CERTIFICAREA PRODUSELOR PROGRAM PRIN AMPRENTE

Ion Ivan
Cosmin Apostol

Academia de Studii Economice

Rezumat: Producţia software presupune elaborare de proceduri şi reutilizare de componente. Procesele de dezvoltare software original necesită cheltuieli deosebit de mari. Lucrarea îşi propune identificarea produselor - program originale, care îndeplinesc o serie de caracteristici de calitate. Se defineşte amprenta software. Sunt prezentate etapele certificării folosind amprente.

Cuvinte cheie: certificare, calitate, amprentă, produs – program.

1. Proiecte de dezvoltare software

Pentru elaborarea de aplicaţii informatice complexe, se lansează programe orientate pe probleme şi se identifică fonduri care asigură finanţarea.

Pentru implementarea unui astfel de program se procedează la traversarea următoarelor etape:

- definirea obiectivului programului;
- stabilirea criteriilor de eligibilitate;
- elaborarea documentaţiei pentru ofertant şi pentru evaluatori;
- prezentarea formularisticii necesare întocmirii unei oferte;
- eşalonarea termenelor privind competiţia, încheierea de contracte şi eliberarea de tranşe;
- definirea modalităţilor de auditare;
- stabilirea criteriilor de control pe parcurs a implementării;
- monitorizarea;
- recepţionarea produsului.

Proiectele de dezvoltare software sunt, de regulă, de dimensiuni mari şi antrenează surse financiare de ordinul sutelor de mii de euro sau chiar de ordinul milioanelor de euro.

Se dezvoltă produse - program originale prin:

- natura problemei de realizat;
- gradul de generalitate urmărit;
- interdependenţa om-calculator;
- tehnologiile utilizate;
- generaţia de instrumente folosite;
- nivelul de calitate urmărit;
- stadiul de dezvoltare al societăţii informaţionale.

Un produs software original se deosebeşte radical de celelalte produse existente pe piaţă, chiar dacă se referă la soluţionarea aceleiaşi probleme prin:

- uşurinţa cu care se implementează;
- arhitectura hardware necesară;
- uşurinţa cu care este lansată în execuţie, de către utilizatori, fiecare opţiune;
- limbajele utilizate pentru elaborare;
- gradul de generalitate;
- diversitatea utilizatorilor;
- naturaleţea dialogului om-maşină;
- noile funcţii introduse, în comparaţie cu produsele existente;
- flexibilitatea pe care o oferă în obţinerea de rezultate intermediare;

- crearea unei diversităţi de date de intrare;
- volumul valorilor implicite considerate;
- facilităţile în introducerea de proceduri proprii de prelucrare;
- gradul de vecinătate asigurat;
- diversitatea modalităţilor de conectare a posturilor la un server central.

Originalitatea unui produs software apare la nivelul diferenţelor care se evidenţiază din aproape în aproape.

Pe baza unor specificaţii diferite se obţin produse software diferite.

Pe baza unui set de specificaţii diferit, în vederea atingerii unui obiectiv bine definit, datele de intrare şi rezultatele finale sunt aceleaşi indiferent de produsul - program elaborat.

Diferenţele apar în zona:

- structurii modulelor ca număr de niveluri şi număr de componente pe nivel;
- lungimea modulelor;
- complexitatea modulelor;
- structurile de date utilizate.

Pentru a evidenţia originalitatea unui produs software, realizat în cadrul unui program de dezvoltare aplicaţii informatice, se impune utilizarea atât a analizei calitative, cât şi a analizei cantitative.

2. Amprenta produsului - program

Se consideră o mulţime de produse - program formată din elementele $P_1, P_2, ..., P_n$.

Fiecare program P_i este stocat într-un fişier, F_i, i=1, 2, ..., n.

Diferenţele de ordin cantitativ dintre programe apar prin efectuarea unor măsurători folosind o serie de indicatori aplicaţi fişierelor normalizate.

Operaţia de normalizare constă în prelucrarea primară a unui fişier iniţial F_i, prin implementarea la nivel de linie a unei singure instrucţiuni şi obţinerea la ieşire a fişierului normalizat G_i.

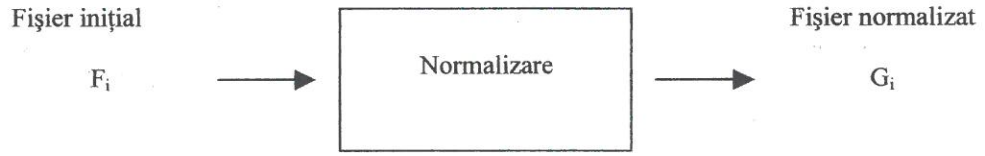

Figura 1. Normalizarea fişierelor

În continuare, sunt luate în considerare programe scrise în limbajul C, trecerea către oricare alt limbaj de programare fiind o problemă de rutină.

În cele ce urmează, sunt definiţi indicatorii ce servesc la analiza fişierelor.

Lungimea L_i a programului normalizat G_i este dată de numărul de caractere din care este format textul fişierului normalizat. De exemplu, programul G_1:

```
int a;
a = 0;
a = a + 1;
```

are lungimea $L_1 = 25$ caractere (se includ şi spaţiile).

Numărul de instrucţiuni NI_i ale programului P_i este dat numărul de linii sursă din fişierul programului normalizat G_i.

Pentru programul de mai sus, numărul de instrucţiuni NI_1 este egal cu 3.

Lungimea medie a instrucțiunilor normalizate LM_i este dată de media aritmetică a lungimilor, numărul de caractere, al instrucțiunilor din fișierul normalizat G_i.

Pentru programul de mai sus lungimea medie a instrucțiunilor normalizate LM_1 este este egală cu 8.

După normalizarea fișierului inițial, obținem un fișier normalizat. Pasul următor este sortarea descrescătoare a instrucțiunilor din acest fișier, sortare care este realizată după lungimea instrucțiunilor.

După operația de sortare, obținem un fișier sortat S_i.

La acest pas, se calculează numărul de interschimbări între instrucțiuni NIS_i și rangul normat al sortării RN_i.

Un indicator important îl constituie lungimea vocabularului. *Lungimea vocabularului* LV_i se definește ca fiind numărul total de cuvinte distincte, ce intră în componența fișierului G_i.

Se face diferență între cuvintele cheie ale limbajului de programare C și cuvintele utilizatorului. Pe baza acestei diferențe, se calculează alți doi indicatori și anume:

- complexitatea H a programului, rezultată ca

  $H = n_1 * \log_2 n_1 + n_2 * \log_2 n_2$,

  unde

  n_1 – numărul operanzilor din vocabularul programului,

  n_2 – numărul operatorilor din vocabularul programului.

- complexitatea normată

$$\tilde{H} = \frac{H}{(n_1 + n_2) \cdot \log_2 (n_1 + n_2)} \qquad (1)$$

Amprenta A_i a programului P_i normalizat în programul G_i este definită astfel:

$A_i = < a_{i1}, a_{i2}, a_{i3}, a_{i4}, a_{i5}, a_{i6}, a_{i7}, a_{i8} >$, unde:

- a_{i1} reprezintă lungimea L_i a programului P_i;
- a_{i2} reprezintă numărul de instrucțiuni NI_i ale programului P_i;
- a_{i3} reprezintă lungimea medie a instrucțiunilor normalizate (LM_i);
- a_{i4} reprezintă numărul de interschimbări între instrucțiuni în cadrul sortării (NIS_i);
- a_{i5} reprezintă rangul normat al sortării (RN_i);
- a_{i6} reprezintă lungimea vocabularului programului P_i;
- a_{i7} reprezintă complexitatea H a programului P_i;
- a_{i8} reprezintă complexitatea normată $\tilde{H}$ a programului P_i.

3. Certificarea calității software

În conformitate cu definiția adoptată de Organizația Europeană pentru calitate (EOQ), certificarea reprezintă procedura și activitățile efectuate de un organism de certificare privind analizarea, verificarea și confirmarea scrisă a calității produselor și/sau serviciilor/proceselor în concordanță cu specificațiile (reglementările specifice).

Conform ISO-8402 "Managementul și asigurarea calitații –Vocabular", calitatea este definită ca fiind ansamblul caracteristicilor unei entități, care îi conferă aptitudinea de a satisface nevoile exprimate sau implicite.

Certificarea – procedură prin intermediul căreia un organism neutru (terță parte) furnizează asigurarea scrisă că un anumit produs, serviciu sau sistemul calității unei organizații satisface cerințele specificate într-un referențial (standard, specificație tehnică etc.).

Certificarea asigură:

- protejarea elaboratorilor față de concurenți;
- protejarea utilizatorilor față de produsele de slabă calitate.

Certificarea implică mai multe avantaje pentru elaboratori (autori) şi pentru utilizatori. Din punct de vedere al utilizatorului, putem spune că acesta este avantajat deoarece foloseşte produse având un nivel de calitate demonstrat (prin dovezi) şi garantat (prin documente oficiale). Utilizatorii fac, de asemenea, economie de resurse şi obţin satisfacţie în utilizare.

Din punct de vedere al elaboratorilor se obţin următoarele avantaje:

- prestigiu pe piaţă;
- satisfacţia clientelei;
- credibilitate şi încredere.

O dată cu obţinerea certificării, se atestă faptul că acel produs - program respectă cerinţele utilizatorilor, deci, clienţii sunt satisfăcuţi de calitatea produsului - program. Dacă clienţii sunt satisfăcuţi atunci vor cumpăra acel produs - program, ceea ce măreşte prestigiul pe piaţă al produsului respectiv. Certificarea produsului - program sporeşte credibilitatea acestuia si încrederea în el. Dacă o anumită caracteristică a produsului - program este certificată înseamnă că acea caracteristică a fost testată de specialişti, deci, probabilitatea ca acea caracteristică să nu ruleze la parametrii standardizaţi tinde către 0.

Certificările se împart în următoarele categorii:

- certificare de produs;
- certificare de sistem (organizaţie);
- certificare de persoane.

Certificarea unui produs - program atestă faptul că acel produs respectă anumite standarde şi va rula la parametrii specificaţi. *Certificarea de sistem (organizaţie)* atestă faptul că acea organizaţie poate să producă produse - program de calitate deoarece respectă standardele din domeniu. *Certificarea de persoane* atestă faptul că acele persoane sunt capabile să utilizeze un anumit produs - program sau să scrie cod program de calitate.

Concepţia modernă despre calitate o leagă pe aceasta de alte două elemente: calitatea concepţiei (proiectului) şi calitatea fabricaţiei. Conform standardelor ISO, calitatea fabricaţiei reprezintă gradul de conformitate a produsului cu documentaţia tehnică. Calitatea proiectului exprimă măsura în care proiectul produsului asigură satisfacerea cerinţelor beneficiarilor şi posibilitatea de folosire, la fabricaţia produsului respectiv, a unor procedee tehnologice raţionale şi optime din punct de vedere economic.

Importanţa calităţii produselor software constă în mai multe aspecte: erorile din programele de aplicaţie pot fi fatale în anumite domenii unde vieţile oamenilor depind de acestea (controlul traficului aerian sau feroviar, călătoriile în spaţiul cosmic, sistemele de menţinere a vieţii în condiţii vitrege - sub apă, la mare înălţime, la temperaturi sau presiuni improprii vieţii umane); aceste erori pot provoca pierderi umane, financiare, materiale şi tot felul de alte tipuri de insatisfacţii sau pierderi.

Calitatea software reprezintă totalitatea însuşirilor tehnice, economice şi sociale ale produselor software. Ea reprezintă ansamblul însuşirilor ce exprimă gradul în care acestea satisfac nevoia utilizatorilor, în funcţie de parametrii tehnico-economici, de gradul de utilitate şi de eficienţa economică în exploatare.

Gradul de utilitate al produselor - program cuprinde:

a) *calitatea de concepţie şi proiectare a produsului - program* - măsura în care proiectul produsului - program asigură satisfacerea cerinţelor utilizatorilor;

b) *calitatea de execuţie a produsului - program* - măsura în care procesul de elaborare se desfăşoară conform fluxurilor stabilite, cu utilizarea resurselor adecvate;

c) *calitatea de conformitate a produsului - program* - gradul de concordanţă dintre însuşirile reale ale produsului program şi cele prezentate în documentaţia finală;

d) *capacitatea de utilizare a produsului - program* - comportamentul produsului - program în rezolvarea curentă a problemelor aparţinând clasei pentru care a fost elaborat;

e) *capacitatea de mentenanţă a produsului - program* - măsura în care pot fi eliminate anomaliile ce apar în timpul execuţiei sau pot fi puse de acord noi cerinţe de prelucrare cu efortul pentru implementare.

Calitatea produselor software prezintă o serie de particularităţi datorate specificului acestora, diferenţelor care există între produse, în general, şi produsele software, şi anume:

- *producția de software este o producție de unicate:* aceasta face ca activitatea de stabilire a standardelor din acest domeniu să fie mult mai dificilă; pot fi standardizate procesele, metodele şi tehnicile de elaborare a software-ului astfel încât, în final, să se obțină produse software de un anumit nivel al calității;
- *produsele software sunt reproductibile* cu costuri aproape nule, ceea ce înseamnă că, o dată realizat un produs, el este disponibil într-un număr nelimitat de exemplare; calitatea cópiilor este aceeaşi cu cea a originalului, deci o calitate necorespunzătoare se poate propaga cu efecte nefavorabile, antrenând costuri de remediere foarte mari şi afectând imaginea firmei producătoare;
- *produsele software nu sunt supuse uzurii fizice*, ele fiind afectate de uzura morală, atunci când nu mai corespund noilor cerințe sau când apar produse software mai performante care realizează aceleaşi funcțiuni; un produs software depăşit este înlocuit cu altul nou sau poate fi adaptat pentru a satisface noile cerințe formulate de utilizator; decizia se ia în raport de costurile fiecăreia dintre variante, iar dacă produsul software este uşor modificabil, costurile aferente adaptării sale vor fi mult mai scăzute;
- *pentru a putea utiliza produsele software este nevoie de un procesor* care să le interpreteze; sistemele de calcul şi operare sunt supuse, de asemenea, uzurii morale, punându-se, şi în acest caz, problema transferării produsului software într-un alt mediu; un produs software, care posedă o portabilitate ridicată, poate fi modificat cu uşurință, cu costuri mici;
- *comportamentul instrucțiunilor este egal în timp*, nu se deteriorează;
- *erorile* sunt provocate de folosirea sau combinarea incorectă a instrucțiunilor sau a altor componente elementare, şi nu de componentele ca atare;
- *interacțiunile dintre componentele* unui program (secvențe, proceduri, module) sunt complexe, mai ales dacă acestea rulează în cadrul unor aplicații complexe;
- erorile există deja în program, ele sunt eliminate cu timpul, prin depanare, deci, *programul nu-şi pierde calitățile prin trecerea timpului, ci şi le îmbunătățeşte; se demonstrează că, oricâte corecții se efectuează, programul rămâne în continuare cu erori;*
- eliminarea unei erori nu dă siguranța că a diminuat numărul total de erori cu o unitate;
- noncalitatea programelor poate fi atribuită în întregime greşelilor umane, de proiectare, concepție, programare, documentare.

Utilizatorul este cel care certifică calitatea unui produs - program, având drept criteriu satisfacerea cerințelor sale. Producătorii de software trebuie să obțină, cel puțin, acel nivel al calității care să permită satisfacerea aşteptărilor utilizatorului. Un nivel mai scăzut, duce la alegerea produselor software concurente, care la acelaşi preț oferă nivelul necesar al calității. Pe de altă parte, un nivel mult mai ridicat al calității antrenează costuri foarte mari, care se reflectă în prețul produsului. În acest caz, utilizatorul alege acele produse concurente care la un preț mai scăzut, oferă un nivel suficient al calității.

Calitatea unei aplicații rezidă în calitatea ingineriei specificațiilor, a analizei şi a proiectării, a testării, a documentației, a conducerii proiectelor etc. Abilitatea de a colecta specificații cât mai complete de la clienți, de a-l determina pe client să vadă funcționalitatea viitorului produs, se va reflecta în satisfacția ulterioară a clientului față de produs. Astăzi, analiza şi proiectarea sunt considerate fazele cheie ale unui proiect şi ele ar trebui să acopere 40% din durata totală a proiectului. Din această cauză, s-au dezvoltat metodologii de analiză RAD (Rapid Application Development) sau OO (Object Oriented) şi instrumente CASE (Computer Aided Software Development). Testarea se orientează spre instrumente automate de test. Este planificată, este compusă din scenarii de test dezvoltate pe baza analizei şi se încearcă restrângerea pe cât posibil sau chiar eliminarea testării manuale.

4. Software pentru evaluarea indicatorilor asociați amprentei

Principalele funcții ale aplicației pentru determinarea amprentei unui produs program P sunt definite prin următoarele prototipuri:

- int normalizare(char *fisier_initial, char *fisier_normalizat, int marime[])
- void sortare(char *fisier_normalizat, char *fisier_sortat, int nr_instr, int marime[], int &interschimbari, float &rang)
- void selectie(char *fisier_sortat, char *fisier_esantion, int nr_instr, int marime[])

- void vocabular(char string[], char *cuvinte_distincte[], int &nr_cuvinte_cheie, int &nr_cuvinte_distincte, int total_cuvinte)

Funcţia "**normalizare**" primeşte numele fişierului iniţial ce se va normaliza şi numele fişierului normalizat. Funcţia va întoarce numărul de instructiuni din fişierul normalizat. Prin intermediul vectorului **marime** se vor putea memora şi lungimea fiecărei instrucţiuni din fişierul normalizat.

Procedura "**sortare**" primeşte numele fişierului normalizat şi numele fişierului sortat, precum şi numărul de instrucţiuni din fişierul normalizat. Se vor sorta instrucţiunile din fişierul normalizat după lungimea lor. Rezultatul va fi stocat în fişierul sortat. La acest pas se vor calcula şi cei doi indicatori din amprenta produsului – program, asociaţi sortării: numărul de interschimbări între instrucţiuni şi rangul normat al sortării.

Procedura "**selectie**" primeşte numele fişierului sortat şi numele fişierului ce va conţine eşantionul reprezentativ de instrucţiuni, precum şi numărul de instrucţiuni din fişierul sortat şi mărimea fiecărei instrucţiuni. Se va selecţiona un eşantion reprezentativ de instrucţiuni şi se vor stoca aceste instrucţiuni în fişierul eşantion.

Procedura "**vocabular**" calculează numărul cuvintelor dintr-un şir de caractere, care se primeşte ca parametru de intrare. De asemenea, stabileşte câte din aceste cuvinte sunt cuvinte cheie, rezervate, ale limbajului de programare folosit şi câte sunt cuvinte care aparţin utilizatorului.

5. Validarea amprentei produsului program

Pentru procesul de validare a amprentei produsului program se iau în considerare două cazuri distincte.

Se presupune o mulţime de programe P_1, P_2, ..., P_N . Se evaluează amprentele acestor programe cu produsul SEAP (**S**oftware **E**valuare **A**mprenta **P**rogram). Toate programele sunt ortogonale în urma analizei calitative.

Se calculează indicatorii şi se construieşte matricea diferenţelor dintre indicatori.

$$d_{ij} = f(A_i, A_j) \qquad (2)$$

$d_{ij} = 0$, când cele două amprente sunt identice;

$d_{ij} = 1$, când cele două amprente sunt absolut diferite.

Dacă din cele N programe peste 78% sunt cu amprente diferite, rezultă că amprenta este reprezentativă în raport cu programul original.

Se presupune o mulţime de programe P_1, P_2, ..., P_n, foarte asemănătoare în urma analizei calitative.

Se calculează d_{ij} şi, dacă rezultă că, din cele N programe peste 78% sunt neoriginale, înseamnă că amprenta este reprezentativă şi pentru programele neoriginale.

Dacă amprenta este reprezentativă pentru cele două mulţimi simultan rezultă că aceasta este validată şi se utilizează pentru analiza cantitativă a produsului software.

Folosind produsul - program SEAP în pentru analiza amprentelor unor secvenţe de cod foarte asemănătoare din punct de vedere semantic şi din punct de vedere sintactic, s-au obţinut următoarele rezultate:

Nr.secvenţă i	L_i	NI_i	LM_i	NIS_i	RN_i	LV_i	H_i	$\tilde{H}_i$
1	81	8	10	10	1.000	16	52.861	0.826
2	67	7	9	7	0.821	15	52.861	0.826
3	50	5	10	4	0.800	12	35.219	0.819
4	79	9	9	20	0.667	16	58.058	0.836
5	79	9	9	20	0.667	15	63.358	0.844
6	77	8	9	11	0.938	14	47.774	0.815
7	78	8	9	11	0.938	14	47.774	0.815
8	79	8	9	11	0.938	14	47.774	0.815
9	79	8	9	11	0.938	14	47.774	0.815
10	63	6	10	7	0.938	16	52.861	0.826

unde:

- L_i reprezintă lungimea programului P_i;
- NI_i reprezintă numărul de instrucţiuni ale programului P_i;

- LM_i reprezintă lungimea medie a instrucțiunilor normalizate LM_i ;
- NIS_i reprezintă numărul de interschimbări între instrucțiuni în cadrul sortării;
- RN_i reprezintă rangul normat al sortării;
- LV_i reprezintă lungimea vocabularului programului P_i;
- H_i reprezintă complexitatea H a programului;
- $\widetilde{H}_i$ reprezintă complexitatea normată a programului.

6. Concluzii

În cazul în care programele au dimensiuni mari se impune lucrul cu eşantioane reprezentative de instrucțiuni.

De asemenea, amprentele programelor P_1, P_2, ... sunt stocate într-o bază de date şi, la includerea unui nou program, se verifică dacă este sau nu clonă.

Bibliografie

1. **IVAN, I., L. TEODORESCU**: Managementul calității software, Editura Inforec, Bucureşti, 2001.
2. **SOARE, I., AL. D. COLCERU**: Certificarea calității, Editura Tribuna Economică, Bucureşti, 1996.
3. **BRAKHAHN, W., U. VOGT**: ISO 9000 pentru servicii, Editura Tehnică, Bucureşti, 1998.
4. **MOȚOIU, R.**: Ghid pentru managementul calității şi calitatea managementului anilor 2000, Editura FiaTest, Bucureşti, 2000.
5. **FRENȚIU, M., B. PÂRV**: Elaborarea programelor. Metode şi tehnici moderne, Editura PROMEDIA, Cluj-Napoca, 1994.
6. **McCALL, J.**: Factors in Software Quality, Editura General Electric Co., 1977.
7. **MEYER, B.**: Object Oriented Software Construction, Editura Prentice Hall, Englewood Cliffs, 1988.
8. **COOPER, A.**: Implementing a Same Process for Application Development http://www.newarchitectmag.com/archives/2002/08/