Factori de influenta a testarii

Asist. Paul POCATILU
Catedra de Informatica Economica, A.S.E. Bucuresti

Software testing - an important activity that helps to increase the quality of software application - is influenced by many factors. The effort associated with testing is very big. Identifying and controlling some factors that affect software testing could lead as to less testing effort and less costs with testing. In this paper are described the most important factors that affect software testing and the cost associated with it.
***Keywords**: software testing, software complexity, software quality.*

1 Clasificarea factorilor de influenta a testarii

Testarea software este influentata de numerosi factori. Varietatea acestor factori impune existenta unor criterii de clasificare a lor precum si luarea în considerare a acelora care influenteaza în mod semnificativ procesul de testare. Cu cât influenta factorilor este mai puternica si valoarea acestora este mai mare, testarea aplicatiei software devine mai complexa, necesitând un efort mai mare si deci un cost mai ridicat.

Exista numeroase criterii de clasificare a factorilor. Se identifica o serie de clasificari dupa modul de actiune a factorilor, dupa raportul dintre factori si procesul de dezvoltare software.

Dupa modul de actiune se identifica *factori cu actiune directa* si *factori cu actiune indirecta.* Astfel, factori cu actiune directa sunt:

- complexitatea produsului software,
- gradul de omogenitate a echipei de realizare software,
- tehnicile de dezvoltare utilizate
- stadiul de certificare a firmei
- numarul de utilizatori
- specificul aplicatiei software

Principalii factori cu actiune indirecta sunt:

- existenta de produse similare pe piata
- gradul de noutate software si hardware.

Testarea software este influentata atât de *factori externi* cât si *factori interni*. În cadrul acestui criteriu de clasificare se are în vedere locul factorilor de influenta în raport cu procesul de dezvoltare software.

Tabelul 1 – Clasificarea factorilor de influenta a testarii software

Factor	Tip
Complexitatea produselor software	Intern, Direct
Gradul de omogenitate a echipei de realizare software	Intern, Direct
Tehnicile de dezvoltare utilizate în realizarea produsului	Intern, Direct
Stadiul de certificare a firmei	Intern, Direct
Numarul utilizatorilor	Extern, Direct
Existenta de produse similare pe piata	Extern, Indirect
Gradul de noutate software si hardware	Extern, Indirect
Specificul aplicatiei software	Intern, Direct

Un alt criteriu de clasificare a factorilor îl constituie posibilitatea de cuantificare a acestora. Astfel, exista *factori cuantificabili* si *factori necuantificabili.* Factorii cuantificabili pot fi masurati în schimb pentru cei necuantificabili nu exista posibilitatea de masurare a lor.

Factorii care influenteaza testarea software pot fi clasificati si în *factori obiectivi* si *factori subiectivi.*
În tabelul 1 sunt prezentati o serie de factori care influenteaza testarea software precum si încadrarea acestora în categoriile de clasificare prezentate.

2. Complexitatea software
Complexitatea produsului software este poate cel mai important factor care influenteaza testarea . S-a constat ca pe masura ce complexitatea unui program este mai mare, cu atât probabilitatea ca programul sa contina erori este mai ridicata.
Exista mai multe metode de determinare a complexitatii produselor program, dintre care cea mai utilizata în testare este complexitatea McCabe. McCabe [McCA76] a construit o masura a complexitatii prin intermediul numarului ciclomatic. Numarul ciclomatic se determina în mai multe moduri, dintre care cel mai simplu este de a aduna valoarea unu la numarul de blocuri de decizie din program.
S-a stabilit o relatie între numarul ciclomatic si numarul minim de cazuri de test necesare pentru parcurgerea tuturor cailor independente din modulul sau programul care se testeaza. Cu cât complexitatea este mai ridicata, cu atât mai multe exemple de test sunt necesare, rezultând un efort mai mare pentru testare.
În cazul în care complexitatea McCabe este mare – numarul ciclomatic are o valoare peste 10 – o posibilitate de reducere a acesteia consta în împartirea modulului sau programului în doua sau mai multe parti, fiecare parte având o complexitate mai mica.
O alta metrica de complexitate utilizata este cea dezvoltata de Halstead, calculata pe baza operatorilor si a operanzilor din cadrul unui program. În [BEIZ90] este calculat numarul de erori care exista în program pe baza complexitatii Halsted.

3. Dimensiunea aplicatiei
Principale moduri de masurare a *dimensiunii aplicatiei* sunt numarul de linii de cod sursa (LOC – Lines of Code), de cele mai multe ori exprimate în mii (KLOC) si puncte functionale (PF – Point Function).
LOC reprezinta numarul de linii sursa ale unui program, fara includerea comentariilor sau a elementelor de documentare. În mod frecvent se utilizeaza sub forma KLOC (Kilo LOC), mii de linii sursa. Masurarea pe baza liniilor de cod sursa LOC/KLOC este dependenta de limbajul de programare. În [PRES00] este prezentata corespondenta dintre numarul de linii sursa pentru o serie de limbaje de programare, tabelul 2.

Tabelul 2 – Corespondenta LOC pentru diverse limbaje de programare

Limbajul de programare	**Linii sursa**
Limbaj de asamblare	320
C	128
COBOL	106
Pascal	90
C++	64
Visual Basic	32
SQL	12

Punctele functionale masoara indirect software si procesul prin care acesta este dezvoltat. Punctele functionale se axeaza asupra functionalitatii sau utilitatii programului. Punctele functionale sunt derivate utilizând o relatie empirica dintre domeniul informational al programului si evaluarea complexitatii software. Punctele functionale sunt independente de limbajul de programare în care s-a realizat aplicatia software.
Pe baza informatiilor culese pe masura dezvoltarii software se calculeaza o serie de indicatori privind productivitatea, calitatea, costul si documentatia ce revin la o unitate de dimensiune (KLOC sau FP).

4. Gradul de omogenitate a echipei de dezvoltare software
Productia de software este o activitate de echipa. Personalul este înzestrat cu calitati individuale, dar un rol esential îl are capacitatea fiecarui specialist certificat de a se integra în echipa, de a produce componen-

te care prin asamblare sa conduca la obtinerea produsului finit ca întreg.
Un rol esential îl are comunicarea în cadrul echipei precum si cunoasterea de catre coordonatorul acesteia a potentialului fiecarui membru. În acest fel are loc distribuirea de sarcini care corespunde acestui potential, ceea ce conduce la încadrarea în grafice a consumurilor de resurse si la asigurarea parcurgerii etapelor în termenele planificate.
Personalul specializat în testarea software trebuie sa cunoasca tehnicile de analiza, proiectare si programare utilizate, sa înteleaga problema pe care programul o rezolva si sa aiba capacitatea de a distinge diferentele care apar între rezultatele oferite de program si cele obtinute prin specificatii sau rezultate proprii.
Gradul de omogenitatea a echipei are o importanta foarte mare în desfasurarea testarii software. Daca echipa este neomogena sau cu un grad de omogenitate scazut, apar o serie de probleme în cadrul procesului de testare, legate de comunicare, lipsa de experienta neîntelegeri între membrii echipei.

5. Concluzii

Influenta acestor factori se manifesta în special asupra efortului de testare. Efortul de testare este masurat fie în unitati de timp (ore, luni, ani), fie în alte unitati de masura specifice resurselor implicate în testare (persoane). Cunoscând costul pe unitatea de timp alocata testarii sau costul pe unitatea de resursa rezulta costul testarii indus de factorii ce influenteaza testarea software. Cu cât efortul de testare este mai mare cu atât mai mult cresc si costurile asociate acestui proces. În tabelul 3 sunt prezentate relatiile dintre factorii care influenteaza testarea si efortul de testare.

Tabelul 3 – Influenta factorilor asupra testarii software

Factor	**Efort crescut de testare**	**Efort scazut de testare**
Complexitate	Ridicata	Scazuta
Dimensiune	Ridicata	Scazuta
Gradul de omogenitate a echipei de dezvoltare software	Scazut	Ridicat
Limbajul de programare utilizat	Nivel scazut	Nivel înalt
Tehnicile utilizate	Neorientate pe reutilizare	Orientate pe reutilizare
Gradul de noutate software si hardware	Ridicat	Scazut
Numarul si diversitatea utilizatorilor	Mare	Mic
Gradul de certificare a firmei	Scazut	Ridicat
Existenta de produse similare pe piata	Exista	Nu exista

Pentru reducerea efortului testarii software este necesar sa se controleze acesti factori care influenteaza procesul de testare, activitate care este destul de dificil de realizat tinând cont de diversitatea factorilor si de imposibilitatea de cuantificare a unora dintre acestia.

Bibliografie

[ARHI00] Arhire, Romulus – *Evaluarea complexitatii sistemelor de programe – Teza de doctorat*, Academia de Studii Economice, Bucuresti, 2000
[BEIZ90] Beizer, Boris, *Software Testing Techniques – Second Edition*, Van Nostrad Reinhold, New York, 1990

[BOEH00] Boehm, Barry W. et al – *Software Cost Estimation with COCOMO II*, Prentice Hall, 2000

[MCCA76] McCabe, T. J., – *A Complexity Measure*, IEEE Transactions on Software Engineering, 2/1976, pag 308-320.

[IVAN00b] Ivan, Ion, Teodorescu, Laurentiu, Pocatilu, Paul – *Cresterea calitatii software prin testare*, în Q-media, vol 2, no.5, 2000, pag 20-24

[POCA01b] Pocatilu, Paul – *The Role of Software Testing in Increasing the Software Reliability*, Lucrarile "Software Reliability Workshop", Bucuresti, 2001

[POCA02] Pocatilu, Paul – *Costurile testarii software*, în Informatica Economica vol. VI, nr. 1, 2002, pag. 90-93

[PRES00] Pressman, Roger S. – *Software Engineering – A Practitioner's Approach, European Adaptation Fifth Edition*, McGraw-Hill, 2000