Universitatea Politehnică Bucureşti
Facultatea de Electronică, Telecomunicații şi Tehnologia Informației

Tema

Inginerie Software

Cerintele SW; procese pentru ingineria cerintelor; managementul de proiect SW

Studenti:
Mengheris Ioana – 442A
Banu Laura – 442A
Robitu Paul – 442A
Constantin Sebastian – 442A

2012-2013

CUPRINS

.

MENGHERIS IOANA 442A

CAPITOLUL 1: CERINTE SOFTWARE

Cerintele software reprezinta faza de analiza a problemei reale ce se doreste a fi rezolvata cu ajutorul ingineriei software. Exista probleme ale ingineriei software care fac ca cerintele software sa fie o parte foarte importanta a acesteia, si anume: intotdeauna costurile programelor au depasit previziunile initiale de cost, niciodata un produs software nu a satisfacut cerintele pe deplin, odata realizat produsul este greu de modificat, imbunatatit. In jurul cerintelor se desfasoara totul: cerintele trebuie culese de la clienti, cerintele trebuie documentate, trebuie gestionate, dezvoltate, testate. in fond, modelul creat prin specificatiile software este un model compus din cerinte care trebuie sa se transforme in produsul final.

Cerintele sunt descrieri (specificatii), intr-un limbaj accesibil tuturor celor implicati a ceea ce un sistem informatic trebuie sa poata acoperi, atât prin comportamente (behaviour) cât si prin atribute ale sale.

Institute of Electrical and Electronics Engineers (IEEE) a elaborat cea mai completa definitie a cerintei. Conform acestei organizatii, prin standardul 610.12-1990 IEEE Standard Glossary of Software Engineering Terminology, cerinta software este definita astfel:

" Cerinta software este:

1. o **conditie** sau **capabilitate** necesar a fi indeplinita de catre un sistem, pentru ca un utilizator sa poata rezolva o anumita problema sau sa atinga un anumit obiectiv;

2. o **conditie** sau **capabilitate** pe care un sistem trebuie sa o realizeze sau sa o posede pentru a satisface un contract, standard, specificatie sau alt document formal impus;

3. un **document** – reprezentarea unei conditii sau capabilitati, asa cum este descrisa la punctele 1 si 2 de mai sus. " [1][2]

Notiunea de "capabilitate", folosita de IEEE, reprezinta ceea ce un produs software ofera utilizatorilor, fie un anumit comportament fie un anumit atribut. De exemplu, "o functionalitate de genul „sistemul valideaza formatul datei atunci când utilizatorul inregistreaza factura in sistem” este un comportament al sistemului, in timp ce „pozitia unui buton pe ecran” este un atribut (un câmp dintr-o baza de date sau o proprietate a unui obiect poate corespunde unui atribut al unei entitati, in timp ce o metoda a unui obiect inseamna comportament.)"[7]

Astfel, definitia cerintei are trei parti:

Prima parte, reprezinta punctul de vedere al utilizatorului. Astfel, cerinta exista atat timp cat reprezinta o realizare a dorintei utilizatorului, care, desigur, daca nu poate fi exprimata, cerinta nu poate exista.

A doua parte a definitiei reprezinta punctul de vedere al dezvoltatorului. Cerinta devine ceea ce dezvoltatorul trebuie sa realizeze, pentru el fiind ceva impus. Dezvoltatorul trebuie sa ofere logica si sa valideze cerintele utilizatorului in ceva ce poate rezulta intr-un produs software.

Partea a treia a definitiei exprima faptul ca aceste cerinte, vazute atat din punctul de vedere al utilizatorului cat si al dezvoltatorului, trebuie documentate, altfel nu

exista. Evolutia cerintelor in timp, precum si cele doua puncte de vedere trebuie sa aiba un element comun.

Astfel, cerinta, vazuta atat din ambele puncte de vedere, este esentiala pentru rezolvarea unei probleme atat reale cat si software, utilizatorul trebuie sa o exprime iar dezvoltatorul sa o realizeze. Pentru ca aceasta sa fie rezolvata corect, cerinta trebuie exprimata clar si concis.

Cerinta astfel exprimata realiza infaptuirea corecta de la problema clientului la sistemul software, cerinta fiind un pas intermediar, pentru utilizator reprezinta solutia problemei sale iar pentru dezvoltator problema care trebuie solutionata.

[1]

Cerintele exprimate in limbaj accesibil, sub forma documentelor de specificatii software, acceptate de catre client si de catre dezvoltator, constituie referinta fundamentala pentru toti cei implicati in proiectele software: pentru manageri de proiect, in precizarea si supravegherea task-urilor sau pentru inginerii de testare, in efectuarea testelor.

Analistul software aduna, analizeaza si specifica cerinte, dar indiferent de modul in care specifica cerintele, in limbaj natural, in UML sau in orice alt limbaj, sub forma de use case-uri sau full text, indiferent de tipul lor, cerintele trebuie sa respecte anumite caracteristici care le fac sa fie cerinte adevarate, corect specificate si posibil de realizat, in parametrii bugetari si de calitate determinati.

Privita dintr-un punct de vedere cerinta se vede ca o problema a unui client, privita din celalalt punct de vedere este o solutie furnizata de un anumit sistem. De aceste puncte de vedere depind caracteristicile cerintei software:

- **Necesara:** O cerinta exista daca si numai daca este necesara. in caz contrar cerinta nu exista, nefiind corect exprimata sau reprezentand ceva ce e inutil ori redundabil. Indispensabilitatea cerintei este utila pentru managementul cerintelor, ce ne arata daca cerinta e folositoare proiectului.
- **Corecta:** O cerinta este corecta daca ofera solutie problemei. De exemplu, un use case format pentru realizarea unui anumit task este corect daca succesiunea pasilor enumerati duce la realizarea task-ului. Altfel cerinta descrisa in acest mod nu este corecta.

3. Completa: O cerinta este completa daca reprezinta o solutie exhaustiva pentru rezolvarea in intregime a problemei. Desi cazurile de incompletitudine sunt greu de descoperit sau determinat, organizarea de review-uri formale si teste efectutate de echipa tehnica ce se ocupa de proiect, da rezultate.

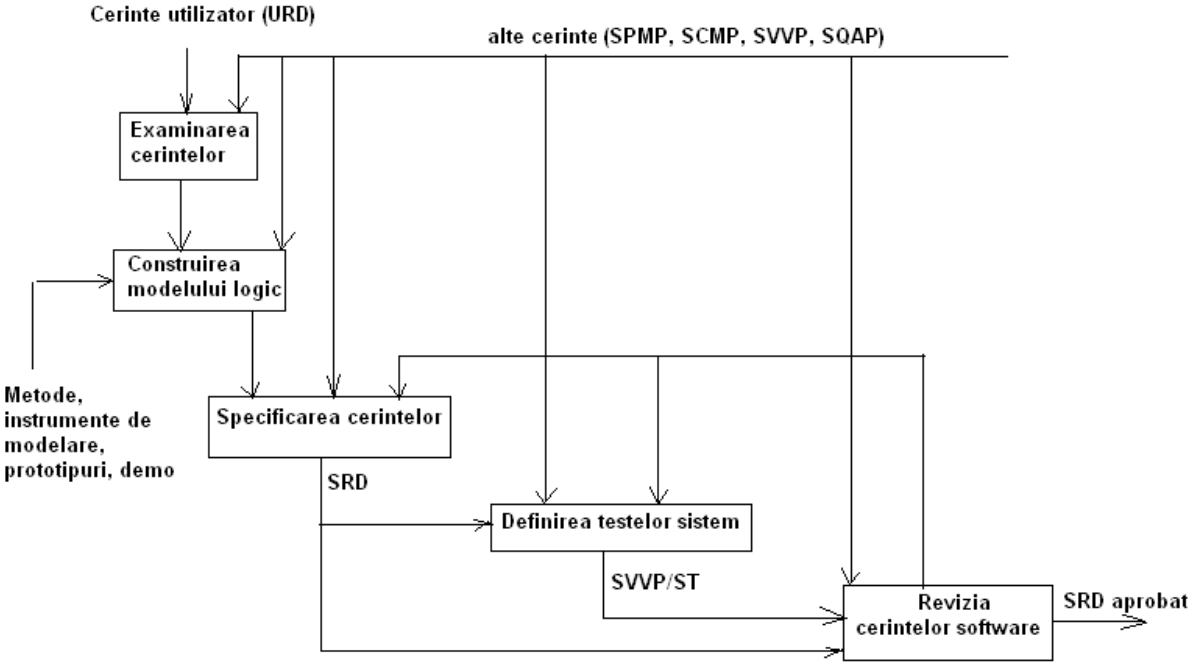
- **Consistenta:** O cerinta este consistenta daca nu intra in contradictie cu alta cerinta.
- **Verificabila:** O cerinta este verificabila daca permite validarea solutionarii ei prin testare.

6. Clara (fara ambiguitati): O cerinta este clar formulata atunci când poate fi interpretata intr-un singur fel. Daca lasa loc interpretarilor multiple atunci cerinta este ambigua. Se fac review-uri ale specificatiei pentru inlaturarea amiguitatii, necesare deoarece specificatiile vor fi folosite pentru crearea planurilor de teste si a manualului de utilizare.
7. Trasabila: O cerinta este trasabila daca are posibilitatea de a genera traseul pe care cerinta s-a infaptuit, pornind de la exprimarea problemei de catre client. Aceasta caracteristica a cerintei asigura justificarea existentei cerintei si refacerea traseului prin care s-a infaptuit aceasta, atunci cand apar neclaritati privind sursa sau logica ei.

Cerintele software au avantajul ca stau la baza contractului dintre clienti si furnizori, reduc efortul de dezvoltare, stau la baza estimarii costului si planificarii, permit planificarea testelor de validare si verificare, usureaza transferul produsului la noi utilizatori sau pe platforme noi, servesc ca baza pentru viitoarele imbunatatiri sau modificari ale produsului.

Definirea cerintelor software este o raspundere ce revine dezvoltatorului. Deasemenea poate fi o raspundere asumata si de catre: utilizatori, ingineri de sistem, ingineri hardware si personal de operare.

Activitatile si fluxul documentelor in etapa de definire a cerintelor software



[3]

Exista 2 tipuri de cerinte software:

- Functionale
- Ne-functionale

Cerinte functionale

- Descriu functiile pe care trebuie sa le efectueze sistemul, intr-o maniera separata de implementare.
- Exprima transformarile ce trebuiesc aplicate intrarilor pentru a obtine iesirile dorite

Cerinte ne-functionale

- Sunt anexate cerintelor functionale
- Pot proveni din constrangerile incluse in Specificatia Cerintelor Utilizatorilor
- Cerinte de: performanta, interfata, de operare, de verificare, de portabilitate, de intretinere, de fiabilitate, s.a.

Dintre aceste cerinte, se remarca urmatoarele:

"Cerinte de performanta

- Valori numerice atasate unor parametri masurabili cum ar fi: viteza, capacitatea, precizia, frecventa.
- Pot fi reprezentate ca un domeniu de valori:
- Valoare acceptabila (nivel minim de performanta)
- Valoare nominala
- Valoare ideala

Cerinte de interfata

- Protocoalele de comunicatie de utilizat
- Fluxul de date prin interfata
- Cand au loc schimburi de date prin interfata, etc

Cerinte operationale

1. Modul de comunicare cu operatorii umani: aspecte fizice si ergonomice ale interfetei utilizator:

    1. Descrierea dialogului,
    2. Aspectul ecranului in timpul dialogului
    3. Stilul limbajului de comenzi

Cerinte impuse resurselor fizice

- Puterea de prelucrare
- Memoria necesara

- Spatiul pe disc

Cerinte de verificare

1. Efectuarea unor simulari
2. Cerinte impuse mediului de testare
3. Posibilitati de diagnosticare

Cerinte pentru testarea de acceptare

Cerinte de portabilitate

- "Poate rula pe calculatorul X fara modificara codului sau ..modificand cel mult 2% din codul sursa”
- "Nici o parte a software-ului nu trebuie scrisa in assembler."

Cerinte de intretinere

"Timpul de reparare a unei erori nu va depasi niciodata o saptamana"

Cerinte de fiabilitate

- Frecventa acceptata a caderilor software, in functie de categoria caderii: Severa, avertizare, informare.

- Exprimate folosind parametrii Mean Time Between Failure (MTBF) si Mean Time To Repair (MTTR). Exemple:

  - "Timpul minim intre doua caderi severe va fi mai mare de o luna”

Cerinte de securitate

Cum sa fie securizat sistemul impotriva pericolelor:

- Erori utilizator (distrugerea accidentala a software-ului sau datelor)
- Hazarduri fizice (foc)
- Access ne-autorizat
- Virusi

Cerinte de siguranta

Protectia impotriva distrugerilor cauzate de caderile software

Alte cerinte de calitate

- Utilizarea anumitor standarde de produs sau de proces

- Utilizarea de personal extern pentru asigurarea calitatii."

Aceste cerinte se gasesc la [3].

Cerintele pot fi incorporate intr-un document al cerintelor software (SRD), care **c**ontine:

- O descriere generala a scopului produsului
- O descriere a mediului de operare: echipamentele (hardware) si sistemul de operare
- Mediul de dezvoltare care urmeaza sa fie utilizat
- Daca produsul este un sistem independenmt sau o parte a unui sistem mai mare sau inlocuieste un alt sistem. Caracteristicile esentiale acestui sistem
- O descriere a modelului logic al sistemului.
- Lista cerintelor functionale
- Lista cerintelor nefunctionale
- Modelul logic

The Software Requirements Document - sablon definit in standardele ESA

a.	Abstract	
b.	Table of Contents	
c.	Document Status Sheet	Status sheet for configuration control.
d.	Document Change Records since previous issue	A list of document changes.
1. Introduction		
1.1	Purpose	The purpose of this particular SRD and its intended readership.
1.2	Scope	Scope of the software. Identifies the product by name, explains what the software will do.
1.3	List of definitions	The definitions of all used terms, acronyms and abbreviations.
1.4	List of references	All applicable documents.
1.5	Overview	Short description of the rest of the SRD and how it is organized.
2. General description		
2.1	Relation to	The context of this project in relation to other current

	current projects	projects.
2.2	Relation to predecessor and successor projects	The context of this project in relation to past and future projects.
2.3	Function and purpose	A general overview of the function and purpose of the product.
2.4	Environment	Hardware and operating system of target system and development system. Who will use the system (see user roles in URD).
2.5	Relation to other systems	Is the product an independent system, part of a larger system, replacing another system? The essential characteristics of these other systems.
2.6	General constraints	Reasons why constraints exist: background information and justification (analogous to URD).
2.7	Model description	A description of the logical model.
3. Specific requirements		
3.1	Functional requirements	A list of all functional requirements. Note the general remarks about requirements.
3.2 – 3.14	Non-functional requirements	A list of all non-functional requirements. These requirements are linked to functional requirements. Note the general remarks about requirements. Each category of non-functional requirements has its own subsection.
4. Requirements traceability matrix		A table showing how each user requirement of the URD is linked to software requirements in the SRD.

[3]

Revizia cerintelor software **(Software Requirements Review) e**ste o revizie tehnica, poate fi: interna, la care asista conducatorul proiectului, utilizatori si dezvoltatorul; sau externa la care asista departamentul de calitate. Prin aceasta se are in vedere verificarea claritatii cerintelelor, consistentea lor, completitudinea si detalierea lor suficienta pentru proiectarea produsului software. Aproximativ 20-25% din timpul total al proiectului trebuie utilizat in scopul definirii cerintelor. Se utilizeaza 5% din timpul proiectului pentru actualizarea cerintelor dupa ce a inceput proiectarea. Aceasta documentatia poate fi minima daca produsul va fi folosit pentru o perioada de timp scurta sau daca este folosit de putini utilizatori. Se aloca mai mult timp cerintelor complicate, in defavoarea celor mai simple, usor inteligibile. SRD trebuie actualizat ori de cate ori se fac modificari.

CAPITOULUL 2: INTRODUCERE IN MANAGEMENTUL PROIECTELOR

"Ingineria software (din engleza: software engineering) este un domeniu ce implica proiectarea, crearea și intreținerea de software aplicând tehnologii și practici din informatica (știința calculatoarelor), managementul proiectelor, inginerie, proiectarea interfețelor și a altor domenii." [4] http://ro.wikipedia.org/wiki/Inginerie_software.

"Ingineria software analizeaza toate aspectele privind productia software de la primele etape ale specificatiilor de sistem si pâna la mentenanta sistemului dupa ce a fost dat in functiune."[6] Managementul proiectelor software urmareste gestionarea si evaluarea ingineriei software.

Intregul proces de management al proiectul consta in definirea necesitatii procurarii sau dezvoltarii acelui proiect si reprezinta ciclul de viata al proiectului/produsului. Ciclul de viata reprezinta totalitatea etapelor parcurse in realizarea unui produs software.

Managementul proiectelor poate fi structurat pe nivele:

Managementul de **TOP**, daca este specificat corect trebuie sa aiba cea mai mare stabilitate structurala.

Managementul **TACTIC** formuleaza politicile necesare pentru indeplinirea obiectivelor managementului de **TOP**. Daca este necesar, stabilitatea structurala a acestor politici poate fi schimbata.

Managementul **OPERATIV** face tot ceea ce este necesar pentru a operationaliza politicile stabilite de managementul **TACTIC**. Nivelul operativ al managementului este supus, cu prioritate, schimbarilor atunci cand se pune problema adaptarii sistemului condus la conditii de performanta noi.

Un management slab poate periclita usor sansele de succes ale proiectului realizat de o anumita echipa calificata profesional; un management bun poate gestiona eventualele probleme datorate nivelului profesional defectuos al unora dintre membrii echipei de proiectare.

Avand in vedere managementul proiectului, putem caracteriza desfasurarea unui proiect prin urmatoarele etape:

- Inaintea inceperii proiectului se observa un entuziasm general, o insufletire a

echipei

- Atunci cand proiectul nu poate fi predat la timp, din diverse motive, se observa un punct de criza
- Spre sfarsitul proiectului se observa : un volum de munca impresionabil (24h/24h), resurse suplimentare, tensiune si relatii incordate intre membrii echipei de proiectare

Aceste etape sunt parcurse si in marile companii de soft atunci cand se lucreaza la proiecte, de o anumita echipa, datorita neptutintei de planificare si gestionare a resurselor (timp, oameni, documentatie, utilitare, cunostinte, etc) .

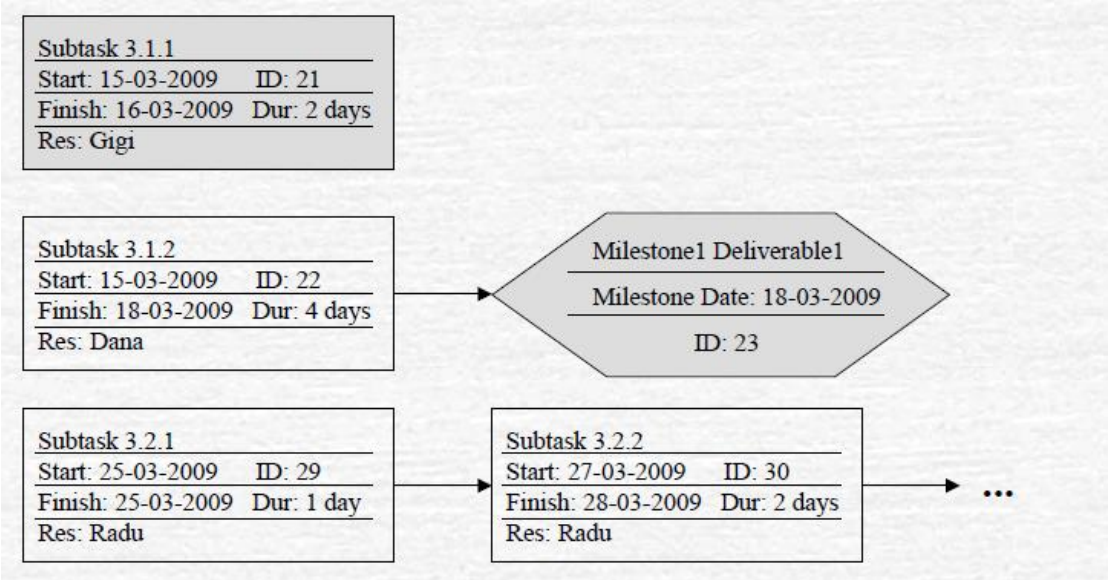
Este necesar ca managementul proiectelor sa aiba in vedere urmatoarele activitati:

4. "Managementul proiectului cu uneltele de management specifice
5. Managementul personalului cu managementul echipelor implicate in dezvoltarea unui proiect software
6. Estimarea costurilor; estimarea de preturi, costuri si a productivitatii procesului de dezvoltare software
7. Managementul calitatii cu standarde, masuratori si metrici referitoare la calitatea procesului software" [5]

Managementul proiectului se refera la repartizarea si controlul bugetului, timpului si personalului, dupa motto: "Ce nu poti planifica, nu poti nici realiza". Managementul eficient prevede folosirea unor instrumente pentru planificarea si controlul activitatilor, pentru estimarea costurilor, pentru colectarea metricilor. Instrumentele actuale folosite in ingineria software sunt incluse frecvent in medii integrate pentru management care uneste managementul cu planificarea strategica, modelarea afacerii, gestiunea portofoliului, gestiunea documentelor, gestiunea fluxului de productie etc.

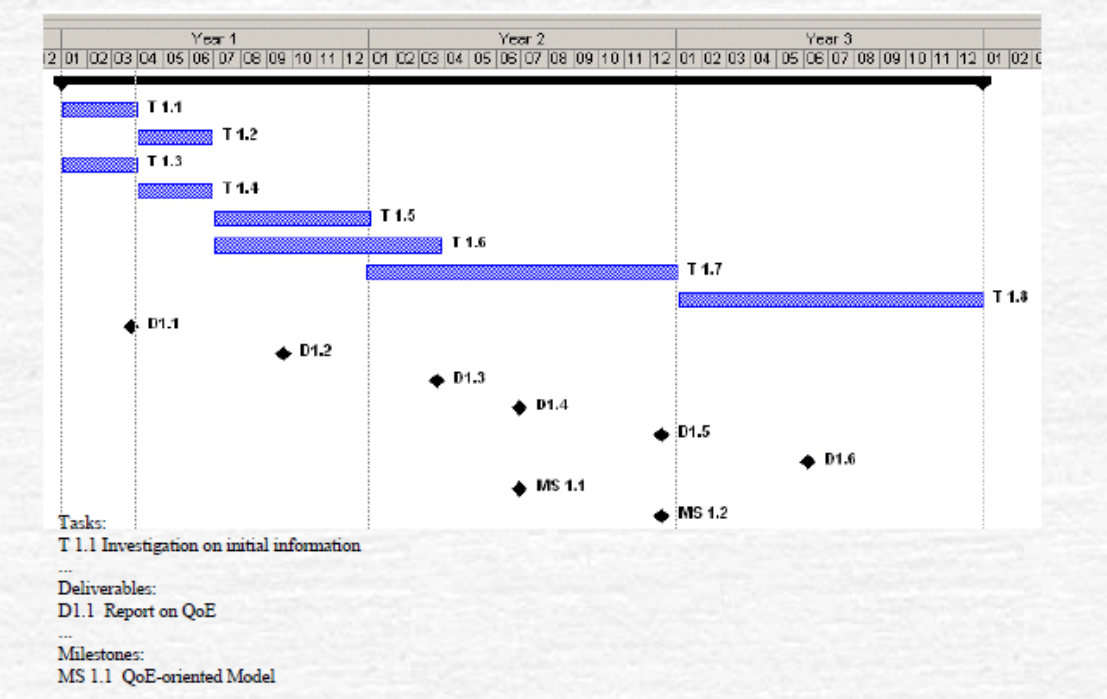
Planificarea si controlul proiectului sunt etape necesare in managementul proiectului. Exista grafice ale retelelor de activitati care recunoaste evenimentele care trebuie sa apara inainte ca o activitate sa inceapa, mentioneaza durata si momentul de incheiere al fiecarei activitati, distribuie personalul si celelalte resurse intre toate activitatile. Pentru realizarea acestor grafice avem retele CPM (Critical Path Method), diagrame Gantt. Aceste grafice sunt generate de catre instrumentele de management al proiectului procurand informatii despre proiect. Instrumentele de management sunt capabile sa adapteze sau sa modifice automat planificarea ori de cate ori apar schimbari la nivelul unei activitati sau resurse.

Exemplu de retea CPM:



[5]

Exemplu de diagrama Gantt:



[5]

In management, metricile se refera la masurarea procesului de dezvoltare software prin adunarea de informatii care sa foloseasca la planificarea viitoare. Proiectarea arhitecturala trebuie sustinuta prin metrici care sa garanteze comprehensivitatea, mentenabilitatea si scalabilitatea sistemului. Uneltele de colectare a metricilor trebuie sa fie usor de folosit. Pentru identificarea dependentelor intre o entitate (clasa) si restul entitatilor din program se pot folosi diverse metode de analiza, cum ar fi cele de tipul what-if.

Managementul personalului se ocupa de cea mai importanta resursa, si

anume oamenii ce formeaza echipa profesionala desemnata proiectului. Principala preocupare a unui manager sunt oamenii ce formeaza aceasta echipa. Nu poate exista un management de succes daca nevoile echipei nu sunt intelese. Un management slab va duce la esuarea proiectului.

Factorii care influenteaza managementul personalului sunt:

8. Consistenta - membrii echipei trebuie tratati toti in acelasi mod, fara favoruri si discriminari
9. Respectul - diferentele dintre membrii echipei trebuiesc respectate, deoarece fiecare are anumite aptitudini sau talente diferite
10. Includerea - membrii echipei trebuie implicati in acelasi mod iar punctele lor de vedere apreciate
11. Onestitatea - ceea ce merge bine sau rau in desfasurarea proiectului trebuie recunoscut de catre managerul proiectului in fata membrilor echipei

Managementul grupurilor este de asemenea important, deoarece cea mai mare parte a ingineriei software este o activitate de grup, de exemplu planificarea proiectelor.

Estimarea si planificarea proiectului sunt activitati de management ce se intercaleaza. Estimarile sunt utilizate pentru determinarea costurile producerii proiectului. Intre costul de dezvoltare si pretul cerut clientului exista o relatie complicata deoarece trebuie luate in considerare diverse cauze organizationale, economice, politice si de afaceri.

Managementul calitatii are in vedere garantarea atingerii unui anumit nivel de calitate al proiectului. Necesita definirea unor standarde de calitate si proceduri care apoi sunt urmarite spre a fi respectate. Are in vedere cresterea unei "culturi a calitatii", in care "calitatea este vazuta ca responsabilitatea fiecaruia"[5]. Managementul calitatii este foarte important pentru proiectele software. Documentatia despre calitate este o consemnare a proceselor efectuate pentru realizarea proiectelor software si asigura continuarea proiectelor daca se schimba echipa profesionala ce se ocupa cu dezvoltarea lor.

Un plan pentru calitate determina calitatile necesare ale proiectului si cum sunt evaluate ulterior. Defineste cele mai semnificative atribute de calitate, pe baza carora formeaza procesul de evaluare a calitatii. Trebuie sa prevada ce standarde sunt aplicate si unde este necesar sa se formeze noi standarde.

Structura unui astfel de plan este:

- Scurta descriere a proiectului
- Planuri de proiect
- Descriere proiect
- Tinte de calitate
- Riscuri si managementul riscului

BANU LAURA – 442A

CAPITOULUL 3: MANAGEMENTUL TIMPULUI

Managementul timpului este procesul de planificare si exercitare a controlului constient asupra perioadei de timp consumate pentru anumite activitati, pentru a creste eficienta, eficacitatea sau productivitatea. Un sistem de management al timpului este un ansamblu de procese, instrumente, tehnici si metode. Pentru a putea fi respectati termenii de executie prevazuti in contracte, managementul timpului a devenit indispensabil in dezvoltarea oricarui tip de proiect.

Cei mai importanti pasi referitori la managementul timpului sunt:

1. Crearea unui mediu propice pentru a creste eficacitatea.
2. Definirea activitatilor - trebuie sa avem in vedere ce activitati sunt necesare pentru finalizarea proiectului.
3. Stabilirea prioritatilor si, totodata, identificarea acelor activitati care se pot desfasura in mod independent fata de celelalte sau dimpotriva. Nu se poate initia o activitate care are nevoie de rezultatele unei alte activitati.
4. Aproximarea timpului necesar desfasurarii fiecarei activitati in parte.
5. In functie de resursele disponibile(date, personal, echipamente etc) se realizeaza un program potrivit pentru proiect.
6. Controlul programului – in momentul in care apar modificari in proiect, acestea trebuiesc controlate astfel incat durata totala aproximata initial a proiectului sa nu fie depasita. Acest pas este foarte important deoarece depasirea timpului impus printr-un contract poate duce la pierderi enorme de ordin financiar sau poate duce la anularea proiectului.

Crearea unui mediu pentru a creste eficacitatea

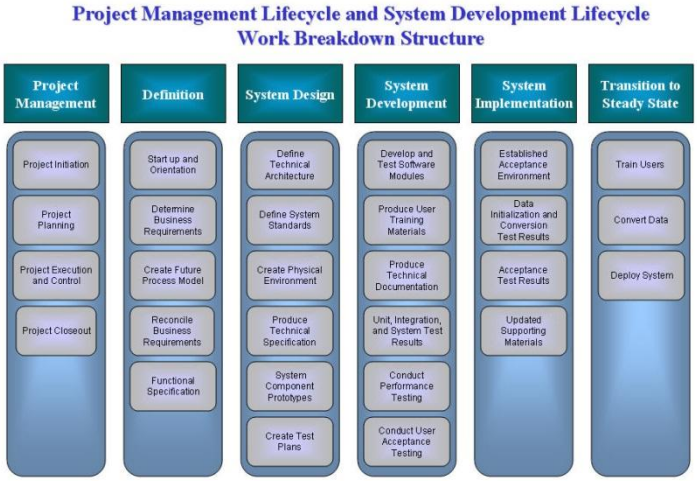
In literatura de specialitate sunt prezentate cateva sfaturi pentru asigurarea unui mediu propice pentru a creste eficacitatea:

- Organizeaza-te! (“Get organized!”)- se refera la triajul actelor si al task-urilor;
- Ai grija de timpul tau. (“Protect your time!”)
- Incearca sa reduci activitatile ce consuma mult timp.("Recover from Bad Time Habits")

Definirea activitatilor

Pentru definirea eficienta a activitatilor se face descompunerea proiectului in subproiecte (engl.“ Work Breakdown Structure”(WBS). WBS realizeaza descompunerea ierarhica a proiectului in: faze, rezultate si pachete de activitati (“ work packages”). Este o structura de tip arbore, care arata subdiviziuni ale activitatilor necesare pentru a atinge un tel (finalitate); de exemplu: un proiect, un contract. In cazul unui proiect sau contract, ordinea in WBS este inversa, se porneste de la rezultat adica de la scopul final, dupa care se efectueaza

descompunerea succesiva a partilor componente pana cand acestea pot fi controlate atat ca marime, cat si ca durata sau responsabilitate; in acest mod sunt atinsi toti pasii necesari indeplinirii obiectivului.



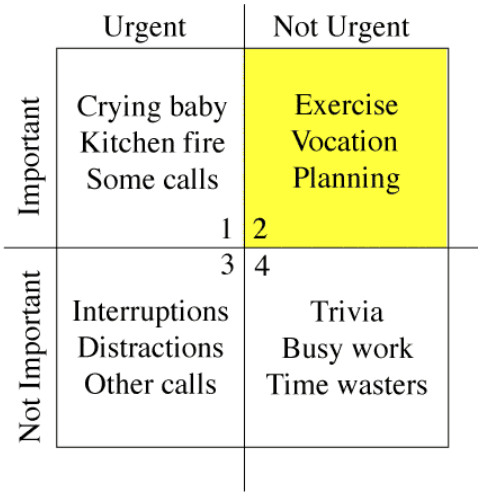
Exemplude descompunere in sub-proiecte [12]

Stabilirea prioritatilor si ordonarea activitatilor

Se poate observa o stransa legatura dintre strategiile managementului de timp si adoptarea unor strategii pentru atingerea scopurilor personale.

In management este utilizata o tehnica ce imparte activitatile ce trebuiesc desfasurate in mai multe grupuri, si anume : A, B, C. Acestea pun in evidenta importanta sarcinilor, atat ca scop cat sic a timp:

- A - Sarcini care sunt percepute ca fiind urgente si importante
- B - Sarcini care sunt importante, dar nu sunt urgente
- C - Sarcini care nu sunt nici urgente, nici importante.



Tabelul Eisenhower [20]

Metoda Eisenhower evalueaza task-urile folosind criteriile: important/neimportant sau urgent/nu e urgent . Presedintele Dwight D.

Eisenhower a spus: "*What is important is seldom urgent and what is urgent is seldom important*."

Chiar daca o activitate este prioritara trebuie sa ne asiguram ca aceasta are intrarile necesare pentru lucrul efectiv. Totodata, fiecare iesire a unei activitati trebuie sa fie ori un rezultat final al proiectului, ori sa fie utilizata de o alta activitate.
In functie de modul in care depinde o activitate de o alta, putem distinge: dependentele care nu provin din proiect (dependente externe), dependente definite de management(dependente care au un anumit scop in sensul managementului) si dependente impuse de natura proiectului(ele sunt indispensabile, deci sunt dependente obligatorii).

In acest context putem deosebi mai multe tipuri de relatii de dependenta: relatii in care activitatile consecutive se pot termina doar in acelasi timp (activitatea succesoare nu se poate finalize pana cand nu este terminata activitatea predecesoare) si se numesc relatii de tipul sfarsit-sfarsit, relatii in care activitatile consecutive nu pot incepe decat in acelasi timp (doar in momentul in care activitatea predecesoare incepe atunci poate incepe si cea succesoare)- acestea sunt denumite relatii de tip inceput-inceput, relatii in care o activitate ce urmeaza unei alte activitati se poate termina doar inaintea inceperii celei de-a doua activitati (inceput-sfarsit) sau relatii in care o activitatea poate incepe dupa terminarea activitatii ce o precede (sfarsit-inceput).

Pentru aproximarea timpului necesar desfasurarii fiecarei activitati in parte sunt folositi diferiti algoritmi: metoda drumului critic (CPM), metoda PERT ("Program Evaluation and Review Technique"), simularea Monte Carlo.
Trebuie precizat faptul ca toata aceasta activitate de planificare are un scop bine stabilit.

Realizarea unui program potrivit pentru proiect este o forma de a pune in evidenta un angajament intre fiecare persoana dintr-o echipa sau organizatie, confirmand ceea ce persoana trebuie sa realizeze intr-o perioada de timp.
Un alt scop al planificarii este acela de a incuraja personalul ce lucreaza la un proiect sa isi vada eforturile ca parte a unui intreg si sa se straduiasca ca partile proprii sa se imbine cu ale altora. Atat timp cat exista un program, un grafic de lucru, se pot privi realist cerintele si astfel se constientizeaza ce parte a proiectului poate fi dusa la bun sfarsit in perioada de timp stabilita. Chiar daca in program apar modificari, angajamentele anterioare se vor mentine.
Deasemenea, este important ca planificarea sa asigure echipei o unealta care permite observarea evolutiei si totodata, descompunerea volumului de lucru in bucati gestionabile.
Planificarea este cu atat mai importanta cu cat proiectul presupune un volum de lucru mai mare. Totusi o buna planificare, nu asigura rezolvarea tuturor problemelor pe care le au proiectele. Un plan nu poate preveni deficientele proiectului legate de o conducere slaba, de obiective neclare, de proiectarea sau practicile ingineresti gresite.

Managementul timpului unui proiect se refera la un plan de baza. WBS furnizeaza activitatile individuale planificate, iar timpul necesar desfasurarii activitatilor este determinat prin procesul de estimare.

CAPITOLUL 4 : ESTIMAREA COSTURILOR SOFTWARE

Estimarea costului, in cele mai multe domenii este usor de realizat. Insa in cazul unei aplicatii software lucrurile se complica si se bazeaza mai mult pe aproximari. In acest sens exista o serie de metode care permit aproximarea costului total pentru un proiect software pe baza unui numar restrains de generatori relevanti de costuri.

Problema estimarii costului poate fi rezolvata prin determinarea ecuatiilor diversilor algoritmi; pentru aceasta exista diferite abordari.

Pentru a descoperi modele algoritmice de estimare a costului putem pleca de la analiza datelor unui proiect real, dar si pe un suport teoretic. Asadar, o organizatie poate colecta date de la alte organizatii despre un numar de sisteme software care au fost dezvoltate.

Tot in acest context, ecuatia poate fi calculata pe baza rezultatelor experimentale. In general, este de dorit varierea unui singur parametru pentru a putea observa comportamentul celorlati parametri.

Rezultatele astfel obtinute reprezinta o medie, o extrapolare a rezultatelor obtinute anterior, de aceea este necesara atentie sporita in momentul aplicarii acestor rezultate.

Uneori calculele pot fi flexibile, cu un grad mare de generalitate si se pot mula pe mai multe proiecte. Acest aspect poate fi benefic, putand aplica modelul pe o multitudine de proiecte, dar prezinta dezavantajul modificarii mai multor parametri specifici pentru fiecare proiect in parte. Trebuie facuta o evaluare a timpului necesar pentru crearea unui nou model, dar si pentru adaptarea calculelelor existente la cerintele proiectului; aceasta din urma poate fi mult mai costisitoare din punctul de vedere al timpului.

Desi sunt obtinute formule de calcul pe baza modelelor existente, acestea nu rezolva problema estimarii costului deoarece fiecare model are particularitatile sale si necesita o adaptare la mediul in care va fi folosita. Din aceasta cauza, fiecare etapa a proiectului trebuie analizata pentru a se extrage date. Aceste date sunt folosite in metode statistice pentru a calibra diversi parametri ai modelului.

In ciuda diferentelor existente, modele de estimare a costului au si multe caracteristici comune. Astfel au fost identificate mai multe strategii de crestere a productivitatii software:

- Motivarea personalului sa lucreze la capacitate maxima: In momentul in care angajatii sunt competenti si motivati (prin conditii bune de lucru, prime, cursuri de perfectionare) productivitatea creste.
- Reducerea codului folosit: Marimea sistemului este un factor important al efortului si costului. Reduceri semnificative sunt inregistrate in momentul in

care se incearca reutilizarea componentelor si utilizarea de limbaje de nivel inalt.

- Evitarea refacerii componentelor dezvoltate anterior: Daca sunt folosite prototipurile se evita rescrierea a ceea ce s-a produs deja.

Modele algoritmice

Modelul COCOMO (Constructive COst Model) este un model algoritmic de estimare a costului dezvoltat Barry W Boehm. Acesta a fost publicat in 1981 in "Software Engineering Economics" ca model pentru estimarea efortului, costului si planificare proiectelor software.

COCOMO de baza calculeaza efortul (si costul) de dezvoltare software ca functie a marimii programului.

COCOMO se aplica la trei clase de proiecte software:

- Proiecte organice – In cazul acestor proiecte cerintele nu sunt stricte, iar echipele destinate realizarii lor sunt cu experienta in domeniu;
- Proiecte dedicate – In cazul acestor tipuri de proiecte regulile sunt definite clar, nu se admit abateri de la aceste reguli.
- Proiecte semi-detasate- Acestea sunt un mix intre proiectele dedicate si proiectele organice: cerintele fac parte din ambele categorii, sunt prezente atat cerinte stricte, cat si cerinte flexibile. Personalul desemnat are experienta medie in domeniu.

Ecuatiile COCOMO-ului de baza sunt de forma:

Efortul depus (E)= $a_b(KLOC)^{b}_{b}$ [om-luni]

Timpul de dezvoltare(D)= $c_b(\text{Efort depus})^{d}_{b}$ [luni]

Personal necesar (P)=Efort Depus/Timp de dezvoltare [numar]

unde, KLOC este numarul estimat de linii a codul pentru proiect. Coeficientii a_b, b_b, c_b si d_b sunt dati in urmatorul tabel:

Proiect software	a_b	b_b	c_b	d_b
Organic	2.4	1.05	2.5	0.38
Semi-detasat	3	1.12	2.5	0.35
Dedicate	3.6	1.2	2.5	0.32

[20] -tradus

O metoda care incearca sa evite problemele determinare de estimare dimensiunii codului este analiza punctelor functionale (AFC). In cadrul acestei metode se numara diferite structuri de date utilizate, pornindu-se de la presupunerea ca acest numar este un bun indicator pentru dimensiunea proiectului. Se urmareste ca structura sa fie relevanta pentru aplicatia respectiva. In cazul proiectelor in care algoritmul joaca un rol important, metoda AFC nu este indicata. Sistemul indeplineste urmatoarele functii:

- Functii referitoare la date:

- fisiere interne logice
  -fisiere externe de interfata
- Functii tranzactionale:
  - intrari externe
  -iesiri externe
  -cereri externe

In cazul in care utilizatorii doresc sa utilizeze date fisiere interne logice (FIL):, acestia trebuie sa si intretina acele date.
Exista cazul in care datele nu trebuiesc intretinute de utilizator deoarece acestea sunt situate intr-un alt sistem, cum este cazul fisierelor externe de interfata (FEI)
Prin intermediul intrarilor externe (IE) se permite utilizatorului sa intretina fisierele interne logice prin operatii de adaugare, modificare si stergere.
Daca sunt folosite iesirile externe (EE) utilizatorul este autorizat sa produca date de iesire.
Daca utilizatorul doreste sa afiseze sau sa selecteze date din fisiere, el trebuie sa introduca informatii referitoare la selectie pentru a putea gasi datele respective; acestea se fac prin intermediul functiilor de cereri externe (CE).
Numarul de puncte funtionale neajustate este:
PFN=10*FIL+7*FEI+4*IE+5*EE+4*CE [32]
Pentru o mai buna estimare sunt considerate alte 14 caracteristici care influenteaza dezvoltarea aplicatiilor. Pe o scara de la 0 la 5 se masoara modul in care fiecare caracteristica influenteaza sau nu dezvoltarea aplicatiilor. Gradul de influentare (GI) este suma acestor puncte pentru cele 14 caracteristici.
Putem calcula FCT (factol de complexitate tehnica):
FCT=0,65+0.01*GI [32]
Punctele functionale ajustate :
PF=PFN*FCT [32]
In cazul aceste metode, productivitatea este un rezultat natural, punctele functionale nu depind de tehnologie si in acest mod, pot fi utilizate pentru a compara productivitatea pe platform diferite si cu instrumente diferite.
Asadar, estimarea costului unui produs software nu este un process usor de realizat, dar odata gasita metoda potrivita proiectului se pot aproxima diversi parametrii ai unor formule care duc la rezultate apropiate de realitate.

CAPITOLUL 5 : MANAGEMENTUL RESURSELOR UMANE

Managementul resurselor umane se refera la multimea de activitati orientate catre dezvoltarea, motivarea, asigurarea si mentinerea resurselor umane in cadrul organizatiei in vederea realizarii eficiente a obiectivelor acesteia si ,totodata, satisfacerii nevoilor angajatilor.

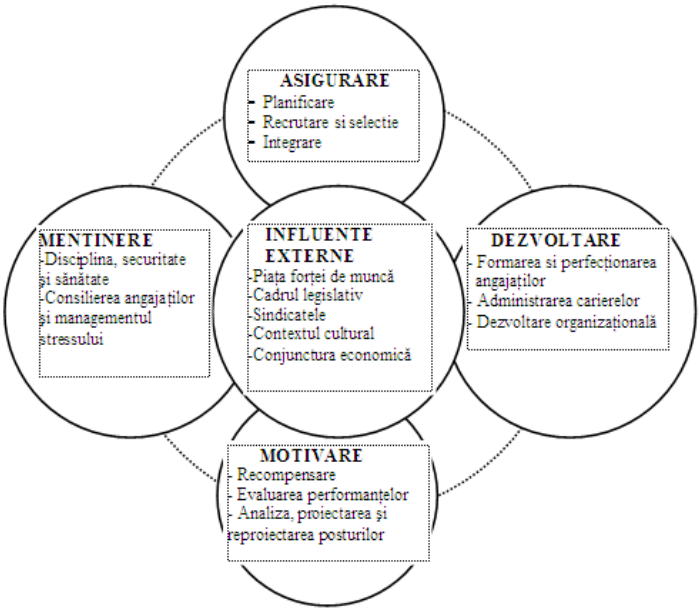
Caracterizare	Concepţii privind personalul	
	Teoria tradiţională a întreprinderii	Managementul resurselor umane
Noţiuni folosite	Forţă de muncă, mână de lucru	Resurse umane
Categorii cu caracter discriminatoriu	Muncă productivă şi creatori de bunuri (categorii privilegiate) / muncă neproductivă şi personal neproductiv (categorii discriminate)	Resurse umane
Modul de abordare a personalului de către manageri	În mod global, ca masă de oameni capabili să muncească	Ca individualităţi, cu personalităţi, nevoi, comportamente şi viziune specifică
Principiul fundamental de salarizare	În funcţie de munca depusă	În funcţie de rezultatele obţinute
Activitatea de evaluare a performanţelor	Nesemnificativă, formală	Esenţială
Stimularea iniţiativei salariaţilor	Absentă; iniţiativa salariaţilor este considerată o diminuare a autorităţii şefilor ierarhici	Susţinută şi promovată prin recompense (promovare în funcţie, salarii mai mari etc.)

[16]

Managementul resurselor umane are cateva principi esentiale, si anume:
persoana, factorul uman este o resursa indispensabila, scopurile se pot atinge doar daca fiecare individ se concentreaza si depune eforturi considerabile in acest sens si crearea unor legaturi intre politicile si sistemele privind resursele umane cu strategia organizatiei.

Prin managementul resurselor umane se urmareste: cresterea productivitatii personalului, incercarea prin conditii de munca decente, prin prime de a motiva personalul si, totodata, imbunatatirea capacitatii de rezolvare a problemelor si de schimbare a organizatiei.

Functiile managementului resurselor umane
Acestea sunt: dezvoltarea, motivarea si mentinerea resurselor umane.



Functiile managementului resurselor umane [16]

Managementul resurselor umane urmareste ocuparea posturilor dintr-o organizatie cu oamenii potriviti. Pentru a fi atins acest scop trebuie sa se identifice necesarul de personal, recrutarea, selectarea, angajarea, motivarea, salarizarea, promovarea , formarea si perfectionarea, precum si activitatile cu caracter social.

Deoarece firmele sunt in continua dezvoltare, departamentul de resurse umane capata o tot mai mare importanta.

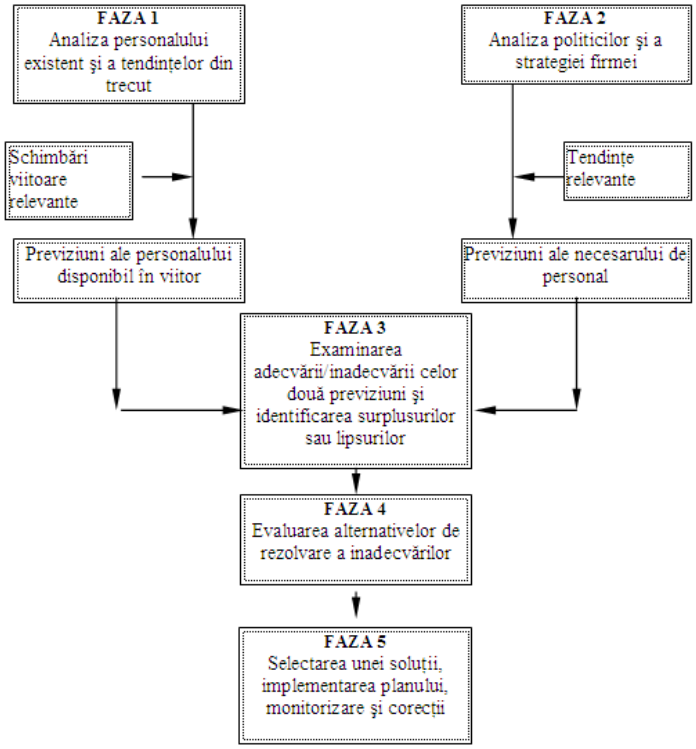
In cazul firmelor mici, in domeniul resurselor umane sunt doar cativa specialisti sau managerii insisi preiau aceasta functie.

Intr-o firma de dimensiuni medii, exista o persoana specializata care coordoneaza activitatile legate de resursele umane. In momentul in care activitatile legate de resursele umane devin complexe, sunt create servicii separate , conduse de un director de resurse umane.

Managementul resurselor umane nu este numai responsabilitatea departamentului de specialitate, ci si a managerilor superiori. De aceea este esential sa existe comunicare intre acestia.

Asigurarea resurselor umane cuprinde numeroase activitati:

1.Planificarea resurselor umane - se refera la obiectivele pe termen lung privind resursele umane ale organizatiei, raportandu-se totodata, si la piata muncii .



Planificarea resurselor umane [16]

2.Recrutarea si selectia – aceste activitati se desfasoara in momentul angajarii de personal. Ele sunt complementare.

Prin recrutare intelegem deschiderea unei pozitii noi din cadrul unei organizatii si, in acelasi timp incercarea de a identifica si a atrage persoanele interesate de aceste posturi. Recrutarea nu presupune luarea unei decizii, aceasta se ia in etapa de selectie.

Selectia consta intr-un ansamblu de procese prin care sunt alese persoanele care se potrivesc , care au aptitudinile si deprinderile necesare pentru anumite posturi.

3.Un rol esential il are integrarea angajatilor. In aceasta etapa se poate vorbi atat de integrare din punctul de vedere al postului ocupat si al sarcinilor ce trebuiesc indeplinite, cat si din punct de vedere social. Este indicat ca atunci cand o persoana noua este angajata sa se desemneze un ghid (o persoana cu experienta care poate fi: un coleg, seful direct etc) pentru a se usura etapa de integrare a angajatului respectiv.

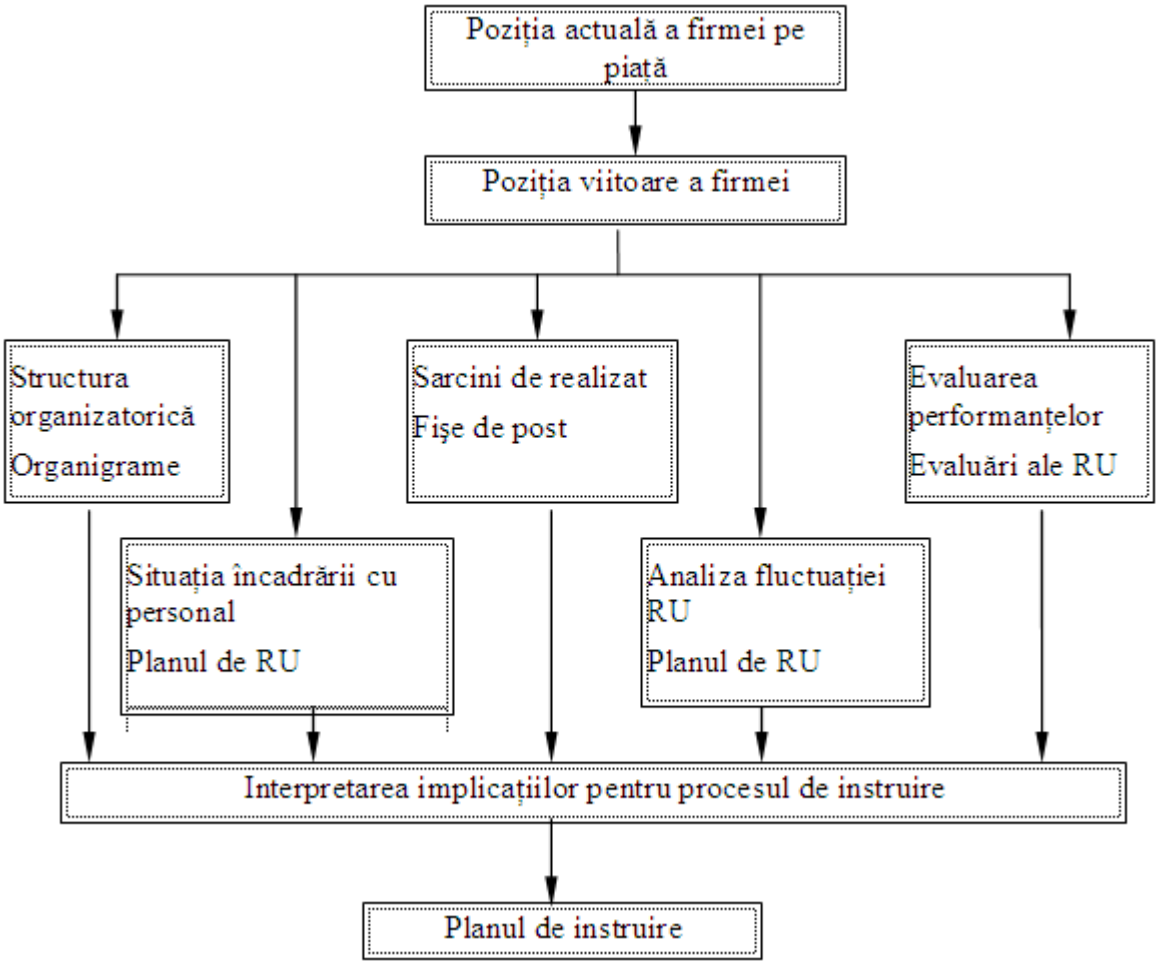
Dezvoltarea resurselor umane

Aceasta are consta in mai multe activitati.

De exemplu, se doreste ca angajatii sa fie intr-o continua evolutie; perfectionarea unui angajat se poate face atat in cadrul organizatiei, cat si in afara ei.

Deasemenea, inca de la inceputul contractului, un angajat trebuie sa stie ce posibilitati exista in firma, in organizatia respectiva de a promova.

O alta activitate ce face parte din dezvoltarea resurselor umane consta in crearea unor relatii stabile, atat in interiorul unor grupuri, cat si la nivelul grupurilor. Aceasta are ca scop ajutorarea reciproca a grupurilor pentru a se putea intui, declansa si controla schimbarea.



Planificarea instruirii [16]

ROBITU PAUL- 442A

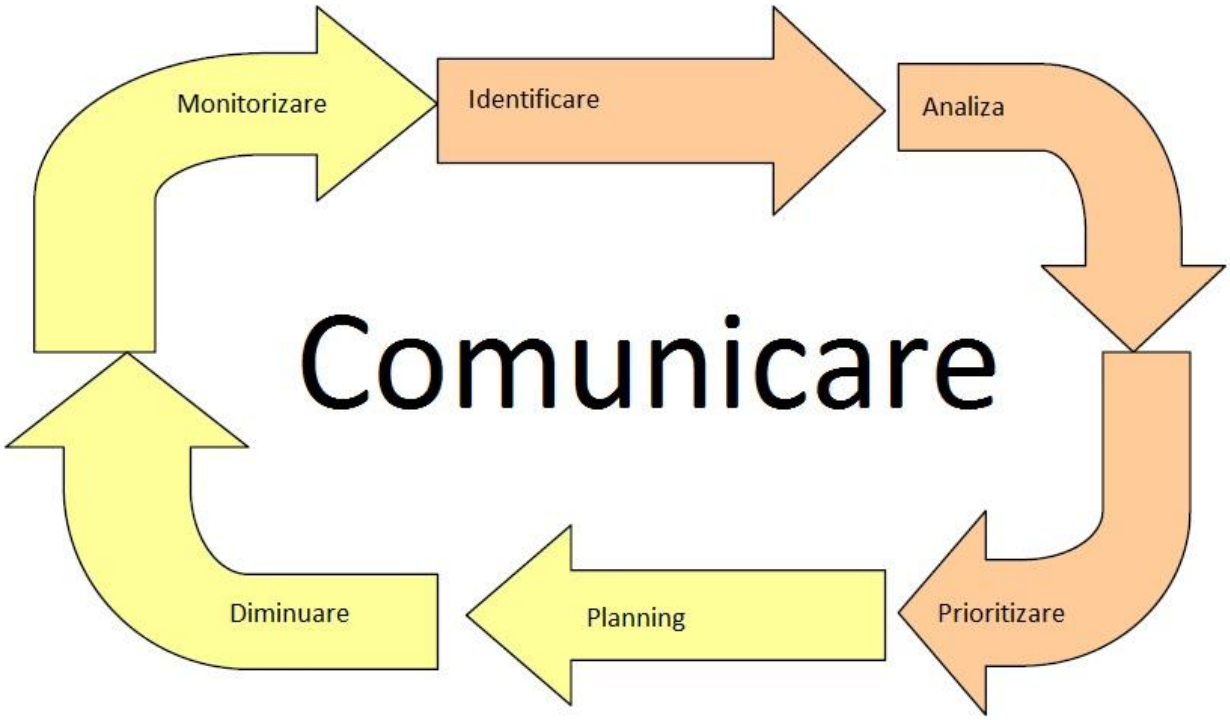
CAPITOLUL 6 : MANAGEMENTUL RISCULUI

Secolul XXI a adus nevoia automatizarilor nu doar in domeniul industrial dar si in alte activitati diverse. Producerea de software competent a devenit din ce in ce mai complicata o data cu cresterea complexitatii acestuia. Acest lucru a dus desigur si la un procentaj mai mare al proiectelor software neduse la bun sfarsit sau aflate in colaps. Studiile arata ca doar 51% din proiectele de software incepute ajung sa vada lumina zilei dar nu se incadreaza in predictiile de cost, iar 31% sunt anulate inaintea terminarii.

Managementul riscului isi gasea aplicabilitatea in alte domenii, dar in ultimii ani a devenit din ce in ce mai interesant pentru industria software. Desi nu este inca la maturitate, vom incerca sa desciem fundamentele lui. Riscurile in aceasta industrie sunt la nivel de proiect sau de proces.

O definite a riscului in sens software este greu de dat, fiind mult mai usor de definit in domenii precum cel medical sau cel financiar. In cazul nostru, putem spune ca mangementul riscului reprezinta o serie de actiuni care sunt realizate cu scopul reducerii incertitudinilor asocitate diferitelor riscuri. Deci putem spune ca este la nivelul celor care au puterea de decizie intr-un proiect. Astfel de practici pot ajuta proiectele software sa fie finalizate intr-un deadline, buget si in cadrul unor specificatii bine stabilite de catre client. Se urmareste gasirea diferitelor pericole, analizarea lor, si gasirea de strategii pentru controlul si diminuarea acestor riscuri. Nu inseamna neaparat anularea acelor proceduri sau sarcini care au un risc ridicat, multe dintre sarcinile din industria software au riscuri cunoscute, companiile reusind sa iasa in evidenta in momentul in care isi asuma mai multe riscuri si le controleaza mai bine, avand un impact cat mai mic asupra clientilor sau utilizatorilor.

Scopul managementului riscurilor este cunoasterea tuturor riscurilor ce pot aparea in cadrul proiectului, incadrarea lor intr-o clasa de severitate, si in consecinta, luarea de decizii. Putem deci imparti managementul riscului in doua faze: evaluarea riscurilor si controlul riscurilor. Evaluarea riscurilor implica identificarea, analiza si prioritizarea iar controlul riscurilor implica planning, diminuare, monitorizare. Aceste concepte au fost introduse de catre Boehm in 1989 si sunt ilustrate in fig 1. De asemenea, este important sa tratam acesti pasi in mod iterativ pe tot parcursul proiectului, si sa devina o obisnuita pentru membrii care au puterea de decizie (project manager).



Ciclul Managementului riscurilor[29]

Comunicarea permanenta intre membrii echipei proiectului este vitala. Riscurile trebuie dezbatute atat cu echipa de programatori cat si cu cea de analisti sau cu reprezentantii clientilor. Acest lucru permite propagarea informatiei si sta la baza unui management eficient.

Identificarea riscurilor

Identificarea se realizeaza prin sesiuni de brainstorming in care membrii echipei se gandesc si creeaza o lista cu toate riscurile posibile inainte de a deveni probleme reale. Este important sa fie punctate tipurile de riscuri pe care o echipa le poate intalni pentru a le putea analiza. Acestea pot fi: Riscuri Generale sau Riscuri Specifice.

Riscurile generale sunt cele care reprezinta o problema pentru orice proiect software. Din aceasta categorie fac parte pierderea unui membru vital al echipei, schimbarea specificatiilor, falimentul companiei software sau diminuarea bugetului clientului. O companie puternica de producere a soft-ului trebuie sa aiba informatii la orice moment despre programatori, manageri, clienti, etc.

Riscurile specifice produsului pot fi usor identificate si diferentiate de riscurile generale de catre cei care cunosc foarte bine tehnologia folosita, se implica in comunicarea cu membrii echipei. Cele doua categorii de riscuri se pot imparti in continuare in riscuri legate de produs, de business sau de proiect. Riscurile legate de produs sunt acelea care pot afecta performatele sau calitatea produsului software in curs de implementare. Riscurile de proiect sunt acelea care intervin in resursele de buget sau de personal si pot afecta atingerea deadline-ului. Riscurile legate de business sunt foarte grave si se leaga de cat de potrivit este produsul pe piata actuala. Acesta poate sa fie excelent dar este posibil sa nu fie comercializat datorita dinamicii

pietei.
Cativa dintre factorii care trebuie luati in considerare in momentul analizarii riscurilor de business, de proiect si de produs pot fi:

- Riscurile *tehnologice* apar din cauza sistemelor software si al hardware-ului folosit in implementarea noului produs software. De exemplu, anumite librarii de cod sau bucati de software disponibile gratuit pe internet pot cauza probleme de stabilitate, intrucat nu au ajuns nici ele la versiuni stabile. Este de preferat sa se foloseasca versiuni stabile, bine consacrate si documentate si sa se evite update-ul acestora.
- Riscurile de *proces* sunt legate procesele folosite de catre echipa. Ele trebuie urmate indeaproape de catre toti membrii echipei
- Riscurile de *complexitate* sunt asociate „marimii" produsului (foarte complex) dar si a „marimii" echipei (foarte greu de coordonat).
- Riscuri de *clienti* provin din schimbarile produse neasteptat de catre clienti. Acestia nu cunosc exact impactul pe care il poate avea o schimbare majora la nivelul specificatiilor cand proiectul este intr-o faza de implementare. Asadar trebuie acordata o importanta sporita comunicarii dintre client si echipa pentru a asigura realizarea produsului software in concordanta cu cererile.
- Riscuri de *estimare* sunt introduse de catre estimarile proaste legate de resursele umane, bugetare sau de timp. O estimare initiala ofera o idee generala despre evolutia proiectului. Cateva erori aparute la aceasta estimare pot fi foarte jenante pe parcursul desfasurarii etapelor proiectului.
- Riscuri *organizationale si manageriale* sunt legate mai ales de compania care produce software-ul. Situatia sa financiara, stabilitatea acesteia au un potential ridicat de a distrage atentia de la sarcinile proiectului.

In acest sens, este indicat ca participantii la buna desfasurare a proiectului sa fie incurajati sa se implice in identificare si raportarea eventualelor riscuri pentru a putea fi monitorizate si gestionate.
Analiza riscurilor
Aceasta faza se desfasoara dupa ce riscurile au fost identificate si enumerate si presupune transformarea lor in informatii pretioase pentru persoanele care au puterea de decizie. Fiecare risc este luat in calcul si se cauta sa se eticheteze cu un grad de severitate si o probabilitate ca acesta sa devina o problema. Daca unele riscuri au o probabilitate mai mare sa apara, altele sunt mai putin semnificative. In acest sens se poate folosi o scara, cea procentuala sau o scara de la 1 la 10, asociata unor anumite „etichete" precum imposibil,foarte putin posibil sau foarte posibil.
Se trece apoi la evaluarea impactului pierderii care poate fi creata de catre riscul respectiv. Este necesar sa se atribuie valori numerice de pierderi specifice riscului studiat pentru a putea fi usor de inteles magnitudinea pierderii posibile.

In anul 1996, Gupta si Clarke au propus o metoda ingenioasa de a determina ce „eticheta” poate fi atribuita fiecarui risc in parte. Fiecare membru al echipei ofera estimarea proprie, anonima a nivelului de risc pentru fiecare problema in parte. Aceste informatii sunt colectate si discutate intr-o sedinta. Dezbaterea va avea loc in cazul in care exista discrepante evidente in alegerile facute de catre membrii echipei, iar mai apoi se reitereaza procedeul pana se ajunge la rezultate convergente. Metoda poata numele de Tehnica Delphi.
Dupa ce s-a ajuns la o convergenta a rezultatelor tehnicii Delphi, rezultatele trebuie sa fie trecute intr-un tabel pentru a pune in evidenta diferentele dintre ele. Un exemplu de tabel de risc poate cuprinde coloane pentru: gradul de risc, probabilitatea (de preferabil numerica), impactul, rang-ul, rang-ul saptamanii trecute (pentru o monitorizare mai usoara a schimbarilor), actiune (ce trebuie echipa sa faca pentru a putea gestiona riscul). In general, actiunea nu se completeaza pe loc, se prefera in primul rand prioritizarea riscului.

Prioritizarea riscurilor

Impactul fiecarui risc determina daca o actiune va fi luata sau nu in acest sens. Faza de prioritizare este cea in care echipa decide la care dintre riscurile evidentiate in tabelul de analiza trebuie atribuita o actiune. In mod evident, primele riscuri din tabel sunt cele cu cea mai mare prioritate si cu cel mai mare impact, tabelul fiind ordonat dupa acesti doi parametrii.
In 1989, Boehm propune formula de calcul a expunerii la risc (ER)[29]:

$$ER = P \times I$$

unde P reprezinta probabilitatea de aparitie a riscului iar I reprezinta impactul pe care pierderea cauzata de acest risc o poate aduce.
Dupa ce fiecare rist a fost prioritizat si intreg tabelul a fost sortat, echipa are sarcina de a alege un prag pana la care riscurile sunt trecute in urmatoarele faze. Riscurile de sub acest prag raman totusi in tabelul creat anterior pentru a fi monitorizate.

Planning-ul

Scopul acestei etape este gasirea masurilor care trebuie implementate pentru fiecare dintre riscurile deasupra pragului setat in faza anterioara. Putem avea diferite tipuri de masuri:

- Masuri de contingenta – descriu ceea ce trebuie facut in cazul in care riscul se materializeaza. Se creeaza practic o strategie gata sa fie pusa in executie pentru a rezolva problema.
- Masuri de reducere – planul poate contine masuri de tip „orar potential” in concordatan cu bugetul. Daca echipa este ingrijorata ca un anumit factor

poate duce la o intarziere se poate aplica acesta masura pentru a reduce potentialul impactul financiar asupra companiei.

- Masuri de informare – se pot aplica daca din experienta anterioara o tehnologie noua a creat un risc. Echipa poate decide sa investeasca o anumita suma de bani pentru a aprofunda acea tehnologie.
- Masuri de acceptare – uneori este foarte greu sa se gaseasca niste masuri concrete de actiune pentru evitarea unui anumit risc. Uneori, echipa trebuie sa faca si compromisuri si sa accepte riscul respectiv si pierderea aferenta. In acest caz, nu se noteaza nicio actiune.

Diminuarea

Este etapa logica si imediat urmatoare dupa un planning bine executat. Echipa dezvolta strategii de reducere a impactului riscurilor si sunt imaginate situatii in care riscul este fie eliminat fie diminuat. Aceste actiuni pot fi si ele documentate in tabelul de risc pentru a usura luarea deciziei in situatii critice. Cateva exemple pot include : *protejarea de risc* – se poate aplela la companii de asigurare care sa se ocupe de despagubire in cazul in care acel risc devine critic; *evitarea riscului* – daca se descopera o strategie in care orice cale duce la pierdere, se va opta pentru eliminarea completa a acelui risc.

Este esentiala ca aceste doua faze (planning-ul si diminuarea) sa aiba o stransa legatura si radacini solide in realitate. Adica, trebuie sa se faca anliza din punct de vedere al beneficiilor si al costurilor pentru a decide daca merita efortul. Shari L. Pfleeger propune solutia calcularii unei parghii:

$$Parghia = \\ = \frac{(Expunerea\ la\ risc\ inaintea\ diminuarii - Expunerea\ la\ risc\ dupa\ diminuare)}{costul\ reducerii\ riscului}$$

Daca aceasta parghie este mai mica sau egala decat 1, este evident ca reducerea riscului respectiv nu este benefica. Daca este mai mare decat 1 dar nu cu mult, beneficiul este inca sub semnul intrebarii si echipa trebuie sa caute solutii alternative.

Monitorizarea

Dupa ce riscurile au fost identificate, analizate, prioriziate si s-au stabilit masurile care trebuie luate pentru fiecare risc in parte, este foarte important ca echipa sa monitorizeze produsul software si riscurile in sine. Acest lucru poate fi parte a rutinei fiecarui membru al echipei sau se poate face prin actiuni specifice de managementul riscurilor.

Reevaluarea riscurilor trebuie sa se realizeze in mod regulat pentru a se determina daca alte circumstante au cauzat schimbari ale impacturilor. In mod evident, se pot adauga sau elimina riscuri in functie de fiecare situatie in parte si se va trece din nou prin etapa de prioritizare pentru a se identifica locul in care se va plasa noul risc. Cu cat timpul trece si proiectul se maturizeaza, informatia obtinuta pe parcurs poate

altera profilul riscului sau timpul poate sa rafineze riscul respectiv impartindu-l in mai multe riscuri detaliate care pot fi mai usor diminuata si monitorizate.
Comunicarea
Toate aceste etape mentionate mai sus se pot realiza doar in jurul unei comunicatii eficiente intre echipa de manageri, echipa de development, marketing si reprezentatii clientului.

CAPITOLUL 7 : MANAGEMENTUL COMUNICARII

Proiectele software de astazi au ajuns la o complexitate foarte ridicata pentru a incerca sa tina pasul cu evolutia fulgeratoare a hardware-ului, devenind proiecte foarte ample, pe perioade foarte lungi, avand echipe mari de specialisti din diverse tari avand background diferit. O componenta critica a procesului de management al unui astfel de proiect este managementul comunicarii, cuprinzand patru mari componente:

- Procese de management de comunicare si masurile aferente.
- Partile interesate si nevoile acestora.
- Echipa care realizeaza proiectul.
- Instrumente de comunicare.

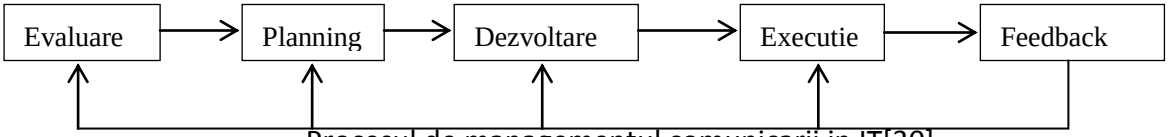
Procesul de management are rolul de a stuctura activitatile care intervin in comunicarea interna, folosind diverse instrumente folosite in IR. Fiecare parte interesata (stakeholders) are nevoie de informatii diferite, furnizate cu o anumita intensitate si primite pe anumite canale de comunicatie.
Fazele procesului de managementul comunicatiei:

1. Evaluarea – critica pentru succesul comunicarii. Necesitatile trebuie analizate in profunzime si fiecare grup de parti interesate este identificate si ii sunt listate cerintele, canalele de comunicatie, instrumentele folosite pentru o comunicare eficienta.
2. Planning – la acest pas se va dezvolta un plan de comunicare detaliat. Se definitiveaza emitatorii si receptorii, modul in care acestia comunica, instrumentele folosite si ce tip de feedback trebuie sa fie interschimbat. Toate activitatile trebuie sa corespunda cererilor partilor interesate.
3. Dezvoltarea – se va implementa infrastructura continand toate elementele analizate in etapa anterioara. Aceasta faza poate sa fie costisitoare, atat din punct de vedere al timpului dar si a costului, in functie de cat de sofisticate sunt instrumentele de proiectare si canalelel de comunicare. Costurile nu trebuie sa depaseasca asteptarile si este vital ca toate canalele de comunicatie sa fie testate in pralabil intrucat o problema aparuta mai tarziu

in procesul de comunicare poate duce la scaderea increderii si se poate intarzia proiectul in sine.

4. Executia – activitatile sunt executate in strictetea etapelor anterioare. Inaintea executiei toate informatiile trebuie sa fie verificate cu atentie deoarece exista riscul transmiterii de informatii care nu sunt la curent cu stadiul actual la proiectului.
5. Feedback – implica monitorizarea si evaluarea. Se poate intampla ca in urma feedback-ului partilor interesate sa se decida schimbarea planului de comunicare, a canalelor sau instrumentelor folosite.

Diagrama ilustreaza sumar etapele procesului de management al comunicarii.



Procesul de managementul comunicarii in IT[30]

Identificarea partilor interesate (stakeholders):

- Echipa – angajatii direct implicati in activitatile de dezvoltarea a produsului software. Este un grup relativ restrans care are nevoie constanta de informatii legate de progres, riscuri, schimbari etc.
- Angajati colaboratori – un grup format din toti angajatii companiei care sustin echipa mai sus mentionata in timpul implementarii prin coordonarea activitatilor in propriile departamente, oferind feedback si informatii concludente echipei. Informatiile vitale in acest grup sunt legate de progresul proiectului.
- Utilizatori (end users) – reprezinta un grup foarte necesar in IT. Acestia au nevoie de informatii despre solutia oferita de echipa de dezvoltare.
- Restul angajatilor – cel mai mare grup. In companiile mari de IT, proiectele dezvoltate afecteaza pe toti angajatii. Informatiile necesare acestui grup sunt legate de functionalitatea proiectului dezvoltat si beneficiile pe care acesta le va aduce.

Responsabilitatile managerului in ceea ce priveste comunicarea sunt : dezvoltarea planului de comunicare, identificarea grupurilor de stakeholders, supervizarea executiei activitatilor de comunicare, monitorizarea si analizarea feedbackului, analizarea informatiilor necesare pentru stakeholderi etc.
Instrumentele de comunicare trebuie sa fie adaptat pentru proiectul in cauza si pentru a satisface nevoile fiecarui grup care ia parte la dezvoltarea proiectului software. Tehnologiile folosite includ software-uri de colaborare, platforme si aplicatii internet, aplicatii desktop din domeniul business etc. Comunicatia eficienta este dependenta de calitatea informatiei gestionata de instrumentul folosit, cat de des este actualizata, dimensiunea proiectului, nevoile steakholderilor.

CONSTANTIN SEBASTIAN -442A

CAPITOLUL 8 : MANAGEMENTUL CALITATII

Managementul calității este un ansamblu de activități ce au ca scop indeplinirea unor obiective, prin folosirea optimă a resurselor. Acest ansamblu urmareste urmatorii pasi: planificare, coordonare, organizare, control şi asigurare a calității. Intreprinderea îşi propune o anumite obiective: economice, sociale, tehnice, comerciale, care se realizează prin intermediul unor obiective operaționale.

Un bun Sistem de Management al Calitații indeplineste urmatoarele conditii:

- să fie stabilit în scris;
- să asigure indeplinirea cerințelor clienților;
- să asigure indeplinirea cerințelor organizației;
- să fie aplicabil în toate activitațile organizației.

8.1 Principiile managementului calității

8.1.1 : Orientarea către client

"Din moment ce organizatiile depind de clientii lor, acestea trebuie sa inteleaga necesitatiile de moment si de viitor ale acestora, sa satisfaca cerintele clientilor si sa incerce sa depaseasca asteptarile acestora. ."[21]

Beneficii:

- marirea venitului;
- dezvoltarea nivelului de utilizare a resurselor;
- dobandirea loialității clientului;

8.1.2: Conducerea (leadership)

"Liderii unei organizatii stabilesc unitatea scopului si directia acestuia. Acestia trebuie sa opteze pentru crearea si mentenanta unui mediu in care persoanele pot sa se implice la capacitate maxima in implinirea obiectivului companiei." [21]

Beneficii:

- motivarea angajatilor în concordanta cu nivelul companiei;
- dezvoltarea aplicatiilor ca un tot unitar;
- diminuarea problemelor de comunicare.

8.1.3: Implicarea personalului

"Persoanele de pe absolut orice nivel al unei companii reprezinta esenta acesteia. Implicarea totala a acestora permite abilitatilor lor sa fie utilizate pentru beneficiul organizatiei. " [21]

Beneficii:

- motivarea personalului implicat;
- inovare şi creativitate;
- dezvoltarea performantelor membrilor;
- implicarea membrilor in procesul de îmbunătățire.

8.1.4: Abordarea bazată pe proces

"Obtinerea rezultatului dorit este posibila atunci cand activitatile si resursele sunt organizate sub forma unui proces. " [21]

Beneficii:

- reducerea costurilor;
- perioade de inactivitate mai scurte
- rezultate mai bune;
- posibilitati de dezvoltare.

8.1.5: Abordarea sistematica a managementului

"Eficienta organizatiei este atinsa atunci cand identificarea, intelegerea si organizarea tuturor proceselor se face asemenea unui sistem."[21]

Beneficii:

- realizarea obiectivelor impuse;
- abilitatea de concentrare pe proiect;
- impunerea unei eficiente ridicate asupra tuturor proiectelor.

8.1.6: îmbunătățirea continuă

"Dorinta si incercarea de a ajunge la o imbunatatire continua este o calitate necesara unei organizatii."[21]

Beneficii:

- îmbunătățirea capabilității organizației ce duce la dezvoltare;
- subordonarea activităților de îmbunătățire din toate nivelele companiei la strategia organizației;
- posibilitatea de a reacționa rapid şi flexibil la oportunități.

8.1.7: Luarea deciziilor bazate pe fapte

"Deciziile eficiente sunt luate intotdeauna pe baza analizelor si informatiilor."[21]

8.1.8: Relatii cu beneficii mutuale cu furnizori

"Din moment ce o organizatie este independenta de furnizorii ei, este necesara o relatie bazata pe beneficii mutuale"[21]

8.2 Standardul ISO

"ISO (International Organization for Standardization) este cel mai mare dezvoltator din lume de standarde internationale. Aceste standarde internationale ofera specificatii despre produse, servicii incercand sa faca industria mai eficienta. Acestea incearca sa treaca de limitele impuse in comertul international."[22]

Standardul ISO pune in evidenta urmatoarele concepte ale calitatii:

MANAGEMENTUL CALITĂȚII - determină politica în domeniul calității cu ajutorul planificarii calității, controlului calității, asigurari calității şi îmbunătățirii calității.

PLANIFICAREA CALITĂȚII – se stabilesc obiectivele şi condițiile dezvoltarii proiectului(cu referinta la calitate).

CONTROLUL CALITĂȚII – functii si reguli aplicate pentru mentinerea condițiilor referitoare la calitate.

ÎMBUNĂTĂȚIREA CALITĂȚII – creşterea eficientei activităților şi proceselor în scopul dezvoltarii companiei.

CAPITOLUL 9 : ANALIZA DECIZIILOR

9.1. Introducere

Analiza deciziilor este o disciplina ce utilizeaza filozofia, teoria, metodologia si practica profesionala necesara pentru a lua decizii importante intr-un mod formal. Analizia deciziilor include mai multe proceduri, metode pentru identificarea, reprezentarea clara si luarea formala a deciziilor .

"Principalele elemente implicate în procesul de decizie sunt:

- un număr de posibile acțiuni, A_i , dintre care va fi selectata una;
- un număr de evenimente sau "stări ale naturii", S_j , din care oricare poate avea loc;
- valoarea, profitul sau consecința, C_{ij}, pentru decident, de realizare a uneia din acțiunile disponibile, având în vedere posibilele stări ale naturii;
- criteriul în funcție de care decidentul alege între acțiunile alternative."[23]

9.2. Decizii în condiţii de certitudine

9.2.1. Analiza Pareto

Analiza Pareto este o tehnică ce foloseşte principiul cu acelasi nume - "cunoscut si ca regula 80-20 sau legea celor putini importanti, principiul spune ca pentru multe evenimente, aproximativ 80% din efecte rezulta din aproape 20% din cauze. "[24]

9.2.2. Analiza comparării pe perechi.

"Se refera la orice proces pentru compararea entitatilor in perechi pentru a stabili care entitate este preferata."[25]

Acest algoritm facilitează alegerea celei mai importante probleme care trebuie rezolvată, sau a soluției care aduce cel mai mare profit.

9.2.3. Analiza matricii de decizii

Este o tehnica cantitativa pentru a ordona optiunile multi dimensionale ale unui set. O matrice de decizii de baza consta in stabilirea unui set de optiuni care sunt punctate si adunate pentru a obtine scorul total care apoi este optimizat.[26]

9.3. Decizii în condiţii de incertitudine

9.3.1. Criteriul Laplace

Probabilitatile anumitor conditii sunt necunoscute dar se pot presupune egale. Asadar scopul initial este considerata o problema de luare a deciziei in conditii de risc cand un anumit eveniment este favorit sa se intample.

9.3.2. Criteriul maximax

In teoria deciziilor, reprezinta regula deciziei optimiste in cazul incertitudinii. Criteriul enunta ca persoana care ia decizia trebuie sa aleaga actiunea a carei castig maxim este mai bun decat castigul tuturor celorlalte actiuni posibile in conditiile date.

9.3.3. Criteriul maximin

In teoria deciziilor, reprezinta regula deciziei pesimiste in cazul incertitudinii.

Criteriul enunta ca persoana care ia decizia trebuie sa aleaga actiunea a carei pierdere maxima este mai mare pierdea tuturor celorlalte actiuni posibile in conditiile date.

Bibbliografie

1. http://www.techit.ro/inginerie_software.php
2. http://www.ieee.org
3. http://andrei.clubcisco.ro/cursuri/
4. http://ro.wikipedia.org/wiki/Inginerie_software.
5. http://bigfoot.cs.upt.ro
6. http://www.oeconomica.uab.ro
7. http://www.techit.ro/analiza_software11.php
8. Inginerie software - curs, prof. Stefan Stancescu
9. [Sommerville 2006] Ian Sommerville Software Engineering, Pearson International Edition, Addison-Wesley ed., 2006.
10. [Pohl 2010] Klaus Pohl, Requirements Engineering, Springer, 2010
11. http://en.wikipedia.org/wiki/Work_breakdown_structure
12. http://www2.cit.cornell.edu/computer/robohelp/cpmm/WBS.jpg
13. http://en.wikipedia.org/wiki/Cost_estimation_in_software_engineering
14. http://en.wikipedia.org/wiki/Function_point_analysis
15. http://en.wikipedia.org/wiki/Human_resource_management
16. http://www.scribd.com/doc/12410853/managementul-resurselor-umane
17. http://ebooks.unibuc.ro/StiinteADM/cornescu/cap12.html
18. http://www.personneltoday.com/blogs/human-resources-guru/2007/08/cat bert-shows-tougher-side-t o-human-resources.html
19. http://www.bls.gov/ooh/Business-and-Financial/Human-resources-specialists .html
20. http://en.wikipedia.org/wiki/Time_management
21. http://en.wikipedia.org/wiki/Quality_management
22. http://www.iso.org/iso/home/about.htm
23. http://en.wikipedia.org/wiki/Decision_analysis
24. http://en.wikipedia.org/wiki/Pareto_principle
25. http://en.wikipedia.org/wiki/Pairwise_comparison
26. http://en.wikipedia.org/wiki/Decision-matrix_method
27. http://demo.activemath.org/ActiveMath2/main/view.cmd;jsessionid=8B655AB17B741C5F20836189976F3819?book=OperResearchCustom_course_engin&page=15

28. S.C. Misra, V. Kumar, U. Kumar, ”Different Techniques For Risk Management In Software Engineering: A Review”, Eric Sprott School of Business, 2006 Carleton University, Banff, Alberta, Canada.
29. Williams L. ”Risk Management” 2004.
30. [3] Rafal W. Ceigielski, Jaroslaw A. Chudziak, Joanna Meyer, ”Communication Management and its impact on successful IT Program”, Prokom Software S.A and Institute of Computer Science, Warsaw University of Technology, Warsaw, Poland
31. Chris Croft, “Time management”, 1996, Cengage Learning EMEA
32. http://users.cs.tuiasi.ro/~fleon/Curs_MPS/CursMPS05.pdf
33. “Recruitment and selection-Hiring the people you want”, Eric Garner
34. “Human Resource Management-12th Edition”, Robert L. Mathis, John Harols Jackson, 2008