Høgskolen i Molde

Inntasting i nettbank

Om brukergrensesnitt. Data fra et laboratorieeksperiment

Arbeidsnotat 2008:5 (Høgskolen i Molde)

Professor Kai A. Olsen, 05.03.08

Sammendrag

I juni 2006 tastet en nettbankkunde inn et feilaktig kontonummer. 500.000 kroner gikk til en gal konto, til en uføretrygdet person som brukte opp nesten 400.000 kroner av pengene på spill. Siden banken ikke ville ta ansvar for tapet går kunden og Forbrukerrådet til rettssak. Undertegnede er bedt av Forbrukerrådet om å gi en faglig uttalelse om det aktuelle brukergrensesnittet i nettbanken.

Rapporten gir en kort innføring i fagfeltet menneske-maskin interaksjon (brukergrensesnitt, Human Computer Interaction) og beskriver et forskningsprosjekt der vi har studert inntastingsfeil i nettbanksystemer - med fokus på det systemet som kunden benyttet.

Et sentralt krav til moderne brukergrensesnitt, som et Internett-basert banksystem, er at systemet så langt som mulig skal detektere feil gjort av brukeren. Det banksystemet som den aktuelle kunden benyttet registrerte bare de første elleve siffersom blir tastet inn i kontonummerfeltet. Dvs. at feiltastinger som medførte et for langt kontonummer ikke ble detektert.

Forskningsprosjektet, der en fylte ut nesten 1800 giroer, viser at det å kun benytte de første elleve siffer vil gi det riktige kontonummeret i 2 av 3 tilfeller der det ble tastet mer enn elleve siffer. Sjekk på kontrollsifferet vil ta 90% av feilene i de resterende tilfellene. Men 0,2% av de 1800 giroene (3 giroer i testen) vil slippe gjennom. Dersom man isteden gir feilmelding når kontonummeret er for langt viser testen at alle feil gjort i kontonummeret vil bli stoppet (vi har her ikke tatt med situasjonen der brukeren taster et annet kontonummer fra datasettet i testen istedenfor det som skulle benyttes for den aktuelle giroen).

En hyppig tastefeil er at brukeren har tastet et ekstra siffer i en sekvens av like siffer, f.eks. å taste fire 5-tall istedenfor tre. I testen skjedde dette i 18 tilfeller for kontonummer, 9 for KID. I ingen av disse 27 tilfellene der det ble satt inn et ekstra siffer klarte testbrukeren likevel å komme i mål med riktig lengde på kontonummeret. Et ekstra siffer i en sekvens betyr, med stor sannsynlighet, at tallet blir for langt. Eller, sagt på en annen måte, sannsynligheten for å gjøre to feil (et ekstra siffer i sekvens og å glemme at annet siffer) er meget liten. Testen viser dermed, med stor sannsynlighet, at den aktuelle bankkunden har tastet 12 siffer, altså et ugyldig kontonummer.

I testen studerer vi også operasjonen å ”godkjenne” en transaksjon. Ideen er at banksystemet viser alle data om transaksjonen og ber kunden godkjenne. Testen viser at denne operasjonen er proforma i de fleste tilfeller, i og med at testbrukerne har ”godkjent” en rekke transaksjoner med galt kontonummer.

Innhold

1. Forord

Notatet beskriver et prosjekt for å studere inntasting i nettbank. Bakgrunnen for prosjektet er en tastefeil som medførte at 500.000 kroner gikk til gal konto. Bankkunden skulle overføre dette beløpet til sin datter, men tastet et 5-tall for mye i kontonummeret. Web-grensesnittet til banken aksepterte kun 11 siffer, overskytende siffer ble ikke registrert. Uheldigvis resulterte tastefeilen i et godkjent kontonummer, et kontonummer som eksisterte, og der innehaveren var en person uten formue eller inntekt. Han som mottok beløpet brukte opp nesten 400.000 kroner på spill i løpet av noen dager. Den resterende del ble konfiskert av politiet og er nå returnert til vår uheldige kunde.

Banken nektet å dekke kundens tap, og saken kom opp i Bankklagenemda. Der tapte kunden. Flertallet tok for gitt at kunden hadde gjort en feil, og at det ikke var lover som begrenset hennes tap. Sammen med Forbrukerrådet går hun nå til sak mot banken. Undertegnede ble bedt om å være ekspertvitne for saksøker.

Det ble gjort forsøk på å finne litteratur om tastefeil i internasjonale vitenskapelige kilder. Men dette har vært vanskelig å finne. Det ble utført noe forskning på 60- og 70-tallet, men vi har ikke funnet relevante artikler for de siste 20 årene. Selv om tastaturet er det samme, har de tekniske forutsetningene for inntasting endret seg siden 70-tallet. Den gang var punching på hullkort vanlig, i dag foregår inntasting i skjema som vises på en skjerm. Den gang var det stort sett profesjonelle som tastet (jfr. ”punchedamer”), i dag er det ofte den vanlige forbruker. Ut fra dette, og ut fra at vi ønsket å studere spesielle aspekter ved inntasting, laget vi vår egen undersøkelse.

For undersøkelsen ble det utviklet en banksimulator, mest mulig lik de Web-grensesnittene som bankene benytter. I alt 69 studenter og elever fra videregående skole deltok i testen. Hver av disse ble bedt om å taste inn 30 giroer fra et ferdiglaget datasett. Til sammen 1778 giroer ble utfylt under testen. Disse danner grunnlaget for analysen i denne rapporten.

Rapporten starter med en faglig presentasjon av feltet *brukergrensesnitt*. Dernest beskrives banksimulatoren og opplegg for undersøkelsen. Til slutt gis en analyse av resultatene.

Prosjektet er finansiert av Høgskolen i Molde og Norges Forskningsråd over programmet for ”Små driftsmidler”.

2. Brukergrensesnitt

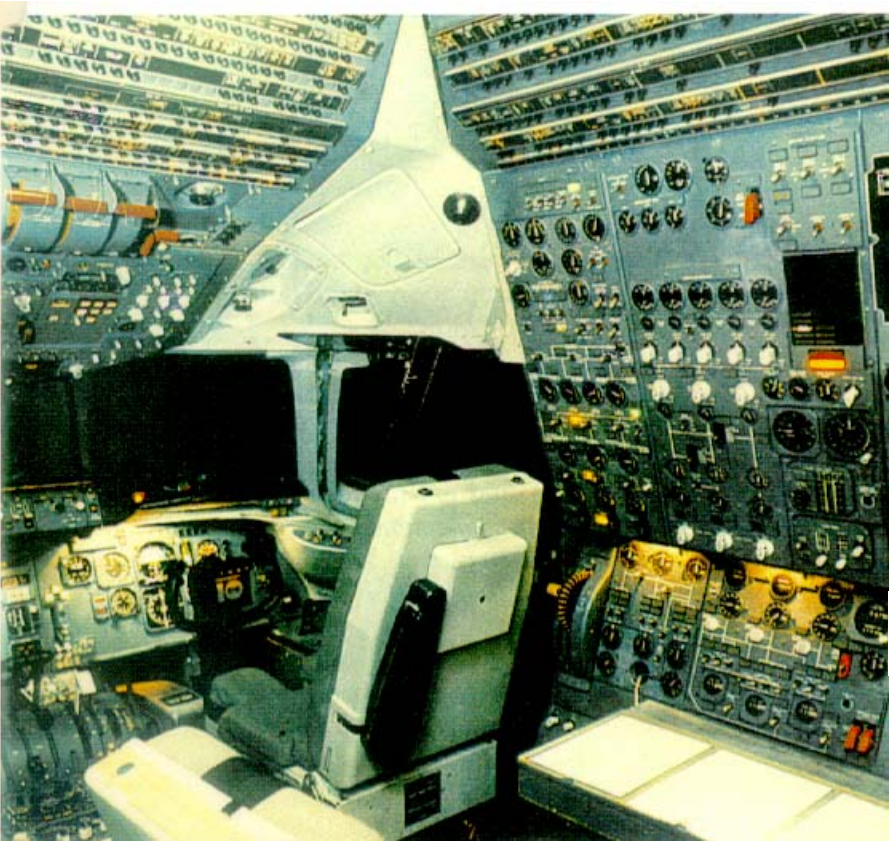
Grensesnittet mellom menneske og maskin kalles brukergrensesnitt, ”user interface” på engelsk. De Web-skjemaene vi benytter for å bestille flybilletter eller for å gjøre banktjenester er typiske brukergrensesnitt. Vi kan også si at en giro på papir har et brukergrensesnitt. På datamaskinen vil typiske elementer i brukergrensesnittet være skjema på skjermen, felt for inntasting, ledetekster og knapper vi kan trykke på. Til hjelp har vi vanligvis mus og tastatur.

Brukergrensesnittet vil hjelpe kunden ved å gi beskjeder . Behov for inndata struktureres ved å presentere ryddige skjema for inntasting der ledetekster forteller hva en vil ha i hvert felt. I tillegg vil gode brukergrensesnitt kontrollere det brukeren har tastet inn, og gi feilmeldinger og advarsler dersom ikke alt ser ut til å stemme. Ideen er at vi ser på menneske og maskin som et helhetlig system som vi ønsker skal fungere best mulig.

2.1. Menneske-maskin kommunikasjon

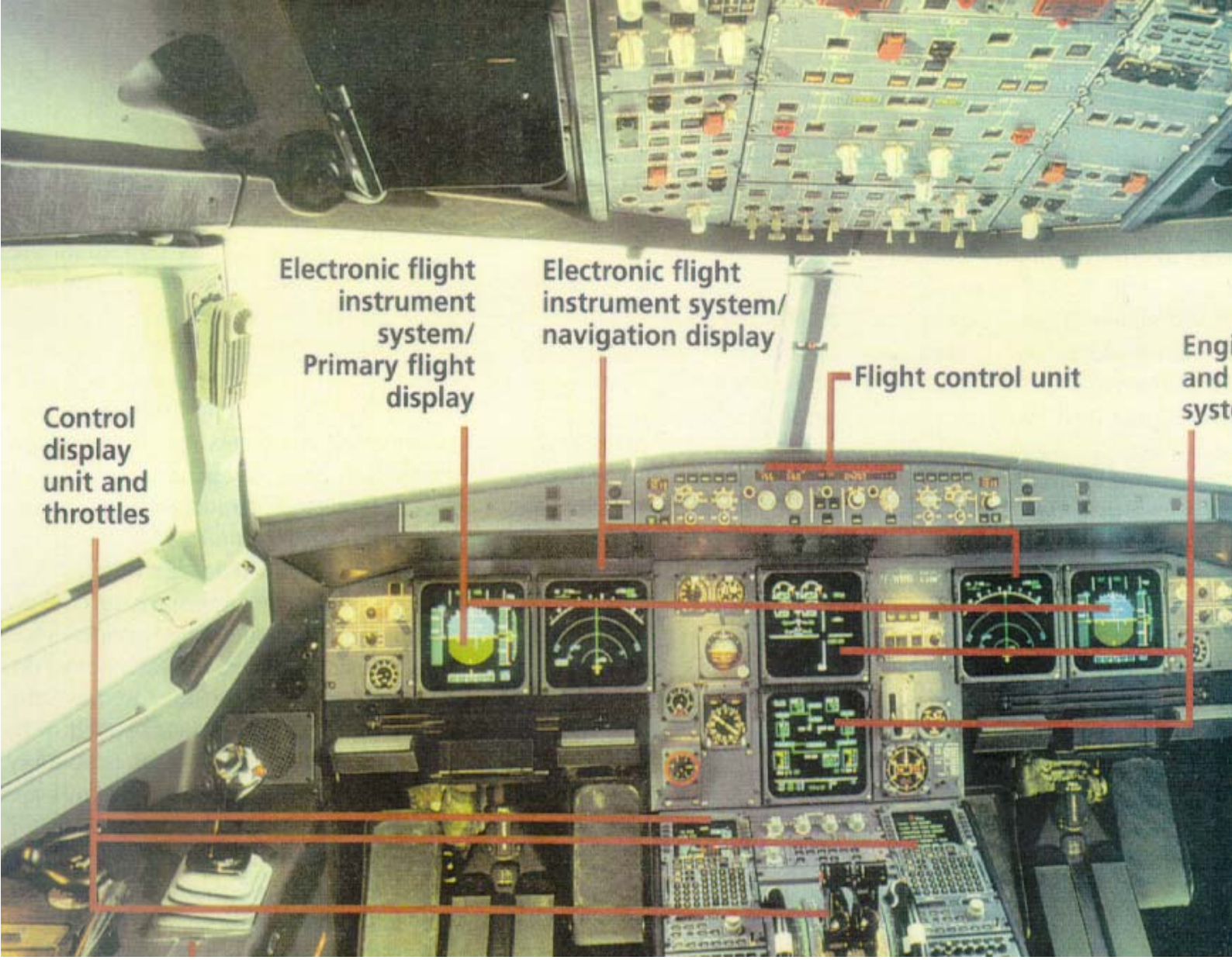
Brukergrensesnitt har vært studert lenge før datamaskinen kom. Med de første industrimaskinene kom spørsmålet om hvor en skulle plassere instrumenter, knapper og spaker for at de skulle være effektive og sikre i bruk. Hvordan skal vi lage kontrollpanelet for styringsrommet i en industribedrift slik at operatørene har full oversikt til enhver tid? I jagerfly har vi spesielle krav. Piloten må kunne se framover og samtidig ha oversikt over de viktigste instrumentene, ofte i en tidskritisk og stresset situasjon. Brukergrensesnittet er også viktig i mer overordnede sammenhenger. Hvordan skal vi beskrive et jernbanenett, som T-banenettet i London, slik at passasjerene finner fram til riktig perrong og riktig tog? Studier i menneske-maskin kommunikasjon og ”usability” har derfor vært viktige forskningsfelt i over hundre år.

Med datamaskinen har vi fått mulighet til å la mennesket styre enda mer komplekse prosesser. Ofte skjer dette i samarbeid med dataprogrammet, kanskje ved at mennesket tar overordnede avgjørelser og datamaskinen tar seg av detaljene. Et eksempel er tekstbehandling, der vi tar oss av mening og utforming, mens tekstbehandleren holder orden på bokstaver og layout. I banksystemet velger vi dato, beløp og kontonummer når vi skal betale en regning. Datamaskinens jobb er å kontrollere inntastingen, og å stå for selve overføringen. En spesiell egenskap med datamaskinen er at vi kan lage *dynamiske* grensesnitt, dvs. grensesnitt som endrer seg underveis i operasjonen.



Figur 1. Fra cockpit i en DC-10.

Der en før var begrenset av mekaniske koplinger kan en nå ha virtuelle kontrolltaster, kanskje symbolisert med knapper på dataskjermen, som en kan plassere akkurat slik en ønsker. Inntil for få år siden hadde alle instrumentene og knappene fast plassering i flycockpiten (se Figur 1). En forsøkte da å sette de viktigste instrumentene og knappene foran flygeren.



Figur 2. Fra cockpit i en moderne Airbus A320, knapper og instrumenter er erstattet av dataskjermer.

Problemet med det var at hva som var viktig endret seg med situasjonen. Før take-off var motordata særdeles viktig, vel oppe i luften var navigasjonsdata sentrale.

Løsningen i dag er å bruke dataskjermer istedenfor fysiske instrumenter. Da kan innholdet i skjermene tilpasses til den situasjonen flyet er i. I tillegg har vi fått langt bedre muligheter til å *kontrollere* det brukeren gjør. En moderne Airbus, for eksempel, har styringssystemer som hindrer flygeren i å gjøre feil. Dette har gitt enda sikrere fly (Figur 2).

Men slike dynamiske brukergrensesnitt setter store krav til at de er utformet riktig. Derfor bruker flyfabrikkene ”usability” eksperter på å utvikle systemene. Ofte setter en sammen grupper av IT spesialister, psykologer og designere for å utvikle brukergrensesnitt som er enkle i bruk, og som minimaliserer feil. Testing er en sentral del av dette arbeidet. Bak dette står erkjennelsen av at en ikke vet nok til å lage det ideelle systemet i første omgang, selv om en bruker dyktige eksperter til å utforme grensesnittene. Apple Computer var noen av de første som bygget opp egne testlaboratorier for brukerstudier. Ideen var å finne feil før produktene ble sluppet ut på markedet. Apple regnet den gang med at de hadde hundre dollar avkastning på hver dollar som ble investert i slike studier. Mye av suksessen til Apple ligger nettopp i brukervennlige produkter, som Mac, iPod og iPhone.

Den første fase av testingen utføres vanligvis av utviklerne. Hensikten her er å plukke ut tekniske feil fra systemet. I neste fase kan systemet testes i laboratoriet, gjerne med innleide testpersoner som tar systemet gjennom en rekke operasjoner. Det er i denne fasen at typiske brukerproblemer (”usability” feil) plukkes ut. Etter at systemet er kommet i drift er en sentral oppgave å logge alle feilsituasjoner m.m.. Dette kan igjen gi data som kan brukes for å forbedre systemet.

Dessverre er det slik at mange bedrifter nøyer seg med å kun utføre en teknisk test av systemet. Systemene blir da satt i drift med store svakheter, også med feil som burde vært unngått med et minimum av faglig ekspertise. Et eksempel i dag er SAS som opererer med referansenummer for flybilletter som inneholder både 0 og O. I enkelte fonttyper kan disse være meget like, og det er ikke lett for kunden å se at det nest siste tegnet her er bokstaven O og ikke sifferet 0: XBJ3O2. Klager en over problemet til personalet på flyplassen får en til svar at mange kunder har dette problemet. Det er forståelig, men unødvendig. Helt siden datamaskinens barndom har en vært klar over at en skulle unngå å bruke 0 og O i slike tilfeldige koder (Holmes, 1975).

Et annet eksempel er www.posten.no der det er svært vanskelig å finne opplysninger om portotakster, selv for helt vanlige brev. I en undersøkelse som vi har gjort tok det studenter 10-15 minutter å få svar på dette fra Postens lite oversiktlige Web-sider. For ti år siden kunne vi kalt dette barnesykdommer, men i dag burde vi ha tilstrekkelig faglig kunnskap til å unngå slike feil og mangler.

Utvikling av brukergrensesnitt et særdeles viktig fagfelt i dag. Et søk på nettbokhandelen Amazon gir oss mer enn 50 lærebøker innen fagfeltet ”user interface design/human computer interaction”. Vi finner i tillegg en rekke vitenskapelige tidsskrift og konferanser dedikert til dette området.

2.2. Krav til brukergrensesnitt

Sentrale krav til brukergrensesnitt er:

1. Grensesnittet skal understøtte brukeren i å utføre oppgaven.
2. Grensesnittet skal vise tilstanden til systemet, slik at brukeren til enhver tid har oversikt over situasjonen.
3. Kommandoer fra brukeren, som tastetrykk og inntasting, skal reduseres til et minimum og foregå mest mulig effektivt.
4. Systemet skal detektere alle feil som er mulig å oppfange.
5. Systemet skal redusere konsekvensen av feil.

Gjennom 1) skal vi forsikre oss om at brukergrensesnittet er laget slik at det understøtter brukeren i å gjøre oppgaven, det være seg en banktransaksjon eller bestilling av flybilletter. Med 2) skal vi forhindre feil. Ideen er at brukeren vil gjøre det riktige når han eller hun har oversikt over systemet. Gjennom 3) vil vi redusere inntasting, og dermed redusere muligheten for tastefeil. Om brukeren gjør feil skal systemet forsøke å detektere disse (4). Det kan f.eks. være å kontrollere at der bare er siffer i tallfelt, at personnummer har 11 siffer og at de to siste (kontrollsifrene) stemmer. Skjer det feil som systemet ikke oppdager skal systemet gjøre alt for å redusere konsekvensene av feil (5). Det kan for eksempel være å tilby en ”undo” mulighet, å lagre tidligere versjoner og å ta sikkerhetskopi av viktige data. Om det da skjer en teknisk feil, eller om brukeren sletter feil fil, vil det være mulig å finne tilbake til en kopi av filen.

2.3. Kjente feil i brukergrensesnitt

I 1979 oppstod en delvis kjernefysisk nedsmelting i atomkraftverket Three Mile Island utenfor byen Harrisburg i USA. Det var bare i aller siste liten at operatørene klarte å få kontroll over situasjonen. Det som kunne blitt en katastrofe på linje med Tsjernobyl resulterte bare i et mindre radioaktivt utslipp. Men dette var ulykken som ikke skulle kunne skje. Hendelsen ga amerikanske myndigheter et alvorlig støkk. Regelverket for å drive kjernekraftverk ble nå så strengt, at det stoppet for videre utvikling av slike kraftverk i USA i tyve år.

Som vanlig var det flere faktorer som var årsak til nestenulykken. Et vesentlig skritt på vei mot katastrofen var at operatørene misforstod kontrollpanelet. De trodde at et lys på panelet varslet at en ventil var stengt. I virkeligheten varslet lyset bare at en hadde trykket på knappen for å stenge ventilen. Andre data på panelet indikerte imidlertid at ventilen fortsatt var åpen, men operatørene klarte å bortforklare disse (Walker, 2004). Dette skjer ofte. Har en først funnet en årsak vil en lett forsøke å holde på denne begrunnelsen, kanskje lenge etter at det burde være åpenbart at en har tatt feil.

I 1988, under en spent situasjon i den Persiske Gulf, skjøt den amerikanske marine ned et Iransk passasjerfly, en Airbus A320, med 290 mennesker ombord. Krigsskipet som skjøt ut raketten trodde at de ble angrepet av et jagerfly. Feilen skyldes en kombinasjon av en stresset situasjon for operatørene og et forvirrende brukergrensesnitt i det avanserte krigsskipet. Hadde alle data vært presentert på en oversiktlig måte ville de ha sett at dette åpenbart var et passasjerfly i en internasjonal flyrute, og ikke et angripende Iransk jagerfly (Craig et al, 2004; US Department of Defense, 1988).

Jakob Nielsen, en internasjonal ekspert på brukergrensesnitt, har vist i en rapport ”Medical Usability: How to Kill Patients Through Bad Design ”(2005) hvordan leger lett kan gi pasienter gal dose medisiner ved bruk av medisinske informasjonssystemer. Han har analysert 22 tilfeller av feilmedisinering og funnet ut at årsaken var klassiske brukergrensesnitt-problemer. To av disse er:

- Misledende standardverdier. Systemets standardverdier var pakningsstørrelser, ikke vanlige doseringer som legene trodde. Avvik mellom brukernes forventninger til systemet og hvordan systemet virkelig opererer er en hyppig årsak til feil.
- Når doseringen ble endret, beholdt systemet både gammel og ny dose. Tilsvarende feil kan skje i banksystemer, der en bruker taster samme giro to ganger (se kapittel 3.3).

Lignende problemer er også funnet i laboratorietester av brukergrensesnitt for medisineringssystemer (se f.eks. Kushniruk et al, 2004).

2.4. Feil og glipp

Sentralt i all utvikling av brukergrensesnitt står ideen om at mennesker gjør feil. Mens det er vanskelig å beskytte seg mot bevisst misbruk av de som har adgang til systemet, skal et brukergrensesnitt beskytte mot inntastingsfeil m.m.. Brukeren kan gjøre en glipp, derfor må systemet ha rutiner for å forhindre at dette skjer eller for å redusere skadevirkningen.

Inntastingsfeil er hyppige. I en undersøkelse fra 1980 viser Card et al at inntil en fjerdedel av en eksperts tid kan gå med til å rette feil. MacKenzie et al (2002) rapporterer at rettetasten er den nest mest brukte tasten på tastaturet (etter blank tasten, men før bokstaven ”e”). I tekstbehandleren er ”undo” knappen like sentral. Gjør vi feil kan vi trykke på denne for å komme tilbake til situasjonen vi var i før feilen ble gjort. Dette er et godt eksempel på hvordan en kan redusere konsekvensen av feil.

Kravet til å beskytte brukeren mot egne feil går igjen også i andre sammenhenger. Som jegere har vi lært at våpenet aldri skal peke mot et menneske. Likevel har alle geværer sikring. Konsekvensene av en feil er så store at en har funnet det nødvendig med ekstra sikkerhet. I en bil med automatgir må vi ha foten på bremsen før vi starter bilen, har vi ikke foten på bremsen vil bilen ikke starte.

I et moderne datasystem vil programkode for å detektere og håndtere feil være en viktig del av programmet. I enkelte systemer kan så mye som 30% av koden være for feilhåndtering. Ideen er å oppdage feil så tidlig som mulig, slik at konsekvensene blir redusert. Brukeren skal få gode feilmeldinger, slik at minst mulig tid går tapt for å rette opp feilen. Men viktigst er det å lage brukergrensesnittet slik at feil unngås i første omgang.

Adresselisten på mobiltelefonen er et eksempel. Gjennom denne kan vi velge ut personen vi skal ringe ut fra navn, og vi unngår å taste inn nummeret hver gang. Da unngår vi mange feiloppringninger. Moderne banksystem lager lister over mottakere. Da kan vi hente mottakeren fra listen neste gang og dermed redusere risikoen for feiltasting. Fordelen med å bruke navn er at dette er meningsfylt for brukeren, i motsetning til telefonnummer, kontonummer m.m.

2.5. Feilmeldinger

Der systemet oppdager feil i en transaksjon gis en melding. Meldingen skal gjøre det klart for brukeren hva som er feil, og hvordan feilen skal rettes opp. Teknisk sett er det enkelt å lage systemer som gir gode meldinger. Etter at feilmeldingen er gitt vil systemet vanligvis la brukeren få mulighet til å rette opp feilen. Etter dette utføres en ny kontroll. Først når systemet ikke lengre kan finne feil vil transaksjonen bli gjennomført.

Ved å telle opp forekomsten av hver feilmelding vil systemutvikleren lett få en ide om hvor brukerne feiler. Slike data kan danne grunnlag for å lage et bedre brukergrensesnitt for neste versjon av systemet.

2.6. Advarsler og bekreftelser

For risikofylte operasjoner gir de fleste systemer advarsler. Skal du slette en fil i Windows vil systemet gi deg en advarsel. Det nye Vista systemet til Microsoft vil gi deg et utall advarsler når du skal installere et sertifikat fra nettbanken. Problemet er at en vanlig datamaskinbruker har små muligheter til å vurdere risikoen i hvert enkelt tilfelle. Han eller hun vet imidlertid at om de ikke svarer Ja på alle spørsmålene, så får de heller ikke kontakt med nettbanken. Etter hvert kan vi bli immune. Vi svarer Ja overalt, kanskje også i de tilfellene der vi burde sagt Nei.

Psykologen Donald Norman gir dette eksempelet i sin bok, Psychology of Everyday Things (1988),

> USER: Remove file ”My-most-important-work”
>
> COMPUTER: Are you certain that you wish to remove the file ”My-most-important-work”?
>
> USER: Yes.
>
> COMPUTER: Are you certain?
>
> USER: YES
>
> COMPUTER: The file ”My-most-important-work” has been removed.
>
> USER: Oops, damn.

I følge Norman bekrefter vi operasjonen, ikke meningen med denne. Moralen må være at om vi gir for mange advarsler blir disse ignorert. Her burde systemet ha gitt ytterligere data, f.eks. om filstørrelse, forside på dokumentet, m.m. Da ville nok brukeren ha oppdaget feilen.

2.7. Testing

Arbeidet med å utvikle gode brukergrensesnitt er komplekst. Løsningen er på langt nær bare teknologisk. Her kommer også forståelse for hvordan vi mennesker tenker og handler inn i bildet. Ofte setter en sammen team av IT-folk, psykologer, ergonomer og grafiske designere når viktige brukergrensesnitt skal utformes. Selv da er det ikke mulig å få alt rett i første omgang. Løsningen er å teste.

For et system som skal benyttes av vanlige brukere er dette enkelt, siden hvem som helst kan benyttes for å teste systemet. Vi kan hyre inn folk fra gaten, eller f.eks. bruke studenter. Vanligvis vil testpersonene få et sett av oppgaver mot systemet. De kan få beskjed om å snakke høyt (”La meg se, nå skal jeg finne en knapp for å bekrefte transaksjonen. Ja, her ser jeg den…”), de kan bli overvåket av eksperter og hele sesjonen kan bli tatt opp på video. Ofte benyttes en spesiell variant av dataprogrammet, en variant som også registrer hva bruken gjør. Det er denne metoden som er brukt i testen beskrevet senere i dette notatet.

Hensikten med testen er å oppdage svakheter i brukergrensesnittet, steder der brukeren velger feil, trykker feil tast, der det tar lang tid å forstå hva som skal gjøres, osv. Dataene fra testen benyttes så til å forbedre systemet.

Seriøse bedrifter vil også overvåke systemet etter at det er tatt i bruk. I motsetning til laboratorietesten, der vi kanskje hadde færre enn hundre testpersoner, vil vi nå kunne få store datamengder – kanskje fra millioner av transaksjoner. Da kan nye svakheter komme til syne. For eksempel kan det vise seg at fargeblinde får problemer, eller at en meget spesiell feilsituasjon kan ha store konsekvenser. Det kan også tenkes at det som engang var bra, ikke er i overensstemmelse med kunnskap og erfaring til nye brukere. De kan ha bakgrunn fra Google, spillprogrammer, m.m., og forventer dermed at systemet skal oppføre seg akkurat som disse andre systemene. Med kontinuerlig overvåkning av systemet vil vi fort oppdage slike situasjoner. Systemet kan da forbedres slik at feilen bare får konsekvenser for et lite antall brukere.

3. Feildetektering ved utfylling av en giro

Vi skal her se på hva som kan gjøres automatisk for å forhindre og detektere feil i et nettbankssystem i forbindelse med utfylling av en giro. Vi skal først studere de mest grunnleggende og enkle testene, og til slutt se på mer utfyllende metoder.

3.1. Grunnleggende tester

I en bankgiro oppgir vi forfallsdato, et KID-nummer eller en melding, beløp og kontonummer. I utgangspunktet, uten tilgang til andre data, kan datasystemet for brukergrensesnittet kontrollere at:

1. Dato er på et gyldig format, for eksempel 29.12.07
2. Forfallsdato ikke er passert.
3. KID nr. kun inneholder siffer
4. Kontrollsifferet for KID-nummeret er riktig
5. Beløpet kun inneholder siffer (og desimalkomma om hele beløpet gis i ett felt)
6. Ørebeløpet er på to siffer
7. Kontonummeret bare inneholder siffer
8. Kontonummeret ikke er lengre enn 11 siffer
9. Kontonummeret ikke er kortere enn 11 siffer (ordinære bankkontonr)
10. Kontrollsifferet for kontonummeret er riktig

Det er en smal sak å kontrollere disse ti punktene. Vi vil derfor som et minimum forvente at et godt brukergrensesnitt foretar alle disse. I tillegg vil det i ettertid blir gjort en rekke andre kontroller, som f.eks. at til kontoen er i bruk, at kredittgrenser ikke blir overskredet, etc. Siden disse kontrollene er relatert til bankenes forretningsprosesser og ikke til brukergrensesnittet behandles de ikke her. Vi skal konsentrere oss om de ti grunnleggende testene over. Det er dette en vil forvente at brukergrensesnittet håndterer.

Siden vi er mest interessert i kontonummeret her skal vi i detalj vise hvordan testene for å kontrollere dette kan utføres. La oss anta at det inntastede kontonummeret er lagret i variabelen *Knr*. En variabel er et felt i datamaskinen som kan inneholde en verdi, her et kontonummer med et antall siffer. Mens brukeren ser et felt for kontonummer på skjermen, kan datamaskinen ha en tilsvarende variabel i programmet der verdien av det inntastede nummeret er lagret.

I eksempelet skal vi bruke vårt eget programmeringsspråk, men dette kan direkte erstattes med et vanlig språk (som Java eller Visual Basic) uten at programkoden blir mer kompleks.

Sjekk nr. 7 kan utføres med denne koden:

```
HVIS IKKE Numerisk (KNr) SÅ
    GiMelding (”Kontonummeret kan kun inneholde siffer”)
```

Numerisk er her en operasjon som sjekker om variabelen i parentes, her kontonummeret, bare inneholder siffer. Alle programmeringsspråk har en slik operasjon. GiMelding skriver ut en melding på skjermen. Alle programmeringsspråk har en slik operasjon.

Sjekk nr. 8 kan utføres med denne koden:

```
HVIS lengde (KNr) ER STØRRE ENN 11 SÅ
    GiMelding  (”Kontonummeret er for langt”)
```

Lengde er her en operasjon som gir lengde på variabelen oppgitt i parentes, i antall siffer. Alle programmeringsspråk har en slik operasjon.

Mange dataprogrammer innholder hundre tusener av programlinjer. Til sammen vil en kunne utføre alle disse 10 testene på under 100 programlinjer. Arbeidet med å legge inn testene er derfor helt bagatellmessig. Men som vi skal se er de viktige for å forhindre feil.

3.2. Modulo 11-testen

Ideen med et kontrollsiffer i kontonummeret er at vi regner ut dette når nummeret opprettes. Samme regnestykke benyttes så for å sjekke om kontonummeret er riktig hver gang dette brukes. I kontrollen av kontonummeret benyttes en Modulo 11-test. Dvs. vi summerer hvert av de ti (første) sifrene i kontonummeret, dividerer med 11, kaster resultatet men beholder resten. Kontrollsifferet 0 og 10 benyttes ikke. Da står vi tilbake med et siffer mellom 1 og 9 som legges bakerst og blir da det ellevte sifferet i kontonummeret.

Dersom vi hadde foretatt en ren summasjon av sifrene i kontonummeret ville Modulo 11-testen ikke ha oppdaget feil der vi bytter om rekkefølgen på to siffer. Derfor multipliserer vi hvert siffer med en faktor før summasjon. Formelen er beskrevet i et notat fra Bankenes Standardiseringskontor (2003).

Sjekk nr. 10 (test av kontrollsiffer) kan da utføres med denne koden:

> **HVIS** (11– (Knr (1) * 5 + Knr (2) * 4 + Knr (3) * 3 +Knr (4) * 2 + Knr (5) * 7 + Knr (6) * 6 + Knr (7) * 5 + Knr (8) * 4 + Knr (9) * 3 + Knr (10) * 2) Mod 11) **FORSKJELLIG FRA** Knr (11) **SÅ**
> GiMelding (”Ugyldig kontonummer”)
>
> Knr(1) gir her første siffer i kontonummeret, Knr(2) det andre, osv.

Det forventes ikke at leseren skal forstå detaljene her, men som vi ser kan hele sjekken settes opp med en programsetning.

Testen vil ta alle forekomster av *en* tastefeil, uansett hvor vi taster feil og hvilket siffer vi taster. Den vil også ta alle ombyttinger av to etterfølgende siffer. Gjør vi imidlertid to tastefeil vil testen slippe igjennom hele 7% av alle feiltastingene. Dette varierer litt med hvor vi gjør feilen, om vi f.eks. har feil i siffer 9 og 10, som vår bankkunde hadde, er sannsynligheten 4% for at kontonummeret likevel godkjennes. Gjør vi fem feil øker dette til ca. 9%. På helt tilfeldige kontonummer vil testen slippe igjennom noe over 8% av feiltastede kontonummer.[1]

3.3. Utfyllende metoder

Med ytterligere data og noe mer programkode kan vi gjøre langt mer for å forhindre og detektere feil enn det som ble antydet over. Vi skal her kort presentere noen metoder som kan benyttes.

Med et mottakerregister må vi kun taste kontonummer første gang det benyttes. Senere vil vi kunne betale giroer på navn, dvs. på samme måte som vi kan ringe fra telefonboken i mobilen. Fordelen med dette er at mens navn er meningsfylt, er kontonummeret sjelden det. Et mottakerregister hindrer ikke bare feil, men det gir oss mulighetene til en mer effektiv utfylling.

Systemet kan gi advarsler ved overføring av store beløp. Det kan gjøres første gang vi overfører til et nytt kontonummer og – enda smartere – hver gang vi bryter vårt vanlige mønster. For eksempel, om vi vanligvis fyller ut en giro til kraftselskapet på mellom to og tre tusen kroner, kan systemet gi oss en advarsel om beløpet avviker mye fra dette.

---

[1] Dette siste tallet kan vi beregne matematisk. Resten etter divisjon med 11 (modulo 11) vil være et tall mellom 0 og 10. Siden det første og det siste er ugyldig står vi tilbake med 9 av 11 muligheter, dvs. 82% av alle kontonummer. Sjekk på kontrollsifferet vil plukke ut 90% av disse, altså slipper vi igjennom 8.2%.

Likeens kan systemet gi advarsel om samme giro blir utfylt to ganger. Det kan tenkes at kunden har to like regninger som skal betales, men sannsynligheten er stor for at dette er en feil.

I de fleste systemer vil vi få opp navn om vi taster et kontonummer som eies av en bedrift. Dette skjer ikke for privatpersoner. Et argument er at dette vil stride mot bankenes taushetsplikt (Finansnæringens Hovedorganisasjon, 2006) og personvernhensyn. Men hadde systemet kunnet gi fornavn, eventuelt et alias, ville vi ha en god kontroll - uten å komme i konflikt med personvernet. Et alternativ er å be kunden taste både navn og nummer, og så sjekke at disse stemmer overens. Problemet er at navn kan skrives på mange måter. Dog kan en god mønstergjenkjenningsmetode brukes for å beregne sannsynligheten for likhet mellom det kunden har skrevet og det som er registrert i systemet. Da kan det være uinteressant om vi skriver ”Hansen” eller ”Hanseen”, systemet vil likevel finne likheten med det navnet (kanskje ”Hanssen”) som er registrert for kontoeier . En slik løsning vil imidlertid kreve at det er en viss overensstemmelse mellom navnet som kontoen er registrert på og det som benyttes ved betaling. Det er ofte ikke tilfelle i dag, men kunne gjerne vært satt som et krav for å få sikrere betalingsformidling.

De første Internettbankene kom midt på 90-tallet. Siden det har vi fått noe bedre verktøy for å utvikle slike systemer. Ikke minst har vi fått erfaring med systemene. Likevel, det vi har tilgjengelig i dag, må kalles første-generasjonsløsninger. De har enkel funksjonalitet og relative primitive brukergrensesnitt. Etter hvert vil disse forbedres, og bli mer brukervennlige. Noen av de mulighetene som er beskrevet her er allerede innført, andre må vi håpe kommer etter hvert. Det vil gi oss mer effektive og sikrere systemer.

Skal dette skje er det viktig at bankene samler inn data om hvordan brukergrensesnittet fungerer. Dette kan skje gjennom laboratorietester, slik som for testen beskrevet under. I tillegg er det viktig å samle inn data om systemet i bruk. Dette er meget enkelt å få til. Samme datateknologi som benyttes for å lage nettbanksystemet kan benyttes for å logge det brukerne gjør. Med enkle analyseprogrammer, tilsvarende det som er beskrevet under, kan en få data om hvilke deler av systemet som ikke fungerer mot brukerne.

4. Inntastingstest

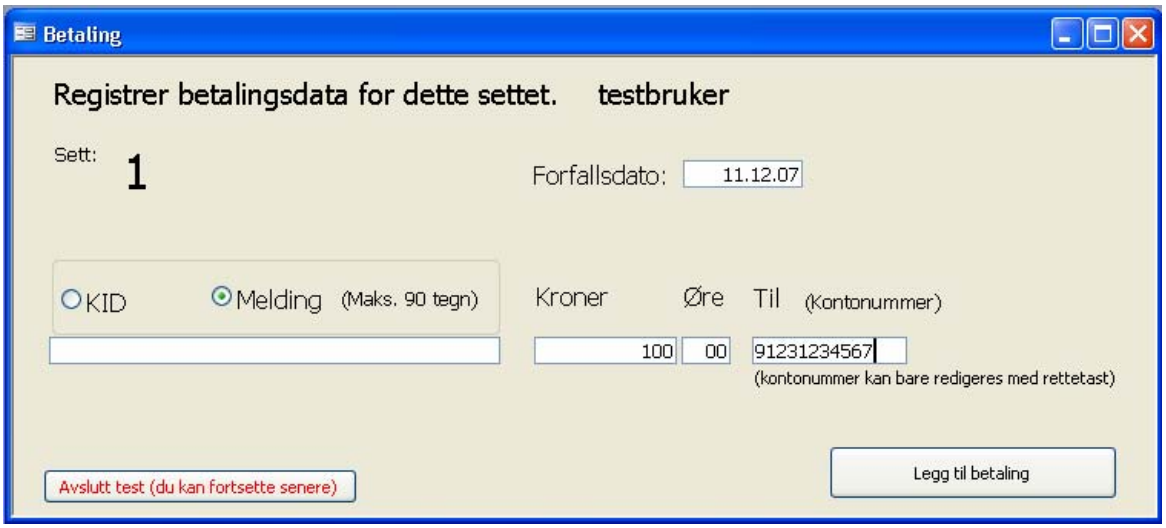
Av mangel på vitenskapelige artikler og andre data om inntasting i nettbank ble det gjennomført et forskningsprosjekt ved Høgskolen i Molde, høsten 2007. For prosjektet ble det utviklet en ”banksimulator”. Det ble lagt vekt på å etterligne det systemet som den aktuelle kunden benyttet i juni 2006. Simulatoren fungerer på samme måte som brukergrensesnittet i en vanlig nettbank. Men i tillegg logger det alle data i kulissene. Simulatoren viser bare de elleve første siffer i kontonummeret på skjermen, men, i motsetning til det virkelige banksystemet, registrerer simulatoren mer enn elleve siffer. Med denne løsningen får vi muligheter til å analysere feil der brukeren har tastet et for langt kontonummer. Siden hensikten med simulatoren er å analysere tastefeil gir denne ingen feilmeldinger.

Studenter fra Høgskolen og elever fra videregående skoler i Molde ble benyttet som testpersoner. Hver tester ble bedt om å taste inn inntil tretti giroer fra et ferdig datasett. Hver giro hadde en forfallsdato, et KID nummer eller en melding, et beløp og et kontonummer.

I det følgende er selve simulatoren presentert. Deretter gis en detaljert beskrivelse av testopplegget, før vi diskuterer resultatene.

5. Banksimulator

Banksimulatoren er laget mest mulig likt Web-grensesnittet for girobetaling i Internettbanker. Det består av to skjermbilder, ett for registrering av betaling og ett for godkjenning.



Figur 3. Skjema for å registrere en giro i simulatoren..

Skjemaet for betaling er vist i Figur 3. I testversjonen tar kontonummeret imot mer enn elleve siffer, men viser kun de elleve første. For brukeren vil dette derfor operere likt med Web-løsninger som bare aksepterer elleve siffer i kontonummerfeltet.

Figur 4. Skjema for å godkjenne en giro i simulatoren..

Av praktiske grunner er godkjenningsdelen lagt til et eget skjema, slik som vist i Figur 4. Her kan brukeren enten trykke ”Godkjenn betaling” eller ”Rett opp”. Den siste tasten har samme funksjon som å trykke redigeringssymbolet (✎) i det virkelige Web-grensesnittet.

Simulatoren er utviklet i programsystemet Microsoft Access og kjører på et client-server system (systemet kjører på en sentral server, hver bruker er tilkoplet via en PC– klient). Siden vi skal benytte grensesnittet til å studere feiltastinger gir systemet ingen feilmeldinger. I praksis vil systemet fungere meget likt til bankenes Internett løsninger.

6. Testopplegg

Opplegget for testen, testpersoner, registrering og analysemetoder presenteres under.

6.1. Testpersoner

Tre store grupper på ca. 20 personer hver ble invitert til å delta i testen. Dette er studenter ved Høgskolen i Molde som tar et innføringskurs i Web-design, samt en IT klasse fra Molde Videregående skole og en IT klasse fra Romsdal videregående skole. I tillegg ble andre studenter fra Høgskolen invitert til å delta. Alt i alt deltok 69 personer i testen. De fleste hadde over middels kunnskaper om databehandling. Det har lite å si for testen. Imidlertid behersket alle testpersoner mus og tastatur godt.

6.2. Testdata

Inntasting forgikk ut fra et sett på 30 giroer, testdataene. Disse er vist i appendiks B. Kontonumrene som ble brukt har gyldig kontrollsiffer og er alle på elleve siffer. For KID ble det ikke lagt vekt på å ha gyldig kontrollsiffer, siden dette ikke har interesse for testen. Datasettene er utviklet mht å ha en god bredde av forskjellige giroer, men det er lagt vekt på å ha en rekke kontonummer, KID og beløp med sekvenser av like siffer. Dette fordi vi ønsket spesielt å studere hva som skjer i disse tilfellene.

Siden vi primært ønsker å studere feiltastinger er det viktig med store datamengder. Da de fleste taster riktig, må vi ha et stort antall inntastinger for å få data om feil. I testen tastet en inn i underkant av 1800 giroer. Vi kunne gjerne hatt flere, men de viktigste resultatverdiene stabiliserte seg allerede etter de første 600 utfylte giroene.

6.3. Testopplegg

Hver testperson ble invitert til å taste inn 30 giroer. De studentene som tok testen som en del av innføringskurset fikk tilbud om å gjøre denne en gang til. To studenter gjorde det.

Inntasting skulle skje ut fra et ferdig sett som ble utdelt i laboratoriet, eller som kunne skrives ut fra simulatoren. De fleste som deltok tastet inn alle giroene. Gruppene brukte ca tjue minutter på dette. Testpersonene ble holdt uvitende om hensikten med testen til etter at inntastingen var gjennomført. De som deltok i testen ble med på en trekning av en digital bilderamme til en verdi av to tusen kroner.

6.4. ”Outlayers”

Testpersoner som tastet mye feil, mer enn 25% feil i kontonummer, er holdt utenfor testen. Dette kan være personer som ikke helt forstod hva de skulle gjøre, eller som ikke har vært interessert i å registrere ut fra det oppgitte datasettet. Alt i alt gjaldt dette fem testpersoner. Data fra disse er ikke tatt med i resultatene.

6.5. Analyseprogram

Det er utviklet en egen modul som analyserer de inntastede data. Denne sammenligner det som er inntastet med fasit, det originale datasettet. Systemet teller opp feil av forskjellig kategori, som for eksempel for langt kontonummer, ekstra siffer i sekvens, for kort kontonummer, riktig lengde med galt kontrollsiffer (etter modulo 11 sjekken), etc. Også KID, dato og beløp sjekkes. Analysen er imidlertid konsentrert om feil i kontonummer og KID.

Analysemodulen vil også telle opp antall forsøk som hver person har utført for å få giroen riktig. Det er også beregnet tid brukt på hver giro. Har brukeren gjort flere forsøk er det bare det siste, dvs. det som er godkjent, som benyttes i feilanalysen.

6.6. Endring av kontonummer

For tre av de tretti giroene vil systemet endre det inntastede kontonummeret til ett som ligner. Dette gjøres etter inntasting men før godkjenning. For eksempel kan kontonummer 70581555022 bli endret til 70581555502. Vi ber altså brukeren å godkjenne en giro der vi har endret kontonummeret uten at han eller hun er klar over det. Ideen er å sjekke hvor nøyaktig brukerne er i selve godkjenningsdelen av prosessen.

6.7. Generaliserbarhet

Åpenbart er det forskjeller mellom å taste inn tretti giroer i en test og å bruke en nettbank. I nettbanken taster vi sjelden så mange som tretti giroer på en gang. Blant annet er det i testen mulighet for å blande sammen data fra to giroer, noe som kan skje i virkeligheten, men som er mindre sannsynlig der enn i testen. I et virkelig banksystem er det også reelle penger som overføres. Derfor må vi forvente at den totale feilprosenten er større i testen enn i virkeligheten.

Det vi tester er situasjonen der alle data, inklusive kontonummeret, for en transaksjon tastes inn. Dette er representativt for situasjonen til nye kunder i nettbanken og for de tilfeller der vi har et nytt kontonummer. Kunder som har hatt konto i en nettbank over lengre tid vil imidlertid ofte velge mottaker fra et register. Er kontonummeret i mottakerregisteret korrekt vil det selvfølgelig også bli korrekt for alle påfølgende transaksjoner til denne mottakeren. Testen er derfor ikke representativ for denne situasjonen.

For øvrig skulle testsituasjonen være som i nettbanken. Skjermløsningene og prosessene i simulatoren er tilsvarende som i nettbanken. En forskjell er at simulatoren ikke gir feilmeldinger. De totale feilprosentene vil da selvfølgelig være høyere her. Men vi kan anta at den innbyrdes fordelingen av hvilke feil som skjer er rimelig lik i testen og i et virkelig banksystem (med unntak av feilen der testpersonen taster et kontonummer fra et annet datasett).

I et prosesskrift fra advokaten som representerer Sparebank Nord-Norge oppgis en del feilprosenter for Sparebank 1 Nord-Norge og hele Sparebank 1 Alliansen (Keiserud, 2008). Alliansen hadde 26 209 853 transaksjoner i 2006. Av disse hadde 7 758 (0,03%) for kort kontonummer og 449 814 (1,72%) feil ble detektert av modulo 11-testen. Vi går ut fra at det i totaltallene både inngår transaksjoner som er registrert ut fra et mottakerregister og transaksjoner der kontonummeret som er fylt inn direkte. I kapittel 8 sammenlignes disse tallene med resultatene fra testen.

7. Resultater

En datautskrift av de mest sentrale testresultater er vist i appendiks A. Vi skal gi en kort omtale av resultatene her.

Resultatene ble inndelt etter gruppe (høgskolestudenter, klasse fra Molde videregående, klasse fra Romsdal videregående). Testen viste at Høgskolestudentene hadde lavere feilfrekvens enn gymnaselevene, ca. 4% mot 8%. For øvrig var resultatene like. Vi kontrollerte også om feilraten var større på de siste inntastede giroene (nr 21 til 30) enn på de første (1 til 10), men fant ingen signifikante forskjeller. Ut fra dette har vi valgt å presentere de samlede resultatene fra alle testpersonene.

Av de 1778 giroene hadde 124, dvs. 7%, feil i kontonummer. Av disse 124 skyldes feilen i 29% av tilfellene at kontonummeret var for langt. Brukeren har altså tastet 12 siffer eller mer. I 50% av tilfellene der kontonummeret var for langt skyldes feilen at brukeren har tastet et ekstra siffer i sekvens, der en sekvens er en serie på to eller flere like siffer. For eksempel kan brukeren ha tastet fire 5-tall der det bare skulle være tre. Nå må det igjen nevnes at i datasettet som brukes i testen har mange kontonummer sekvenser av like siffer, dog er alle gyldige kontonummer.

Om vi ser bort fra feilen der testpersonen tastet et annet kontonummer fra testen, var ”for langt kontonummer” den hyppigste feilårsak. Prosentene blir da ”for langt”, 41%; ”for kort”, 35%, og ”galt 11 sifret kontonummer”, 24%. I 68% av tilfellene der kontonummeret ble for kort er årsaken at brukeren har glemt et siffer i en sekvens.

I 64% av tilfellene der brukeren har tastet et for langt kontonummer ville dette blitt riktig om siste siffer var fjernet. Dvs. her ligger feilen i det ekstra, siste sifferet. Merk at grensesnittet ikke viser dette sifferet. Feil i dette vil ikke bli oppdaget av brukerens egen validering. Derfor vil kontonummeret ofte bli rett når det ekstra sifferet er fjernet. Men når banksystemet fjerner siste siffer vil dette også være gal løsning i ca. 36% av tilfellene. Det utgjorde 13 giroer i testen. I disse gjenstående 13 tilfellene ville Modulo 11-testen ha plukket ut 10 giroer, men 3 (0,2%) ville ha sluppet igjennom.

Vi kan ofte vurdere et datasystem med å sammenligne med hva som ville skje i den virkelige verden. La oss anta her at vi skrev et brev til banken og ba dem overføre 500.000 kroner til konto nr. 705815555022. Bankfunksjonæren ser med en gang at nummeret er for langt. Hva gjør hun? Fjerner hun siste siffer og overfører pengene til konto 70581555502, eller sender hun brevet i retur med melding om at kontonummeret er for langt?

Der et galt kontonummer var på elleve siffer hadde testpersonen tastet et annet kontonummer fra datasettet enn det som skulle tastes for akkurat dette settet i 36 tilfeller. Dette er en feil som selvfølgelig forekommer langt oftere i testen, der brukeren taster et stort antall giroer fra en liste, enn det vi ville ha i virkeligheten. Men feiltypen vil også forekomme i virkelige situasjoner. Denne type feil er selvfølgelig ikke mulig å detektere med de grunnleggende metodene. En kan merke seg at Modulo 11-testen her plukket ut alle andre feiltastinger.

I ingen av de tilfellene der kontonummeret var galt, og på 11 siffer, har brukeren tastet et siffer for mye i sekvens. Dvs. kombinasjonen av å taste et siffer for mye og å likevel komme i mål med 11 siffer forekommer ikke i testgrunnlaget. Imidlertid har en bruker klart å utelate et siffer i sekvens og ”kompensert” ved å taste et ekstra siffer på slutten.

Resultatene for inntasting av KID er tilsvarende (her ble det ikke utført noen sjekk på kontrollsiffer). 40% har tastet for lang KID, 1/3 av disse har gjort dette ved å ha et ekstra

siffer i en sekvens. Ingen har tastet et ekstra siffer og likevel kommet i mål med riktig lengde på KID.

Feil i kronebeløp forekom i 2,6% av tilfellene. Siden dette feltet er kortere enn felt for kontonummer og KID må vi forvente lavere feilrate her.

I 178 tilfeller endret systemet det inntastede kontonummeret til ett som ligner. I 97% av tilfellene ble dette ikke oppdaget av brukeren.

8. Sammenligning med bankenes tall

Sammenligner vi med de aggregerte tallene fra Sparebank 1 Alliansen oppgir bankene 1,7 % feil detektert av modulo 11-testen. I vår test detekterte modulo 11-testen feil i 21 giroer på elleve siffer, i tillegg til 10 giroer som ble forkortet til elleve siffer. Dette gir en feilprosent på 1,7 – helt likt tallene fra Sparebank 1 Alliansen[2].

Alliansen hadde 0,03% transaksjoner med for korte kontonummer. Her gir testen en verdi på 1,7%, altså et meget stort avvik. Som nevnt tidligere kan en argumentere med at en i virkelige systemer vil ha lavere feilrate enn i testen, men her har vi samme feilrate på feil funnet av modulo 11-testen, og stor forskjell når det gjelder andel for korte kontonummer. I tillegg er det vanskelig å forklare at brukerne taster feil siffer i 1,7% av tilfellene, men mister et siffer i så lite som 0,03%. Selv med bankens tall alene virker denne store forskjellen underlig. Det hadde derfor vært interessant å få ytterligere detaljer om hvordan bankens tall er framskaffet.

9. Konklusjon

Laboratoriesituasjonen medfører at det sannsynligvis blir gjort flere feil her enn i virkelige Web-baserte banksystem. En viktig årsak er at vi har valgt å ikke gi feilmeldinger i testen. Likevel har vi samme andel modulo 11-feil i testen som det Sparebank 1 Alliansen kan opplyse basert på sine erfaringstall. Uansett bør vi kunne forvente at fordelingen på feiltyper er som i den virkelige verden (med unntak av feilen der testpersonen tastet et kontonummer fra et annet datasett). Testen viser at for 27 giroer ble det tastet et ekstra siffer i sekvens i kontonummer eller KID. I alle disse tilfellene medførte dette at nummeret ble for langt. Altså, blant de 1778 giroene forekom det ikke et eneste tilfelle der brukeren både tastet et ekstra siffer i en sekvens og glemte et siffer et annet sted.

Vi kan derfor konkludere med at det å gjøre to feil:

- **å taste ett siffer for mye på ett sted**
- **og et siffer for lite på et annet sted**

er lite sannsynlig.

I 173 av 178 tilfeller har brukerne oversett at systemet har endret på et kontonummer. Kun i 3% av tilfellene har brukerne fanget opp feilen i godkjenningsfasen.

Vi kan derfor konkludere med at godkjenningsfasen i bare liten grad vil fange opp feil der systemet har endret på de inntastede data.

Banken vil fjerne en rekke typiske tastefeil når de omgjør kontonummer med for mange siffer til et elleve sifret tall. Ulempen med dette er at modulo 11 sjekken er så grov at den slipper igjennom flere tilfeller av gale kontonummer (3 tilfeller i vår test). Hadde en registrert mer

[2] Vi savner en detaljering av bankenes tall. For eksempel vet vi ikke hvor stor andel av transaksjonene der kunden har benyttet mottakerregisteret, og som vanligvis ikke vil gi feil i kontonummeret. Er mange av transaksjonene av denne typen kan bankens tall på 1,7% virke høyt.

enn elleve siffer ville antall feilmeldinger øket, men antall feil ville bli redusert. Med å velge en ”forkortningsløsning” har banken prioritert effektivitet (i form av færre feilmeldinger) framfor sikkerhet.

Går vi tilbake til bankkunden som hevder å ha tastet 705815555022 istedenfor 70581555022, der resultatet ble 70581555502, kan vi med stor sannsynlighet fastslå at hun må ha tastet 12 siffer. Når hun da ble bedt om å godkjenne overføringen til konto 70581555502 har hun gjort samme feil som i 97,3% av testtilfellene og godkjent giroen.

Bankene benytter i dag et kontrollsiffer i kontonummeret. Til tross for at det bare er ett kontrollsiffer (personnummeret har to) er Modulo 11-testen som benyttes i stand til å ta alle enkle tastefeil. Derimot får testen problemer når kunden setter inn et ekstra siffer. Fra kundens side er dette bare en feil, men fra datamaskinens side vil alle siffer fra og med dette bli feil. Derfor svikter Modulo 11-testen her. Den går fra 100% effektivitet til 92%. Det er ikke tilstrekkelig til å ta alle feiltastinger som forekommer når vi vet at det ble fylt ut 150 millioner nettbankgiroer i 2006 (Norges Bank).

Gir en feilmelding på for langt (og for kort) kontonummer, og bruker modulo 11 sjekken på alle kontonummer av riktig lengde vil systemet detektere alle enkeltfeil: tastefeil på et siffer, ombytting av to etterfølgende siffer, inntasting av et ekstra siffer, glemme et siffer.

Det kreves kun to programlinjer for å legge inn testen på for langt kontonummer, og det finnes ingen argumentasjon for å utelate dette. I testen der studenter fylte ut 1800 giroer og gjorde 124 feil i kontonummeret ville ingen ha sluppet igjennom en slik enkel feilkontroll.

Bankene har åpenbart innsett dette og i dag gis slike feilmeldinger i alle Internett-banker i Norge. Men bankene har åpenbart latt kundene foreta den endelige uttestingen av systemene, og – fra denne saken – også bedt om at kundene selv bærer ansvaret for de feilene som er gjort i utformingen av grensesnittet.

10. Referanser

Bankenes Standardiseringskontor (2003) Telepayformatet, versjon 2.0.

Card, S.K., Moran, T.P., Newell, A. (1980) The keystroke-level model for user performance time with interactive systems, Communication of the ACM, ACM Press, 23 (7).

Craig, D., Morales, D. Oliver, M. (2004) USS Vincennes Incident; M.I.T. Aeronautics & Astronautics, Spring 2004

Finansnæringens Hovedorganisasjon (2006) Trygghet ved bruk av nettbank – næringens anbefalinger, brev til Kredittilsynet av 18.10.

Holmes, W. N. (1975) Identification Number design, Computer Journal, 18 (2), 102.107.

Keiserud, E. (2008) Prosesskrift til Nord-Troms Tingrett, Advokatfirmaet Hjort DA, 27.02

Kushniruk, A. W., Triola, M. M., Borycki, E. M., Stein, B. , Kannry, J. L. (2004) Technology induced error and usability: The relationship between usability problems and prescription errors when using a handheld application, International Journal of Medical Informatics Volume 74, Issues 7-8

MacKenzie, I.S., Soukoreff, R. W. (2002) Text entry for mobile computing: Models and methods, theory and practice. Human Computer Interaction, Lawrence, Erlbaum Associates, .

Nielsen, J. (2005) Medical Usability: How to Kill Patients Through Bad Design, http://www.useit.com/alertbox/20050411.html

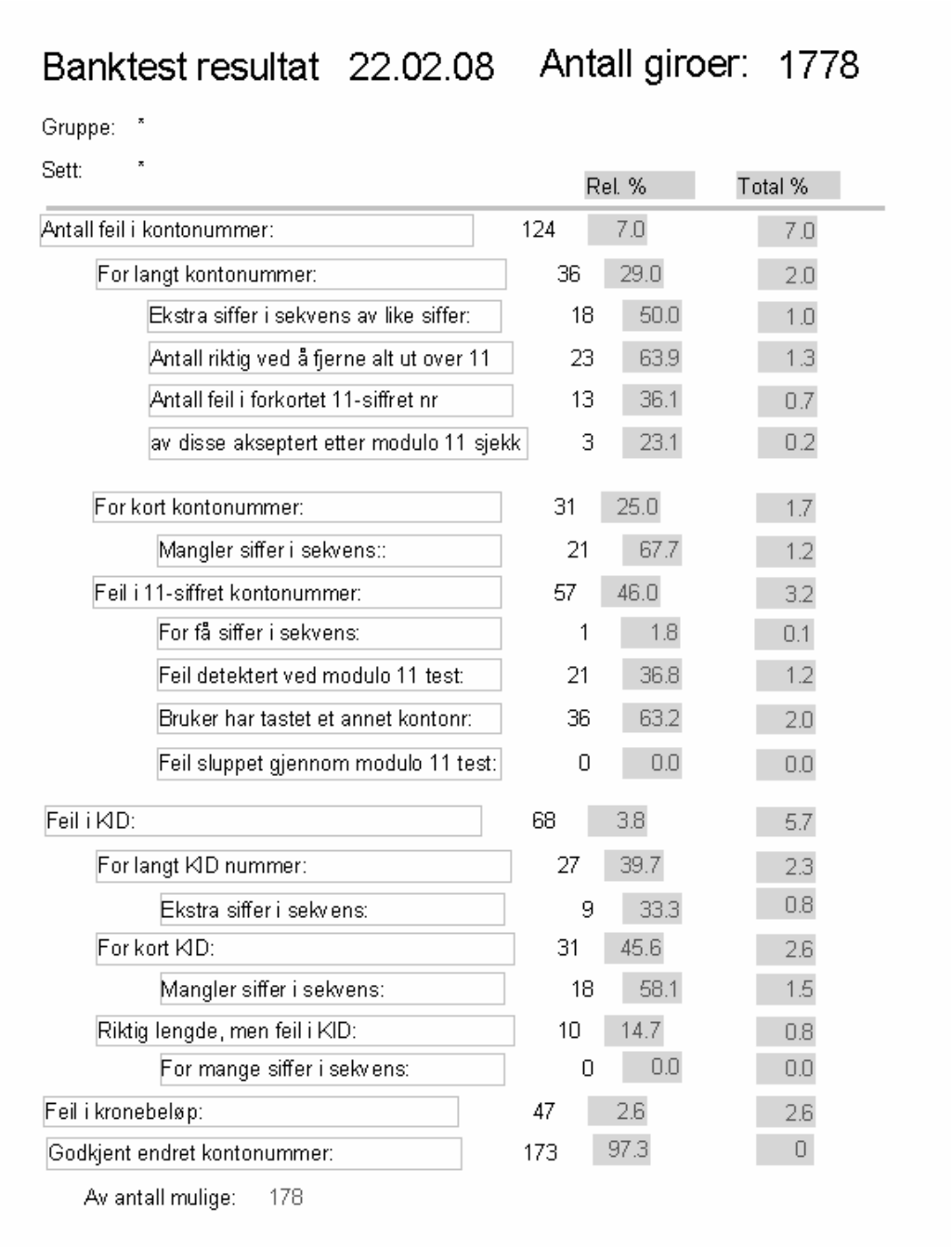
Norges Bank (2006), Årsrapport om betalingssystem

Norman, Donald A. (1988) The Psychology of Everyday Things, Basic Books, New York.

US Department of Defence (1988) Formal Investigation into the Circumstances Surrounding the Downing of Iran Air Flight 655 on 3 July 1988.

Walker, J. Samuel (2004) Three Mile Island: A Nuclear Crisis in Historical Perspective, Berkeley: University of California Press.

Appendiks A – Datautskrift resultater



Banktest resultat 22.02.08 Antall giroer: 1778

Gruppe: *

Sett: *

		Rel. %	Total %
Antall feil i kontonummer:	124	7.0	7.0
For langt kontonummer:	36	29.0	2.0
Ekstra siffer i sekvens av like siffer:	18	50.0	1.0
Antall riktig ved å fjerne alt ut over 11	23	63.9	1.3
Antall feil i forkortet 11-siffret nr	13	36.1	0.7
av disse akseptert etter modulo 11 sjekk	3	23.1	0.2
For kort kontonummer:	31	25.0	1.7
Mangler siffer i sekvens::	21	67.7	1.2
Feil i 11-siffret kontonummer:	57	46.0	3.2
For få siffer i sekvens:	1	1.8	0.1
Feil detektert ved modulo 11 test:	21	36.8	1.2
Bruker har tastet et annet kontonr:	36	63.2	2.0
Feil sluppet gjennom modulo 11 test:	0	0.0	0.0
Feil i KID:	68	3.8	5.7
For langt KID nummer:	27	39.7	2.3
Ekstra siffer i sekvens:	9	33.3	0.8
For kort KID:	31	45.6	2.6
Mangler siffer i sekvens:	18	58.1	1.5
Riktig lengde, men feil i KID:	10	14.7	0.8
For mange siffer i sekvens:	0	0.0	0.0
Feil i kronebeløp:	47	2.6	2.6
Godkjent endret kontonummer:	173	97.3	0

Av antall mulige: 178

Sentrale data fra banktesten er vist over. Prosentverdiene helt til høyre er absolutte, dvs. i forhold til de 1778 giroene. Andre prosenter er relative innen gruppen. bare 2/3 av alle transaksjoner hadde KID, og den totale feilprosenten er justert for dette.

Appendiks B – Datasett for inntasting

1

Forfallsdato: 22.12.07
KID: 100000555566
Beløp: 10000,00
Kontonummer: 70581555022

2

Forfallsdato: 20.11.07
Melding: Her er pengene
Beløp: 570,00
Kontonummer: 97104444435

3

Forfallsdato: 12.11.07
Melding: faktura 6456
Beløp: 1000,50
Kontonummer: 97101123433

4

Forfallsdato: 06.11.07
KID: 340000011111199
Beløp: 565,00
Kontonummer: 90585555028

5

Forfallsdato: 19.11.07
KID: 785555513455557
Beløp: 123,30
Kontonummer: 70201111453

6

Forfallsdato: 03.11.07
Melding: Gratulerer med dagen
Beløp: 200,00
Kontonummer: 34577677117

7

Forfallsdato: 02.11.07
KID: 2323232300022
Beløp: 12500,00
Kontonummer: 23453333116

8

Forfallsdato: 29.11.07
Melding: Takk for lånet
Beløp: 100,00
Kontonummer: 90879989999

9

Forfallsdato: 29.11.07
KID: 0000001110101000
Beløp: 120000,00
Kontonummer: 67903333569

10

Forfallsdato: 29.12.07
KID: 40011409071721011
Beløp: 16,00
Kontonummer: 74500508098

11

Forfallsdato: 29.11.07
KID: 356789
Beløp: 94,67
Kontonummer: 34900000563

12

Forfallsdato: 19.11.07
Melding: Billetter
Beløp: 900,00
Kontonummer: 89911111455

13

Forfallsdato: 19.11.07
KID: 200700111122121212
Beløp: 200,00
Kontonummer: 10922211219

14

Forfallsdato: 19.12.07
Melding: Leilighet B36
Beløp: 1436000,00
Kontonummer: 34000001114

15

Forfallsdato: 02.11.07
KID: 677766767
Beløp: 400,50
Kontonummer: 70585555027

16

Forfallsdato: 17.12.07
KID: 90000011111010101
Beløp: 1298,40
Kontonummer: 88088887802

17

Forfallsdato: 17.12.07
KID: 56756
Beløp: 34,00
Kontonummer: 70785155554

18

Forfallsdato: 17.11.07
KID: 1110010100101
Beløp: 958000,00
Kontonummer: 20200202229

19

Forfallsdato: 15.12.07
Melding: Etter avtale
Beløp: 456,00
Kontonummer: 99099967761

20

Forfallsdato: 05.12.07
KID: 78334760
Beløp: 67,00
Kontonummer: 97500707899

21

Forfallsdato: 09.12.07
KID: 03133000200655
Beløp: 900,00
Kontonummer: 70581555022

22

Forfallsdato: 21.11.07
KID: 011560074500075
Beløp: 3930,55
Kontonummer: 86019502866

23

Forfallsdato: 30.12.07
KID: 00217950001000053
Beløp: 2868,00
Kontonummer: 96502543935

24

Forfallsdato: 12.12.07
KID: 74947611
Beløp: 801,00
Kontonummer: 97600516993

25

Forfallsdato: 22.12.07
Melding: Pegeout 207
Beløp: 183000,00
Kontonummer: 78009919197

26

Forfallsdato: 03.11.07
KID: 3444333343561
Beløp: 90,34
Kontonummer: 12900910914

27

Forfallsdato: 22.12.07
KID: 623456776689
Beløp: 145,66
Kontonummer: 26666444417

28

Forfallsdato: 09.11.07
KID: 1678901
Beløp: 678,00
Kontonummer: 90588876771

29

Forfallsdato: 01.12.07
Melding: Ved
Beløp: 300,00
Kontonummer: 65555510009

30

Forfallsdato: 06.12.07
KID: Siste innbetaling
Beløp: 500000,00
Kontonummer: 70581555022
