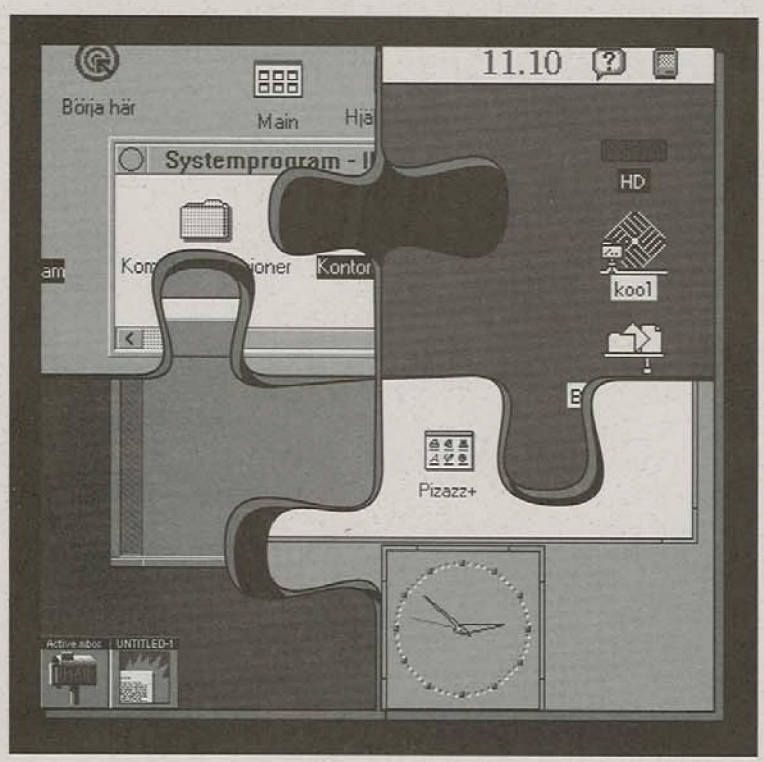

Standarder för grafiska användargränssnitt

RAPPORT NR 19

PEETER KOOL
ULF WINGSTEDT

SVENSKA INSTITUTET FÖR SYSTEMUTVECKLING
SISU

INNEHÅLL

ISSN 1102-1829
ISRN SISU-R--19--SE

FÖRORD

Syfte: I takt med introduktionen av grafiska användargränssnitt hamnar allt fler i en situation där man överväger att standardisera alla företagets program enligt någon av de industristandarder för grafiska användargränssnitt som finns på marknaden. Det kan handla om att välja MS Windows eller Macintosh, Motif eller Open Look et c. Allt för att användarna ska känna igen sig i olika program och kunna arbeta mer effektivt.

I denna rapport har vi, utifrån specifikationerna till de mest betydande industristandarderna på marknaden, först beskrivit standarderna var för sig och sedan jämfört deras innehåll, för att underlätta beslut om standardisering i svenska företag och förvaltningar. Förklaringar till många av de begrepp som används flitigt i gränssnittssammanhang ges också.

Målgrupp: Rapporten riktar sig främst till datastrateger och alla andra i en organisation som tänker utvärdera och välja en standard för grafiska användargränssnitt. Även den som bara vill veta mer om vad som rör sig i gränssnittsdjungeln kan finna mycket nyttigt.

Bakgrund: I samband med publiceringen av denna rapport ges också SISU Rapport nr 20 ut. Den heter *Verktyg för grafiska gränssnitt* och behandlar olika former av programmeringshjälpmedel för utveckling av program med grafiska användargränssnitt enligt de standarder som beskrivs i denna rapport. I *Verktyg för grafiska gränssnitt* beskrivs de olika typer av verktyg som finns på marknaden. Dessutom provas ett antal verktyg av olika typer mer ingående mot ett gemensamt praktikfall.

Läsanvisning: Efter en kort sammanfattning följer ett introduktionskapitel där de viktigaste begreppen i gränssnittssammanhang beskrivs.

De följande kapitlen, 3 till 8, kommer att behandla specifikationerna av marknadens viktigaste de facto standarder för grafiska användargränssnitt. Fem av dem hör till bjässarna på marknaden, de används på i stort sett samtliga moderna datorer med tillräcklig prestanda för grafik.

De facto standarderna är:

Common User Access, CUA, från IBM

MS Windows från Microsoft

Motif från Open Software Foundation

Open Look från Unix International

Macintosh från Apple

Vi har också valt att ta med en intressant nykomling och uppstickare:

- Nextstep från Next.

Kapitel 9 innehåller en beskrivning av de ansträngningar som görs inom det internationella standardiseringsorganet ISO för att nå fram till en *officiell* standard.

I kapitel 10 sammanfattar vi och jämför de olika standardförslagen. Därefter kapitel 11 med referenser.

Den som bara vill få en översiktlig bild av hur gränssnittsdjungeln ser ut kan läsa sammanfattningen och introduktionen (kapitlen 1 och 2) samt kapitel 10. Den som är intresserad av en speciell standard kan läsa det kapitel som handlar om just den standarden, medan den som vill få en tydligare bild, och kunna jämföra de olika standarderna, bör läsa hela rapporten.

1. Sammanfattning

Ett grafiskt användargränssnitt kännetecknas framförallt av att det åskådliggör programmets funktion med hjälp av ett antal grafiska symboler. Användaren kan dessutom kommunicera med programmet genom att klicka på dessa grafiska objekt, t ex knappar eller menyer, med hjälp av en markör som manövreras med något pekdon, t ex mus, rullboll, penna eller piltangenter.

Det finns många exempel på nyttan av grafiska användargränssnitt. Vanligen brukar man hävda att produktiviteten höjs eftersom program med denna typ av gränssnitt är lättare att använda och med grafikens hjälp kan programmens funktion åskådliggöras bättre. Andra exempel på nyttan med grafiska användargränssnitt är att de medför lägre inlärningströskel, är roligare att använda och att de lättare accepteras av användarna.

FUNKTIONEN ÅSKÅDLIGGÖRS

Till varje persondator och arbetsstation hör idag ett grafiskt användargränssnitt. Datortillverkarna gör försök att standardisera till ett enda gränssnitt först och främst för sina egna produkter, men ingår också ofta i olika sammanslutningar med andra tillverkare där målet är att nå fram till ett gemensamt gränssnitt. Det finns därför ett antal olika användargränssnitt som blivit mycket vanliga på marknaden och närmast kan betraktas som industristandarder.

INDUSTRISTANDARDER

Det är viktigt att skilja på ett gränssnitts *specifikation* av utseende och beteende, "Look-and-feel", och dess *realisering* på någon dator. I denna rapport beskrivs specifikationerna för följande industristandarder:

- *Common User Acces* (CUA) från IBM
- *MS Windows* från Microsoft
- *Motif* från Open Software Foundation (OSF)
- *Open Look* från Unix International
- *Macintosh* från Apple
- *Nextstep* från Next

Dessutom beskrivs de ansträngningar som görs för åstadkomma en officiell standard inom det internationella standardiseringsorganet ISO.

ISO

Rapporten påvisar ett antal skillnader mellan de olika industristandarderna, men dessa är av relativt ytligt karaktär.

Även om vissa standardleverantörer gärna hävdar att det egna förslaget är det enda som ger användarvänliga program, så kan vi här konstatera att i stort sett lika bra, eller för den delen lika dåliga, gränssnitt kan åstadkommas enligt alla industristandarder.

LÅG NIVÅ

Ett problem med föreliggande standarder är att de håller sig på en mycket låg nivå vad beträffar sättet att utföra uppgifter med dator. Det enda som klart specificeras är utseende och beteende för exempelvis knappar och rullningslister, medan mer avancerad interaktion som t ex sökning i databaser eller navigering i bildbibliotek lämnas därhän. Industristandarderna koncentreras mest på interaktionsobjektens utseende och mindre på de större användningsproblemen. Den officiella standarden från ISO har därför en viktig uppgift att fylla med sin ansats att standardisera på användbarhet snarare än "knappars och fönsters utseende".

CUA och i viss mån Open Look framstår som de bäst specificerade standarderna, både vad gäller innehåll och detaljeringsgrad. De har högre ambitionsnivå än sina konkurrenter genom att de förutom att ange interaktionsobjektens utseende och beteende, också försöker slå ett slag för de större problemens lösning. Speciellt innehåller CUA-specifikationen många goda råd om gränssnittsdesign som alla utvecklare har nytta av att ta del av.

Datorer med nya former för interaktion har redan introducerats eller kommer att göra det inom en snar framtid. För dessa nya interaktionsformer saknas riktlinjer i alla existerande standarder. I alla standarder görs, underförstått eller uttryckligen, antagandet att interaktionen med datorn sker med tangentbord och eventuellt en mus.

ENHETLIGT GRÄNSSNITT OLIKA DATORMÄRKEN

En klar trend är att den hårda bindningen mellan datorfabrikat och gränssnittsstandard kommer att försvinna. Resultatet på sikt kommer troligen bli att ett företag kan bestämma sig för en gränssnittsstandard utan att vara låst till en datortillverkare, det kommer att vara möjligt att blanda datormärken men ändå behålla ett enhetligt gränssnitt.

2. Introduktion

Ett gränssnitt kan i allmänhet beskrivas som den teknik eller mekanism som möjliggör förbindelse eller kommunikation mellan två olika enheter. Ett exempel är en vanlig stickpropp som i Sverige har ett visst gränssnitt för att kunna stoppas i en väggkontakt, i Storbritannien ett annat.

När den ena kommunicerande enheten är en människa, en användare, och den andra en apparat talar vi om användargränssnitt, d v s den teknik en människa utnyttjar för att använda apparaten. Telefonens knappar och bilens ratt och pedaler är alla exempel på användargränssnitt.

Ovanstående resonemang innebär att tillvägagångssättet vid användning av datorprogram också kan betecknas som ett användargränssnitt. Vi kan grovt urskilja tre typer av användargränssnitt:

- *Kommandostyrda* användargränssnitt — **KOMMANDO**
- *Menystyrda* användargränssnitt — **MENY**
- *Grafiska* användargränssnitt — **GRAFIK**

I ett kommandostyrt användargränssnitt styr användaren programmet genom att ge skriftliga kommandon via ett tangentbord. Ett menystyrt användargränssnitt presenterar ett antal valmöjligheter; användaren kan sedan göra ett val genom att via tangentbordet ange t ex numret eller första bokstaven för något alternativ. Menyalternativen är ofta organiserade i någon form av hierarki.

Ett grafiskt användargränssnitt kan förvisso innehålla element från båda de ovan beskrivna användargränssnittstyperna men kännetecknas framförallt av att programmen visualiseras av ett antal grafiska symboler. Användaren kommunicerar med programmet genom att klicka på olika grafiska objekt, t ex knappar eller menyer, med hjälp av en markör som manövreras med något pekdon, t ex mus, rullboll, penna eller piltangenter.

Grafiska användargränssnitt kan vara mer eller mindre *direktstyrda*. Direktstyrning innebär att grafiska objekt i programmet direkt kan påverkas på ett sätt som är likt påverkan av objekt i verkligheten. Användaren känner igen sig och förstår intuitivt mycket av hur man kan arbeta. Ett grafiskt användargränssnitt är direktstyrt om man t ex kan flytta en fil genom att fysiskt flytta den grafiska bilden av filen. Direktstyrning är en många gånger önskvärd egenskap hos ett användargränssnitt eftersom det tydligt visar effekterna av en utförd operation. — **DIREKTSTYRNING**

Det finns många exempel på nyttan av grafiska användargränssnitt. Vanligen brukar man hävda att produktiviteten höjs eftersom program med denna typ av gränssnitt är lättare att använda, med grafikens hjälp kan programmens funktion åskådliggöras bättre. Andra exempel på nyttan med grafiska användargränssnitt är att de ger en lägre inlärningströskel, är roligare att arbeta med och lättare accepteras av användarna.

De grafiska gränssnitten började användas på bred front först i samband med introduktionen av arbetsstationer och då ofta inom specifika tillämpningar som till exempel CAD/CAM (Computer Aided Design / Computer Aided Manufacturing). Utvecklingen av grafiska användargränssnitt fick sitt genombrott 1981 i och med att Xerox lanserade sin arbetsstation Star [Johns89]. Det revolutionerande med Xerox/Star var att man här för första gången kommersiellt använde en teknik med överlappande fönster för att tillåta att flera program samtidigt var aktiva och synliga på skärmen. Principerna bakom användargränssnittet i Star grundades i sin tur på ett flertal prototyper under 60- och 70-talet där den objektorienterade utvecklingsmiljön Smalltalk, även den från Xerox, var den kanske främste inspiratören.

FÖNSTRETS URMODER

Trots att Xerox/Star inte fick någon större spridning utanför akademiska kretsar har den alltså fått stor betydelse för den senare utvecklingen. I princip kan man säga att alla på marknaden idag förekommande grafiska gränssnitt bygger på samma principer som introducerades i Xerox/Star och Smalltalk, och att de inte skiljer sig på några avgörande punkter från fönsterteknikens urmoder. Välkända standarder för grafiska gränssnitt som CUA, Motif eller Open Look är alla baserade på fönstret som det grundläggande interaktionsobjektet.

Moderna datorers prestanda har inneburit att det nu är möjligt att använda direktstyrda grafiska användargränssnitt på i stort sett alla typer av persondatorer och arbetsstationer. De förväntade nyttoeffekterna av direktstyrda grafiska användargränssnitt innebär att de röner ett stort intresse bland svenska företag. Resten av denna rapport ägnas därför åt denna typ av användargränssnitt. Det kortare uttrycket "gränssnitt" kommer här allmänt att betyda grafiskt användargränssnitt, om inte annat anges.

De flesta datoranvändare utnyttjar idag flera olika program för att utföra sitt dagliga arbete, t ex faktureringsprogram, CAD-program, kalkylprogram och ordbehandlare. Dessutom ökar antalet program per användare. Vad vi då ofta upplever är att gränssnitten på de program vi använder inte stämmer överens med varandra utan skiljer sig både till utseende och funktion.

Bristen på överensstämmelse mellan olika programs gränssnitt hotar därför att reducera de nyttoeffekter och produktivitetsvinster som de moderna gränssnitten annars utlovar. En lösning på det problemet är att olika programleverantörer kommer överens om hur programmen ska

användas, d v s att de utvecklar standarder för programmens grafiska gränssnitt. I resten av rapporten kommer vi att diskutera olika förslag till sådana standarder.

2.1 Standarder

Det är egentligen felaktigt att påstå att det existerar någon standard för grafiska användargränssnitt, även om det pågår arbete med att komma fram till en sådan (ISO). Med standard menar vi i den här rapporten snarare *de facto* standard, *informell* standard eller *industri*-standard. Anledningen är att utvecklingen helt är i händerna på dator- och programtillverkarna, som enskilt eller i olika sammanslutningar försöker genomdriva ett stort antal skilda men, som vi kommer att se, ändå likartade standardförslag.

Till varje persondator och arbetsstation hör idag ett grafiskt användargränssnitt, t ex MS Windows för PC eller Motif för Decstation m fl. Datortillverkarna gör försök att standardisera till ett enda gränssnitt först och främst för sina egna produkter, men ingår också ofta i olika sammanslutningar med andra tillverkare där målet är att nå fram till ett gemensamt gränssnitt. Ett exempel på en sådan sammanslutning är Open Software Foundation (OSF), där IBM, Digital och Hewlett Packard hör till de tongivande medlemmarna. Datortillverkarna i OSF erbjuder alla gränssnittet Motif på sina datorer.

OPEN SOFTWARE FOUNDATION (OSF)

Även om det är viktigt att följa en gränssnittsstandard vid utveckling av nya program, är det dock inte ett tillräckligt villkor för att programmet ska bli lätt att använda. Standarder i sig är i allmänhet relativt neutrala vad gäller de viktiga valen av *metafor* och *användningsmodell*, två aspekter av användargränssnitt som programutvecklaren ingalunda kan bortse från. En metafor brukar definieras som en "systematisk liknelse med tidigare känd verklighet" och används ofta för att förenkla förståelsen av program. Ett vanligt exempel är den s k skrivbordsmetaforen, som genom att använda symboler som dokument, mappar och papperskorg liknar filhantering på datorn med arbetet vid ett skrivbord. Om utvecklaren gör ett olämpligt val av metafor, t ex en hällristningsmetafor för en ordbehandlare, blir programmet svårt att använda även om det följer standard till punkt och pricka.

METAFOR

Gränssnittet måste också vidarebefordra en lämplig användningsmodell till användaren, d v s få användaren att inse hur han på bästa sätt kan utnyttja programmet för att uppnå ett visst mål. Om användningsmodellen innebär att operation A bör utföras före operation B för bästa resultat, så måste gränssnittet tydligt visa detta för användaren.

ANVÄNDNINGSMODELL

Att realisera ett grafiskt användargränssnitt för ett program enligt en standard, handlar inte så mycket om innovation utan mer om sammansättning av delar. Programmerarens uppgift är att besluta om hur delarna ska fogas samman, till bäst nytta för användaren. På samma sätt som bilkonstruktörer förser varje bil med en ratt, så måste programmeraren förse programmet med vissa egenskaper för att det ska passa in i användningsmiljön. Standarder erbjuder därför inte bara en specifikation över gränssnittsdelarnas utseende utan, i bästa fall, också över deras kombination och funktion.

2.2 Specifikation och realisering

STILGUIDE

Det är viktigt att skilja på ett gränssnitts *specifikation* av utseende och beteende, "Look-and-feel", och dess *realisering* på någon dator. En gränssnittsstandards specifikation kallas ibland för stilguide (eng style guide) och innehåller riktlinjer för hur ett gränssnitt måste utformas för att det ska följa standarden i fråga.

Den entydiga koppling mellan gränssnitt och datortyp som ofta görs är ingen nödvändighet, utan det är i princip fullt möjligt att realisera vilken gränssnittsstandard som helst på valfri datortyp. När vi i denna rapport diskuterar standarder gör vi det med utgångspunkten att de endast är beskrivningar av interaktionsobjektens utseende och beteende, d v s hur de ser ut på skärmen samt vad som händer när de aktiveras.

X WINDOWS

X Windows är ett bra exempel på hur olika industristandarder kan baseras på samma plattform. X Windows är en industristandard för ett kommunikationsprotokoll mellan program och användargränssnitt. X Windows säger inget om gränssnittets utseende och beteende utan definierar endast *hur* ett program kan visa sitt gränssnitt på en dator. På detta kommunikationsprotokoll har sedan flera olika industristandarder för användargränssnitt specificerats.

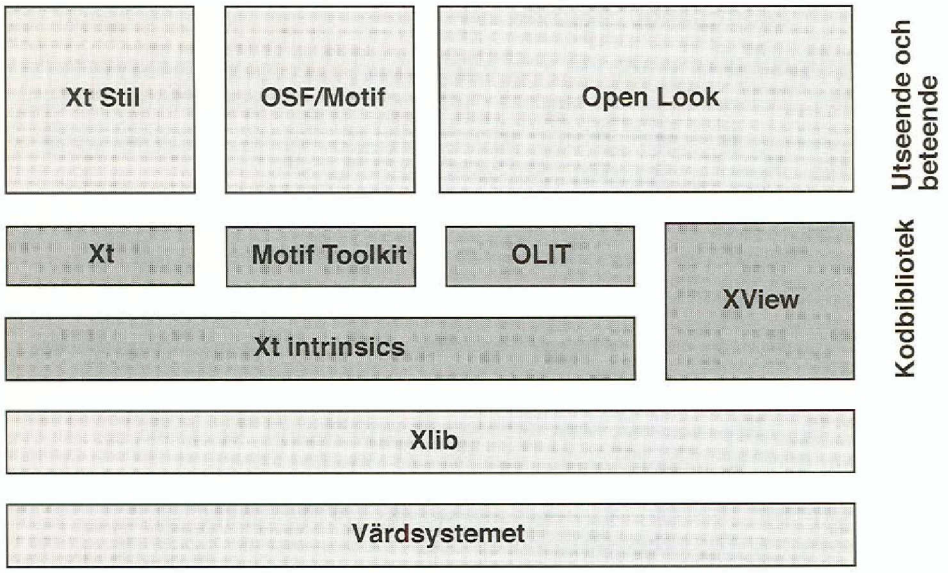

Bild 1. *Olika gränssnitt för samma X Windows-plattform*

I bild 1 visas några möjliga systemlösningar som alla kan baseras på en Unix-arbetsstation som använder X Windows. I botten finns först värdsystemets datorutrustning och systemprogram. Därutöver finns ett lager med de grundläggande grafikrutinerna för X Windows, det s k Xlib. Xlib innehåller generella grafikrutiner som kan användas i valfri gränssnittsstandard.

VERKTYGSLÅDOR

Grafikrutinerna i Xlib har sedan kombinerats i olika kodbibliotek eller verktygslådor (eng toolkit), till komponenterna i de olika gränssnitten, d v s interaktionsobjekten. Programmeraren kan välja den verktygslåda som passar till den standard han valt, t ex Motif eller Open Look. Gränssnitten till program som realiserats med hjälp av verktygslådorna Motif Toolkit eller OLIT (se bild 1) kan i princip köras samtidigt på samma dator, även om blandningen av gränssnittsstandarder förvirrar.

Mer om hur gränssnitt realiseras finns att läsa i SISU-rapport nr 20: Verktyg för grafiska gränssnitt [SISU92].

2.3 Interaktionsprinciper och viktiga begrepp

När man diskuterar gränssnitt och olika standarder för sådana, finns det ett antal vanliga begrepp som ofta dyker upp. Vi ska här försöka ge en kortfattad beskrivning av några sådana begrepp som vi kommer att använda i resten av rapporten.

De grafiska objekt som utnyttjas av användaren för att kommunicera, eller interagera, med ett program brukar kallas *interaktionsobjekt*. Några vanligt förekommande typer av interaktionsobjekt beskrivs nedan.

IKONER

Filer, dokument eller program kan representeras av en grafisk symbol på skärmen. Denna kallas då *ikon*. I direktstyrda gränssnitt kontrollerar ofta användaren program genom att manipulera ikoner, t ex med hjälp av "dra-och-släpp"-principen (eng"drag-and-drop").

MENYER

En meny är en lista av fördefinierade kommandon som användaren kan välja bland. Menyer är ofta hierarkiskt indelade, d v s val av ett visst menyalternativ leder till en undermeny med fler alternativ.

Det finns huvudsakligen två typer av menyer i grafiska användargränssnitt:

- Rullgardinsmenyer, som är fästa på en speciell, fix, plats exempelvis i överkanten av skärmen eller ett fönster.
- "Pop-up"-menyer som kan dyka upp var som helst på skärmen. Till exempel kan användaren genom att hålla en musknapp nedtryckt få en meny att dyka upp.

Val av menytyp är ofta en smaksak men pop-up-menyer kan vara lämpliga för kommandon som ofta används eftersom de kan nås utan att markören behöver flyttas.

KNAPPAR

Knappar används för att göra olika val mellan kommandon och operationer. När en viss knapp trycks ner så utförs en viss operation. För utom denna grundform finns det huvudsakligen tre olika varianter av knappar:

- Av/på-knappar, som kan anta ett av två tillstånd. När knappen trycks ned byter den tillstånd.
- Radioknappar (eng radio buttons), som alltid förekommer i grupper om minst två stycken. Minst en av knapparna i gruppen är alltid vald. När någon annan knapp i gruppen väljs, blir den som varit nedtryckt återställd i uppsläppt läge.
- Flervalsknappar, som liknar radioknappar men där flera knappar kan vara valda samtidigt.

Notera att knappar i gränssnittssammanhang är ganska generella objekt som används på ett flertal sätt. Gemensamt för dem alla är dock att användaren utnyttjar dem genom att klicka på dem.

RULLNINGSLISTER

Rullningslister används för att bläddra i eller rulla t ex en bild som inte får plats i sin helhet i det fönster där den visas.

DUBBELKLICK

Med att dubbelklicka menas att användaren i snabb följd trycker ned musknappen två gånger. Dubbelklick är en vanlig metod för att starta program med hjälp av direktstyrning, man dubbelklickar då på programmets ikon.

FÖNSTER

Fönster är kanske det mest grundläggande interaktionsobjektet i dagens gränssnittsstandarder och finns i många olika varianter. Först och främst finns det tillämpningsspecifika fönster där det huvudsakliga arbetet med ett program sker, t ex ett fönster där text matas in i en ordbehandlare. I dialogen med användaren kan sedan ett antal andra fönster användas:

- *Meddelandefönster*, där meddelanden från programmet till användaren visas. Programmet kan i en dialogbox också kräva att användaren tar ställning till någon fråga, t ex om användaren verkligen vill avsluta programmet utan att spara det gjorda arbetet. Ett meddelande visas ofta mitt på skärmen, framför alla andra fönster för att tvinga användaren till att ta ställning till något viktigt.
- *Dialogboxar* som hanterar mer avancerad dialog än meddelandefönster. En dialogbox kan liknas vid ett formulär där information kan fyllas i eller ändras.
- *Paletter* som finns synliga på skärmen under en längre tid. Från en palett kan användaren hämta objekt av olika slag. Paletter är vanliga i t ex ritprogram där olika ritverktyg eller färger kan väljas. De används lämpligen i program där man har ett antal alternativ att välja bland.

TANGENTBORDS-KOMMANDON

Det finns tre olika sätt att via tangentbordet ge kommandon till program med grafiskt gränssnitt:

- Funktionstangenter, som är en sorts anpassningsbara knappar som varje program kan tilldela specifika uppgifter. Knapparna heter ofta något i stil med F1, F2 o s v.
- Etikettknappar (eng labelled buttons), som ska ha samma uppgifter i alla program. T ex kan speciella etikettknappar tilldelas vanliga kommandon som "kopiera" eller "starta program".
- Kortkommandon, som använder de vanliga tangenterna för att i unika kombinationer, ofta tillsammans med någon specialtangent som t ex Control eller Escape, ge den vane användaren genvägar till olika kommandon specifika för ett visst program.

De olika typerna av tangentbordskommandon kan naturligtvis tjäna samma syften. Exemplevis kan etikettknappar också användas för att ge möjlighet till en mer effektiv dialog.

En dialog är antingen *modal* eller *icke-modal*. Modala dialoger delar in dialogen i olika tillstånd där det i varje tillstånd endast finns ett fåtal alternativ för användaren att ta ställning till.

DATORN ELLER ANVÄNDAREN

De moderna grafiska gränssnitten strävar efter att vara så icke-modala som möjligt, d v s programmet ska inte tvinga användaren att tillämpa ett visst arbetssätt eller viss sekvens utan tillåta att användaren själv be-

stämmer vad som ska göras. Med en icke-modal dialog styr användaren programmet i stället för tvärtom.

I vissa fall är det ändå önskvärt att dialogen är modal. Om programmet till exempel identifierar ett viktigt fel som kräver omedelbar åtgärd, kan det meddela detta till användaren i en modal dialogruta. Användaren kan då inte göra någonting på datorn utan att först ta ställning till dialogrutans innehåll.

Man hör ofta talas om objektorienterade gränssnitt. Det är då viktigt att veta att den typ av objektorientering som åsyftas i dessa sammanhang inte har mer än ett avlägset släktskap med t ex objektorienterad programmering eller objektorienterade databaser.

I ett objektorienterat gränssnitt ser man interaktionsobjekt och tillämpningsdata som självständiga *objekt* på skärmen. Objekten kan påverkas med en uppsättning olika verktyg, meddelanden eller händelser. Att dubbelklicka på ett objekt som t ex ett program är en händelse som startar programmet. Att applicera kommandot "ändra storlek" är ett meddelande som förändrar ett objekt, t ex en cirkel.

Interaktionen i objektorienterade gränssnittt kan ske på två olika sätt:

- *Handling-objekt*, som innebär att man först väljer vad som ska göras, handlingen, och sedan vilket eller vilka objekt som ska påverkas av handlingen. Detta tillvägagångssätt är vanligt i äldre gränssnitt, t ex CAD/CAM-program från mitten av 80-talet.

OBJEKT-ÅTGÄRD DOMINERAR

- *Objekt-åtgärd*, som innebär att man först väljer ett eller flera objekt och sedan en eller flera handlingar som påverkar objekten.

Interaktion enligt principen objekt-handling, är idag helt dominerande.

DIREKTSTYRNING

Att direkt kunna påverka grafiska objekt på skärmen med t ex en mus, brukar kallas *direktstyrning* eller *direkt manipulation*. Ben Shneiderman, som myntade begreppet, [Shnei83] studerade några framgångsrika program, bl a kalkylprogrammet Visicalc, för att försöka ta reda på varför de var så användbara. Han fann att programmens viktigaste gemensamma egenskap var att de minskade avståndet mellan användaren och programmet genom att utnyttja direktstyrning. Användarna kunde direkt påverka objekt på skärmen istället för att gå omvägen via något kommandospråk.

Det finns många exempel på direktstyrda program. Det kanske vanligaste är ritprogram där man enkelt kan placera en cirkel på ritytan genom att välja cirkel i en palett och sedan klicka på den plats där den ska ligga. Cirkeln kan kanske sedan göras större genom att man drar i den med musen. Det icke direktstyrda alternativet kan t ex innebära att man skriver kommandot: cirkel(123, 456), där siffrorna står för dess koordinater. För att förstora cirkeln måste ett motsvarande kommando skrivas.

Ett direktstyrt gränssnitt bör ha följande egenskaper:

- En kontinuerlig representation på skärmen av de, för användaren, intressanta objekten. Kontinuerlig representation innebär att objekten alltid finns tillgängliga på skärmen, även när de inte är aktiva.
- Fysiska handgrepp istället för skrivna kommandon.
- Snabba, korta, stegvisa operationer vars effekt omedelbart blir synlig och som dessutom kan ångras genom att utföra operationerna i omvänd ordning.

Ytterligare exempel på hur direktstyrda gränssnitt lyckosamt har kunnat utnyttjas är naturligtvis alla de datorspel där användaren kör bil, skjuter med pistol o s v. Frånvaron av användarhandledning hindrar varken novisen eller proffset som lärt sig behärska programmet till fulländning.

Dra-och-släpp (eng "drag-and-drop") kallas en teknik som innebär att man genom att först med markören dra och sedan släppa ett objekt kan utföra en viss åtgärd. Till exempel kan ett program startas genom att dess ikon dras ut på skärmytan från något fönster och släpps.

Ett annat exempel på användningen av dra-och-släpp är att till exempel dra ikonen för ett dokument och släppa det på bakgrunden, "skrivbordet", varvid ett program startas med dokumentet som indata. Vilket program som väljs beror på vad som är standard för den aktuella dokumenttypen. Ytterligare exempel på nyttan med dra-och-släpp är att en fil med indata kan släppas på olika typer av program. Om man till exempel släpper ikonen för ett dokument med C-källkod på ikonen för en kompilator kommer koden att kompileras. Om C-koden istället släpps på ikonen för en editor startas den med C-koden inladdad, där den sedan kan redigeras.

Dra-och-släpp användes konsekvent redan i datorn Star från Xerox men har först på senare tid blivit populär i andra miljöer, speciellt där direktstyrning utnyttjas.

3. Common User Access (CUA)

CUA är en välspecificerad industristandard som ingår som en del i IBM:s System Application Architecture, SAA. Interaktionsobjekten är specificerade på detaljnivå vad gäller beteende. Däremot specificeras inget specifikt utseende på interaktionsobjekten. Gränssnitt från CUA används idag framförallt av program som körs på presentation manager OS/2.

CUA är uppdelad i tre böcker: Basic Interface Design Guide [IBM89], Guide to User Interface Design [IBM91] och Advanced Interface Design Reference [IBM92].

Basic Interface Design Guide är en gränssnittsspecifikation för teckenbaserade system och faller därför utanför ramen för denna rapport. Den del av CUA som tas upp här avser därför Guide to User Interface Design och Advanced Interface Design Reference.

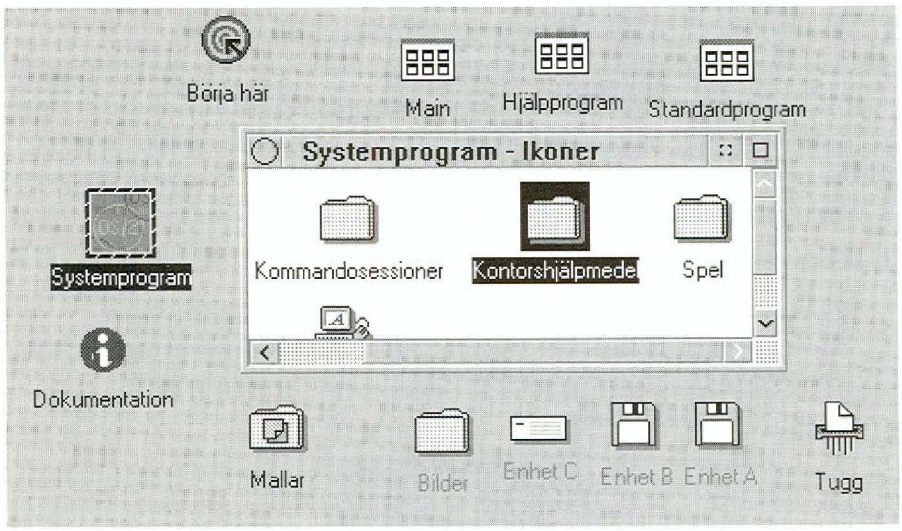

Bild 2. *Arbetsplatsen i CUA.*

CUA är ingen statisk standard, utan utvecklas hela tiden. Första definitionen av CUA kom ut 1987. Den bestod enbart av en bok som behandlade både textbaserade och grafiska gränssnitt. Nästa definition kom 1989 och bestod av två böcker. En för textbaserade och en för grafiska

gränssnitt. Den senaste definitionen kom 1991. Utvecklingen i CUA har genom åren inriktats på grafiska gränssnitt med mer direktstyrning. Dra-och-släpp-principen och pop-up-menyer kom till exempel med i den senaste utgåvan.

STANDARD-SPECIFIKATION

CUA är en standardspecifikation som inte bygger på en specifik realisering. Den förutsätter inte vilken typ av hård/mjukvara som används och därför finns det flera system som kan sägas följa CUA. Presentation Manager i OS/2 2.0 är den mest kompletta CUA realiseringen men även Motif och Microsoft Windows kan till viss mån sägas följa CUA då de har ett starkt släktskap med CUA.

TVÅ TYPER AV ANVÄNDARGRÄNSSNITT

CUA innehåller specifikationer för två typer av användargränssnitt. Den första kallas för programorienterad och är den som funnits med i CUA längst tid. Den fullständigaste realiseringen av det programorienterade CUA finner man i OS/2 1.3. I ett programorienterat CUA-gränssnitt finner man ett skrivbord, där minimerade programfönster representeras av ikoner. För att aktivera ett stängt program dubbelklickar man på ikonen. Man kan inte som användare placera ut annat än minimerade programfönster på sitt skrivbord. För att flytta data inom och mellan program används Klipp, Kopiera och Klistra-in menyalternativen som föreskrivs av CUA. Naturligtvis definieras även funktionaliteten för fönster, menybalkar och dialogboxar med sina beståndsdelar.

Den andra och mer moderna gränssnittsspecifikationen som ingår i CUA är det objektorienterade användargränssnittet som kallas för arbetsplatsen (eng workplace environment), se bild 2. I denna miljö betecknar inte ikoner minimerade program utan objekt. Det objektorienterade delen av CUA utgår ifrån den programorienterade. De har gemensamma definitioner för utseende och funktionalitet (för fönster, dialogboxar et c). Det objektorienterade gränssnittet är en utvidgning och förfining av det programorienterade användargränssnittet. Den mest kompletta av realiseringen av detta är OS/2 2.0.

Man definierar tre huvudklasser av objekt i CUA:

FÖRVARINGSOBJEKT

- *Förvaringsobjekt* är de objekt vars uppgift är att förvara andra objekt. CUA specificerar fyra typer av förvaringsobjekt: mapp, borttagsmapp (eng delete folder), arbetsytor och arbetsplatser.

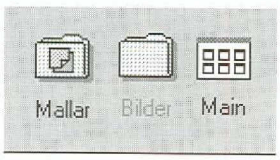

Bild 3. *Förvaringsobjekt (eng Container objects).*

DATAOBJEKT

- *Dataobjekt* är de objekt som innehåller användardata såsom text, grafer, video et c (se bild 4). CUA specificerar inga dataobjekt utan innehåller enbart riktlinjer för interaktion med dessa. Dock finns en speciell typ av dataobjekt, s k mallar, som har den speciella egenskapen "Create on drag". Create-on-drag innebär att om man tar tag i objektet och släpper det så bildas det ett nytt objekt utifrån mallen.

Bild 4. *Dataobjekt.*

UTRUSTNINGSOBJEKT

- *Utrustningsobjekt* representerar ofta ett fysiskt objekt såsom skrivare, vilket visas i bild 5. De kan även representera t ex en elektronisk brevlåda.

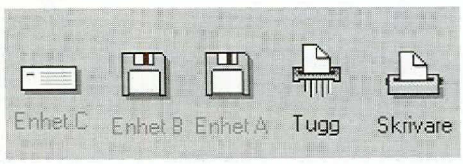

Bild 5. *Utrustningsobjekt (eng Device objects).*

Dessa tre klasser utgör grunden för användargränssnittsspecifikationen i CUA. Men vad är det för skillnad mellan de program- och objektorienterade gränssnitten? I det programorienterade talar man om program och filer. Man startar program, öppnar filer och sparar filer. I det objektorienterade gränssnittet existerar bara objekt, inga program. Man öppnar objekt istället för att starta program.

I OS/2 2.0 har man gjort en kompromiss: även program ses som en typ av dataobjekt. Detta för att enkelt kunna hantera program som endast fungerar i programorienterade miljöer. När ett objekt öppnas får man normalt upp ett fönster som beskriver en vy av objektet (se bild 6). Det kan finnas flera vyer till ett objekt, som exempel kan nämnas ett kalkylark som kan ses som siffror eller en graf.

I bild 6 illustreras två vyer av ett förvaringsobjekt; översta fönstret visar en ikonvy över innehållet i objektet medan det undre fönstret visar inställningsvyn.

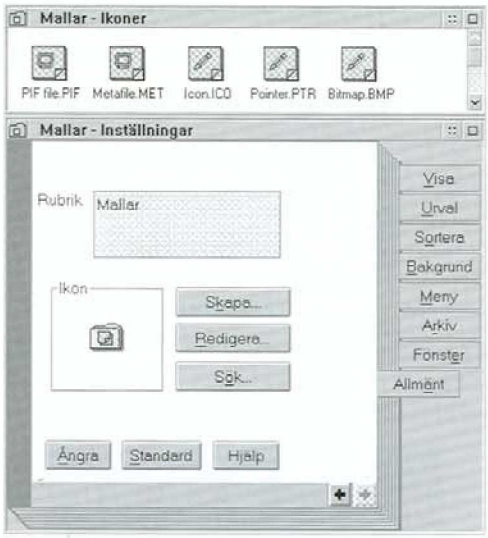

Bild 6. *Två vyer av samma objekt, ikon och inställningsvy.*

KONCENTRERAR PÅ ARBETSUPPGIFTER

Skärmytan i CUA kallas för arbetsplats (eng workplace) och är ett förvaringsobjekt. Man kan säga att arbetsplatsen är det förvaringsobjekt som innehåller alla objekt. Man definierar även mappar och arbetsytor (eng work area). Arbetsytor är speciella i den meningen att de är förvaringsobjekt där man kan lagra allt som behövs för en arbetsuppgift. En arbetsyta kan tex innehålla inkorg för fax, faxmaskin och faxar om den är avsedd för att användas för faxhantering. Idén att koncentrera på arbetsuppgifter är idag unikt för CUA. Stänger man en arbetsyta så stängs även alla fönster som är associerade med denna. Detta gör det enkelt att byta arbetsuppgift utan att få en mängd öppna fönster över från en annan uppgift.

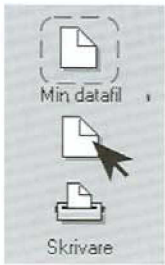

Bild 7. *Dra-och-släpp-operation i CUA.*

Interaktionen i den objektorienterade definitionen av CUA sker mycket med direktstyrning. Kopiering och flyttning av objekt sker med den så kallade dra-och-släpp-principen (bild 7). CUA specificerar standardresultat för dra-och-släpp-funktioner. Släpper man t ex ett kalkylark på skrivarsymbolen kommer det att skrivas ut. Standardvärdena baserar sig på vilken typ (förvaring, data eller utrustning) av objekt som blir påverkat.

SLÄPP ARKET PÅ SKRIVAREN

Som tidigare nämnts består CUA för grafiska användargränssnitt av två böcker. Advanced Interface Design Reference är en referensbok som talar om hur t ex rullningslister ska se ut och uppföra sig för att följa CUA. Kraven är indelade i två kategorier, sådana som måste följas och sådana som helst ska följas. För att få kalla sitt gränssnitt för ett CUA-gränssnitt räcker det med att följa de tvingande kraven. Det bör även tilläggas att CUA inte definierar ett visst utseende utan kraven gäller endast funktionaliteten. Detta innebär att olika realiseringar av CUA kan ha olika utseende men ändå följa CUA. Även de illustrationer som finns med här är endast att betrakta som exempel på hur CUA ser ut. Illustrationerna är hämtade från OS/2 2.0.

KRAV BARA PÅ FUNKTIONEN

Den andra boken Advanced Interface Design Guide beskriver hur CUA är tänkt att användas. Boken kan varmt rekommenderas då den även innehåller en introduktion till MDI (Människa-Dator-Interaktion) området. Dessutom innehåller den ett avsnitt som behandlar en fallstudie om hur man bygger ett grafiskt användargränssnitt som följer CUA. Fallstudien går igenom en metod för att skapa användargränssnittet från tecknade prototyper fram till ett fungerande system.

FALLSTUDIE

3.1 Sammanfattning av CUA

- Är en välspecificerad industristandard som ingår som en del i IBM:s System Application Architecture (SAA).
- Är specificerat oberoende av mjuk- och hårdvaruplattform.
- Specificerar två typer av användargränssnitt: program- och objektorienterat.
- Specificerar absoluta krav som måste uppfyllas för att följa CUA.
- Interaktionsobjekten är specificerade på detaljnivå vad gäller beteende. Däremot specificeras inget specifikt utseende på interaktionsobjekten.
- Innehåller en fallstudie som behandlar hur man skapar ett program med användargränssnitt som följer CUA.
- Gränssnitt från CUA används idag framförallt av program som körs på presentation manager under OS/2.

4. Motif

Motif är den gränssnittsstandard som Open Software Foundation (OSF) har valt. Open Software Foundation är en sammanslutning av Unix-leverantörer. I OSF ingår bl a IBM, HP och Digital.

Det har kommit signaler på senare tid om att CUA och Motif kommer att närma sig varandra och i det längre perspektivet förenas. Motif finns idag i stort sett bara på Unix-baserade arbetsstationer utrustade med X Windows – den har blivit den antagligen populäraste fönsterhanteraren till X Windows.

Motif har sitt ursprung i ett gemensamt förslag från Microsoft och Hewlett Packard 1988. Standarden kan ses som en korsning mellan dåvarande Presentation Manager och Hewlett Packards NewWave.

MOTIF DEFINIERAR ETT KODBIBLIOTEK

Motif är beskrivet i en mängd böcker. Då Motif även definierar ett kodbibliotek består de flesta böckerna av specifikationer av programgränssnittet till Motif. En av böckerna som specificerar Motif är en stilguide [Motif91].

Stilguiden för Motif specificerar vilka krav som ställs för att ett gränssnitt ska följa Motif-standarden. Det gäller:

- Fönster, menyer som hör till fönstret och fönsterkontroller (t ex maximeringsknapp)
- Dialogboxutformning.
- Standardmenyalternativ t ex Cut, Copy och hjälpalternativ.
- Knappar, rullningslister et c.
- Tangentbordskommandon som måste finnas med.

TVINGANDE KRAV

Stilguiden är indelad i två klasser: tvingande krav och rekommendationer. Uppfyller ett program alla de tvingande kraven får man rätten att hävda att det är ett Motif-program.

STILGUIDEN INTE OMFATTANDE

Stilguiden till Motif är inte särskilt omfattande. Det finns t ex ingen utförlig specifikation av gränssnittet och interaktionen mot filsystemet. Definitionen av gränssnittet mot filsystemet i Motif begränsar sig till hur en dialog för att öppna en fil ska se ut och uppföra sig (bild 8).

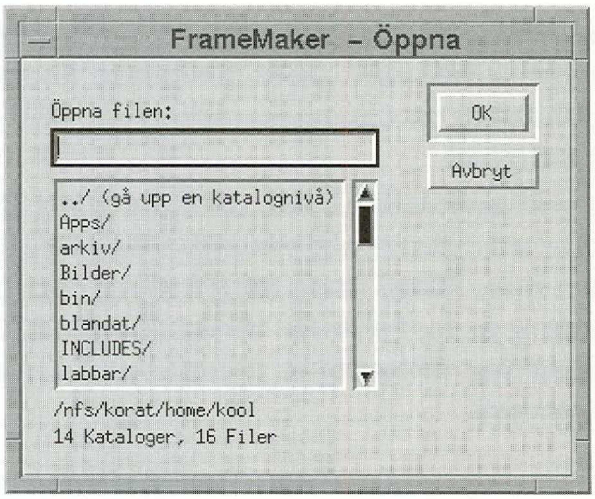

Bild 8. *Dialog för att öppna en fil i Motif.*

SPECIFICERAR BARA DET SOM REDAN FINNS I KODBIBLIOTEKET

Men den största bristen i stilguiden för Motif är att den i stort sett bara specificerar det som redan är realiserat i de kodbibliotek som finns för Motif. Den ger ingen ytterligare ledning för hur ett program bör utformas för att fungera användargränssnittsmässigt väl i en Motif-miljö.

Det är värt att lägga märke till att Motif inte kräver ett speciellt utseende på gränssnittet. Det finns t ex inga krav på hur ett fönster ska se ut med sina ramar, utan det finns bara krav på att det ska finnas fönstermaximeringsknapp et c. Man kan därför med gott fog påstå att stora delar av Motif definieras av de realiseringar som finns idag. De illustrationer som finns med här är tagna ifrån en Motif-realisering av ett fönstersystem som heter mwm.

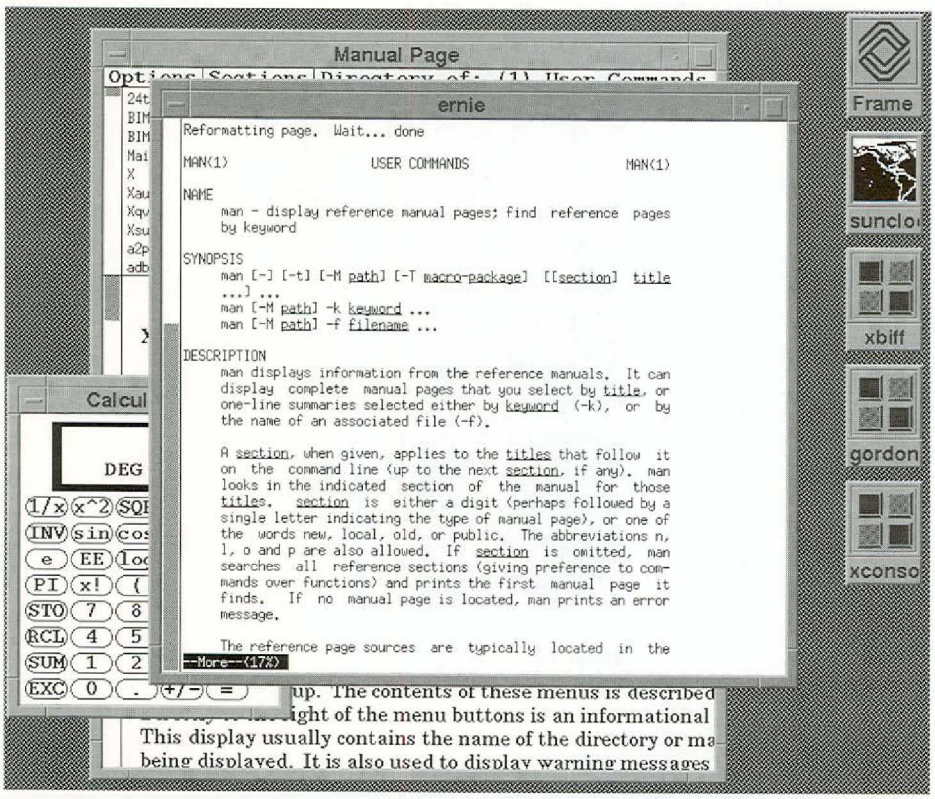

Bild 9. *Skärmbild i Motif med fönster och ikoner.*

En typisk skärmbild i Motif (se bild 9) innehåller fönster och ikoner. Ikonerna representerar stängda fönster. Motif specificerar två sätt att aktivera fönster:

- *Implicit aktivering* innebär att det fönster man har under sin markör automatiskt blir det aktiva fönstret.
- *Explicit aktivering* innebär att man måste klicka i fönstret för att göra det aktivt.

Skärmytan är inte enbart en lagringsyta för fönster och ikoner. Håller man nere en mustangent på skärmytan får man i mwm upp olika s k pop-up-menyer, som ofta innehåller möjligheter att starta program. I dagens realiseringar av Motif är dessa pop-up-menyer vanliga och det är förhållandevis enkelt för användaren att lägga till och ta bort menyalternativ. Däremot är det något förvånande att det inte nämns något om dessa pop-up-menyer i stilguiden.

POP-UP-MENYER NÄMNS INTE I STILGUIDEN

Motif har idag ett utpräglat tredimensionellt utseende med utstående knappar och menyalternativ (se bild 10).

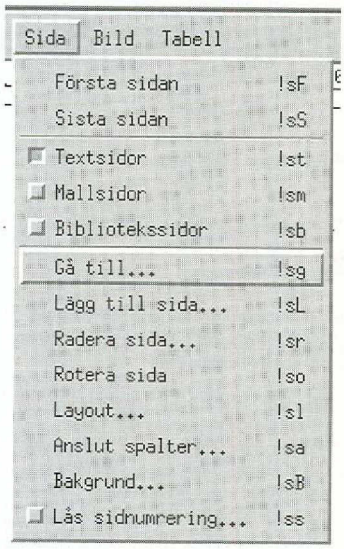

Bild 10. *Meny i Motif.*

4.1 Sammanfattning av Motif

- Motif är en industristandard specificerad av Open Software Foundation (OSF).
- Interaktionsobjekten specificeras enbart till sin funktionalitet. Motif specificerar inte utseendet.
- Specificerar absoluta krav som måste uppfyllas om man vill hävda att ett program följer Motif.
- Gränssnitt med Motif används framför allt av program som använder sig av X Windows.

5. MS Windows

Windows är idag den i särklass mest spridda gränssnittsstandarden. Den är specificerad av Microsoft, har sina rötter i CUA men börjar nu sakta att utvecklas till en egen standard. Windows har också fått en egen stilguide [MIC92] som definierar användargränssnittet. Windows finns idag enbart på s k IBM-kompatibla persondatorer.

Stilguiden definierar dialogboxar, interaktionsobjekt och kortkommandon. En del dialogboxar, som öppna fil och spara fil, definieras fullt ut i stilguiden. För övrigt är det rekommendationer för utseende och funktionalitet som förekommer. Något förvånande för en så pass ny stilguide som Windows är att den inte definierar dra-och-släpp utförligt utan enbart har en allmän diskussion om detta.

DEFINIERAR INTE DRA-OCH-SLÄPP UTFÖRLIGT

I och med att Windows innehåller avancerad funktionalitet för kommunikation mellan program som är tänkt att utnyttjas av användare finns det även rekommendationer för hur dessa ska realiseras i användargränssnittet. Windows använder sig av två tekniker för att realisera detta; objektlänkar (OLE) och datalänkar (DDE). Det är till exempel möjligt att ha en länk till ett kalkylark i ett textdokument - varje gång kalkylarket uppdateras så görs även uppdateringen i textdokumentet.

Windows finns i en speciell penndatorversion och därför innehåller stilguiden också rekommendationer för penndatorprogram. Det som definieras för penndatorprogrammen är de speciella interaktionsobjekten som används på penndatorer, t ex textinmatningsfält av en speciell typ. Gester definieras också för de vanligaste redigeringsfunktionerna klipp, klistra, radera o s v.

REKOMMENDATIONER FÖR PENNDATOR-PROGRAM

Windows stilguide innehåller inte några rekommendationer för hur egendefinierade interaktionsobjekt ska te sig. Stilguiden definierar enbart interaktionsobjekt som finns med i realiseringen av Windows.

Trots allt följer Windows fortfarande i huvudsak CUA av den programorienterade varianten. En punkt där Windows idag skiljer sig från CUA är kortkommandona för klippa och klistra som nu i Windows är ctrl-x och ctrl-v. Windows innehåller alla de interaktionsobjekt som finns definierade för den programorienterade delen av CUA. Dessutom nämner man i stilguiden att den ska vara kompatibel med CUA89.

FÖLJER I HUVUDSAK CUA

Vår genomgång av Windows grundar sig på realiseringen av Windows 3.1. De bilder som finns med är även dessa hämtade ifrån Windows 3.1.

BARA PÅ PC

Windows finns idag enbart för persondatorer som använder sig utav operativsystemet MS-DOS och som bygger på Intels 80 X86 processorarkitektur. Windows är med andra ord hårt knuten till en viss typ av hårdvaruplattform. Man har flaggat för att Windows i framtiden även ska flyttas till andra hårdvaruplattformar i och med introduktionen av Windows NT.

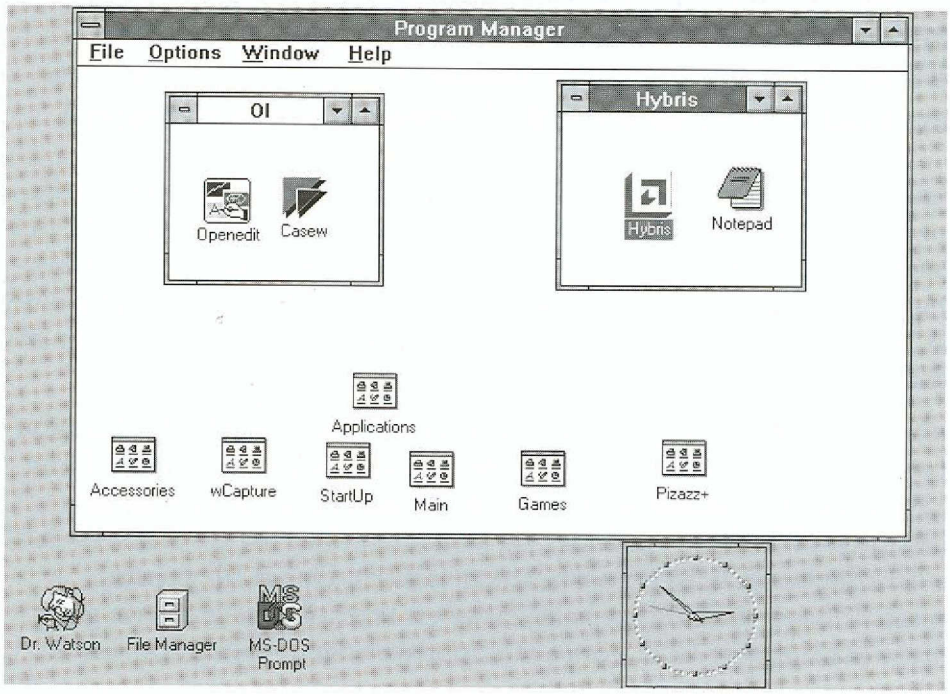

Bild 11. *Windows skärmutseende.*

Windows består av ett skrivbord (se bild 11) som fyller hela skärmen. På skrivbordet placeras fönster och ikoner. Ikonerna representerar program som är minimerade.

Till filsystemet finns ett speciellt gränssnitt som kallas för filhanteraren (bild 12).

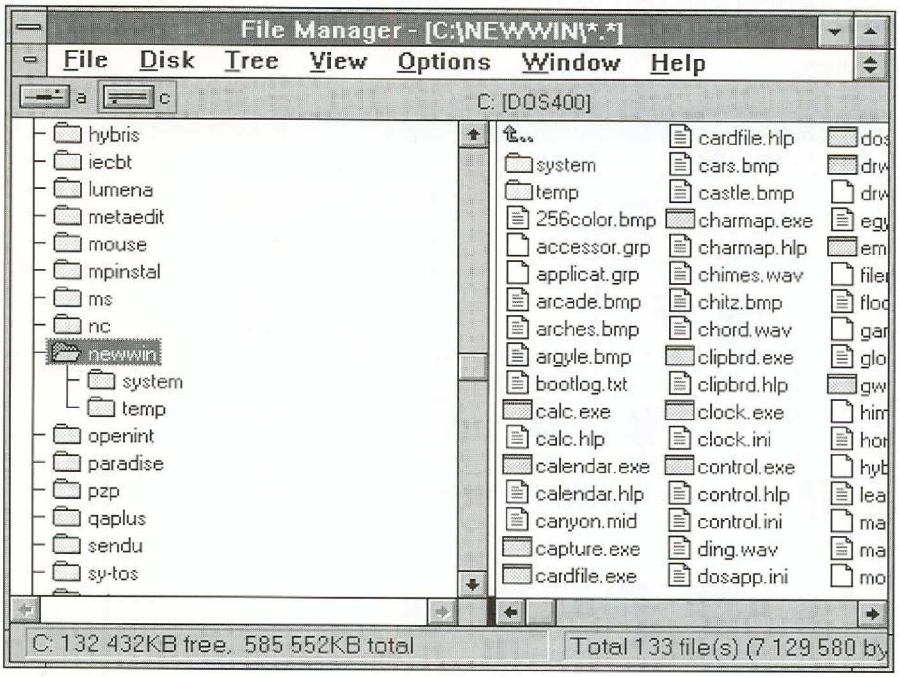

Bild 12. *Filhanteraren i Windows.*

Filhanteraren understödjer "dra-och-släpp". För att flytta en fil mellan mappar tar man tag i filsymbolen och släpper den på den mapp man vill flytta den till. Användaren kan även starta tillämpningar via filhanteraren genom att dubbelklicka på programmet. Dessutom är det möjligt att koppla program till en viss filtyp, d v s definiera att alla filer som slutar på ".txt" ska öppnas med programmet Anteckningar. Då är det möjligt att i filhanteraren öppna filer som slutar med .txt med programmet Anteckningar genom att dubbelklicka på dem.

KOPPLA PROGRAM TILL EN VISS FILTYP

För att hantera program finns även ett speciellt gränssnitt som kallas Programhanteraren (se bild 13). I Programhanteraren representeras program med hjälp av ikoner. För att starta ett program dubbelklickar man på ikonen.

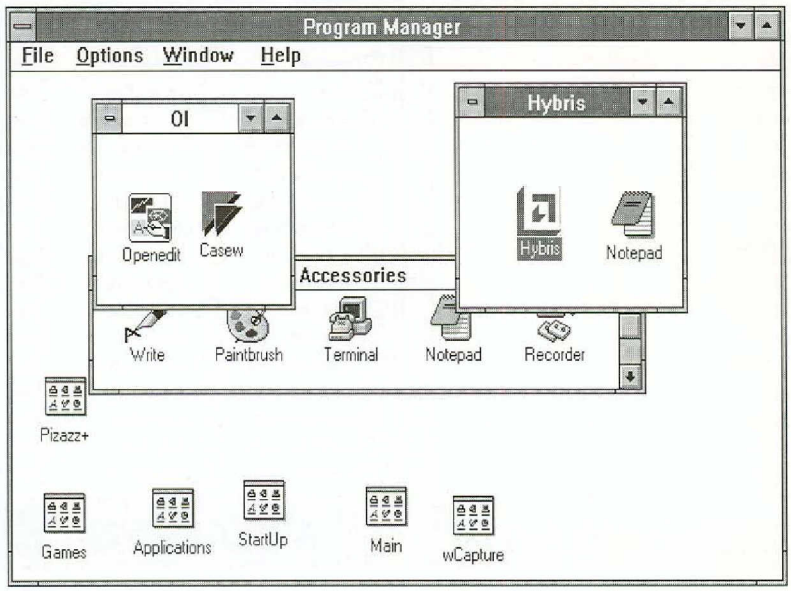

Bild 13. *Programhanteraren i Windows.*

KRÄVER INGET PEKDON

Windows användargränssnitt bygger på menyer och fönster. Men det kräver inte att man använder sig av något pekdon. Man kan klara sig enbart med tangentbord då det finns tangentbordskommandon som motsvarar det man kan utföra med ett pekdon.

Menyerna till ett program sitter alltid i anslutning till programmets huvudfönster (se bild 14). Om ett program kan öppna flera dokument på en gång så kommer de fönster som innehåller dokument alltid att befinna sig inom programmets huvudfönster (bild 14).

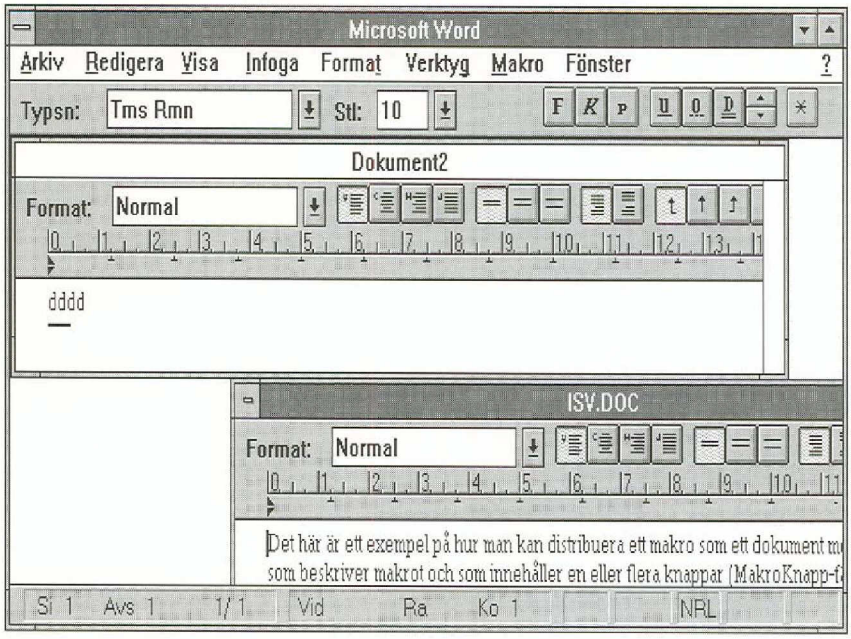

Bild 14. *Programfönster i Windows.*

5.1 Sammanfattning av MS Windows

- Windows är en industristandard från Microsoft.
- Windows har en stilguide som även täcker in penndatorversionen.
- Stilguiden innehåller ingen hjälp för att skapa egna interaktionsobjekt.
- Stilguiden är kompatibel med CUA av den programorienterade versionen CUA89.
- Windows finns idag enbart på s k IBM-kompatibla persondatorer.

6. Open Look

Open Look är en gränssnittsstandard som har utvecklats av Sun Microsystems tillsammans med AT&T. Den har specificerats fristående från någon existerande mjuk- eller hårdvara, men kräver grafisk terminal, tangentbord och mus. Interaktionsobjektens utseende och beteende är specificerade på detaljnivå.

Open Look stöds av Unix International, en sammanslutning av företag med Sun och AT&T i förgrunden. Open Look-gränssnittet används i Unix Systems Laboratories Unix-variant System V Release 4.2 såväl som i Suns operativsystem Solaris. Standarden baseras på arbete som ursprungligen utförts av Xerox [Johns89].

VÄLSPECIFICERAD

Open Look är välspecificerad och beskrivs ingående i två böcker:

- Graphical User Interface Functional Specification [Sun89].
- Graphical User Interface Application Style Guidelines [Sun90].

GUI Functional Specification är främst av intresse för den som själv ska realisera interaktionsobjekten i Open Look. Vanligtvis använder man dock ett kodbibliotek där interaktionsobjekten redan finns färdiga i form av kod som kan anropas från det egna programmet.

GUI Application Style Guidelines är mer användbar för utveckling av egna program med hjälp av kodbibliotek. Den beskriver i detalj hur och vilka knappar och menyer som ska användas och hur interaktionen med olika typer av dialogboxar och funktioner ska gå till. Man ger också noggranna anvisningar om hur fönster bör se ut och hur interaktionsobjekten bör placeras. Boken innehåller många bra tumregler för god gränssnittsdesign.

TUMREGLER FÖR GRÄNSSNITTSDESIGN

Skärmytan kallas enligt Open Look för en arbetsyta (workspace) och där används en enkel form av skrivbordsmetafor med dokument, verktyg och papperskorg. På arbetsytan finns sedan ett antal verktyg och dokument, representerade av fönster och ikoner. Bild 15 visar en typisk arbetsyta enligt Open Look.

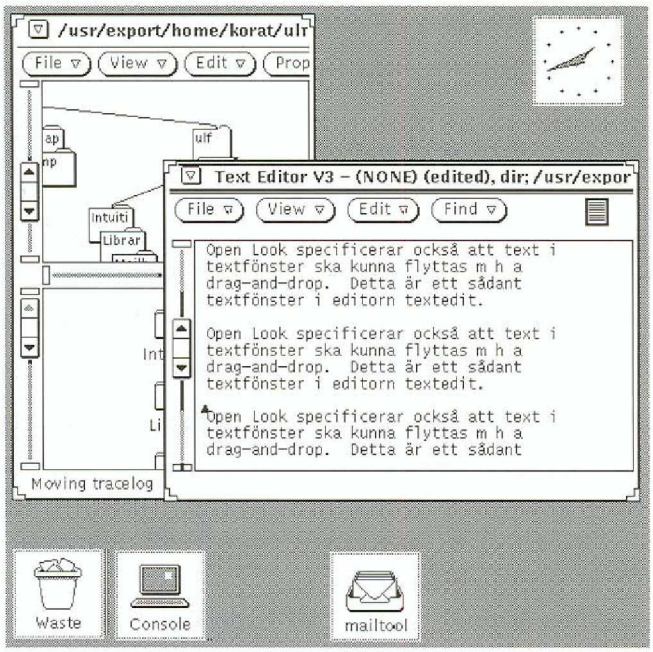

Bild 15. *En typisk arbetsyta enligt Open Look.*

TRE-KNAPPARS MUS I PRAKTIKEN

I Open Look förutsätts att ett pekdon av något slag, företrädesvis en mus men också andra typer som t ex en penna eller rullboll, kan användas. Även om det inte enligt specifikationen ska spela någon roll ifall musen är försedd med en, två eller tre knappar så är en tre-knappars mus i praktiken det självklara valet av pekdon. Om man tänker använda en mus med endast en knapp, tvingas man utnyttja tangentbordet för att modifiera betydelsen av en knapptryckning.

GRAFISK FILHANTERARE

Förutom att definiera ett antal interaktionsobjekt beskriver Open Look också hur interaktionen med filsystemet ska gå till med hjälp av en grafisk filhanterare, File Manager, som alltså är en del av Open Look-specifikationen. I File Manager kan man med direkt manipulation av grafiska objekt, ikoner, starta program, kopiera, flytta och ta bort filer m m. Den teknik som används är "dra-och-släpp".

För att starta ett program med "dra-och-släpp" kan man från File Manager dra ikonen för ett dokument och släppa det på bakgrunden. Då startas programmet med dokumentet som indata. Vilket program som väljs beror på vad som är standard för den aktuella dokumenttypen.

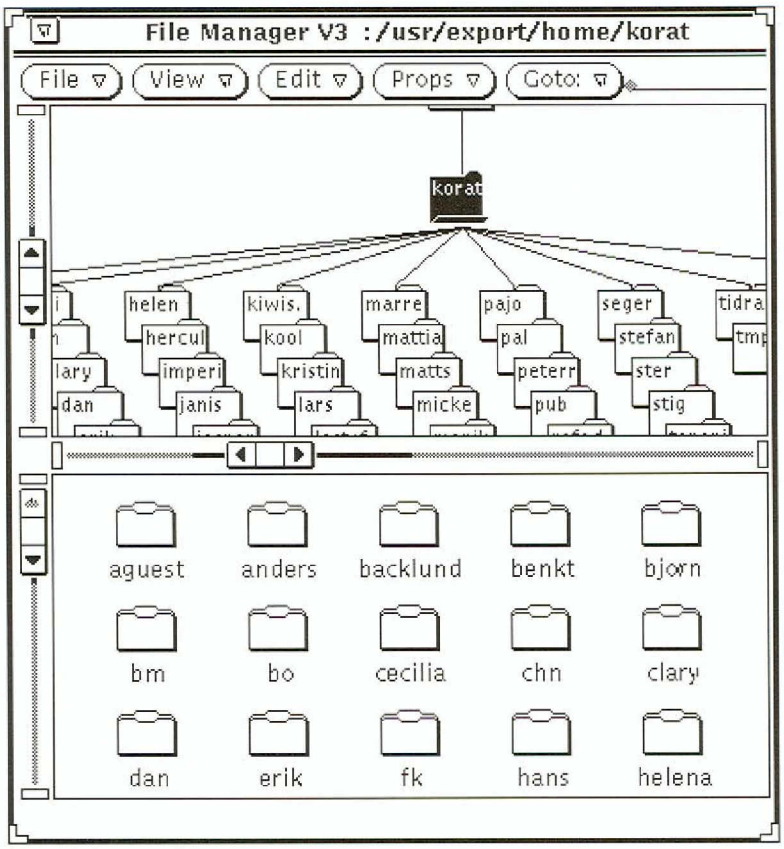

Bild 16. *Filhanteraren File Manager*

Open Look specificerar också att text i ett textfönster ska kunna flyttas med hjälp av dra-och-släpp-principen, d v s texten markeras och kan sedan dras till en ny position, eventuellt i ett annat fönster, med hjälp av musen.

Open Look innehåller en del speciella interaktionsobjekt som inte fungerar på samma sätt som i andra standarder, även om funktionerna i sig finns specificerade även där. Några exempel är:

NÅLAR

- Nålar, eller "Push-Pins" som används för att fästa dialogboxar och menyer på arbetsytan. Om man t ex alltid vill att en viss meny ska vara framme på skärmen kan den nålas fast. En fastsatt och en lös nål visas i bild 17.

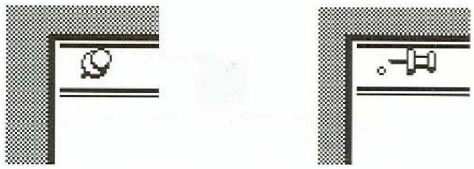

Bild 17. *Fastsatt respektive lös nål.*

LEXIVISUELLA MEDDELANDEBOXAR

- Meddelandeboxar som genom att använda lexivisuell presentationsteknik tydligt visar från vilket program eller programdel som meddelandet kommer (bild 18). Lexivisuell teknik utnyttjar en kombination av text och grafik för att tydligt åskådliggöra sammanhanget. Denna egenskap är speciellt bra när flera program körs samtidigt på skärmen.

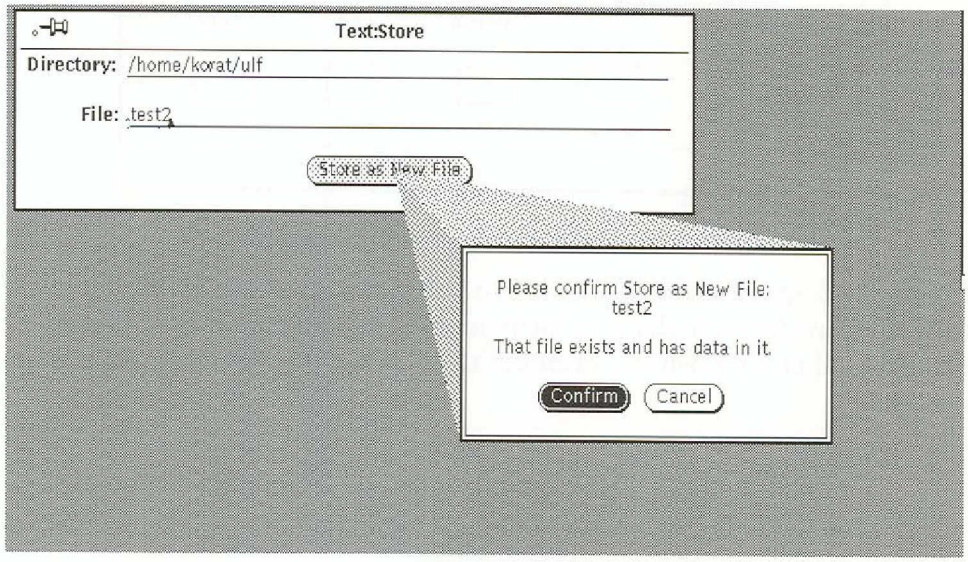

Bild 18. *Lexivisuellt presenterad meddelandebox.*

- Markören kan automatiskt hoppa och placeras över den knapp i t ex en meddelandebox, som oftast ska tryckas (default button). När meddelandeboxen försvinner återgår markören till sin gamla position. Den hoppande markören är en fiffig funktion som underlättar arbetet, speciellt när

en stor skärm används. Hoppande markör kan också användas för andra fönstertyper, t ex hjälpfönster och kommandofönster.

HOPPANDE MARKÖR

- Hjälpfönster som kan aktiveras genom att markören placeras över det man vill ha hjälp om och en hjälptangent trycks ned. I bild 19 visas ett hjälpfönster om nålar efter det att användaren har placerat markören över en nål och hjälpfunktionen aktiverats från tangentbordet med hjälptangenten.

HJÄLPFÖNSTER

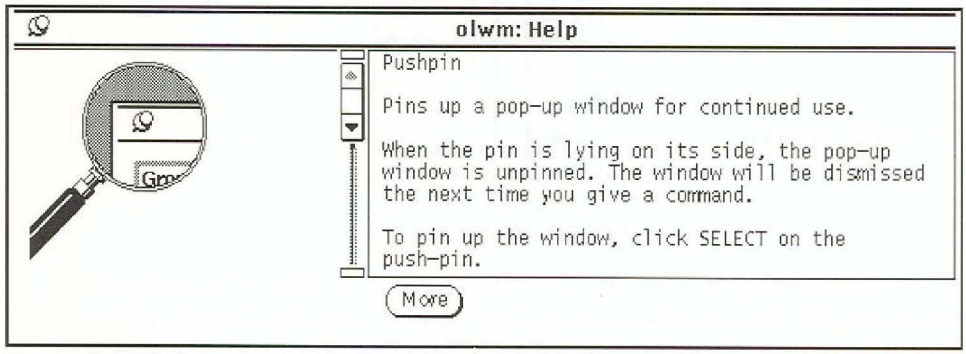

Bild 19. Hjälpfönster.

- Rullninglister som tillhör de mer sofistikerade på marknaden (bild 20). Rullningslisten kan liknas vid en hisskorg på en kabel. Man rullar t ex en text genom att klicka med markören på pilen för endera upp- eller nerfärd. Då rullas texten i motsvarande riktning och hisskorgen förflyttas under tiden för att med sin position visa var i texten man befinner sig. Markören behåller dock sin placering på den valda pilen så länge rullningen pågår, d v s den rör sig med hisskorgen. Det är också möjligt att med ett musklick förflytta sig till textens början eller slut, samt att hoppa sidvis i texten genom att klicka på kabeln, under eller över hisskorgen. En svärtad del av kabeln ger information om hur stor del av texten som visas i fönstret.

SOFISTIKERADE RULLNINGSLISTER

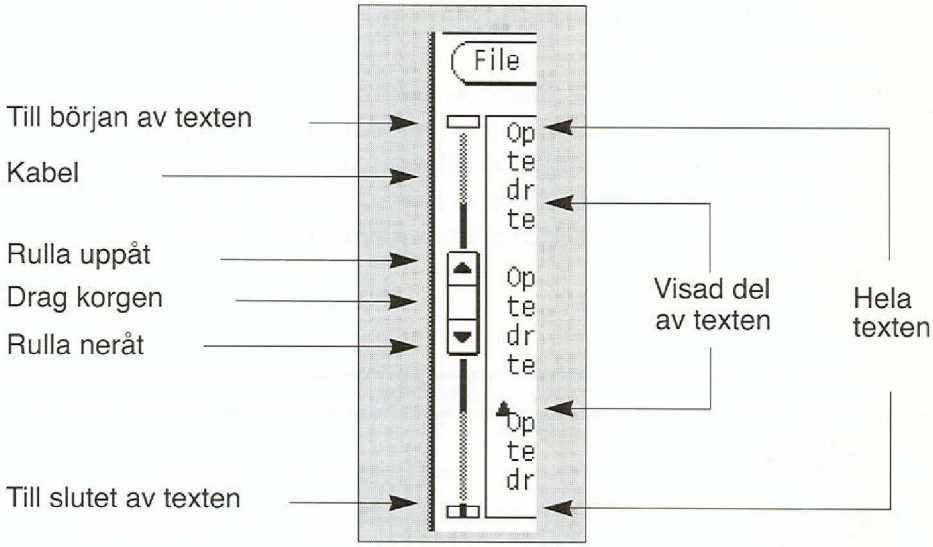

Bild 20. *Rullningslist.*

HANDBOK EN DEL AV ANVÄNDAR-GRÄNSSNITTET

I specifikationen ges konkreta anvisningar om vilken terminologi som bör användas i handböcker och andra skrifter som beskriver gränssnitt enligt Open Look. Det är naturligtvis lika viktigt med en enhetlig beskrivning av gränssnittet i en handbok som vid dess presentation på skärmen. Detta är ett bra exempel på synsättet att också beskrivningar och handböcker utgör en del av användargränssnittet.

AT&T GER TILLSTÅND I TRE NIVÅER

Rätten att hävda att ett program följer Open Look är knuten till en prövning av AT&T. De tillstånd som utfärdas delas in i tre nivåer:

- Nivå 1 kräver ett komplett gränssnitt med alla väsentliga funktioner som t ex klipp & klistra, menyer, knappar, enkla rullningslister, ikoner et c.
- För nivå 2 ställs högre krav och kräver att man förutom vad som krävs för nivå 1 har exempelvis stöd för färg, ett flertal kortkommandon, mer avancerade rullningslister, möjlighet att kopiera eller flytta text genom att dra den o s v. De flesta Open Look-program kommer så småningom att klara kraven för nivå 2 anser man från Unix Internationals sida.
- Nivå 3 är ej specificerad ännu men är tänkt att användas för mer speciella interaktionsmöjligheter som senare kan behöva föras in i Open Look-standarden.

Den mest kända tillämpningen av Open Look idag är den X Windows-baserade fönsterhanteraren Open Windows som kommer som standard på datorer från Sun. Bilderna i detta kapitel kommer alla från Open Windows. Den senaste versionen av Open Windows realiserar stora delar av Open Look nivå 1 och 2.

Open Look har inte fått något genombrott utanför Sun och AT&T utan används huvudsakligen på datorer från Sun samt på s k kloner av sådana datorer från andra tillverkare, t ex Solbourne.

I och med att Sun planerar att släppa Unix-operativsystemet Solaris också för PC-datorer med Intel-processorer, kan Open Look komma att spridas i en större omfattning.

SOLARIS KAN SPRIDA OPEN LOOK

6.1 Sammanfattning av Open Look

- Open Look är en välspecificerad industristandard från Unix International. Den har specificerats fristående från mjuk- och hårdvaruplattformar.
- Interaktionsobjektens utseende och beteende är specificerade på detaljnivå. Till viss del behandlas också principer för hur interaktionsobjekten ska kombineras i programmens användargränssnitt. Ett antal översiktliga principer för gränssnittsdesign ges också.
- Bland interaktionsobjekten i Open Look hittar man alla vanliga typer, även om flera har fått delvis unika egenskaper. Rullningslister i form av hissar eller uppnålningsbara menyer och dialogrutor är några exempel på former av interaktionsobjekt som inte finns i någon annan industristandard.
- Gränssnitt från Open Look används framförallt av program som körs på operativsystemet Solaris från Sun Microsystem.

7. Macintosh

Den första egentliga standard för grafiska gränssnitt som utgivits var Apples Human Interface Guidelines [Apple87]. Den beskriver användargränssnittet på Macintosh och definierar de olika interaktionsobjektens beteende. Standarden finns bara spridd på Apple-datorer, men kan väl påstås vara anfadern till de övriga stilguiderna.

ENBART REKOMMENDATIONER

Stilguiden för Macintosh innehåller enbart rekommendationer om hur programvara på Macintosh ska se ut och uppföra sig. Det finns inga formella krav som måste uppfyllas för att man ska få hävda att ett program följer stilguiden för Macintosh. I stilguiden specificeras utseende och funktionalitet hos knappar menybalkar, textfält et c. Även gränssnittet mot filsystemet specificeras i stilguiden. Stilguiden förutsätter dessutom att man använder någon form av pekdon t ex mus.

FÖRUTSÄTTER PEKDON

LJUD SKA ÄVEN SYNAS

Stilguiden innehåller ett avsnitt om hur program anpassas för handikappade. Som exempel kan nämnas att alla ljudåterkopplingar även ska ha en visuell återkoppling så att även människor med nedsatt hörsel ska kunna utnyttja programmet.

Men stilguiden som sådan är inte heltäckande utan mycket får anses definierat genom realiseringen av Macintosh användargränssnitt.

I själva verket är det så att om man utnyttjar Macintosh kodbibliotek vid realiseringen av ett program så kommer man att uppfylla det mesta av riktlinjerna i stilguiden.

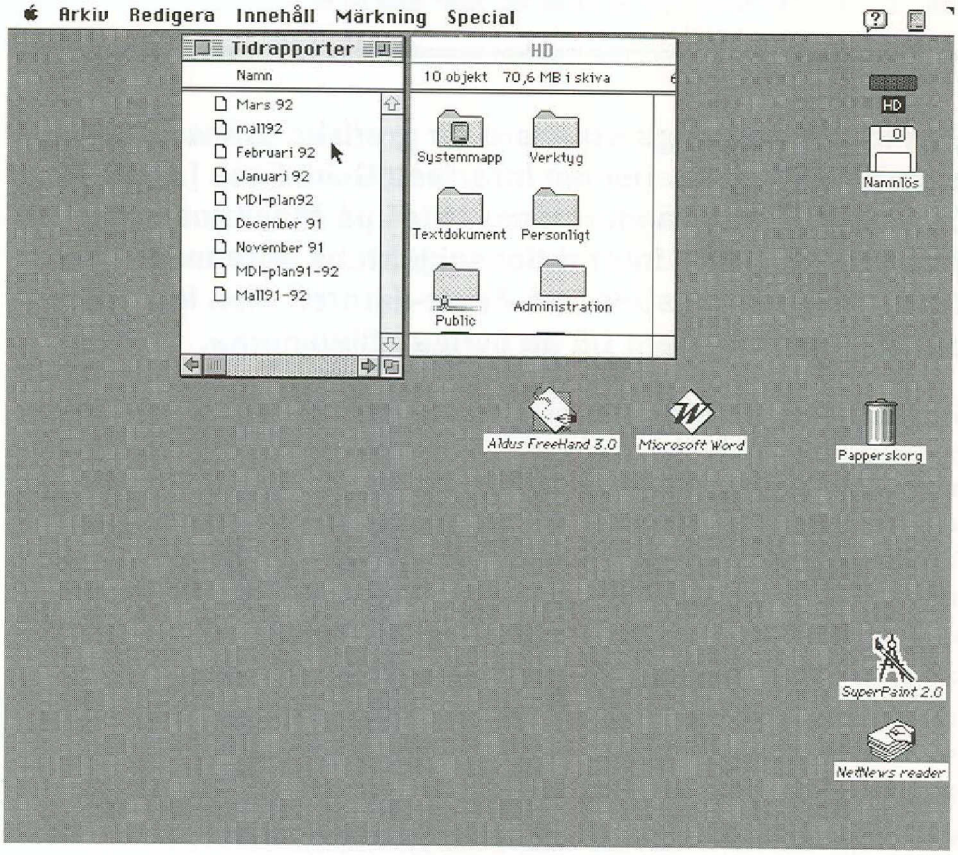

Bild 21. *Finder i Macintosh.*

Filsystemet som kallas för Finder bygger på en skrivbordsmetafor med dokument, mappar och papperskorg (se bild 21). Filsystemet bygger på direktstyrning, dra-och-släpp utnyttjas till att flytta dokument och för att starta program (man släpper dokumentet på programmet). Man öppnar dokument och mappar genom att dubbelklicka på dem.

INKONSEKVENSER

Tyvärr är inte dra-och-släpp till fullo realiserat, om man vill skriva ut ett dokument så markerar man först genom att klicka på symbolen och sedan väljer man menykommandot "Skriv ut". Det hade varit naturligare att ha en skrivarsymbol på skrivbordet som man kunde släppa dokumentet på. Filsystemet är inte helt konsekvent realiserat. Att dra ett dokument mellan två mappar kan ha två betydelser. Om mapparna ligger på samma enhet (d v s disk) så flyttar man dokumentet, men om de ligger på olika enheter så kopieras dokumentet. En annan inkonsekvens är att man matar ut disketter genom att slänga diskettsymbolen i papperskorgen. En operation som annars betyder att man kastar bort dokument.

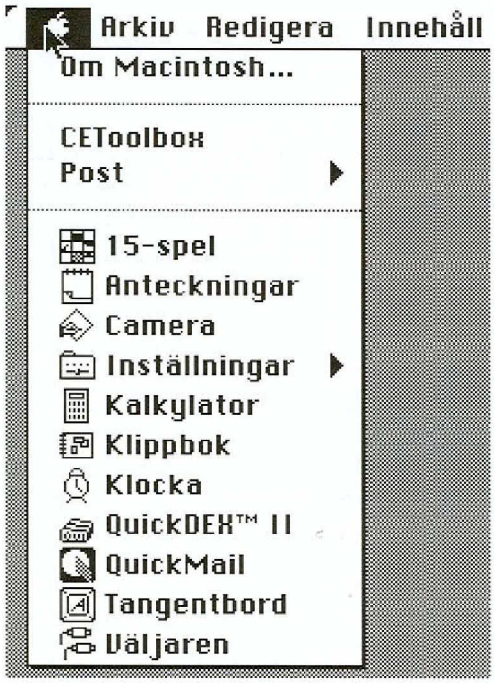

Bild 22. *Apple-menyn i Macintosh.*

Till skillnad från de flesta andra standarder har Macintosh programmenyn på samma ställe för alla tillämpningar. Menyn som kallas menybalk sitter alltid placerad högst upp på skärmen. På en liten skärm har detta ingen betydelse, men på en stor skärm får man förflytta musen - långa sträckor för att komma till menyn med markören.

Man har också definierat något som kallas för Applemenyn (bild 22). Applemenyn är en meny som alltid uppträder längst till vänster på menybalken. Applemenyn finns alltid att tillgå oavsett vilket program man använder (om de följer stilguiden). I Applemenyn finns de program som man alltid vill ha tillgång till som t ex klippboken. Applemenyns innehåll bestäms av användaren.

ANVÄNDAREN BESTÄMMER INNEHÅLL

En unik egenskap hos Macintosh och dess stilguide är den form av kontextuell hjälp som finns att tillgå, den s k ballonghjälpen. Ballonghjälpen fungerar genom att det kommer upp ballonger med förklarande text till det objekt man har under sin pekare. Ett exempel finns i bild 23.

HJÄLP-BALLONG

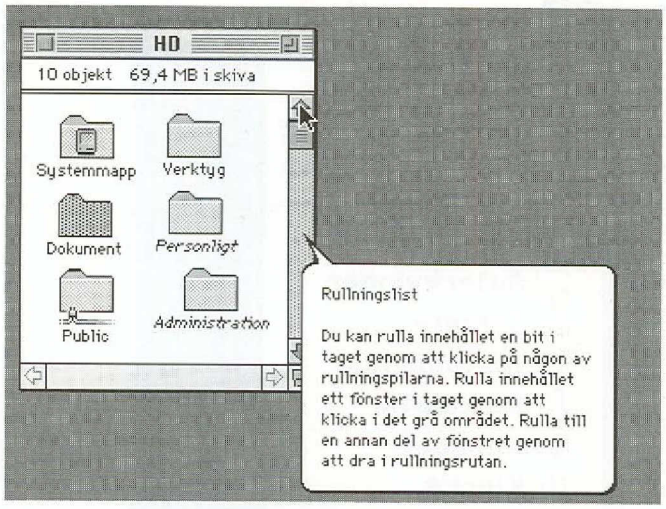

Bild 23. *Ballonghjälp*

FINNS ENBART PÅ MACINTOSH

I dag finns Apple Human Interface Guidelines enbart realiserat på Macintosh-datorer. Det verkar inte heller finnas några planer på att göra realiseringar på andra plattformar.

7.1 Sammanfattning av Macintosh

- Macintosh är industristandard skapad av Apple.
- Stilguiden definierar de olika interaktionsobjektens beteenden.
- Standarden definieras till stor del av sin realisering.
- Stilguiden innehåller ett avsnitt om anpassning av program för handikappade.
- Gränssnittet används idag enbart på Macintosh-datorer.

8. Nextstep

Nextstep är namnet på den de facto standard för grafiska gränssnitt som (än så länge) används bara på Unix-datorer av märket Next. Next är en uppstickare på marknaden för Unix-arbetsstationer med främsta säljargument att Next-datorn, tack vare gränssnittet Nextstep, är en "användarvänlig Unix-arbetsstation". Standarden betonar direktstyrning.

Nextstep har ett karakteristiskt utseende i huvudsakligen svart, vitt och grått med tydlig 3D-känsla. Nextstep påminner annars i hög grad om övriga fönsterbaserade gränssnitt med alla de vanliga interaktionsobjekten som rullningslister, knappar och menyer.

TYDLIG 3D-KÄNSLA

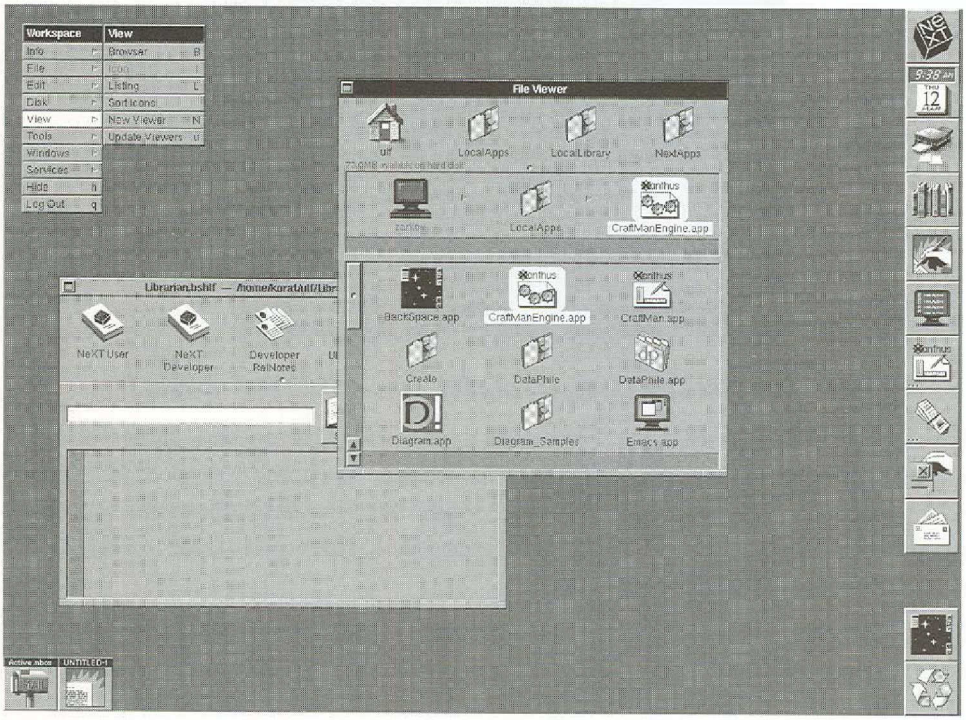

Bild 24. Typiskt Nextstep-gränssnitt med menyer i övre vänstra hörnet, aktiva programikoner i nedre vänstra hörnet och favoritprogrammen längs högra kanten i den s k Application Dock.

Nextstep, med sitt ursprung i en Unix-miljö, är anpassat för att utnyttja en multitasking-miljö, d v s att flera program kan köras samtidigt. Modala dialogboxar är t ex endast modala för det program de tillhör. Andra program blockeras ej utan kan användas som vanligt även om den modala dialogboxen visas.

FLERA PROGRAM SAMTIDIGT

Flera program kan alltså köras samtidigt men endast menyn för det för tillfället aktiva programmet visas på skärmen. För att byta till ett annat program kan man klicka antingen på programmets ikon eller i något fönster på skärmen som tillhör det önskade programmet.

Menyerna placeras inte tillsammans med de programfönster de hör till, utan finns alltid på en bestämd plats på skärmen. Undermenyerna i hierarkiska menyer kan plockas loss från "modermenyn" och därmed visas permanent på skärmen.

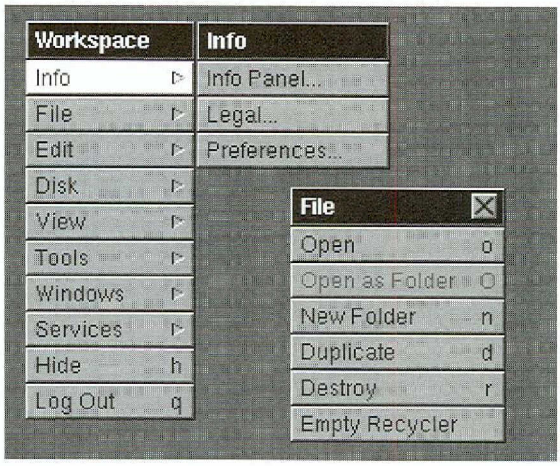

Bild 25. *Nextstep-menyer. Undermenyn File är fristående.*

En viktig designprincip för Nextstep är att användaren ska styra vad som händer på skärmen. T ex påpekas vikten av att fönster som har en viss placering på skärmen innan det stängs behåller sin placering när det öppnas på nytt, detta för att användaren ska uppleva kontroll över utseendet på skärmens arbetsyta.

HELST DIREKTSTYRNING

Mus, inte tangentbord, är det primära verktyget för att styra ett program enligt Nextstep. Nextstep-specifikationen argumenterar starkt för användningen av direktstyrning där så är lämpligt. Ett bra exempel är - Nexts egen realisering av ett program för elektronisk post där man med dra-och-säpp kan bifoga filer i ett elektroniskt brev genom att dra filens ikoner från filhanteraren och släppa dem på fönstret med brevet. Däremot kan inte program startas med dra-och-släpp, varken genom att dra och släppa ett program eller ett dokument.

Nextstep innehåller ett grafiskt gränssnitt mot filsystemet i filhanteraren File Viewer. I filhanteraren kan man med hjälp av musen flytta och kopiera filer samt starta program genom att dubbelklicka på ikoner för filer eller program. Filhanteraren kan visa filsystemet på tre olika sätt, som ikoner, i en hierarkisk bläddrare (eng browser) eller som en lista liknande den som fås av Unix-kommandot "ls-l".

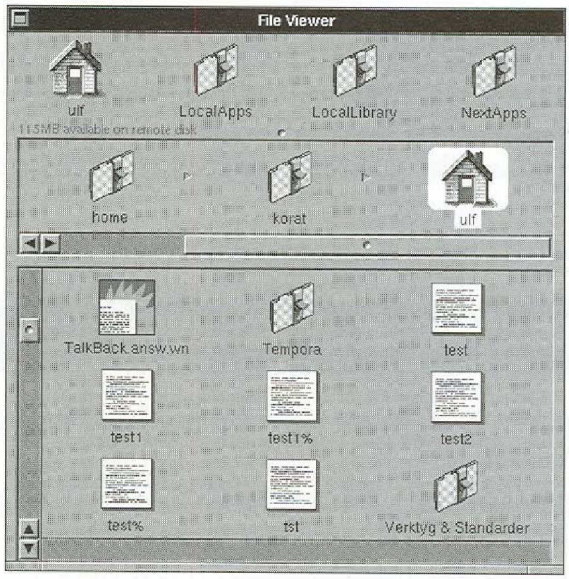

Bild 26. *Nextsteps grafiska filhanterare File Viewer med filer som ikoner.*

För att enklare kunna hitta och starta sina favoritprogram, kan ikoner för dessa flyttas ut och läggas direkt på skärmens bakgrundsyta i den s k Application Dock.

AKTIVA MINIFÖNSTER

Fönster kan krympas till s k minifönster som påminner om ikoner men med den skillnaden att de fortfarande är aktiva. För att göra fönstret normalstort igen, dubbelklickar man på minifönstret.

Nextstep beskrivs i de handböcker som medföljer vid köp av en Next-dator, t ex [Next 90]. Dessutom finns en fylligare beskrivning i den dokumentation som finns on-line på Next-datorerna. Dessa beskrivningar ger en god bild av vilka interaktionsobjekt som finns tillgängliga samt hur de bör se ut och hur de beter sig

DIREKTSTYRNING DEL AV IMAGEN

I specifikationen beskrivs Nextsteps interaktionsobjekt utifrån programmerarens synvinkel i syfte att realisera en tillämpning på en Next-dator. Program med moderna gränssnitt som intensivt utnyttjar direktstyrning är en del av Nexts image och ett klart mål med specifikationen är därför att få utvecklare att skriva program som väl ansluter sig till den "Look-and-feel" som Nexts egna program har.

Specifikationen innehåller beskrivningar av interaktionsobjektens utseende och beteende. Några exempel på speciella interaktionsobjekt är

Color Well (ung färgkällor) där färg kan hämtas för att färglägga objekt i t ex ritprogram. Ett annat interaktionsobjekt, Browser, är speciellt avpassat för att navigera i hierarkiska strukturer, t ex ett filsystem.

Olika interaktionsformer med mus beskrivs också. Man skiljer där på händelser som enkel-, dubbel- och trippelklick, dragning av objekt, kontinuerlig tryckning o s v.

STÖRRE SPRIDNING I SIKTE

Nextstep-specifikation baseras helt på realiseringen på Next-datorn och dess kodbibliotek Appkit. Nextstep finns för tillfället bara realiserat på Next-datorer och har därför en mycket begränsad spridning. En version för PC-datorer med Intels 80486-processorer har dock demonstrerats och kommer enligt Next att finnas för försäljning under början av 1993. Detta ger naturligtvis helt nya möjligheter till ökad användning av Nextstep.

8.1 Sammanfattning av Nextstep

- Nextstep från Next liknar i mycket konkurerande gränssnittsstandarder.
- Nextstep poängterar vikten av att användaren styr programmen.
- Nextstep utnyttjar direktstyrning i stor utsträckning.
- Nextstep används idag bara på datorer från Next men kommer från och med våren 1993 (enligt Next) också att kunna användas på PC-datorer med Intel 80486-processorer.

9. Officiella standardiserings-förslag

Syftet med det officiella standardiseringsarbetet är bl a att det ska vara lättare att byta mellan olika program och datormiljöer. Ett speciellt syfte med strävan mot en officiell standard, är att underlätta för kunder att skriva tydliga kravspecifikationer för användargränssnitt.

I ISO 9241 har man försökt att formulera standarden oberoende av dagsaktuella tekniker och i stället fokuserat på användningseffektiviteten.

Sedan mitten av 80-talet har det pågått standardiseringsverksamhet som omfattar den logiska uppbyggnaden av användargränssnitt, s k ergonomisk standard för mjukvara. Den bedrivs framförallt inom den internationella standardiseringsorganisationen ISO men också inom flera nationella standardiseringsorganisationer. I Sverige bedrivs inget eget standardiseringsarbete inom området, men svenska representanter deltar i ISO-arbetet.

ERGONOMISK STANDARD

Standardiseringsarbete är normalt en långdragen process som naturligtvis sätts på hårda prov inom ett område som utvecklas i den snabba takt som användargränssnitten gör. En standard måste därför formuleras oberoende av aktuell teknisk nivå, samtidigt som den måste vara konkret nog för att kunna användas praktiskt vid exempelvis upphandling.

KONKRET OCH OBEROENDE

Det för användargränssnitt mest intressanta standardiseringsarbetet pågår inom ISO 9241, *Ergonomic requirements for office work with visual display terminals*, som bl a redovisas i [ISO91a, ISO91b, ISO92a, ISO92b]. För att i möjligaste mån undvika "problemet" med den snabba tekniska utvecklingen har man, till skillnad från t ex olika standarder för datorutrustning, fokuserat på att specificera faktorer som påverkar användningseffektiviteten, hellre än på fysiska krav för utrustning och program.

ISO 9241

ISO 9241 är indelad i 17 olika delar som behandlar olika aspekter av dialogergonomin. Delarna 1-9 handlar huvudsakligen om krav på olika

typer av utrustning som t ex tangentbord och skärmar, men också om de uppgifter som ska lösas av användaren samt om dennes miljö. Delarna 10-17 handlar om krav på programmen som användaren utnyttjar. Orsaken till att man diskuterar användaren, användarens miljö och kunskap, är att de krav som kan ställas på ergonomi till stor del beror av och varierar med dessa faktorer.

Del 10 är alltså den första delen av ISO 9241 som direkt behandlar *programmens* ergonomi. Där definieras ett antal dialogprinciper som antas kunna vara oberoende av teknikskiften. Dessa dialogprinciper är viktiga för att konstruera och utvärdera en dialog:

LÄMPLIG FÖR UPPGIFTEN

- Dialogen stöder användaren i att effektivt slutföra en uppgift.

SJÄLVBESKRIVANDE

- Varje dialogsteg kan omedelbart förstås av användaren eller eventuellt förklaras på begäran av denne.

KONTROLLERBAR

- Användaren har kontrollen över dialogen hela vägen fram till att uppgiften är löst.

ANVÄNDARNAS FÖRVÄNTNINGAR

- Dialogen överensstämmer med användarens kunskap, utbildning och tidigare erfarenheter.

FELTOLERANT

- Om felaktigheter uppstått, ska användaren kunna slutföra sin uppgift utan stora korrektioner.

INDIVIDUELL ANPASSNING

- Dialogen ska kunna anpassas efter användarens individuella behov.
- Lämplig för inlärning.

För att genomföra själva dialogen med användaren, kan ett flertal olika *dialogtekniker* användas. Dialogen kan baseras på:

- Menyer
- Kommandospråk
- Direktstyrning
- Formulär
- Fråga-svar
- Naturligt språk

Gränserna är naturligtvis flytande mellan flera av dialogteknikerna, som dessutom kan överlappa varandra. Till exempel kan kommandospråk och fråga-svarsdialoger ofta överlappa.

SPECIFIKA STANDARDER

Från de allmänna principerna i delarna tio och elva angående dialog och användbarhet av program, formuleras sedan mer specifika standarder. De specifika standarderna har som syfte att kunna ge konkret stöd i konstruktion och utvärdering av olika aspekter av programergonomin.

Tyvärr har man inom ISO inte hunnit särskilt långt med arbetet på dessa standarder. Våren 1992 fanns endast förslag till del 14, Menydialoger. Där ges konkreta anvisningar om hur olika menydialoger ska byggas

upp, samt förslag till hur sådana dialoger kan utvärderas. För att förtydliga standarden ges ett otal exempel som alla utvärderas mot dialogprinciperna ovan.

Det standardarbete som pågår är ambitiöst och verkar kunna leda till användbara standarder. De har inte samma inriktning som de kommersiella, informella, standardförslagen. Av naturliga skäl ligger de officiella standardförslagen på en högre abstraktionsnivå, men uttrycks ändå både kompakt och konkret på ett sätt som kan göra dem omedelbart tillämpbara. De har dock en brist i att de bara behandlar en dialogteknik i taget när dagens gränssnitt mer och mer blir en blandning av flera.

9.1 Sammanfattning av ISO 9241

- ISO 9241 är ett pågående standardiseringsarbete för programmens ergonomi inom den internationella standardiseringsorganisationen ISO.
- I ISO 9241 har man försökt att formulera standarden oberoende av dagsaktuella tekniker och i stället fokuserat på användningseffektiviteten. Ett antal dialogprinciper som ska vara oberoende av teknikskiften definieras därför i standardförslaget.
- ISO 9241 finns idag endast färdigt som utkast för några få delar av den kompletta standarden, mycket arbete återstår innan allt är klart. Stora delar av utkasten är dock av intresse i befintligt skick.

10. Slutsatser och diskussion

Skillnaden mellan de olika industristandardernas innehåll och användarvänlighet är bara ytlig. Även om vissa leverantörer gärna hävdar att det egna förslaget till standard är det enda som ger användarvänliga program, så kan vi här konstatera att i stort sett lika bra, eller för den delen lika dåliga, gränssnitt kan åstadkommas enligt alla industristandarder.

På sikt kommer ett företag att kunna bestämma sig för en gränssnittsstandard utan att vara låst till en datortillverkare. Det kommer att vara möjligt att blanda datormärken men ändå behålla ett enhetligt gränssnitt.

UTVECKLAREN STYR ANVÄNDBARHETEN

Om ett program blir användarvänligt eller ej, beror helt på utvecklarens förmåga att utveckla sådana program. Programmens ergonomiska kvalitet blir endast marginellt bättre av att följa en industristandard eftersom de inte tar hänsyn till faktorer som användarens tidigare kunskaper eller arbetsuppgiftens utformning. Industristandarderna ger i och för sig gott stöd för utveckling av enhetliga program, val av interaktionsobjekt o s v, men sådant måste ändå betraktas som små problem i sammanhanget.

STANDARDER LÖSER BARA SMÅ PROBLEM

En undersökning bland ett antal utvecklare som speciellt arbetar med gränssnittsproblem, bekräftar den begränsade betydelsen av specifikationerna för industristandarderna i systemutvecklingsprojekt [Niels92]. Av de studerade utvecklingsprojekten hade dylika stilguider endast använts i en fjärdedel av fallen och dessutom haft liten påverkan på slutresultatet.

Orsaken till den begränsade nyttan av stilguider är att de till stor del specificerar sådant som redan är realiserat i ett antal olika kodbibliotek som oftast används vid utveckling av program. Om man beslutat att använda ett kodbibliotek som stöder Open Look, t ex OLIT (Open Look Intrinsics Toolkit), så finns alla interaktionsobjekt redan färdiga där. Utvecklingsarbetet handlar sedan mer om att sätta samman fördefinierade interaktionsobjekt än att utveckla nya.

Stöd i standarder för val av metaforer och användningsmodell är alltså mycket begränsat, även om det i riktlinjerna för t ex CUA och Open Look ges en del förslag till allmänna tumregler för gränssnittskonstruk-

tion. Naturligtvis är det svårt att i en allmänt hållen standard ge riktlinjer för t ex metaforval eftersom detta val till stor del är beroende av programmets funktion och syfte samt användarnas bakgrundskunskap och kompetens. Problemet kvarstår dock och den officiella standarden ISO 9241 har här en plats att fylla med sin ansats att standardisera på användbarhet snarare än "knappars och fönsters utseende".

ANVÄNDBARHET ELLER BARA UTSEENDE

Industristandarderna diskuterar inte heller avancerad domänspecifik interaktion som t ex sökning i databaser eller navigering i bildbibliotek. Det ringa bidraget från marknadens aktörer inom det här området kompenseras dock av pågående arbete inom det europeiska forskningssamarbetet inom Esprit. Bland andra deltar SISU där i två projekt, Intuitive och Kiwis, som arbetar med gränssnittsprinciper för informationssökning i databaser [Intui92, Verme91].

DATABAS-SÖKNING

10.1 Jämförelse

CUA BÄST SPECIFICERAD

CUA och i viss mån Open Look framstår som de bäst specificerade standarderna, både vad gäller innehåll och detaljeringsgrad. De har högre ambitionsnivå än sina konkurrenter genom att de förutom att ange interaktionsobjektens utseende och beteende, också försöker slå ett slag för de större problemens lösningar. Speciellt innehåller CUA-specifikationen många goda råd om gränssnittsdesign som alla utvecklare har nytta av ta del av. Det går utmärkt att ta till sig många av de goda råden från CUA, framförallt om man tänker sig att realisera sitt program i MS Windows eller Motif, men även i t ex Macintosh-miljö.

CUA OBEROENDE AV REALISERINGSMILJÖ

En av de tänkbara orsakerna till att specifikationerna av CUA och Open Look är kvalitativt bättre, är att de är helt fristående från något kodbibliotek eller datormiljö (se tabell 1). Även om CUA gradvis har växt fram ur IBM:s gränssnittslösningar, är den i och med den senaste versionen (CUA91) en standard som är oberoende av realiseringsmiljö. Macintosh och Nextstep har däremot helt specificerats utifrån sina respektive realiseringar på Macintosh- och Next-datorer. (Notera att detta inte betyder att *realiseringar* av Macintosh och Nextstep-standarderna är sämre än realiseringar av CUA och Open Look. Här diskuterar vi enbart specifkationerna, där det finns en klar skillnad.)

Tabell 1: Sammanställning av industristandarder.

	CUA	Motif	MS Windows	Macintosh	Open Look	Nextstep
Leverantör	IBM	OSF	Microsoft	Apple	Unix International	Next
Realiserat för mer än en datortyp	Ja	Ja	Nej	Nej	Nej[a]	Nej[b]
Specificerad oberoende av realisering i kodbibliotek	Ja	Nej	Nej	Nej	Ja	Nej

a. En version som ingår i operativsystemet Solaris 2.X kommer till våren -93 för Intel 80486-maskiner (PC), enligt Sun. En betaversion har demonstrerats hösten 1992.
b. En version som ingår i operativsystemet Nextstep 3.0 kommer till våren -93 för Intel 80486-maskiner (PC), enligt Next. En betaversion har demonstrerats hösten 1992.

MS Windows och Motif har stora likheter med CUA. Men likheten mellan MS Windows och CUA gäller framförallt CUA89, vad som kommer hända med MS Windows framgent är osäkert. Microsoft har inte, till skillnad från OSF/Motif, deklarerat att de tänker följa CUA91. Den troliga utvecklingen är att Microsoft väljer en egen väg vid sidan om CUA, eftersom de nu kommit ut med en egen stilguide. I den aktuella realiseringen av Windows (3.1) är det trots allt möjligt att bygga gränssnitt som till stor del ansluter sig till CUA91.

MICROSOFT GÅR EGEN VÄG?

I tabell 2 jämförs innehållet i stilguiderna för de olika industristandarderna på ett antal punkter.

De flesta standarderna innehåller ett *direktstyrt gränssnitt mot filsystemet* med möjlighet att flytta mappar, öppna dokument och starta program enbart med hjälp av direktstyrning. Många standarder har också möjligheten att *starta program med dra-och-släpp* genom att släppa ett dokument på programsymbolen.

STARTA PROGRAM MED DIREKTSTYRNING

Dra-och-släpp är en princip för interaktion som funnits länge men som först på senare tid definierats i olika industristandarder. Den gamla versionen av CUA, CUA89, definierar till exempel inte dra-och-släpp medan CUA91 har den som en viktig komponent. Dra-och-släpp är mest utnyttjad i Open Look och CUA, men finns också delvis i Nextstep (ej starta program) och Macintosh.

Alla standarder definierar rullningslister som indikerar var i exempelvis ett dokument man befinner sig. Vissa ger också *information om hur stor del* av hela dokumentet som visas, vilket är bra.CUA slår fast att det ska vara möjligt att göra alla operationer från tangentbordet vilket innebär att det inte i teorin måste finnas något *pekdon*. Det kan vara användbart

OLIKA INFORMATION I RULLNINGSLISTERNA

i vissa fall, men i realiteten är det svårt att tänka sig ett program med gränssnitt enligt CUA som saknar mus eller annat pekdon.

PAPPERSKORG ELLER ÅTERVINNING

Papperskorgen har blivit stilbildande som metafor för att ta bort filer. Nyare standarder har dock försökt att komma på alternativ. I CUA används en papperstugg och i Nextstep en ikon med en symbol för återvinning! (Den första versionen av Nextstep hade metaforen att filerna kastades i ett svart hål. Det fungerade inte riktigt bra, t ex var det svårt att förklara att det svarta hålet kunde öppnas och filen hämtas tillbaka ur dess grepp.)

Tabell 2: Stilguidernas innehåll.

	CUA	Motif	MS Windows	Macintosh	Open Look	Nextstep
Filhantering med direktstyrning	Ja	Nej	Ja	Ja	Ja	Ja
Starta program med dra-och-släpp	Ja	Nej	Nej	Ja	Ja	Nej
Rullningslisten indikerar andelen visad yta	Ja	Ja	Nej	Nej	Ja	Ja
Kräver pekdon	Nej	Nej	Nej	Ja	Ja	Ja
Sätt att ta bort filer	Borttagsmapp (CUA) Papperstugg (OS/2)	-	-	Papperskorg	Papperskorg	Återvinningssymbol (Recycler)
Pop-up-menyer	Ja	Nej	Ja	Nej	Ja	Ja
Placering av systemmenyer	Pop-up	-	Fönster	Menybalk	Pop-up	Övre vänstra hörnet
Placering av programmenyer	I programfönster	I programfönster	I programfönster	Menybalk och programfönster	I programfönster	Övre vänstra hörnet och i programfönster
Stöd för utomeuropeiska språk	Ja	Nej	Nej	Nej	Nej	Nej
Hjälpfunktion	Ja	Ja	Ja	Ja	Ja	Ja

Systemmenyerna för de olika fönstersystem som realiserar standarderna placeras vanligast på skärmens bakgrundsyta i form av pop-up-menyer. Fördelen med pop-up-menyer är att de är lätta att få fram, medan de har nackdelen att sakna kontinuerlig representation på skärmen.

Placeringen av tillämpningsprogrammens menyer är löst enligt två olika ansatser. I CUA, Motif, Windows och Open Look finns menyerna i de fönster som tillhör programmen medan Nextstep och Macintosh har menyerna fast placerade på en plats på skärmen, i en menybalk i skärmens överkant för Macintosh respektive valfri fix plats på skärmen för - Nextstep. I en miljö där flera program kan köras samtidigt är det enligt vår uppfattning en fördel att menyerna placeras i anslutning till programmens fönster, det ger nämligen en varaktig representation av menyerna på skärmen. I annat fall, som på Macintosh och Nextstep, växlar menyerna efter det program som för tillfället är aktivt.

PROGRAMMENYERNA ALLTID PÅ SKÄRMEN

Användningen av stora bildskärmar ökar kontinuerligt vilket innebär ytterligare fördel för ansatsen att placera menyerna i respektive fönster, man slipper då flytta markören onödigt långt för att välja ett menyalternativ. Open Looks hoppande markör och fönsterhissar där markören följer hisskorgen effektiviserar ytterligare användningen av program på stor skärm. Stora skärmar medför vidare att det är bra om pilar på rullningslister sitter nära varandra, allt för att reducera flyttningssträckan för markören.

MARKÖRPROBLEM MED STORA SKÄRMAR

Endast CUA innehåller riktlinjer för hur gränssnitt med *utomeuropeiska språk* behandlas. Bland annat förklaras hur språk som läses från höger till vänster, t ex arabiska, ska påverka gränssnittets utformning. Alla stilguider ger riktlinjer för hur funktioner för *direkthjälp* ska utformas.

CUA KLARAR UDDA SPRÅK

Open Look är den stilguide som innehåller flest unika interaktionsobjekt som inte går att finna i andra. Några exemmpel är nålar och hisskorg. Det finns också exempel på unika operationer. I CUA finns t ex Create-on-drag som inte har någon motsvarighet i andra stilguider.

OPEN LOOK HAR FLEST EGNA OBJEKT

Att stänga ett fönster har olika betydelse i olika standarder vilket kan vara förvirrande för en användare som arbetar i flera olika miljöer. I exempelvis Open Look, Nextstep och CUA kan ett fönster göras till en ikon (ev minifönster) och fortfarande vara aktivt, t ex behöver ändringar inte sparas. På Macintosh innebär motsvarande handling att fönstret helt deaktiveras, d v s om eventuella ändringar ska sparas så måste detta göras innan fönstret stängs.

Olikheterna vad gäller betydelsen av att krympa eller stänga ett fönster till ikon beror delvis av att en del standarder konstruerats för att kunna hantera datormiljöer där flera program kan användas samtidigt (s k multitasking). Exempel på sådana standarder är Open Look och Motif som båda har sitt ursprung i Unix-världen där multitasking är standard och har funnits länge. I takt med att operativsystem i persondatorvärlden får

ANVÄNDA FLERA PROGRAM SAMTIDIGT

multitasking-kapacitet kommer förmodligen gränssnitten där att behöva förändras.

I tabell 3 sammanfattas några av stilguidernas viktigaste egenskaper. Med *detaljeringsgrad* avses hur noggrant gränssnittet specificeras. Där har Windows och Nextstep relativt ytliga stilguider.

CUA HJÄLPER PROGRAMMERAREN

Vissa standarder är mer till stöd för programutvecklaren än andra, de är mer *tillämpningsinriktade* och ger riktlinjer för hur ett program ska realiseras enligt standarden. Ett bra exempel är CUA där stilguiden *innehåller ett fullständigt genomarbetat exempel* som visar hur ett program kan utvecklas. Stilguiderna för Motif, MS Windows, Macintosh och Nextstep ger dock väldigt lite utöver vad programmeraren får genom att använda något kodbibliotek som realiserar standarderna.

EGET KODBIBLIOTEK

För den som planerar att *utveckla ett eget kodbibliotek* som stöder en viss standard ger CUA, Motif och Open Look mest stöd för det. I Open Look specificeras i detalj hur till exempel en meny-knapp ska se ut på skärmen.

Tabell 3: Stilguidernas egenskaper.

	CUA	Motif	MS Windows	Macintosh	Open Look	Nextstep
Detaljeringsgrad	Hög	Medel	Medel	Medel	Hög	Medel
Tillämpnings-inriktade	Ja	Nej	Ja	Nej	Ja	Ja
Stöd för utveckling av kodbibliotek	Ja	Ja	Nej	Nej	Ja	Nej
Ordlista för översättning mellan olika språk	Ja	Nej	Ja	Nej	Nej	Nej
Fullständigt genomarbetat exempel	Ja	Nej	Nej	Nej	Nej	Nej
Generella riktlinjer för gränssnittsdesign	Ja	Nej	Nej	Ja	Nej	Nej
Kräver specifikt utseende	Nej	Nej	Ja	Ja	Ja	Ja
Kräver specifikt beteende	Ja	Ja	Ja	Ja	Ja	Ja

Standarderna varierar mycket i *vad som krävs för att följa dem.* I Open Look och Macintosh ställs strikta krav på såväl utseende som beteende, medan CUA och Motif nöjer sig med krav på beteende.

10.2 Framtiden

I den populära datapressen, både i Sverige och utomlands, har man länge debatterat "Gränssnittskriget", d v s spekulerat i vilket av de på marknaden förekommande gränssnitten som kommer att vinna kampen om användarna. Orsaken till intresset för kampen är att det ligger mer än gränssnittspengar i potten för datortillverkarna. Många datortillverkare har försökt och försöker fortfarande använda gränssnittet som en viktig särskiljande egenskap till förmån för datorer av det egna märket, men en klar trend är att den hårda bindningen mellan datorfabrikat och gränssnittsstandard kommer att försvinna.

BINDNINGEN MELLAN DATOR OCH GRÄNSSNITT FÖRSVINNER

Processen har redan startat i och med bildandet av OSF där bl a IBM, Digital och HP enats om Motif som standard för sina Unix-arbetsstationer. Den har fortsatt med att Sun och Next sprider sina respektive gränssnitt, Open Look och Nextstep, till PC. Ett annat tydligt tecken på denna trend är samarbetet mellan IBM och Apple om PowerPC.

Resultatet på sikt kommer troligen bli att ett företag kan bestämma sig för en gränssnittsstandard utan att vara låst till en datortillverkare. Det kommer att vara möjligt att *blanda datormärken men ändå behålla ett enhetligt gränssnitt.*

Datorer med nya former för interaktion har redan introducerats eller kommer att göra det inom en snar framtid. Till exempel kan vi ha en penna och en bärbar dator där man skriver som på ett A4-block, eller en penna och en väggskärm där man skriver som på en emaljskrivtavla.

Användningen av en penna som interaktionshjälpmedel är ett mycket hett område just nu, se till exempel [Goldb91, Kurte91], där man speciellt undersöker hur man kan ge kommandon med olika gester. Till exempel kan man ta bort ett ord i en text genom att kryssa över ordet med pennan.

Röststyrning är också en kommande interaktionsform, programmen kan där ta emot talade kommandon.

När flera olika inmatningsformer blandas brukar man tala om multimodalitet, d v s kommunikation i flera moder samtidigt, t ex tal och pekning. SISU arbetar med multimodal inmatning i Esprit-projektet Intuitive [Intui92] som syftar till att utveckla verktyg för informationssökning i stora multimediadatabaser. Gränssnittet i Intuitives verktyg ska kunna hantera talade frågor som t ex "Vad heter den där personen" om användaren samtidigt som han uttalar frågan pekar på en bild av en person.

Andra intressanta interaktionsformer som studeras i forskningslabben är t ex konstgjorda världar (eng Virtual Reality) där man med hjälp av specialutrustning (t ex datahandske och hjälm) kan arbeta med programmet i en tredimensionell konstgjord värld. Specialutrustningen ger användaren möjlighet att "krypa in" i datorns konstgjorda värld och där manipulera de objekt som finns där som om de vore verkliga.

INGA NYA FORMER FÖR INTERAKTION I STANDARDERNA

För alla dessa nya interaktionsformer saknas huvudsakligen riktlinjer i alla existerande standarder, undantaget är Windows som innehåller avsnitt om pennstyrda system. Utvecklingen mot nya interaktionsformer kommer därför att innebära ständiga omarbetningar av dagens standarder. Man bör därför vara medveten om att standarder inte är några statiska företeelser utan förändras i takt med den snabba utvecklingen.

11. Referenser

[IBM89] Systems Application Architecture Common User Access Basic Interface Design Guide (BIDG), IBM 1989, SC26-4583

[IBM91] Systems Application Architecture Common User Access Guide to User Interface Design, IBM 1991, SC34-4289

[IBM92] Systems Application Architecture Common User Access Advanced Interface Design Reference, IBM 1991, SC34-4290

[ISO91a] First Commitee Draft, ISO 9241, del 10, *Dialogue Principles*, 1991.

[ISO91b] Draft International Standard, ISO 9241, del 14, *Menu Dialogues*, 1991.

[ISO92a] International Standard, ISO 9241, del 1, *Introduction*, 1992.

[ISO92b] Second Commitee Draft, ISO 9241, del 11, *Usability Principles*, 1992.

[MIC91] Microsoft, *The GUI guide, localizing the graphical user interface,* Microsoft Press 1991, ISBN 1-55615-426-7

[MIC92] Microsoft, *The Windows Interface, An Application Design Guide,* Microsoft Press 1992, ISBN 1-55615-384-8

[SISU92] P Bergsten, M Bern, P Kool, U Wingstedt, *Verktyg för grafiska gränssnitt*, SISU Rapport nr 20, november 1992.

[Apple87] Apple, *Human Interface Guidelines*, 1987.

[Goldb91] D Goldberg, A Goodisman, *Stylus Interaface for Manipulating Text*, Proceedings of the ACM Symposium on User Interface Software and Technology sid. 127-135, november 1991.

[Intui92] The Intuitive Project Consortium, *Technical Annex*, 1992

[Johns89] J Johnsson et al, *The Xerox Star: A Retroperspective,* IEEE Computer sid. 11-26, september 1989.

[Kurte91] G Kurtenbach, W Buxton, *Issues in Combining Marking and Direct Manipulation Techniques*, Proceedings of the ACM Symposium on User Interface Software and Technology sid. 137-144, november 1991.

[Motif91] OSF/Motif Style Guide, Prentice Hall 1991, ISBN 0-13-640616-5

[Next90] Next Computer, *The NeXT User Interface,* Nextstep Concepts kapitel 2, 1990.

[Niels92] J Nielsen, *The Usability Engineering Life Cycle,* IEEE Computer, sid 12-22, mars 1992.

[Shnei83] B Shneiderman, *Direct Manipulation: A Step Beyond Programming Languages*, Computer v 16 sid. 57-69, augusti 1983.

[Sun89] Sun Microsystem, *Open Look, Graphical User Interface Functional Specification*, Addison-Wesley, 1989.

[Verme91] D Vermeir et al, *The Kiwis Knowledge Base Management System*, ur CAiSE91 Proceedings, Trondheim, Springer Verlag, 1991.

SNABBFAKTA OM SVENSKA INSTITUTET FÖR SYSTEMUTVECKLING

Svenska Institutet för Systemutveckling är en forskningsorganisation som arbetar för att nya rön inom informationsteknologi-forskningen ska bli praktiskt användbara och komma till konkret nytta inom näringslivet.

Instiftat av regeringen 1984.
1991 omsatte institutet, som finansieras av svenskt näringsliv och NUTEK, ca 30 Mkr.
De fyrtio anställda i Stockholm och Göteborg leds av professor Thomas Falk.
Institutet är indelat i fyra forskningsområden:

- Verksamhetsutveckling
- Informationssystem
- Datorstödd utveckling
- Människa-datorinteraktion

Kunskaper överförs genom rapporter, utredningar, kurser, seminarier, kompetensnät och, framför allt, genom medlemmarnas deltagande i projekt.
Den långsiktiga forskningsinriktningen läggs fast i treåriga ramprogram, i samverkan med medlemmarna.

SISU

STOCKHOLM
Box 1250, 164 28 Kista. Electrum, Isafjordsgatan 26, Kista.
Telefon: 08-752 16 00. Telefax: 08-752 68 00.

GÖTEBORG
Box 14225, 400 20 Göteborg. Mölndalsvägen 17, Göteborg.
Telefon: 031-83 02 50. Telefax: 031-83 10 47.