Cerin\e ale integrabilit`\ii pentru software economic

George DINU
BANCPOST S.A.

Privite ca piesele unui uria] puzzle ce modeleaza realitatea economica, aplica\iile economice [ncep s` nu mai reprezinte numai r`spunsuri punctuale cerin\elor, ci si s` lege [ntre ele diversele componente ]i, mai mult, s` ajute pe cei ce se [ncumet` s` conduc` jocul [n a cuprinde ]i a st`pâni complexitatea [ntregului tablou.
Iat` c` cerin\ele pentru noile aplica\ii economice devin din ce [n ce mai exigente]i realizatorii de soft r`spund acestor cerin\e [n m`sura [n care pot s` se organizeze din ce [n ce mai bine, ob\inând o productivitate mai buna (dezvoltarea de sofware a devenit o activitate industriala, de altfel).
***Cuvinte cheie**: software, aplica\ie economic`, reutilizare.*

1. Tendin\e [n dezvoltarea aplica\iilor economice

Crescând numarul firmelor produc`toare de software, a crescut ]i concuren\a ]i, ca urmare, calitatea produselor oferite. Mana-gerul unei activita\i economice poate alege acum [ntre a cere echipei proprii de infor-maticieni s` realizeze o aplica\ie sau s` achizi\ioneze de pe pia\` un software de firma.

{n contextul cre]terii calit`\ii produselor oferite pe pia\`]i a diversific`rii ofertei, folosirea software de firm` devine una din op\iunile cele mai frecvente ale celor ce decid provenien\a aplica\iilor economice.

Un aspect nou este cel al specializ`rii firmelor [n a realiza aplica\ii ce modeleaza un anumit segment al activit`\ii econo-mice. Superspecializarea astfel ap`rut` are ca o consecin\a o cre]tere a calit`\ii pro-dusului, firma ofertant` devenind astfel un lider de pia\`.

O schimbare de con\inut, ce are drept consecin\e modificari de adresabilitate este trecerea de la activitatea "monopost" la cea cu mai mul\i utilizatori. Cre]terea puterii de calcul, sc`derea pre\ului la componente hardware, integrabilitatea sistemelor, [mbun`t`\irile aduse lucrului [n echipa au dus la o cre]tere a num`rului celor care [ntegreaza componentele individuale de calcul in re\ele complexe, dar]i [n aplica\ii ce dau posibilitatea utilizatorilor de a lucra [n echipe, pe domenii.

De]i tendin\ele amintite pân` acum privesc mai mult utilizatorii de soft, iat` c` ]i pentru realizatorii de aplica\ii putem men\iona câteva direc\ii noi de ac\iune, care izvor`sc din "goana" acestora ([n sen-sul bun al cuvântului) dupa rezultate eco-nomice cât mai bune [n contextul cre]terii calit`\ii produselor oferite pe pia\`.

Folosirea mediilor de dezvoltare, cu facili-ta\ile oferite de acestea pentru gestionarea proiectelor complexe, a [nsemnat un salt important pentru cei ce doreau s` realizeze aplica\ii mari, [ntr-un timp cât mai scurt.

O alt` cale pentru echipele de dezvoltare [n concuren\a cu timpul ]i cu restric\iile manageriale, inclusiv privitoare la costuri, a fost integrarea componentelor aflate [n uz, ceea ce repezinta, [n cele din urm`, o buna cale de reducere a costurilor]i de cre]tere a calit`\ii, odat` cu reducerea timpului de realizare a produsului finit.

2. Reutilizarea componentelor software

{n realizarea aplica\iilor din ce [n ce mai complexe, pentru a m`ri productivitatea [n condi\iile de pia\` descrise mai sus, realizatorii au recurs, printre altele, la reutilizarea componentelor software.

Conform unei idei pe cât de r`spândit` [n literatura de specialitate, pe atât de adevarat`, [n loc s` scrii cod mai rapid, este mai bine s` scrii mai pu\in, folosind ceea ce s-a realizat ]i testat deja. Aceasta [nseamna c` reutilizarea reprezint` o sursa ieftin` de software de buna calitate.

Toate cele de mai sus sunt valabile [n condi\iile [n care [ntregul proces de dezvoltare de software este condus având in minte dualitatea *dezvoltarea **cu** reu-tilizare*]i *dezvoltarea **pentru** reutilizare*, cu alte cuvinte, dezvoltarea unui software de calitate.

Se afirm` in literatura dedicat` acestui subiect c`, dat fiind faptul c` sistemele software sunt compuse [n general din par\i similare, majoritatea sistemelor software noi pot]i trebuie asamblate din com-ponente reutilizabile predefinite[STAN1].

Trebuie sa men\ion`m aici c` nu orice folosire a unui component pre-definit poate fi catalogat` ca fiind re-utilizare. Poulin arat` c`, [n general, numai componentele din *surse externe* pot fi considerate ca fiind reutilizate. Re-folosirea unor componente dezvoltate [n cadrul aceluia]i proiect, mai mult, [n cadrul aceleia]i organiza\ii poate fi catalogat` drept o *buna practica [n pro-gramare*[POUL1].

Evolu\ia diverselor medii de dezvoltare determin` apari\ia unui num`r din ce [n ce mai mare de facilit`\i, cu aplicarea con-creta a cerin\elor tehnicilor de reutilizare. Bibliotecile de componente reprezint` una dintre aceste facilita\i.

3. Prototipizarea rapid`

De]i folosit` ini\ial [n ingineria industrial`, prototipizarea a ap`rut [n practica dezvo l-t`rii proiectelor software industriale. Proto-tipizarea rapid` [nseamn` dezvoltarea unui model func\ional al unei par\i sau al [ntregii aplica\ii, ca baz` a evalu`rii]i revizuirii specifica\iilor aplica\iei.

Aceast` tehnic` ofer` proiectan\ilor de aplica\ii posibilitatea de a-]i testa [ntele-gerea problemelor modelate]i solu\iile [nainte de implementarea sistemului, care ar implica unele costuri suplimentare.

Din compara\ia [ntre modelul clasic de dezvoltare]i prototipizarea rapid` rezult` ca o prim` ] esen\ial` concluzie faptul c` aplicând prototipizarea se poate ob\ine mai rapid satisfac\ia utilizatorilor ]i o abordare dinamic` a cerin\elor acestora.

Modelul clasic (WATERFALL)	Prototipizarea rapida (SPIRAL CYCLE)
1. Definirea conceptual`	1. Definirea conceptual`
2. Definirea cerin\elor	2. Implementarea “scheletului” sistemului
3. Proiect de ansamblu	3. Evaluarea utilizatorului]i rafinarea conceptual`
4. Proiectul de detaliu	4. Implementarea cerin\elor rafinate
5. Programarea	5. Evaluarea utilizatorului]i rafinarea conceptual`
6. Teste]i acceptan\`	6. Implementarea cerin\elor rafinate
7. Exploatare	Etc., etc., [ntr-un ciclu continuu

Dinamica fiind cuvântul cheie [n judecarea aportului prototipiz`rii [n dezvoltarea software, putem spune f`r` s` gre]im c` produsele cu un [nalt grad de integrabilitate sunt absolut

necesare [ntr-un ciclu spiral de dezvoltare.

4. Masurarea gradului de integrabilitate

Integrabilitatea este o caracteristic` de cali-tate software ce presupune [nzestrarea unui produs informatic cu acele elemente care s` asigure "cuplarea" cu alte produse/ componente cu efort minim.

Factorii care influenteaz` integrabilitatea sunt:

- complexitatea produsului de realizat]i a celui ce se integreaz` (bucata)
- diferen\a de genera\ie (produs rezultat: genera\ia k, produs integrat: genera\ia k-n)
- gradul de dezvoltare al instrumentelor de interfa\are
- calificarea personalului
- costurile
- resursa de timp disponibil`

Vom [ncerca [n cele ce urmeaz` s` exemplific`m câteva metode de m`surare a integrabilit`\ii.

Considerând num`rul de linii surs` ale produsului baz` *Lb* ]i ale produsului integrat *Li*, precum ]i faptul c` pentru integrare trebuiesc scrise *Ls* linii surs` avem:

$$G = 1 - \frac{Ls}{Lb + Li} \qquad (1)$$

unde *G* este gradul de integrabilitate. Acesta poate lua valori [ntre 0 si 1. Notând cu *Lt* num`rul total de linii ale produsului ob\inut prin integrare, unde

$$Lt = Lb + Li + ls \qquad (2)$$

putem nota cu *L'* num`rul de linii ale produsului ce s-ar fi dezvoltat de la [nceput, f`r` reutilizare. Avem astfel o alt` formul` de calcul pentru gradul de integrabilitate:

$$G = \frac{|L' - Lt|}{Lt} \qquad (3)$$

Integrabilitatea se pune [n eviden\` prin intermediul indicatorilor de complexitate software. Dac` se consider` o aplica\ie c`reia i se dezvolt` software complet de complexitate $C_n$ ]i dac` aceleia]]i aplica\ii i se asociaz` software existent de complexitate $C_e$ ]i integrabilitatea este obtinut` prin software nou de complexitate C_i, justificarea integrabilit`\ii este data de rela\ia $C_n >> C_i + C_e$ (4)

Complexitatea C [n sens Halstead este data de rela\ia

$$C = N_1 \log_2 n_1 + N_2 \log_2 n_2 \quad (5),$$

unde **N_1** este num`rul total al operatorilor utiliza\i [n program; **$N_2$** este num`rul total al operanzilor utiliza\i [n program; **n_1** este num`rul de operatori distinc\i; **$n_2$** este num`rul de operanzi distinc\i.

Dac` $C_e = N_{1e} \log_2 n_{1e} + N_{2e} \log_2 n_{2e}$ ]i $C_i = N_{1i} \log_2 n_{1i} + N_{2i} \log_2 n_{2i}$ ]i dac` produsul rezultat prin integrabilitate are complexitatea

$$C_t = (N_{1i} + N_{1e})\log_2(n_{1i} + n_{1e}) + (N_{2i} + N_{2e})\log_2(n_{2i} + n_{2e}), \qquad (6)$$

[ntre C_t]i $C_i + C_e$ exist` rela\ia $C_t > C_i + C_e$ (7). In acest context, integrabilitatea este justificata cu atât mai mult cu cât $C_n > C_t$ (8).

Productivitatea muncii]i costul proiectului software sunt propor\ionale cu com-plexitatea. Aceste inegalita\i sunt]i criterii de eficien\a a gradului de integrabilitate.

5. Concluzii

De]i este descris mai pu\in [n literatura de specialitate, gradul de integrabilitate este un criteriu ce ar trebui luat tot mai mult [n considerare [n practica m`sur`rii reutili-z`rii software, a m`sur`rii calit`\ii soft-ware. {n esent`, acesta reflect` capacitatea unui produs sau component software de a participa, [mpreun` cu alte produse, la compunerea unor replici mai mult sau mai pu\in exacte ale realit`\ii.

Bibliografie

[BARO1] Baron T., Ivan I., Balog Al. The Characteristic System of Software Products Quality, Economic Computation an Economic Cybernetics Studies and Research, ASE Bucharest, No 1, 1982.

[COMS1] Coman S., Prototyping techni-ques, Computer science. The Proceedings of the 3rd International Symposium of Economic Informatics, Editura INFOREC, Bucuresti, 1997.

[POUL1] Poulin J. S. Measuring Software Reuse. Addison-Wesley, Massachusetts, 1997.

[STAN1] Stanciu E. Reutilizarea compo-nentelor program [n sisteme complexe. Referat, 1999.

[MCCL1] McClure C. Software reuse techniques. Prentice Hall, New Jersey, 1997