1. 

**Calitatea proiectării software**

 **Studenţi Grupa**

**Stroescu Ana-Raluca 441 A**

**Dragomir Ana-Maria**

**Profesor Coordonator: Profesor Doctor Inginer Ştefan Stăncescu**

 **Bucureşti 2016**

**Cuprins**

**1.Conceptul de calitate software – Stroescu Ana Raluca**

**1.1. Aspecte generale**

**1.2. Necesitatea măsurării calităţii produselor software**

**1.3 Preocupări în domeniul măsurării calităţii produselor software**

**2. Modelele calitatii in domeniul tehnologiei informaţiei – Stroescu Ana Raluca**

**2.1 Modele tradiţionale ale calităţii produselor software**

**2.2 Aspecte specifice ale modelelor actuale ale calităţii produselor software**

**3.Masurarea calitatii produselor software – Stroescu Ana Raluca**

**4.Caracteristicile de calitate – Dragomir Ana Maria**

**5. Managementul calităţii software - Dragomir Ana Maria**

**5.1. Planificarea calitatii**

**5.2 Asigurarea calitatii**

**5.3 Imbunatatirea calitatii**

**6. Concluzii – Stroescu Ana Raluca si Dragomir Ana Maria**

**7. Bibliografie**

1. **Conceptul de calitate software**
	1. **Aspecte generale**

Calitatea este un concept care se utilizează în toate domeniile vieții economice și sociale. Aceasta prezintă un caracter subiectiv și are semnificații particulare pentru diferite domenii, sectoare, funcțiuni sau obiecte specifice. Calitatea este un termen general, aplicabil la cele mai diferite trăsături sau caracteristici, fie individuale, fie generice și a fost definită în diferite moduri de către diverși experți sau consultanți în calitate, care îi atribuie acestui termen semnificații diferite.

Conform standardului ISO 8402, calitatea reprezintă ansamblul

caracteristicilor unei entităţi (produs, activitate, proces, organizaţie, persoană)

care conferă acesteia aptitudinea de a satisface cerinţele exprimate sau implicite.

Conform unor aspecte parţiale ale calităţii unui produs sau serviciu, calitatea

semnifică:

• aptitudinea de a fi utilizat sau întrebuinţat;

• satisfacerea cerinţelor clientului;

• conformitatea cu o documentaţie sau cu exigenţele beneficiarului.

 Produsele software au devenit o parte esenţială a multor sisteme moderne şi,

în general, tind să transforme decisiv mediul de lucru cotidian prin automatizarea

activităţilor zilnice. De fiecare data când sistemele software greşesc, consecinţele

sunt foarte grave şi conduc la scăderea productivităţii, pierderi importante de

bunuri de orice natură, pierderea clienţilor şi scăderea vânzărilor, potenţiale

ameninţări la siguranţa vieţii oamenilor, breşe în asigurarea securităţii

informaţiilor, pierderi importante de date şi informaţii, cheltuieli mari pentru

recuperarea datelor şi repunerea în funcţiune a sistemelor. Ca urmare a aderării

României la Uniunea Europeana, a apărut necesitatea dezvoltării de produse şi

servicii care sa fie compatibile şi conforme cu standarde specifice pentru

acceptarea lor pe diverse pieţe. [1]

Conform standardului ISO 9000 -3, produsul software este definit ca fiind: “

o creaţie intelectuală care cuprinde programe, proceduri, reguli şi orice

documentaţie asociată referitoare la funcţionarea unui sistem de prelucrare a

datelor”.

Creşterea complexităţii produsului software necesită implementarea

diferitelor procese de management. Acestea gestionează calitatea produsului sau

cantitatea resurselor necesare şi se implică în etapele ciclului de viaţă al produsului

final. Se impune modularizarea produsului program în mai multe componente, fapt ce prezintă avantaje prin:

· reutilizarea acestora, incluzându-le în alte produse;

· buna gestiune a proceselor de analiză, proiectare şi implementare;

· obţinerea unei imagini clare a componentelor funcţionale;

· specializarea analiştilor, designerilor şi programatorilor.

* 1. **Necesitatea măsurării calităţii produselor software**

Realizarea de produse software pentru aplicaţii complexe presupune repartizarea unor programatori cu experienţă şi utilizarea unor sisteme de calcul performante. În acest scop, trebuie să se specifice câţi programatori sunt necesari şi cât timp vor lucra. De asemenea, trebuie specificate câte posturi de lucru se vor aloca, ce resurse sunt necesare (compilatoare, periferice, memorie ) şi pe ce durată. Dezvoltarea de aplicaţii complexe presupune analiză, proiectare, testare, implementare, toate necesitând forţă de muncă şi resurse.

Cumpărătorul (utilizatorul) alege după diferite criterii un anumit produs. Dominantă este corelaţia cost-calitate. Atât costul, cât şi calitatea, trebuie exprimate cantitativ, pentru a evidenţia mărimea avantajului pe care o determină alegerea făcută.

 Analiza comparată prin prezentarea în paralel a unor programe destinate rezolvării aceleiaşi probleme scoate în evidenţă ceea ce au ele în comun, precum şi ceea ce le diferenţiază. Trăsăturile importante care lipsesc vor determina perfecţionări ulterioare. Noul produs trebuie caracterizat în mod riguros, atât prin costurile implicate, cât şi prin nivelul caracteristicilor de calitate obţinute. Fără măsurare, orice încercare de a stabili modele care să descrie legăturile structurale între valorile tehnice şi fizice, nu se poate realiza

În domeniul IT, există dificultăţi fundamentale în măsurarea produselor software. Originalitatea şi creativitatea umană sunt strâns legate de calitatea proiectului. Aceste aspecte ale calităţii sunt greu de măsurat, cu atât mai mult cu cât programatorii îşi văd munca mai degrabă ca pe o” operă de artă”, decât ca pe un produs comercial.

 Evaluarea calității înseamnă evaluarea valorii produsului, indiferent dacă este de tip software sau nu . Un produs care are o calitate mai mare are mai multă valoare decât un produs de o calitate inferioară . Măsurarea valorii necesită răspunsul la o întrebare : valoarea cui? În ceea ce priveşte calitatea software , este utilă concentrarea asupra a trei grupuri de persoane care sunt interesate de valoarea produsului. Figura 1 prezintă aceste grupuri.



Fig.1 [2]

 După cum se poate observa, procesul de dezvoltare transformă o idee întru-un software utilizabil. Cele trei grupuri interesate de calitatea produsului sunt:

* Utilizatorii – aplică software-ul în rezolvarea unei probleme;
* Echipa de dezvoltare – creează software-ul;
* Sponsorii proiectului - oamenii care plătesc pentru crearea software-ului . Pentru software-ul dezvoltat de o organizație pentru uz propriu , de exemplu , aceşti sponsori sunt, de obicei, oameni de afaceri din cadrul organizației .

**1.3 Preocupări în domeniul măsurării calităţii produselor software**

 Primele studii şi cercetări pentru măsurarea calităţii în domeniul produselor software au fost efectuate în anii 1970, în Statele Unite ale Americii. Printre obiectivele acestor studii se numără:

• definirea cadrului conceptual al calităţii software-ului, pentru a putea specifica obiectivele calităţii pentru produsele software;

• evaluarea calităţii intrinseci a produselor software pe parcursul dezvoltării sale.

 Trei dintre caracteristicile majore ale software-ului:

• software-ul poate fi considerat un produs complex, dacă definim complexitatea prin numărul de componente elementare care-l constituie (software de câteva sute de mii de instrucţiuni este acum ceva obişnuit);

• software-ul este un produs imaterial, intangibil;

• software-ul este un produs unic, adică realizarea sa se opreşte în momentul în care prototipul este sau este gata pentru vânzare. Software-ul nu are o faza de producţie de serie. În cazul livrării către mai mulţi clienţi, el este doar multiplicat. Acestor caracteristici li s-au adăugat începând cu anii 1970 constatările economiştilor, printre care amintim: creşterea exponenţială a costurilor, absenţa managementului proiectelor , numeroase eşecuri în dezvoltarea unor produse din domeniul IT, insatisfacţia utilizatorilor si mentenanţă defectuoasă. Concluziile studiilor au fost grupate pe două direcţii de acţiune :

• raţionalizarea producţiei prin : descrierea proceselor de dezvoltare a software-ului, descompunerea proceselor în faze succesive, definirea entităţilor (activităţi şi resurse) care pot fi planificate, administrate şi evaluate;

• definirea si punerea in aplicare a procedurilor si tehnicilor care să permită managementul calităţii produselor software în timpul dezvoltării acestora.

Pe parcursul studiilor au fost identificate componentele calităţii software-ului din punctul de vedere al utilizatorului. În identificarea criteriilor a trebuit să se răspundă la întrebarea următoare: pentru a atinge nivelul calităţii cerut de un factor dat, care sunt criteriile asupra cărora echipa de dezvoltare a software-ului poate acţiona? Criteriile au fost apoi legate de un ansamblu de atribute măsurabile, care au fost numite "metrici".

****

Fig 1. Modelul ierarhic al masurarii calitatii software-ului [4]

În fig.1 este prezentat modelul ierarhic al măsurării calităţii software-ului. Această abordare a fost sugerată pentru prima dată de B. Boehm şi echipa sa, apoi de Mc.Call, Richards şi Walters, care au început şi demersurile privind standardizarea modelului şi a terminologiei aferente. În privinţa standardizării, una dintre organizaţiile de standardizare din SUA, „Institute of Electrical and Electronics Engineers” (IEEE) a realizat în 1988 un proiect de standard care a apărut în 1992 ca standard - IEEE STD 1061:1992 "Standard for Software Quality Metrics Methodology"7; în 1991 şi organizaţia de standardizare internaţională „International Organization for Standardization” (ISO) împreună cu organismul internaţional de standardizare din domeniul electrotehnic „International Electrotechnical Commission” (IEC) au realizat un standard şi anume ISO /IEC 9126:1991 "Information technology – Software product evaluation – Quality characteristics and guideline for their use" [5].

1. **Modelele calităţii în domeniul tehnologiei informaţiei**

Cercetările specialiştilor au condus la elaborarea unui model conceptual al sistemuluide caracteristici, cunoscut sub denumirea de „modelul calităţii software-ului”. Modelul a fost sugerat pentru prima dată de Barry Boehm şi colaboratorii săi de la TRW Systems Group (SUA). Ulterior, James McCall, P.Richard şi G. Walters de la General Electric Company au contribuit la perfecţionarea şi extinderea acestuia. Modelul este recomandat de Organizaţia Internaţională de Standardizare (ISO) prin standardul ISO/ IEC 9126:1991.

Caracteristicile calităţii servesc la definirea cerinţelor utilizatorului şi la stabilirea obiectivelor calităţii pentru produsul software. Unii specialişti numesc acest nivel, ca find nivelul factorilor. Pe următorul nivel, se identifică un set de caracteristici derivate ale calităţii care se combină în seturi de criterii necesare pentru evaluarea caracteristicilor de pe primul nivel. Caracteristicile derivate au semnificaţie tehnică şi sunt relevante pentru personalul de specialitate (analişti, proiectanţi, programatori,). Ele facilitează comunicarea între şeful de proiect şi personalul de specialitate în ceea ce priveşte atingerea obiectivelor calităţii. La ultimul nivel de definesc metricile, care permit măsurarea caracteristicilor derivate şi, în consecinţă, exprimarea cantitativă şi estimarea factorilor de pe primul nivel.

**2.1 Modele tradiţionale ale calităţii produselor software**

În această categorie includem modelele calităţii care au stat la baza dezvoltărilor actuale, incluse în standardele internaţionale. Cele mai cunoscute sunt modelele grupului de cercetare de la TRW Systems Group din SUA, condus de B.W.Boehm şi modelul dezvoltat la General Electric Company de J.McCall şi colaboratorii săi. Există şi alte modele menţionate în lucrările de specialitate, însă acestea reprezintă extensii ale modelelor clasice amintite.

1. Modelul calităţii produselor software propus de Boehm (1978)

 În figura 2 este prezentat modelul calităţii propus de Boehm, cu evidenţierea caracteristicilor şi a relaţiilor dintre ele.



Fig. 2 Modelul calităţii produselor software propus de Boehm [6]



Tabel nr 1- Semnificaţia caracteristicilor calităţii produselor software (modelul propus de Boehm) [7]



Tabel nr 2 - Semnificaţia caracteristicilor derivate ale calităţii produselor software (modelul propus de Boehm)[7]

1. Modelul calităţii produselor software propus de McCall (1977)

Modelul calităţii produselor software propus de McCall este prezentat în fig.3, cu evidenţierea caracteristicilor calităţii şi a relaţiilor dintre ele.



Fig. 3 Modelul calităţii produselor software propus de McCall [6]

c) Modelul calităţii produselor software definit de standardul ISO/ IEC 9126

Modelul calităţii ISO/IEC 9126 recomandă utilizarea următoarelor caracteristici principale: funcţionalitate, utilizabilitate, fiabilitate, eficienţă, mentenabilitate şi

portabilitate. Datorită terminologiei utilizate, modelul este denumit „modelul caracteristici principale-caracteristici derivate- metrici*”.*



Tabelul 4 Caracteristicile şi semnificaţia caracteristicilor derivate din modelul ISO/ IEC 9126

**2.2 Aspecte specifice ale modelelor actuale ale calităţii produselor software**

Un aspect important al modelelor calităţii produselor software îl reprezintă divizarea ierarhică a caracteristicilor calităţii, metricile fiind la nivelul de bază. În funcţie de modelul calităţii stabilit şi de cerinţele specifice ale produsului, se determină, pentru fiecare proiect, un număr specific de caracteristici principale, caracteristici derivate şi de metrici. Numărul acestora poate varia considerabil. De exemplu, în cazul unui produs cu cerinţe deosebite privind mentenanţă, trebuie să se acorde atenţie suplimentară modularităţii arhitecturii software, clarităţii şi uşurinţei de înţelegere a codului în proiectare şi codificare. Aceste însuşiri pot fi în relaţie inversă cu eficienţa codului executabil, de exemplu, un program cu o structură modulară clară şi un cod clar poate fi mai încet în execuţie decât unul nestructurat. Rezultă că pot fi conflicte de ţeluri în definirea diferitelor caracteristici derivate. În general, în cazul multor proiecte se specifică numai cerinţele funcţionale, cerinţele de performanţă şi cerinţele interfeţei cu utilizatorul. Cerinţele privind uşurinţa de întreţinere, portabilitatea sau caracteristici similare, sunt în majoritatea cazurilor ignorate. Motivul este că cei responsabili cu specificaţia nu au suportul necesar pentru formularea acestor cerinţe privind calitatea. Modelele calităţii şi metricile calităţii oferă posibilitatea de a specifica aceste cerinţe şi ajută la îmbunătăţirea calităţii procesului. Problemele tipice ale procesului sunt de exemplu, o planificare şi un control nesatisfăcătoare ale proiectului, nedeterminarea costului de dezvoltare la începutul proiectului şi întârzierea în livrarea produsului. Alegând metricele potrivite pentru factori specifici ai procesului, cum ar fi planificarea proiectului şi leadership, sau proceduri standardizate în cadrul proiectului, problemele referitoare la calitatea produsului sau procesului pot fi mai uşor rezolvate. Modelele calităţii ne pot ajuta, de asemenea, să înţelegem şi să demonstrăm, mai bine, relaţia dintre caracteristicile calităţii produselor software. Structura sistemului de caracteristici evidenţiază anumite relaţii de subordonare şi de dependenţă ce sunt reflectate în mod diferit de modelele calităţii. Astfel, se pot identifica următoarele tipuri de relaţii: ierarhice, de influenţă de tip conflictual şi de influenţă de tip complementar.

a) Relaţiile ierarhice, de subordonare, care sunt definite prin structura sistemului de caracteristici, diferă de la un model la altul. Aceste relaţii fac caracteristicile respective să fie “neconectate”. Două caracteristici ale calităţii sunt neconectate dacă nu există o interacţiune vizibilă între ele, de exemplu dacă nu există o interacţiune vizibilă între caracteristicile derivate. De exemplu, portabilitatea şi corectitudinea sunt considerate ca fiind neconectate.

b) Relaţiile de influenţă de tip conflictual, semnifică faptul că pe măsură ce creşte nivelul unei caracteristici, este de aşteptat ca nivelele celorlalte caracteristici să scadă. Aceste relaţii fac caracteristicile respective să fie concurente. Două caracteristici sunt concurente dacă îmbunătăţirea unei caracteristici duce la diminuarea celeilalte. De exemplu, creşterea fiabilităţii duce, de regulă, la diminuarea performanţei.

c) Relaţiile de influenţă de tip complementar, semnifică faptul că dacă nivelul unei caracteristici creşte, atunci este de aşteptat ca şi nivelul celorlalte caracteristici să crească. Aceste relaţii fac caracteristicile respective să fie “revitalizante”. Două caracteristici se revitalizează dacă îmbunătăţirea uneia afectează îmbunătăţirea celeilalte. De exemplu, corectitudinea şi fiabilitatea sunt caracteristici care se revitalizează. [4]

**3.Masurarea calităţii produselor software**

Efectele noilor tehnologii sunt cele care determina aplicarea lor, numai in

masura in care, prin cuantificare, conduc la rezultate superioare. De exemplu,

programarea structurata, prin sistematizarile determinate, a condus la reducerea

complexitatii, la cresterea productivitatii si facilitarea mentenantei software-ului.

Toate acestea au fost puse in evidenta cantitativ, printr-o serie de indicatori.

Programarea orientata spre cresterea gradului de reutilizabilitate determina

economie de munca, chiar daca nivelul de complexitate a problemelor ce se

rezolva este mult mai mare. Aceste tehnologii se implementeaza cu ajutorul unor

produse software dedicate. Studiul profitabilitatii, necesar achizitionarii acestor

instrumente, trebuie sa raspunda la intrebarea in cat timp se recupereaza investitiile

de achizitionare. Acest raspuns este posibil numai prin efectuarea unor cuantificari

ale avantajelor generate de programarea structurata.

Observarea este un proces prin care se apreciază valoarea

caracteristicilor unui produs în timpul utilizării. De exemplu, timpul de

răspuns al programelor este observat. Scopul acestei măsurări este de a evalua

conformitatea cu proiectul.

Prin evaluare, raportarea unui tip de caracteristică aşteptată are loc folosind

un produs în diferite faze de dezvoltare cât şi într-o aplicare simulată.

Estimarea permite evaluarea valorii anticipate a unei caracteristici cu

ajutorul descrierii unui model al produsului sau al contextului aplicaţiei. De

exemplu, timpii de răspuns într-un sistem informaţional pot fi estimaţi folosind

predicţii pe un model simulat.

Dezavantajul folosirii observaţiei constă în faptul că dezvoltarea produsului

este deja terminată, iar erorile pot deja să existe. Prin estimare pot fi evitate parţial

aceste dezavantaje, din moment ce vorbim aici de o observare “ipotetică” asupra

tipurilor de caracteristici.

Una dintre diferenţele dintre predicţie (estimare) şi observare constă în

faptul că, condiţiile implementării produsului sau aplicaţiei produsului sunt

numai parţial cunoscute. O altă diferenţă este aceea că în cazul predicţiei

ideile se aplică pe un model de produs abstract.

Pentru procesul de dezvoltare al produselor software, se poate aplica

postulatul lui Galileo „numără ce este numărabil, măsoară ce este măsurabil şi

dacă ceva nu este măsurabil, încearcă să-l faci măsurabil”.

Prin măsurare, în domeniul IT, se înţelege obţinerea unei mulţimi de

valori ale unei metrici a unui proces de dezvoltare, a unui proces de mentenanţă

sau a unui produs software prin intermediul unor instrumente (de exemplu: prin

analize statice, colectare de date de proiectare şi prin sisteme de evaluare a

proiectului).

Procesul de masurare implica parcurgerea urmatoarelor etape:

· Determinarea obiectelor masurarii;

· Fixarea metricilor masurarii si a scalelor masurarii;

· Alocarea metodelor de masurare si a instrumentelor de masurare;

· Determinarea valorilor;

· Interpretarea rezultatelor.

Masurarea caracteristicilor software presupune elaborarea unei descrieri

cantitative adecvate, prin cuantificare, care sa faciliteze si sa pregateasca

masurarea propriu-zisa.

Cuantificarea constituie operatia de introducere a conceptelor cantitative, de

trecere in studiul unui fenomen de la aprecieri atributive la concepte cantitative. In

urma operatiilor de cuantificare numerica se obtin metricile.

Măsurarea unui atribut care nu depinde de măsurarea altui atribut se numeşte

măsurare directă. Rezultatul măsurării directe caracterizează atributele simple, cu

preponderenţă atributele interne ale produsului.

Măsurarea unui atribut care necesită măsurarea unuia sau mai multor

atribute ale aceleiaşi entităţi sau ale entităţilor din alte clase se numeşte

măsurare indirectă. Rezultatul măsurării indirecte se utilizează preponderent

pentru caracterizarea atributelor externe ale produselor, în scopuri de evaluare,

control şi predicţie.

În acest context, numeroase lucrări de specialitate descriu experimente

ale căror rezultate evidenţiază existenţa unor relaţii de dependenţă şi de

interdependenţă între atributele interne şi atributele externe ale software-ului

fapt foarte important pentru domeniul măsurării şi estimării calităţii software-ului,

deoarece este mult mai uşor să avem măsurări directe ale atributelor interne

decât ale atributelor externe. [4]

**4. Caracteristicile de calitate**

Calitatea produselor software este definita în ca masura în care acestea satisfac cerintele utilizatorilor prin caracteristici tehnice, economice si psiho-sociale. Aceasta definitie se bazeaza pe doua concepte care, puse în legatura, dau masura calitatii unui produs software, si anume cerintele utilizatorilor si caracteristicile produsului. **[8]**

Caracteristicile de calitate ale unui produs software sunt proprietati ale produsului la care utilizatorii sunt sensibili. (de exemplu : usurinta de utilizare, fiabilitatea, timpul de raspuns, etc.) **[9]**

Caracteristicile si atributele de calitate formeaza un sistem dinamic complex; cresterea nivelului unei caracteristici conduce atat la cresterea nivelurilor unei caracteristici, cat si la scaderea nivelurilor pentru altele. Programatorii trebuie sa cunoasca modul in care interactioneaza caracteristicile si atributele de calitate, pentru a asigura realizarea echilibrului necesar incadrarii sistemului de programe intre limite de performante admise.

Urmatoarele criterii de calitate le sunt asociate caracteristicilor globale ale unui produs software:

· *Operationalitatea* – functionarea corecta intr-un mediu de utilizare

prestabilit, fara modificari si la nivelul performantelor specificate;

· *Utilitatea* – satisfacerea cerintelor impuse in conditii de eficienta

economica ridicata;

· *Transferabililitate* – folosirea fara modificari, sau cu modificari

minime, in alt mediu de utilizare decat cel prestabilit.

· *Adaptabilitatea* – folosirea sistemelor de programe intr-un mediu de

utilizare prestabilit, prin operarea de modificari cu un efort minim;

Toate aceste criterii sunt de calitate tehnica, mai putin utilitatea care este un criteriu de calitate functionala.

 Fiecare client are preferinte individuale,care pot fi satisfacute prin caracteristici de calitate diferite. Aceasta relatie se reflecta puternic în industria de software. Astfel,alaturi de unii factori ai caracteristicii de flexibilitate, se afirma tot mai mult posibilitatea de personalizare a produselor software, ca o caracteristica de calitate tot mai apreciata.Pentru satisfacerea cerintelor, este important ca relatia calitate-cumparator sa fie puternic reflectata nu numai în definirea calitatii, dar si în managementul si gestiunea acesteia, deoarece cumparatorul hotaraste,în final, ce este calitatea. **[8]**

 Costurile ridicate ce caracterizează activitatea de construire a sistemelor

de programe impun analiza corelaţiei dintre performanţele acestora şi efectele

pe care le are utilizarea lor curentă. Performantele se stabilesc prin luarea in

considerare a duratei in care se obtine solutia finala si a consumului de resurse

necesare, pentru rezolvarea integrala a problemelor curente. Nivelul performantei

este influentat de modul in care sistemlui de programe i s-au asigurat calitatea de

conceptie-proiectare, calitatea de executie, calitatea de conformitate, capacitatea de utilizare curenta si capacitatea de mentenanta.

Incercarile de standardizare a terminologiei referitoare la calitatea produselor software au condus la standardul ISO 9126 *(InformationTechnology-Software Product Qualit*y, Part 1: Quality Model, 1998). Standardul contine definitii in special pentru produsul final.

Sunt definite 6 caracteristici de calitate, impartite in 21 de subcaracteristici. **[9]**

1. **Functionalitatea**: realizarea scopului de baza pentru care a fost realizat produsul
	* **Oportunitatea:** prezenta unui set de functii adecate pentru tascuri specificate
	* **Precizia:** furnizarea unor rezultate sau efecte corecte sau agreate
	* **Interoperabilitatea:** capacitatea produsului de a interactiona cu sisteme specificate
	* **Securitatea**: capacitatea de a preveni accesul neautorizat, accidental sau deliberat, la programe sau date
	* **Conformitatea:** adeziunea la standarde, conventii, legi si protocoale
2. **Fiabilitatea:** capacitatea produsului de a-si mentine nivelul de performanta, in conditii definite, pentru o perioada de timp definita.
	* **Maturitatea:** atribut bazat pe frecventa caderilor datorate greselilor in software
	* **Toleranta la defecte (robustetea):** capacitatea de a-si mentine un nivel de perfomanta specificat in cazuri de caderi software sau intrari neasteptate
	* **Restabilirea dupa caderi:** capacitatea si efortul necesar pentru restabilirea nivelului de performanta, recuperarea datelor afectate, dupa posibile caderi
	* **Conformitatea**

**3) Utilizabilitatea**: efortul necesar pentru utilizarea sa de catre un set de utilizatori definit

* **Usurinta de intelegere:** efortul solicitat unui utilizator de a recunoaste conceptul logic si aplicabilitatea sa
* **Usurinta de invatare :** efortul solicitat unui utilizatorde a invata aplicatia, operarea, intrarile si iesirile
* **Operabilitatea:** usurinta de operare si de control de catre utilizatori
* **Puterea de atractie:** capacitatea produsului de a fi atragator pentru utilizatori
* **Conformitatea**

4) **Eficienta:** relatia intre nivelul de performanta al produsului si cantitatea de resurse utilizate, in conditii definite

* **Timp la executie:** viteza de raspuns, timpi de prelucrare, rata iesirilor la realizarea functiilor
* **Utilizarea resurselor:** cantitatea de resurse utilizate si durata utilizarii pentru realizarea functiilor sale
* **Conformitatea**

5) **Usurinta de intretinere**: efortul necesar pentru efectuarea modificarilor, inclusiv corectii, imbunatatiri sau adaptari ale produsului la schimbari ale mediului de functionare, a cerintelor si schimbarilor functionale

* **Usurinta de analiza:** efortul necesar pentru diagnoza defectelor, a cauzelor caderilor, pentru identificarea partilor care trebuie sa fie modificate
* **Usurinta de modificare:** efortul necesar pentru inlaturarea defectelor sau schimbari
* **Stabilitatea:** riscul efectelor neasteptate in urma modificarilor
* **Usurinta de testare:** efortul necesar pentru a valida produsul modificat
* **Conformitatea**

6) **Portabilitatea:** capacitatea produsului de a fi transferat de la o organizatie sau platforma software/hardware la o alta

* **Adaptabilitatea:** capacitatea de adaptare la diferite medii specificate
* **Usurinta de instalare:** efortul necesar pentru instalarea produsului intr-un mediu specificat
* **Co-existenta:** capacitatea de a co-exista cu alte produse independente in acelasi mediu
* Oportunitatea si efortul necesar pentru a folosi produsul in locul altui produs intr-un mediu particular
* **Conformitatea**

**Caracteristici de calitate software care afecteaza procesul de inginerie software**

* Stilul codului
* Reutilizabilitatea codului
* Modularitatea codului si independenta modulelor **[9]**

Intre caracteristicile de calitate exista o multime de relatii de subordonare,

interdependenta, ierarhizare, agregare sau de compozitie, iar complexitatea acestor

relatii face ca ansamblul caracteristicilor de calitate sa alcatuiasca un sistem, care

se bazeaza pe urmatoarele considerente:

· Conexiunea elementelor interne ale sistemului sa fie mai puternica decat legaturile sistemului cu mediul;

· Indiferent de complexitatea unui sistem, el este la randul sau un subsistem al altuia mai cuprinzator;

· Unitatea si complexitatea unui sistem presupune o anumita ordine in asezarea si functionarea elementelor sale;

· Structura reprezinta o caracteristica definitorie a fiecarui sistem in parte, ce trebuie privita sub forma de reunire a tuturor subsitemelor sau prin urmarirea diferitelor structuri componente;

· Orice subsitem este alcatuit dintr-o multime de bucle de reactie care se inchid pe anumite portiuni de proces, pe anumite portiuni de sistem, sau chiar la nivelul intregului sistem; la nivelul sistemului caracteristicilor de calitate acest lucru se traduce prin interdependentele si contradictiile existente intre caracteristici.

**Conceptul de fiabilitate** se defineste drept probabilitatea ca o functie data sa

fie executata in mod corect de un sistem pe o perioada de timp impusa. Fiablitatea

este privita ca masura increderii in conceptia si in capacitatea unui sistem de programe de a functiona corect in toate conditiile avute in vedere de la

inceput. Definitia se refera mai mult la fibilitatea procesului decat la cea a sistemului de programe. Este legata de probabilitatea ca o eroare din cadrul programelor componente ale sistemului sa fie activata de un set specific de date de intrare.

In comparatie cu fiabilitatea produselor hardware, fiabilitatea produselor

software exprima particularitatile industriei producatoare de software. Abordarile

cantitative ale fiabilitatii software exprima probabilitatea ca un sistem de programe

sa-si indeplineasca functiile cu anumite performante si fara erori intr-un interval de

timp si in conditii de exploatare date. Abordarile calitative privesc fiabilitatea ca o

capacitate a sistemului de programe.

Tehnicile de obtinere a tolerantei la defecte se aplica in special la componentele hardware si la sistemele de operare. Principala problema in programare este corectitudinea sistemelor de programe, acestea trebuie

sa indeplineasca specificatiile functionale stabilite.

Aprecierea fiabilitatii sistemelor de programe se face prin trei elemente:

* + *Disponibilitatea* – implica conservarea unei capacitati ridicate de prelucrare a unui sistem de programe, chiar daca apare o anomalie in textul sursa al componentelor acestuia;
	+ *Integritatea* – impune ca rezultatele sa fie exacte;
	+ *Reproductibilitatea prelucrarii* – urmareste obtinerea acelorasi rezultate, cu ajutorul unor seturi de date identice, la momente si conditii de executie diferite.

Functionalitatea este definita in standardul ca un set de atribute bazate pe existenta unui set de functii si pe proprietatile lor specificate. Functiile satisfac necesitatile stabilite sau implicite. Datorita acestui set de atribute care definesc functionalitatea se demonstreaza scopul caruia serveste sistemul de programe, pentru a indeplini cerintele. In acest timp, alte seturi arata momentul si modul in care sistemul de programe corespunde nevoilor utilizatorilor. Totalitatea trasaturilor si caracteristicilor unui produs sau serviciu bazat pe abilitatea sa de a satisface nevoile stabilite sau implicite este reprezentata de calitate.

Eficienta este definita ca o corelatie optima intre consumul de resurse

implicat de utilizarea produsului software si complexitatea problemei de rezolvat. Notiunea de iesire a produsului software este definita ca o relatie intre complexitatea datelor initiale si a rezultatelor finale. Aceasta notiune se refera la un set de atribute bazate pe relatia dintre nivelul de performanta al produsului software si cantitatea de resurse utilizate, in anumite conditii stabilite. Standardul ISO 8402 ofera un inteles mai larg termenului “*resurse consumate*" incluzand: memoria, timpul de executie, precizia, serviciile de operare, mentenanta sau suport de personal.

 Un produs software este mentenabil in masura in care permite o rapida si usoara actualizare astfel incat sa se utilizeze in continuare, in bune conditii. Sensul in care sunt privite aceste actualizari este mult mai restrans, deoarece ele se refera la operatiile realizate la nivelul sistemului de programe, al programelor si modulelor componente, secventelor de instructiuni ale instructiunilor. Nivelul mentenabilitatii se stabileste pe baza datelor obtinute in etapele de proiectare, testare si utilizare curenta.

Pentru unele sisteme de programe, varietatea mediilor de dezvoltare specifice fiecarui tip de calculatoare, precum si volumul modificarilor necesar adaptarii face ca in cele mai mutle cazuri sa se opteze pentru solutia rescrierii acestor programe. Cazurile in care se alege varianta modificarii sistemelor existente sunt cele in care acestea trebuie transferate intr-un timp scurt. In aceasta situatie s-a constatat insa ca nivelul fiabilitatii este scazut, iar costurile de exploatare sunt mari si nu se justifica din punct de vedere economic.

**5. Managementul calitatii software**

Principiile dezvoltarii software sunt legate de calitate, de management si de inginerie. Aceste principii se reflecta în calitatea produsului software, în termenele de realizare, precum si în costurile implicate.

Managementul calităţii software este o componentă a managementului general al calităţii. Aceasta reprezintă o subcomponentă a funcţiei generale de conducere, care determină şi implementează politica în domeniul calităţii şi deţine responsabilităţile pentru toate activităţile care concură la îndeplinirea obiectivelor privind calitatea. Sunt definite trei functii pentru managementul calitatii software si anume: **planificarea**, **asigurarea** si **imbunatatirea calitatii**.

**5.1 Planificarea calitatii**

Procesul de planificare urmăreşte dezvoltarea de produse şi procese care să corespundă cerinţelor clienţilor. Planificarea calităţii cuprinde activităţile prin care se stabilesc obiectivele şi cerinţele privind calitatea software precum şi resursele umane, financiare şi materiale necesare realizării obiectivelor. La nivel strategic, sunt formulate principiile de bază şi orientările generale ale organizaţiei în domeniul calităţii software, care se regăsesc în politica referitoare la domeniul calităţii. La nivel operaţional, planificarea externă are drept scop identificarea clienţilor şi a cerinţelor acestora, iar prin planificarea internă se urmăreşte transpunerea cerinţelor clienţilor în caracteristici ale sistemelor de programe şi dezvoltarea unor produse software adecvate.

Planificarea fiabilităţii software este o parte integrantă a planificării unui proiect software. Activitatea de planificare este concretizată în planul de fiabilitate. Acesta este corelat cu planul de dezvoltare a sistemelor de programe şi cu planul de calitate software.

Planificarea fiabilitatii software trebuie sa includa urmatoarele activitati:

* + Alocarea cerintelor de fiabilitate software;
	+ Definirea strategiei pentru obtinea fiabilitatii software;
	+ Definirea strategiei pentru evaluarea fiabilitatii software obtinuta.

Aceste activităţi sunt interconectate şi iterative. Ele presupun un process de inginerie software adecvat, insuficient pentru obţinerea fiabilităţii software. Este necesară definirea unui proces care să asigure folosirea corectă a metodelor şi tehnicilor corespunzătoare, în momente potrivite din ciclul de dezvoltare. Totuşi, procesul nu oferă predicţii şi nu demonstrează nivelul fiabilităţii software. Pentru a dovedi că sistemele de programe îşi ating cerinţele de fiabilitate, evaluarea fiabilităţii software trebuie planificată ca o activitate specifică.

Planul de fiabilitate software descrie:

· Cerintele de fiabilitate software derivate din cele sistem;

· Procesul de inginerie software;

· Tehnicile, metodele si instrumentele folosite pentru evaluarea fiabilitatii software obtinuta in cadrul fiecarei faze din ciclul de viata;

· Analiza riscului proiect pentru software;

· Procedurile pentru raportarea progresului inregistrat de fiabilitatea software;

· Documentele si datele care trebuie livrate;

· Structura organizationala care se refera la: stabilirea posturilor individuale implicate in dezvoltarea software; descrierea nivelurilor minime de competenta pentru posturile identificate; modalitatile de constientizare a managerilor fata de cerintele fiabilitatii software si de responsabilitatiile ce le revin; activitatile si cerintele de instruire.

**5.2 Asigurarea calităţii**

Asigurarea calităţii cuprinde activităţile preventive prin care se urmăreşte, în mod sistematic, corectitudinea şi eficienţa activităţilor de planificare şi control, pentru a garanta obţinerea unor produse software care să satisfacă cerinţele de calitate aşteptate de clienţi.

Această funcţie necesită atât o abordare internă realizată prin prisma conducerii companiei care dezvoltă software, cât şi una externă efectuată prin prisma clienţilor. Pentru asigurarea calităţii sistemelor de programe se întreprind şi acţiuni de control. Controlul calităţii software include activităţile de monitorizare a desfaşurării proceselor şi de evaluare a rezultatelor referitoare la calitatea software, în raport cu obiectivele stabilite, în scopul eliminării deficienţelor şi prevenirii apariţiei acestora.

În ceea ce priveşte fiabilitatea, ca una dintre principalele caracteristici de calitate software, ea a fost abordată indirect, prin standardele de asigurare a calităţii software în ansamblu - ISO 9000 şi ISO 9126 sau direct, prin standarde care vizează definirea terminologiei, precum şi măsurarea şi evaluarea fiabilităţii software pe întreg ciclul de viaţă.

Asigurarea fiabilităţii sistemelor de programe necesită implementarea unui set integrat de activităţi care cuprind estimarea, predicţia, alocarea şi modelarea fiabilităţii, precum şi activităţi de testare. Aceste activităţi permit o evaluare continuă atât a fiabilităţii globale a sistemului, cât şi a componentelor sale. Activităţile de estimare şi predicţie ale fiabilităţii sistemelor de programe cuprind patru categorii fundamentale:

* stabilirea scopurilor de fiabilitate pentru componentele sistemelor prin modelare şi alocare;
* estimarea fiabilităţii designului software prin procesul de predicţie a fiabilităţii;
* creşterea fiabilităţii sistemelor de programe prin testarea profilului operaţional;
* evaluarea fiabilităţii sistemelor de programe obţinute prin testarea calificării formale.

**5.3 Îmbunătăţirea calităţii**

Îmbunătăţirea calităţii cuprinde activităţile desfăşurate în fiecare etapă a ciclului de viaţă a unui sistem de programe, în scopul obţinerii unui nivel calitativ superior celui planificat, în condiţiile desfăşurării corespunzătoare a activităţilor de planificare, control şi asigurare a calităţii. Îmbunătăţirea fiabilităţii, ca urmare a asigurării calităţii, se face prin verificarea şi validarea sistemelor de programe.

Cea mai puternică reflectare a problemelor optimizării şi îmbunătăţirii calităţii la nivelul managementului, este apariţia şi dezvoltarea conceptului de management al calităţii totale. Cele trei componente de bază ale managementului calităţii totale sunt îmbunătăţirea continuă, satisfacerea clienţilor şi creşterea organizaţiei. Îmbunătăţirea calităţii are, la rândul ei, trei componente: standardizarea, îmbunătăţirea şi inovaţia.

Managementul calităţii software constituie responsabilitatea tuturor nivelurilor de conducere din companiile implicate în producţia de software, însă rolul cel mai important revine conducerii de vârf. În implementarea managementului calităţii software sunt implicaţi toţi angajaţii companiei.

Managerii sunt responsabili cu:

* + monitorizarea stării proiectelor software;
	+ predicţia datei de livrare a sistemelor;
	+ determinarea momentului în care se permite efectuarea de modificări în sistemele operaţionale;
	+ alegerea tehnicilor, metodelor şi tehnologiilor de inginerie software ce se aplică unui proiect;
	+ luarea deciziei de acceptare a livrării sistemelor de programe.

În cadrul companiilor mari de software se desemnează un inginer de fiabilitate care face parte din colectivul responsabil cu asigurarea calităţii. Lui îi revine rolul cheie în acest proces; ţine legătura cu personalul implicat în colectarea informaţiilor despre defectări; este responsabil cu eliminarea defectelor şi utilizarea instrumentelor care implementează modele de creştere şi predicţie a fiabilităţii.

**6. Concluzie**

Produsele software sunt componente esenţiale ale multor sisteme

, precum şi un nou factor major în comerţul de produse si servicii.

Îmbunătăţirea calităţii produselor software constituie una din principalele

forme în care se materializează capacitatea de creaţie. Valoarea economică a unui

produs software rezultă din modul în care calitatea acestuia este percepută de

clienţii sau utilizatorii finali. Din ce in ce mai mult, calitatea este percepută ca un

atribut critic al produsului software, deoarece lipsa calităţii determină insatisfacţia

utilizatorilor, pierderi financiare şi chiar pierderi de vieţi omeneşti.

În scopul elaborării produselor software având un nivel ridicat al calităţii şi

asigurării satisfacţiei clienţilor, producătorii de software au adoptat cele mai bune

practici şi standarde refritoare la managementul calităţii în ciclul de viaţă al

produselor software.

**7. Bibliografie**

[1] http://pecps.ici.ro/

[2]David Chappel-The three aspects of software quality

[4] Posibilităţi de evaluare a calităţii produselor software- Creţu Emil

# [5] 1062-1998  -  IEEE Recommended Practice for Software Acquisition

# [6] Gillies, A., Software Quality – Theory and Management, International Thomson Computer Press, London, UK, 1997)

# [7] Balog, A., Analiza statistică şi evaluarea calităţii software-ului, Editura Calipso 2000, Bucureşti, 1997

[8] http://revistaie.ase.ro/content/12/ivan.pdf

[9]http://andrei.clubcisco.ro/cursuri/3ip/curs/8.1\_Calitatea\_produselor\_software.doc