AUTOTESTAREA APLICAȚIILOR DE SCROLLING

Ion Ivan, Cristian Ciurea, Sorin Vinturis

Academia de Studii Economice, Bucureşti

ionivan@ase.ro, cristian.ciurea@ie.ase.ro, sorin.vinturis@yahoo.com

Rezumat: Se definesc aplicațiile de scrolling şi se stabilesc cerințele de calitate pentru ele. Se prezintă etapele ciclului de realizare a aplicațiilor de scrolling şi se analizează conținutul digital existent în cadrul acestor aplicații. Aplicațiile de scrolling sunt proiectate ca aplicații informatice orientate spre cetățean. Se identifică tipurile de erori care apar în procesul de dezvoltare a aplicațiilor de scrolling şi se descriu cauzele care conduc la apariția acestora. Se definește obiectivul autotestării ca proces de control şi corecție totală. Se implementează metrici pentru evaluarea efectelor produse de erorile apărute în procesul de dezvoltare şi autotestare a aplicațiilor de scrolling.

Cuvinte cheie: scrolling, autotestare, calitate, procese.

Abstract: The scrolling applications are defined and the quality requirements are set for them. The developing stage cycles for scrolling applications are presented and the existing digital content in these applications is analyzed. Scrolling applications are designed as citizen-oriented applications. The error types that appear in the scrolling application development are identified and the causes of their occurrence are described. It is defined the objective of the autotesting as a process of control and total debugging. Are implemented metrics to evaluate the effects of errors arising in the process of development and autotesting of scrolling applications.

Keywords: scrolling, autotesting, quality, processes.

1. Aplicația de scrolling

Dezvoltarea aplicațiilor de scrolling porneşte de la ideea de a oferi un conținut, cu ajutorul căruia persoanele ce accesează astfel de aplicații să soluționeze probleme, să atingă obiectivul pentru care au efectuat accesarea.

Se consideră o problemă P formată din subproblemele P_1, P_2, ..., P_n cu $P_i \cap P_j = \Phi$, i, j = 1..n. Fiecare subproblemă P_i are şi o parte concentrată, asociată cu un text TS_i de lungime LS_i şi un text extenso TE_i de lungime LE_i cu $LE_i >> LS_i$.

Se defineşte o formă de prezentare a problemei P, alcătuită din două părți, prima parte conținând textele TS_1, TS_2, ..., TS_n simultan, iar cea de-a doua componentă conținând textul TE_j selectat, ca în figura 1:

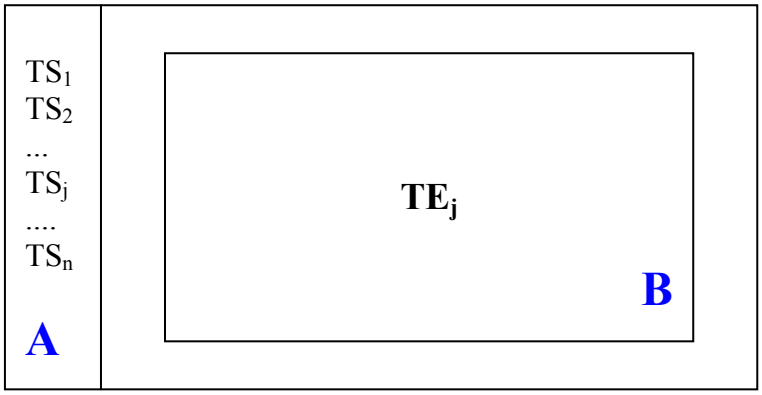

Figura 1. Formă de prezentare a aplicației de scrolling

Structura aplicației din figura 1 include două componente, respectiv A şi B, în cadrul cărora se află conținutul digital reprezentat de textele TS1, TS2, ..., TSn şi textul TEj selectat. Textul TSi reprezintă rezumatul textului TEj, fără a fi obligatoriu ca TSi să fie inclus în TEj ($TSi \subset TEj$). Conținutul digital al aplicației de scrolling din figura 1 este dat de reuniunea conținuturilor digitale din componentele A şi B, respectiv $C = A \cup B$, cu condiția ca $A \cap B \neq \emptyset$. Conținutul digital se structurează astfel încât traversarea lui să se bazeze pe reguli acceptate, în număr cât mai restrâns.

Aplicația de scrolling este oportună atunci când *n* este suficient de mare, astfel încât TS_1, TS_2, ... TS_n să permită:

- prezență simultană;
- suficientă informație care să permită selectarea textului extenso fără multe elemente repetitive, ci textul selectat să fie exact cel căutat;
- elementul de scrolling să permită referirea tuturor elementelor din mulțimea textelor scurte şi o bună raportare pe submulțimi direct accesibile.

Scrolling-ul presupune:

- o submulțime formată din *k* elemente care fac obiectul referirii directe;
- o mulțime formată din *n* elemente, unde $n>>k$, ce sunt disponibile în procesul de referire.

În prima componentă apar elementele TS_1, TS_2, ... TS_k. Pe măsură ce se derulează procesul de scrolling se va deschide accesul la textele TS_x, TS_{x+1}, …, TS_{x+k}, cu $x + k \leq n$.

În aplicațiile uzuale TS_i sunt imagini la dimensiuni reduse, în timp ce TE_i sunt aceleaşi imagini, dar la dimensiuni normale.

În figura 2 se prezintă o aplicație de scrolling conținând fotografii reprezentative din domeniul sporturilor extreme, respectiv X-treme Skyflyer:

Figura 2. Aplicația de scrolling X-treme Skyflyer

Aplicația de scrolling din figura 2 prezintă un conținut digital format din poze de diferite dimensiuni. În partea stângă se află pozele la dimensiuni reduse, iar în partea dreaptă se afişează aceleaşi poze, dar la dimensiuni normale.

Se consideră poza Pl cu dimensiunile reduse l_1 şi l_2 şi poza PL cu dimensiunile normale L_1 şi L_2, unde $L_1>>l_1$ şi $L_2>>l_2$. Cele două poze sunt ortogonale dacă $l_1 * l_2 < 0{,}75 * L_1 * L_2$, adică aria dreptunghiului reprezentat de prima poză este mai mică decât 75% din aria dreptunghiului reprezentat de cea de-a doua poză.

În alte aplicații, se prezintă la nivelul TS_i elemente care concentrează informații referitoare la entități complexe, precum coperți de reviste, imagini de produse, imagini de persoane, iar TE_i reprezintă conținutul efectiv al revistei sau descrierea completă a produsului sau prezentarea persoanei, astfel încât să existe acces la toate informațiile de interes legate de respectiva persoană.

Aplicația de scrolling presupune existența unei cantități mari de informație, *I*, care trebuie afișată într-o suprafață mică, *S*, limitată de dimensiunile ecranului calculatorului. De regulă, se verifică situația în care $I >> S$. Suprafața *S* conține textele TS_1, TS_2, ..., TS_n simultan, iar la click pe unul din acest texte se afișează textul TE_j selectat, reprezentând o parte din informația *I*. Pe măsură ce se accesează un alt TS_i, se afișează o altă parte din informația *I*. Tranziția de la un element TS_i la altul pentru afișarea unui text TE conduce la a considera aplicația de scrolling sub forma unui automat finit foarte simplu.

Un automat finit se reprezintă cu ajutorul diagramei de stări, dată sub forma tabelului de tranziție dintr-o stare în alta:

Tabel 1. Diagrama de stări pentru un automat finit

Conditie / Stare	Starea 1	Starea 2	...	Starea i	...	Starea n
Conditia 1						
Conditia 2		Starea 3				
...						
Conditia j				Starea k		
...						
Conditia m						

Conform reprezentării din tabelul 1, automatul finit trece din starea 2 în starea 3 în urma declanșării condiției 2. De asemenea, automatul finit trece din starea *i* în starea *k* prin aplicarea condiției *j*. În cazul aplicației de scrolling, condițiile sunt reprezentate de selectarea unui element TS_i prin click efectuat de utilizator, iar stările sunt date de textele TS_1, TS_2, ..., TS_n.

Prezentarea Power Point este aplicație de scrolling, deoarece permite vizualizarea unui slide selectat dintr-o mulțime de slide-uri disponibile în partea stângă a ecranului. În figura 3 se prezintă modalitatea de vizualizare a unei prezentări Power Point.

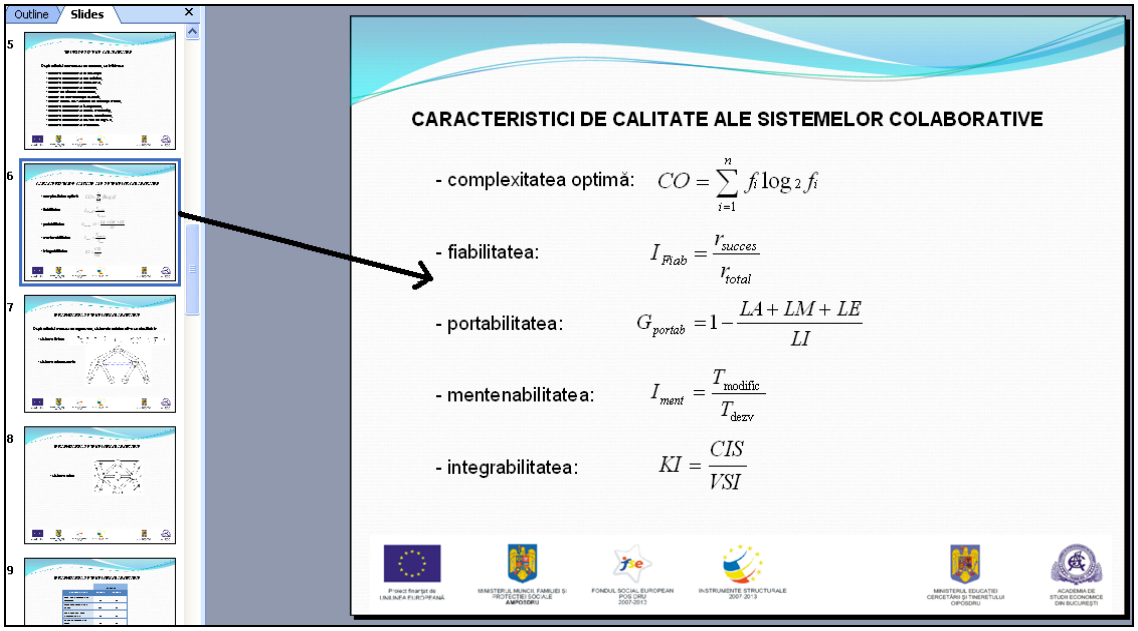

Figura 3. Vizualizare prezentare Power Point

Conform reprezentării din figura 3, elementele TS_1, TS_2, ..., TS_{11} sunt reprezentate de cele 11 slide-uri, afișate în miniatură în partea stângă, iar TE este reprezentat de slide-ul selectat.

2. Ciclul de realizare a aplicațiilor de scrolling

Ciclul de realizare a aplicațiilor de scrolling presupune parcurgerea etapelor:

E1. Definirea obiectivului, reprezentând referirea rapidă, directă a subproblemelor SP_1, SP_2,..., SP_n de descriere şi soluționare pentru problema P; obiectivul se defineşte şi în funcție de resursele disponibile, momentul în care aplicația trebuie să devină operațională şi, mai ales, de experiența personalului;

E2. Obținerea textelor TS_1, TS_2, ..., TS_n şi a textelor extenso TE_1, TE_2, ..., TE_n şi formarea de perechi (TS_1, TE_1), (TS_2, TE_2), ..., (TS_n, TE_n);

E3. Construirea subvocabularului SVS pentru a pune în corespondență elementele mulțimii TS cu cuvinte;

E4. Construirea subvocabularului SVE pentru a pune în corespondență elementele din mulțimea extenso cu cuvinte din acest subvocabular;

SVS = {cs_1, cs_2, …, cs_n}.

SVE = {ce_1, ce_2, …, ce_n}.

E5. Construirea perechilor (cs_1, ce_1), (cs_2, ce_2), ..., (cs_n, ce_n), respectiv a asocierilor dintre cuvintele din cele două subvocabulare;

E6. Elaborarea componentei din aplicația de scrolling pentru TS_i ;

E7. Elaborarea componentei de scrolling pentru TE, adică fundalul aplicației;

E8. Elaborarea componentelor pentru referirea elementelor TE_1, TE_2, ..., TE_n ;

E9. Lansarea în execuție, de probă, şi identificarea măsurii în care perechile (TS_i, TE_i) există în realitate prin referirile generate de cuvintele celor două subvocabulare (cs_i, ce_i), care se găsesc în cele trei categorii de componente elaborate în aplicație;

E10. Derularea procesului de autotestare;

E11. Implementarea în momentul în care aplicația are zero erori.

În figura 4 se reprezintă grafic etapele ciclului de realizare a aplicațiilor de scrolling:

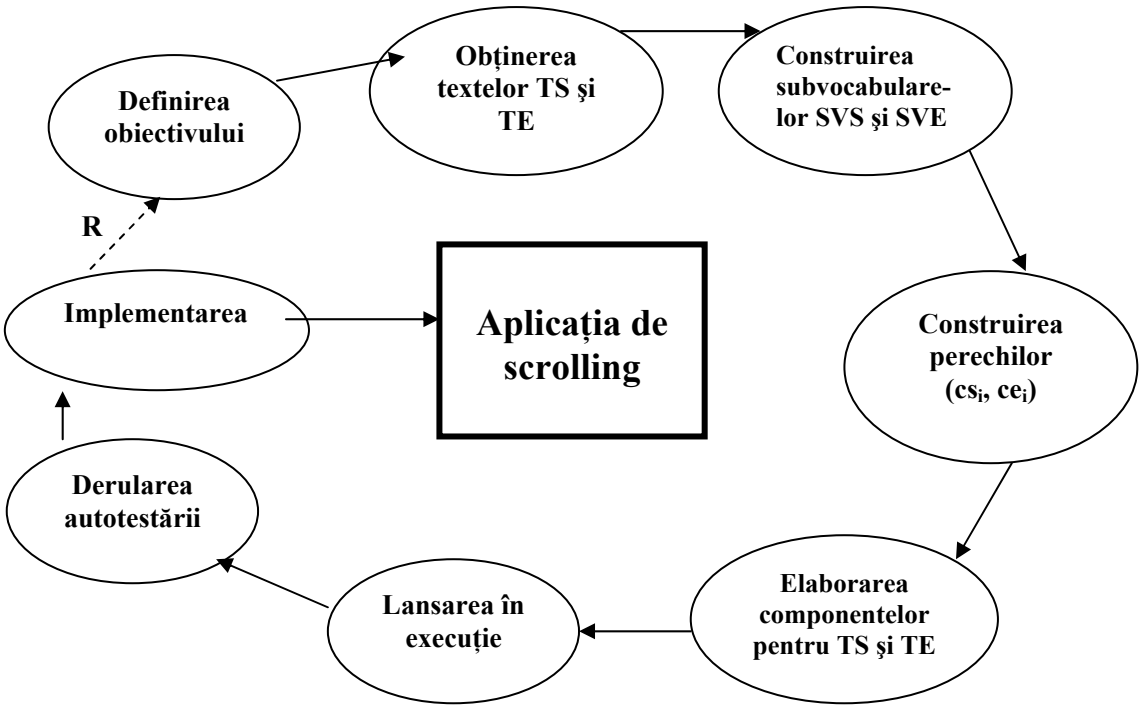

Figura 4. Ciclul de realizare a aplicațiilor de scrolling

Arcul *R* de la etapa de implementare la cea de definire a obiectivului este întâlnit în contextul specific proceselor de reinginerie.

Legătura dintre elementele cs_i, TS_i, TE_i, ce_i este reprezentată în figura 5:

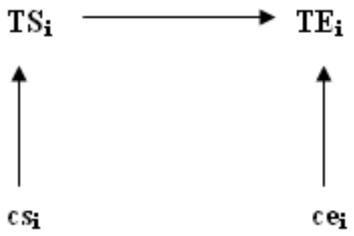

Figura 5. Asocierea dintre elementele cs_i, TS_i, TE_i, ce_i

Conform reprezentării din figura 5, elementul cs_i din subvocabularul SVS referă obiectul TS_i, care la rândul său referă obiectul TE_i. Obiectul TE_i este referit şi de către elementul ce_i din subvocabularul SVE.

Se consideră aplicația de scrolling pentru realizarea unei prezentări de fotografii. Ciclul de realizare a acestei aplicații presupune obținerea imaginilor la dimensiuni reduse TS_1, TS_2, ..., TS_n, obținerea imaginilor la dimensiuni normale TE_1, TE_2, ..., TE_n, formarea perechilor (TS_1, TE_1), (TS_2, TE_2), ..., (TS_n, TE_n), construirea subvocabularelor SVS şi SVE pentru denumirea fotografiilor, construirea perechilor de cuvinte din SVS şi SVE, elaborarea paginilor aplicației pentru elementele TS_i, dezvoltarea componentei pentru elementele TE_i, autotestarea aplicației şi verificarea corectitudinii asocierilor dintre TS_i şi TE_i.

În specificațiile de proiectare a aplicației de scrolling se dau perechile de elemente TS_1, TS_2, ..., TS_n şi TE_1, TE_2, ..., TE_n, iar realizarea aplicației necesită construirea perechilor TS'_1, TS'_2, ..., TS'_n şi TE'_1, TE'_2, ..., TE'_n. În procesul de construire a perechilor de obiecte TS' şi TE' apar erori de asociere între obiecte sau se pierd o parte din elemente, rezultând mulțimile TS' şi TE' cu *m* elemente, unde $m < n$.

Aplicațiile de scrolling conțin elemente care se derulează vertical sau orizontal, în funcție de specificațiile de proiectare. Realizarea unei aplicații de scrolling presupune împărțirea ecranului în cel puțin două frame-uri, primul conținând elementele TS_1, TS_2, ..., TS_n, iar cel de-al doilea elementul TE_j selectat. Opțiunea de scrolling din primul frame se activează în cazul în care numărul elementelor TS_1, TS_2, ..., TS_n este suficient de mare, astfel încât acestea să nu încapă pe ecran şi să fie necesară derularea ecranului pentru vizualizarea lor. Codul sursă pentru elaborarea componentei din aplicația de scrolling pentru TS_i şi TE este:

```
<html>
<head>
<title>
JACS - Journal of Applied Collaborative Systems
</title>
</head>
<frameset framespacing="0" frameborder="0" rows=15%,*>
<frame framespacing="0" noresize="noresize" frameborder="0"
scrolling="yes" src=sigla.html></frame>
<frameset framespacing="0" frameborder="0" cols=20%,*>
<frame framespacing="0" noresize="noresize" frameborder="0"
scrolling="yes" src=meniu.html name=meniu></frame>
<frame framespacing="0" noresize="noresize" frameborder="0"
src=home.html name=continut></frame>
</frameset>
</html>
```

În figura 6 este prezentată aplicația de scrolling în care numărul elementelor TS_1, TS_2, ..., TS_n nu este suficient de mare astfel încât să fie necesară activarea scrolling-ului în frame-ul pentru TS_i, ci doar în frame-ul pentru vizualizarea elementului TE:

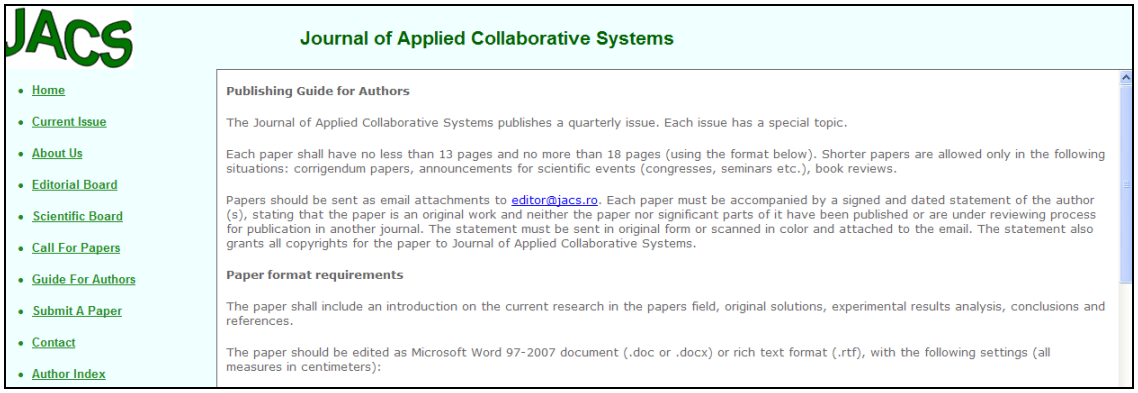

Figura 6. Aplicația de scrolling pentru revista JACS

În aplicația de scrolling din figura 6, elementele TS_1, TS_2, ..., TS_n sunt texte, numărul acestora fiind limitat.

Activarea scrolling-ului depinde şi de browser-ul în care se vizualizează aplicația. Dacă aplicația de scrolling pentru revista JACS, disponibilă la adresa www.jacs.ro, este accesată din browser-ul mobil Opera Mini, atunci elementele TS_i şi TE sunt aşezate în pagină, astfel încât vizualizarea lor să fie cât mai completă:

Figura 7. Vizualizarea aplicației de scrolling pentru revista JACS din browser-ul mobil Opera Mini

Realizarea aplicațiilor de scrolling trebuie să ia în considerare şi diferitele tipuri şi versiuni de navigatoare, astfel încât vizualizarea aplicațiilor să fie cât de cât mai uniformă, indiferent de tipul browser-ului.

Aplicațiile de scrolling sunt aplicații informatice orientate spre cetățean. Ciclul de dezvoltare al unei aplicații informatice orientată spre cetățean este următorul:

- se defineşte grupul țintă de către proprietarul aplicației şi dezvoltatorul acesteia;
- se stabilesc obiectivele grupului țintă, în scopul definirii problemei de rezolvat;
- se elaborează specificații pentru a satisface dorințele grupului țintă;
- se scrie cod pentru dezvoltarea aplicației;
- se testează aplicația de către grupul țintă.

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

Aplicațiile de scrolling sunt aplicații deschise, în sensul că dacă apar elemente noi, acestea se adaugă la sfârşit, ca în cazul evenimentelor precum olimpiadele sportive:

TS_1 – olimpiada din anul 1896 de la Atena;

TS_2 – olimpiada din anul 1900 de la Paris;

.....

TS_n – olimpiada din anul 2008 de la Beijing;

TS_{n+1} – olimpiada din anul 2012 de la Londra.

Ciclul de dezvoltare a unei aplicații de scrolling este diferit de cel al unei aplicații informatice orientată spre cetățean prin faptul că, înainte de etapa de testare a aplicației de către grupul țintă, se realizează autotestarea aplicației de către dezvoltatorul acesteia.

Pe măsură ce evoluează un fenomen existent, se inserează elemente în mulțimea TS. Se consideră că TS_i reprezintă un film realizat de un regizor, pus în ordinea perioadei. Dacă regizorul mai realizează un film comic, TSN, acesta va fi intercalat. Mulțimea TS_1, TS_2, ..., $TS^{(k)}_i$, $TS^{(k)}_{i+1}$, ..., TS_n devine TS_1, TS_2, ..., $TS^{(k)}_i = TS^{(k+1)}_i$, $TSN = TS^{(k+1)}_{i+1}$, $TS^{(k)}_{i+1} = TS^{(k+1)}_{i+2}$, ..., TS_n.

Dacă experiența practică impune schimbarea pozițiilor între TS_i şi TS_k, atunci acest lucru se realizează fără dificultăți în textul sursă. În acelaşi fel se pune problema dezvoltării de elemente din şirul TS_1, ..., TS_n.

3. Erori în procesul de dezvoltare a aplicației de scrolling

Se consideră că mulțimea textelor scurte are *n* componente, iar mulțimea textelor lungi are *m* componente. Situația corectă este cea în care $n=m$.

Dacă $n>m$ atunci înseamnă că:

- au fost pierdute elemente din mulțimea TE;
- au fost incluse ca diferite mai multe elemente din TS.

Construirea perechilor (TS_i, TE_i) este incorectă atunci când se construieşte (TS_i, TE_j), unde $i \neq j$, $i,j = 1..n$.

În figura 8 se prezintă situația corectă de asociere între elementele TS şi TE:

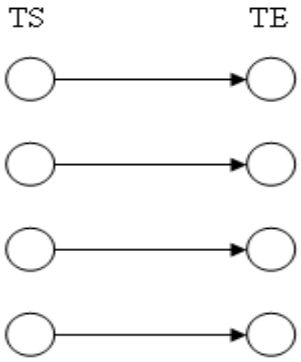

Figura 8. Situația corectă de asociere între elementele TS şi TE

Fiecărui element de pe poziția *i* din mulțimea TS trebuie să-i corespundă un element de pe aceeaşi poziție din mulțimea TE.

În figura 9 se prezintă o asociere incorectă între elementul TS_i şi varianta extinsă a acestuia, respectiv TE_j :

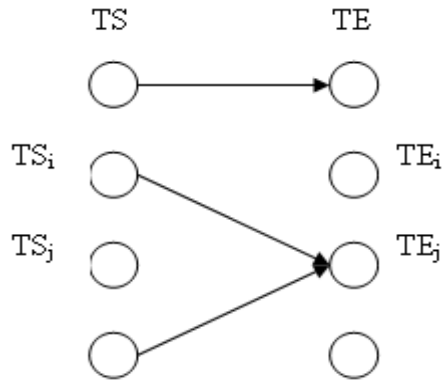

Figura 9. Asociere incorectă între TS_i şi TE_j

Conform asocierii din figura 9, elementului de pe poziția *i* din mulțimea TS îi corespunde elementul de pe poziția *j* din TE. De asemenea, un alt element din TS referă acelaşi element de pe poziția *j* din TE.

Figura 10 prezintă o altă asociere eronată, de data aceasta încrucişată, între elementele mulțimilor TS şi TE:

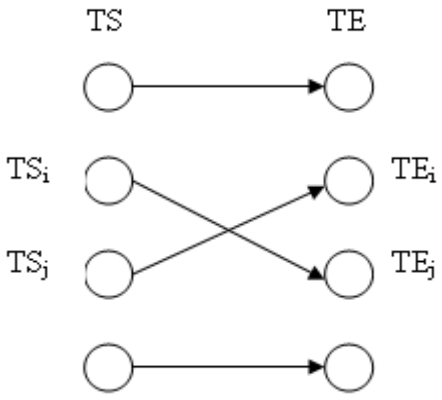

Figura 10. Asociere încrucişată

În figura 10, elementele de pe pozițiile *i* şi *j* din mulțimile TS şi TE se referă încrucişat, conducând la apariția perechilor (TS_i, TE_j), respectiv (TS_j, TE_i).

Figura 11 prezintă o altă eroare apărută în procesul de dezvoltare a aplicației de scrolling, respectiv aceluiaşi element TS_i îi corespund variantele extinse TE_i şi TE_j, iar TS_j nu are corespondent în extenso:

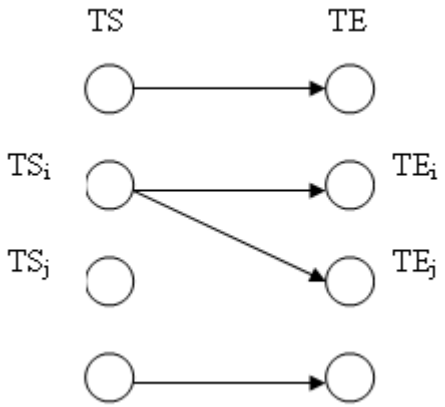

Figura 11. Asociere multiplă

Situația din figura 11 este rar întâlnită şi se traduce prin asocierea dintre o imagine la dimensiuni reduse cu două imagini la dimensiuni normale, în acelaşi timp. Practic, la nivel fizic, elementele mulțimilor TS şi TE se încurcă.

Un alt tip de erori vizează conținutul digital însuşi, erorile fiind generate de:

- inexactități de terminologie;
- localizări eronate;
- indicarea incorectă a numelor de persoane;
- utilizarea incorectă a momentelor de timp;
- punerea incorectă în corespondență a textelor cu imaginile;
- prezentarea de texte şi imagini trunchiate;
- utilizarea de reguli ortografice, altele decât cele aflate în vigoare.

La nivel de subvocabulare este important ca:

- numărul de cuvinte din SVS să fie mult mai mare ca numărul de elemente TS_1, TS_2, ..., TS_n;
- numărul de cuvinte din SVE să fie mult mai mare ca numărul de elemente TE_1, TE_2, ..., TE_n.

De asemenea, este important să se respecte condițiile de ortogonalitate, respectiv cuvintele din cele două subvocabulare să fie cât mai ortogonale.

Erorile care apar la nivel de subvocabulare se referă la perechile (TS_i, TE_i), cărora corect este să le corespundă perechile de cuvinte (cs_i, ce_i).

În realitate, lui TS_i îi corespunde cs_k, iar lui TE_i îi corespunde ce_h.

Atunci când asocierile dintre TS şi TE sunt incorecte, utilizatorul aplicației de scrolling selectează corect elementul TS_i, dar elementul TE este incorect selectat, ceea ce conduce la alocarea greşită a resurselor şi creşterea costurilor. În astfel de situații, utilizatorul aplicației de scrolling nu alocă resursele corect şi nu dezvoltă corect activitățile. Dacă elementul TS_i selectat face referire la detalii despre susținerea examenului EX_i, iar elementul TE oferă informații

despre susținerea examenului EX_j, atunci consecințele acestei asocieri eronate dintre TS şi TE sunt devastatoare.

În figura 12 se prezintă situația corectă de asociere între elementele TS şi cuvintele din subvocabularul SVS, respectiv între elementele TE şi cuvintele din SVE.

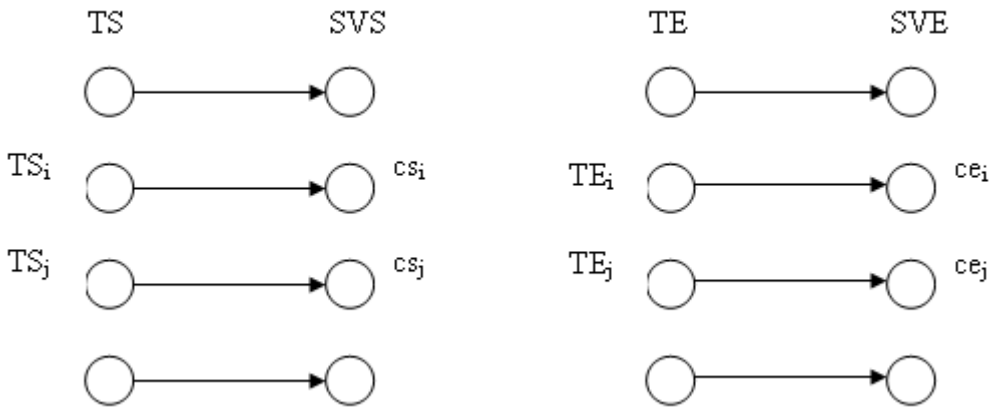

Figura 12. Asociere corectă între elementele TS-SVS şi TE-SVE

În figura 13 se prezintă erorile de asociere între elementul TS_i şi cuvântul cs_k, respectiv între TE_i şi ce_h.

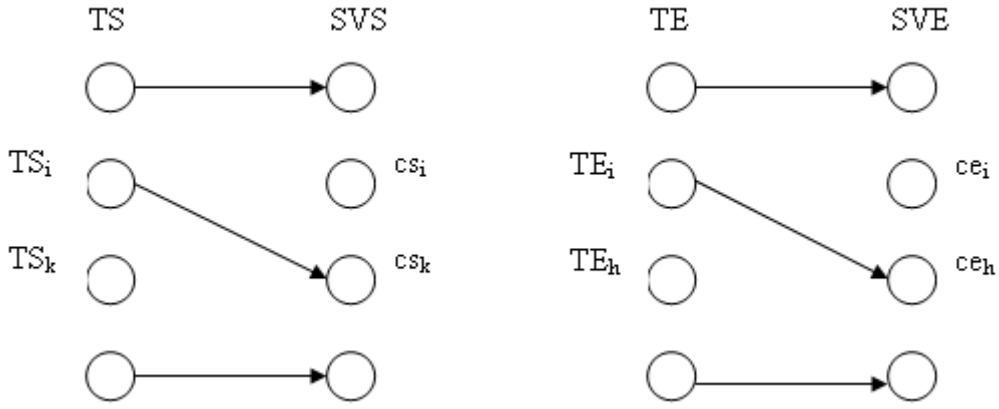

Figura 13. Asociere incorectă între elementele TS-SVS şi TE-SVE

Situațiile care apar sunt următoarele:

- se pun mai multe cuvinte pe o imagine;
- se suprascriu mai multe imagini pe un cuvânt;
- există foldere cu mai multe poze având aceeaşi denumire;
- o poză are mai multe nume sau există mai multe poze cu acelaşi nume.

Efectele produse sunt legate de faptul că, la scrolling informația rezumat nu mai este în concordanță cu informația de detaliu. Eroarea de neconcordanță între conținutul anunțat şi cel referit are la bază numeroase cauze, dintre care cea mai frecventă este legată de codificările construite pentru fişiere, mai ales atunci când se lucrează cu un număr foarte mare de fişiere.

Se consideră aplicația de scrolling pentru vizualizarea a *r* reviste, mulțimea TS fiind reprezentată de rezumatele articolelor din cele *r* reviste, iar mulțimea TE este reprezentată de revista din care face parte articolul selectat.

O situație care se remediază uşor este aceea în care se încurcă rezumatele a două articole cu

revistele de care aparţin, respectiv se consideră ca rezumatul articolului *i* aparţine revistei *j*, iar rezumatul articolului *j* aparţine revistei *i*:

rezumat 1 -> revista 1
rezumat 2 -> revista 2
rezumat 3 -> revista 3
rezumat 4 -> revista 4
...
rezumat i -> revista j
...
rezumat j -> revista i
...
rezumat r -> revista r

O altă situaţie este aceea în care, la chestii diferite există aceeaşi extensie, respectiv rezumatele articolelor *i*, *j* şi *k* se consideră că aparţin revistei *i*:

rezumat 1 -> revista 1
rezumat 2 -> revista 2
...
rezumat i -> revista i
rezumat j -> revista i
rezumat k -> revista i
...
rezumat r -> revista r

O situaţie dificilă este aceea în care toate elementele sunt încurcate, adică $i \neq j$, $i, j = 1..r$.

rezumat 1 -> revista 3
rezumat 2 -> revista 1
rezumat 3 -> revista 4
rezumat 4 -> revista 2
rezumat 5 -> revista 7
rezumat 6 -> revista 5
rezumat 7 -> revista 6
...
rezumat r -> revista m

Unele dintre erori trec neobservate, însă cele mai multe compromit integral aplicaţia. Utilizatorii au comună caracteristica în ceea ce priveşte referirea de aplicaţii informatice, referire care presupune ca la prima eroare detectată să întrerupă accesul şi să nu repete încercarea la un alt interval de timp. Există însă o soluţie care îmbunătăţeşte această abordare: lansarea numai după efectuarea autotestării.

Cazul cel mai grav de eroare este când $ns \neq ne$, $ncs \neq ns$, $nce \neq ne$ unde:

ns – numărul de fişiere TS;

ne – numărul de TE;

ncs – numărul de cuvinte din subvocabularul SVS;

nce – numărul de cuvinte din subvocabularul SVE.

Acest lucru se produce când *n* – numărul de fişiere este foarte mare şi se lucrează pe submulţimi care se concatenează. Este situaţia în care se lucrează distribuit şi colaborativ.

Se consideră M locaţii în care se află secvenţe de scrolling şi fişiere de acces la detaliu. Reprezentarea acestei situaţii se realizează în figura 14:

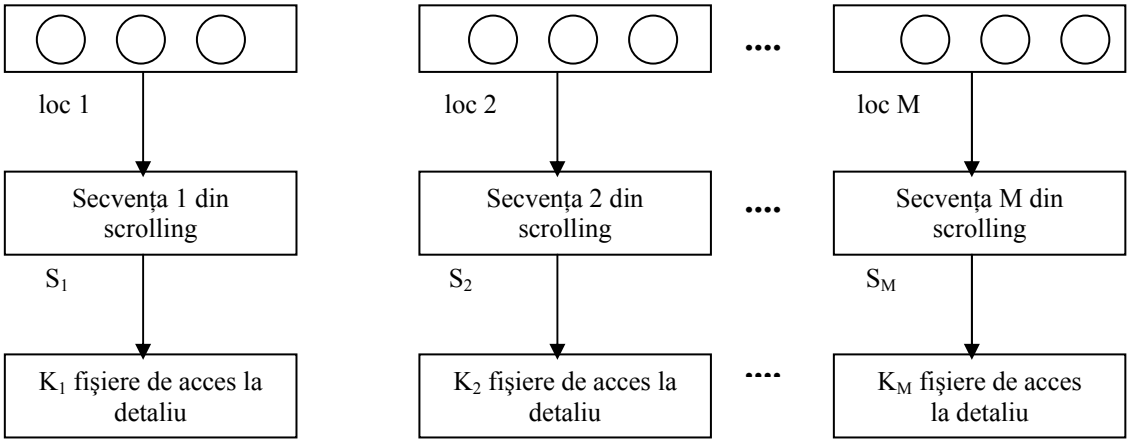

Figura 14. Proces de scrolling colaborativ

Prin concatenarea celor M secvențe de scrolling rezultă fişierul sursă de scrolling, fiind realizat de mai mulți utilizatori, care accesează cele M secvențe de scrolling din locații diferite. Reuniunea celor M secvențe de scrolling formează mulțimea fişierelor de detaliu.

Aplicația de scrolling, a cărei structură este prezentată în figura 14, este distribuită şi colaborativă. Este o aplicație colaborativă, deoarece utilizatorii lucrează împreună la atingerea obiectivului comun reprezentat de obținerea fişierului complet de scrolling prin concatenarea secvențelor de scrolling.

4. Dezvoltarea procesului de autotestare

Autotestarea se efectuează de către dezvoltatorul aplicației de scrolling. Se defineşte autotestarea ca proces de evidențiere de către dezvoltator a măsurii în care un stadiu al aplicației informatice este în concordanță cu cerințele specificațiilor. Necesitatea autotestării este dată de:

- reducerea numărului de erori de la un stadiu la altul;
- detectarea erorilor în faza de realizare a aplicației, întrucât persoanele implicate în realizarea unui stadiu al aplicației găsesc erorile mult mai repede şi le corectează fără costuri prea mari;
- erorile dintr-un stadiu nu sunt amplificate de dezvoltarea stadiilor următoare.

Procesul de autotestare este complet şi presupune parcurgerea următoarelor etape:

E1. Se construieşte lista TS_1, TS_2, ..., TS_m ;

E2. Se construieşte lista TE_1, TE_2, ..., TE_n ;

E3. Se verifică să existe egalitatea m=n ;

E4. Se alege o regulă de construire a cuvintelor de vocabularele SVS şi SVE; acestea trebuie să fie deschise, astfel încât să suporte corecții, adăugări, eliminări şi interschimburi de TS-uri şi TE-uri; prin denumirea fişierelor în varianta *colita0010, colita0020, ..., colita0070*, se oferă posibilitatea adăugării de noi elemente cu denumirile *colita0011, colita0012*, etc.;

E5. Se construiesc perechile de elemente (TS_i, TE_i);

E6. Se construiesc perechile de cuvinte (cs_i, ce_i);

E7. Se intră pe aplicație şi se merge pe scrolling;

E8. Se ia imaginea 1;

E9. Se vede dacă în lista de cuvinte apare denumirea fişierului;

E10. Se vede ce poze referă extensia;

E11. Se observă ce apare în zona TE;

E12. Se verifică corelațiile (TS_i, TE_i) şi (cs_i, ce_i).

Dacă s-au parcurs complet listele, atunci înseamnă că toată aplicația este în regulă. Dacă s-a terminat mult din partea de scrolling, iar în listă mai există perechi de poze şi de cuvinte, înseamnă că aplicația este incompletă, deşi tot ce s-a testat a fost în regulă.

În acest caz, se încarcă fişiere de poze, respectiv TS şi TE. Se adaugă secvențe în scrolling şi se construiesc fişiere de referire TE.

În autotestare trebuie văzute neconcordanțele. Există TS_i pus în corespondență cu TE_i. Se realizează autotestarea pentru toate perechile şi se observă ce nu este în regulă, acționând pe texte pentru a corecta. Se stabilesc cauzele apariției erorilor. Cea mai frecventă cauză este aceea că multe din secvențele de cod se construiesc prin copy-paste.

În scrolling există fişiere de acces TE_i şi elemente de referire TS_i. Există riscul de a actualiza în TS şi nu în fişierul de referire sau se realizează actualizarea în fişierul de referire, dar nu şi în TS.

Regula de construire a cuvintelor din subvocabularele SVS şi SVE trebuie să respecte teoria codurilor, astfel încât să se obțină coduri deschise, de lungime fixă. Denumirea pozelor reprezentând sportivi de la diverse olimpiade se realizează astfel:

- se alocă două caractere pentru codul țării în care s-a desfăşurat olimpiada;
- se introduce în interiorul codului cuvântul OLIMPIC pentru a oferi o descriere completă a respectivelor fişiere;
- se alocă şapte caractere pentru denumirea oraşului unde s-a desfăşurat olimpiada; în cazul în care denumirea oraşului este mai mică de şapte caractere, pozițiile rămase neocupate se completează cu spațiu sau underscore;
- se alocă patru caractere pentru a reprezenta numărul pozei în cadrul colecției; pentru a asigura deschiderea codificării, numerotarea fişierelor se realizează astfel: 0010, 0020, 0030, etc.

Conform acestei variante de codificare, denumirile fişierelor reprezentând poze ale olimpicilor sunt de forma:

JP**OLIMPIC**TOKYO__0010

SP**OLIMPIC**BARCELO0020

RO**OLIMPIC**BUCURES0030

Dacă după ce s-a realizat codificarea se mai include în colecție o nouă poză, atunci trebuie identificată poza cu denumirea cea mai apropiată de denumirea noului fişier. Dacă noua poză este din Madrid, atunci colecția arată astfel:

JP**OLIMPIC**TOKYO__0010

SP**OLIMPIC**BARCELO0020

SP**OLIMPIC**MADRID_0021

RO**OLIMPIC**BUCURES0030

Procesul de autotestare este iterativ şi complet. Autotestarea se face de la A la Z de fiecare dată, deoarece:

- modificările au caracter de antrenare multiplă;
- corecțiile trebuie să aibă în final garanția că sunt complete.

Metricile pentru evaluarea efectelor produse de erorile apărute în procesul de dezvoltare şi autotestare a aplicației de scrolling sunt următoarele:

- pierderea suportată de dezvoltatorul aplicației de scrolling, PD, ca urmare a erorilor apărute în procesul de dezvoltare a aplicației:

PD = FI * NE, unde

FI – factorul de impact ;

NE – numărul de erori apărute în procesul de dezvoltare a aplicației.

În cazul dezvoltării aplicației de scrolling pentru revista JACS, s-a identificat un număr de 18 erori, valoarea factorului de impact fiind 0.3 mii de euro per eroare. Pierderea suportată de dezvoltatorul aplicației de scrolling, în acest caz, a fost de 5.400 euro.

- numărul de erori pe dimensiunea aplicației de scrolling, NED:

$NED = \frac{NEA}{DAS}$, unde:

NEA – numărul total de erori identificate în aplicație;

DAS – dimensiunea aplicației de scrolling, măsurată ca număr de linii de cod (LOC sau KLOC).

Pentru aplicația de scrolling referitoare la revista JACS, numărul total de erori identificate a fost 18, iar dimensiunea aplicației de scrolling este 4793 linii de cod. Rezultă că numărul de erori pe dimensiunea aplicației de scrolling este 0.00375.

- costul autotestării aplicației de scrolling, CTS:

$CTS = \sum_{i=1}^{N} CTC_i$, unde:

N – numărul de componente ale aplicației de scrolling;

CTC_i – costul autotestării componentei *i*.

Numărul de componente ale aplicației de scrolling referitoare la revista JACS este reprezentat de numărul elementelor din mulțimea TS, respectiv 10 elemente. Costul autotestării fiecărei componente s-a stabilit experimental la valoarea de 7 euro pe fiecare componentă, ceea ce indică un cost total al autotestării aplicației de scrolling de 70 de euro.

Trebuie să se asigure echilibru între dimensiunea modelului şi capacitatea acestuia de a oferi rezultate semnificative. În aceste condiții, metricile nu trebuie să fie prea simple, deoarece se pierde din relevanța variabilelor măsurate.

În funcție de problema de rezolvat, de cerințele utilizatorilor şi de resursele financiare ale dezvoltatorului aplicației de scrolling se definesc:

- durata de realizare a aplicației, ca fiind perioada cuprinsă între etapa de elaborare a specificațiilor până la terminarea etapei de testare şi lansarea pe piață;
- nivelul de complexitate, măsurat în funcție de numărul de componente ale aplicației, conexiunile cu baze de date şi interacțiunea cu utilizatorii;
- nivelul de calitate, rezultat în urma testelor efectuate şi 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 şi ceea ce s-a realizat efectiv;
- conexiunea cu alte aplicații, din care rezultă gradul de integrabilitate.

Obiectivul informaticii orientate spre cetățean este de a dezvolta aplicații foarte diverse şi care închid ciclul specific unei acțiuni.

Testarea aplicațiilor informatice se realizează atât de către dezvoltatorul aplicației, situație în care se vorbeşte despre autotestare, cât şi de către echipe de testeri din cadrul organizației în care se utilizează.

Dezvoltarea unor noi funcționalități ale serviciului de internet banking din cadrul unei bănci presupune testarea aplicației de către angajații băncii, înainte de lansarea acesteia în producție. Firma dezvoltatoare a aplicației îi livrează băncii aplicația conținând noile funcționalități implementate, urmând ca testarea să se realizeze de către bancă. În situația în care banca descoperă erori în cadrul aplicației, aceasta le va transmite către firma dezvoltatoare pentru a le corecta. Dacă aplicația nu are erori, atunci ea este lansată în producție şi utilizată de către clienții băncii.

În figura 15 se prezintă schema logică a procesului de testare a unei aplicații de internet banking de către beneficiarul aplicației.

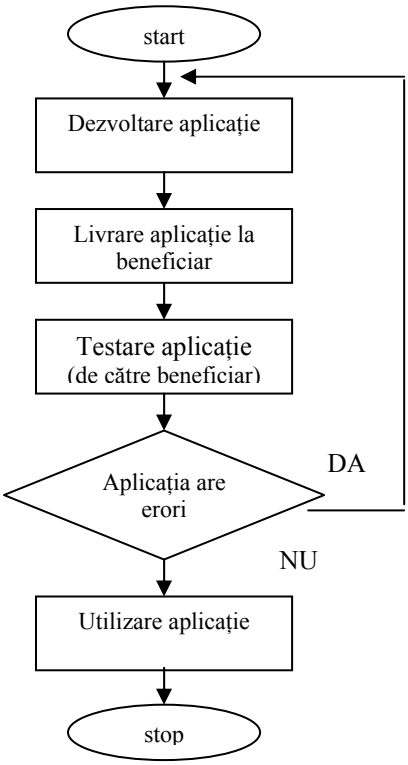

Figura 15. Testarea de către beneficiar

Pentru a reduce costurile cu testarea, tot mai multe companii dezvoltatoare de aplicații informatice recurg la testarea realizată direct de către utilizatorii finali. Aceştia sunt stimulați cu ajutorul unor premii sau sume de bani să descopere vulnerabilitățile sau problemele majore ale respectivelor aplicații [6]. Această modalitate de testare este aplicată în situația în care produsul software lansat în producție nu prezintă erori majore de funcționare, care să determine afectarea imaginii companiei dezvoltatoare.

Autotestarea este diferită de testarea totală, care constă în următoarele:

- verificarea, element cu element, a corectitudinii şi completitudinii celor n elemente din mulțimea TS;
- verificarea, element cu element, a corectitudinii şi completitudinii celor k elemente din mulțimea TE;
- analiza perechilor (TS_i, TE_i) din punct de vedere al concordanței, corectitudinii şi completitudinii.

5. Concluzii

Aplicațiile de scrolling sunt orientate spre cetățean şi declanşează alocări de resurse. Numai prin autotestare se gestionează ciclul de dezvoltare a aplicației, respectiv lungimea ciclului ca durată.

Aplicațiile de scrolling sunt componente în aplicații agregate. Poziția dezvoltatorului este esențială prin autotestare. O autotestare foarte bună, completă, garantează calitatea aplicației de scrolling, dar şi poziția dezvoltatorului acesteia.

Autotestarea nu exclude testarea de către altcineva. Dacă autotestarea şi testarea se realizează pe listele complete, atunci totul este ok.

Dacă autotestarea se realizează pe listele complete, iar testarea nu, dar autotestarea este ok, atunci şi testarea are şanse să fie ok.

Dacă listele sunt incomplete în autotestare şi complete în testare, atunci procesul de dezvoltare se reia pentru corecții, cu costuri foarte mari.

Dacă listele sunt incomplete atât la autotestare, cât şi la testare, atunci aplicația include

riscuri majore de a efectua prelucrări neadecvate.

Autotestarea este specifică procesului de dezvoltare. Ciclul de dezvoltare este format din etape, iar testarea se realizează atât în cadrul fiecărei etape, cât şi în final, pentru a lua fie decizia de implementare, fie decizia de continuare a ciclului de dezvoltare prin efectuarea de corecţii.

Autotestarea aplicaţiilor de scrolling presupune:

- o tehnologie de a constitui scrolling;
- existenţa unui text T, care se rupe în n texte mai mici, $ST_1, ST_2, ..., ST_n$, a căror reuniune formează textul T;
- din textul ST_j se preia un rezumat pentru a constitui elementul TS_j;
- gestiunea perechilor (ST_j, TS_j) şi codificarea fişierelor asociate, astfel încât să nu existe erori de asociere.

Codificarea elementelor TS se realizează astfel încât să permită identificarea cu uşurinţă a erorilor de asociere între un element din mulţimea TS şi varianta lui extinsă.

Autotestarea este diferită de testare prin obiective, sarcini şi prin persoanele care o efectuează. Dacă testarea este efectuată de personal specializat, autotestarea este realizată de către dezvoltatorul aplicaţiei.

Crearea procedurilor de autotestare, însuşirea şi aplicarea lor întocmai, are ca efect reducerea ponderii costului etapelor de autotestare şi de testare în structura costului total generat de realizarea oricărei aplicaţii de scrolling.

BIBLIOGRAFIE

1. **SAD, H. H.; F. POIRIER.** Evaluation and Modeling of User Performance for Pointing and Scrolling Tasks on Handheld Devices Using Tilt Sensor, Second International Conf. on Advances in Computer-Human Interactions - ACHI '09, 1-7 Feb. 2009, pp. 295-300.
2. **IVAN, I.; B. VINTILĂ; C. CIUREA; D. PALAGHITA; S. PAVEL.** Autotesting of the citizen oriented informatics applications, Ekonomika, statistika i informatika. Vestnik UMO, MESI, Russia, No. 4, 2009, ISSN 1994-7844.
3. **IVAN, I.; C. CIUREA; D. MILODIN.** Autotestarea aplicaţiilor informatice distribuite cu conţinut oferit, Revista Română de Informatică şi Automatică, Vol. 19, Nr. 3, 2009.
4. **BAKER, S.; F. AU; G. DOBBIE; I. WARREN.** Automated Usability Testing Using HUI Analyzer, 19th Australian Conf. on Software Eng. - ASWEC 2008, 26-28 March , pp. 579-588.
5. **IVAN, I.; P. POCATILU.** Testarea automată a produselor software specializate, Revista Informatica Economică, vol. VIII, nr. 2(30), 2004, pp. 116-120.
6. http://www.incont.ro/it-c/poti-castiga-bani-doar-stand-cu-ochii-pe-google-si-mozilla.html
7. **CĂPRIŢA, D.** Automated Web Database Applications Testing, The Proc. of the 9th Int. Conf. on Informatics in Economy - IE 2009, May 7-8, 2009, Bucharest, ASE Printing House.
8. **POCATILU, P.** Software Testing Costs, ASE Publishing House, Bucharest, 2004.
9. **POCATILU, P.** Software Security Testing, Informatica Economica Journal, Vol. IX, No. 4(36), 2005, ISSN 1453-1305, pp. 78-82.
10. **TARHINI, A.; Z. ISMAIL; N. MANSOUR.** Regression Testing Web Applications, ICACTE '08. International Conference on Advanced Computer Theory and Engineering, 2008, 20-22 Dec. 2008, pp. 902-906.
11. **MURLEWSKI, J.; J. WOJCIECHOWSKI; B. SAKOWICZ; A. NAPIERALSKI.** Comprehensive Approach to Web Applications Testing and Performance Analysis, 9th Int. Conf. - The Experience of Designing and Appl. of CAD Systems in Microelectronics, 2007, CADSM '07, 19-24 Feb. 2007, pp. 429-431.
12. **GODEFROID, P.; P. DE HALLEUX; A. V. NORI; S. K. RAJAMANI; W. SCHULTE; N. TILLMANN; M. Y. LEVIN.** Automating Software Testing Using Program Analysis, Software IEEE, Vol. 25, No. 5, Sept.-Oct. 2008, pp. 30-37.