Interaksjonsdesign som prosess 1

TDAT2003 Systemutvikling 2 med web-applikasjoner
Institutt for informatikk og e-læring
Anette Wrålsen september 2016

NTNU
Kunnskap for en bedre verden

Innholdsfortegnelse

Interaksjonsdesign som prosess

Som vi har vært inne på tidligere, er interaksjonsdesign en iterativ prosess med fire trinn som gjentas om og om igjen til vi er fornøyd med resultatet:

1. **Definere krav til det som skal lages.** Hvem er brukerne, og hva ønsker de fra produktet? Disse spørsmålene besvares typisk gjennom forskjellige former for datainnsamling og analyse.
2. **Designe alternativer.** Vi lager en konseptuell modell og bruker denne som utgangspunkt for forskjellige fysiske design.
3. **Lage prototyper.** For virkelig å forstå designene vi har gjort, er det ofte nødvendig å lage prototyper.
4. **Evaluering.** Her undersøker vi brukskvaliteten i det vi har laget, basert på kravene fra første fase.

I denne leksjonen skal vi fokusere på de tre første trinnene i denne prosessen. Men først noen ord om *brukerorientert* eller *brukersentrert utvikling*, begreper du fort kan møte på i utviklingssammenheng:

Å involvere brukerne – brukerorientert utvikling

Noe som er essensielt i forbindelse med å designe interaksjoner, er å ha fokus på brukerne underveis i utviklingen. Dette gjøres ofte rett og slett ved å involvere brukere i utviklingsprosessen. Det er spesielt i to sammenhenger dette er viktig – først når vi definerer kravene til systemet, og senere når vi evaluerer det. Ofte kalles utviklingsprosesser der man involverer brukere mye for *brukersentrert utvikling* eller *brukerorientert utvikling*. Det varierer litt hva som legges i disse begrepene, men hovedideen er å ha fokus på brukerne og deres behov gjennom hele utviklingsprosessen.

Det viser seg gang på gang at det lønner seg å bruke tid på å involvere brukere i utviklingsprosessen. Allikevel er det viktig å være bevisst på hvordan og hvor mye man involverer brukerne. Hvis man ikke planlegger denne delen godt, risikerer man å bruke mye tid og ressurser på noe som man får lite igjen for. Det er for eksempel farlig å støtte seg for mye på tilbakemeldinger man får fra enkeltbrukere, uten å se disse i en kontekst. En bruker er nettopp det, *en* bruker. Hvor representativ er denne brukeren? Er det flere som mener det samme? Hvis jeg tar hensyn til det han sier, kan jeg komme til å ignorere behovene til andre viktige brukergrupper? Er det i det hele tatt teknisk mulig innenfor prosjektrammene å ta hensyn til det han ønsker, for ikke å snakke om kost-nytte? Og kan denne brukeren ha andre motiver (f.eks. å beskytte sin egen jobb) som ikke blir klart kommunisert?

Brukerinvolvering er viktig, og tenkes ofte på som en måte å *demokratisere* en utviklingsprosess på. Det å bli involvert kan også gi brukergruppen en mer positiv holdning til den nye teknologien når den skal innføres. Det er altså allikevel viktig å tenke nøye gjennom *hvordan* og *når* i prosessen vi gjør dette. Designerne er ekspertene, og de bør ha klare mål med å involvere brukerne, man bør ha en plan for hvilken type informasjon man ønsker fra dem og hvordan denne skal samles inn. Generell kunnskap om brukskvalitet og interaksjonsdesign er viktig i denne sammenhengen.

Men vi bruker Scrum!

I utviklingsprosjekter er det vanlig å ha egne personer som har spesielt ansvar for, eller utelukkende jobber med utvikling av brukergrensesnittet. Disse følger gjerne en prosess lik den over. Så hvordan integrerer man dette med andre utviklingsmetoder? Vel, dette varierer gjerne fra prosjekt til prosjekt, men generelt, på grunn av sin iterative natur, er det spesielt velegnet til å

integrere med iterative og smidige metoder som f.eks. Scrum. Mange av ideene i manifestet for smidig programvareutvikling[12] harmonerer veldig godt med interaksjonsdesign og brukerorientert utvikling.

Mange lykkes godt med å integrere interaksjonsdesign med smidige metoder som Scrum, men det er ikke noen faglig enighet ennå om hva som er den "riktige" måten å gjøre dette på. Det betyr at i praksis finner man gjerne en løsning som passer det enkelte prosjektet, der man typisk planlegger utviklingen slik at man får iterasjoner som følger Scrum-sprintene. Uansett hvordan man velger å gjøre dette, er det viktig med god kontakt og mye kommunikasjon mellom de som driver med utviklingen av brukergrensesnittet og de andre utviklerne. Dette kan for eksempel gjøres ved å la en eller flere av designerne være en del av Scrum-teamet. Det kan også gjøres ved å la hele Scrum-teamet være aktive i utviklingen av brukergrensesnittet på en eller annen måte.

Å definere krav til produktet

Når et utviklingsprosjekt starter opp, er det med utgangspunkt i en idé om et nytt system som enten skal erstatte et eksisterende system, eller som er helt nytt og innovativt. Man har altså allerede en idé om hva systemet skal gjøre, og hvorfor det er et poeng å lage et slikt system i det hele tatt. Dette er oppdragsgiverens idè om produktet og brukergrensesnittet, og er bakteppet for kravdefinisjonen vi nå skal gjøre.

Det å definere kravene til et system er mye mer enn en liste over hva initiativtakerne ønsker ut av systemet. For å kunne gjøre det må vi så godt som alltid finne ut av hvem brukerne av systemet er, hva deres mål med å bruke systemet er, og vi bør også finne en måte å beskrive ideelle brukersamhandlinger med systemet på.

I tillegg er det viktig å formulere krav til brukskvalitet, universell utforming og brukeropplevelsen knyttet til systemet. Disse kravene formulerer vi etterhvert som vi forstår brukerne godt nok til å bli konkrete rundt dette. Nå skal vi se litt mer spesifikt på hva vi bør finne ut av og hvordan vi kan gå fram for å gjøre det.

Hvem er brukerne?

Dette er et spørsmål som fort kan tenkes på som litt underordnet spørsmålet om hva brukerne ønsker ut av systemet. Men ofte så vet ikke brukere helt hva de ønsker ut av et nytt system, de er preget av vaner og tidligere erfaringer, og det kan være vanskelig å bryte ut dette og tenke på *mulighetene*. Dette er heldigvis vi som designere eksperter på!

Det betyr at å finne ut av hvem brukerne er, er et viktig steg i å forstå dem. Hvis vi vet hvem de er, hva de gjør i dag og hva de ønsker å oppnå generelt i situasjonene der de skal bruke det nye systemet (og ikke hva de selv tror de ønsker at et nytt system skal gjøre for dem), så har vi et mye bedre utgangspunkt for å bruke ekspertisen vår som designere og lage alternativer for dem. Det gir oss også et utgangspunkt for å *evaluere* og velge mellom alternativer.

Hva vi trenger å vite om brukerne kan variere fra system til system. Når det gjelder kartlegging av hvem brukeren *er,* er vi typisk interessert i ting som

- ✓ Alder
- ✓ Kjønn
- ✓ Kulturell bakgrunn
- ✓ Relevant utdanning og erfaringer (inkludert erfaringer med relevant teknologi)
- ✓ Fysiske trekk som er relevante

[12] http://agilemanifesto.org/iso/no/

- ✓ Psykologiske trekk som er relevante

Ofte vil man definere flere brukergrupper som har forskjellig utgangspunkt og forskjellige interesser knyttet til bruken av et system.

Personaer

En vanlig teknikk i denne sammenhengen, er å bruke *personaer*. Det vil si at designerne rett og slett finner på en liten samling av personer som representerer de mest typiske *brukermotivasjonene*. Med dette mener jeg at man ikke fokuserer på hvem en typisk bruker *er*, men hva en typisk bruker *ønsker å bruke systemet til*. Man beskriver hvem personaen er, hva som motiverer ham/henne og hvordan personaen i dag går fram for å løse problemene systemet er tenkt å hjelpe til med.

Eksempel

For nettstedet Stereoking (en stereoanleggbutikk) bruker vi personaen Tore. Han er 29 år, bilmekaniker og interessert i stereoanlegg. Han ønsker å bruke nettstedet til å handle, og ellers surfe rundt og se på anlegg og priser. Han er vant til å bruke internett og har en kraftig pc, men er ingen ekspert og gir fort opp hvis noe er for knotete og vanskelig. Han bruker Windows (nyeste versjon), surfer med oppdatert versjon av Firefox og har alle de «vanlige» plugin'ene installert (og muligens et virus eller fem).

Dette kan være en veldig nyttig teknikk for å få noe mer konkret enn en ofte ganske vag beskrivelse av en målgruppe å forholde seg til. Det kan også være motiverende å se for seg en faktisk person som man skal designe for, samtidig som det gjør det lettere for designeren å visualisere hvordan konkrete personer bruker systemet som er under utvikling. Det er ikke uvanlig å bruke mye tid på å utvikle realistiske personaer, med forhistorie og godt beskrevne fysiske og psykologiske trekk. Disse personaene kan man så ta fram senere i prosessen og bruke dem til å undersøke viktige *brukerscenarioer*.

Datainnsamling

Når vi skal finne ut informasjon om brukerne er det mange måter å gå fram på. Nøyaktig hva vi bør gjøre vil variere fra system til system, og er en avgjørelse designerne må ta. Disse valgene bør være basert på *eksplisitte mål for hva man ønsker å finne ut*. Disse målene vil både påvirke hvilke metoder du velger, hvor du samler data hen og hvordan du går fram (f.eks. hvilke spørsmål du stiller i et spørreskjema eller hvem du velger å intervjue). Ikke still spørsmål etter innfallsmetoden, tenk nøye gjennom hvem du spør og hva du spør om, slik at du vet at du får maksimal ut av det (det er ofte vanskelig nok i seg selv å få folk til å svare på spørreskjemaer eller delta i intervjuer).

Her er noen metoder for datainnsamling:

Spørreskjemaer

Spørreskjemaer har noen klare fordeler: Man kan samle mye informasjon fra mange brukere på en enkel måte, og informasjonen man får kan være både kvantitativ og kvalitativ alt etter hvilken type spørsmål man stiller. Ulempen er at det generelt er umulig å si noe om hvor allenngyldige resultatene er, med mindre en veldig høy prosentandel av brukerne svarer. Kanskje vil brukere som generelt er ekstra engasjert i tematikken, for eksempel de som helst vil beholde ting slik de er i dag eller som aller helst vil ha nye løsninger ha større sannsynlighet for å svare? Man må altså

være forsiktig med å trekke slutninger fra denne typen undersøkelser, med mindre man har god grunn til å tro at et representativt utvalg av brukere har svart.

Det å svare på en spørreundersøkelse er noe en bruker gjør for å hjelpe deg. Det betyr at du bør ta hensyn til det når du lager skjemaet. Hvis brukeren ikke får noe annet igjen for det enn en mulighet til å si sin mening, bør du passe på at skjemaet er godt laget og lett å fylle ut, og ikke tar for lang tid. Hvis du tilbyr brukeren noe for å delta (for eksempel å være med i en loddtrekning med premier) kan du tillate deg litt mer krevende spørsmål. Men da bør du ha i bakhodet at brukeren tjente noe på å delta når du analyserer resultatene, slik at du kan ha fått flere useriøse svar enn ellers. Dessuten kan det at man må legge igjen epostadresse for å delta i en konkurranse føre til at brukere som ønsker å være anonyme lar være å svare av frykt for at svarene deres vil bli koblet til epostadressen.

Her er noen retningslinjer for å lage gode spørreskjemaer:

- ✓ Det er vanlig å starte et spørreskjema med å be om demografisk informasjon (kjønn, alder, yrke og lignende). Det er nyttig, og kan i tillegg si noe om hvor representativt utvalget av brukere som har svart er. Merk at med mindre du har en veldig høy svarprosent kan du ikke konkludere bare ut fra dette at utvalget er representativt (det kan for eksempel fortsatt være hovedsakelig de mest misfornøyde brukerne som har svart), men hvis det er tydelig at nesten ingen representanter for viktige brukergrupper har svart så vet du i hvert fall at utvalget ikke er representativt.
- ✓ Når du ber om demografisk informasjon, be om så lite som mulig og kun om informasjon som faktisk er nyttig for deg. Det kan sikkert være artig å vite mye om brukerne sine, men mange brukere er skeptiske til å gi fra seg personlig informasjon som de ikke opplever som relevant, og kan finne på å la være å svare på grunn av dette. Hvorfor i all verden vil karishjemmelagdesaft.no vite hvilket parti jeg stemmer på?
- ✓ Jobb med spørsmålene til du er sikker på at de er utvetydige og bortimot umulige å misforstå. Uklarheter får brukerne til å gi opp, og gir deg dårlige data (fordi noen brukere kan komme til å svare på noe annet enn det du mente).
- ✓ Jobb også med rekkefølgen og eventuell gruppering av spørsmålene, for å gjøre det så logisk og enkelt som mulig for brukerne å svare. Noen ganger kan rekkefølgen på spørsmål få noe å si for svarene, og dårlig organiserte skjemaer kan være forvirrende.
- ✓ Ha klare retningslinjer for brukerne – må alle spørsmål besvares? Hvis ikke, er det allikevel noen som er obligatoriske for å få gå videre i undersøkelsen? Kan man krysse av for flere enn ett alternativ? Denne informasjonen bør gis på en tydelig og vennlig måte.
- ✓ Tenk gjennom hvordan du lager alternativer. Pass på at alle muligheter dekkes, og tillat som hovedregel svar som «vet ikke» og «ingen av delene». Unngå også kategorier som overlapper, slik som alder 0-20, 20-30, 30-50 osv. Hva skal en som er 20 år krysse av for da? Sin følelsesmessige alder?
- ✓ Hvis du bruker skalaer (type 1-6, svært misfornøyd til svært fornøyd osv.) så tenk gjennom hvor mange trinn skalaen har. Jo flere trinn, jo mer må brukerne tenke for å plassere seg selv på skalaen. Vurder også om skalaen skal ha et midtpunkt eller ikke. Dropper du midtpunktet kan ikke brukeren stille seg nøytral!

Her kan vi jo nevne en liten avstemning på vg.no sine nettsider, der spørsmålet var «Deltar du i avstemninger på nettet?». 11,31% av deltakerne svarte «nei» (http://www.vg.no/poll/?id=1120).

Mer at det imidlertid sjelden vil være nok i seg selv å bare

kjøre spørreundersøkelser, vi trenger vanligvis mer dybdeforståelse og bør dermed også bruke andre metoder!

Å snakke med brukere – intervjuer og fokusgrupper

Intervjuer med brukere er en god måte å samle inn dybdeinformasjon, både for å forstå mer om brukernes motivasjoner og ønsker, og hvordan de faktisk bruker dagens teknologi. Intervjuer deles gjerne opp etter hvor *strukturerte* de er.

- ✓ Et **strukturert** intervju er fullstendig planlagt på forhånd, og intervjueren har en liste med spørsmål som skal stilles og skriver rett og slett ned svarene. Fordelen er at alle intervjuobjekter blir stilt nøyaktig de samme svarene, og vi får gjerne et datamateriale som er lett å behandle og sammenligne på tvers av intervjuer.
- ✓ Et **semi-strukturert** intervju har typisk noen spørsmål som utgangspunkt, men intervjueren kan stille oppfølgingsspørsmål ut fra hvilke svar hun får, og kan til en viss grad la seg lede ut på avstikkere av intervjuobjektet. Dette er ofte en nyttig måte å samle informasjon på, og en måte som kanskje bedre enn det strukturerte intervjuet legger til rette for at man kan oppdage ny informasjon.
- ✓ Et **utstrukturert** intervju vil typisk ha et planlagt tema, men utover dette lar man samtalen gå ganske fritt.

Fokusgrupper, gjerne sammensatt av brukere fra forskjellige brukergrupper, er en fin måte å involvere brukere på – man får gjerne mye nyttig informasjon, samtidig som brukerne i større grad ser systemet i et større perspektiv når de samtidig forholder seg til andre brukere. Samtalen flyttes fra individnivå til gruppenivå, så det er gjerne andre saker som kommer opp enn i intervjuer med enkeltpersoner.

I tillegg kommer det sosiale inn her, da fokusgrupper ofte møter hele utviklergruppen, i motsetning til intervjuer som ofte er en-til-en. Det å møte brukere er en viktig motivasjonsfaktor for utviklerne, og det er ofte motiverende for brukere å møte utviklerne av et system de senere skal bruke. Det kan for eksempel senke terskelen for å delta aktivt i andre brukerorienterte aktiviteter, som å la seg intervjue eller komme med innspill i andre deler av prosessen.

Observasjon

Det finnes mange måter vi kan observere brukere på. Vi kan rett og slett gå ut i “felten” og se på hva som skjer, enten ved å sende ut observatører eller ved å (selvsagt etter avtale) gjøre opptak av samtaler eller skjermer. Vi kan også oppsøke informasjon som andre samler inn men som kan fortelle oss noe om bruksmønster, som f.eks. logger over datatrafikk eller bruk av eksisterende systemer.

Å studere eksisterende systemer

Å undersøke eksisterende systemer (og kanskje spesielt lese brukerdokumentasjon knyttet til disse!) kan gi mye informasjon om hvem brukerne er og hvordan systemene brukes.

Å beskrive brukerhandlinger

Når vi undersøker kravene til et system, trenger vi ofte å beskrive *brukerhandlinger* eller *brukerinteraksjon* med systemet på en måte. Det er gjerne to typer slike handlinger vi er interessert i:

- ✓ Hvordan brukere ønsker å samhandle med systemet vi utvikler (altså som en del av kravene).

- ✓ Hvordan brukere samhandler med eksisterende systemer (som en del av informasjonsinnsamling).

Vi skal nå se på to teknikker for å beskrive slike samhandlinger. Den første er en du kjenner godt fra før, nemlig *use cases*.

Use cases

Hva use cases er og hvordan vi beskriver dem, vet du allerede. Denne teknikken kan gjerne brukes til å beskrive brukerhandlinger i interaksjonsdesign. Den har fokus på brukere som har konkrete mål, som de når ved å samhandle med systemet. Det betyr at en god use case-analyse er en god måte å beskrive brukerhandlinger: du trenger ikke å bli spesifikk om hvordan systemet løser oppgaver i det hele tatt, det som er i fokus er hele tiden brukeren og brukerens mål.

Scenarioer

Scenarioer kan tenkes på som en parallell til eller utvidelse av personaene. Det handler rett og slett om å beskrive tenkte situasjoner der systemet brukes, gjerne av nettopp personaene. Hovedforskjellen er at vi beskriver handlinger, og ikke personer, og fokuset bør ikke ligge på å beskrive systemet, men på å beskrive hvordan en persona samhandler med det.

Det å fortelle historier er en naturlig måte å uttrykke seg på for mange, og i tillegg blir det lettere å legge vekt på *motivasjon* og *mål* med handlinger mer enn handlingene i seg selv (for eksempel ved at vi kan beskrive tankene til brukeren i situasjonen) i denne formen.

Hvordan scenarioet skrives (detaljnivå, hvilke situasjoner, fokus på kontekst eller fokus på bruk av systemet) avgjøres av når og hvordan det skrives. Scenarioer kan både skrives som en del av beskrivelsen til kravene til systemet, og som brainstorming i fellesskap når man skal utforske ideer og design av systemet. Det er en veldig fleksibel teknikk som kan brukes på mange måter og som er lett å komme i gang med. Merk også at et scenario vanligvis er en kort beskrivelse av en konkret situasjon med et bestemt fokus, det vil si at det ikke er snakk om lange tekster (skriv i så fall heller flere scenarioer med utgangspunkt i flere personaer).

Å designe alternativer

Da har vi kommet til neste trinn i prosessen – å designe alternativer. Hvordan går man fram da? Generelt fødes nye design ut fra kreativitet, inspirasjon, nysjerrighet, erfaring og studier av eksisterende design. Noen kommer kanskje på grensesprengende design bare ut fra egen kreativitet, men mye skapes i fellesskap og diskusjoner mellom designere som har studert andre systemer som er relevante for det som skal lages. Kanskje er det nyttig å diskutere med andre eksperter som kanskje ikke fokuserer på designet men har god forståelse av aspekter med systemet du selv ikke egentlig forstår veldig godt? Det kan gi helt nye perspektiver og ideer om hvordan problemer kan løses.

Designfasen starter gjerne med å gjøre et *konseptuelt design*, før man kan lage mer konkrete designforslag basert på den konseptuelle modellen. Disse forslagene uttrykker man gjerne som enkle prototyper, noe som gjør at designfasen og prototypingsfasen ofte kan sies å gå litt over i hverandre (hvilken fase man er i avgjøres i så fall av hva som har størst fokus – designforslagene eller produksjon av prototyper).

Konseptuelt design

Først lager vi altså en *konseptuell modell* av systemet vi skal lage. Konseptuelle modeller blir beskrevet i leksjonen **Mer om design av brukergrensesnitt**, så om du ikke har klart for deg hva vi mener med dette begrepet anbefales det å stoppe opp og repetere litt før du leser videre.

Basert på kravene fra forrige fase, kan vi nå altså begynne å formulere en konseptuell modell for systemet. Det vil også være naturlig å la kravene til brukeropplevelsen og brukskvalitet vi kom fram til i kravene spille inn her.

Så snart vi begynner å få en modell på plass, kan vi gjerne involvere brukergrupper, for eksempel ved å invitere en fokusgruppe til diskusjon.

Det betyr at du bør velge metaforer, hvilke typer grensesnitt brukeren skal møte og hvilken stil det skal være på samhandlingen med brukeren (mye instruksjoner og forklaringer? Eller skal det legges opp til at brukeren selv aktivt utforsker grensesnittet? Går samhandlingen stort sett ut på at brukeren manipulerer objekter i grensesnittet, og/eller er det mye diaglogbokser?). Alt dette blir en del av den konseptuelle modellen.

Etter hvert blir man så klar til å bli mer konkret i designet – eksakt hvilke konkrete funksjoner skal grensesnittet tilby brukeren? Hvordan henger de sammen? Hvilken informasjon trenger brukeren tilgjengelig for å kunne bruke funksjonene? Her kan man gjerne lage skisser av grensesnitt, men typisk er dette bare skisser som fungerer som utgangspunkt for diskusjon og for å forstå og utforske sider ved den konseptuelle modellen, og ikke konkrete forslag til endelig design.

Fra konseptuelt til fysisk design og prototyper

Til slutt vil man bevege seg over til konkrete forslag til design. Her blir det ekstra relevant å tenke alternativer – den konseptuelle modellen er gjerne det felles grunnlaget for de forskjellige konkrete designene man utvikler som forslag til hvordan systemet kan realiseres. Scenarioene er en mulig og veldig relevant inspirasjonskilde her!

Vi må lage designforslag for mange forskjellige aspekter ved systemet – farger, grafisk designstil og typografi, valg av ikoner og symboler, hvor mange og hvilke typer grensesnitt brukeren skal møte, hvordan brukeren konkret samhandler med disse grensesnittene også så videre.

Å velge blant alternativene

Ofte vil man starte med mange alternativer, og så gradvis kutte ned til man sitter igjen med noen få. Det er sjelden noe poeng å begrense seg i starten, det er viktig å få mange alternativer på bordet, de fleste vil uansett utelukke seg selv etterhvert. En av de beste måtene å få gode ideer på, er som kjent rett og slett å få mange ideer!

Men etterhvert blir det nødvendig å ta noen valg, og det kan noen ganger bli vanskelig. Av praktiske årsaker velger man gjerne bort de fleste alternativene og står igjen med noen få som fortjener videre evaluering og kanskje en runde til i interaksjonsdesignsyklusen for å bli enda bedre. Men hvordan tar man et slikt valg? Det er primært to måter å få input til dette valget på (utover designernes egne meninger):

- ✓ Snakk med oppdragsgiver og brukergrupper. Hva mener de?
- ✓ Gjør et valg med utgangspunkt i *kvalitet*, det vil si at du velger noe du vil legge vekt på (f.eks. grad av brukskvalitet eller hvor godt grensesnittet er tilpasset de fysiske omgivelsene det skal brukes i) og bruker det til å ta et valg.

Prototyping

Til slutt skal vi si litt om protyping. Å lage prototyper betyr her å lage modeller av aspekter ved et design. Prototyper kan ha varierende grad av realisme, og de kan representere forskjellige sider ved et system. De trenger ikke å være virkelighetsnære (såkalte high-fidelity-prototyper) for å

være nyttige. Hovedmålet med prototypene er å forstå designet man lager bedre, og få ideer til å utvikle det videre eller utvikle flere alternativer. Prototypene kan også brukes i brukertesting.

Low-fidelity-prototyper

Dette er en samling prototypingsteknikker som ikke etterstreber å være virkelighetsnære. Disse brukes gjerne tidlig i utviklingen av et grensesnitt, for eksempel i arbeidet med den konseptuelle modellen (eller i videreutviklingen av en slik konseptuell modell), og tar gjerne utgangspunkt i bruk av penn og papir. Penn og papir er lavteknologiske verktøy som er lett tilgjengelige og som de fleste av oss kan bruke raskt og effektivt, og det gjør slike prototyper til raske og lettvinte måter å utforske en designidé på! De kommer i mange former, for eksempel:

- ✓ **Wireframes**: Wireframes er rett og slett skisser av mulige grensesnitt. Du kan skissere selv på frihånd, eller bruke et av mange wireframeverktøy som finnes (det finnes mange tilgjengelig på nett, en del av dem kan brukes gratis). Slike verktøy kommer gjerne med ferdige sett av typiske grensesnittobjekter som knapper og tekstfelter som du bare kan dra inn i wireframen din. Dette er veldig vanlig for eksempel i webdesign.
- ✓ **Papirprototyping:** Bruk wireframene dine til å simulere bruken av et grensesnitt! Dette er en mye brukt teknikk til enkel brukertesting og/eller tidlig utforsking av grensesnitt, og er kanskje best forklart ved å se på et videoeksempel.
- ✓ **Storyboarding:** Storyboarding er en teknikk som tar utgangspunkt i en samhandling en bruker gjør med systemet (for eksempel beskrevet i et scenario eller et use case) og viser en sekvens med skisser som beskriver samhandlingen mellom brukeren og systemet sammen med tekstforklaringer. En storyboard vil altså både lage representasjoner av et grensesnitt (slik som wireframes) og tenkt brukersamhandling. Du kan tenke på det som å lage en slags tegneserie av et brukerscenario!
- ✓ **Wizard of Oz:** Dette er en teknikk der man raskt lager et grensesnitt som *ser ut som* et forslag til et brukergrensesnitt, men i stedet for å bruke tid på å skrive logikk for å få det til å oppføre seg som et ekte grensesnitt, lar man en person sitte bak og "trekke i trådene" når en designer eller bruker tester det. (Det blir på en måte det motsatte av et forsøk på å bestå Turingtesten – i stedet for at en maskin forsøker å late som om den er et menneske, forsøker et menneske å late som om det er en maskin...)

High-fidelity-prototyper

Virkelighetsnære prototyper er ofte krevende å lage, så de bør vi begrense til sene iterasjoner og situasjoner der vi har et reelt behov for å teste en veldig virkelighetsnær modell av et system. De har klare fordeler i at de kan gi bedre totalbilde av systemet og det blir lettere for brukere å gi fullstendige tilbakemeldinger, men prisen å betale er at det ofte er dyrt og tidkrevende å utvikle, og selv små feil og mangler kan ende opp med å gi brukere feil inntrykk av systemet.

Merk at noen mulige teknikker for å lage relativt virkelighetsnære prototyper på grensesnitt kan være å lage nettsider med enkel HTML og JavaScript for å simulere mer komplekse programvaresystemer (det kan være kostnadseffektivt i forhold til å utvikle "ordentlige" grensesnitt) eller å bruke verktøy som PowerPoint eller KeyNote for å lage prototyper. Det sistnevnte brukes ofte i forbindelse med nettsteder (og kalles da gjerne for *mockups*), og presentasjonsprogrammer er overraskende velegnet til dette – de har cirka samme størrelse som en nettside og er lette å endre layout på, man kan sette inn tabeller, og overskrifter, bilder og videosnutter, man kan lage lenker mellom slides og lage maler for flere sider. I tillegg finnes egne sett med bilder av for eksempel knapper og nedtrekkslister som kan brukes til å lage slides som ligner veldig på nettsider (enkelte kan lastes ned gratis fra nett). Se på denne videoen for å lære mer om dette!

Evolusjonær prototyping vs. bruk-og-kast

Når det gjelder prototyping, er det to forskjellige tilnærminger. Noen ganger tenker man på prototypene som rene hjelpemidler i designprosessen som skal kastes når man har kommet fram til en endelig design. Dette er *bruk-og-kast*-tilnærmingen. Et alternativ er *evolusjonær prototyping*, det vil si at man lar prototypene utvikle seg gjennom iterasjonene, for til slutt å faktisk bli det endelige produktet. Valget av strategi bestemmer hvordan man lager prototypene – en prototype som skal utvikle seg til det endelige produktet stiller mye høyere krav til kvalitet og testing underveis. Man kan ikke gjøre like mange kompromisser og forenklinger som man gjerne ellers gjør i prototyper, og man må være nøye med å dokumentere utviklingen av prototypene!

Oppsummering

Vi har nå sett nærmere på de tre første fasene i en syklus i interaksjonsdesignprosessen (evaluering skal vi se på i neste leksjon). Vi må først innhente så mye relevant informasjon som mulig for å kunne lage gode *krav* til systemet. Dette kan vi gjøre for eksempel ved å bruke teknikkene beskrevet (spørreundersøkelser, intervjuer, observasjon eller studier av eksisterende systemer). I designfasen lager vi et konseptuelt design, og bruker dette til å generere mer konkrete designforslag. Disse forslagene uttrykkes gjerne ved hjelp av prototyper, som både tjener til utforsking av idéer, inspirasjon og enkel brukertesting.

Designforslagene som overlever testing og evaluering, blir med til neste iterasjon. Her bør vi gå gjennom kravene på nytt med hensyn på å forbedre dem, og vurdere om vi trenger å innhente enda mer informasjon. Kanskje har brukertestingen i forrige syklus vist oss at ikke alle kravene var helt riktige, og vi har fått ny forståelse av noen av dem? Vi jobber videre med å forbedre den konseptuelle modellen og konkrete designforslag på grunnlag av de forbedrede kravene (eller annen kunnskap tilegnet i forrige syklus) og kommer igjen til evalueringen. Slik holder vi på til vi har et grensesnitt som er godt nok til at vi kan avslutte!

Noen spørsmål for å se hva du har fått med deg:

1. Forklar hva de tre fasene *definere krav*, *designe alternativer* og *prototyping* går ut på, og hva som er de viktigste tingene å tenke på i hver av disse fasene.

2. Forklar kort hvordan vi kan gå fram for å samle inn informasjon i kravspesifikasjonsfasen av interaksjonsdesign.

3. Forklar teknikkene *personaer* og *scenarioer*, og hvorfor disse kan være nyttige.

4. Beskriv noen forskjellige typer prototyper, og hva som er fordelene og ulempene med dem.

Referanser

Rogers, Yvonne; Sharp, Helen og Preece, Jenny. 2011.
Interaction Design: beyond human-computer interaction. Tredje utgave.
West Sussex, United Kingdom: John Wiley & Sons Ltd.
ISBN 978-0-470-66576-3

Kujala, S. 2003.
User involvement: A review of the benefits and challenges.
Behaviour & Information Technology, 2003, 22(1), 1-16.