Noi abordări în evaluarea automată a utilizabilității

Adriana-Mihaela Guran
Universitatea Babeş-Bolyai
Facultatea de Matematică şi Informatică
Str. M. Kogălniceanu nr. 1, Cluj-Napoca
adriana@cs.ubbcluj.ro

Daniela-Maria Onacă
Universitatea Babeş-Bolyai
Facultatea de Psihologie şi Ştiinţe ale Educaţiei
Republicii nr. 37, Cluj-Napoca
daniela.onaca@apio.ro

Horia D. Pitariu
Universitatea Babeş-Bolyai
Facultatea de Psihologie şi Ştiinţe ale Educaţiei
Republicii nr. 37, Cluj-Napoca
horia.pitariu@apio.ro

REZUMAT

În acest articol sunt prezentate două abordări noi în evaluarea automată a utilizabilității. Este vorba de folosirea unor paradigme moderne de programare, precum programarea orientată pe aspecte şi programarea orientată pe agenți pentru a identifica probleme de utilizabilitate.

Cuvinte cheie

Utilizabilitate, programarea orientată pe aspecte, agenți inteligenți, sarcină, proiectare centrată pe utilizator

Clasificare ACM

H5.2. Information interfaces and presentation (e.g., HCI): Evaluation/methodology.

INTRODUCERE

Utilizabilitatea a fost definită în numeroase forme de-a lungul timpului. De o largă acceptabilitate se bucură definiția ISO care stabileşte criteriile utilizabilității a fi eficiența, eficacitatea şi satisfacția utilizatorului.
Evaluarea utilizabilității se realizează prin diverse metode, cele mai eficiente şi în acelaşi timp cele mai costisitoare, sunt metodele care folosesc testarea cu utilizatori reali. Tipic, la un test de utilizabilitate evaluatorul propune un scenariu de utilizare a sistemului informatic. Sarcinile incluse în scenariu au scopul de a testa anumite aspecte ale interfeței utilizator. În timpul testului de utilizabilitate specialistul în utilizabilitate observă modul în care utilizatorul execută sarcinile, comportamentul său şi comentariile pe care le face. La sfârşitul testului are loc o discuție privind modul de realizare a sarcinilor şi gradul de satisfacție al utilizatorului în raport cu sistemul folosit. Deoarece producătorii de sisteme informatice sunt interesați de dezvoltarea de sisteme cu un grad cât mai sporit de utilizabilitate, pentru aceştia este de nevoie de dezvoltarea de metode mai puțin costisitoare pentru realizarea evaluării utilizabilității unui sistem informatic. Una din metodele cele mai uşor de aplicat în mod automat este măsurarea unor metrici ale utilizabilității precum numărul de sarcini realizate cu succes, numărul de sarcini eşuate, timpul de execuție a unor sarcini, numărul şi frecvența erorilor, frecvența de utilizare a unor funcționalități ale aplicației, frecvența de accesare a documentației de utilizare, etc.

Evaluarea utilizabilității este un aspect major din dezvoltarea de interfețe utilizator care încearcă identificarea problemelor de utilizabilitate. Indiferent de abordarea adoptată în evaluarea utilizabilității, sunt efectuate următoarele trei activități:

- *achiziția* – se referă la culegerea de date de utilizabilitate, precum timpul de efectuare a unei sarcini, încălcări ale regulilor de proiectare, erori;
- *analiza* – se referă la interpretarea datelor de utilizabilitate pentru identificarea problemelor;
- propunerea de *sugestii de îmbunătățire* a interfeței utilizator [2].

Primele două activități din cele mai sus amintite pot fi supuse unui proces de automatizare.

Avantajele pe care le aduce automatizarea evaluării utilizabilității sunt: reducerea necesarului de resurse umane, reducerea necesarului de expertiză umană, completitudine şi posibilitatea comparării rezultatelor testelor de utilizabilitate.

Automatizarea procesului de evaluare a interfeței utilizator necesită jurnalizarea evenimentelor din interacțiunea dintre utilizator şi sistemul supus analizei. Jurnalizarea şi analiza evenimentelor care apar la nivelul interfeței utilizator au fost recunoscute a fi surse valoroase de determinare a problemelor de utilizabilitate. Prin jurnalizarea evenimentelor din interfața utilizator se pot obține informații precise despre ce face utilizatorul, ce date foloseşte, ce funcționalități accesează, în ce ordine execută acțiunile. Astfel, metrici ale utilizabilității precum timpul de execuție a unei sarcini, frecvența de folosire a unei funcționalități sau numărul de erori pot fi calculate din fişierele de jurnalizare.

Desigur, există şi alte modalități de evaluare a acestor metrici (înregistrarea video a sesiunii de testare, notarea manuală), dar folosind jurnalizarea evenimentelor acest lucru se poate realiza automat. Datele înregistrate în fişierele de jurnalizare au un format specific care poate fi transformat în alte formate, operație urmată de supunerea datelor spre analiză. Jurnalizarea automată a evenimentelor din interfața utilizator are şi neajunsuri, unul din acestea fiind imposibilitatea surprinderii reacțiilor utilizatorului. În mod obişnuit jurnalizarea se realizează prin inserarea unor linii de cod în cadrul codului aplicației pentru a se surprinde elementele de interes din cadrul interacțiunii. Această abordare necesită o foarte bună cunoaştere a structurii aplicației şi acces la codul sursă al aplicației. Decizia asupra locațiilor în care să se facă instrumentarea trebuie să aparțină unui expert în utilizabilitate care să colaboreze cu dezvoltatorii sistemului. În secțiunile următoare propunem două noi abordări în evaluarea automată a utilizabilității şi în identificarea problemelor de utilizabilitate.

O ABORDARE BAZATĂ PE PROGRAMAREA ORIENTATĂ PE ASPECTE PENTRU EVALUAREA UTILIZABILITĂȚII

Programarea orientată pe aspecte (AOP) este o paradigmă recent dezvoltată care se adresează situațiilor de cerințe încrucişate (o caracteristică a unui sistem a cărei implementare este distribuită în tot sistemul) [3]. Pentru implementarea unor astfel de cerine AOP introduce patru noțiuni noi :

- *join-point* - punct bine definit în execuția programului;
- *pointcut* - grupează o mulțime de join-pointuri şi expune anumite valori din contextul de execuție al join-point-urilor;
- *advice* - secvență de cod care se execută la fiecare join-point dintr-un pointcut;
- *aspect* - este un tip care încapsulează pointcut-uri, advice-uri şi proprietăți statice [2].

Un *aspect* este unitatea de modularizare a AOP. Aspectele sunt integrate într-un sistem folosind un instrument special care se numeşte *weaver*. Există extensii AOP pentru limbaje de programare bine-cunoscute precum Java, C++ sau C#.

Activitatea de culegere de date din evaluarea utilizabilității constă în jurnalizarea evenimentelor care apar în timpul interacțiunii utilizator-sistem precum apăsarea unui buton, traversarea unui meniu, alegerea unei comenzi, începerea execuției unei sarcini. Inserarea codului necesar jurnalizării în sistemele deja dezvoltate necesită modificarea tuturor modulelor interfeței grafice, dar folosind AOP este necesară doar scrierea unor aspecte şi integrarea acestora folosind un weaver. Activitatea de analiză constă în evaluarea unor metrici de utilizabilitate. Folosind AOP calculul metricilor de utilizabilitate poate fi realizat în timpul interacțiunii.

Avantajele pe care le aduce folosirea AOP în evaluarea utilizabilității sunt:

- unele metrici pot fi calculate automat scăzând efortul necesar analizei;
- evenimentele sunt jurnalizate cu uşurință pentru analiza detaliată ulterioară;
- dacă se consideră necesară tratarea unor noi situații, sunt necesare doar modificări la nivelul aspectelor.

În cele ce urmează prezentăm o proiectare posibilă pentru un modul de evaluare a utilizabilității folosind AOP. Este necesară identificarea punctelor din fluxul de programului de care suntem interesați. Primul pas este determinarea punctului din care începe măsurarea, notat prin *entryPoint()* şi a punctului în care se încheie măsurarea, notat prin *exitPoint()*. Acestea vor constitui pointcut-urile din aspectul denumit AppAspect. Apoi, depinzând de informațiile pe care vrem să le culegem, se definesc aspecte noi.

S-a realizat un studiu de caz pe două aplicații de dimensiuni medii. Pentru evaluarea automată a utilizabilității am efectuat următorii paşi:

- înregistrarea fiecărei erori împreună cu momentul apariției;
- preluarea unei capturi de ecran la apariția fiecărei erori pentru a vizualiza datele introduse de utilizator;
- calcularea frecvenței pentru fiecare tip de eroare;
- calcularea timpului de execuție a fiecărei sarcini;
- calcularea numărului de sarcini încheiate cu succes, abandonate (se începe execuția sarcinii) sau eşuate (se începe execuția sarcinii dar apar erori).

Acest lucru înseamnă că se vor scrie aspecte pentru diferitele metrici care se doresc a fi evaluate. Avantajul pe care îl aduce abordarea propusă de noi este acela că modulele care realizează procesele de evaluare a utilizabilității sunt separate de codul sursă al aplicației. În plus, ataşarea acestor module la aplicație sau modificările apărute în modulele de evaluare a utilizabilității nu necesită recompilarea codului aplicației, ci doar o recompilare a aspectelor.

Aspectele din modulul de evaluare a utilizabilității sunt prezentate în Fig.1.

Cercetările ulterioare vor fi îndreptate în următoarele direcții:

- identificarea unor metode de evaluare a unor măsuri calitative ale utilizabilității (satisfacției utilizatorilor);
- verificarea aplicabilității acestei abordări pentru aplicații web;
- dezvoltarea unui cadru general de evaluare a utilizabilității bazat pe Programarea Orientată pe Aspecte.

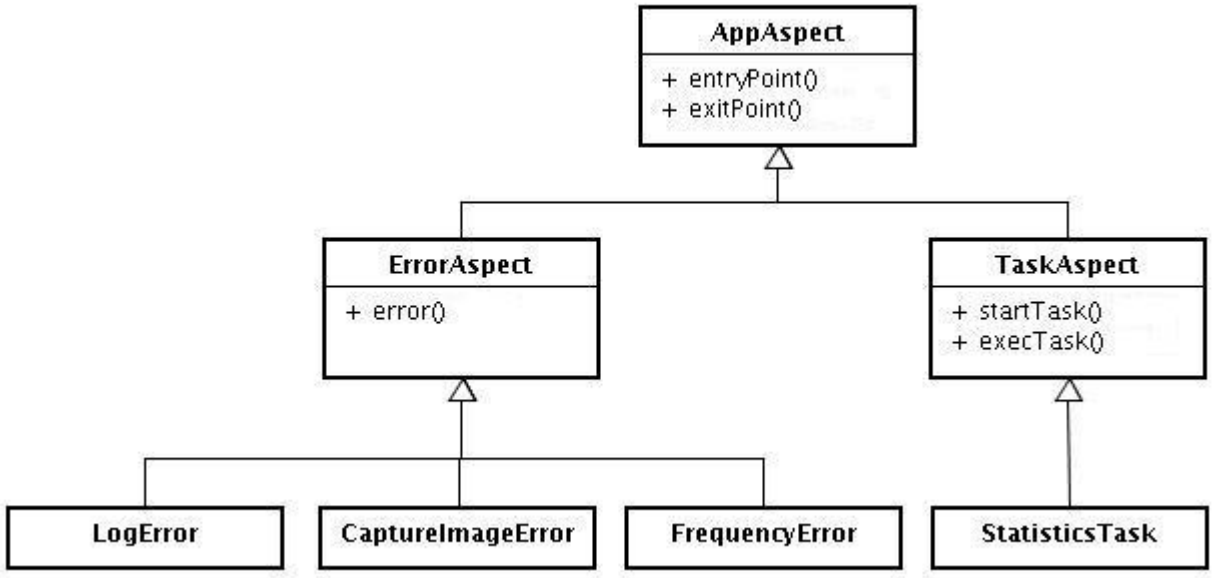

Figura 1 – Ierarhia de aspecte

O ABORDARE BAZATĂ PE AGENȚI PENTRU EVALUAREA UTILIZABILITĂȚII

Am extins abordările noastre spre evaluarea automată a utilizabilității prin folosirea *agenților* pentru identificarea problemelor de utilizabilitate. Un *agent* [4] este o entitate care își percepe mediul prin senzori și acționează asupra mediului prin acțiuni. Un agent este caracterizat prin *partea de arhitectură* – sau comportamentul agentului dat de acțiunile agentului după o secvență de percepții și *partea de program* – care se referă la funcționalitatea agentului. Un sistem utilizabil se bucură de o compatibilitate sporită între modelul sarcinilor utilizator și modelul sarcinilor proiectant. Abordarea propusă în această secțiune încearcă identificarea acelor puncte din interacțiune în care există divergențe între modul în care este proiectat sistemul și modul în care utlizatorii își imaginează că se folosește respectivul sistem.

Abordarea propusă folosește agentul pentru a compara modelul sarcinilor utilizator cu modelul conceptual al proiectantului. Mediul în care acționează agentul este un mediu soft. Agentul folosește AOP pentru a culege informații din mediu. În Fig. 2 propunem o arhitectură generală pentru un sistem bazat pe agenți pentru evaluarea utilizabilității. Componentele sistemului sunt:

Aplicația soft (*SA*) - reprezintă sistemul soft a cărui utilizabilitate o evaluăm. Aplicația are un model (arbore) al sarcinilor asociat (**M**odelul **S**arcinilor proiectant) dezvoltat de proiectantul interfeței utilizator;

Utilizatorul (*U*) – este o persoană care folosește sistemul *SA* pentru a îndeplini un set predefinit de sarcini;

Modulul de învățare – în arhitectura generală acest agent învață comportamentul utilizatorului și transmite feedback evaluatorului interfeței utilizator.

Modulul de învățare este folosit pentru a culege informații despre evenimentele generate de utilizator cu interfața: apăsări de mouse, completare de câmpuri text, alegerea de opțiuni din meniuri, etc. Aceste evenimente sunt receptate de agent și pe baza lor se reconstruieşte modelul sarcinilor utilizator (**MS** utilizator).

Agentul are ca și cunoștințe inițiale modelul conceptual al sarcinilor sau modelul sarcinilor proiectant (MS proiectant). În implementarea noastră, cunoștințele inițiale sunt stocate într-un fișier XML. Modulul de evaluare al agentului, care îi furnizează acestuia comportamentul, compară MS utilizator cu MS proiectant, verificând dacă MS utilizator este un subarbore al MS proiectant și salvează rezultatul comparației pe un dispozitiv de stocare. Modulul AOP este folosit pentru a capta evenimentele utilizator care sunt folosite de agent pentru a reconstrui modelul sarcinilor utilizator (MS utilizator). Cercetările noastre se focalizează spre construcția modelului de învățare care va spori acuratețea evaluării. Un alt aspect de interes în cadrul cercetărilor noastre este determinarea unor măsuri de calitate ale rezultatelor evaluării și de a compara rezultatele produse de agent cu rezultatele altor metode de evaluare a utilizabilității (SUMI [5], SUS [1]).

Cercetările viitoare se vor îndrepta în următoarele direcții:

- adăugarea modulului de învățare versiunii curente a sistemului, crescând astfel acuratețea evaluării utilizabilității;
- definirea de măsuri de calitate a rezultatelor evaluării utilizabilității;
- aplicarea abordării noastre la sisteme complexe;
- compararea rezultatelor obținute cu rezultatele altor abordări [1], SUMI [5].

CONCLUZII

În această lucrare au fost prezentate două direcții de cercetare noi în domeniul evaluării automate a

utilizabilității sistemelor informatice. Avantajele abordării automatizate în evaluarea utilizabilității sunt remarcabile, însă rămâne sensibil aspectul evaluării satisfacției utilizatorului. Cercetările ulterioare se vor îndrepta în direcția identificării de soluții care să cuprindă şi acest aspect.

CONFIRMARE (MULȚUMIRI)

Aceste cercetări au fost posibile prin sprijinul oferit din proiectul de cercetare CRONIS, contractul nr. 11-003/2007, din cadrul programului de cercetare PN II, Programul 4 -"Parteneriate in domenii prioritare", finanțat de Guvernul României.

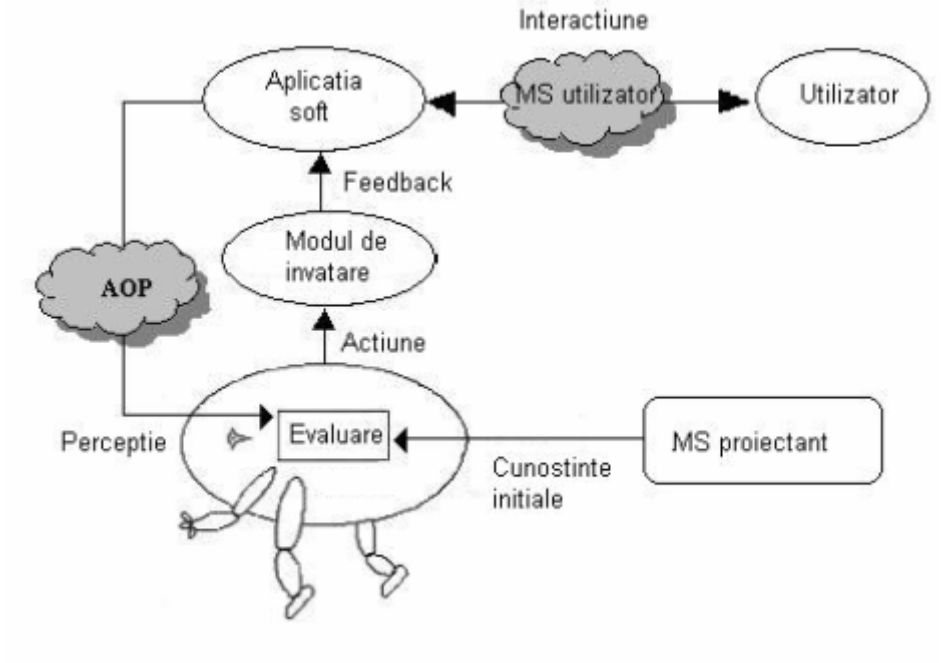

Figura 2 Arhitectura sistemului

REFERINȚE

1. J. Brooke. SUS - A “quick and dirty” usability scale. http://www.cee.hw.ac.uk/ph/sus.html. User information Architecture A/D Group, Digital Equipment Co. Ltd.
2. G. Kiczales, J. Lamping, A. Menhdhekar, C. Maeda, C. Lopes, J.-M. Loingtier, and J. Irwin. Aspect-Oriented Programming. In Proceedings European Conference onObject-Oriented Programming, volume 1241, pages 220–242. Springer-Verlag, 1997.
3. D. Kieras and P. G. Polson. An approach to the formal analysis of user complexity. Int. Journal Human-Computer Study, 51(2):405–434, 1999.
4. S. Russell and P. Norvig. Artificial Intelligence - A Modern Approach. Prentice-Hall, Inc., 2003
5. SUMI Inventory, 1996. http://www.ucc.ie/hfrg/questionnaires/sumi/.