MODELE PENTRU ESTIMAREA COSTULUI FIABILITĂȚII SOFTWARE

Ion Ivan

Marius Popa

Academia de Studii Economice, Bucureşti

Marian Cristescu

Universitatea "Lucian Blaga" Sibiu

Rezumat: Această lucrare evidențiază o serie de elemente privind tehnicile şi metodele utilizate în estimarea costului fiabilității software. Sunt prezentate caracteristici pe baza cărora este definit conceptul de calitate software. De asemenea, se evidențiază conceptul de fiabilitate software şi modele asociate acesteia. Se defineşte costul calității software şi sunt prezentate categorii de cost de acest tip, precum şi caracteristici privind costul fiabilității software şi factorii care influențează acest tip de cost.

Cuvinte cheie: calitate software, fiabilitate software, cost, estimare.

1. Cerințe ale calității software

Problema determinării, asigurării şi evaluării calității este o problemă destul de veche; ea a fost abordată de mult timp ceea ce a făcut ca, încă de la sfârşitul anilor '80, I.S.O. să publice primele standarde referitoare la calitate. Conform standardului ISO-8402, calitatea este ansamblul caracteristicilor unei entități, care îi conferă aptitudinea de a satisface nevoile exprimate sau implicite [9]. Aceasta se exprimă printr-un sistem de caracteristici, nu este de sine stătătoare şi se defineşte în relația cu terții - clienți, producători, furnizori, sau comercianți. Este o variabilă continuă şi presupune satisfacerea atât a nevoilor exprimate, cât şi a celor implicite. Conceptul de calitate este tratat, în prezent, ca un concept dinamic, ce leagă calitatea produsului sau a serviciului de alte două elemente fundamentale: calitatea proiectului şi calitatea fabricației.

Asigurarea calității produsului final se realizează treptat, pe măsură ce se avansează în procesul de obținere a acestuia. Calitatea sistemelor de programe se creează în procesul de realizare şi se manifestă în procesul de utilizare a acestora. În industria de software, o importanță deosebită o are relația de dependență dintre specificul şi calitatea procesului generator, calitatea proiectului şi calitatea produsului.

Conform ISO 8402, calitatea este prezentă în fapt printr-un set de caracteristici ce pot fi împărțite în:

- **caracteristici economice**: se exprimă prin intermediul costurilor, al economiilor de resurse, precum şi prin creşterile de randament şi de productivitate;
- **caracteristici sociale şi psiho-senzoriale**: se manifestă prin punerea în valoare a elementelor creatoare, prin eliminarea rutinei şi a stereotipiei precum şi prin instruirea asistată a operatorilor;
- **caracteristici tehnice şi de utilizare**: sunt prezentate în literatura de specialitate, iar I.S.O. a elaborat un model care reprezintă esența standardului ISO 9126, [10].

Între caracteristicile de calitate există o mulțime de relații de subordonare, interdependență, ierarhizare, agregare sau decompoziție, iar complexitatea acestor relații face ca ansamblul caracteristicilor de calitate să alcătuiască un sistem. Caracteristicile de calitate constituie agregări ale unor atribute de calitate, ce sunt corespunzătoare unor proprietăți concrete, pe care trebuie să le posede sistemele de programe.

Managementul calității este componenta funcției generale de conducere, care determină şi pune în aplicare politica în domeniul calității şi deține responsabilitățile pentru toate activitățile care concură la îndeplinirea obiectivelor privind calitatea. Există patru funcții principale ale managementului calității:

- **planificarea calității**: cuprinde activitățile prin care se stabilesc obiectivele şi cerințele privind calitatea;
- **controlul calității**: include activitățile de monitorizare a desfăşurării proceselor şi de evaluare a rezultatelor referitoare la calitate, în raport cu obiectivele stabilite, în scopul eliminării deficiențelor şi/sau prevenirii apariției acestora;
- **asigurarea calității**: cuprinde activitățile preventive prin care se urmăreşte, în mod sistematic, corectitudinea şi eficiența activităților de planificare şi control, pentru a garanta obținerea rezultatelor la nivelul calitativ aşteptat;
- **îmbunătățirea calității**: cuprinde activitățile desfăşurate în fiecare etapă a ciclului de viață a produsului, în scopul obținerii unui nivel calitativ superior celui planificat, în condițiile desfăşurării corespunzătoare a activităților de planificare, control şi asigurare a calității.

2. Fiabilitatea software

Fiabilitatea unei componente sau unui sistem este o funcție de timp $F(t)$, definită prin probabilitatea ca, în condiții de mediu specificate, acestea să funcționeze adecvat în intervalul de timp $[0,t)$. Conform [4], fiabilitatea software este probabilitatea funcționării fără eşec a unui program, pentru o perioadă de timp specificată şi într-un mediu precizat. În

[10], fiabilitatea est definită ca fiind abilitatea unui sistem sau a unei componente, de a-şi îndeplini funcţiile, în anumite condiţii, pe o perioadă de timp specificată. O altă modalitate de exprimare a fiabilităţii este cea care o consideră ca timp mediu între defectări. Indicatorul este folosit şi pentru exprimarea fiabilităţii hardware.

Fiabilitatea software se exprimă şi ca intensitate a defectelor - număr de defecte în unitatea de timp. Relaţia dintre intensitatea defectelor şi fiabilitate depinde, în mare măsură, de modelul utilizat pentru evaluare. Exprimarea fiabilităţii prin intermediul intensităţii defectelor este o modalitate mai potrivită pentru acele sisteme software pentru care riscul de defectare în orice moment reprezintă motivul principal de îngrijorare.

Modelarea fiabilităţii oferă o reprezentare funcţională a unui sistem de programe, iar modelul cel mai bun furnizează un mecanism viabil pentru estimarea fiabilităţii. Modelele de fiabilitate software existente sunt grupate în funcţie de diferite sisteme de clasificare. Două din cele mai utilizate criterii sunt natura erorilor proceselor studiate şi etapa din ciclul de viaţă al unui sistem de programe în care se aplică modelul. Spre exemplificare, modelele folosite în timpul etapei de testare/validare sau în etapa operaţională sunt împărţite în următoarele patru categorii:

- modele de timpi între erori;
- modele de numărare a erorilor;
- modele de însămânţare a erorilor;
- modele ale intrărilor pe domeniul de bază.

Relativ la etapa din ciclul de viaţă al unui sistem de programe, în care se aplică modelul, apar următoarele categorii de modele:

- modele de predicţie: sunt aplicate în fazele preliminare din ciclul de realizare a sistemelor de programe, iar utilizarea lor permite determinarea nivelului fiabilităţii preliminate şi obţinerea de predicţii referitoare la comportamentul sistemelor de programe în timpul etapelor de testare şi exploatare;
- modele de estimare: sunt utilizate în etapele finale din ciclul de viaţă - testarea integrării la nivel de componentă, testarea la nivel de sistem, testare funcţională şi exploatare;
- modele de creştere a fiabilităţii: aceste modele realizează predicţii bazate pe numărul de erori observate pe parcursul testului, iar parametrii sunt estimaţi prin metoda verosimilităţii maxime sau a celor mai mici pătrate şi produc rezultate diferite pentru aceleaşi date de intrare;
- modele de analiză a efortului de testare: iau în discuţie problema efortului de testare variabil în timp şi propun o estimare a lui printr-o funcţie pe care o numesc rata de expunere; în timpul procesului de testare densitatea defectelor descreşte pe măsură ce sistemele de programe sunt testate tot mai mult, iar rata de expunere la defecte descreşte şi ea.

O categorie aparte de modele o reprezintă cele care combină arhitectura sistemelor de programe cu comportamentul erorilor. În [8], sunt prezentate trei abordări ale metodei utilizate pentru combinarea arhitecturii software cu comportamentul erorilor care au permis dezvoltarea a trei subcategorii de modele:

- modele bazate pe stare: folosesc graful fluxului de control pentru a reprezenta arhitectura sistemului;
- modele bazate pe căi: calculează fiabilitatea sistemelor de programe pe baza căilor de execuţie a programelor componente fie experimental prin testare, fie algoritmic;
- modele aditive: presupun că fiabilitatea unei componente este modelată printr-un proces Poisson neomogen, iar intensitatea de defectare a sistemului este exprimată ca sumă a intensităţilor de defectare ale componentelor.

O problemă importantă pe care trebuie să o aibă în vedere producătorii de software o reprezintă creşterea fiabilităţii sistemelor de programe. Aceasta se bazează pe analiza multilaterală a factorilor care influenţează fiabilitatea software în toate etapele prin care trece un sistem de programe: definire cerinţe, proiectare, codificare, testare, exploatare şi întreţinere. Strategia de abordare a problemei creşterii fiabilităţii urmăreşte evitarea greşelilor efectuate în procesul de elaborare a sistemelor de programe, în scopul prevenirii introducerii de erori. Măsurile ce trebuie luate pentru asigurarea creşterii fiabilităţii sistemelor de programe se împart în două categorii:

- măsuri cu caracter general: valabile cu ponderi diferenţiate în toate etapele de elaborare a unui sistem de programe;
- măsuri cu caracter specific: valabile pentru o anumită etapă din procesul de elaborare a unui sistem de programe.

În [7] se recomandă utilizarea următoarelor principii şi practici pentru creşterea fiabilităţii sistemelor de programe:

- implicarea utilizatorilor în elaborarea sistemelor de programe;
- conducerea, coordonarea şi controlul proiectului;
- evitarea defectelor;
- programarea N-versională;

- întinerirea programelor;
- verificarea formală;
- expunerea defectelor;
- utilizarea unui sistem de raportare a erorilor şi de generare a acţiunilor corective;
- implementarea algoritmilor aleatori;
- creşterea gradului de reutilizare a componentelor software (module şi programe);
- creşterea fiabilităţii sistemelor de programe prin utilizarea de componente cu fiabilitate ridicată şi care au fost certificate din punct de vedere al fiabilităţii;
- asigurarea unui nivel optim între complexitatea sistemelor de programe şi nivelul lor de fiabilitate;
- creşterea gradului de pregătire profesională a personalului implicat în procesul de dezvoltare software şi realizarea unei omogenizări corespunzătoare a echipei;
- stabilirea de sarcini concrete pentru cei care efectuează testarea componentelor sistemului şi a întregului sistem, precum şi delimitarea lor de sarcinile concrete de proiectare şi implementare.

Toate aceste măsuri au ca scop creşterea nivelului fiabilităţii sistemelor de programe şi, implicit, satisfacerea într-o măsură cât mai mare a cerinţelor beneficiarilor, cu implicaţii directe asupra costurilor de exploatare a sistemelor software.

3. Costul calităţii software

Costurile privind calitatea software cuprind un ansamblu de activităţi care generează cheltuieli grupate în următoarele categorii: de prevenire, de evaluare, de identificare şi remediere, pe parcursul dezvoltării, a defectelor care sunt costurile noncalităţii. În tabelul de mai jos sunt prezentate grupele de factori ai costurilor menţionate.

Tabelul 1. Gruparea costurilor privind calitatea produselor software

Costuri de prevenire a defectelor	- costuri determinate de implementarea, funcţionarea şi îmbunătăţirea sistemului calităţii; - costuri determinate de audituri ale calităţii, de verificarea de ansamblu şi de detaliu a calităţii proiectelor şi a documentaţiei intermediare; - costuri legate de constituirea fluxului de control şi dirijare a proceselor şi produselor în vederea asigurării calităţii; - costuri de evaluare a subcontractanţilor, costuri de remediere a unor programe, documentaţii, primite de la eventualii subcontractanţi; - costuri determinate de ridicarea calităţii personalului, de motivarea acestuia, de angajare a unor experţi din exterior sau de personal cu calificare superioară.
Costuri de evaluare	- costul unor procese suplimentare de verificare, de exemplu costul inspecţiilor pe codul sursă, ale auditării sistemelor de programe software intermediare şi finale; - timpul suplimentar, consumat pentru procesele de analiză a erorilor sistematice, în vederea stabilirii cauzelor.
Costuri de identificare şi remediere a defectelor, pe parcursul dezvoltării	- costuri legate de achiziţionarea, testarea, exploatarea produselor utilizate în procesul dezvoltării; - costuri suplimentare apărute în procesul dezvoltării, generate de identificarea şi remedierea unor erori la produsele intermediare, cum sunt erori în proiectare, în codificare, în elaborarea documentaţiilor, evidenţiate, la intrarea şi ieşirea fiecărui proces, atât de compartimentele de asigurare a calităţii, cât şi de personalul de dezvoltare software.
Costurile Remedierii defectelor	a. Costuri înregistrate la producător: - costuri suplimentare la finalul procesului de dezvoltare, determinate de remedierea unor erori apărute după testarea şi inspecţia finală a calităţii; - costuri suplimentare determinate de repetarea unor teste şi verificări; - costuri determinate de nerespectarea eventualelor obligaţii contractuale; - costuri privind stingerea reclamaţiilor făcute de beneficiari; - costuri suplimentare privind activităţile de service; - costuri suplimentare determinate de repregătirea produsului pentru livrare. b. Costuri la beneficiar: - costuri privind remedierea erorilor la produsele reclamate de beneficiari ca

Costurile Remedierii defectelor	necorespunzătoare sau refuzate, în cazul în care recepția se face la aceştia ; - costuri determinate de înlocuirea produselor menționate la aliniatul anterior; - costul privind testările, controalele, expertizele la produsul ajuns la destinație şi reclamat la beneficiar; - costuri privind activitatea de mentenanță în perioada de garanție; - costuri privind stingerea reclamațiilor pentru calitate necorespunzătoare făcute de beneficiari, în cazul în care recepția are loc la beneficiar; - costuri determinate de daunele provocate activității beneficiarului şi reclamate de acesta, cauzate de calitatea necorespunzătoare a produsului software livrat.

Managementul costului începe în momentul conceperii unui nou produs. În general, între 70 şi 80 % din costul unui sistem de programe este determinat de deciziile luate în faza de concepție. În cazul produselor software, aceasta este etapa stabilirii şi analizării cerințelor. Din acest considerent, reducerea costurilor este necesar să se concentreze pe faza de concepție, fără a neglija, însă, celelalte etape ale ciclului de viață.

Determinarea costurilor noncalității furnizează managementului informații referitoare la eficiența proceselor. Acestea reprezintă un indicator necesar pentru stabilirea măsurilor de îmbunătățire a calității şi de inițiere a unor activități corespunzătoare de prevenire şi corecție.

4. Costul fiabilității software

Estimarea costurilor şi controlul proceselor trebuie redefinite, în conformitate cu caracteristicile noilor tehnologii de dezvoltare software şi cu efectele acestora asupra procesului de dezvoltare a sistemelor de programe. Datorită nivelului înalt de standardizare oferit de tehnologia orientării pe obiecte, sistemele de programe orientate pe obiecte permit estimarea mai precisă a costurilor dezvoltării decât sistemele de programe procedurale.

În [1] este prezentat un model de estimare a costului unui produs software, dezvoltat pe baza tehnologiei orientării pe obiecte. Acest model este destinat să sprijine firmele producătoare de software în procesul de evaluare a prețului unui anumit sistem de programe. Modelul este alcătuit din două module interactive diferite, conform figurii 1. Primul modul are ca scop estimarea costului direct al procesului de dezvoltare şi are la bază un model de estimare a costului pentru sistemele de programe procedurale. Al doilea modul se bazează pe costul activității de bază şi este menit să aloce produsului costurile indirecte.

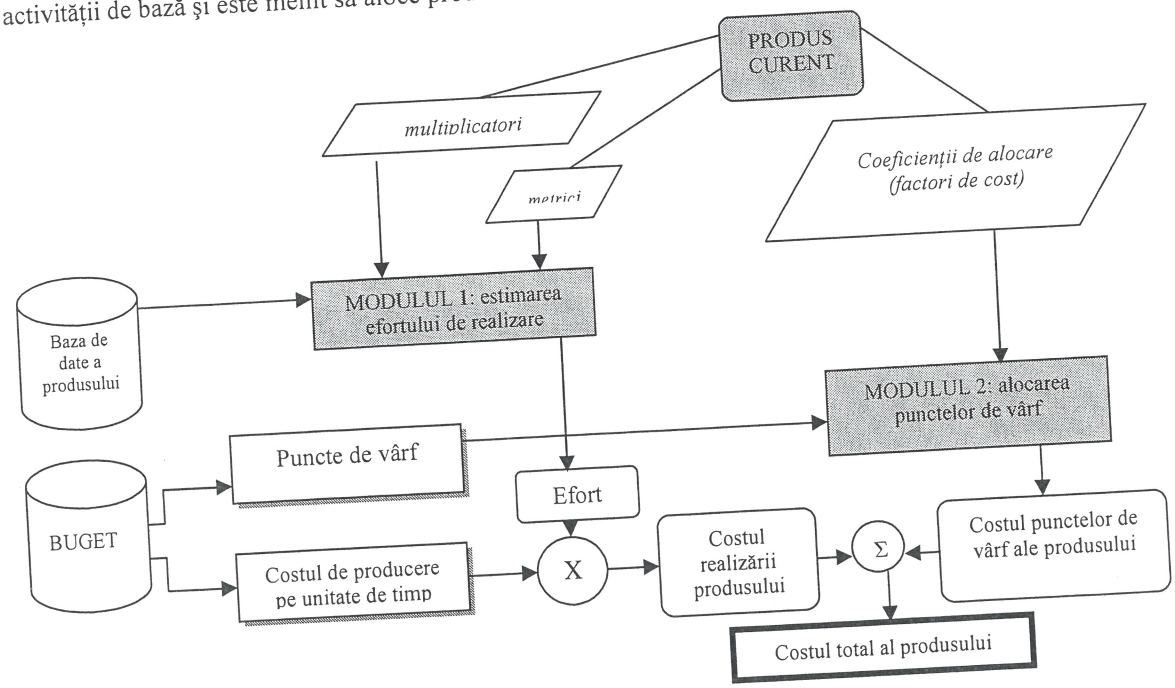

Figura 1. Schema modelului de estimare a costului sistemelor de programe

Modulul de estimare a efortului de dezvoltare - costul de elaborare al unui produs C_d este calculat prin:

$$C_d = C_u \cdot effort \quad (1)$$

unde: C_u - este costul de elaborare per unitatea de timp, exprimat în mod tipic prin cost/lună, care se referă la resursele de muncă umană implicate în acest proces; *effort* - se referă la timpul de care are nevoie un singur factor, implicat în procesul de dezvoltare, pentru a-şi efectua sarcina ce i-a fost stabilită.

Acest modul este concentrat pe estimarea efortului; efortul este dependent de diferite caracteristici ale produsului (metrici) $m_j, j = 1,...,n$

$$effort = f(m_1, m_2, ..., m_n) \quad (2)$$

Această ipoteză este mai complexă, datorită numărului mare de variabile independente, care trebuie luate în calcul, în funcție de metoda de estimare aleasă. O ipoteză alternativă furnizează o procedură mai simplă. Aceasta constă în estimarea unei valori nominale a efortului, în funcție de o singură metrică, şi care măsoară dimensiunea produsului software:

$$effort\ nominal = f(m) \quad (3)$$

iar ulterior are loc corectarea acestei valori în funcție de cei *N* factori de corecție $K\ (k_1, k_2, ..., k_N)$ şi se obține:

$$effort = K(k_1, k_2, ..., k_N) \times f(m) \quad (4)$$

În cadrul modelului, este luată în considerare următoarea funcție pentru estimarea efortului:

$$effort = \prod_{j=1}^{N} k_j \cdot a \cdot m^b \quad (5)$$

unde: *m* reprezintă numărul total al metodelor publice folosite în cadrul sistemului de programe, funcția de corecție este un produs de $k_j (j = 1,..., N)$, care depinde de caracteristicile produsului şi ale procesului, iar *a* şi *b* sunt parametrii dependenți de informațiile stocate în baza de date a produselor.

Modelul necesită determinarea următoarelor cantități pentru un sistem de programe specific:

- funcția *f* a efortului nominal;
- metrica *m* care oferă o dimensiune a sistemului de programe care va fi dezvoltat;
- parametrii *a* şi *b*;
- funcția de corecție $K(k_j), j = 1,..., N$.

Funcția efortului nominal şi metrica *m* au fost alese ca:

$$effort\ nominal = f(m) = a \cdot m^b \quad (6)$$

În această funcție, parametrul *a* indică unitățile de măsură ale productivității, deoarece acesta reprezintă timpul de lucru necesar per unitatea de dimensiune a software. Funcția de corecție este exprimată ca un produs al anumitor factori de corecție:

$$K(k_1, k_2, \ldots, k_N) = k_1 \cdot k_2 \cdot \ldots \cdot k_N \quad (7)$$

Pentru sistemele de programe, dezvoltate conform tehnologiei orientării pe obiecte, sunt luați în considerare factorii de corecție prezentați în tabelul 2.

Tabelul 2. Factorii de multiplicare pentru funcția de corecție a efortului nominal

	Factor	Descriere	Evaluare
K1	DR	Gradul de refolosire	Empirică
K2	DFR	Dezvoltare pentru reutilizare	Calcul
K3	CPLIF	Complexitatea interfeței om-maşină	Calcul
K4	AEXP	Experiență în domeniul problemei	Empirică
K5	ACAP	Capabilitatea analistului	Empirică
K6	DPCAP	Capabilitatea proiectantului şi programatorului	Empirică

K7	CPLX	Complexitatea produsului	Calcul
K8	CPLXE	Complexitatea mediului operațional	Empirică

Factorii de la K4 la K7 au fost aleşi dintre cei cunoscuți din literatura de specialitate, [1]. Primii trei factori sunt specifici tehnologiei orientării pe obiecte. Față de factorii cunoscuți din literatura de specialitate, în lucrare s-a adăugat un factor nou, K8, care exprimă complexitatea mediului operațional şi este evaluat în mod empiric.

5. Concluzii

În structura generală a costului unui sistem de programe, costul fiabilității are o pondere însemnată. O serie de studii arată că, în prezent, costurile corectării nonfiabilității şi cele necesare pentru prevenirea şi evaluarea ei reprezintă, în medie, 25% din cifra de afaceri a unei companii producătoare de software şi circa 5-15% din costurile de producție. Cunoaşterea categoriilor de costuri menționate anterior şi a relațiilor dintre ele constituie instrumente importante ale managementului pentru reducerea costurilor, concomitent cu menținerea unui nivel al calității ridicat. Determinarea costurilor nonfiabilității furnizează managementului informații referitoare la eficiența proceselor. Acestea reprezintă un indicator necesar pentru stabilirea măsurilor de îmbunătățire a fiabilității şi de inițiere a unor activități corespunzătoare de prevenire şi corecție. Proiectarea orientată spre costuri reprezintă un element de management strategic, care utilizează metodologiile de realizare a unui sistem de programe prin tratarea costului țintă ca un parametru independent al proiectării, care trebuie realizat în timpul dezvoltării produsului.

Bibliografie

1. **CARMEL, E., S. BECKER**: A Process Model for Packaged Software Development. În: IEEE Transactions on Engineering Management, 2000, Vol. 42, No. 1, pp. 50-61.
2. **CRISTESCU, M.**: Estimarea costurilor în dezvoltarea sistemelor de programe orientate obiect. Conferința științifică internațională "Identitatea şi universalitatea economiilor în tranziție în debutul mileniului trei", Ediția a X-a jubiliară, Sibiu, 8-12 mai 2003.
3. **IVAN, I., P. POCATILU, L. TEODORESCU**: Creşterea calității software prin testare. În: Qmedia, nr. 5, 2000, pp. 9-14;
4. **MUSA, J.D., A. IANNINO, K. OKUMOTO**: Software Reliability: Measurement, Prediction, Application, Professional Edition: Software Engineering Series, McGraw-Hill, New York, NY., 1990.
5. **STOICA, M.**: Evaluarea fenomenelor socio-economice. În: Studii şi Cercetări de Calcul Economic şi Cibernetică Economică, vol. 37, nr. 4, 2003.
6. **STOICA, M.**: Mulțimi subtile în economie. În: Studii şi Cercetări de Calcul Economic şi Cibernetică Economică, vol. 36, nr. 4, 2002.
7. **MIHALACHE, A.**: Când calculatoarele greşesc. Fiabilitatea sistemelor de programe(software), Editura Didactică şi Pedagogică, Bucureşti, 1995.
8. **YACOUB, S., B. CUKIC, H. AMMAR**: Scenario-based Reliability Analysis of Component Based Software. În: Proc. of the 10-th International Symposium on Software Reliability Engineering, ISSRE'99, 1999, pp. 22-31.
9. **ISO 8402** - International Standard - Quality Vocabular, 1986, Geneve, Switzerland.
10. **ISO/IEC 9126** International Standard - Information Technology - Software Product Evaluation - Quality Characteristics and Guidelines for their Use, 1991, Geneve, Switzerland.