**Universitatea POLITEHNICA din Bucuresti**

**Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei**

**Design de Interfete pentru**

**Utilizatori**

**Studenti:** Frone Catalin-Alexandru

 Jerca Constantin

**Grupa:** 441A

**Profesor**

**coordonator:** Dr. Ing. Ştefan Stăncescu

**Bucuresti 2016**

**Cuprins**

1. **Probleme de design** – Jerca Constantin
	1. Interactia cu utilizatorul
	2. Prezentarea informatiei
2. **Procesul de design al UI** – Jerca Constantin
3. **Analiza utilizatorului** – Frone Catalin
4. **Prototipuri de interfete cu utilizatorul** – Frone Catalin
5. **Evaluarea interfetelor** – Frone Catalin
6. **Bibliografie**
7. **Probleme de design**

Inainte sa se discute procesul de design al interfetelor pentru utilizatori, trebuie sa se semnaleze cateva probleme de design pe care un producator de software UI le-ar putea intampina. El ar trebui sa-si puna cateva intrebari cheie: „cum ar trebui user-ul sa interactioneze cu sistemul?” sau „cum ar trebui prezentata informatia user-ului?”.

 O interfata cu userul coerenta ar trebui sa integreze atat interactia cu user-ul, cat si prezentarea informatiei.

* 1. **Interactia cu user-ul**

Interactia cu user-ul consta in trimiterea de comenzi si date catre un sistem. Aceste forme de interactie ale user-ului cu sistemul au fost clasificate in cinci stiluri primare:

1. *Manipularea directa* – user-ul interactioneaza direct cu obiectele de pe ecran (sunt necesare dispozitive specifice: mouse, touch screens, etc);
2. *Selectia din meniu* – user-ul selecteaza o comanda dintr-o lista de posibilitati (meniu);
3. *Scrierea intr-un camp* – user-ul completeaza un camp de tip fill-in (in majoritatea cazurilor, aceste campuri sunt insotite de butoane care, apasate, initiaza actiuni specifice);
4. *Limbaj de comanda* – user-ul introduce comenzi speciale asociate cu diversi parametrii care specifica sistemului ce sa faca;
5. *Limbaj natural* – user-ul introduce comenzi in limbajul natural (acest stil e, de obicei, o interfata catre stilul cu limbaj de comanda).

Fiecare dintre aceste stiluri are avantaje si dezavantaje si e potrivit diverselor aplicatii si diversilor useri.



*Tabel 1 – Avantaje si dezavantaje*

*ale tipurilor de interactii*

*Shneiderman - 1998*

 In aplicatiile mai dezvoltate sunt regasite mai multe stiluri de interactie cu user-ul. Este posibil ca pentru acceasi aplicatie sa se regaseasca stiluri diferite de interactie (ex: la sistemul de operare Linux se gasesc stiluri diferite de interactie pentru utilizatorii casual – GUI si utilizatorii experimentati – Unix shell interface).



*Figura 1*

*Inginerie Software – A. Sommerville*

* 1. **Prezentarea informatiei**

Toate sistemele interactive trebuie sa aiba o modalitate de a prezenta informatia catre user. O buna tehnica de design este sa diferentiezi software-ul de prezentare de informatia propriu-zisa (Fig. 2).



*Figura 2*

*Inginerie Software – A. Sommerville*



*Figura 3*

*Smalltalk – Goldberg si Robson 1983*

Modelul MVC (Fig. 3) e o cale efectiva de combinare a mai multor prezentari de date. Userii pot interactiona cu fiecare prezentare in functie de tipul prezentarii si al datelor ce se doresc a fi vizualizate. Datele sunt incapsulate intr-un obiect model. Fiecare obiect de date are un numar separat de tipuri de vizualizare (diferite display-uri care reprezinta modelul de date).

Pentru a gasi cea mai buna prezentare de informatie, trebuie sa cunosti abilitatile user-ului si in ce fel va folosi acesta sistemul (ex: pentru reprezentantii de vanzari ai unei companii e necesar atat prezentarile text, care ilustreaza valorile exacte, cat si prezentarile grafice, care pun in evidenta diferite trand-uri – Fig. 4) .



*Figura 4*

1. **Procesul de design al UI**

Designul UI e un proces iterativ unde utilizatorul interactioneaza cu designerul pentru a decide care vor fi particularitatile interfetei. De obicei, atunci cand e folosita metoda de dezvoltare iterativa, designul de interfete se desfasoara odata cu dezvoltarea software-ului, insa inainte de a incepe programarea, ar trebui sa ai cateva prototipuri de interfete pregatite.

****

*Figura 5*

*Inginerie Software – A. Sommerville*

 In figura 5 e prezentat procesul de design al unei interfete pentru utilizator. Aici se regasesc trei activitati principale:

1. ***Analiza user-ului*** – dezvoltarea unei intelegeri a activitatilor pe care utilizatorul le face, mediului de lucru, altor sisteme pe care le folosesc, cum interactioneaza cu alti useri, etc;
2. ***Prezentarea de prototipuri*** – utilizatorilor le este foarte greu sa fie specifici asupra facilitatilor necesare interfetei pana cand nu le este prezentat ceva concret, deci e important sa dezvolti prototipuri de interfete si sa le expui utilizatorilor, astfel incat acestia sa poata ajuta la dezvoltarea lor;
3. ***Evaluarea interfetelor*** – desi un designer de interfete pentru utilizatori trebuie sa aibe discutii cu acestia in timpul procesului de prototyping, el trebuie sa aiba si o activitate de evaluare mai formala unde colecteaza informatii despre experientele utilizatorului cu interfata dezvoltata.

**3. Analiza utilizatorului**

O parte critical a design-ului interfetei este analiza activitatilor efectuate de catre utilizator care vor fi efectuate de catre software-ul ce va fi creat. Daca nu se stie in ce mod va fi folosit programul final, atunci nu se va putea proiecta o interfata eficienta.

Principala provocare pe care proiectanti de interfete o intampina este sa descrie principalele sarcini ale programului atat celorlalti proiectanti care vor lucre impreuna cu el cat si utilizatorilor. Diagrame cum ar fi cea UML sunt ideale pentru a descrie fluxul de sarcini ale programului insa aceste tipuri de diagrame pot fi utilizate doar pentru a prezenta analiza catre ceilalti proiectanti implicate in dezvoltarea software-ului intrucat utilizatori obisnuiti ar considera prea tehnice aceste tipuri de diagrame.

**Tehnici de analiza**

Exista trei tehnici de analiza pentru utilizatorii de baza:

* analiza sarcinilor (se concentreze pe individ și munca individului);
* interviuri și chestionare (se concentreze pe individ și munca individului);
* etnografie (se considera o perspectivă mai largă ce se concentereaza pe modul în care oamenii interacționează unii cu alții, cum trebuie aranjat mediul lor de lucru și modul în care acestia cooperează pentru a rezolva problemele).

Forma de analiza a datelor cel mai des utlizata este Analiza Ierarhica a Sarcinilor ( HTA – Hierarchical Task Analysis). Aceasta a fost inițial dezvoltata pentru a ajuta la scrierea manualelor de utilizare, dar poate fi de asemenea utilizata pentru a identifica ceea ce utilizatorii fac pentru a atinge un scop. In Analiza Ierarhica a Sarcinilor, o sarcina la nivel inalt este defalcata in sub-sarcini si planurile sunt identificate pentru a vedea ce se poate intampla intr-o situatie specifica. In notatia Analizei Ierarhice a Sarcinilor, o linie sub chenarul activitatii indica in mod normal, ca aceasta nu va fi descompusa in subactivitati mai detaliate.

Avantajul Analizei Ierarhice a Sarcinilor fata de scenariul limbajului natural este ca obliga sa se ia in considerare fiecare sarcina și sa se decida daca aceasta ar trebui sa fie descompusa. Cu scenariul limbajului natural, este ușor sa omiti sarcini importante. De asemenea, citirea scenariilor poate deveni lunga si plictisitoare daca se doreste adaugarea unei multimi de detalii.

In mod normal, colectarea informațiilor HTA se face prin observare și intervievarea utilizatorilor. In acest proces de intervievare, se pot colecta o parte din informatiile aditionale si se pot inregistra alaturi de analiza sarcinilor. In timpul interviurilor ce au scopul de a descoperi sarcinile utilizatorilor, intrebarile ar trebui formulate astfel incat utilizatorii sa furnizeze informațiile relevante. Acest lucru inseamna ca interviul trebuie sa aiba o oarecare flexibilitate, sa nu ramana doar in zona intrebarilor prestabilite, ci sa se dezvolte in sensul incurajarii utilizatorilor pentru a oferi informatiile necesare analizei.

Pe langa interviurile individuale, se pot organiza si interviuri cu grupuri tinta. Avantajul folosirii grupurilor tinta este ca utilizatorii se stimuleaza reciproc pentru a oferi informatii. Analiza sarcinilor se concentreaza pe modul in care indivizii functioneaza, dar, desigur, munca reprezinta o cooperare. Oamenii lucreaza impreuna pentru a atinge un obiectiv, iar utilizatorilor li se pare dificil sa identifice modul in care aceasta cooperare are loc de fapt. Prin urmare, observarea directa a modului in care utilizatorii lucreaza și a modului de a folosi sisteme bazate pe calculator este o tehnica suplimentara importanta de analiza.

O abordare relevanta pentru analiza sarcinilor este si cea a etnografilor. Acestia observa indeaproape modul in care oamenii lucreaza, modul in care interactioneaza cu altii si modul in calitatile individuale sunt folosite pentru a sprijini activitatea. Avantajul acestei abordari este ca etnograful poate observa acțiuni intuitive și colaborari informale ce pot conduce la alte discutii despre munca echipei.

Asadar, individual, niciuna din tehnicile de analiza nu poate oferi o imagine completa a activitatii utilizatorilor. Tehnicile trebuie utilizate in paralel, pentru a se completa si pentru a conduce la intelegerea intregului proces de lucru al utilizatorilor.

**4. Prototipul interfetei cu utilizatorul**

Din cauza naturii dinamice a interfeței utilizatorului, descrierile textuale și diagramele nu sunt suficient de bune pentru a exprima cerințele de interfața de utilizator. Prototipul evolutiv sau de explorare cu implicarea utilizatorului final este singura modalitate de a proiecta și de a dezvolta interfețe grafice pentru sisteme software. Implicarea utilizatorului in procesul de proiectare și dezvoltare este un aspect esențial al design-centrat pe utilizator (Norman și Draper, 1986), o filosofie de design pentru sistemele interactive.

Scopul prototipurilor este de a permite utilizatorilor sa castige experiența directa cu interfața. Cei mai multi dintre noi gasim dificila gandirea abstracta despre o interfața de utilizator și de a explica exact ceea ce ne dorim. Cu toate acestea, atunci cand ne sunt prezentate cu exemple, este ușor sa se identifice caracteristicile care ne plac sau displac.

La crearea de prototipuri ar trebui sa se adopte un proces in doua etape:

1. Foarte devreme in acest proces, ar trebui sa dezvolte prototipuri pe hartie - machete de desene agreate cu utilizatorii finali;
2. Rafinarea design-ului si dezvoltarea in procese automate din ce in ce mai sofisticate, apoi livrarea catre useri pentru testare si simularea activitatii.

Prototipurile pe hartie sunt o abordare ieftina si eficienta pentru a dezvolta prototipuri (Snyder, 2003). Nu aveți nevoie de dezvoltarea unui software executabil și modelele nu trebuie sa fie elaborate la standarde profesionale. Se pot desena pe hartie versiuni ale ecranelor de sistem cu care utilizatorii interactioneaza cu un design si un set de scenarii care descriu modul in care ar putea fi utilizat sistemul. Cu cat scenariul progreseaza, se vor schita informatiile disponibile pentru utilizatori. Apoi se trece prin aceste scenarii cu utilizatorii, pentru a simula modul in care sistemul ar putea fi folosit. Aceasta este o modalitate eficienta de a obține reactii initiale ale utilizatorilor la acea interfata, informatiile de care au nevoie de la sistem si modul in care acestia ar interacționa cu sistemul in mod normal.

Alternativ, se poate utiliza o tehnica de “storyboarding” prin care se prezinta interfata designului. Un storyboard reprezinta o serie de schite care ilustreaza o secventa de interactiuni. Aceasta tehnica poate fi mai convenabila atunci cand sunt prezentate propunerile de interfata pentru grupuri si nu decat pentru indivizi.

Dupa experimentele initiale cu un prototip pe hartie, ar trebui sa se puna in aplicare un program prototip de interfata. Aceasta presupune un sistem functional cu care utilizatorii pot interacționa. Daca sunt prototipuri de tip interfata utilizator foarte devreme in procesul de dezvoltare a sistemului, acest lucru ar putea sa nu fie disponibil. Pentru a trece de aceasta problema, se poate folosi prototipul "Vrajitorul din Oz”: utilizatorii interacționeaza cu ceea ce pare a fi un sistem informatic, dar intrarile lor sunt de fapt directionate catre o alta persoana ascunsa care simuleaza raspunsurile sistemului. Ei pot face acest lucru in mod direct sau prin utilizarea unui alt sistem pentru a calcula raspunsurile necesare. In acest caz, nu aveți nevoie de un alt software executabil in afara de interfața cu utilizatorul.

Exista trei abordari ce pot fi utilizate pentru prototipul interfetei cu utilizatorul:

* Abordarea bazata pe script: se creeaza ecrane cu elemente vizuale, cum ar fi butoane si meniuri, și se asociaza un script cu aceste elemente. Atunci cand utilizatorul interacționeaza cu aceste ecrane, script-ul este executat, iar ecranul urmator este prezentat, aratandu-le rezultatele acțiunilor lor.
* Limbaje de programare vizuale (ex: Visual Basic): incorporeaza un mediu de dezvoltare puternic, acces la o gama larga de obiecte reutilizabile și un sistem de dezvoltare al interfetei utilizatorului care permite crearea rapida de interfete, cu componente și script-uri asociate cu obiectele interfata.
* Prototipuri bazate pe Internet (ex: Java, browser de internet): ofera o interfata predefinita. Functionalitatea se adauga prin asocierea de segmente de programe Java cu informațiile care urmeaza a fi afisate. Aceste segmente (numite applet) sunt executate in mod automat atunci cand pagina este incarcata in browser.

**5. Evaluare interfetei**

Evaluare interfetei este etapa finala prin care se verifica daca interfata este utilizabila si daca respecta cerintele clientului.

Ideal evaluarea ar trebui bazata pe atributele de folosire a interfetei:

* **Viteza de operare** : cat de bine se potriveste sistemul cu cerintele si modul de lucru al utlizatorului.
* **Recuperabil** : cat de bun este sistemul la remedierea unei erori introduse de catre utilizator.
* **Adaptabil** : daca sistemul poate fi adaptat si pentru alte sarcini sau este facut strict pentru o singura sarcina.
* **Usurinta de invatare** : cat de usor ii este unui utilizator nou sa invete sa foloseasca programul
* **Toleranta** : cat de tolerant este sistemul la erorile introduse de catre utilizatori.

Evaluarea sistematica a unei interfete poate fi destul de costisitoare, este posibil ca pentru realizarea aceste evaluari sa fie facute un numar destul de mare de experimente cu diferite tipuri de utilizatori. Realizara unei astfel de evaluari a interfetei este destul de improbabila din punct de vedere financiar pentru dezvoltatori ce lucreaza in companii mici cu resurse limitate. Din acest motiv in mod normal pentru evaluare se aplica una din aceste metode:

* Chestionare care colecteaza informatii despre ce cred utilizatori despre interfata;
* Observarea utilizatorilor in timp ce folosesc interfata;
* Includerea in software a unui cod care colecteaza infromatii despre cele mai folosite facilitati ale sistemului si cele mai des aparute erori.
1. **Chestionare**

Evaluare interfetei prin chestionare este o metode destul de ieftina insa aceastea nu vor oferi foarte multe informatii daca intrebarile nu vor fi elaborate bine. De exemplu un utilizator nu ar trebui intrebat cum i s-a parut interfata, deoarece la aceasta intrebare se vor primi foarte multe raspunsuri diferite iar realizarea unei analiza pornind de la aceste raspunsuri va fi greu de facut. In realizarea chestionarelor ar trebuie folosite intrebari de genul ‘Evaluati usurinta de utilizare a sistemului de la 1 la 5’ deoarece la aceste intrebari este mai usor de raspuns dar este si mai usor de facut o analiza pornind de la ele.

De asemenea in chestionare ar trebui inclus si istoricul utilizatorului pentru a vedea daca o anumita tipologie de utilizatori intampina probleme in folosirea lui.

1. **Observarea utilizatorilor**

Aceasta evaluare este bazata pur si simplu pe privirea unui utilizatorilor in timp ce folosesc programul, unde apar erori, ce facilitate ale programului folosesc cel mai des acestia. Aceasta metoda poate fi suplimentata prin intalniri in care cativa utilizatori sa prezinte care sunt puntele slabe si punctele forte ale programului.

1. **Includerea in software a unui cod care colecteaza infromatii**

Implementarea unui cod care sa colecteze date statistice permite imbunatatirea interfetelor in mai multe moduri. Prin implementarea acestui cod pot fi detectate cele mai folosite operatii, interfata poate fi reorganizata astfel incat aceste operatii sa fie executate mai rapid.

1. **Bibliografie**
* [A. Sommerville – “Software engineering”, ed. VIII, Addison Wesley, 2007](http://stst.elia.pub.ro/news/IS/IS_PPT/Software%20engineering%208th%20ed.pdf)
* Smalltalk – Goldberg si Robson, 1983
* Avantaje si dezavantaje ale tipurilor de interactii - Shneiderman, 1998
* PRINCIPII DE DESIGN ALE INTERFEŢELOR – UAV – M.Sucan, I. Dzitac
* <https://en.wikibooks.org/wiki/C%2B%2B_Programming/Code/Design_Patterns>