F genvej for at sætte den markerede tekst som fed skrift. Dette forårsager en 'Find' dialog-boks, hvor brugeren bliver forvirret og annullerer dialogen [9], vælger at bruge knappen i stedet. Dette sker grundet en subjektiv viden omkring genveje, hvor det sker grundet forskel i genveje i forskellige internationaliseringer af Office produkter (ctrl-F(ind) overfor ctrl-F(ed)).

Der sker et breakdown i [A] som ikke direkte registreres af fokusskiftanalyse (med vores objekter, som så kan henføres til vores skærmoptager fejl) - at når man har et tastatur man ikke er vant til at skrive på, er der en større chance for at når man forsøger handlingen "Marker-og-Ctrl-F", at man ikke får ramt Ctrl og derved udføre en genvej, og den markede tekst slettes. Dette bliver dog hurtigt klaret af undo.

I [10] sker der med fokus på notesblokken igen, at der forsøges at resizes et billede - hvor at kanten omkring et billede er meget lig et tekstområde. Et breakdown: der sker det at brugeren bliver forvirret over han ikke kan resize et billede ved at hive i kanten og giver op. Hvor faktisk at kanten han trak i, var indholdsboksen. Denne resizing lykkedes faktisk for brugeren i [12], men det sker på en måde at brugeren ikke opfatter hvad der faktisk gjorde det (at resize billedets kant, ikke canvasset).

Fokusskifte [11] sker for at ville lave en tabel, men får antal kolonner og rækker forkert den første gang. Brugeren søger rundt i menuerne, og får fundet indsæt kolonne formatet. Dog efter at have indsat en tabel inde i en tabel første gang, som kan noteres som et breakdown.

Redesign og optimering

Det er for os at komme med gode bud på, hvordan OneNote kan redesignes, fordi vi ikke har indblik i alle detaljerne for programmet, og vores fokus er ret snævert. Der er dog ting der kan optimeres, således at en almindelig studerende ville kunne bruge programmet nemmere.

OneNote indeholder mange features, som typiske brugere ikke vil bruge, såsom netværkstilgang, så flere brugere kan logge på det samme OneNote dokument. Som sådan lyder dette som en god feature, men i

praksis er featuren ikke nødvendig for de formål vi har analyseret. Dette betyder, at det er en feature man kunne fjerne for at simplificere interfacet.

Det er mange genvejstaster, som almindelige brugere ikke er bekendte med. Dette kan være et problem, hvis den givne bruger skal tage hurtige noter. F.eks. oplevede vi at vores testbruger lavede tabeller med de almindelige menuer, hvor han blot kunne trykke på `tab`, så ville programmet lave tabellen med det samme. Der burde være en oversigt over disse genveje, som blev gjort hurtigt tilgængelig for begynderbrugere.

Refleksion

CW og AW overlapper så meget, at man i en enkelt opgave ikke bør benytte begge metoder. Vi kan dog godt se applikationen af begge to, da de blot skal bruges i hver deres situation. AW kræver mere arbejde, men giver mere holdbare resultater, mens CW udelukkende skal bruges på små snævre problemer.

Vi er nødt til at samle mere empiri, således at vi kan lave en form for fejlkorrektion, når der sker ting som der skete for os (skærmoptageren gik i stykker). Derudover kan vi ikke drage empiriske konklusioner ud fra en enkelt person, da vedkommende ikke nødvendigvis er repræsentativ for andre brugere.

[præ|post] WIMP

Synet på mennesker og maskiner i historisk kontekst.

Carsten V. Munk 20030576
Søren E. Andersen 20061647
Nicki T. Hansen 20030605
11-12-2008

Indholdsfortegnelse

Introduktion

Denne opgave vil se nærmere på præ-WIMP, WIMP og post-WIMP. Vi vil se nærmere på disse tre forskellige paradigmers syn på mennesker og maskiner og inddrage vores analyseværktøjer for derved at belyse deres syn på mennesket og maskinen. Vi vil endvidere sætte analyseværktøjerne ind i en historisk kontekst.

Præ-WIMP[1]

I begyndelsen, var synet på computeren, at det var en industriel maskine, med fokus at få den til at yde optimalt, og synet af mennesket ift. maskinen fulgte derved, at mennesket var en operatør der skulle optrænes i brugen af maskinen.[2] Det var mainframernes tid hvor der blev arbejdet med batch-processing – man gav maskinen data, et program skrevet i datidens sprog, og fik et output. En "simpel" interaktion hvor maskinen fik en ordre, udførte det så godt den nu lige kunne, eller fejlede og meldte dette.

Tanken om brugeren som operatør forsatte med etableringen af time-sharing systemer (ca. 1961), hvor der blev en direkte interaktivitet med maskinen med terminaler, hvor man kunne begynde at snakke om en reel brugerflade, dog tegnbaseret men terminal-kommandoer relateret til visning af raster-grafik eksisterede.[3]

Eksistensen af terminaler, som kunne betragtes som workstations – som det ligger i ordet, værende arbejdsstationer, gjorde det muligt at udbrede computerkraftens muligheder til andre typer professionelle, end bare maskin-operatører. Det medførte naturligvis behovet for HCI,

[1] Vi definerer præ-WIMP som brugerflader før introduktionen af WIMP-interaktion, introduceret af Xerox PARC m. Xerox Alto, i 1973.
[2] Bannon91, p208
[3] Bell86, p11

grundet at programmering af maskinen var udenfor de intenderede brugeres domæne og dette var den traditionelle tilgang til brug af maskinen i de dage.

Den tidlige HCI-tilgang til mennesket ift. maskinen, reflekteret i informationsprocesserings-psykologien, var at se mennesket lig maskine[4], med software (bevidsthed), hardware (hjerne), input (sanser) og output (lemmer), illustreret ved den såkaldte Model Human Processor.

Et af de kvantitative analyseværktøjer denne tilgang medførte, var bl.a. "the keystroke-level model" (KLM).[5] Med fokus på brugerens udførsel af opgaver, meget lig en maskines udførsel af et program, blev der fokuseret på udførselstid, antal fejl, læretiden, funktionalitet m.fl.

Der blev dog indset at modsat maskiner som i samme modeller har samme karakteristikker, at der ingen type-model eksisterer af en bruger. Dette medfører at Card, et al. opstiller KLM som modelleret ud fra antagelsen om at det er en ekspertbruger der udfører opgaven, uden hensyn til menneskers forskellighed.

Selv om denne antagelse eksisterer, giver KLM en mulighed for at beregne tiden det vil tage at udføre et sæt opgaver, ud fra brugen af variable repræsenterende antal gange opgavens operatorer (tastetryk, pegning, tegning, tænkning og respons) bliver brugt og konstanter der siger noget om tiden det tager at bruge disse operatorer.

KLM gjorde det muligt at kvantitativt at sætte to interaktionsmåder op mod hinanden. Det vil sige at man kunne tage to forskellige interaktioner, og se om den ene er mere effektiv end den anden, som et fundament for yderligere analyse.

WIMP

WIMP er en forkortelse for "Windows, Icons, Menus and Pointers", altså vinduer, ikoner, menuer og markører. Det er en type interface, der blev populær sidst i 80'erne med Macintosh-computeren, men havde været under udvikling siden 60erne på bla. Xerox PARC. Det mest udbredte og definitive WIMP-system er Microsoft Windows, som har gennemgået en række versioner, men altid har beholdt WIMP-paradigmet.

WIMP-paradigmet indebærer et bestemt menneskesyn, der drejer sig meget om metaforer. De tidlige GUI'er introducerede nogle klassikere som "Skrivebordet", "Papirkurven", "Mappen" og "Dokumentet", som stadig er en del af de nyeste interfaces. Brugen af metaforer viser et meget fysisk menneske- og verdenssyn, hvor computerens funktionalitet gøres tilgængelig for mennesker ved at krystalliseres til metaforer for virkelige objekter, og dermed bygge ovenpå den menneskelige brugers erfaring om verden fra den almindelige hverdag.

[4] Card83
[5] Card80

Mange analyseværktøjer indenfor HCI er meget tæt knyttet med WIMP paradigmet. Cognitive Walkthrough, for eksempel, har som mål at identificere delopgaver, der er utilgængelige for den almene bruger, som oftest på grund af et forvirrende visuelt interface. I præ-wimp paradigmer som kommandolinjeinterfacet, er alle mulige operationer stort set lige tilgængelige, når det gælder tydelighed for brugeren; de er alle tekststrenge, som det er op til brugeren at huske, eller finde i dokumentationen, og er altså umiddelbart temmelig uudgrundelige for en nybegynder. Her ville en cognitive walkthrough være unyttig, fordi den kun ville give meget pessimistiske konklusioner. Den er WIMP-centrisk, og er et symptom på paradigmets fokus på usability og manglen på samme i foregående paradigmer.

Activity walkthrough er en anden analysemetode, der passer godt in i WIMP-paradigmet. Denne metode er om muligt endnu mere usability-fokuseret end cognitive walkthrough, men på et makro-plan, i stedet for detaljerede beskrivelser af banale opgaver på mikro-plan, som CW hovedsagligt drejer sig om (fx ”er knappen synlig for brugeren?”). Her fokuseres på brugskonteksten, og ikke bare i den traditionelle, tayloristiske forstand, at workflowet skal optimeres til en produktionslinje, men brugerens personlige og/eller kulturelle baggrund. Den relaterer sig til WIMP, idet at den relaterer sig til usability, og skal hjælpe med at udpege svagheder i det grafiske interface.

Usability er i det hele taget WIMPs største styrke. I modsætning til vores to andre definerede paradigmer, præ-WIMP og post-WIMP, er WIMP karakteriseret ved at talrige funktioner er tilgængelige for brugeren på ethvert givent tidspunkt, på en lettilgængelig måde. Som nævnt tidligere, har en kommandoprompt ikke ”knapper” eller en windows-agtig bundlinje med minimerede programmer, elle andre ting der gør multitasking nemt (for brugeren, ikke multitasking i computer science-forstanden). Det er let at se at usability-overvejelser ikke havde en stor rolle i udviklingen af disse systemer. I post-WIMP kunne AW i teorien også bruges, men da dette paradigme er så bredt som det er, er det ikke sikkert at metoden ikke ville skulle omformes til at passe det nye paradigme. Roden i AW, Virksomhedsteori, er dog generel nok til at den sandsynligvis ville kunne fungere som fundament i udforskningen af det nye paradigme, som den har gjort det i WIMP.

Post-WIMP

Hvor WIMP paradigmet er meget godt at bruge til almindelige desktop opgaver, har det sine begrænsninger. Specielt i spil kan vi se de første egentlige eksempler på post WIMP interfaces.

I post WIMP paradigmet fjerner vi elementer fra WIMP, som begrænser arbejdet. Flere spil har helt fjernet konceptet om ikoner og pointers. F.eks. kan vi se spillet Black & White, som var et direkte forsøg på at fjerne ikoner helt fra et strategispil. Her havde man i stedet for en almindelig pointer en ”hånd”, som man kunne bruge til at trække sit kamera rundt i 3D verdenen og for at kaste magi var man nødt til at lave specielle bevægelser (gestures) som gjorde at hånden ændrede

sig til at visuel repræsentation af den spell man ville kaste. Derudover kunne man direkte bruge den guddommelige hånd til at smide med sten og trække træer op af jorden. Der er således en direkte manipulation med spillets verden igennem det komplet transparente interface. Hånden bliver en naturlig forlængelse af brugeren og den forsvinder fra hans bevidsthed som et objekt i sig selv.

Direkte manipulation og transparens

I Black & White får vi således vist to essentielle facetter i post WIMP nemlig *direkte manipulation* og *transparens*.

Direkte manipulation blev defineret af Shneiderman i 1983 og opstod som en mulig løsning på lokale overfor permanente serverbaserede filer. Det at en bruger kunne arbejde *direkte* med en fil, i stedet for at skulle kopiere filen fra den permanente placering på en central server til den lokale terminal, og efter redigering overskrive den permanente udgave med den lokale. Ordet er senere blevet diffuseret af diverse tolkere, som har farvet det til sine egne formål, men stort set betyder det, at en bruger arbejder direkte med et interesseobjekt i stedet for at arbejde med det indirekte, ved hjælp af et værktøj. Det er relateret til Heidiggers vorhanden og zuhanden, hvor man kan anskue et værktøj som et objekt i sig selv eller bruge værktøjet, som derved forsvinder fra bevidsthedssfæren, idet fokus retter sig mod det behandlede objekt i stedet. Interfacet skal således gøre sig usynligt og ikke trænge sig på, hvilket betyder at det skal være hurtigt responsivt, derudover skal en bruger kunne relatere sig til objekterne og værktøjerne for at denne intuitive handling kan finde sted.

Transparens er mere specifikt og betyder det at brugerinterfacet er bygget til ikke at gøre opmærksom på sig selv, således at brugeren kan arbejde uden at opmærksomheden falder på interfacet. Transparens er et element, som er en indbygget del af direkte manipulation.

Det er lidt misvisende, at sige vi har direkte manipulation med objekter i Black & White, fordi vi arbejder stadig igennem eksterne instrumenter. Dvs. der bruges stadig keyboard og mus, når vi tilgår spillet, hvilket derfor er en indirekte interaktion igennem disse instrumenter. Dette er hvad Bertelsen et al kalder *Instrumentness,* hvor brugeren benytter interfacet som et instrument:

> *In simple terms, transparent interaction occurs when the conscious actions are directed to the object of interest and the instrument is interacted with through non-conscious operations; the violin is transparent, when the violinist thinks of the music instead of the violin as such.*

Dette falder igen tilbage på Heidiggers vorhanden/zuhanden, og handler om interfacets transparens.

Syn på menneske og maskine

Post-WIMP anskuer maskinen som værende uafhængig. Værktøjer skal virke over det hele – et skrive apparat skal kunne skrive på alle overflader, der logisk giver mening at kunne skrive på. Interfacet bevæger sig nærmere mennesket og længere fra maskinen i handlemåde. Maskinen integrerer sig i mennesket miljø og der opstår en form for symbiose imellem menneske og maskine, men på menneskets præmisser.

Dimensionalt syn på [præ|post] WIMP

Når vi laver et stort overblik over de tre paradigmer, opstår der et tydeligt spor af udviklingen. I præ-WIMP æraen fungerede computeres interface til mennesker i 2D – et program var dedikeret én skærm, og brugeren arbejdede ikke med mere end det ene program ad gangen.

WIMP paradigmet introducerer ideen med dedikerede vinduer, således at flere programmer kan overlappe hinanden. Man kan sige at man nu arbejder i 3D. Den ekstra dimension i brugerinterfacet gør, at brugeren kan multitaske og arbejde på flere internt afhængige opgaver på samme tid. WIMP har indtil videre vist sig at være det hurtigst og mest effektive til disse ”kontorløsninger”, da oversættelsen fra papirer og mapper til vinduer og *mapper* er nem og hurtig.

Hvor WIMP er en tydelig opdimensionering af præ-WIMP, bliver det lidt mere kompliceret med post-WIMP. Post-WIMP er på mange måder opstået som et modsvar til WIMP, og man kan se mange eksempler, hvor interfaces fjerner elementer fra WIMP paradigmet og derved bevæger sig væk fra grundideen. iPhone fjerner f.eks. ideen om en pointer, da man direkte skal interagerer med objekterne med sine fingre igennem en touchscreen. Pointeren er således flyttet væk fra skærmen og ud på hånden. Instrumenteringen elimineres og der er en tydeligere direkte manipulation med computerens elementer.

Ideen med pervasive computing, hvor interfacet bevæger sig ud over computerens skærm og ud i vores naturlige miljø er en anden form for post-WIMP. Alle overflader er potentielt et interface mellem computeren og mennesket.

Det er en fejl at tænke, at post-WIMP skal bryde komplet med WIMP. I mange applikationer vil det være en dårlig ide at bruge noget komplet anderledes, og når vi kigger på skiftet fra præ-WIMP til WIMP bruges der stadig ideer som kommandoprompt i WIMP miljøer. Når vi ser på det tidligere Black & White eksempel, kan det nævnes at hvor spillet selv arbejder komplet uden ikoner og menuer, er der introduktionsmenuer og statistik i spillet, som kan tilgås igennem menuer. At skulle fremvise disse informationer helt uden vinduer er at komplicere sagen mere end man bør, og post-WIMP paradigmet, hvis man kan kalde det således, bør derfor ses som en opskalering af WIMP, hvor elementer *kan* fjernes.

Litteraturliste

[Bannon91] Bannon, L. From Human Factors to human actors, Greenbaum, J. & Kyng, M. Design at work Cooperative design of computer systems, Erlbaum 1991, pp. 25-44

[Bell86] Bell, G. 1986. Toward a history of (personal) workstations. In Proceedings of the ACM Conference on the History of Personal Workstations (Palo Alto, California, United States, January 09 - 10, 1986). J. R. White and K. Anderson, Eds. ACM, New York, NY, 1-17. DOI= http://doi.acm.org/10.1145/12178.12179

[Card83] Card, S. K., Newell, A., and Moran, T. P. 1983 The Psychology of Human-Computer Interaction. L. Erlbaum Associates Inc.

[Card80] Card, S. K., Moran, T. P., and Newell, A. 1980. The keystroke-level model for user performance time with interactive systems. Commun. ACM 23, 7 (Jul. 1980), 396-410. DOI= http://doi.acm.org/10.1145/358886.358895

Konklusion

Gennem såvel den konkrete anvendelse af analyseværktøjerne, som vores historiske gennemgang af teorierne og paradigmerne har vi dannet et indblik i metodernes anvendelighed indenfor forskellige historiske kontekster. Første delaflevering arbejder direkte med analyseværktøjerne, mens anden aflevering handler om WIMP og det bredere overblik.

Det analyserede objekt i første aflevering, OneNote, fremviser nogle af problematikkerne ved at bruge analyserværktøjerne i et post-WIMP miljø, da OneNote internt fungerer som en post-WIMP program, idet at hver tekstboks kan ses som sit eget vindue, og der derfor er et ”vindue-i-vindue” og en opdimensionering af det traditionelle WIMP.