SISTEMELE INTERACTIVE - PREZENTARE DIN PERSPECTIVĂ ISTORICĂ

drnd. ing. Cristina Niculescu

CCAIAPLNMC-Academia Română
ncristin@racai.ro; http://www.racai.ro/~ncristin

Rezumat: Lucrarea tratează subiectul evoluției istorice a domeniului proiectării sistemelor interactive. După o succintă introducere în domeniul interacțiunii om-calculator, se prezintă paradigmele și principiile de interacțiune, care stau la baza unei proiectări corecte, precum și modelele utilizatorului și sistemului. Etapele procesului proiectării sistemelor interactive și metodele de evaluare ale acestuia constituie subiecte ale capitolelor următoare. Articolul evidențiază evoluția sistemelor interactive, de la cele monoutilizator, la cele mai complexe, multiutilizator. În finalul lucrării, sunt analizate câteva probleme specifice proiectării sistemelor CSCW. Apariția unui nou tip de sisteme de interacțiune, sistemele de management al fluxului de lucru în Internet, vor revoluționa tehnologia informatică actuală, creând cadrul globalizării "afacerilor1".

Cuvinte cheie: interacțiune om-calculator (HCI), interfață om-calculator, paradigme și principii de interacțiune, modele ale utilizatorului și sistemului în proiectare, hipertext, psihologie cognitivă, arhitecturi cognitive, CSCW (*Computer Supported Cooperative Work*), sisteme de management al fluxului de lucru.

1. Introducere

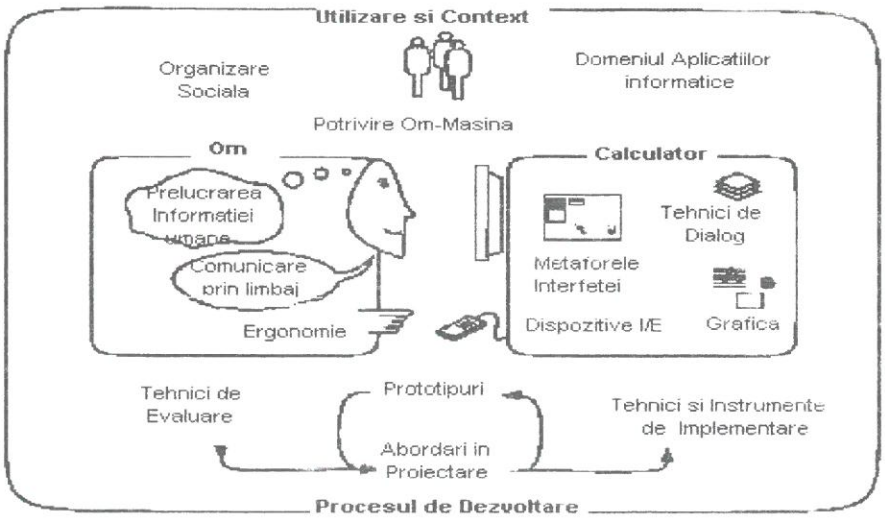
Interacțiunea om-calculator (HCI) este o disciplină care are ca obiect proiectarea, evaluarea și implementarea sistemelor interactive computaționale pentru o utilizare eficientă de către om, cu studierea fenomenelor majore care le însoțesc. HCI constituie, cu siguranță, un subdomeniu al ingineriei de calculatoare (hard și soft), dar presupune, de asemenea, o serie de cunoștințe din alte domenii de activitate, cum sunt: psihologie cognitivă și comportamentală, sociologie, lingvistică, semantică, semiotică etc. În afara faptului că HCI este o activitate interdisciplinară, problematica ei poate fi tratată atât din perspectiva ingineriei, cât și din cea a științelor umaniste.

Interfața om-calculator reprezintă acea parte a sistemului de calcul (software-hardware) prin intermediul căreia o persoană (operatorul) interacționează pentru a folosi posibilitățile de lucru ale unui anumit program informatic. Ea permite utilizatorului execuția unor acțiuni și, totodată, este mijlocul prin care calculatorul raportează informațiile-rezultat către acesta.

Există mai multe modalități prin care utilizatorul poate comunica cu sistemul. La o extremă ar fi intrarea de tip comenzi *batch*, prin care utilizatorul furnizează toate comenzile calculatorului deodată, lăsând mașina să execute aplicația. Această abordare nu implică interacțiunea om-calculator, dar nu permite nici realizarea multor programe în mod satisfăcător. La extrema cealaltă, ar fi paradigmele și dispozitivele de intrare interactive cum sunt *manipularea directă* și *aplicațiile de realitate virtuală*. În astfel de cazuri, utilizatorul furnizează continuu instrucțiuni și primește feedback-ul corespunzător.

Alături de interfața propriu-zisă, interacțiunea om-calculator implică un context de utilizare, trăsături specifice omului (prelucrarea informației, comunicarea prin limbaj, ergonomia cognitivă etc.), precum și caracteristici specifice mașinii (produse software de interacțiune, implementate pe o anumită structură hard); de asemenea, prin utilizarea tehnicilor de evaluare a produsului procesului de proiectare, omul poate alege varianta optimă de proiectare a unui sistem de interacțiune într-un context dat.

Iată aceste idei redate într-o schemă de Keith Andrews în cursul său pe această temă [1]:



1 Prin *afaceri* înțelegem orice activitate (fie ea educațională sau de cercetare) desfășurată de o organizație.

În Internet, există un grup de interes, dedicat interacţiunii om-calculator: *SIGCHI* (*S*pecial *I*nterest *G*roup on *C*omputer-*H*uman *I*nteraction) [17], care are acum şi reprezentare în România [18]. Scopul acestui grup este studierea factorilor umani în procesul interacţiunii om-calculator, incluzând cercetarea, proiectarea, dezvoltarea şi evaluarea sistemelor computaţionale interactive. Accentul se pune pe comunicarea umană şi pe interacţiunea cu sistemele de calculatoare. SIGCHI furnizează un forum de schimb de idei între oamenii de ştiinţă din domeniul calculatoarelor, al ştiinţelor cognitive ale comportamentului uman, între proiectanţii de sisteme şi utilizatorii terminali.

Interfeţele inteligente sunt acele interfeţe care îşi duc la îndeplinire rolul, facilitând operatorului utilizarea funcţiilor sistemului într-un mod intuitiv, acordând asistenţă în caz de necesitate; conform gradului de inteligenţă, acest ajutor este personalizat în funcţie de utilizator. Inteligenţa, în acest caz, nu înseamnă o prelucrare suplimentară a informaţiei, ci semnifică prezentarea informaţiei într-un mod adecvat.

Interfeţele sunt proiectate de specialişti în calculatoare, dar aceştia trebuie să ţină cont că ele vor fi utilizate, în majoritatea cazurilor, de nespecialişti.

Noii utilizatori de sistem au nevoie de ceva mai mult decât de o reprezentare grafică care "*să invite*", ei au nevoie de *a şti* cum să procedeze pentru a obţine ce doresc. O dată cu creşterea complexităţii seturilor de caracteristici ale programelor, asigurarea unui *help* "on-screen" a devenit o necesitate. Cerinţele de ajutor se împart în trei clase [4]: *"Just do it for me"* (totul de-a gata), *"Lead me through it"* (dirijare) şi "*Watch as I do it and make suggestions*" (urmărire şi sugestii).

Pentru manipulările în timp real şi pentru mediile interactive, sunetul are un rol important. O interfaţă 3-D se justifică pentru manipularea elementelor 3-D (obiecte vizibile pe ecran). Există tendinţa creării interfeţelor utilizator alternative; vorbirea şi scrisul de mână, cu amprenta utilizatorului, vor putea înlocui scrisul la tastatură; se fac cercetări în acest domeniu.

Interfeţele care implică interacţiunea om-calculator prin text, grafică, sunet şi video, folosind şi concepte de tip *hipertext* sunt interfeţe cu facilităţi *multimedia*, unele dintre ele putând fi şi de *realitate virtuală* (când reprezentările definesc o realitate posibilă mental şi senzorial) sau de *realitate mixtă* (când cele două tipuri de reprezentare, virtuală şi reală, coexistă).

O nouă tehnologie, *agenţii software* ajută la personalizarea interacţiunii cu calculatorul. Ei reprezintă programe cu un anumit grad de inteligenţă şi mobilitate în reţea, care se instruiesc din acţiunile utilizatorului, adunând informaţiile solicitate pentru realizarea acţiunilor specificate.

Interfaţa dintre utilizator şi sistem trebuie să traducă efectiv semnalele, astfel încât interacţiunea să poată avea succes. Utilizarea modelelor de interacţiune ajută la înţelegerea exactă a ceea ce se întâmplă în interacţiune şi la identificarea cauzelor dificultăţilor de implementare.

Lucrarea tratează subiectul evoluţiei şi proiectării sistemelor interactive. Se prezintă, de asemenea, paradigmele şi principiile de interacţiune om-calculator, care stau la baza unei proiectări corecte, precum şi modele ale utilizatorului şi sistemului. În finalul lucrării, sunt analizate câteva probleme specifice proiectării sistemelor CSCW.

2. Paradigme şi principii ale proiectării interfeţei om-calculator

Paradigme şi principii ale proiectării interfeţei om-calculator, prezentare generală

Strategiile efective, de construcţie a sistemelor interactive, furnizează *paradigme* de proiectare a acestor sisteme astfel încât acestea să poată fi utilizate eficient. Evoluţia acestor paradigme ilustrează o perspectivă asupra istoriei calculatorului.

Un proiectant al unui sistem interactiv are de răspuns la două întrebări:

- cum poate fi dezvoltat un sistem interactiv pentru asigurarea eficienţei în utilizare?
- cum poate fi măsurată sau demonstrată eficienţa în utilizare a unui astfel de sistem?

Sunt două modalităţi de a da un răspuns acestor probleme. Prima modalitate este prin intermediul exemplelor, în care sisteme interactive de succes sunt considerate eficiente şi, prin urmare, servesc ca paradigme pentru dezvoltarea produselor viitoare.

A doua abordare este mult mai teoretică, derivând din *principiile* abstracte ale interacţiunii efective, din cunoştinţele legate de aspecte psihologice, computaţionale şi sociologice ale domeniilor problemei. Aceste principii direcţionează proiectarea şi evaluarea unui produs program.

Distincția între paradigme și principii reprezintă o reflectare a *interacțiunii om-calculator* (HCI) ca disciplină.

Progresele mari înregistrate în tehnologia calculatorului au mărit posibilitățile de comunicare om-calculator, dar eficiența comunicării se bazează pe creativitatea umană în construirea aplicațiilor. Paradigmele interacțiunii au fost pentru mult timp dependente de progresul tehnologiei și de aplicarea lor creativă în îmbunătățirea interacțiunii. Principiile interacțiunii sunt independente de tehnologie; ele depind, în cea mai mare parte, de o înțelegere mai aprofundată a elementului uman din interacțiune.

Viitorul în proiectarea sistemelor interactive se bazează pe o abordare complementară a celor două elemente. Creativitatea care dă naștere unor noi paradigme, trebuie stimulată de dezvoltarea unei teorii care furnizează principii care să susțină paradigmele.

Paradigme pentru interacțiune

Marile progrese înregistrate în interacțiunea om-calculator s-au înregistrat datorită explorărilor și a proiectărilor creative. Vom discuta, mai jos, câteva paradigme, în ordinea apariției lor istorice, accentul punându-l pe inovația tipului de interacțiune.

Timpul de lucru partajat ("time-sharing")

În anii 1940-1950, revoluția a fost în domeniul tehnologiei hardware: releele mecanice au fost înlocuite de tuburile electronice. Apoi tuburile au fost înlocuite de tranzistoare, tranzistoarele de circuite integrate. În anii următori, datorită cercetărilor organizate și întreprinse de J. C. R. Licklider, directorul Oficiului Tehnicilor de Prelucrare a Informației al Agenției Proiectelor de Cercetare Avansată (ARPA) al Departamentului de Apărare al Statelor Unite, s-a dezvoltat conceptul de "timp de lucru partajat" (*time-sharing*). Implementarea acestui concept dă posibilitatea unui calculator să suporte, simultan, mai mulți utilizatori. Înainte, utilizatorul-programator era restricționat la sesiuni de lucru secvențiale (*batch*).

Unitățile video display

La începutul anilor 1950, s-a experimentat utilizarea primului video-display, în aplicațiile militare din Statele Unite. Acestea furnizau un mediu de interacțiune mai adecvat decât tipărirea pe hârtie.

Calculatoarele puteau fi folosite acum nu numai pentru prelucrarea datelor; ele fiind capabile să traducă probleme abstracte în forme concrete perceptibile. Astfel, calculatorul a fost forțat să se apropie de limbajul uman în loc ca omul să utilizeze un limbaj mai pe înțelesul calculatorului.

Seturi de unelte de programare

În 1960, Douglas Engelbart a organizat o echipă cu care a făcut cercetări asupra învățării cu ajutorul calculatorului. Echipa de programatori a dezvoltat un set de unelte de programare, necesare construirii sistemelor interactive.

Ideea de a crea componente ale unui sistem, care vor permite construirea unor sisteme mai complexe, a fost folosită cu succes și în următorii ani, ducând la creșterea productivității.

Calculatorul personal

În anii 1970, dezvoltarea uneltelor de programare a dus la proiectarea calculatoarelor destinate unui singur utilizator: *calculatoare personale*.

Sistemul Window și interfața WIMP

Pentru ca un calculator să corespundă modului uman de gândire asociativ, acesta a trebuit să fie proiectat astfel încât să devină mai flexibil.

Diferite aplicații trebuiau să poată ajunge simultan la un dialog cu utilizatorul; astfel a fost creată reprezentarea rulării aplicațiilor în ferestre, interfața cu utilizatorul fiind de tip WIMP (*Windows, Icons, Menues, Pointers*). Acest tip de interfață a fost prezent prima dată pe piață în aprilie 1981.

Metafora

Asimilarea utilizării calculatorului de o mare parte a populației, se bazează pe înțelegerea folosirii lui prin analogie cu alte activități cunoscute anterior. De exemplu, tastatura seamănă la prima vedere cu o mașină de scris standard. Metafora mașinii de scris este benefică pentru o înțelegere preliminară a unui procesor de texte.

Metafora biroului (*office desktop*) este folosită pentru înțelegerea folosirii interfeței WIMP. Conform acestei metafore, diversele informații sunt grupate în ferestre, similar informațiilor de pe paginile de hârtie; aceste ferestre putând fi manevrate asemănător foilor scrise de pe biroul de lucru.

De fapt, metafora reușește să redea o primă modalitate a înțelegerii calculatorului de către un nou utilizator.

Asimilarea internațională a software-ului presupune ca o metaforă să fie valabilă și în afara granițelor naționale.

Un exemplu extrem de metaforă apare în cazul sistemelor cu *realitate virtuală*. Într-un astfel de sistem, metafora nu se poate capta pur și simplu de pe display; mai degrabă, utilizatorul face parte din metaforă, creându-și o realitate alternativă, virtuală.

Manipularea directă

O conexiune inversă (*"feedback"*) rapidă vizuală și auditivă, pe un ecran de înaltă rezoluție și cu un sistem audio de calitate, face posibilă furnizarea imediată a informației de răspuns a sistemului pentru fiecare acțiune a utilizatorului. Această reacție rapidă este numai una din trăsăturile tehnicii de interacțiune numită "manipulare directă".

În 1982, Ben Shneiderman [15], [16] a scos în evidență caracteristicile unei interfețe de manipulare directă:

- vizibilitatea obiectelor de interes;
- acțiuni incrementale asupra interfeței cu reacție rapidă la ele;
- posibilitatea reversibilităților tuturor acțiunilor, astfel încât utilizatorul să fie încurajat să exploreze acțiuni noi fără penalizări mari;
- corectitudinea sintactică a tuturor acțiunilor, astfel încât orice acțiune a utilizatorului să fie o operație legală;
- înlocuirea limbajelor de comandă complexe cu acțiuni de manipulare directă a obiectelor vizibile (de aici, vine numele de manipulare directă).

O consecință a paradigmei de manipulare directă este că nu mai există o distingere clară între intrare și ieșire; expresiile de ieșire fiind utilizate în formularea expresiilor din intrarea următoare.

Oarecum, în legătură cu vizualizarea furnizată de manipularea directă este paradigma WYSIWYG (*"What you see is what you get"* – Ce vezi este ceea ce obții), utilizată de unele procesoare de texte.

Limbaj contra acțiune

Acțiunile efectuate asupra unei interfețe, ca aceea prezentată mai sus, înlocuiesc nevoia de înțelegere a lor mai în profunzime, la nivel de sistem. Deci, comunicarea utilizator-sistem este făcută prin intermediul limbajului indirect, oferit de interfață, în locul acțiunilor directe.

Nu este necesar ca acțiunea și paradigmele limbajului să fie complet separate. O combinație interesantă între cele două apare în *programarea prin exemplu*, când un utilizator poate realiza câteva task-uri de rutină, folosind paradigma acțiunii, iar sistemul înregistrează acestea ca pe o procedură generică.

Hipertext

Formatul linear al informației nu furnizează un suport eficient pentru mintea umană. Organizarea asociativă, în rețele de informații, asemenea unui tratat științific cu trimiteri la alte documente, este mai apropiată de modelul oferit de memoria și gândirea umană.

La mijlocul anilor 1960, Ted Nelson a numit această structură nelineară a textului *hipertext*. Au trecut încă două decenii până când primele sisteme hipertext au fost comercializate.

Termenul de *hypermedia* definește o extindere a noțiunii de *hipertext*, la nivelul informației, reprezentată nu numai prin text, ci și prin: sunet, grafică, imagine, animație, film. Aceste reprezentări mai poartă denumirea și de *multimedia*, termen, care la origine exprima o colecție de reprezentări pe mai multe tipuri de mediu fizic.

Multimodalitate

Un sistem interactiv *multimodal* este un sistem care se bazează pe utilizarea mai multor canale de comunicație umană. Fiecare canal reprezintă pentru utilizator o formă (modalitate) de interacțiune. Oamenii sunt capabili să prelucreze, în mod natural, informații venite simultan pe mai multe canale (vizual, auditiv, tactil etc.). Cei care au proiectat astfel de sisteme au mimat realitatea înconjurătoare; de exemplu se poate modifica gestul făcut de un dispozitiv de *pointare*, prin vorbire, indicând ce operație să fie efectuată asupra obiectului selectat.

Sistemele *multimodal, multimedia* şi cele cu *realitate virtuală* fac parte dintr-o categorie mai largă de sisteme, numite *multisenzoriale*.

CSCW (Computer Supported Cooperative Work)

În anii 1960, s-au dezvoltat primele reţele de calculatoare, făcându-se posibilă comunicarea între calculatoare separate. Un rezultat al acestei conectări, a fost colaborarea între utilizatorii calculatoarelor din reţea, prin intermediul calculatorului. Strategia de lucru în echipă a unor oameni care utilizează în procesul de colaborare facilităţi ale tehnicii de calcul este definită de o paradigmă recentă: CSCW (*Computer Supported Cooperative Work*).

Groupware este un termen ce descrie tehnologiile electronice şi aplicaţiile software, proiectate să asigure procesul de colaborare între oameni, în cadrul diferitelor grupuri.

Groupware se bazează pe: *comunicare* (pentru schimb de informaţii), *colaborare* (participanţii lucrează împreună) şi *coordonare* (integrarea lucrărilor individuale într-o lucrare comună, cu obiectiv integrator).

Ca aplicaţii pentru sistemele deschise de calculatoare, produsele groupware sunt proiectate să funcţioneze pe platforme eterogene, cu sisteme de operare diferite şi arhitecturi diverse de reţea, inclusiv pentru utilizatori mobili.

Cercetarea interdisciplinară din domeniul CSCW reprezintă o schimbare fundamentală în abordarea proiectării sistemelor de calculatoare. Principalul factor în această proiectare îl reprezintă *modul de coordonare* a utilizatorilor sistemului, prin integrarea activităţilor lor individuale. În scopul dezvoltării sistemelor de calculatoare ce furnizează suportul adecvat pentru colaborare în organizaţii flexibile, virtuale de lucru, este crucială înţelegerea *mecanismelor de cooperare*.

Dar, astăzi, *groupware* este mai mult decât o tehnologie, este un *fenomen cultural*, care cere o planificare minuţioasă şi testare înainte de implementare. Din acest considerent, termenul *groupware* mai este folosit ca sinonim pentru *CSCW*. Aplicaţiile groupware utilizează tehnologii de colaborare într-un mediu specific de lucru, care include şi participanţii. Succesul implementării unui produs groupware este asigurat dacă, pe lângă alegerea tehnologiei adecvate, se va ţine seama de elementele culturale, economice şi sociale ale acestui mediu.

În ceea ce priveşte gama de servicii concrete de reţea, care implementează facilităţi de cooperare interumană, ea porneşte de la servicii tradiţionale, de *comunicare textuală simplă* (E-mail, *talk* etc.), pentru ca recent să se îndrepte către *servicii de tip multiutilizator multimedia* (gen: videoconferinţă, suprafeţe comune de proiectare). Acestea din urmă sunt încă în stadiu experimental, punând atât probleme teoretice (exemple: rutarea optimă în cazul transmisiei de tip *multicast*[2], definirea unor parametri de calitate a serviciilor care să caracterizeze serviciile respective, cu asigurarea flexibilităţii lor vizavi de garantarea respectării valorilor de către reţeaua de transport), cât şi tehnologice (legate de performanţele sistemului de telecomunicaţie, înregistrare/redare video, prelucrare rapidă computerizată).

Sistemele de cooperare, bazate pe *agenţi mobili* (şi eventual inteligenţi) reprezintă o nouă direcţie de dezvoltare a sistemelor CSCW.

Un loc aparte îl ocupă aplicaţiile groupware, bazate pe WWW. Facilităţile de acces a paginilor de *hypermedia* prin intermediul programelor de tip browser de Web au schimbat mult modul de colaborare distribuită. Furnizorii de aplicaţii groupware şi-au adaptat produsele pentru WWW, adăugându-le funcţionalităţi noi. De asemenea, aplicaţiile software de reţea (intranet/Internet) din piaţa informatică actuală includ facilităţi de colaborare.

Aducerea interfeţelor de colaborare la nivelul colaborării directe, nemediate de calculator nu este dezideratul actual. Este o abordare greşită a problemei, pentru că nu se vor obţine aceleaşi rezultate. Nu imitarea comunicării directe va face eficientă o astfel de comunicare prin reţea, ci exploatarea unor noi posibilităţi pe care le oferă noul mediu (de exemplu, cele de realitate virtuală).

Tehnologiile suport şi modelele aplicaţiilor de reţea pentru colaborare sunt în plină evoluţie. În viitor, acest tip de produse vor putea susţine organizaţiile printr-un management eficient al cunoştinţelor.

Managementul de cunoştinţe reprezintă o strategie deliberată de obţinere de cunoştinţe adecvate fiecărui participant, la timpul potrivit. Cunoştinţele pot proveni atât din baza de cunoştinţe a organizaţiei, cât şi din Internet. Al doilea aspect îl constituie ajutorul dat utilizatorilor în partajarea informaţiilor şi în punerea lor în acţiune pentru îmbunătăţirea performanţelor personale şi ale organizaţiei sau grupului din care fac parte.

Aşa cum "nodurile" reţelelor interconectate constituie Internetul şi între ele există o comunicare din aproape în aproape (punct-la-punct), serviciile pentru utilizatori au fost modelate iniţial printr-o arhitectură client-server, în care

---

2 Ruterele *multicast*, care înţeleg protocolul de rutare special folosit în reţeaua virtuală *Multicast backbone* [11], creează între ele *tuneluri* [2], prin care-şi trimit pachetele de difuzare.

un client comunică cu un singur server la un moment dat. Modelele de cooperare interumane sunt mai complexe decât acest dialog. Prin urmare, s-a născut o nouă necesitate: de a crea un suport de rețea, care să furnizeze facilitățile adecvate de comunicare, printr-o utilizare inovatoare a tehnologiilor existente sau prin crearea unora noi.

În dezvoltarea tehnologiilor de rețea se poate evidenția o stratificare a lor (infrastructura rețelelor, aplicațiile de rețea, aplicații de colaborare sau groupware, managementul de cunoștințe). Fiecare nivel se bazează pe funcționarea nivelului inferior corespunzător, acesta din urmă, perfecționându-se pentru a-i asigura o interfață adecvată.

Principii ale interacțiunii

În acest capitol, ne vom referi la principiile generale, care trebuie respectate în proiectarea unui sistem interactiv, pentru a putea fi utilizat în bune condiții.

Principiile ce urmează a fi prezentate se împart în trei mari categorii [6]: *asimilarea*, *flexibilitatea* și *robustețea*.

Asimilarea (ușurința în învățare) reprezintă facilitatea prin care noii utilizatori pot începe interacțiunea efectivă și ating performanța maximă.

Principiile care afectează asimilarea sunt:

- predictibilitatea (suport pentru utilizator în determinarea efectului unei acțiuni viitoare, bazat pe istoricul interacțiunilor);
- sintetizabilitatea (suport pentru utilizator în asimilarea efectului operațiilor trecute stării curente);
- familiaritatea (extensia prin care cunoștințele și experiența utilizatorului din alte domenii poate fi aplicată când interacționează cu sistemele noi);
- generalizabilitatea (suport pentru utilizator de extindere a cunoștințelor de interacțiune specifică din aplicații la alte situații similare);
- consistența (asemănarea în comportările de intrare și ieșire, care reies din situații similare sau obiective asemănătoare).

Flexibilitatea reprezintă diversitatea modalităților prin care utilizatorul și sistemul pot schimba informații.

Principiile care afectează flexibilitatea sunt:

- inițiativa dialogului (permisiunea libertății utilizatorului în locul constrângerilor artificiale impuse de sistem);
- fire de execuție multiple (abilitatea sistemului de a suporta interacțiunea utilizatorului, rulând simultan mai multe task-uri);
- migrarea task-urilor (abilitatea de a trece controlul execuției unui task dat, astfel încât el devine fie asimilat de utilizator sau sistem, fie partajat de ambele);
- substituirea (permite valorilor echivalente ale intrării și ieșirii să fie substituite arbitrar una cu alta);
- modificarea interfeței (posibilitatea de modificare a interfeței utilizator de către utilizator sau de sistem).

Robustețea reprezintă nivelul suportului furnizat utilizatorului în atingerea scopurilor propuse.

Principiile care afectează robustețea sunt:

- observabilitatea (abilitatea utilizatorului de a evalua starea internă a sistemului din reprezentarea lui perceptibilă);
- recuperabilitatea (abilitatea utilizatorului de a acționa corectiv de îndată ce a recunoscut o eroare);
- modalitatea de răspuns (cum percepe utilizatorul viteza de comunicare cu sistemul);
- concordanța cu task-ul (gradul în care serviciile de sistem suportă ca toate task-urile dorite să fie realizate de utilizator și modalitatea în care acesta le înțelege).

3. Procesul de proiectare a sistemelor interactive

3.1. Caracteristici ale procesului de proiectare

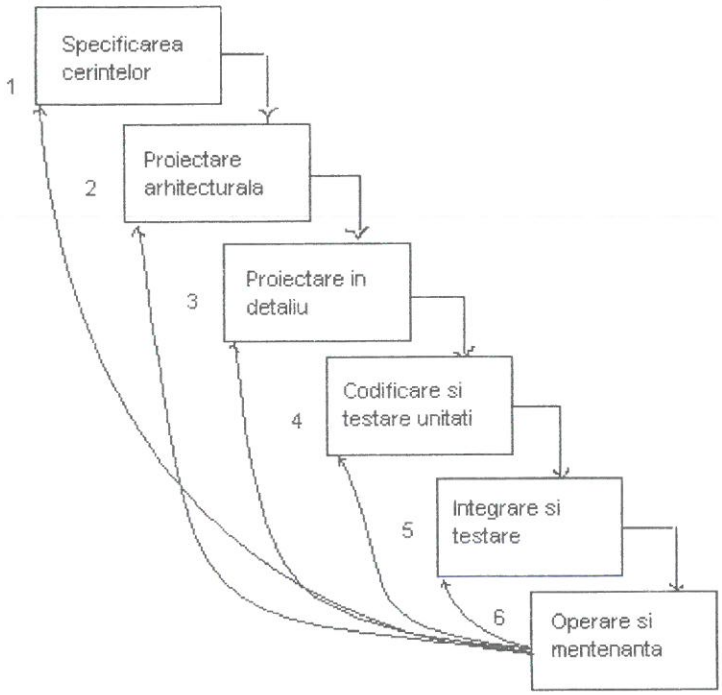
Procesul de proiectare a sistemelor interactive om-calculator trebuie să țină seama de următoarele considerente:

- ingineria software furnizează o înțelegere a structurii procesului de design și acest proces poate fi evaluat din punctul de vedere al eficienței în proiectare;
- regulile de proiectare sub forma standardelor și liniilor directoare furnizează direcții de proiectare în scopul îmbunătățirii proprietăților de interactivitate ale sistemului;
- ingineria utilității sistemului promovează criterii explicite de estimare a succesului unui produs în termeni de eficiență în funcționare (usability);

- practica de proiectare iterativă are ca rezultat încorporarea feedback-ului utilizatorului în deciziile cruciale, care afectează eficienţa în funcţionare;
- proiectarea presupune alegerea unor decizii din numeroasele alternative; proiectarea raţională furnizează un mijloc explicit de înregistrare a acestor decizii de proiectare şi al contextului în care aceste decizii au fost luate.

3.2. Ciclul de viaţă al unui produs software

Ciclul de viaţă al unui produs software descrie activităţile care au loc de la conceperea unui sistem până la înlocuirea lui. Iată aceste activităţi evidenţiate în "modelul cascadă" al ciclului de viaţă [6]:



Ciclul de viata al produselor software

Proiectarea sistemelor de interacţiune om-calculator presupune tehnici ce afectează toate etapele ciclului de viaţă al sistemului software.

Cele două părţi care concură la obţinerea performanţelor dorite ale sistemului sunt: proiectantul (sau echipa de *design*) şi beneficiarul (utilizatorii sistemului).

Voi prezenta, pe scurt, etapele ciclului de viaţă a unui produs informatic, având ca scop realizarea interacţiunii om-calculator în rezolvarea unei anumite probleme.

1. *Specificarea cerinţelor* reprezintă etapa în care sunt formulate cerinţele din punctul de vedere al beneficiarului (ceea ce trebuie să furnizeze sistemul, nu cum!), într-un limbaj potrivit implementării (în termeni precişi în interpretare şi semantică). Tot în această etapă sunt specificate informaţiile despre mediul de lucru şi tipurile de utilizatori.
2. *Proiectarea arhitecturală* are rolul de a descompune (la nivel înalt) sistemul, în componente ce pot fi dezvoltate independent sau care există deja sub forma unor produse software; totodată în această etapă sunt determinate serviciile acestor componente, interdependenţele între ele, precum şi partajarea resurselor necesare funcţionării lor.
3. *Proiectarea în detaliu* este etapa de rafinare a descrierii componentelor furnizată de etapa (2); deoarece există mai multe posibilităţi de proiectare în detaliu, care satisfac funcţionalităţile componentelor, se va alege aceea care satisface cel mai bine şi alte cerinţe ale sistemului.
4. *Codificare şi testare unităţi* se referă la implementarea într-un limbaj de programare a componentelor şi apoi a testării funcţionalităţii lor. Pentru această etapă există cercetări de automatizare a generării codului adecvat (pe baza proiectării în detaliu) şi a testelor corespunzătoare de verificare.

5. *Integrare şi testare* este etapa de integrare a componentelor, conform specificaţiilor de la nivelul (2); testele, în acest caz, sunt făcute pentru verificarea îndeplinirii cerinţelor sistemului şi asigurării folosirii partajate a resurselor.

6. *Operare şi mentenanţă* reprezintă etapa cu cea mai mare durată în viaţa unui produs software. Mentenanţa presupune corectarea erorilor sistemului, revizuirea serviciilor lui şi satisfacerea cerinţelor care nu s-au realizat în timpul dezvoltărilor anterioare; mentenanţa furnizează feedback (vezi figura de mai sus) către toate celelalte etape, reproiectarea adaptând produsul cerinţelor concrete.

3.3. Validare şi verificare

Proiectarea trebuie verificată pentru a asigura cerinţele de nivel înalt ale beneficiarului, precum şi consistenţa internă a sistemului (adică: sistemul face cum trebuie – *verificare*, ceea ce trebuie – *validare*).

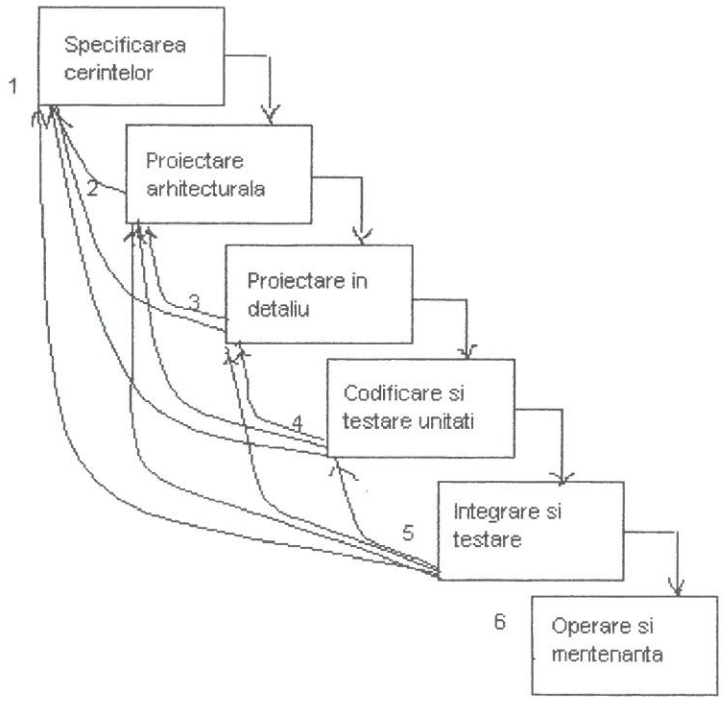
Pentru validarea şi verificarea sistemelor interactive om-calculator, expertul trebuie să aibă cunoştinţe în domeniul *psihologiei cognitive*, astfel încât specificaţiile de proiectare vor fi interpretate din perspectiva psihologică şi validate cu cerinţele interactive ale sistemului. Trecerea de la exigenţele de funcţionare a sistemului, exprimate în limbaj natural (de beneficiarul sistemului), la cele exprimate în limbajul precis al proiectării structurate este făcută de proiectant şi se bazează pe o extensie a cunoştinţelor acestuia în domeniul psihologiei. Această transformare subiectivă între cele două limbaje poartă denumirea de "*formality gap*", eficienţa ei depinzând de pregătirea proiectantului.

3.4. Managementul proiectării

Managementul proiectării adoptă o perspectivă mai largă asupra *design*-ului. Această perspectivă ia în considerare, pe lângă caracteristicile tehnice ale dezvoltării software-ului, posibilităţile de vânzare ale sistemului, necesităţile de testare, disponibilitatea de personal calificat, posibili subcontractanţi ai produsului etc.

3.5. Sistemele interactive şi ciclul de viaţă al software-ului

Nu pot fi determinate de la început toate cerinţele unui sistem interactiv. Sistemul trebuie construit şi apoi trebuie observată şi evaluată interacţiunea sa cu utilizatorii, în scopul determinării modalităţilor de schimbare pentru a-l face mai eficient în utilizare.



Reprezentarea iteratiei in modelul cascada al ciclului de viata al unui produs HCI

Modelele psihologice şi sociologice umane (ale indivizilor izolaţi sau în grup) de care dispunem sunt incomplete pentru predicţia celei mai bune proiectări a unui anumit sistem care să fie utilizabil cu eficienţă

maximă. Utilizatorul va dori să realizeze noi lucrări cu sistemul după o familiarizare cu acesta; deci, cerințele beneficiarului se pot modifica după o perioadă de operare.

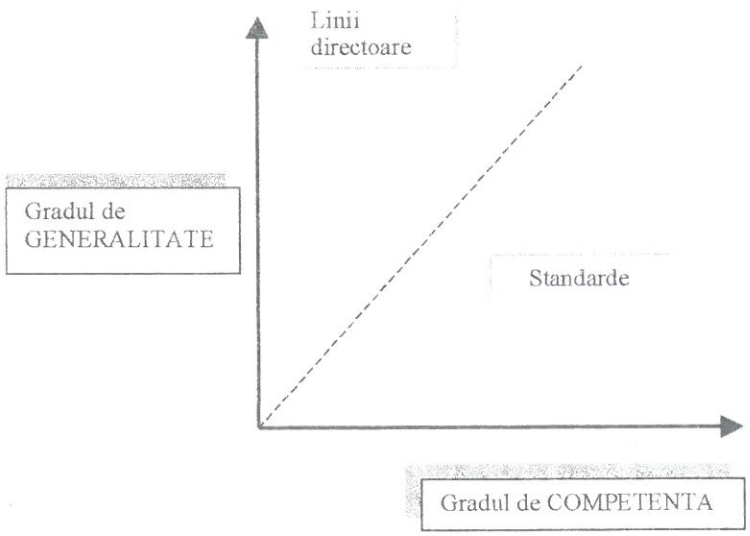
De asemenea, după cum se vede în figura de mai sus, fiecare etapă de proiectare poate influența o alta anterioară ei, datorită problemelor concrete de realizare pe care le întâmpină.

3.6. Reguli de proiectare

Regulile de proiectare, pe care un designer de sisteme interactive le poate urma în scopul creșterii eficienței în utilizare a sistemului proiectat, se pot clasifica de-a lungul a două dimensiuni, bazate pe *competență* și *generalitate* [6]. Prin *competență* se înțelege o indicație prin care se determină dacă o regulă este obligatoriu să fie urmată în proiectare sau este numai sugerată; prin *generalitate* se înțelege gradul prin care o regulă poate fi aplicată în multe situații de proiectare sau într-un număr restrâns. Este importantă și determinarea originii regulilor de proiectare. Cele două tipuri de reguli de proiectare sunt *standarde* și *linii directoare*.

Standardele sunt puternice în *competență* și limitate în aplicație, în timp ce liniile directoare tind să fie cu *competență* redusă și mult mai generale în aplicații:

Regulile de proiectare pentru sistemele interactive se pot întemeia pe *teoriile psihologiei cognitive, ergonomiei, sociologiei, economice sau computaționale*, ce pot avea sau nu rădăcini în *evidența empirică*.



Caracterizarea competentei si generalitatii pentru standarde si linii directoare, ca reguli in proiectare

Principiul care stă la baza acestor reguli este cel al *cauzalității*. Acest principiu se aplică atunci când ceva se întâmplă corect în urma unei acțiuni și apare ca și cum ar fi fost cauzat de acea acțiune.

Regulile de proiectare în sistemele de interacțiune om-calculator se bazează pe regulile de utilizare a obiectelor obișnuite, cu care oamenii iau contact zilnic (ex. butoane). Ei și-au dezvoltat "modele mentale" ale funcționării obiectelor, așa-numitele *modele conceptuale*; acestea se formează pe baza mapărilor, restricțiilor, principiului cauzalității, familiarității cu dispozitive similare, instrucțiuni și interacțiuni. Dacă la acțiuni similare, efectuate asupra a două obiecte se obțin rezultate similare (respectă principiul "cutici negre"), atunci, probabil că cele două obiecte sunt similare din punct de vedere conceptual.

Numărul posibilităților de creare a acestor reguli este limitat de o serie de restricții.

Restricții asupra regulilor de proiectare:

- *fizice* (accesări imposibile din punct de vedere fizic ale unor comenzi),
- *semantice* (semnificații diverse pentru diferite categorii de utilizatori),
- *culturale* (de ex. culoarea roșie, reprezintă pentru: America – pericol, Egipt – moarte, India – viață, China – fericire),
- *logice* (trebuie păstrată o mapare naturală între amplasarea componentelor și comenzile asupra lor).

Standardele pentru proiectarea sistemelor interactive sunt, de obicei, fixate de organisme naționale (ex.: BSI – *British Standard Institute*) sau internaționale (ex.: ISO – *International Standard Organisation*), punând de

acord reguli folosite de comunităţi mari. Există standarde atât pentru sistemele hardware, cât şi pentru produsele software, utilizate în interacţiune.

Liniile directoare pentru proiectarea sistemelor interactive sunt publicate în cataloage şi rapoarte tehnice; cu cât este mai abstractă o astfel de linie directoare, cu atât este mai potrivită aplicarea ei în etapa de proiectare *Cerinţe de specificare*; dacă este mai specifică, atunci se va aplica în etapa *Proiectare în detaliu*.

Iată, pentru exemplificare, categoriile de linii directoare de bază descrise de Smith şi Mosier [19]: date de intrare, afişarea datelor, secvenţa de control, ghidarea utilizatorului, transmisia datelor, protecţia datelor.

Cele două tipuri de interfeţe grafice utilizator (GUI), descrise în liniile directoare sunt: *OPEN LOOK* şi *Open Software Foundation* (OSF) *Motif*. Acestea reuşesc să menţină consistenţa utilizării în interiorul aplicaţiilor şi între aplicaţii, pe aceeaşi platformă de lucru.

Regulile de proiectare sunt mecanisme de restricţionare a spaţiului opţiunilor de proiectare, prevenind alegerea unor variante de proiectare, care ar face sistemul neutilizabil.

3.7. Ingineria utilizării eficiente *(Usability engineering)*

Ingineria utilizării eficiente sau a *uzabilităţii (Usability engineering)* încurajează încorporarea explicită a obiectivelor de utilitate ale produsului în procesul de proiectare, furnizând mijloace prin care poate fi judecată uzabilitatea produsului.

Proiectantul trebuie să fie capabil să *evalueze* din punctul de vedere al utilizării şi funcţionalităţii prototipurile iniţiale, pentru a le putea corecta caracteristicile în mod corespunzător.

Există mai multe *metode evaluare a proiectării*, care nu se exclud reciproc [6]:

- metoda cognitivă (în termenii psihologiei cognitive, din punctul de vedere al utilizatorului);
- metoda euristică (verificarea unui set de criterii de uzabilitate, formulate de experţi);
- metoda bazată pe analiza retroactivă (implică rezultate ale psihologiei experimentale şi evidenţei empirice);
- metoda bazată pe modele utilizator (ex. GOMS).

Procesul de proiectare este compus dintr-o serie de decizii asupra unei serii de sisteme potenţiale, procesul iterativ având ca rezultat sistemul ce va fi livrat, în final, beneficiarului. Acest proces se numeşte metoda *prototipizării rapide*. Proiectarea versiunilor noi de sisteme va ţine seama de rezultatele studiilor asupra utilizării sistemelor din generaţia anterioară.

4. Modele ale utilizatorului şi ale sistemului în proiectare

4.1. Modele ale utilizatorului în proiectare

Pentru o bună proiectare a interfeţelor software pentru anumite tipuri de utilizatori, proiectantul are nevoie de diverse *modele cognitive* de reprezentare a utilizatorului şi a acţiunilor acestuia.

Categorii ale modelelor cognitive:

- modele ierarhice (pentru reprezentarea structurii operaţiilor de interacţiune),
- modele lingvistice (reprezentarea gramaticii utilizator-sistem),
- modele fizice (deprinderi motorii umane).

Arhitecturile cognitive combină toate aceste modele, furnizând un model predictibil al utilizatorului pentru proiectarea unui anumit sistem de interacţiune.

Modelele cognitive pot fi *evaluative* (dau informaţii asupra faptului dacă un proiect dat are proprietăţi adecvate într-un anumit context de utilizare) sau *generative* (se dezvoltă în timpul procesului de proiectare).

Modelele cognitive ale utilizatorilor unei interfeţe cu calculatorul modelează aspecte ale înţelegerii, cunoştinţelor, intenţiilor sau ale acţiunilor utilizatorului asupra interfeţei. Nivelul reprezentării diferă de la tehnică la tehnică: de la modele cu obiective de nivel înalt şi rezultate ale activităţilor de rezolvare de probleme la descrieri de activităţi motorii (de exemplu modalitatea de apăsare a tastelor sau de clicare a mouse-ului).

Formalismele au fost dezvoltate de psihologi sau specialişti în calculatoare, al căror domeniu de interes este înţelegerea comportamentului utilizatorului.

Astfel, au fost definite două **tipuri de modele cognitive**:

- modele de competenţă: acele modele ce pot prezice "secvenţe de comportare legale" (tipuri de comportamente aşteptate de la utilizatori);
- modele de performanţă: acele modele ce descriu atât secvenţele de comportament necesare, cât şi nevoile de cunoaştere ale utilizatorului şi cum se reflectă acestea în realizarea lucrării (task-ului); astfel, acest tip de model furnizează o privire analitică asupra unor comportamente de rutină într-un număr limitat de aplicaţii.

Tipurile de modele cognitive se deosebesc şi prin activitatea la care se adresează: la formularea unui plan de activitate sau la execuţia acelui plan.

O parte din modele sunt legate de înţelegerea utilizatorului şi a limbajului lui de interacţiune asociat, o altă parte sunt în legătură cu *articularea* (traducerea între acest limbaj *task* şi limbajul de intrare - *input*).

În psihologia cognitivă, sunt utilizate des analogiile cu sistemele de calcul; acest lucru are ca avantaj facilitarea comunicării şi a analizei sistemului om-calculator, dezavantajul rămânând cel al riscului perceperii mecaniciste a utilizatorului.

Modelele ierarhice utilizează modele ale procesării mentale, în care utilizatorii ating obiectivele finale prin atingerea celor secundare, într-o modalitate "divide şi cucereşte".

Pot fi citate, din literatura de specialitate, două astfel de modele:

- GOMS (Goals, Operators, Methods, Selection) şi
- CCT (Cognitive Complex Theory).

Descrierea modelului GOMS constă în specificarea celor 4 elemente:

- Obiectivele (Goals) descriu ceea ce doreşte utilizatorul să obţină în final;
- Operaţiile (Operators) reprezintă nivelul cel mai de jos al analizei fiind acţiuni de bază ale utilizatorului;
- Metodele (Methods) - modalităţile de spargere a unui obiectiv de nivel înalt al utilizatorului în subobiective (posibilităţile de descompunere a acţiunilor utilizatorului pentru îndeplinirea unui task);
- Selecţia (Selection) – alegerea unei metode de realizare a task-ului; această alegere depinde utilizator, starea sistemului, detalii asupra obiectivelor.

Modul de descompunere a obiectivului de nivel înalt al utilizatorului în subobiective implică înţelegerea detailată a strategiilor utilizatorului în rezolvarea problemelor precum şi cunoaşterea domeniului aplicaţiei cu care acesta interacţionează.

Modelul original GOMS (al lui Card, Moran şi Newell-[3]) a servit ca model de bază pentru multe cercetări în domeniul modelelor cognitive HCI. A fost bun în descrierea activităţilor de rutină, desfăşurate de experţi; luând în considerare şi modelele dispozitivelor fizice, poate fi folosit în prezicerea performanţelor acestor utilizatori în termenii timpilor de execuţie.

Modelul CCT a fost introdus de Kieras şi Polson [10] şi începe cu premisele de bază ale descompunerii obiectivului modelului GOMS, îmbogăţindu-l, însă, printr-o creştere a predictivităţii. CCT descrie, în paralel, cele două părţi ale interacţiunii: acţiunile utilizatorului şi reacţia sistemului de calcul. Descrierea obiectivelor se bazează pe ierarhia GOMS, dar este făcută utilizând "reguli de producţie". Aceste reguli reprezintă o succesiune de tipul:

if <condiţie> *then* <acţiune>

Condiţia este o instrucţiune despre conţinutul memoriei de lucru a utilizatorului; dacă această condiţie este adevărată, regula de producţie dă liber execuţiei acţiunii; acţiunea poate fi formată din una sau mai multe acţiuni elementare, care vor produce schimbări în memoria operativă (de lucru) sau acţiuni externe (cum este apăsarea tastelor).

Scopul acestui model este de a fixa reguli de utilizare pentru novici; ei vor apăsa aceleaşi taste ca şi experţii, dar modalitatea de memorare a cunoştinţelor va fi diferită.

Modelele lingvistice se referă la formalismele de modelare a interacţiunii om-calculator în termeni lingvistici. Aceste modele, deşi similare în forma notaţiilor de proiectare a dialogului, au fost propuse cu intenţia de a ţine seama de comportamentul utilizatorului în analiza dificultăţilor de înţelegere ale interfeţei.

Pentru specificarea dialogului sunt frecvent utilizate gramaticile BNF (*B*ackus-*N*our *F*orm) şi gramatica TAG (*T*ask-*a*ction grammar); ambele sunt prezentate în [6].

4.2. Modele ale sistemului în proiectare

Există formalisme standard, de descriere a sistemelor informatice, proiectate pentru interacţiunea om-calculator. Aceste formalisme au devenit o practică uzuală în ingineria programării şi reprezintă o modalitate de

clarificare a ideilor proiectanţilor de sisteme pentru cei care le implementează, deci, pentru dezvoltatorii de hardware şi software pentru un anumit sistem de interacţiune.

Din punctul de vedere al programatorului, scopul utilizării unui limbaj formal de specificare este acela de a te putea detaşa de detaliile de implementare, în procesul rezolvării unei probleme, şi de a crea programe a căror corectitudine poate fi demonstrată [12].

Notaţiile formale sunt utilizate pentru specificarea funcţionalităţii şi prezentării unui anumit sistem.

Problemele esenţiale la care trebuie să ofere soluţie un limbaj formal sunt:

- elementele limbajului, care trebuie să fie neambigue, adică să aibă un sens strict definit şi
- descrierea exhaustivă a modificării stării sistemului determinate de acţiunile utilizatorului.

Astfel de formalisme standard sunt prezentate pe larg în literatura de specialitate De exemplu, în [6] este descrisă utilizarea limbajului formal *Z*. Atât din punct de vedere sintactic, cât şi semantic, *Z* se bazează pe notaţiile şi conceptele matematicii clasice, respectiv, logica bivalentă şi teoria mulţimilor, utilizând astfel, abstractizări procedurale şi de reprezentare.

Z este un formalism orientat spre model. Într-un astfel de limbaj, există notaţii formale de specificare a comunicării cu sistemul (verifică înlăturarea ambiguităţilor în descrierea sistemului), iar alte notaţii sunt pentru analiza specificaţiilor (verifică două tipuri de consistenţe: consistenţa internă în funcţionare şi cea externă vizavi de alte cerinţe (de ex. de securitate).

Alte notaţii de specificare prezentate în [6] sunt cele *algebrice* (*OBJ*, *Larch* şi *ACT-ONE*), precum şi cele cu *logică temporală.*

Cele 3 tipuri de formalisme (*Z*, algebrice şi temporale) sunt descrise în [20].

Alegerea unui anumit tip de formalism pentru modelarea sistemului este în funcţie de tipul aplicaţiei care urmează a fi implementată.

5. Probleme specifice în proiectarea sistemelor de cooperare distribuită (CSCW)

Proiectarea sistemelor de cooperare necesită o înţelegere profundă a lucrului în cooperare, în cadrul grupurilor sau organizaţiilor, implicând atât artefactele, cât şi convenţiile sociale. În domeniul proiectării acestor sisteme, contribuie multe discipline: psihologia, ergonomia, lingvistica, ştiinţele calculatoarelor (sistemele informaţionale, cele de suport de luare a deciziilor, ingineria de cunoştinţe, interfeţele utilizator, inteligenţa artificială distribuită), sociologia, ştiinţele de organizare şi management.

Există două abordări metodologice opuse pentru proiectarea sistemelor de cooperare distribuită: o abordare "bottom-up", cunoscută ca pragmatică, iar cealaltă de tip "top-down", bazată pe teorii şi modele.

Abordarea pragmatică, numită şi "metoda prototipizării rapide", constă în dezvoltarea unui sistem iniţial, căruia i se îmbunătăţesc performanţele iterativ, prin experienţe de utilizare în practică, până la obţinerea produsului final.

Abordarea bazată pe modele teoretice constă în înţelegerea problemei printr-o "teoretizare" a domeniului de aplicaţie, ghidând proiectarea şi chiar dezvoltarea producţiei acestor sisteme conform acestor modele teoretice.

Sistemele de cooperare se încadrează în categoria mai largă, a sistemelor interactive om-calculator, al căru proces de proiectare a fost descris în capitolul precedent.

O dată cu creşterea complexităţii sistemelor software, a devenit necesară proiectarea arhitecturilor software. În sens general, o *arhitectură software reprezintă o descriere la nivel înalt a proprietăţilor modulelor sistemului (adică a componentelor şi interacţiunilor lor)*. La acest nivel de descriere, poate fi înţeleasă mai bine proiectarea ansamblului, descrierea arhitecturală scoţând în evidenţă restricţiile de proiectare, precum şi motivarea alegerii unui anumit tip de arhitectură.

O trăsătură de bază a sistemelor groupware o constituie *arhitectura distribuită*, ea definind care dintre părţile aplicaţiei rulează pe un server central, care dintre ele rulează (descentralizat) în noduri ale reţelei şi cum sunt interconectate logic toate aceste părţi.

Alegerea unei anumite arhitecturi distribuite are o influenţă majoră asupra modului cum poate fi dezvoltată şi utilizată aplicaţia groupware.

5. Remarci finale

Ca şi în cazul altor tehnologii, şi în cel al interfeţelor are loc un *proces de standardizare, dar, în acest caz apare şi tendinţa inversă, spre personalizare*. Într-adevăr, calculatorul a devenit tot mai personal, configurat software după preferinţele şi necesităţile utilizatorului, însă protocoalele de comunicare în reţea, compatibilităţile formatelor fişierelor utilizate trebuie să se păstreze.

Pentru *aplicaţiile sistemelor CSCW* este necesar un anumit tip de *transparenţă* şi de *adaptare în funcţie de necesităţile utilizatorului* ("tailoring") pentru a fi eficiente.

O dată cu creşterea utilizării sistemelor computerizate, aplicaţiile software nu sunt concepute pentru a fi utilizate de un singur tip de utilizator, pentru o singură sarcină ("task"). În schimb, software-ul comercial asigură funcţionalitatea unei întregi palete de astfel de sarcini. Un astfel de software generic nu răspunde necesităţilor tuturor utilizatorilor lui, imediat după instalare, dar oferă acestora mijloacele de adaptare conform cerinţelor lor. Activitatea de adaptare a software-ului după instalarea sa, pentru a satisface exigenţele utilizatorului poartă denumirea de "tailoring". Aceste exigenţe pot fi legate de preferinţele personale, caracteristicile *task*-urilor sau modificările locului de muncă. Handerson & Kyng [8] disting 3 niveluri de *tailoring*: alegerea unui mod de acţiune al software-ului din cadrul mai multor alternative, construirea unor noi tipuri de comportament din informaţiile existente şi modificarea artefactului.

Evoluţia tehnologiei în materie de echipamente de calcul va fi dublată de o evoluţie rapidă a facilităţilor de transmisie de date. Prin urmare, este utilă crearea unor noi modele software, care să aibă în vedere performanţe de transfer, superioare celor prezente. Evoluţia reţelelor din ultimii ani este caracterizată de *mobilitatea* utilizatorilor de servicii şi de *mobilitatea* componentelor software, care realizează aceste servicii.

Noile instrumente de interacţiune (sunetul, gesturile, *pen*-ul, *video*-ul, vorbirea, agenţii, obiecte ale realităţii virtuale) nu au avut impactul imediat scontat, asupra interfeţelor utilizator. Modificările vor fi progresive. Complexităţile aplicaţiilor şi compatibilitatea datelor vor mai acţiona un timp scurt pe post de frâne, forţând firmele să evalueze cu grijă modificările din sistemele de operare.

Totuşi, interfeţele utilizator ale serviciilor de reţea îşi propun să iasă din acest tipar, promovând flexibilitatea care decurge din posibilitatea de a actualiza dinamic şi, eventual, transparent pentru utilizator componentele software, care îi facilitează accesul la informaţiile din reţea, de a le face portabile şi interoperabile, prin standardizare.

Tehnologia care a revoluţionat de curând domeniul aplicaţiilor distribuite, Java [9], se află, în momentul de faţă, într-o fază de încercare de creare rapidă a unor standarde, cu aliniere la cele deja existente, în speţă a standardului CORBA (*Common Object Broker Request*) [5]. Speranţa definirii unui sistem de operare de reţea nativ Java va deveni foarte probabil, în următorii ani, realitate. Încă din 1996, autorul articolului [13] lansează o ipoteză interesantă: *noul sistem de operare universal de reţea ar putea fi de tip browser*. Argumentarea acestei idei se bazează pe caracteristicile protocolului HTTP (*HyperText Transport Protocol*): toate resursele din reţea sunt văzute în mod uniform (ca fişiere); protocolul este simplu, manipulând caractere ASCII; poate fi folosit în regim de tranziţie; este simplu de implementat.

Noua generaţie de servere HTTP va include o *maşină de stare*, care va permite păstrarea conexiunilor, va dispune de un subsistem de securitate a datelor şi un subsistem *cache*. Problema care mai rămâne de rezolvat este ca, prin evoluţia sa, pe lângă posibilităţile de cuplare la reţea pe care le oferă, browserul să fie capabil să gestioneze toate resursele unei maşini hardware. Netscape încorporează în acest moment Java şi Java script, permite în acelaşi timp gestionarea sistemului I/E pentru a efectua convorbiri telefonice, vizionarea de filme, audiţie de muzică, grafică 3D şi alte facilităţi.

Numai cu câţiva ani în urmă, până la apariţia calculatoarelor personale, sistemele de interacţiune om-calculator erau centralizate, dedicate calculatoarelor mainframe sau minicalculatoarelor. O dată cu extinderea reţelelor locale de calculatoare (LAN), s-au dezvoltat sistemele multiutilizator, reprezentate de aplicaţiile de cooperare distribuită, de tip CSCW. Prin interconectarea la nivel planetar a reţelelor locale, multe grupuri de lucru se formează dinamic, prin parteneriat de afaceri, în funcţie de cerinţele "pieţei". Companiile îşi desfăşoară activitatea principală în cadrul *profilului propriu de afaceri*, iar activităţile secundare reprezintă *relaţiile de cooperare cu alte organizaţii*. Ca rezultat, se formează *organizaţii virtuale noi*. Pentru ca funcţionarea acestor noi organizaţii să devină efectivă, infrastructurile informaţionale ale partenerilor de afaceri trebuie să fie interconectate. Elementul cheie în acest proces îl reprezintă **sistemele de management al fluxului de lucru**, care controlează procesul de schimb de informaţii. Sistemele interconectate ale fluxului de lucru trebuie să permită unei organizaţii (consumatoare de servicii) să pornească un proces (un serviciu) în interesul propriu, în cadrul altei organizaţii (furnizoare de servicii) şi să primească rezultatele acestui proces.

Metalimbajul XML [23] *a devenit cadrul de standardizare a comunicațiilor dintre sistemele de colaborare în Internet.* În [7] sunt evidențiate eforturile de standardizare în domeniu ale asociației *Workflow Management Coalition* [22]. Modelul de referință propus [14] utilizează standardul de interoperabilitate a fluxului de lucru în Internet *Wf-XML*. Wf-XML, bazat pe metalimbajul XML, furnizează o arhitectură bazată pe mesaje de comunicare între motoarele de flux de lucru.

Noile modele de interacțiune vor servi mai bine diversele categorii de oameni. Extinderea la nivel mondial a utilizării calculatoarelor interconectate nu ar trebui să reprezinte un pericol de desocializare; din contră, noile tehnologii permit apropierea dintre oameni atât spațial și temporal, cât și spiritual, cu mijloace specifice. Secolul următor este de neconceput fără accesul universal la informație, mediat de calculator. Rețeaua internațională de calculatoare va fi o rețea "umană", utilizatorii, prin interacțiunea lor, vor face parte integrantă din structura și funcționalitatea ei.

Bibliografie

1. **ANDREWS, K.**: Course on Human Computer Interaction, Graz University of Technology, http://www.iicm.edu/hci/, 1999.
2. **BUDIU, M.**: Tuneluri (tunneling), 11 martie 1999, http://www.cs.cmu.edu/~mihaib/
3. **CARD, S.K., T.P. MORAN, A. NEWELL**: The Keystroke-Level Model for User Performance with Interactive Systems. În: Communications of the ACM, 23, 1980, pp.396-410.
4. **CECAL, L.**: Cuvânt înainte: Tendințele interfețelor în proiectarea interfețelor utilizator de Alan Cooper, trad. Liana Cecal, București, Editura Tehnică, 1997.
5. **KEAHEY, K.**: A Brief Tutorial on CORBA, http://www.cs.indiana.edu/hyplan/kksiazek/tuto.html
6. **DIX, A., J. FINLAY, G. ABOWD, R. BEALE**: HUMAN-COMPUTER INTERACTION, Second Edition, Prentice Hall International (UK) Limited, © 1998.
7. **HAYES, J.G. et al.**: Workflow Interoperability Standards for the Internet. În: Internet Computing, May/June 2000, pp.37-45.
8. **HENDERSON, A., M. KYNG**: There's no Place like Home: Continuing Design in Use. În: J. Greenbaum; M. Kzng, Design at Work-Cooperative Design of Computer Systems. Lowrence Erlbaum Associates, Hillsdale, NJ, 1991, pp. 219-240.
9. [Java, *] *Java Technology*, http://java.sun.com/.
10. **KIERAS, D.E., P.G. POLSON**: An approach to the formal analysis of user complexity. În: International Journal of Man-Machine Studies, 22: 1985, pp.365-394.
11. [Mbone, *] Rețeaua *Multicast Backbone*: http://www.mbone.com; arhitectura rețelei Mbone: http://www.cs.berkeley.edu/~elan/mbone.html.
12. **NEGRANU, L.**: Prelucrarea specificațiilor formale cu aplicație în proiectarea asistată a programelor. Teză de doctorat, UPB,1999.
13. **NELU, M.**: Sisteme de operare: Quo Vadis? În: BYTE România - iunie 1996.
14. [RefMWf, *] http://www.aiim.org/wfmc/standards/docs/tc003v11.pdf
15. **SCHNEIDERMAN, B.**: The future of interactive systems and the emergence of direct manipulation. În: Behaviour and Information Technology, 1(3): 1982, pp.237-256.
16. **SCHNEIDERMAN, B.**: Designing the User Interface: Strategies for Effective Human Computer Interaction, Addison-Wesley, New York, 1987.
17. [SIGCHI,*] Special Interest Group on Computer-Human Interaction http://sigchi.design.nl/english/sigchi.html
18. [SIGCHIR, *] SIGCHI Romania: http://www.ici.ro/chi-romania
19. **SMITH, S.L., J.N. MOSIER**: Guidelines for designing user interface software. Mitre Corporation Report MTR-9420, Mitre Corporation, 1986.
20. **TRIFĂNESCU, D., CR. NICULESCU, O. POPESCU**: Analiza interfeței om-calculator în contextul particular al serviciilor de rețea, Subprogramul de cercetare: Dezvoltări în arhitectura serviciilor din rețelele de calculatoare, Raport de cercetare CCAIAPLNMC, decembrie, 1997.
21. [Wfmc, *] http://www.aiim.org/wfmc/mainframe.htm
22. [Wfmc, *] http://www.aiim.org/wfmc/mainframe.htm
23. [XML, *] Extensible Markup Language (XML) 1.0; World Wide Web Consortium Recommendation, http://www.w3.org/TR/REC-xml.