A X-a Conferință Națională multidisciplinară - cu participare internațională,
"Profesorul Dorin PAVEL - fondatorul hidroenergeticii românești",
SEBEŞ, 2010

CERINȚE: DEFINIȚII ŞI CARACTERISTICI

Lavinia-Gabriela SOCACIU, Ioan BLEBEA

REQUIREMENTS: DEFINITIONS AND CHARACTERISTICS

The literature of requirements engineering provide us the material foundation to define a good requirements, classification type of requirements and characteristics to define a good requirements, as well key issues to be taken into account when we use requirements.

Cuvinte cheie: cerințe, ingineria cerințelor

1. Definiții ale cerințelor

Glosarul de standarde al terminologiei ingineriei produselor software [1] defineşte cerințele astfel:

● o ***condiție*** sau ***capabilitate*** necesar a fi îndeplinită de către un sistem, pentru ca un *utilizator* să poată rezolva o anumită *problemă* sau să atingă un anumit *obiectiv*;

● o ***condiție*** sau ***capabilitate*** pe care un *sistem* trebuie să o realizeze sau să o posede pentru a satisface un contract, standard, specificație sau alt *document* formal impus;

● un ***document*** – reprezentarea unei *condiții* sau *capabilități*, aşa cum este descrisă la punctele 1 şi 2 de mai sus.

Prima parte, a definiției cerinței reprezintă *punctul de vedere al consumatorului* care are nevoie de ceva specific pentru a rezolva o problemă. O cerință există atâta timp cât ea reprezintă soluția pentru o problemă a consumatorului.

A doua parte a definiției reprezintă *punctul de vedere al dezvoltatorului*, care trebuie să îndeplinească "o condiție sau o

capabilitate a sistemului". Aşa cum se poate vedea din definiție, pentru dezvoltator, referința este un "document formal impus".

Partea a treia a definiției exprimă un punct de vedere comun, sau un punct de vedere general al consumatorului şi al dezvoltatorului. Primele două puncte de vedere nu pot exista dacă nu există documentul pe care ambele părți să îl poată folosi ca referință. Aşadar, conform punctului trei din definiția cerinței, o cerință care nu este documentată, *nu există*.

Din definiția cerinței, prezentată mai sus, rezultă că *cerințele* privite din perspectiva consumatorului reprezintă o soluție utilă pentru rezolvarea unei probleme, în timp ce, cerința din perspectiva dezvoltatorului reprezintă ceea ce el trebuie să realizeze, în conformitate cu specificațiile. Prin urmare, cerința trebuie exprimată în aşa fel încât să poată fi interpretată fără dubii de ambele părți [2].

Literatura de specialitate, ne oferă însă mai multe definiții pentru termenul de „cerință", fiecare având însă o abordare diferită. În continuare sunt prezentate câteva definiții şi abordări ale cerințelor.

Abbott a definit cerințele ca fiind: „orice funcție, constrângere, sau orice altă proprietate care trebuie furnizată, întâlnită sau care satisface nevoile utilizatorilor țintă".

Davis a definit cerințele astfel: „nevoile utilizatorilor, caracteristicile necesare, funcțiile sau atributele sistemului care pot fi înțelese dintr-o poziție externă sistemului".

Young, a definit cerința ca fiind un „atribut necesar într-un sistem, o declarație care identifică o capacitate, sau o caracteristică, factorul de calitate al unui sistem, astfel încât acesta să aibă valoare şi utilitate pentru un client sau pentru un utilizator".

Cerințele sunt considerate ca fiind fundamentul necesare activităților de dezvoltare al sistemelor, iar cele mai frecvente erori îşi au originea în faza de colectare a cerințelor.

În timpul fazei de colectare şi de analiză a cerințelor costul privind corectarea erorilor este cel mai scăzut, pe măsură ce proiectul evoluează, corectarea erorilor devine din ce în ce mai costisitoare şi necesită mai mult timp pentru corectarea erorilor (figura 1). Acest lucru demonstrează faptul că cerințele au un rol important în timpul procesului de dezvoltare [3].

2. Clasificarea cerințelor

Literatura de specialitate prezintă mai multe moduri în care pot fi clasificate cerințele. În această lucrare am ales clasificarea realizată de Robertson şi Bracket în funcție de tipul cerințelor (figura 2).

Cerinţele funcţionale specifică funcţiile pe care un sistem sau componente ale sistemului trebuie să fie capabil să le execute. Potrivit lui Robertson, cerinţele funcţionale descriu ceea ce produsul trebuie să facă sau acţiunile pe care produsul trebuie să le execute pentru a oferi funcţionalitate utilă. Cerinţele funcţionale au ataşate următoarele cerinţe de performanţă:

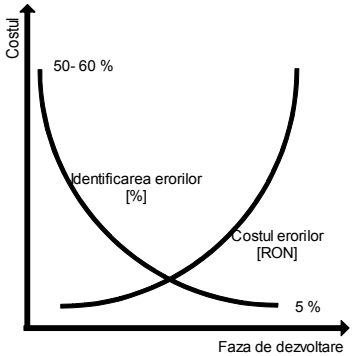

Fig. 1 Costul erorilor în timpul dezvoltării sistemului

● ***capacitatea*** - se referă la cât de mult din capabilitatea respectivă este necesară la un moment dat

● ***viteza*** - arată cât de rapid trebuie să se execute o operaţie, măsurată în număr de operaţii pe unitatea de timp.

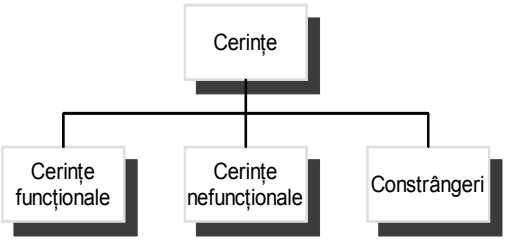

Fig. 2 Tipuri de cerinţe

● ***acurateţea unei operaţii*** - este măsurată prin diferenţa dintre ceea ce se intenţionează să execute o operaţie şi ceea ce execută ea în realitate.

Potrivit lui Robertson, cerinţele nefuncţionale sunt acele proprietăţi sau calităţi pe care produsul trebuie să le deţină, cum ar fi aspectul, viteza sau precizia.

Cerinţele nefuncţionale includ cerinţele de calitate precum:

- cerinţele de performanţă (viteza de răspuns, disponibilitatea sistemului, timpul de recuperare în caz de indisponibilitate temporară a sistemului, utilizarea resurselor);
- siguranţa în funcţionare (frecvenţa avariilor, uşurinţa recuperării);
- suportabilitatea (posibilităţile de adaptare, posibilităţile de extindere, configurabilitatea, compatibilitatea cu alte sisteme, localizare);

▪ cerințe de implementare (standarde, legislație aplicabilă, politici de securitate, limitări de resurse);

▪ uşurința în utilizare (consistența interfeței utilizator, standarde de ergonomie aplicabile, documentație);

▪ cerințe de interfațare cu alte sisteme.

Proiectarea şi implementarea **constrângerilor** reprezintă condițiile limită privind cerințele de proiectare şi implementare. De exemplu, o astfel de constrângere în ingineria software este faptul că softul trebuie să ruleze utilizând un anumit sistem de date, sau bugetul de dezvoltare este o constrângere care limitează numărul şi gradul de dificultate al cerințelor [2, 4].

3. Caracteristici ale cerințelor

Caracteristicile esențiale pe care trebuie să le posede o cerință (figura 3) sunt:

▪ **Necesară.** O cerință există dacă şi numai dacă este necesară. Amintind de prima parte a definiției cerințelor din standardul IEEE Std 610.12 1990, putem spune că cerința există dacă există o problemă reală de la care porneşte. În caz contrar cerința nu există.

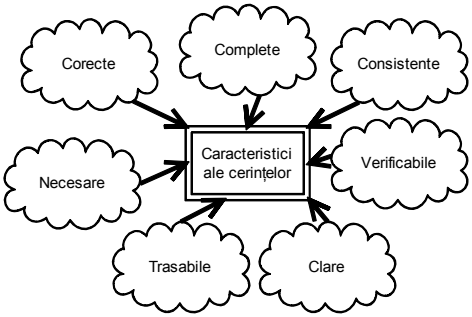

Fig. 3 Caracteristicile cerințelor

Această caracteristică este extrem de utilă pentru gestiunea cerințelor. Problema din spatele cerinței fiind, în mod necesar, un fragment din problemele proiectului. Doar pe baza problemei acesteia vom putea şti dacă o cerință se încadrează sau nu în proiect. Pentru a demonstra că o cerință este necesară este nevoie de existența unei legături către cerințele de pe nivelul imediat superior.

▪ **Corectă.** Atunci când spunem că o cerință trebuie să fie corectă, ne apropiem deja de a doua parte a definiției cerințelor din standard (sau mai degrabă le cuprindem pe amândouă). O cerință este corectă dacă fațeta denumită *soluție* este, nimic altceva decât soluția corectă la problema dată. În general pentru a determina corectitudinea unei cerințe este necesar să se facă referire la cerințele de pe nivelul superior.

▪ **Completă.** O cerință este completă dacă reprezintă o soluție compactă pentru rezolvarea problemei. Deşi cazurile de incompletitudine sunt greu de descoperit, organizarea de întâlniri formale cu participarea tehnicienilor din echipa de dezvoltare, de obicei dă rezultate.

▪ **Consistentă.** O cerință este considerată consistentă dacă nu intră în contradicție cu alte cerințe.

▪ **Verificabilă.** O cerință este verificabilă (testabilă) dacă permite realizarea validării fără echivoc a soluționării ei prin măsurare sau testare.

▪ **Clară** (fără ambiguități). O cerință poate fi considerată lipsită de ambiguități atunci când poate fi interpretată într-un singur fel. Dacă mai mulți cititori înțeleg lucruri diferite atunci cerința este ambiguă. Pentru a ține sub control fenomenul existenței ambiguităților trebuie organizate întâlniri pentru clarificarea tuturor specificațiilor. De asemenea, specificațiile vor fi folosite ca sursă primară pentru crearea planurilor de teste şi a manualului de utilizare.

▪ **Trasabilă.** Trasabilitatea se referă la posibilitatea de a reface traseul pe care îl urmează o cerință pornind de la solicitarea inițială a unui reprezentant al clientului. Acest mod de abordare asigură informația care justifică existența sau nu a cerinței, precum şi posibilitatea de a reface drumul pe care a apărut o cerință, atunci când apar dubii asupra acesteia, asupra sursei sau asupra rațiunii ei [2].

4. Aspecte cheie privind cerințele

Aspectele cheie care trebuie luate în considerare atunci când utilizăm cerințele sunt:

- identificarea părților interesate,
- separarea scopului de mijloace,
- identificarea cerințelor cheie,
- cuantificarea succesului şi a eşecului,
- înțelegerea trecutului şi a viitorului,
- termenele de livrare a cerințelor,
- evitarea „capcanelor cu ambiguități”,
- tratarea cerințelor complexe,
- permite cerințelor să evolueze.

● ***Identificarea părților interesate.*** Părțile interesate de un anumit sistem sau produs sunt acele persoane sau companii care interacționează cu sistemul/produsul sau care pot avea impact asupra

lui. Sistemul include utilizatorii sistemului, susținătorii, finanțatorii, managerii, dezvoltatorii, criticii şi alte părți interesate.

● ***Separarea scopului de mijloace.*** Este important să distingem „scopurile cerințelor" de „mijloacele lor". „Mijloacele" reprezintă ideile de design alese: arhitectură, tehnologii, strategii şi alte sinonime.

● ***Identificarea cerințelor cheie.*** Trebuie să identificăm cerințele părților interesate care sunt „vitale", „profitabile" sau „cele mai riscante" pentru sistem. Cerințele cheie au cel mai mare impact asupra părților critice interesate de sistemul de valori şi de costuri.

● ***Cuantificarea succesului şi a eşecului.*** Cerințele de necesitate trebuie înțelese în termeni de succes şi de eşec. Trebuie să ne asigurăm că avem valori numerice cuantificate şi specificate pentru fiecare dintre performanțele şi resursele atribuite. Trebuie să cunoaştem toate obiectivele („care este scopul") şi constrângerile („limitele pe care trebuie să le respectăm") vitale pentru sistemul de proiectare şi pentru managementul proiectului. Avem nevoie să înțelegem exact care este nivelul aşteptat de realizare şi de design. În mod specific: avem nevoie să ştim când am atins nivelul de cerințe necesar. Atingerea fiecărui nivel planificat este un succes „parțial". Proiectul este „de succes" atunci când nivelurile de succes sunt atinse pentru toate obiectivele de performanță şi se încadrează în buget.

- Eroare: Trebuie să specificăm nivelurile pe care le atingem în ordine pentru a evita unele tipuri de eşecuri ale părților interesate (cum ar fi „nu reuşesc să ating cota de piață dorită").

- Supraviețuirea: Trebuie să stabilim şi să specificăm limitele numerice care clasifică proiectul ca fiind un eşec total. Părțile interesate cunosc cerințele minime de supraviețuire, acestea sunt constrângeri exprimate utilizând parametrii de supraviețuire.

- Potențialul: Este util să păstrăm o înregistrare a cerințelor dorite, dar necontractate şi nerafinate. Cunoscând acestea, atunci când nu putem livra imediat proiectul avem cheia să livrăm primul atunci când aceasta va deveni posibil. (Acestea sunt specificate utilizând parametrii, *Oferă* - o provocare deliberată de inginerie stabilită pentru inginerii de sistem - şi *Doresc* - o expresie a nivelelor la care părțile interesate visează).

● ***Înțelegerea trecutului şi a viitorului.*** Avem nevoie să înțelegem contextul cerințelor părților interesate şi care sunt „etaloanele recente" ale nivelurilor de performanță existente în sistemul actual şi sistemul concurențial.

● ***Termenele de livrare a cerinţelor.*** Pentru a evalua dacă cerinţele specificate includ în mod adecvat „condiţiile de timp”, ar trebui să răspundem la câteva întrebări, cum ar fi:

- Cât de devreme trebuie să fie furnizate beneficiile acestui sistem?
- Cerinţele sunt specificate numai pentru nevoile pe termen scurt?
- Sunt nerealiste investiţiile pe termen lung?

● ***Evitarea "capcanelor cu ambiguităţi".*** Câteodată, cerinţele sunt foarte „generale” şi nu există o idee clară despre ceea ce este necesar să facem. De exemplu: „mărirea securităţii”, „facem sistemul mai uşor de utilizat” şi „să ofere un avantaj competitiv”. Problemele din cauza cerinţelor vagi vor apărea mai târziu în mod inevitabil, pentru că toată lumea crede că ceea ce e „general” înseamnă de fapt ceea ce este diferit. Lipsa unei definiţii exacte înseamnă că diferenţele de opinie nu sunt confruntate într-o etapă timpurie sau în timpul specificaţiilor. Nimeni nu a stabilit cu adevărat cerinţele exacte şi nimeni nu face nimic în legătură cu aceasta. Această problemă de "cerinţe formulate ambiguu" trebuie să fie abordată atât din punctul de vedere al comunicării cât şi al acţiunii. Trebuie clarificate toate cerinţele cheie. Frecvenţa şi livrarea timpurie permit părţilor interesate interacţiunea şi corecţia cerinţelor şi modelelor. Relevanţa proiectului nostru trebuie să verifice: realizarea timpurie, măsurabilitatea şi frecvenţa.

● ***Tratarea cerinţelor complexe.*** Ambiguitatea este o capcană deoarece „permite diferite interpretări”. O capcană complet diferită este să pierdem controlul proiectului, deoarece operăm cu prea puţine cerinţe detaliate. Gradul de detaliu specific este dependent de mărime şi ceea ce încerci să controlezi, precum şi de gradul de risc pe care eşti dispus să-l accepţi. Trebuie să ne menţinem atenţia adânc înrădăcinată în câteva cerinţe cheie, în timp ce ne asigurăm că avem un fundal adecvat de detaliere care să permită un control suficient de riguros. Putem face acest lucru specificând un set de cerinţe complexe („Top zece”) şi, apoi împărţindu-le pe fiecare dintre ele în elemente componente detaliate. Avem posibilitatea de a crea oricâte puncte de vedere utile avem nevoie (cum ar fi „riscuri”, „blocaje” şi „progrese”), cu detaliile adecvate în scopul gestionării proiectului.

● ***Permite cerinţelor să evolueze.*** Cerinţele se schimbă mereu şi nu este nici o cale să le oprim! Ca să realizăm un nou proiect, beneficiem de noi descoperiri în care cerinţele ne sunt utile. Părţile interesate, vor învăţa devreme din experienţele lor folosind un nou sistem, ceea ce ei chiar îşi doresc. Cerinţele economice se vor schimba inevitabil în timp, ca răspuns la mediul de afaceri intern şi extern. Din cauza acestor motive, cerinţele trebuie să fie lăsate să evolueze în

timpul proiectului, şi în timpul de viață al sistemului. Este esențial să păstrăm cerințele specifice realiste în timp. Putem oricând să alegem design-ul, construcția sau testarea unei anumite versiuni a cerințelor, ignorând orice alte actualizări.

5. Concluzii

■ Termenul de „cerință" nu a fost universal acceptat, el având mai multe definiții. Cercetătorii domeniului ingineriei cerințelor au încă păreri împărțite cu privire şi la clasificarea tipurilor de cerințe.

■ Trebuie să construim o legătură între cerințe, modele, planuri ale proiectului şi părțile interesate, deoarece acest lucru va face mai sigură evoluția şi modificarea cerințelor, în timp ce noi vom fi capabili să identificăm posibilele erori, efecte necunoscute şi să recunoaştem cele mai bune modificări [5].

BIBLIOGRAFIE

[1] * * * IEEE Std 610.12 1990.
[2] * * * http://www.techit.ro/analiza_software1.php
[3] Young, R.R., *The requirements engineering handbook*, Editura Artech House, ISBN 1580532667, 9781580532662, 2003.
[4] Brackett, J. W., *Software Requirements*. SEI Curriculum Module SEI-CM-19-1.2, Carnegie Mellon University, Software Engineering Institute,1990.
[5] Gilb,T., Brodie, L., *Competitive engineering: a handbook for systems engineering, requirements engineering, and software engineering using Planguage*, Editura Butterworth-Heinemann, ISBN 0750665076, 9780750665070, 2005.

Drd. Ing. Ec. Lavinia – Gabriela SOCACIU
Universitatea Tehnică din Cluj-Napoca, membru AGIR
e-mail: Lavinia.SOCACIU@muri.utcluj.ro
Prof. Dr. Ing. Ioan BLEBEA
Universitatea Tehnică din Cluj-Napoca, membru AGIR
e-mail: blebea@muri.utcluj.ro