SISU

PUBLIKATION 96:20

RAPPORT – OKTOBER 1996

Metoder för att hitta användbarhetsproblem hos datorsystem

Per Fossum

SVENSKA INSTITUTET FÖR SYSTEMUTVECKLING

SISU

Förord

Rapporten är ett delresultat av det större SISU-projektet Användbar IT, vars syfte är att minska den klyfta som finns mellan MDI-forskning (Människa-Dator-Interaktion) och tillämpningen av dess resultat i Sverige. Inom projektets ram har en enkätundersökning genomförts för att få en bild av hur företagen i Sverige arbetar med användbarhetsfrågor och hur de ser på MDI-forskning. Enkäten besvarades av drygt 100 mjukvaruutvecklare på 37 företag.

Resultatet från undersökningen (Katzeff & Svärd, 1995; Katzeff & Svärd, 1996) visar på ett intresse för användbarhetsfrågor som till viss del börjar omsättas till handling. Fortfarande har man en hel del att lära från MDI-forskningen och enligt enkätundersökningen anser systemutvecklare i Sverige att det viktigaste MDI-forskningsområdet är *metoder för utveckling av användbara system.* Av den anledningen beslutade vi att inom projektets ram publicera en rapport om metoder för att hitta användbarhetsproblem hos datorsystem.

Syftet med rapporten är att ge en kort introduktion till metodområdet, samt att förmedla referenser till den som önskar fördjupa sig. Min förhoppning är att rapporten ska tillfredsställa en del av det intresse som enkätundersökningen påvisade.

Läsaren förutsätts ha ett intresse av användbarhetsfrågor och ha grundkunskaper i hur systemutvecklingsarbete bedrivs.

Innehållsförteckning

1 Sammanfattning

Rapportens fokus är att presentera ett antal kända metoder för att hitta användbarhetsproblem hos datorsystem. Följande sex metoder behandlas: heuristisk utvärdering, kognitiv genomgång, standardinspektioner, pluralistisk användbarhetsgenomgång, kooperativ utvärdering och enkätundersökningar.

I ett inledande kapitel beskrivs användarcentrerad utveckling (ACU) kortfattat – ett arbetssätt för att uppnå användbara datorsystem. Metoder för att hitta användbarhetsproblem utgör en central del i detta arbetssätt. I kapitlet definieras även begreppet användbarhet och vad som utgör användbarhetsproblem hos datorsystem diskuteras.

I kapitel 3 diskuteras rapportens metoder ur ett generellt perspektiv. Det konstateras att det finns olika syften med att använda metoderna. *Systemutvecklaren* kan försäkra sig om att systemet kommer att ha en hög användbarhet när det levereras och *kunden* kan kontrollera att systemet kommer att passa för användargruppen. Dessutom finns det en hel del ekonomiska vinster med att tillämpa metoderna.

I kapitel 4 och 5 presenteras de sex metoderna ur ett praktiskt perspektiv. De delas upp i de som hittar användbarhetsproblem utan att användare deltar och de som kräver användarmedverkan. För samtliga metoder ges en stegvis beskrivning av hur metoden fungerar samt vilka resultat man kan förvänta sig när den tillämpas. Tanken är att den som önskar ska kunna pröva metoderna efter att ha läst rapporten. Metodbeskrivningarna avslutas med läsanvisningar för den som vill fördjupa sig mer.

2 Inledning

Ett ofta rapporterad erfarenhet är att "80 procent av användarna bara använder 20 procent av systemens funktionalitet". Procentsatserna varierar självklart från system till system, men jag tror att de flesta känner igen situationen. Datorsystem är svåra att lära sig, och det är få som behärskar all funktionalitet hos applikationerna och systemen. Vad beror det på? Hur kommer det sig att systemen och människorna har så svårt att fungera ihop? Vad kan göras för att få människor och datorsystem att fungera bättre ihop?

Idag utvecklas datorprogram, datorsystem och websidor i en ständigt ökande hastighet och av fler och fler människor. I ett typiskt utvecklingsprojekt är tiden begränsad och den mesta energin läggs på att åstadkomma tekniskt fulländade system. Systemens funktionalitet testas systematiskt under flera veckor, men åt den kanske viktigaste komponenten i systemet – människan – ägnas lite uppmärksamhet. En anledning till detta är att de som utvecklar systemen inte vet tillräckligt mycket om Människa-Dator-Interaktion (MDI), d v s samspelet mellan människor och datorer.

Även om man som systemutvecklare vill skapa ett användbart system för slutanvändaren är det mycket svårt. Utvecklaren har genom mångåriga datastudier och praktisk erfarenhet av datorer blivit formad till "dataexpert" och har därför svårt att förutse de problem som användarna kommer att stöta på. För att lyckas konstruera mer användbara datorsystem krävs en förändrad syn på utvecklingen. I stället för att enbart sätta tekniken i fokus under utvecklingsarbetet bör användaren av systemet beaktas i större utsträckning. Detta bör göras medvetet med hjälp av en s k *användarcentrerad utvecklingsmetodik (ACU)*.

ACU baserar sig på fyra grundläggande principer (Gould & Lewis, 1983; Katzeff & Svärd, 1995) som när de används bidrar till systemets användbarhet beaktas under hela systemutvecklingsprocessen. Dessa principer är:

- tidig fokus på användare
- tidig och kontinuerlig test av användbarhet
- iterativ utveckling
- integrerad utveckling

Denna rapport behandlar den andra av de ovan nämnda principerna för ACU – tidig och kontinuerlig test av användbarhet – och syftar till att presentera ett antal *metoder för att hitta användbarhetsproblem* hos datorsystem. Det finns en uppsjö av olika metoder för detta ändamål och denna rapport syftar till att ge en introduktion till några av de mest kända. Antalet har medvetet behållits vid ett överblickbart antal för att underlätta för rapportens läsare.

2.1 Användbarhet

För att kunna hitta användbarhetsproblem är det viktigt att ha en klar och tydlig uppfattning om vad som karaktäriserar ett användbart datorsystem. Det kan därför vara intressant att fundera över följande frågor: Vad är det som påverkar vår uppfattning om ett systems användbarhet? Vilka egenskaper har de system som är användbara för oss? Vad krävs för att ett datorsystem ska användas? Det kan konstateras att en mängd aspekter är med och påverkar.

Först och främst måste systemet var stabilt och inte krascha när det används. Dessutom måste det erbjuda nytta, d v s funktionalitet som användarna vill ha. Ett system som inte uppfyller dessa kriterier kommer knappast att användas i någon större utsträckning. Slutligen är systemets användbarhet avgörande för hur väl användarna kan tillvarata funktionaliteten. Men vad menas egentligen med att ett datorsystem är användbart?

De flesta MDI-forskare är eniga om att det är ett sammansatt begrepp, och i Jakob Nielsens definition (1993) förknippas det med följande fem egenskaper:

- *Lättlärdhet*: Systemet ska vara lätt att lära, så att användaren snabbt kan börja utföra sina arbetsuppgifter.
- *Effektivt*: Systemet ska vara effektivt att använda, så att användaren kan nå en hög produktivitetsnivå när han/hon väl lärt sig använda det.
- *Lätt att minnas*: Systemet ska vara lätt att komma ihåg, så att den som använder systemet bara ibland, lätt kan komma tillbaks till systemet efter en längre tid utan att behöva lära sig det från början igen.
- *Felhantering*: Systemet ska leda till att dess användning har låg felfrekvens. Användaren ska därmed göra få fel och de fel han/hon gör ska lätt kunna åtgärdas. Fel med katastrofala följder får inte förekomma.
- *Tillfredsställelse*: Systemet ska vara angenämt att använda.

Andra definitioner förekommer i MDI-litteraturen av bl a Preece et al. (1994), Lövgren (1993), och Dix et al (1993), men de skiljer sig inte nämnvärt från Nielsens definition. Det finns även en ISO-standard för användbarhet, ISO-9241-11, som förknippar begreppet med effektivitet och tillfredsställelse. ISO-definitionen har även en markering att ett systems användbarhet bestäms relativt dess användare, deras uppgifter och den miljö det används i. Det betyder att *ett och samma system kan ha skiftande användbarhet för olika människor, för olika uppgifter och i olika miljöer*. Exempelvis kan en ordbehandlare vara olika användbar för olika människor, med olika uppgifter (t ex att skriva texter med många bilder eller texter utan bilder) och i olika miljöer.

2.2 Användbarhetsproblem

Genom att begreppet användbarhet definierats ovan kan användbarhetsproblem definieras *som de aspekter av ett datorsystem som bidrar till att användbarheten hos systemet försämras*. De flesta av oss har nog en intuitiv känsla för vad som menas med detta, och kanske associerar till olika svårhanterade apparater, t ex den svårprogrammerade videon där hemma. Om man utgår från Nielsens definition av användbarhetsbegreppet blir användbarhetsproblem de aspekter av ett datorsystem som:

- *gör det svårare att lära för användaren*: t ex avvikelser från gängse standarder eller brist på återkoppling
- *försämrar produktiviteten när det används korrekt*: t ex opraktisk in- och utmatning eller långa svarstider
- *gör det svårare att minnas för användaren*: t ex inkonsekvent eller ologisk användning av systemets begrepp
- *leder till att användaren kan göra fel:* t ex oklara och dåliga namn på menyer och knappar
- *försvårar arbetet med att reparera fel:* t ex kryptiska felmeddelanden eller att en operation inte kan ångras
- *är allmänt irriterande för användaren*: t ex skrikiga färger eller konstiga dialoger.

Användbarhetsproblem kan således vara av skiftande karaktär, och dessutom olika allvarliga. Vissa problem är allvarligare än andra och kan vara avgörande för om ett system används eller ej. Till exempel upplevs de långa svarstiderna från vissa platser i USA som så enerverande för europeiska WWW-användare att många drar sig för att kontakta dem. I stället hämtas informationen från andra källor. Detta problem är således mycket allvarligt eftersom det medför att de amerikansk källorna inte kontaktas p g a användbarhetsproblemet med svarstiden.

3 Allmänt om metoderna

3.1 Syften

Det vanligaste syftet med att leta efter användbarhetsproblem är att försöka åtgärda dem i ett senare skede. Om problemen inte rättas till innan systemet släpps kommer de att upptäckas av användarna som spenderar onödigt mycket tid med systemets inlärning, felkorrigering och genomförandet av de dagliga arbetsuppgifterna med systemet. Kostnaden för att använda ett system med låg användbarhet drabbar då företaget som köpt systemet och de anställda i form av frustration och ineffektivitet.

Genom att tillämpa dessa metoder från början i systemutvecklingsprocessen kan många problem identifieras och åtgärdas. Vid upprepad tillämpning av metoderna kan man jämföra antalet funna användbarhetsproblem och förhoppningsvis se att antalet minskar. På så sätt kan man få ett kvitto på att systemets användbarhet förbättras mellan varje utvärdering. Ju tidigare problemen hittas, desto lägre blir också kostnaderna för att rätta till dem, eftersom de aktiviteter som berörs av systemförändringar inte hunnit komma så långt. Pressman (s 21, 1994) uppskattar att det är 60-100 gånger dyrare att genomföra systemförändringar efter att systemet tagits i drift än att genomföra samma förändringar under systemets tidiga utvecklingsfaser.

Ett annat syfte med att hitta användbarhetsproblem hos datorsystem kan vara att göra de som utvecklar systemen uppmärksamma på dem. Få människor vill vara med om att skapa något man vet är dåligt, och när systemutvecklarna väl har förstått att användarnas problem med systemet brukar deras intresse för att åtgärda dem öka. Dessutom kan en ökad förståelse för användbarhetsfrågor i organisationen i förlängningen bidra till att en användarfokuserad utvecklingsmetod får fäste, vilket i sin tur borgar för att mer användbara system utvecklas.

De föregående syftena bygger på att systemet ska förändras och åtgärdas efter att problemen avslöjats. Det kan även vara intressant att utföra en s k summativ användbarhetsutvärdering. Syftet med en sådan är inte att i slutändan förändra systemet, utan att skaffa sig en uppfattning av hur användbart det aktuella systemet är för en specifik användargrupp. En summativ användbarhetsutvärdering ger svar på frågan "Är det här systemet användbart för våra användare, i deras arbetsmiljö?" En kund eller en potentiell köpare av ett system kanske är mera intresserad av tillämpa metoderna för detta syfte, medan den som utvecklar systemet är mer intresserad av att skapa användbara datorsystem. Metoderna i denna rapport kan lämpar sig väl för både kund och utvecklare.

3.2 Klassificering

Metoderna i denna rapport finns sammanfattade i figur 1. De skiljer sig åt vad gäller tillvägagångssätt, grad av användarmedverkan och kunskapskrav för att tillämpa dem. Samtliga metoder resulterar i en lista med användbarhetsproblem hos det aktuella datorsystemet. De talar inte om hur dessa problem ska åtgärdas. Det är upp till de som utvecklar systemet att komma på lösningar. Dock gäller att många nya designidéer uppstår ur de diskussioner som ofta förekommer när problemen identifieras.

I denna rapport är metoderna grupperade efter om de kräver användarmedverkan eller ej. Orsaken till denna indelning är att forskare, bl a Katzeff & Svärd (1995) och Poltrock & Grudin (1994), konstaterat att många systemutvecklare har problem med att få tag på användare under systemutvecklingen. Genom att dela in metoderna i två grupper, de som kan genomföras utan användare och de som baserar sig på användarmedverkan, blir det förhoppningsvis lättare för läsaren att hitta en lämplig metod för det aktuella systemutvecklingsprojektet.

Metoderna i den första gruppen, *metoder utan användare,* baserar sig på inspektioner av gränssnittet och dialogen med användaren. Här identifieras de delar som inspektörerna anser

kan ställa till problem för användaren. Problemen som hittas är således av potentiell natur, men är relevanta och påverkar användbarheten hos systemet i stor utsträckning.

Den andra gruppen, *metoder med användare*, använder sig av slutanvändare för att identifiera problem hos systemet. Problemen som hittas har sin grund i studier av slutanvändare och datorsystemet och blir ofta av annan art än den föregående gruppens. Det kan därför vara bra att använda sig av metoder från båda grupperna för att fånga upp så många problem som möjligt.

Metodens namn	Tillvägagångssätt	Användare som behövs för bra resultat	Fördelar med metoden	Att tänka på när metoden används	Vilka problem hittas?
Heuristisk utvärdering	Individuellt letande av potentiella problem baserat på 10 tumregler.	Inga	Kan användas tidigt i utvecklingen. Snabb och "billig".	3-5 personer som utvärderar är det mest kostnads-effektiva antalet.	Delar av gränssnittet som avviker från de tio tumreglerna.
Kognitiv genomgång	En grupp bedömer hur lätt systemet är att lära genom "utforskning". Analysen baseras på antaganden om användargruppen.	Inga	Gör djupa analyser av systemets lärbarhet. Ger även input till vilken typ av tilläggsinfo som behövs.	Fånga upp allt som sägs när "tänkbara interaktions-sekvenser" beskrivs.	Delar av gränssnittet som gör det svårare för användar-gruppen att bedriva utforskande inlärning.
Standard-inspektioner	Gruppgenomgång där ett systems samstämmighet med en viss gränssnitts-standard bedöms.	Inga	Problemen som hittas är av konkret natur och är oftast lätta att ändra.	Var flexibel! Ibland kan systemets användbarhet öka om standarden inte följs.	Avvikelser från den gränssnitts-standard som systemet ska följa.
Pluralistisk användbarhets-genomgång	Alla i gruppen agerar användare och löser uppgifter. Påföljande diskussion.	2-3	Systemet behöver bara vara på papper. Ger input till vilken typ av tilläggsinfo som behövs.	Viktigt att vara ödmjuk mot användarna och att kunna ta emot kritik.	Oväntat användar-beteende när uppgifterna löses.
Kooperativ utvärdering	Observatör och användare letar tillsammans efter problem.	6-8, men även färre användare ger bra resultat	"Naturlig" situation för användarna.	Få användarna att känna sig delaktiga i metoden.	Oväntat användar-beteende och kommentarer om gräns-snittet.
Enkät-undersökningar	Användarnas subjektiva åsikter om systemet undersöks.	30 och uppåt, men även ett fåtal enkätsvar kan ge värdefull information.	Kostnads-effektiv. Samlar info om hur användarna uppfattar systemet.	Var noggrann när du designar och analyserar enkäterna.	Många olika typer av problem kan härledas ur enkätsvaren.

Figur 1. En översikt över metoderna i rapporten.

3.3 Val av metod

Metoderna i denna rapport utgör en verktygslåda med vilken det är möjligt att hitta många typer av användbarhetsproblem hos datorsystem. Val av metod kan bero på många olika aspekter:

- *När i utvecklingsprocessen vill jag använda metoden?* Några metoder i rapporten är särskilt väl lämpade för tidig användning. Heuristisk utvärdering, kognitiv genomgång och pluralistisk användbarhetsgenomgång kan alla användas när systemet bara är skisserat på papper. De andra tre metoderna lämpar sig bättre för mer eller mindre implementerade system.

- *Vilken typ av problem vill jag hitta?* Eftersom metoderna har olika fokus hittar de delvis olika typer av användbarhetsproblem. Generellt sett kan sägas att de användarbaserade metoderna hittar "riktiga" problem som användarna upplever när de använder systemet. Heuristisk utvärdering, kognitiv genomgång och standardinspektioner hittar aspekter hos datorsystemet som troligtvis kommer att uppfattas som användbarhetsproblem. Dessa utgörs i regel av bristande anpassning till användarens språk, inkonsekventa gränssnitt och avvikelser från standarder och tumregler för gränssnittsutformning.

- *Vilka resurser har jag till mitt förfogande?* Om mycket tid och pengar finns så kan flera metoder tillämpas. Eftersom de största skillnaderna mellan metoderna finns mellan de användarbaserade och de som saknar användare, kan det vara bra att använda en metod ur varje grupp, t ex heuristisk utvärdering och kooperativ utvärdering.

- *Finns användare att tillgå?* Tillgång till användare kan vara en begränsande faktor vid val av metod. Om man vill använda sig av användarbaserade metoder, och inte kan få tag på riktiga slutanvändare kan andra människor agera slutanvändare i metoderna. Kooperativ utvärdering eller pluralistisk användbarhetsgenomgång kan då genomföras med "låtsas-användarna" och problem kan hittas. De mest realistiska resultaten får man om "låtsas-användarnas" datorerfarenhet liknar den riktiga användargruppens.

- *Vilken kompetens finns att tillgå?* De flesta metoder i denna rapport kan användas utan några speciella förkunskaper. Dock gäller att de bästa resultaten får personer som har MDI-kunskaper och/eller har erfarenhet av att tillämpa metoderna. Dessa är mer vana att sätta sig in i användarens situation och hittar därför fler och andra typer av problem än de som saknar dessa kunskaper. Det är också bra att ha kunskaper om det aktuella systemet, för att bättre kunna analysera problemen.

När man väl har bestämt sig för en metod kan det vara väl värt att genomföra ett pilottest av metoden innan den används i full skala. Pilottestet syftar till att öka erfarenheten av metoden och att befästa de teoretiska kunskaperna. Ett praktiskt test med metoden får dig att känna dig säkrare när metoden väl ska tillämpas på ett skarpt system.

4 Metoder utan användare

4.1 Heuristisk utvärdering

Denna metod brukar av Jacob Nielsen (t ex 1993) marknadsföras som en rea-metod eller en "billig" metod. Anledningen är att den är lätt att lära, går fort att genomföra och kan användas av andra än MDI-experter (t ex systemutvecklare). Utgångspunkten för den som genomför en heuristisk utvärdering är en checklista med tio tumregler (på engelska heuristics) för bra gränssnittsutformning. Dessa finns beskrivna i figur 2 och sammanfattar vad man bör tänka på vid design av användbara systemgränssnitt.

Metodens syfte är att identifiera avvikelser från tumreglerna hos det system som granskas. Avvikelserna utgör potentiella användbarhetsproblem, och bör rättas till i nästa version av systemet. En av metodens fördelar är att den kan tillämpas på färdiga systemgränssnitt eller på skisser av ett blivande system. Detta gör heuristisk utvärdering särskilt lämplig att använda i de tidiga faserna av systemutvecklingen, när systemet bara finns skisserat.

1. *Använd en enkel och naturlig dialog*: varje extra funktion eller informationsdetalj blir ytterligare belastningar för användaren. Det ideala är att presentera den information som användaren behöver vid exakt rätt tidpunkt. Den grafiska designen och färgerna i gränssnittet ska förenkla tolkningen av informationen. Systemets gränssnitt bör även stödja användarens naturliga sätt att lösa uppgifterna.
2. *Tala slutanvändarens språk*: Systemets gränssnitt bör så långt möjligt använda användarnas modersmål och de begrepp som normalt används vid uppgiftshanteringen som systemet ska stödja. Detta omfattar bl a menyer, knappar, dialoger och val av ikoner.
3. *Minimera användarens minnesbelastning*: Datorer är bra på att minnas detaljer och människor är bättre på att känna igen saker som våra sinnen uppfattar. Därför bör systemet hjälpa användaren med detaljer och de begrepp som användaren måste lära sig utantill ska vara få och konsekventa.
4. *Var konsekvent*: Om användarna vet att samma kommando eller samma handling alltid har samma effekt känner de sig säkrare i systemet och vågar prova sig fram. Även information och dialoger bör vara formaterade på samma sätt för att stödja det mänskliga igenkännandet.
5. *Ge återkoppling (feedback):* Systemet ska kontinuerligt informera användaren om vad det gör och hur det tolkar användarens inmatningar. Återkoppling är extra viktig när svarstiden från systemet är lång, och ska ges både när systemet utfört kommandon korrekt respektive när de misslyckats.
6. *Ha tydliga vägar tillbaka:* Användare gillar inte att känna sig låsta, utan vill så långt möjligt kunna kontrollera dialogen med datorn. Därför bör systemet erbjuda vägar tillbaka (t ex avbryt) från dialoger och andra komponenter i interaktionen. Det är även viktigt att operationer kan ångras efteråt, eftersom användaren då vågar utforska systemet i större utsträckning.
7. *Stöd genvägar*: Även om det är viktigt att systemet ska kunna hanteras med ett fåtal generella interaktionsprinciper, är det viktigt att den erfarne användaren kan hantera de vanligaste kommandona effektivt. Kortkommandon och ikoner som utför kommandon är exempel på sätt att stödja detta.
8. *Ge bra felmeddelanden*: Om en felsituation uppstår bör detta meddelas användaren med ett klart och tydligt språk som användaren förstår. Användaren ska inte behöva läsa i en manual för att förstå problemet. Felmeddelandets innehåll ska ge tillräckligt med hjälp för att lösa problemet. Meddelandena får heller inte lägga skulden på användaren. Det kan göra att användaren inte vågar försöka igen.
9. *Förhindra fel:* Om det finns en möjlighet att göra fel, så kommer någon av användarna att göra det. Det gäller att systemet är designat så att det inte går att åstadkomma fel. Ett generellt tips är att system inte bör ha olika tillstånd (modes), då detta är förvillande för användarna.
10. *Ge bra hjälp och dokumentation:* Även om de flesta användare inte läser hjälpdialoger och manualer, bör den intresserade kunna förkovra sig snabbt och effektivt. För det krävs att information är vara väl strukturerad och pedagogiskt skriven.

Figur 2. Tio tumregler för design av användbara systemgränssnitt.
(Nielsen, 1993; Nielsen & Mack, 1994)

Så gör man en heuristisk utvärdering

En grupp människor väljs ut att genomföra metoden. Dessa arbetar individuellt och var och en får en lista med de tio tumreglerna samt en pappers- eller datorbaserad modell av systemet. Under utvärderingen beaktas systemet utifrån de tio tumreglerna, t ex genom att frågor ställs: "Minimeras användarens minnesbelastning i den här dialogen?" "Används användarens språk?" "Är dialogen så enkel och naturlig som möjligt?" o s v.

En typisk utvärdering av ett system tar en till två timmar för en person att genomföra. Under denna tid bör utvärderaren gå igenom gränssnittet åtminstone två gånger. Första gången kan han eller hon få en känsla för interaktionen och systemets omfattning. Andra gången kan specifika delar av gränssnittet utvärderas i detalj.

Det är omöjligt för en person att hitta alla användbarhetsproblem hos ett system, och antalet som hittas ökar exponentiellt med antalet utvärderare. Undersökningar har visat att 3-5 personer upptäcker 60-75% av alla användbarhetsproblemen hos systemet och att det är det mest lönsamma antalet (Nielsen & Mack, 1994). Den som genomför en heuristisk utvärdering behöver inte vara expert på Människa-Dator-Interaktion, men studier har visat (se t ex sid 59 i Nielsen & Mack, 1994) att de bästa resultaten uppnås när s k dubbelexperter utför heuristisk utvärdering. En dubbelexpert är en person som är expert på både MDI och systemets domän, d v s den specifika verksamhet och arbetsuppgift som systemet ska stödja. Med heuristisk utvärdering av ett administrativt bankdatorsystem får man därmed de bästa resultaten av en person som är mycket kunnig inom bankens organisation, ekonomiområdet och MDI.

Resultat från heuristiska utvärderingar

Varje person som genomför en heuristisk utvärdering kommer att hitta ett antal ställen i gränssnittet där systemet inte följer tumreglerna. Dessa ställen utgör potentiella användbarhetsproblem som bör åtgärdas. Resultatet av en heuristisk utvärdering är därför en lista med användbarhetsproblem hos systemet. Det är viktigt att utvärderarna noga specificerar varför de identifierade problemen utgör användbarhetsproblem (referera till tumreglerna) och att varje problem beskrivs för sig. Detta görs för att förenkla korrigeringen av problemet och för att möjliggöra prioriteringar bland problemen.

Slutligen sammanställs de olika utvärderarnas problemlistor till en enda lista. Denna ges till systemutvecklarna som får i uppdrag att åtgärda problemen i nästa systemdesign. Ett sätt att överbrygga klyftan mellan utvärderare och systemutvecklare är att ha ett genomgångsmöte med de båda grupperna. Syftet med mötet är att deltagarna tillsammans ska diskutera och försöka lösa. På så sätt kan utvärderarna förklara problemen för systemutvecklarna samtidigt som nya kreativa lösningar kan hittas.

Var kan man läsa mer om heuristisk utvärdering?

- Nielsen (1993), ger på 50 sidor ganska utförliga beskrivningar av de tio tumreglerna och en kortare beskrivning av heuristisk utvärdering.
- Nielsen & Mack (1994): Nielsen presenterar i ett kapitel (2) metoden på 40 sidor och statistiska analyser av dess användning. Här finns dessutom ett par kapitel där effektiviteten hos heuristisk utvärdering jämförs med andra metoder för att hitta användbarhetsproblem.

4.2 Kognitiv genomgång

Kognitiv genomgång (beskrivs i Nielsen & Mack, 1994) är en metod som syftar till att identifiera de delar av ett datorsystem som försvårar utforskande inlärning. Vid utforskande inlärning skaffar sig användarna kunskaper om nya funktioner bara när deras arbete kräver det och medan de arbetar med sina vanliga uppgifter. Studier har visat (Nielsen & Mack nämner bl a en studie av Carroll och Rosson 1987) att många användare föredrar denna typ av inlärning framför djupare och mer formella studier.

Metoden kan tillämpas tidigt i systemutvecklingen eftersom systemet antingen kan vara skisserat på papper eller vara datorbaserat. Kognitiv genomgång liknar andra genomgångsmetoder, t ex kravgenomgångar, i det att en person presenterar ett förslag för en grupp som granskar och utvärderar förslaget. I en kognitiv genomgång studeras hur lätt användarna kan lära sig att utföra ett antal typiska uppgifter genom att använda utforskande inlärning.

Kognitionspsykologer, bland annat Norman[1] (1988), har funnit att människor tillämpar utforskande inlärning av datorsystem på ungefär följande sätt:

1. Formulerar en uppgift som de vill utföra med systemet
2. Utforskar gränssnittet och letar efter ikoner, menyval och kommandon som de tror kan bidra till uppgiften blir löst
3. Väljer och utför den systemoperation som verkar lämpligast för detta syfte
4. Observerar gränssnittets reaktioner och utvärderar om operationen haft avsedd effekt
5. Formulerar en ny uppgift att lösa baserat på utfallet i steg 4.

De flesta människor har en s k etikettbaserad strategi vid denna typ av inlärning. Med det menas att de letar efter och väljer operationer vars bild eller namn matchar uppgiften som de vill utföra. Om en användare till exempel vill skriva ut ett dokument väljer han/hon troligtvis menyvalet "skriv ut..." eller knappen med en skrivare på. De kritiska delarna av ett datorsystem blir därmed de som utgör en länk mellan användarnas mål med uppgiften och den korrekta operationen för att utföra uppgiften. Återkopplingen spelar här en central roll eftersom den används för att utvärdera resultatet av den valda operationen. Här finns det risk för att användbarhetsproblem uppstår.

Så gör man en kognitiv genomgång

Förberedelser

Uppgifterna som ska användas vid den kognitiva genomgången ska väljas ut och sättas samman till ett scenario. De bör vara så typiska som möjligt för användargruppen och för systemet, och både enkla och sammansatta uppgifter bör väljas ut. Scenariot bör utformas så realistiskt som möjligt, t ex bör systemet innehålla samma mängd data som det kommande systemet är tänkt att innehålla. Till varje uppgift ska en lösning formuleras, d v s en sekvens av systemoperationer som löser uppgiften med systemet.

Genom att studera användargruppen kan användarnas verksamhetskunskaper, datorerfarenheter och inställning till datorer hittas. Alla användare är självfallet olika, men eftersom dessa faktorer påverkar den utforskande inlärningens resultat måste en eller flera generella användarprofiler skapas. Dessa innehåller beskrivningar av de förutsättningar som de typiska slutanvändarna har när de ska försöka lösa uppgifterna.

Genomförande

Gruppgenomgången inleds med att en person presenterar ett designförslag av systemet för en grupp utvärderare. Gruppens medlemmar kan bestå av designers, marknadsförare, teknikinformatörer, utbildare och MDI-experter. Uppgifterna i scenariot och beskrivningarna av användargruppens förkunskaper delas ut till samtliga medlemmar.

---

[1] Norman (1988) beskriver en generell sjustegs-modell för hur människor utför saker.

Därefter får gruppen i uppgift att diskutera fram en "tänkbar interaktionssekvens" mellan användaren och systemet för varje uppgift i scenariot. Sekvensen baserar sig på den modell av hur användarna bedriver utforskande inlärning och beskrivningarna av användarnas förkunskaper. Medlemmarna bör för varje uppgift ställa sig följande frågor:

- *Kommer användarna att leta efter rätt systemoperation?*
- *Kommer användarna att upptäcka att systemet erbjuder den rätta systemoperationen?*
- *Kommer användarna att associera den rätta operationen med den uppgift de vill lösa med systemet?*
- *Om den rätta operation väljs, kommer användarna att se att de är på rätt spår?*

När medlemmarna i gruppen diskuterar är det viktigt att det som sägs fångas upp. Speciellt viktigt är det att notera vad användarna behöver kunna för att klara av en uppgift samt vad användaren lär sig när han/hon löser en uppgift med datorsystemet. För detta ändamål kan video, blädderblock eller anteckningar på overheadbilder användas.

Resultat från kognitiva genomgångar

Kognitiv genomgång gör en djupare analys av systemet och användargruppen för att hitta problem som påverkar systemets lärbarhet. Lärbarhetsbegreppet analyseras djupt och många problem hos systemet för att bedriva utforskande inlärning hittas. Dessutom ger metoden en tydlig bild av de kunskaper som krävs för att klara av uppgifterna med systemet.

Resultaten kan påverka det framtida systemet på flera sätt. Dels kan en del konkreta åtgärder göras för att anpassa systemet till användarens förkunskaper eller på andra sätt höja systemets lärbarhet för användargruppen. Dels kan resultaten direkt påverka av manualer och on-linehjälp till systemet.

Det är viktigt att komma ihåg att metoden är smalt fokuserad på lärbarhet och kan missa andra egenskaper som förknippas med användbarhet. Kognitiv genomgång bör därför kompletteras med andra metoder för att utvärdera fler aspekter av systemets användbarhet.

Kvaliteten hos resultaten beror delvis på gruppens sammansättning och kompetens. Studier har visat (t ex Jeffries et al. 1991) att de bästa resultaten uppnås om gruppens medlemmar har en kognitionspsykologisk bakgrund.

Var kan man läsa mer om kognitiv genomgång?

- Nielsen & Mack (1994): I ett kapitel (5) om 35 sidor presenteras metoden av sina grundare Wharton, Rieman, Lewis & Polson. Här finns dessutom ett par kapitel där kognitiva genomgångar jämförs med andra metoder för att hitta användbarhetsproblem.

4.3 Standardinspektioner

Standarder förekommer i många sammanhang och syftar till att underlätta användningen av komplexa och icke intuitiva system. Norman (1988) tar som exempel den gängse standarden att klockor går medsols och menar att standardisering är den sista utvägen när alla andra användbarhetsknep brister. På så vis behöver användaren bara lära sig en gång, och kan därefter använda kunskapen på alla andra tillämpningar. Användaren kan lättare förutse hur systemet kommer att bete sig och gör därför färre fel och får ett ökat självförtroende att lära sig nya system.

Det finns många olika standarder för grafiska gränssnitt hos datorsystem, bl a har Microsoft publicerat standarder för Windows-applikationer. Andra exempel är Macintosh för grafiska applikationer på Appledatorer och Motif för applikationer i Unixmiljö. Det är även vanligt att företag har egna standarder för de interna systemen. De olika standarderna liknar varandra men skiljer sig i hur fönster och dialogrutor ska se ut, hur interaktion med mus och tangent-

bord ska ske och hur menyer ska grupperas och namnges. Det är viktigt betona att det inte är valet av standard som är avgörande för användbarheten hos applikationen utan det faktum att standarden används konsekvent i alla användarens applikationer (Kool & Wingstedt, 1992).

Många företag som utvecklar mjukvara strävar efter att följa dessa standarder i sina applikationer. Detta är svårt eftersom standarderna ofta beskrivs väldigt detaljerat och studier har visat att det sällan lyckas fullt ut (se s 234 i Nielsen, 1993; Nordqvist 1996). En av anledningarna är att standarderna är svåra att greppa och kommunicera och därför ofta tolkas olika.

Ett sätt att förbättra ett systems samstämmighet med gängse standard är att genomföra standardinspektioner. Inspektionerna syftar till att identifiera ett systems avvikelser från en standard, t ex Microsoft Windows, vilka kan utgöra användbarhetsproblem. Anledningen är att den användare som är van vid att använda Windows-applikationer *förväntar* sig att gränssnittet hos systemet ska stödja de kortkommandon, menyalternativ etc som Windows-standarden specificerar. Avvikelser från standarden kan därför försvåra inlärningen av systemet och försämra effektiviteten när användaren inte kan hantera det på samma sätt som de andra applikationerna (som följer Windows-standarden).

Så gör man en standardinspektion

Metoden kräver tillgång till expertis på standarden och kan i korthet beskrivas så här:

1. Experterna på standarden bekantar sig med systemet eller prototypen, för att få en känsla för systemets omfattning. Här kan det vara bra om någon systemutvecklare finns tillgänglig för eventuella frågor.
2. Experterna och representanter för utvecklargruppen samlas till ett genomgångsmöte. Under mötet går man noggrant igenom systemet och experterna pekar ut de ställen där systemet avviker från standarden. Som stöd för de systematiska genomgångarna kan checklistor på det som ska kontrolleras vara bra att ha.
3. Avvikelserna och eventuella ändringar diskuteras. Ofta beror avvikelserna på att utvecklarna inte har funderat över standarderna när de utvecklat systemet, och då är det enkelt att besluta att ändra gränssnittet (kap. 4 i Nielsen & Mack, 1994). Andra avvikelser kan kräva längre tids diskussion och eftertanke. Systemet kanske ibland kan bli mer användbart om inte standarderna följs, vilket kan kontrolleras med andra metoder i denna rapport. Det viktiga är i så fall att avvikelsen från standarden är medveten.
4. När hela systemet har gåtts igenom har en hel del designförändringar blivit beslutade. Det är nu upp till utvecklarna att genomföra dem.

Standardinspektioner är tidskrävande och kan inte heller de hitta ett systems alla avvikelser från standarderna. Ansatser har gjorts för att utveckla datorstöd som stöd för inspektionsprocessen. Vid STIMDI-konferensen 1996 presenterades ett prototypverktyg som automatiskt kontrollerar avvikelser från MS Windowsstandarden (Nordqvist, 1996). Verktyget har bl a funnit att Microsoft Word innehåller ett antal avvikelser från Microsofts egen fönsterstandard.

Resultat från standardinspektioner

Avvikelser från standarder identifieras, vilka kan utgöra användbarhetsproblem. Avvikelserna är av konkret natur, t ex "menyalternativ X kallas i standarden för Y", vilket ofta gör dem lätta att förändra. Ofta leder standardinspektioner till att systemutvecklarna i fortsättningen blir bättre på att följa standarderna. Erfarenheter av att genomföra standardinspektioner på Digital visar bl a på detta (kap 4 i Nielsen & Mack, 1994).

Det är viktigt att betona att inspektionerna bara hittar användbarhetsproblem som har med lärbarhet och minnesbelastning att göra. Det är därför viktigt att komplettera inspektionerna med andra metoder för att komma åt fler aspekter hos systemets användbarhet. Att ett system följer en fönsterstandard behöver heller inte betyda att det är lätt att lära. Det gäller dock att det är lättare att lära ett system som följer en fönsterstandard (som användaren kan) än ett som innehåller en mängd avvikelser från densamma.

Var kan man läsa mer om standardinspektioner?

- Nielsen & Mack (1994): Wixon, Jones, Tse & Cascady berättar på 25 sidor i ett kapitel (4) om sina erfarenheter av att genomföra standardinspektioner och andra inspektionsmetoder på Digital Equipment Corporation.
- Kool & Wingstedt (1992): jämför i en rapport på 60 sidor sex olika standarder för grafiska gränssnitt: CUA, Motif, MS Windows, Open Look, Macintosh och Nextstep.
- Nordqvist (1996): berättar om sina erfarenheter med att utveckla och pröva ett dator-verktyg som inspekterar avvikelser från standarder. Implementationens arkitektur beskrivs även kortfattat.

5 Metoder med användare

5.1 Pluralistisk användbarhetsgenomgång

Pluralistisk användbarhetsgenomgång uppstod på IBM när man p g a tidsbrist i ett utvecklingsprojekt försökte effektivisera användbarhetsarbetet. I stället för att låta användare och MDI-experter titta på systemet var för sig lät man dem göra det samtidigt, tillsammans med systemutvecklarna. Resultatet var lyckosamt och metoden har fått sitt namn eftersom systemutvecklare, MDI-experter och användare tillsammans bidrar till att systemet undersöks utifrån flera olika (pluralistiska) perspektiv.

Vid pluralistiska användbarhetsgenomgångar antar MDI-experter, användarrepresentanter och systemutvecklare alla rollen som användare av systemet. Var och en försöker lösa typiska användaruppgifter baserat på skärmbilder av systemet. De olika lösningsförslagen diskuteras därefter och eventuella användbarhetsproblem utkristalliseras. Under diskussionerna kan även nya designidéer skisseras i syfte att åtgärda problemen och förbättra systemets användbarhet.

En fördel med metoden är att systemet bara behöver vara skisserat på papper. Eftersom det sällan finns någon dokumentation färdig så tidigt brukar någon av systemutvecklarna få agera "levande manual". De som går igenom systemet kan därmed ställa frågor högt, och den "levande manualen" svarar högt. Systemutvecklarna får på så vis se vilka delar av gränssnittet som ställer till problem för slutanvändarna och får tidiga injektioner på vilken typ av dokumentation och hjälp som behövs till systemet.

Så gör man en pluralistisk användbarhetsgenomgång

Förberedelser

För att kunna genomföra metoden väljs ett antal typiska uppgifter med systemet ut, och sätts ihop till ett tänkbart användningsscenario. Till varje uppgift ska en korrekt lösning definieras. I många fall går det ju att lösa en systemuppgift på flera sätt, men vid pluralistiska användbarhetsgenomgångar tvingas alla användare att välja samma väg genom gränssnittet när de ska lösa uppgifterna.

För de delar av systemet som behövs för att kunna lösa uppgifterna ska gränssnittet representeras i form av skärmbilder eller skisser av hur bildskärmen kommer att se ut. Dessa buntas ihop i sekventiell ordning, d v s i den ordning de presenteras för användaren när uppgifterna utförs. Uppgifterna i scenariot samt traven med skärmbilder kopieras upp för att kunna delas ut till samtliga gruppmedlemmar vid systemgenomgången.

Eftersom metodens resultat till mycket stor del beror på klimatet i gruppen under genomgången är det mycket viktigt att förbereda systemutvecklarna på att deras produkt kommer att kritiseras. Det är viktigt att de inte tar kritiken personligt och i stället är medvetna om att genomgångsmötet är till för att göra deras produkt mer användbar i slutändan.

Genomförande

Innan genomgången av systemet kan påbörjas bör en introduktion till metoden ges. Uppgifterna och traven med skärmbilder delas ut till samtliga i gruppen. Metoden förklaras och deltagarna instrueras att inte vända blad och gå vidare till nästa uppgift förrän den föregående är avslutad av samtliga deltagare.

En av systemutvecklarna kan därefter ge en kort översikt av systemets huvudbegrepp och förklara hur det fungerar i grova drag. Denna introduktion är tänkt att ersätta den initiala kunskap som många slutanvändare har när de närmar sig ett system.

Utvärderarna går därefter tillsammans igenom varje uppgift i scenariot på följande sätt:

1. Alla gruppmedlemmar skriver på skärmbilden ner den handling de skulle välja för att försöka lösa uppgiften. Detta bör skrivas så detaljerat som möjligt, t ex "tryck pil-ned 3 gånger och därefter enter-tangenten". Eftersom medlemmarna i gruppen har olika bakgrund och erfarenhet av det aktuella systemet kan det bli en del väntan innan nästa steg kan påbörjas.
2. När alla skrivit ned sitt lösningsförslag läses "rätt" lösning upp.
3. Användarrepresentanterna presenterar sina lösningsförslag för gruppen och diskuterar svårigheter med att lösa uppgiften. MDI-experterna agerar användarnas "advokater" genom att tolka användarnas svar och uttyda användbarhetsproblem. Systemutvecklarna är under tiden tysta och kan på så vis inte påverka användarnas utvärderingsresultat. Det är viktigt att utvecklarna har en välkomnande attityd till användaråsikterna eftersom det är svårt att kritisera systemet när systemutvecklarna sitter med.
4. När användarna har tömt ut sina svar får systemutvecklarna yttra sig och brukar ofta förklara varför systemdesignen ser ut som den gör. Nya designidéer kan därefter eventuellt skisseras i syfte att försöka lösa de identifierade användbarhetsproblemen.
5. Efter att varje uppgift är färdigdiskuterad får deltagarna fylla i ett kort frågeformulär om hur de uppfattat systemgränssnittets användbarhet när de löste uppgiften. Därefter går samtliga gruppmedlemmar vidare till nästa uppgift i scenariot och börjar om med steg 1.

När alla uppgifter har analyserats och diskuterats enligt de fem stegen ovan är den pluralistiska genomgången av systemet klar. Ibland kan även en extra enkät som rör hela systemets användbarhet delas ut och besvaras innan genomgångsmötet avslutas.

Resultat från pluralistiska användbarhetsgenomgångar

Metoden ger snabb återkoppling på systemets användbarhet och kan tillämpas i ett tidigt skede av systemutvecklingen. Ur lösningsförslagen får systemutvecklarna återkoppling på hur väl deras systemdesign fungerar. Vissa uppgifter är kanske lätta att lösa, medan andra är svårare och kanske inte klaras av alla. Genom att diskutera och analysera lösningsförslagen kan en rad konkreta användbarhetsproblem och förslag på designförändringar hittas. Systemutvecklarna kan även i sin roll som "levande manualer" få en uppfattning om vilket dokumentationsbehov som finns.

Resultaten beror som tidigare nämnts mycket på klimatet i gruppen. Användarrepresentanterna måste känna att de vågar kritisera och kommentera systemets design och gränssnitt. MDI-experterna har en nyckelroll för stämningen i gruppen, men även systemutvecklarnas inställning är väsentlig för resultatet.

Det är viktigt att känna till att en normal systemanvändare löser uppgifter i en helt annan miljö än den som förekommer vid pluralistiska användbarhetsgenomgångar. Andra resultat fås när ett körbart system används och när användaren sitter i lugn och ro och kan pröva sig fram med systemet. Pluralistisk användbarhetsgenomgång lämpar sig därför bra för tidig återkoppling i en iterativ process, men bör kompletteras med metoder som testar användbarheten i en mer verklig användningssituation.

Var kan man läsa mer om pluralistiska användbarhetsgenomgångar?

- Nielsen & Mack (1994): I ett kapitel (3) om 13 sidor presenteras metoden av Randolph G. Bias från IBM.

5.2 Kooperativ utvärdering

Kooperativ utvärdering utvecklades vid universitetet i York och har därefter uppmärksammats i MDI-kretsar och "sålts" till industrin. Metodens grundare, Peter Wright och Andrew Monk, anser att många användbarhetsproblem hos datorsystem beror på bristfällig kommunikation mellan de som utvecklar systemen och slutanvändarna. Kooperativ utvärdering är tänkt att tillämpas av systemutvecklare för att på så sätt öka deras kunskaper om användargruppen, men metoden kan även användas av människor med andra arbetsuppgifter i utvecklingsprojektet.

Vid en kooperativ utvärdering arbetar systemutvecklare och slutanvändare tillsammans (kooperativt) för att hitta användbarhetsproblem i interaktionen mellan systemet och användaren. En eller flera utvecklare observerar användare som försöker lösa ett antal specifika uppgifter med systemet. Användaren tänker hela tiden högt så att de som observerar lättare kan följa med i användarens tankekedjor och interaktion med systemet. Systemutvecklarna kan under observationerna själva se hur deras system fungerar ihop med användargruppen. Oväntat användarbeteende och kommentarer om gränssnittet ses som symptom för potentiella användbarhetsproblem.

Kooperativ utvärdering brukar betecknas som en kostnadseffektiv metod, eftersom den kan tillämpas av systemutvecklare utan några speciella färdigheter i MDI, är lätt att lära och tar relativt lite tid i anspråk. Metoden lämpar sig bäst när det finns en, helt eller delvis, körbar datormodell av det kommande systemet och när det finns en uppfattning om vilka arbetsuppgifter som systemet ska stödja för användargruppen. Vid kooperativa utvärderingar strävar man efter att hitta ett antal betydande användbarhetsproblem snarare än att hitta alla problem, vilket gör metoden lämpad för snabba resultat i en iterativ process.

Så gör man en kooperativ utvärdering

Förberedelser

Ett antal användare väljs ut till utvärderingen. Dessa bör så långt möjligt avspegla användargruppen, men om det inte går att få tag på riktiga slutanvändare kan människor med liknande datorbakgrund och domänkunskaper användas i stället. Det ideala antalet är mellan sex till åtta användare men även enbart en användare kan ge mycket värdefull information.

Till utvärderingen väljs även ett antal uppgifter ut. Dessa bör vara relevanta för användarens arbete, täcka de mest centrala delarna av datorsystemet och ta lagom med tid i anspråk. Det senare bör kontrolleras i förväg, så att den kooperativa utvärderingen bättre kan planeras. Tänk även på att de som inte har sett systemet förut behöver extra tid att lära känna det. Uppgifterna bör formuleras så konkret som möjligt, t ex "ändra i texten så att det står X i stället för Y", och skrivas ner på papper i förväg. För att utvärderingen ska få en bra start bör den första uppgiften bör vara så enkel att alla användare kommer att klara av den.

Genomförande

- Användaren löser uppgifterna på papperet och tänker hela tiden högt så att de som observerar kan följa användarens tankekedjor.
- Observatören ställer under tiden frågor för att bättre begripa hur användaren tänker och uppfattar systemet: Vad tänker du nu? Vad tror du händer om du trycker på den knappen? Vad vill du göra? Varför har systemet gjort så? Det är i detta sammanhang viktigt att inte ställa ledande frågor. Användaren måste tillåtas att göra misstag och själv tala om vilka problem han/hon har.
- Användaren får fritt kommentera systemet t ex genom att föreslå nya funktioner eller påtala bra eller dåliga egenskaper hos systemet.

- Observatören antecknar oväntat beteende hos användaren och kommentarer om systemets användbarhet. Om två systemutvecklare är observatörer kan en ställa frågor medan den andra antecknar. Det kan även vara lämpligt att spela in användarens interaktion med systemet på video.
- Om användaren känner sig osäker på systemet kan observatören tillfrågas. Fastnar användaren med en uppgift talar observatören om exakt hur den ska lösas.

Efter att användaren har löst alla uppgifterna avslutas den kooperativa utvärderingen med en diskussion. Systemutvecklaren frågar användaren hur han/hon har uppfattat systemet. Exempel på frågor som kan diskuteras är:

Vad var enklast/svårast?

Vad bör förändras?

Är namnet X på menyvalet Y bra?

o s v

När användaren gått kan det vara bra att sitta ner några minuter och komplettera anteckningarna. Efter att alla användare genomfört den kooperativa utvärderingen sammanställs observationerna från de olika användarna. Som stöd för minnet används videoinspelningen, men eftersom tiden är begränsad i de flesta systemutvecklingsprojekt behöver ingen djupanalys av det inspelade materialet göras. En utvärderingsrapport sammanställs med observationerna som därefter kan beaktas vid design av nästa systemmodell.

Resultat från kooperativa utvärderingar

Kooperativa utvärderingar resulterar i två typer av observationer rörande systemets användbarhet: *oväntat användarbeteende* och *användarkommentarer*. Om en användare gör något oväntat är det troligt att fler kommer att göra samma sak vilket indikerar ett potentiellt användbarhetsproblem hos systemet.

Kommentarer om gränssnittet kan vara tecken på användbarhetsproblem eftersom det först är när användaren får problem som gränssnittet brukar kommenteras, t ex "Jag hittar inte menyvalet...", "Var sitter knappen ....?".

Andra resultat är att de systemutvecklare som deltar i den kooperativa utvärderingen blir mer medvetna om vilka egenskaper och färdigheter deras användare har. Detta kan bidra till att nästa version av systemet blir mer anpassad till användaren och därigenom mer användbar. Om videoinspelningar gjorts kan samma erfarenheter delges till flera systemutvecklare och beslutsfattande chefer. Filminspelningar bör dock inte visas utanför designgruppen utan användarnas tillstånd.

Metoden ger snabba och bra resultat, men man bör vara medveten om att användning vid kooperativ utvärdering. skiljer sig från normal användning. I normala fall brukar inte användaren tänka högt, ha möjlighet att ställa frågor, lösa uppgifter som någon annan valt ut, observeras eller videoinspelas. Resultatet kan påverkas både i negativ eller positiv riktning av den kooperativa användningssituationen.

Var kan man läsa mer om kooperativ utvärdering?

- Monk et al (1993): Boken presenterar kooperativ utvärdering ur ett praktiskt perspektiv. I ett kapitel berättas dessutom om hur metoden introducerades på ett amerikanskt datorföretag.

5.3 Enkätundersökningar

Enkäter används i många olika sammanhang för att samla in personliga åsikter och uppfattningar hos människor. SIFO och TEMO är exempel på organisationer som regelbundet genomför enkätundersökningar i olika sammanhang och på olika människogrupper. Eftersom ett datorsystems användbarhet är ett subjektivt begrepp som är olika för olika människor, för olika arbetsuppgifter och i olika miljöer (jfr sid 5 i denna rapport), kan det vara mycket relevant att undersöka vad systemets användargrupp anser om det aktuella systemet. Uppfattningarna hos användargruppen kan peka på användbarhetsproblem hos datorsystemet.

I enkäten kan frågor ställas som rör alla de aspekter som normalt förknippas med användbarhet (jfr Nielsens definition på sid 4 i denna rapport). Till skillnad från de andra metoderna i denna rapport studeras systemets användbarhet indirekt genom att systemets användargrupp får tala om hur *de* uppfattar systemet. Eftersom den som svarar kan vara anonym kan även känsliga frågor ställas Exempel på typer av frågor som kan ställas är:

- Uppfattar användargruppen systemet som *lätt att lära*? Vad är svårt/lätt?
- Kan de arbeta *effektivt* med systemet när man väl har lärt sig att använda det? Varför/varför ej?
- Är det lätt att *komma ihåg* hur systemet ska användas efter en tids uppehåll? Om ja, vilka delar är svårast?
- Hur ofta *förekommer fel* med systemet? Förekommer fel som inte kan repareras? Hur ofta?
- Är det *roligt* att arbeta med systemet? Varför/varför inte?

Det finns även färdiga enkäter för att testa användbarheten hos datorsystem. Inom ramen för ESPRIT-projektet MUSiC, Measuring the Usability of Software in Context, har det datorstödda enkätverktyget SUMI, Software Usability Measurement Inventory, tagits fram. Det innehåller en enkät som undersöker användbarheten hos datorsystem. En SUMI-undersökning resulterar i att systemet poängsätts på en skala mellan 0-100. På så sätt kan det t ex jämföras med andra system som undersökts med samma enkät. SUMI och andra enkäter kan användas på prototyper av nya system eller på färdiga system.

Så gör man en enkätundersökning

1. *Frågor formuleras*: Tänk igenom vad syftet är med enkätundersökningen, och vilken typ av information den ska generera. Det är vettigt att redan nu fundera över hur enkätens data ska analyseras i slutändan. För varje fråga bör man ha klart för sig varför frågan behövs. Formulera konkreta frågor som rör det du vill undersöka, t ex utgående från exemplen ovan. Ställ frågor som belyser det du vill undersöka från olika synvinklar. Tänk på att språket bör vara neutralt laddat och så klart och tydligt som möjligt.
2. *Sätt samman frågorna till en enkät*: Frågorna som utformas bör struktureras på ett naturligt sätt för den som ska fylla i enkäten. För att öka svarsfrekvensen bör enkäten bör vara "lagom" lång, ungefär två A4-sidor, och inte ta för lång tid att fylla i.
3. *Pilottesta och finjustera enkäten:* Innan enkäten skickas ut till användargruppen är det viktigt att kontrollera att frågorna inte kan missförstås och att din enkät inte tar för lång tid att fylla i. Låt därför några "pilot"-personer besvara enkäten för att testa frågorna och enkätens omfattning.
4. *Enkäten skickas ut och besvaras av målgruppen:* Fördelen med enkäter är att en stor målgrupp kan undersökas, kanske hela användargruppen. Traditionellt har postenkäter varit dominerande men Internet erbjuder nya möjligheter att genomföra enkätundersökningar. För att öka svarsfrekvensen bör man underlätta för svarsgruppen, t ex genom att dela ut "priser" till de första fem svaren eller genom att vid postenkäter bifoga frankerade kuvert.

5. *Svaren sammanställs och analyseras:* Ur enkätsvaren kan en mängd slutsatser dras om hur målgruppen uppfattar och använder datorsystemet. Det är viktigt att inte betrakta alla svar som sanningar. De flesta människor har en tendens att svara enligt det som är socialt accepterat, d v s svara som "man bör svara". Man bör även vara försiktig med att dra alltför långtgående slutsatser från svaren på enstaka frågor. Basera hellre slutsatser på de sammanlagda tendenser som ett större material ger (Wärneryd m fl, 1993).

Resultat från enkätundersökningar

Enkätundersökningar av ett datorsystems användargrupp kan generera mycket värdefull information för den som vill hitta användbarhetsproblem. Alla aspekter av användbarhetsbegreppet kan undersökas, men man bör vara försiktig med att tolka svaren bokstavligt. Många användare har en tendens att svara som man "borde" göra, och det kan därför vara bra att komplettera enkätundersökningar med observationer av användarna.

Nielsen (sid 209, 1993) refererar till en studie (gjord av forskarna Root och Draper) som visar att användare ger bättre svar om de nyligen har använt systemet. Därför kan enkäter med fördel kombineras med t ex kooperativ utvärdering. En metod som kombinerar enkäter med andra sätt att hitta användbarhetsproblem är pluralistisk användbarhetsgenomgång, som beskrivits tidigare i denna rapport.

Ett annat exempel på hur enkäter kan kombineras med andra metoder är s k "pre-post"-undersökningar. Vid sådana undersökningar får användarna fylla i samma enkät både före och efter att de t ex använder ett datorsystem. På så sätt kan man se hur attityder och uppfattningar om systemet förändras över tiden. Ovan nämnda studie genomfördes med denna metodik.

Var kan man läsa mer om enkätundersökningar?

- SUMI: För mer information kan man kontakta Human Factors Research Group, University College Cork, Ireland. Tel: +353 21 276871. Fax: +353 21 270439.
- Wärneryd m fl (1993): innehåller praktiska tips för hur man konstruerar enkäter. Kapitel 4 – (frågornas innehåll, struktur och ordning, 40 sidor), kapitel 5 (om utformningen av svarsalternativ, 20 sidor)och kapitel 6 (om att studera attityder, 22 sidor) är troligtvis de mest intressanta.
- Earthy et al (1995), ger på fyra sidor en kort beskrivning av hur man kan använda enkäter för att utvärdera användbarheten hos datorsystem.

6 Slutord

Denna rapport har redogjort för ett ramverk av de metoder som finns för att identifiera användbarhetsproblem hos datorsystem. Dock förekommer dessa metoder ännu inte i tillräckligt stor utsträckning vid systemutveckling idag (Katzeff & Svärd, 1995) vilket avspeglar sig i systemen.

Svårigheten ligger inte i att det finns för lite kunskaper eller för få metoder att tillämpa, utan att kunskaperna innehas av forskare och inte i tillräckligt stor utsträckning av personer som producerar och utvecklar datorsystem (Katzeff, 1994). Syftet med denna rapport har varit att förmedla kunskaper och metoder för hur man kan identifiera användbarhetsproblem hos datorsystem.

De flesta av metoderna i rapporten är mycket enkla att lära sig och kräver varken special-utrustning, speciella kunskaper eller mycket tid. Tveka därför inte med att komma igång med användbarhetsarbetet i ditt utvecklingsprojekt. Det bästa sättet att försäkra sig om att datorsystem är användbara är att kontinuerligt kontrollera systemets användbarhet under systemutvecklingen. Metoderna i denna rapport kan då vara lämpliga att använda.

7 Referenser

DIX, A., FINLAY, J., ABOWD, G. & BEALE, R. 1993. *Human-Computer Interaction.* Prentice Hall International Ltd. ISBN 0-13-437211-5.

EARTHY, B., BRIGHT, C. & WHEELER, P. 1995. *Evaluation handbook.* Version 1.0. Lloyd's Register. Document INT.WP.EV1.0026.tr.1.0.LR.JVE.

GOULD, J. & LEWIS, C. 1983. DESIGNING FOR USABILITY: Key Principles and What Designers Think. *Proceedings CHI'83 Conference on Human Factors in Computing Systems.* ACM, New York, s. 50-83.

JEFFRIES, R. J., MILLER, J. R., WHARTON, C. & UYEDA, K. M. 1991. User interface evaluation in the real world: A comparison of four techniques. *Proceedings ACM CHI'91 Conference.* New Orleans, LA, April 28-May 2, p. 119-124.

KATZEFF, C. 1994. Tillämpningsklyftan i MDI – ett hinder i utvecklingen av användbara informationssystem. *SISU-dokument nr 21.*

KATZEFF, C. & SVÄRD, P-O. 1995. Användbarhet i praktiken – en enkätstudie. *SISU-publikation 95:20.* http://www.sisu.se/rapporter/95-20/sammanf.html.

KATZEFF, C. & SVÄRD, P-O. 1996. In Search of Key Factors for Usability Maturity. *Adjunct Proceedings of the BCS HCI'96 Conference.* London 20-23 August, s. 94-97.

KOOL, P. & WINGSTEDT, U. 1992. Standarder för grafiska användargränssnitt. *SISU-rapport nr 19.*

LÖVGREN, J. 1993. *Human-Computer Interaction – what every system developer should know.* Studentlitteratur. ISBN 91-44-39651-1.

MONK, A., WRIGHT, P., HABER, J. & DAVENPORT, L. 1993. *Improving your Human-Computer Interface – a practical technique.* Prentice Hall. ISBN 0-13-010034-X.

NIELSEN, J. 1993. *Usability Engineering.* Academic Press Ltd. ISBN 0-12-518405-0

NIELSEN, J. & MACK, R. L. 1994. *Usability Inspection Methods.* John Wiley & Sons Inc. ISBN 0-471-01877-5.

NORMAN, D. A. 1988. *The Psychology of Everyday Things.* Basic Books. ISBN 0-465-06709-3.

NORDQVIST, T. 1996. Computer Supported Evaluation of Guideline and Styleguide Complience. *STIMDI'96 – Användbara system: Perspektiv, Verklighet, Visioner,* s 45-60.

POLTROCK, S. E. & GRUDIN, J. 1994. Organizational obstacles to interface design and development: Two participant observer studies. *ACM Transactions on Computer-Human Interaction,* 1, 1, 52-80.

PREECE, J., ROGERS, Y., SHARP, H., BENYON, D., HOLLAND, S. & CAREY, T. 1994. *Human-Computer Interaction.* Addison-Wesley Publishing Company ISBN 0-201-62769-8.

PRESSMAN, R. S. 1994. *Software Engineering: a practitioner's approach (european edition).* McGraw-Hill International (UK) Ltd. ISBN 0-07-707936-1.

WÄRNERYD, B. M. FL.1993. *Att fråga – om frågekonstruktion vid intervjuundersökningar och postenkäter.* SCB förlag. ISBN 91-618-0382-0.