Principii pentru evaluarea atributelor calitative din cadrul unei arhitecturi software

Lect. dr. Liviu CIORA, prep. Ion BULIGIU
Catedra de Informatica Economica, Facultatea de Stiinte Economice,
Universitatea din Craiova

Software quality is the degree to which software possesses a desired combination of attributes (e.g. reliability, interoperability). In this paper we describe a few principles for analyzing a software architecture to determine if it exhibits certain quality attributes. We show how analysis techniques can provide a base for performing software architecture evaluation.
Keywords: *software quality, attributes, architecture.*

Calitatea software este gradul în care software-ul contine combinatia de atribute dorita (de exemplu siguranta, interoperabilitate, fiabilitate) (IEEE-1061). Alaturi de cost, calitatea este o caracteristica orientata utilizator pentru un produs software.

Maturitatea procesului nu conduce automat la o calitate a acestuia. Calitatea unui software necesita tehnologii de previzionare si atribute de control al calitatii, ajunse la maturitate. Daca se înregistreaza o scadere a tehnologiei, chiar o organizatie matura va avea dificultati în crearea de atribute cu o performanta previzionabila, care dispun de siguranta sau alta atribute.

Calitatea si costul nu sunt independente. O calitate scazuta va afecta costul deoarece software-ul va avea nevoie de operatiuni de corectare, rescriere a codului si chiar o reproiectare pentru a satisface cerintele. Supradimensionarea costului este deseori întâlnita datorita nedescoperirii la timp a problemelor existente, pâna în faza de integrare a sistemului.

De aceea se va trage concluzia ca este mai eficienta din punct de vedere al costului detectarea potentialelor probleme calitative ale software-ului înca din primele faze ale dezvoltarii produsului software. Astfel, arhitectura software-ului va constitui principalul punct pentru detectarea defectiunilor ce afecteaza calitatea produsului; crearea arhitecturii software-ului este momentul în care trebuie sa înceapa analiza calitatii, iar arhitectura software este obiectul analizei.

În continuare vom descrie câteva principii pentru analiza unei arhitecturi în vederea determinarii daca aceasta întruneste atributele de calitate asteptate: identificarea relatiei dintre sistemul creat si mediu, identificarea arhitecturii software, identificarea resurselor hardware alocate pentru componentele software si analiza informatiilor culese prin aplicarea principiilor anterioare.

Pentru a ilustra principiile folosite în evaluarea atributelor orientate pe arhitectura, vom utiliza un sistem în care trei componente ale procesului furnizeaza date de intrare provenite din mediul de sistemului si rezultatele vor fi transmise unei a patra componente. Ultima componenta va trimite rezultatele înapoi catre mediu.

Se vor utiliza trei atribute ale calitatii:

- siguranta – probabilitatea ca sistemul sa furnizeze continuu date de iesire de-a lungul unui interval de timp specificat;
- latenta – intervalul de timp între sosirea unei date de intrare si trimiterea rezultatului corespondent catre mediu;
- productivitatea – rata iesirilor din sistem.

Tehnica acumularii informatiilor

Calitatea este relativa la sistemul studiat. O evaluare a calitatii trebuie sa ia în consideratie pe lânga studiul sistemului si mediul în care se afla sistemul (care îl înconjoara). Sistemul poate fi descompus si oricare componenta a sa poate fi subiect al analizei. Tot ceea ce se afla în exteriorul componentei studiate va constitui "mediul în-

conjurator" al acesteia. De aceea, mediul este strict relativ la sistemul evaluat.
Un sistem este format din componente software si componente hardware, astfel încât se vor face diferentieri pentru a concentra analiza catre arhitectura componentelor software, precum si arhitectura software a întregului sistem.
Se vor folosi termenii de subsistem si componenta pentru a defini elementele din care este compus un sistem; un subsistem este compus la rândul sau din alte subsisteme numite componente – o componenta nemaiputând fi descompusa.
Sistemul si mediul ce îl înconjoara intra în relatie, ambele având cerinte unul fata de celalalt si amândoua au obligatii pentru a îndeplini aceste cerinte (cerintele unei parti trebuie sa rezulte în urma obligatiilor celeilalte parti), relatie pe care o vom numi contract, care va fi guvernat de specificatii precise, similar cu contractele încheiate între doi parteneri. Aceste contracte vor contine obligatii mutuale (preconditii), beneficii (postconditii) si constrângeri (invariante).
Scenariile, punctajele si chestionarele sunt tehnici calitative aplicabile identificarii contractului, alocarii de resurse hardware si arhitecturii software. Pe parcursul unei evaluari, vor fi utilizate una sau mai multe din aceste tehnici în functie de sistem si de mediu. Se disting doua categorii de elemente utilizate în identificarea informatiilor necesare analizei: conceptele fundamentale din proiectare si criteriile împreuna cu specificatiile.

- Conceptele fundamentale din proiectare sunt principiul operational si configuratia normala. Principiul operational descrie modul de functionare a sistemului: defineste sistemul si furnizeaza criteriile ce trebuiesc îndeplinite pentru ca acesta sa functioneze corect. Configuratia normala este reprezentata de aranjamentul structural al sistemului care face ca acesta sa functioneze în conformitate cu principiul operational.
- Criteriile si specificatiile sunt scopurile cantitative ale proiectarii, exprimate în termeni tehnici, caracteristice sistemului si utilizarii acestuia.

Astfel, în proiectarea unui sistem se distinge un circuit recursiv ce necesita precizie în momentul precizarii specificatiilor tuturor componentelor sistemului si asamblarea acestor componente în sistemul final.
Constructia unei arhitecturi se va distinge ca un proces care poate fi descompus într-un set de activitati – cheie:
- Caracterizarea sistemului – acest pas expliciteaza caracteristicile care traseaza designul arhitectural al sistemului. Un chestionar va ajuta concentrarea activitatii si localizarea resurselor necesare, accentuând consideratii fundamentale de proiectare: memorie utilizata, timpi de acces a datelor, dimensiune, identificarea fluxurilor critice, timp de raspuns etc.;
- Definirea entitatilor conceptuale – se definesc si se descriu entitatile conceptuale ale arhitecturii si relatiile dintre ele;
- Definirea teoretica operationala – se va preciza modul de functionare a întregului sistem prin definirea fluxurilor ce controleaza întreaga arhitectura, acoperind toate modurile în care va opera sistemul, cum ar fi starea normala de operare, lansarea în executie si procedurile de initializare, restartare, operatii de depasire a momentelor critice si închiderea sistemului.

Tehnica analizei atributelor specifice

În functie de atributele de calitate care intereseaza, în evaluare se pot folosi diferite tehnici calitative si cantitative pentru realizarea analizei.
Analiza riscului si sigurantei este deseori utilizata într-un mare numar de metode practice aplicate în procesul dezvoltarii software. Pe aceeasi linie, previzionarea calitativa a defectiunilor este orientata spre identificarea, clasificarea si ordonarea modurilor de functionare defectuoasa ori la identificarea evenimentelor sau combinatiilor de evenimente care conduc la efecte nedorite. Previzionarea cantitativa a erorilor (în special în cadrul modelarii si testarii) studiaza estimarile probabilistice ale sigurantei sistemului, astfel încât modelele

de studiere a sigurantei unui sistem se bazeaza pe efectuarea de previziuni care au la baza datele culese din evenimente anterioare de functionare incorecta ale sistemului.

Privitor la siguranta sistemului, elementul supus analizei este întâmplarea sau hazardul – care este luat în calcul prin crearea unei liste de posibile actiuni care pot aparea întâmplator, înainte de finalizarea sistemului. Prin identificarea actiunilor întâmplatoare, procesul de analiza va fi utilizat la realizarea planului de diminuare a riscului. Acest tip de analiza include analiza erorilor, analiza evenimentelor, analiza modurilor de functionare incorecta si a efectelor derivate si analiza punctelor critice.

Tehnicile de analiza folosite în cadrul studierii securitatii sistemului include metode formale (se verifica daca sistemul proiectat întruneste cerintele si specificatiile prevazute), analiza penetrabilitatii (scenarii standard de posibile atacuri pentru a determina daca sistemul este elastic din acest punct de vedere) si analiza canalelor de siguranta (determinarea capacitatii fiecarei magistrale secundare de date identificate în sistem).

Un pas important în evaluarea sistemului îl constituie identificarea cerintelor si obligatiilor sistemului, identificare într-un *contract*. În aceasta analiza se vor întâlni mai multe tipuri de atribute: atribute masurate prin intermediul actiunilor sistemului (de exemplu latenta, disponibilitate), atribute masurate cu ajutorul activitatilor de inspectare (de exemplu cuplajul, coeziunea) si atribute determinate de activitatea utilizatorilor (de exemplu timpul de executare a unei operatiuni).

În functie de atributele studiate, mediul în care se afla sistemul poate fi unul operational (retele, grupuri de utilizatori) sau un mediu de dezvoltare (firme de software).

Cerintele si obligatiile sistemului pot fi concretizate în scenarii. Aceste scenarii sunt scurte descrieri ale unei cerinte, ale unei situatii operationale, ale unei modificari în sistem etc.

Exemple de scenarii identificate dintr-un contract:

- Mediul depinde de datele din sistem si are ca cerinta un timp de cadere a sistemului (system down time) mai mic de 1 ora într-un an (cerinta minimului disponibil)
- Rata de esec a sistemului sa fie mai mica de un esec/luna (cerinta sigurantei minime)
- Mediul asteapta ca timpul de raspuns sau latenta sistemului sa fie mai mica de 100 milisecunde (cerinta latentei minime)
- Sistemul trebuie sa proceseze mai mult de 20 de evenimente de intrare /min. (cerinta de productivitate)
- Sistemul poate rezista fluxului excesiv de evenimente de intrare (cerinta de securitate împotriva acestui tip de atac)
- Mediul cere ca sistemul sa permita upgrad-area procesorului sau a retelei (modificabilitatea resurselor)
- Mediul furnizeaza date de intrare cu timpi de distribuire exponentiali cu rata de transfer λ.

Identificarea arhitecturii software

O arhitectura software este caracterizata de o combinatie particulara de componente software si conexiunile dintre aceste componente. Principiul identificarii arhitecturii software presupune ca evaluarea sa identifice componentele si conexiunile din sistem.

Sunt întâlnite doua tipuri de conexiuni: *structura* caracterizata de legaturile dintre componente ce formeaza fluxul de date si *comportamentul* care pe lânga semantica sistemului include si fluxul de control.

Structura. Vom studia un sistem (figura 1) format din trei componente (numite "participanti") care proceseaza datele de intrare provenite din mediul exterior sistemului si o a patra componenta, care are ca date de intrare rezultatele furnizate de "participanti". Cea de-a patra componenta are rol de interpretare a datelor intrate si închide fluxul de date prin trimiterea rezultatelor pe rol de feed-back catre mediu.

Comportamentul. Pentru realizarea unei sigurante cât mai mari, cei trei "partici-

panti" efectueaza operatii redundante (dar nu neaparat identice), iar cea de-a patra componenta va alege varianta optima din datele furnizate de "participanti" si o trimite ca data de iesire a sistemului, catre mediu. Aceasta ultima componenta are rol decizional în sistem, astfel încât odata detectata o eroare la un participant, va proceda la ignorarea acestuia, continuând operarea cu participantii ramasi.

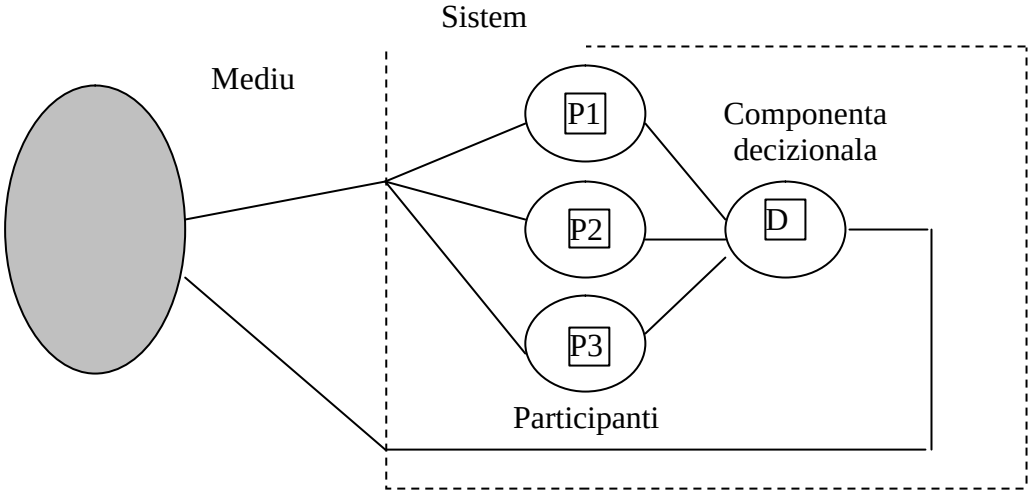

Fig.1. Sistem cu 3 componente

Sunt considerate astfel doua tipuri de comportament ale componentei decizionale:
1. Alegere prin majoritate – vor fi selectate doua din trei intrari percepute, altfel se vor selecta doua intrari din doua posibile, altfel procesul se închide (sistemul nu ia o decizie). Desi cei trei "participanti" primesc aceleasi date de intrare si se asteapta sa efectueze calcule cu aceleasi valori, nu vor folosi acelasi algoritm de calcul. Variante:
- alegere sincrona – componenta de decizie va efectua o preluare a datelor de intrare la fiecare interval T_V. La o astfel de preluare, aceasta componenta asteapta sa primeasca doua intrari identice. Un participant este ignorat daca va trimite datele nesincronizat cu momentul T_V sau daca va trimite o valoare gresita;
- alegere asincrona – componenta decizionala asteapta primirea datelor de intrare, dar are un interval de timp în care vor fi detectate intrarile lipsa.

2. Alegere preferentiala – componenta decizionala selecteaza participantul P1 daca P1 functioneaza sau selecteaza participantul P2 daca P2 functioneaza sau selecteaza participantul P3 daca P3 functioneaza sau procesul se închide (sistemul nu ia o decizie). Fiecare proces poate sa aiba semnificatie diferita referitoare la "functionarea" unui participant (ex. conditia detectiei de erori sau timp optim de raspuns). Variante:
- detectarea valorilor eronate – se va efectua un test de rezonabilitate pentru date sau datele de intrare dispun de indicatori de eroare;
- detectarea depasirilor de interval – componenta de decizie dispune de un test pentru depasirea timpului în care puteau fi primite datele.

Identificarea alocarii resurselor hardware

Principiul presupune ca evaluatorii sa colecteze informatii despre resursele de calcul, de stocare si comunicatie. Aceste informatii sunt necesare analizei deoarece ele sunt utilizate si partajate de catre componentele software, tinându-se cont si de faptul ca aceste resurse hardware sunt limitate si pot sa se defecteze – determinând calitatea generala a sistemului.
Sistemul considerat este implementat cu procesoare standard si se afla cuplat la o retea locala. Variante de arhitecturi:

- componente independente – fiecare componenta software este executata pe procesor separat;
- componente partajate – toate componentele software se vor executa pe acelasi procesor ca procese concurente, având urmatoarele variante:
  - prioritatea componentei decizionale este mai mare decât a participantilor;
  - prioritatea componentei decizionale este mai mica decât a participantilor.
- arhitectura mixta – participantii partajeaza acelasi procesor, însa componenta de decizie foloseste procesor separat.

Desi numarul schemelor de alocare a resurselor pare nelimitat, în realitate resursele hardware sunt disponibile într-o plaja destul de mica de oferte.

Identificarea resurselor hardware si alocarea lor pentru componentele software poate fi realizata cu ajutorul chestionarelor si listelor de optiuni prin care se vor preciza resursele (procesoare, memorie si unitati de stocare a datelor, magistrale, retele) care vor fi utilizate pe fiecare componenta software. Este foarte importanta identificarea resurselor partajate, care nu pot fi deduse din descrierea structurii si comportamentului.

Bibliografie:

- IEEE Standard 1061-1992 . Standard for a Software Quality Metrics Methodology. New York: Institute of Electrical and Electronics Engineers, 1992.
- Kazman, R. et al. "Scenario-Based Analysis of Software Architecture." IEEE Software 13, 6 (November 1996): 47-56.
- Barbacci, M.R., et al. Quality Attributes (CMU/SEI-95-TR-021). Pittsburgh, Software Engineering Institute, Carnegie Mellon University, 1995.