

VALIDAREA DATELOR DE INTRARE ÎN APLICAȚIILE INFORMATICE ORIENTATE SPRE CETĂȚEAN

Ion Ivan

ionivan@ase.ro

Cristian Ciurea

cristian.ciurea@ie.ase.ro

Daniel Milodin

daniel.milodin@ase.ro

Academia de Studii Economice, București

Rezumat: Se definește conceptul de informatică orientată spre cetățean. Se prezintă caracteristicile aplicațiilor informatice construite pentru cetățeni. Se definesc proceduri pentru construirea datelor de intrare. Se stabilesc tipurile datelor de intrare în aplicațiile informatice orientate spre cetățean și modalitățile de validare a acestora. Se analizează efectele dezvoltate de procedurile de validare eficientă.

Cuvinte cheie: informatică pentru cetățean, aplicații informatice, validare date, proceduri de validare.

Abstract: There is defined the concept of informatics oriented towards citizen. There are presented the characteristics of informatics applications built for citizens. There are defined procedures for building entry data. There are established the types used for entry data in informatics application oriented towards citizens and ways of validation. There are analyzed the effects developed by the procedures of efficient validations.

Keywords: informatics oriented towards citizen, informatics applications, data validation, validation procedures.

1. Informatica orientată spre cetățean

Societatea bazată pe cunoaștere presupune creșterea la un nivel foarte ridicat a interacțiunilor om-calculator în vederea soluționării de probleme ale cetățenilor, precum:

- plăți impozite, către bugetul de stat și bugetele locale;
- plăți mărfuri, comandate sau nu pe internet;
- obținerea de aprobări;
- efectuarea testărilor de cunoștințe;
- instruire, prin utilizarea aplicațiilor e-learning;
- rezervări și achiziții de bilete;
- programări în audiențe;
- comenzi de haine, lansate pe internet;
- analize medicale, în vederea stabilirii grupei sanguine sau a stării de sănătate.

Cercetările privind dezvoltarea societății informaționale au la baza rezultatele obținute în următoarele domenii:

- construirea de hardware caracterizat prin viteză de prelucrare foarte mare, capacitate de memorare, de asemenea, foarte mare și o varietate de modalități de achiziție a datelor, care elimină orice restricție legată de încărcarea bazelor de date, din punct de vedere cantitativ, dar, mai ales, din punct de vedere calitativ;
- dezvoltarea de software de către specialiști cu înaltă calificare, ce stăpânesc cerințele lucrului în echipă;
- orientarea spre utilizarea informaticii în vederea soluționării complete a problemelor, în mod direct, de către utilizatori, prin faptul că produsele - program funcționează independent de elaboratori, oferind rezultate complete și corecte pentru date de intrare, fără a presupune elemente de asistență de specialitate; accesul la toate resursele este on-line;
- securitatea informatică la nivelul protecției utilizatorilor, la nivelul bazelor de date, la nivelul software și la nivelul derulării proceselor;
- schimbarea concepției privind colectarea de date, prelucrarea volumelor mari de date și definirea modelelor pentru care se urmărește construirea de algoritmi de optimizare;
- integrarea bazelor de date pentru a gestiona redundanța aplicațiilor informatice, generatoare de fluxuri continue de prelucrare; dacă o nouă aplicație informatică, orientată spre cetățean, necesită elemente existente deja în baze de date aflate în uz curent, atunci se referă acele baze de date în mod obligatoriu, integrarea bazelor de date fiind o condiție de existență a aplicației.

Obiectivul informaticii orientate spre cetățean este acela de a dezvolta aplicații foarte diverse și care închid ciclul specific unei acțiuni. În acest context, împrumutul unei cărți de la bibliotecă se realizează în două modalități, astfel:

- cititorul intră pe site-ul bibliotecii, își creează un cont de utilizator, consultă lista cărților disponibile, selectează o anumită carte care îi va fi livrată la domiciliu;
- cititorul merge la bibliotecă, accesează baza de date cu cărțile disponibile, selectează cartea care îl interesează, iar această carte îi este pusă la dispoziție automat prin intermediul unui mecanism de manevrare a cărților.

Pentru rezervarea și achiziția unor bilete de avion, etapele de parcurs sunt următoarele: se intră pe site, se aleg biletele și se achită on-line, iar, în final, se obține o confirmare electronică având aceeași valoare ca un bilet cumpărat de la o agenție.

Obținerea unui anumit tip de autorizație se realizează astfel: se accesează site-ul și se selectează categoria de autorizație solicitată, se plătește on-line, se intră în baza de date a agenției care gestionează eliberarea de autorizații, se analizează electronic procesul și se primește autorizația respectivă.

Băncile pun la dispoziția clienților un serviciu prin care acestea efectuează confirmarea electronică, în mod automat, a plăților efectuate de client în favoarea anumitor birouri vamale, în scopul obținerii liberului de vamă. Pentru plățile electronice, efectuate de client către birourile vamale, banca livrează un fișier electronic, care reprezintă, pentru birourile vamale, dovada stingerii datoriei de către client. În acest fel, nu mai este necesară prezentarea, de către client, a exemplarului verde al ordinului de plată pe suport hârtie.

Avantajele unor astfel de aplicații sunt următoarele:

- aplicațiile lucrează independent de dezvoltator, de proprietar;
- aplicațiile nu produc greșeli;
- persoanele care utilizează aceste aplicații lucrează cu ele fără efort și fără documentare în prealabil;
- aplicațiile folosesc baze de date care interacționează; baza de date cu populația țării interacționează cu baza de date a diplomelor de învățământ și cu baza de date a întreprinderilor;
- transparența este foarte mare, pentru că sunt puse la dispoziția utilizatorului toate informațiile de care are nevoie.

Societatea bazată pe cunoaștere presupune:

- *existența de baze de date compatibile*, realizate independent, pentru subcolectivități, care, prin concatenare fără modificări esențiale, permit efectuarea de prelucrări ce privesc întreaga colectivitate; în cazul în care se procedează la dezvoltarea de noi aplicații informatice, care vor fi supuse proceselor de agregare, atunci se vor defini șabloane și structuri pe care toți elaboratorii trebuie să le respecte;
- *definirea de vocabulare și de mecanisme de construire a acronimelor* pornind de la texte realizate folosind cuvintele din vocabular, obținându-se, în acest fel, o modalitate foarte eficientă de construire a cheilor de căutare a informațiilor necesare soluționării de probleme ale cetățenilor; lista de cuvinte conține denumiri de țări, denumiri de orașe; la această listă, se adaugă cuvinte ce corespund profesiilor, obiectelor foarte uzuale, acțiunilor comune și tipurilor de locații frecvent utilizate (spital, gară, tren, școală, muzeu, primărie, poliție, vreme, hartă);
- *construirea de reguli pentru designul de interfețe ale aplicațiilor*, astfel încât toate aplicațiile care necesită aceleași structuri de date de intrare sau care oferă aceleași structuri de conținut digital să implementeze interfețe care nu diferă semnificativ prin cuvintele cheie, prin poziția acestora și, mai ales, prin dimensiunea textelor utilizate; se asigură, în acest fel, continuitatea la nivelul tuturor aplicațiilor; un cetățean familiarizat cu o anumită aplicație, frecvent referită, va realiza o trecere fără efort la o nouă aplicație, dacă aplicația nouă preia în proporție foarte mare conținutul interfeței de la vechea aplicație;
- *orientarea spre cerințele naturale*, formulate de utilizatori, ale întregului flux de prelucrare; dacă societatea preinformațională conținea fluxuri definite de către organizații, iar cetățeanul se supunea regulilor care avantajează exclusiv echipa din organizație, noul context creat vine să modifice radical poziția cetățeanului, ceea ce determină ca structurarea aplicațiilor informatice să fie efectuată strict pe cerințele cetățeanului; nivelurile de descriere a caracteristicilor referitoare la cetățean se află dispersate în mai multe baze de date; orice aplicație, care referă niveluri ale caracteristicilor existente în aceste baze de date, este obligată să lucreze cu bazele de date existente, ceea ce presupune o

schimbare radicală a comportamentului proprietarilor de baze de date, dar și a utilizatorilor de baze de date; structurile înregistrărilor pentru baze de date referitoare la colectivități devin publice, iar creatorii de noi aplicații informatice orientate spre cetățean selectează, în funcție de cerințele algoritmilor de prelucrare, bazele de date necesare, plătesc accesul la acestea și elimină reintroducerile de date;

- *armonizarea intereselor* celor care produc software cu cerințele, cât mai variate, de dezvoltare a aplicațiilor informatice orientate spre cetățean, pentru a acoperi marea diversitate a cerințelor de soluționare de probleme ale cetățenilor; acum, există foarte multe aplicații informatice pentru contabilitatea întreprinderii, pentru calculul de salarii, pentru gestiunea de stocuri, pentru contracte și pentru aprovizionări; nu sunt acoperite aplicații de interes major privind instruirea on-line, accesarea de resurse, alocarea temporară de resurse, informarea pe domenii, efectuarea de calcule de comportament, desfășurarea de tranzacții; dispersat, cu mari dificultăți de regăsire, cu abordări parțiale, există soluții parțiale la unele dintre aceste probleme; cetățeanul, rămas pe un plan secund, are soluții care nu țin seama de exigențele sale prin faptul că numele site-urilor au legătură slabă cu conținutul, acronimele folosite au la bază mecanisme de realizare necunoscute și faptul că nu este acoperită întreaga colectivitate; toate referirile fără succes efectuate de către cetățeni accelerează scăderea gradului de satisfacție a acestora în fața aplicațiilor informatice; armonizarea presupune crearea de programe bazate pe proiecte, astfel încât finanțările să fie dirijate strict spre crearea și implementarea de aplicații informatice orientate spre cetățean, care să acopere arii pentru care nu există soluții în plan informatic.

Literatura de specialitate conține lucrări, cărți de inginerie software ca [6] și [10], cărți de managementul calității software, precum [12], care dezvoltă, în plan teoretic și practic, principalele cerințe de a proiecta, realiza și implementa aplicații informatice performante. Abordările sunt unilaterale, vizând alocarea și nivelarea de resurse strict din punctul de vedere al celor care construiesc aplicațiile.

2. Caracteristicile aplicațiilor informatice orientate spre cetățean

Aplicațiile informatice orientate spre cetățean se deosebesc de restul aplicațiilor prin faptul că se dezvoltă pornind de la grupul țintă, și nu de la cerințele proprietarului. Dacă o bancă dorește să dezvolte o anumită aplicație informatică, atunci respectiva aplicație va fi construită pe baza specificațiilor elaborate de bancă. Noile generații de aplicații informatice se construiesc pe baza dorințelor cetățenilor. Se definesc specificații care îl satisfac pe cetățean, pe utilizator, și nu pe proprietar.

Ciclul de dezvoltare al unei aplicații informatice orientată spre cetățean este următorul:

- se definește grupul țintă de către proprietarul aplicației și dezvoltatorul acesteia;
- se stabilesc obiectivele grupului țintă, în scopul definirii problemei de rezolvat;
- se elaborează specificații pentru a satisface dorințele grupului țintă;
- se dezvoltă un cod pentru aplicație;
- se testează aplicația de către grupul țintă.

La proiectarea aplicațiilor distribuite este analizat grupul țintă, întrucât caracteristicile de calitate ale acestora depind de persoanele care accesează resurse definite și care, prin gradul de satisfacție obținut după soluționarea on-line a problemelor, determină dezvoltarea ulterioară a altor aplicații.

Aplicațiile informatice orientate spre cetățean au următoarele caracteristici de calitate:

- *complexitatea* software este o caracteristică evidențiată numai prin comparația între două sau mai multe produse software; un sistem poate fi privit ca un graf în care nodurile sunt reprezentate de subsisteme, iar arcele de legăturile dintre subsisteme; complexitatea software în sens McCabe presupune asocierea fiecărui program a unui graf, în care nodurile corespund instrucțiunilor, iar arcele marchează sensul trecerii execuției de la o instrucțiune la alta;
- *fiabilitatea* care constă în capacitatea unei aplicații de a-și menține nivelul de performanță în anumite situații date, pentru un interval specificat de timp; fiabilitatea este privită ca fiind măsura încrederii în concepția și în capacitatea unui produs software de a funcționa corect în toate condițiile avute în vedere de la început;
- *interfețe prietenoase*, care presupun navigări facile în cadrul meniurilor aplicației, și care nu necesită documentație de utilizare.

Etapele ciclului de dezvoltare trebuie să includă:

- *analiza grupului țintă*, cu luarea în considerare a numărului estimat de persoane care accesează resursele aplicației, structura pe sexe, pe vârste, pe niveluri de calificare a persoanelor ce alcătuiesc grupul țintă; se vor lua în considerare și aptitudinile care influențează persoanele în a aborda accesul la resursele aplicației informatice; toate acestea permit elaborarea unui studiu complet privind comportamentul submulțimilor grupului țintă, în diferite ipostaze; se estimează dimensiunea eșantionului reprezentativ și se planifică toate operațiile de confirmare a comportamentului aplicației raportate la accesarea de către membrii eșantionului a resurselor aplicației;
- *definirea problemei de rezolvat* se realizează pornind de la cerințele reale, formulate de membrii grupului țintă; beneficiarul aplicației are nevoie de anumite rezultate finale; utilizatorii cetățeni au capacitatea de a furniza anumite date de intrare; dacă există diferențe între datele de intrare pe care le furnizează cetățenii, în mod natural, și datele de intrare necesare obținerii rezultatelor de care au nevoie beneficiarii aplicației informatice, atunci noul concept de inginerie software, orientat spre cetățean, impune ca organizația beneficiară a aplicației să plătească accesul la bazele de date existente pentru a completa deficitul de informație; când lucrurile stau invers, beneficiarul de aplicație are responsabilitatea de a-și soluționa problema fără a transfera la nivelul cetățeanului minusurile de informație, transformându-l pe cetățean într-un element dependent de organizație; dezvoltatorul de aplicații are rol pasiv în această etapă, înregistrând toate cerințele grupului țintă, impunând beneficiarului aplicației soluționarea problemelor de acces la alte baze de date pentru a avea informații de intrare complete și corecte ale problemei de rezolvat; rolul pasiv merge mai departe, și anume, în ideea că elaboratorul orientează întregul ansamblu de tehnici, metode, limbaje, personal, echipamente, seturi de date de test spre a soluționa exigențele cetățeanului, ci nu spre a satisface cerințele beneficiarului sau pentru a obține profit în detrimentul beneficiarului, prin transferul de efecte negative, generate de o soluție neadecvată, către numeroșii cetățeni care activează resursele neadecvate ale aplicației, în mod repetat;
- *elaborarea de specificații* reflectă, în primul rând, exigențele membrilor grupului țintă; prin folosirea unui limbaj riguros, se procedează la descrierea problemei de rezolvat ca problemă a membrilor grupului țintă;
- *design-ul aplicației* are la bază cerințele cetățenilor de a realiza o comunicare, în mod natural, și de a dezvolta fluxuri de prelucrare cât mai scurte; se analizează modul în care membrii grupului țintă doresc să soluționeze problemele, fără a le impune acestora restricțiile organizației beneficiare sau restricții de ordin tehnic, specifice domeniului informatic; structura aplicației trebuie să reflecte gradul de flexibilitate pe care îl văd membrii grupului țintă; pornind de la exigențele grupului țintă, proiectarea interfețelor, vocabularele utilizate, modulele de prelucrare, modulele de prezentare a rezultatelor, se realizează astfel încât toate prelucrările să coincidă cu modul în care cetățenii doresc să decurgă interacțiunea om – calculator; aceștia trebuie să remarce ușurința și naturalitatea cu care se ajunge la resurse, astfel încât rata de succes a interacțiunii să fie maximă;
- *codificarea* se realizează respectând nivelurile de calitate impuse pentru proceduri, incluzând vocabularele acceptate de cetățeni, generând texte foarte apropiate de limbajul natural pentru a facilita înțelegerea semnificației rezultatelor; asistarea procesului de codificare are menirea de a prelua în cod toate informațiile existente în specificații pentru a realiza o suprapunere cât mai bună a ceea ce face fiecare modul scris în raport cu ceea ce solicită cetățenii;
- *testarea modulelor și testarea întregii aplicații informatice orientate spre cetățean* se efectuează folosind seturile de date de test, create pe baza informațiilor provenind de la membrii grupului țintă; beneficiarul aplicației și dezvoltatorii aplicației creează seturi de date de test suplimentare, care se adresează laturilor tehnice ale aplicației și care, în raport cu seturile din specificațiile inițiate de cetățeni, aduc un spor de calitate global aplicației, fără a fi în contradicție cu exigențele cetățenilor.

3. Date de intrare în aplicații informatice orientate spre cetățean

Traversarea etapelor definite prin tehnologie garantează obținerea de aplicații orientate spre cetățean, caracterizate prin eficiență, prin operaționalitate și, mai ales, prin maximizarea gradului de satisfacție a membrilor grupului țintă.

Aplicațiile informatice orientate spre cetățean sunt accesibile pe internet sau în cadrul unei rețele de calculatoare folosite de grupul țintă. Aceste aplicații dispun de o serie de formulare care permit introducerea de date și selectarea unor valori sau opțiuni. În varianta selectării unor valori sau opțiuni,

lista elementelor de selectat trebuie să fie completă, adică să cuprindă toate valorile existente pentru câmpul respectiv. Spre exemplu, selectarea tipului de card bancar se face din lista următoare: Visa, Mastercard, Maestro. Nu trebuie să existe situația în care un utilizator să dețină un alt tip de card decât cele menționate în lista elementelor de selectat.

Titularul unui card bancar completează un formular având două câmpuri, unul în care se introduce numele, iar celălalt în care se introduce prenumele persoanei. Valorile din cele două câmpuri nu sunt comutative. Validarea acestor câmpuri trebuie să cuprindă și situația introducerii a trei sau mai multe litere identice consecutive, caz în care acel nume sau prenume este considerat invalid.

Validarea câmpurilor în care se introduce text liber, cum ar fi câmpul destinat comentariilor, presupune definirea unui vocabular de cuvinte interzise. Textul introdus nu trebuie să conțină cuvinte sau subșiruri de caractere din respectivul vocabular. Astfel, se evită completarea în câmpurile care conțin text liber a cuvintelor obscene sau a elementelor de jargon.

Un simpozion conține secțiuni de teme, fiecare secțiune având o denumire și o listă de cuvinte cheie. Validarea încadrării unui articol într-o anumită secțiune impune ca titlul articolului să conțină cuvinte din lista de cuvinte cheie a respectivei secțiuni. În câmpul de pe formular, destinat completării abstractului acelui articol, se stabilește o limită reprezentată ca număr de caractere sau număr de cuvinte de completat. Se realizează, astfel, validarea lungimii textului introdus.

Formularele au câmpuri obligatorii și opționale. Datele de intrare sunt reprezentate de șiruri de litere, date calendaristice, coduri sau valori numerice. Aceste date, pe măsura completării lor, se încadrează în următoarele categorii:

- sunt corecte și complete: pe un formular se află câmpurile de completat $C_1, C_2, \dots, C_n$; fiecare câmp C_i are un domeniu de valori V_i ; situația de corect și complet apare atunci când toate câmpurile $C_i, i=1..n$, aparțin vocabularelor $V_i, i=1..n$;
- sunt corecte, dar incomplete: situație în care toate câmpurile $C_1, C_2, \dots, C_{i-1}, C_{i+1}, \dots, C_n$ aparțin vocabularelor $V_i, i=1..n$, dar lipsește câmpul C_i ; în acest caz, aplicația generează mesaje care indică acele câmpuri care nu au fost completate pe formular, ca în figura 1:

2008 Symposium Registration Form

Title:	Mr	
First Name:	Clurea	
Last Name:	Cristian	
Academic Position:	Junior lecturer	
Specialty:	Economic informatics	
Address:		Please fill in the address!
Street:		Please fill in the street!
Street Number:		Please fill in the street number!
City/County:		Please fill in the city or the county!
State/Province:		Please fill in the state or the province!
Zip/Postal Code:		Please fill in the zip or the postal code!
Country:	Romania	
Contact Phone or Cell:		Please fill in the phone number!
Email:		Please fill in the email address!
Fax:		Please fill in the fax number!
Special Assistance Needs:		Please select if you have or not special assistance needs!
PAYMENT		
Registration Fee:	Trainees - 200 euro	
Credit Card:	Visa	
Card Number:		Please fill in the card number (00000000000000000000)!
Expiration Date:		Please fill in the expiration date (MMYY)!
Comments:		
Submit Registration		

Figura 1. Date de intrare incomplete

- sunt complete, dar incorecte: situația în care toate câmpurile sunt completate, dar unul sau mai multe nu aparțin domeniilor de valori; în acest caz, la fiecare câmp se specifică ce s-a completat greșit, ca în figura 2:

2008 Symposium Registration Form

Title:

First Name:

Last Name:

Academic Position:

Specialty:

Address:

Street:

Street Number:

City/County:

State/Province:

Zip/Postal Code:

Country:

Contact Phone or Cell:

Email: Email must be in the format of name@somewhere.domain!

Fax:

Special Assistance Needs:

PAYMENT

Registration Fee:

Credit Card:

Card Number:

Expiration Date:

Comments:

Figura 2. Date de intrare incorecte

La fiecare încercare de înregistrare a datelor, dacă acestea nu au fost introduse corect și în totalitate, apare o listă de erori și datele nu sunt transmise în baza de date.

Introducerea oarecare a câmpurilor pe formular și posibilitatea de a transmite datele, chiar dacă acestea nu sunt complete, nu trebuie să existe într-o aplicație informatică orientată spre cetățean.

O aplicație informatică orientată spre cetățean conține unul sau mai multe formulare pentru introducerea datelor. Fiecare formular are mai multe câmpuri și pentru fiecare câmp există una sau mai multe validări. Structura unei aplicații informatice orientată spre cetățean este prezentată în figura 3:

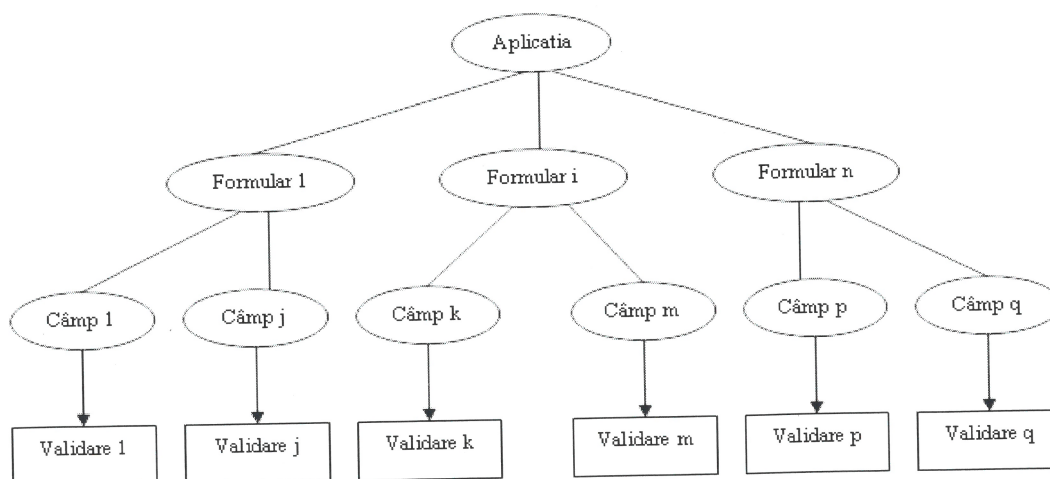


Figura 3. Structura unei aplicații informatice orientată spre cetățean

La nivelul fiecărui câmp, există un contor care numără erorile făcute de un utilizator la completarea datelor. Scopul acestor contoare este de a verifica rigiditatea validărilor făcute fiecărui câmp. Pentru fiecare eroare se calculează o frecvență de apariție, pe baza căreia se modifică interfața cu utilizatorul. Interfața se modifică prin intervenția factorului uman sau în mod automat.

4. Proceduri de validare

Se construiesc proceduri pentru validarea datelor: alfabeticе, numerice și a corelațiilor dintre câmpuri. Pentru fiecare aplicație se utilizează proceduri standard, iar mesajele de eroare trebuie să fie foarte nuanțate astfel încât să ajute la depanarea rapidă. Aceste mesaje de eroare trebuie să-i arate utilizatorului unde și ce anume a greșit. Astfel de aplicații, în care i se comunică utilizatorului exact ceea ce a introdus greșit, sunt ușor de implementat în situația în care câmpurile de completat conțin informații verificabile ca structură, cum ar fi codul numeric personal, numărul de telefon, adresa de e-mail, data calendaristică sau numărul cardului bancar. În cazul în care utilizatorul are de completat o informație liberă, atunci exactitatea mesajelor de eroare este mai dificil de implementat.

Modalitățile de prezentare a rezultatelor validării sunt următoarele:

- se afișează un mesaj confuz, care specifică faptul că sunt erori în date;
- pe un alt formular decât cel în care s-au completat datele, apare o listă a erorilor care precizează câmpul și natura erorii;
- pe formularul în care se completează datele, în partea dreaptă a câmpurilor completate greșit apar mesajele de eroare, colorate cu roșu.

Procesul de înregistrare a datelor în baza de date trebuie să conțină etapele din figura 4:



Figura 4. Etapele procesului de înregistrare a datelor

Etapa de validare asigură completarea corectă a informațiilor care urmează a fi înregistrate și prelucrate, eliminând datele aberante sau neconforme cu realitatea. Etapa de transmitere presupune realizarea conexiunii cu baza de date și înregistrarea efectivă a valorilor în baza de date. Etapa de recepție feedback presupune primirea de către utilizator a unui răspuns complet privind finalitatea acțiunii sale, respectiv dacă plata s-a efectuat cu succes sau dacă documentul solicitat a fost eliberat.

În cazul unei aplicații de internet banking, după ce utilizatorul completează datele necesare pentru efectuarea unei plăți, în momentul apăsării butonului Aprobă, starea plății se transformă în Aprobat de client. La un interval de câteva secunde, când plata intră în procesare, starea acesteia se schimbă în Preluat de bancă. Când plata s-a efectuat cu succes, starea ei devine Aprobat de bancă, confirmându-i utilizatorului finalitatea acțiunii sale.

Validarea datelor este de două tipuri: validare simplă și validare complexă. Validarea simplă este utilă pentru testarea izolată a unui singur control aflat pe un formular. De cele mai multe ori, se validează mai multe controale aflate pe același formular și ale căror informații conținute sunt intercorelate. În această situație, este vorba despre o validare complexă sau încrucișată. Un exemplu de validare încrucișată este prezentat în figura 5 și se referă la corelarea notei obținute de un student la un anumit examen cu situația școlară a acestuia:

Nota:

Situația:

Situația nu corespunde notei

Figura 5. Exemplu de validare încrucișată

Codul sursă, în limbajul C#, pentru validarea simplă a notei în intervalul [1, 10] și pentru validarea încrucișată a corelării notei obținute cu situația școlară este următorul:

```
private void button1_Click(object sender, EventArgs e)
{
    int nota = Convert.ToInt32(textBox1.Text);
    if (nota < 1 || nota > 10)
        errorProvider1.SetError(textBox1, "Nota in afara intervalului 1-10");
    else
        if (nota < 5 && comboBox1.Text == "Absolvent")
        {
            errorProvider1.SetError(textBox1, "Nota nu corespunde situatiei");
            errorProvider1.SetError(comboBox1, "Situatia nu corespunde notei");
        }
        else
            if (nota >= 5 && comboBox1.Text == "Restantier")
            {
                errorProvider1.SetError(textBox1, "Nota nu corespunde situatiei");
                errorProvider1.SetError(comboBox1, "Situatia nu corespunde notei");
            }
            else
            {
                errorProvider1.SetError(textBox1, "");
                errorProvider1.SetError(comboBox1, "");
            }
        }
    }
```

În limbajul C există biblioteci care oferă o serie de funcții pentru validarea unui caracter introdus de la tastatură, cum ar fi: `isalpha`, `isalnum` și `isdigit`, utilizate astfel:

```
char c;
scanf( "%c", &c );
if( isalpha(c) ) printf( "You entered a letter of the alphabet\n" );
if( isalnum(c) ) printf( "You entered the alphanumeric character\n", c );
if( isdigit(c) ) printf( "You entered the digit %c\n", c );
```

Validarea unei adrese de e-mail se realizează în limbajul C#.NET prin intermediul următorului cod sursă:

```
protected void Button1_Click(object sender, EventArgs e)
{
    bool ok = true;
    string[] splite-mail = (tb_e-mail.Text).Split(new Char[] { '@' });
    int ce = 0;
    foreach (string s in splite-mail) if (s.Trim() != "") ce++;
    if (ce != 2)
    {
```


lab

```
v_e-mail.Text = "Adresa de e-mail introdusa nu este valida!"; ok =
false;
}
}
```

Mediul Visual Studio are implementate pentru ASP.NET o serie de controale care facilitează scrierea de cod sursă pentru validarea unor câmpuri al căror format de intrare este prestabilit. Unul dintre aceste controale se numește *RegularExpressionValidator* și permite validarea adresei de e-mail, a numerelor de telefon, a adresei de internet, a codului poștal, precum și a numărului de securitate socială. Validarea adresei de e-mail, cu ajutorul acestui control, se face prin următorul cod sursă ASP.NET:

```
<asp:RegularExpressionValidator ID="v1_e-mail" runat="server"
ControlToValidate="tb_e-mail" ValidationExpression =
"\w+\x40{1}\w{2,}\.\{1}\w{2,}" ErrorMessage = "E-mail must be in the
format of name@somewhere.domain!" Display = "Dynamic">
</asp:RegularExpressionValidator>
```

În afară de *RegularExpressionValidator*, mediul Visual Studio mai conține pentru ASP.NET și alte controale destinate validării datelor de intrare, și anume: *RequiredFieldValidator*, *RangeValidator*, *CompareValidator* și *CustomValidator*.

Data de expirare a cardului bancar se introduce sub forma unui număr format din patru cifre, primele două reprezentând luna, iar ultimele două anul expirării. Câmpul de pe formular, în care se introduc aceste patru cifre, trebuie să conțină o validare prin care se verifică dacă primele două cifre reprezintă un număr mai mic sau egal cu 12, iar ultimele două cifre reprezintă un număr mai mare cu cel mult două unități decât anul curent. Per total, se validează ca numărul format din cele patru cifre să reprezinte o dată calendaristică mai mare ca data curentă. Codul sursă în limbajul C#.NET care implementează o astfel de validare este:

```
protected void Button1_Click(object sender, EventArgs e)
{
string month = tb_expiration.Text.Substring(0, 2);
string year = tb_expiration.Text.Substring(2, 2);
if (Convert.ToInt32(month) > 12) Label2.Text = "The month must be not
greater than 12!";
else if (Convert.ToInt32(year) >
(Convert.ToInt32(Convert.ToString(DateTime.Now.Year).Substring(2, 2))
+ 2)) Label2.Text = "The card is valid only 2 years!";
else if ((Convert.ToInt32(year) <=
Convert.ToInt32(Convert.ToString(DateTime.Now.Year).Substring(2, 2)))
&& (Convert.ToInt32(month) <= DateTime.Now.Month)) Label2.Text = "The
card has already expired!";
}
```

Există mai mult variante de validare a datelor, și anume:

- *varianta 1*: validare rigidă, în sensul că datele introduse sunt validate câmp cu câmp, iar utilizatorul nu are posibilitatea de a trece mai departe până nu completează corect câmpurile anterioare. Această validare este însoțită de o atenționare asupra naturii erorii, sub forma unui mesaj scris sau a unui semnal sonor;
- *varianta 2*: validare flexibilă, pe formular, cu afișarea mesajelor de date eronate;
- *varianta 3*: pe formular, câmpurile eronate devin roșii;
- *varianta 4*: pe formular, după transmitere apare o listă a erorilor.

Oricare dintre cele patru variante de validare s-ar folosi, aplicația trebuie să rezolve o singură problemă: informațiile înregistrate în baza de date trebuie să fie corecte și complete.

5. Eficiența interacțiunii

Funcție de problema de rezolvat, de cerințele utilizatorilor și de resursele financiare ale proprietarului aplicației, se definesc:

- durata de realizare a aplicației, ca fiind perioada cuprinsă între etapa de elaborare a specificațiilor până la terminarea etapei de testare și lansarea produsului pe piață;
- nivelul de complexitate, măsurat în funcție de numărul de componente ale aplicației, conexiunile cu baze de date și interacțiunea cu utilizatorii;
- nivelul de calitate, rezultat în urma testelor efectuate și a aprecierilor date de utilizatori;
- diversitatea resurselor disponibile, reprezentată de opțiunile pe care aplicația le oferă utilizatorilor;
- gradul de acoperire a soluției, măsurat ca diferență între ceea ce s-a dorit a fi implementat și ceea ce s-a realizat efectiv;
- conexiunea cu alte aplicații, din care rezultă gradul de integrabilitate.

Eficiența interacțiunii dintre aplicațiile informatice, orientate spre cetățean, și utilizatorii acestora, se măsoară cu ajutorul unor indicatori precum:

- timpul necesar introducerii datelor într-un formular: se măsoară în secunde și se contorizează din momentul selectării sau completării primului câmp de pe formular până la apăsarea butonului de transmitere a datelor;
- numărul de reintroduceri ale datelor, din cauza greșelilor: la fiecare apăsare a butonului de transmitere a datelor, un contor este incrementat în mod automat;
- costurile de dezvoltare și utilizare.

O aplicație este eficientă dacă utilizatorul termină cu introducerea datelor într-un număr foarte mic de iterații și într-un timp foarte scurt.

Cercetarea de față are menirea de a schimba radical modul de abordare a ciclului de dezvoltare software, deoarece societatea bazată pe cunoaștere presupune cultivarea cetățeanului care interacționează cu aplicații informatice pentru soluționarea a cât mai multe și mai variate probleme cotidiene din viața sa.

În timp, structura meniurilor în aplicațiile informatice orientate spre cetățean se stabilizează și, de aceea, atunci când se construiește o nouă aplicație:

- se preia structura stabilă a meniului;
- se realizează toate validările câmpurilor;
- se minimizează datele ce se introduc sau reintroduc de la tastatură.

Într-o aplicație bancară, pusă la dispoziția cetățeanului pentru efectuarea de plăți electronice, trebuie să existe în mod obligatoriu:

- toate datele de identificare ale plătitorului;
- toate datele de identificare ale beneficiarilor predefiniți;
- lista conturilor din care se fac plățile, precum și lista valutelor asociate acestora;
- toate formele de plată.

O astfel de aplicație necesită elaborarea unei proceduri de lucru, care se respectă de către toți utilizatorii, în care se prezintă pașii de acceptare și de confirmare a unei plăți electronice de către bancă.

Aplicațiile orientate spre cetățean vizează minimizarea disconfortului cetățeanului. Etapele pentru plata unei amenzi de circulație pentru depășirea vitezei legale ar trebui să fie următoarele:

- agentul de circulație are radarul calibrat, cu certificat de calibrare valid, postat pe internet;
- documentul care se întocmește are toate câmpurile completate corect;
- documentul se emite de pe calculatorul mașinii de poliție și are un cod unic;
- se acceptă documentul de către conducătorul auto;

- se plătește amenda astfel: agentul de circulație dispune de dispozitiv de acceptare carduri și se face plata pe loc sau plata se efectuează ulterior, la orice bancă sau electronic; plata efectuată la bancă, în contul poliției sau al primăriei, se reflectă automat la beneficiar, indiferent la ce bancă este făcută sau sub ce formă; în momentul efectuării plății, informația ajunge la primăria de care aparține plătitorul, pe baza codului numeric personal al acestuia, și la poliție, pe baza codului documentului.

Transformarea informațiilor în cunoștințe devine o realitate dacă și numai dacă se produce generalizarea proceselor de culegere automată a datelor privind comportamentul tuturor actorilor din societate și prelucrarea datelor cu algoritmi specifici domeniului data-mining.

Noua tehnologie va include:

- etape ale ciclului de dezvoltare;
- instrumente care să preia noile cerințe;
- metrice de calitate;
- modele pentru costuri;
- sisteme de colectare a datelor privind comportamentul;
- componente de adaptare a structurii aplicației;
- componente de asigurare a securității;
- evaluarea de variante pentru fiecare produs al etapei;
- componente de optimizare locală;
- componente de rafinare;
- subordonarea acțiunilor la exigențele grupului țintă.

Cu ajutorul acestei noi tehnologii, se vor dezvolta aplicații care îl vor completa pe cetățean, utilizatorul devenind parte componentă a sistemului. Cu cât interacțiunea Om-Calculator este mai puternică, cu atât sistemul este mai eficient.

6. Concluzii

Ideea de aplicații informatice orientate spre cetățean se referă la un sistem colaborativ eficient, în care oamenii și echipamentele cooperează în scopul atingerii anumitor obiective. Este vorba despre Human-Human-Interaction și Human-Computer-Interaction. Aplicațiile on-line orientate spre cetățean sunt accesate de un număr foarte mare de utilizatori neomogeni, care trebuie să petreacă cât mai puțin timp în interacțiune cu aplicația. Marile probleme apar din erorile de introducere a datelor. De aceea, aplicațiile trebuie să furnizeze informații foarte exacte privind apariția erorilor și rezolvarea lor. Ideea de bază este ca utilizatorul să finalizeze introducerea datelor într-un număr foarte mic de iterații, iar numărul de introduceri cauzate de modul cum sunt structurate câmpurile în cadrul interfeței să fie cât mai scăzut.

Aplicațiile orientate spre cetățean trebuie construite cât mai simplu, astfel încât cetățenii să folosească facilitățile oferite în mod cât mai natural, fără a necesita o pregătire de specialitate și fără a fi necesare o instruire prealabilă.

Studierea ergonomiei interfețelor asigură utilizarea eficientă a opțiunilor acestora și îmbunătățește timpul de accesare a informațiilor.

Bibliografie

1. **CIUREA, C.:** Metrici ale sistemelor colaborative financiare. Sesiunea cercurilor științifice studențești din Academia de Studii Economice, București, mai, 2007.
2. **CONNOLLY, T.:** Carolyn BEGG – Baze de date: proiectare, implementare, gestionare, Editura Teora, București, 2001.
3. **COOPER, C.:** Proiectarea interfețelor utilizator, Editura Tehnică, București, 1997.
4. **DAVID, B., O. DELOTTE, R. CHALON, F. TARPIN-BERNARD, K. SAIKALI:** Patterns in Collaborative System Design, Development and Use, Laboratoire ICTT, Ecole Centrale de Lyon, France, 2003.
5. **FRAIZER, C., J. BOND:** Java API: manualul interfeței de programare a aplicațiilor, Editura Teora, București, 1998.
6. **GAUSE, D., G. WEINBERG:** Exploring Requirements Quality before Design, Dorset House Publishing, New York, 2004.
7. **IVAN, I., C. BOJA, C. CIUREA:** Metrici ale sistemelor colaborative, Editura ASE, București, 2007.
8. **IVAN, I., M. DOINEA:** Aspecte privind optimizarea proceselor de autentificare în aplicații distribuite, Economia – Teorie și aplicații, 2008.
9. **IVAN, I., A. KARADIMOU, A. LICURICEANU, G. LUPU:** Metrici de complexitate software bazate pe dependențele instrucțiunilor. Revista Informatică Economică, nr. 11, 1999.
10. **PRESSMAN, R.:** Software Engineering: A Beginner's Guide, McGraw-Hill Publisher, New York, 1988.
11. **SMEUREANU, I., M. DARDALA, A. REVEIU:** Visual C#.NET, Editura CISON, București, 2004.
12. **WEINBERG, G.:** Quality Software Management: Systems Thinking, Dorset House Publishing, New York, 2007.