Integration of software packages into a new context while maintaining high quality

Florentin ŞCHIOPU

Fast pace of everyday life and the acute need of new in software applications have lead to reusing functional modules while designing information systems. Specifications have a big impact in reusing modules as they have to present the demands, descriptions and functionality in a natural-like language and in a complete, precise and verifiable manner.
Keywords*: software, reuse, integration, quality.*

1 Refolosirea aplicațiilor software aflate în uz curent.

Refolosirea aplicațiilor aflate în uz curent şi integrarea acestora într-un nou context a fost benefică întotdeauna. Problema aceasta apare cu o mai mare intensitate în prezent datorită modificărilor din ce în ce mai dese aduse produselor program.

Software-ul comercial (şi nu numai acesta) trebuie să îndeplinească mai multe cerințe printre care acelea de utilitate, siguranță, uşurință în exploatare şi noutate. Aceste cerințe impun ca pe lângă o interfață prietenoasă, explicită şi concisă, modulele care intra în componența programelor să fie sigure în exploatare şi să producă rezultate corecte. Dacă aceste module integrate în noul software operațional au fost testate anterior în condiții normale de exploatare şi s-au comportat foarte bine încrederea în ele va fi şi mai mare.

Nu toate aplicațiile pot fi integrate. Sunt preferate acelea care au fost proiectate special pentru reutilizare. Ideea este de a integra programe/module vechi într-o aplicație nouă şi nu de a construi pe structura veche un produs nou.

Acele programe care sunt uzate moral nici nu pot face obiectul unui studiu referitor la integrare. Acestea au fost proiectate să opereze într-un mediu mult diferit de cel prezent, iar unele operații nu mai au nici o semnificație

Într-un context economic instabil sau în tranziție, unde coerența legislativă lasă mult de dorit, unele aplicații, în special cele de gestiune sau contabilitate, nu pot ține pasul cu cerințele mereu noi.

Dacă nu sunt diferențe mari, unele aplicații pot fi modificate în aşa fel încât acestea să corespundă cerințelor rezultatelor, urmând ca ulterior echipa să se ocupe efectiv numai de aspectul referitor la procesul de integrare.

Activitatea de creare de aplicații noi, de la zero, fără nici o altă bază, pornind numai de la cerințele beneficiarului este atât costisitoare cât şi mare consumatoare de resurse de timp.

Hotărârea pentru includerea aplicației existente în structura celei noi se ia după o atentă analiză a tuturor consecințelor. Acest lucru se stabileşte încă din faza de proiectare, luându-se în calcul costurile rescrierii precum şi posibilitățile de includere în cadrul aplicației.

În cazul programelor care conduc procese tehnologice, refolosirea unor părți componente într-o nouă structură este greu de realizat. Proiectarea unor linii de procese tehnologice presupune o atenție deosebită datorită numărului mare de parametri de intrare (cantitate, temperatură, viteză, formă, etc.) care, fiecare în parte, conduce la o mare diversitate de situații ce pot apărea precum şi la o paletă largă de soluții sau acțiuni care vor fi executate.

Procesele automate exclud complet intervenția omului, fiecare parametru având un rol foarte important în continuarea activității.

Dacă modificările operate sunt mici, atunci se pot folosi elemente din vechea aplicație, sau, mai bine spus, se poate modifica aceasta, dar şi în acest caz este necesară multă atenție.

În cazul aplicațiilor de dimensiuni foarte mari, cu mulți parametri de intrare/ieşire e bine ca aplicațiile să fie proiectate pe segmente, de preferat de mici dimensiuni, care să se ocupe de o singură operație, segmente ce pot fi refolosite în cazul unor modificări

ulterioare.
Dacă elementele ce se doresc a fi reutilizate nu au fost create de la început în acest sens, e bine ca proiectarea în cazul aplicațiilor de procese să se facă de la început.

2. Ciclul de realizare a produselor program. Ciclul de viață.

Ciclul de realizare a produselor program include toate etapele, începând cu luarea deciziei de realizare şi se termină cu darea sa în folosință. Este o derivată din ciclul de viață şi acoperă intervalul între luarea deciziei de realizare şi punerea sa în funcțiune pentru întreaga comunitate de utilizatori finali căreia îi este destinat.
Există mai multe metode de etapizare a ciclului de viață în funcție de autori. În principal, ciclul de viață cuprinde următoarele etape:

- Specificare cerințe
- Analiză cerințe
- Proiectare de detaliu
- Implementare
- Testare
- Instalare
- Întreținere

Produsele program se creează ca răspuns dat la un set de nevoi la un moment dat şi odată cu evoluția acestui set de nevoi produsul software se maturizează, atât în perioada de dezvoltare cât şi în perioadele de operare.
În stadiul de concept, viitorul produs poate fi vag perceput. După realizarea proiectului preliminar produsul începe să capete contur, iar după realizarea proiectului de detaliu, acesta este complet conturat.
Etapele prezentate anterior reprezintă un model ideal. În realitate, produsul software nu trece fără opriri şi revizuiri de la o etapă la alta (datorită unor evenimente de natură variată – modificarea cerințelor, descoperirea unor erori de proiectare, etc.). Ca urmare, activitatea de mentenanță se aplică la toate nivelele de dezvoltare ale modelului.
Modificările modelului surprinse în *figura 1* vizează introducerea unor bucle de feedback în două zone: de la proiectarea programelor la specificarea cerințelor şi de la testare la proiectarea programelor. Problema care apare în această situație este aceea că metodologia nu prevede ce în nici un sens ce informații vor trebui să circule pe buclele de feedback de la o etapă la alta.

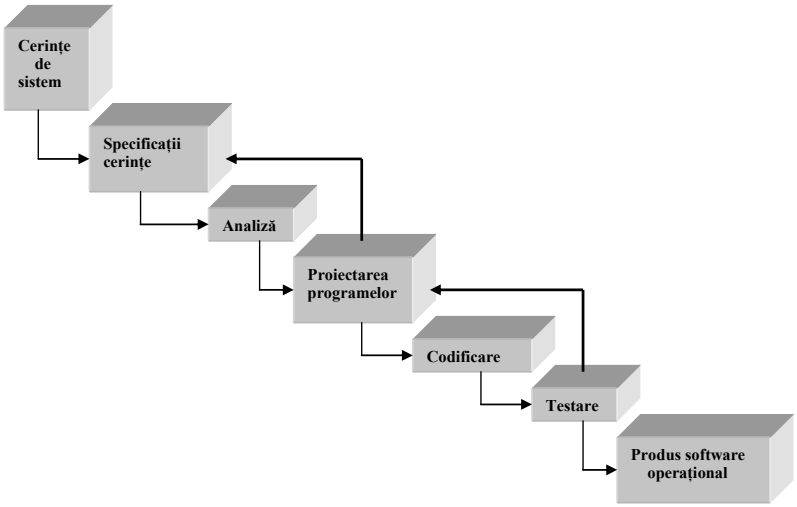

Fig.1. Introducerea buclelor de feedback în modelul secvențial

În practică, după fiecare etapă urmează o buclă de feedback, prin care se urmăreşte ca în stadiul următor să fie admisă varianta cea mai bună.
Totuşi, ca regulă de bază trebuie reținut că „Fie faci totul bine de prima dată, fie refaci totul de la început”.

3. Managementul calității specificațiilor pentru dezvoltarea aplicațiilor informatice.

În procesul de refolosire software specificațiile au un rol important, sunt exprimate într-un limbaj cât mai natural iar cerințele definirii acestora au o strategie bine pusă la punct.

Specificațiile sunt acele documente care prezintă într-o manieră completă, precisă şi verificabilă, cerințele, descrierile, modul de lucru sau alte caracteristici ale aplicațiilor informatice şi adesea procedurile pentru determinarea gradului de realizare a prevederilor inițiale.

În realizarea aplicațiilor informatice sunt utilizate mai multe tipuri de specificații, *figura 2*.

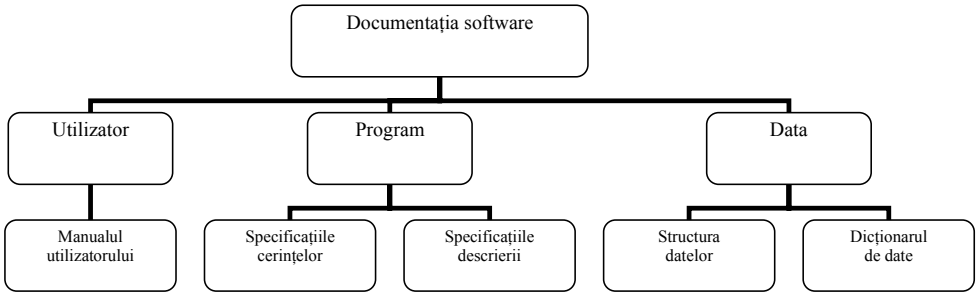

Fig.2. Tipuri de specificații

Specificațiile de avertizare a schimbărilor – *specification change notice (SCN)* - sunt documente utilizate în managementul configurării aplicațiilor informatice care propun, transmit şi înregistrează schimbările din sistem.

Specificațiile de limbaj – *specification language* – sunt documente care utilizează un limbaj, cel mai adesea o combinație între limbajul natural şi cel formal, folosite pentru a elimina ambiguitățile şi a exprima cerințele, descrierile, modul de lucru sau alte caracteristici ale aplicațiilor informatice.

Cerințele specificațiilor de limbaj – *requirements specification lanquage* – sunt utilizate pentru dezvoltarea, analizarea şi verificarea protocoalelor, cerințelor hardware şi software.

Specificațiile formale – *formal specification* - sunt acele specificații concepute şi scrise în concordanță cu standardele în vigoare.

Specificațiile de produs – *product specification* – sunt documente care descriu modul de implementare a aplicației informatice. Din punct de vedere software aceste specificații se referă la prezentarea versiunii aplicației. Ele descriu caracteristicile proiectului sau a produsului existent pentru a arăta utilizatorilor sau potențialilor clienți avantajele acestuia.

Specificațiile cerințelor – *requirements specification* – sunt documente care specifică cerințele aplicației informatice sau a componentelor acesteia. În această categorie sunt incluse cerințele funcționale, de performanță, de interfață, de descriere şi standardele de dezvoltare.

Specificațiile ramificate – *specification tree* – sunt diagrame care cuprind toate specificațiile utilizate pentru o anumită aplicație informatică, arătând şi legăturile dintre diverse aplicații.

Specificațiile descrierii – *specification description* - sunt documente care descriu structura aplicației informatice sau a componentelor ei. Acestea prezintă aplicația informatică sau componentele sale, controlul logic, structura datelor, formatele de intrare/ieşire, descrierea interfeței şi algoritmii de lucru.

Specificațiile funcționale – *functional specification* – sunt documente care definesc foarte clar scopul aplicației informatice. Aceasta depinde de metodologia utilizată în programare şi poate să furnizeze detalii privind modul în care programul este împărțit în module precum şi felul în care modulele interacționează între ele. Specificațiile funcționale descriu aplicațiile informatice, din perspectiva utilizatorului, de genul interfeței aplicație informatică - utilizator precum şi modul de exploatare al acestora.

Specificațiile sunt necesare în toate etapele de dezvoltare a aplicațiilor informatice, începând cu faza de concepție, urmând fazele de elaborare a cerințelor, de descriere, de construcție, de testare, de implementare, de operare şi întreținere, uneori chiar şi în faza de

retragere.
În elaborarea specificațiilor trebuiesc prevenite împrejurările în care au loc testele redundante, precum şi evitarea unor proceduri şi funcții foarte laborioase.
Trebuie dovedit beneficiarului utilitatea, performanțele aplicației informatice şi nu în ultimul rând să oferim garanția calității acesteia.

3. Managementul calității datelor în aplicațiile informatice.

Unul dintre cei mai importanți indicatori pentru evaluarea competitivității unei aplicații software este calitatea. Calitatea oricăror aplicațiilor software va creşte doar printr-o atentă concentrare a conducerii asupra aspectelor acestora.
Prin *managementul calității* se înțelege, în primul rând, luarea deciziilor pe baza unei cunoaşteri clare şi foarte corecte a realității, aceasta implicând folosirea cunoştințelor, informațiilor şi datelor. Este necesar să se asigure calitatea datelor pe parcursul întregului ciclu de viață a aplicațiilor software.
Prin *managementul total al calității datelor* se înțelege un proces continuu de îmbunătățire al tuturor proceselor de dezvoltare a aplicațiilor informatice, având ca prioritate pe termen lung prevenirea noncalității datelor.
Prin *dată* se înțelege un singur element component al informației. Mulțimea datelor formează elementele informației.
Prin *date noncalitative* înțelegem următoarele tipuri de date:
• *date redundante* – date ce pot conduce la costuri ridicate de producție;
• *date incomplete ori neactualizate* – date ce pot conduce la alterarea imaginii firmei;
• *date slab definite* – date ce pot conduce la folosirea eronată a informației.
Dimensiunea calității datelor reprezintă un set de atribute de calitate ale datelor, la care cei mai mulți beneficiari ai aplicațiilor software reacționează într-un mod destul de consistent. Astfel:
• datele trebuie să fie *accesibile* beneficiarilor – beneficiarul să ştie cum să regăsească datele;
• beneficiarul trebuie să poată *interpreta* datele – datele să fie prezentate într-o codificare cunoscută utilizatorului;
• datele trebuie să fie *relevante* pentru beneficiar – datele să fie relevante şi oportune pentru folosirea de către utilizatorul de date în procesul de luare a deciziilor;
• beneficiarul aplicațiilor software trebuie să aprecieze datele ca fiind *corecte* – datele să fie corecte, obiective şi să provină dintr-o sursă cu reputație.
Cele mai frecvent menționate dimensiuni ale calității sunt:
• *credibilitate* – gradul în care datele sunt acceptate sau privite ca adevărate, reale şi credibile;
• *relevanță* – gradul în care datele sunt aplicabile şi folositoare pentru activitatea de realizat;
• *acuratețe/exactitate* – gradul în care datele sunt corecte, de încredere şi certificate ca fiind fără erori;
• *interpretabilitate* – gradul în care datele sunt prezentate într-un limbaj adecvat, definirile sunt clare;
• *uşurința înțelegerii* – gradul în care datele sunt clare, fără ambiguități şi uşor de înțeles;
• *accesibilitate* – gradul în care datele sunt disponibile sau uşor de regăsit;
• *obiectivitate* – gradul în care datele sunt neprejudiciate şi imparțiale;
• *oportunitate* – gradul în care vechimea datelor este adecvată pentru activitatea de realizat;
• *completitudine* – gradul în care datele acoperă calitativ şi cantitativ domeniul activității de realizat;
• *reputație* – gradul în care datele sunt apreciate din punct de vedere al sursei sau conținutului lor;
• *consistența reprezentării* – gradul în care datele sunt mereu prezentate în acelaşi format şi sunt compatibile cu formatele precedente;
• *uşurința operării* – gradul în care datele sunt uşor de manevrat şi manipulat;
• *concizie* – gradul în care datele sunt reprezentate compact şi nu în cantități foarte mari;
• *securitatea accesului* – gradul în care accesul la date poate fi restricționat şi astfel pro-

tejat;
• *cantitatea suficientă de date* – gradul în care cantitatea sau volumul de date este adecvat;
• *flexibilitatea* – gradul în care datele sunt expandabile, adaptabile şi răspund cu uşurinţă şi altor cerinţe.

Datele reprezintă de cele mai multe ori cele mai importante resurse ale unei organizaţii. Calitatea ridicată a datelor nu doar reduce costurile, dar creşte şi disponibilitatea şi bunăvoinţa beneficiarului de a plăti pentru bunurile şi serviciile oferite, ceea ce duce la îmbunătăţirea poziţiei competitive.

Reproiectarea aplicaţiilor software încearcă să simplifice şi să eficientizeze activitatea beneficiarilor acestora, pentru a se minimiza probabilitatea de apariţie a erorilor datelor.

Bibliografie

[BALO99] Balog Al., *Evaluarea proceselor software* - Editura I.C.I., Bucureşti, 1999

[COST99] Costache D., Avram G., Moise M., Sonea E. ş.a., *Manualul calităţii* AISTEDA - editura AISTEDA, Bucureşti, 1999

[GHIL01] Bogdan Ghilic-Micu - *Aspecte ale impactului trecerii la societatea informaţională*, revista "Informatica economică" editată de Catedra de Informatică Economică şi asociaţia INFOREC, cu sprijinul Ministerului Educaţiei si Cercetării volumul V, Nr. 3 (19)/2001 (pag. 10 - 15)

[IVAN99] Ion Ivan, Paul Pocatilu – *Testarea software Orientat obiect*, Editura Inforec, Bucureşti 1999

[IVAN99] Ion Ivan, Gheorghe Noşca, Sebastian Tcaciuc, Otilia Pârlog, Răzvan Căciulă – *Calitatea datelor*, Editura Inforec, Bucureşti 1999

[IVAN01] Ion Ivan, Paul Pocatilu, Mihai Amitroaie - *Metrici ale societăţii informationale*, revista "Informatica economică" editată de Catedra de Informatică Economică şi asociaţia INFOREC, cu sprijinul Ministerului Educaţiei si Cercetării volumul V, Nr. 4 (20)/2001 (pag. 33 - 40)

[ROSC01] Ion Gh. Roşca, Floarea Năstase, Victor Patriciu, Octavian Paiu, Gabriel Şutac, Carmen Stanciu, Ion Bica – Cryptographic methods and techniques for authentication through digital signature in e-comerce. Juridical controversies and national approach, Information Society, The proceedings of the fifth international symposium of economic informatics may 2001 Bucharest 10-13 may 2001, Editura Economică, Bucureşti 2001 (pag. 345-352)

[STAN01] Carmen Stanciu - *Soluţii de e-business pentru întreprinderi cu aplicaţii moştenite şi pentru firme mici start-up*, revista "Informatica economică" editată de Catedra de Informatică Economică şi asociaţia INFOREC, cu sprijinul Ministerului Educaţiei si Cercetării volumul V, Nr. 3 (19)/2001 (pag. 36 - 39)

[STAN01] Carmen Stanciu - *Strategie de planificare şi implementare a sistemelor de e-comert*, revista "Informatica economică" editată de Catedra de Informatică Economică şi asociaţia INFOREC, cu sprijinul Ministerului Educaţiei si Cercetării volumul V, Nr. 4 (20)/2001 (pag. 46 - 51)

[STOI01] Marian Stoica - *Perspectiva pentru telelucru în societatea bazată pe informaţie şi cunoaştere*, revista "Informatica economică" editată de Catedra de Informatică Economică şi asociaţia INFOREC, cu sprijinul Ministerului Educaţiei şi Cercetării volumul V, Nr. 2 (18)/2001 (pag. 29 -33)

[IEEE94] IEEE Standards Collection, *Software Engineering*, Published by the Institute of Electrical and Electronics Engineers, Inc., 1994 edition;

[****96] Key Computer Consultants, Inc., *How to Write a Software Specification,* http://www.worthy.com/~kcc/kccspec.html Key Computer Consultants, Inc., 1996

[****01] *Strategia guvernului privind Informatizarea Administratiei Publice*, http://www.mapgov.ro/, Bucureşti – 2001

[******] *Telecomert*, http://www.ici.ro/telelucru/telelucru/ds_com.html

[******] Functional Specification http://www.pcwebopedia.com/term/f/functional_specification.html,

[******] Requirements definition/specification http://www.crab.rutgers.edu/~ssykes/ch4/tsl006.html

[******] What is a requirement? http://www.maths.bath.ac.uk/~jap/MATH0026/require.html

[******] Sotware requirements Specification http://www.esi.es/Help/Dictionary/Definitions/Software_Requirements_Specification.html

[******] Software Engineering Principles http://www.eecs.wsu.edu/~sheldom/mms/seslides/