Metodologie de proiectare a unui sistem de gestiune si arhivare electronica a documentelor

Romeo Mihai PAVELESCU, p_mihai@hotmail.com

The best way to manage large collections of documents is with a Document Management System. When faced with a complex problem, often we first need to step back and see the obvious. And that's certainly the case with document management. There are many oferts of software on this field in the market place but no methodology widely accepted. In fact, every software company keeps secret their technology about designing Document Management Systems. These guidelines provide a way to design and implement complex systems, such as Document Management Systems.

***Keywords**: document management, workflow, system design, object oriented design, UML, archiving systems, documents, files.*

Deoarece, în prezent, nu exista o metodologie, unanim acceptata, specializata în realizarea sistemelor de gestiune automata a documentelor, în lucrarea de fata am adaptat metodologia de dezvoltare a sistemelor software complexe, descrisa de Bernd Bruegge & Allen H. Dutoit în lucrarea *Object-Oriented Software Engineering – Conquering Complex and Changing*. Alegerea facuta se bazeaza pe faptul ca aceasta combina, cu mult succes, metode utilizate în mai multe metodologii consacrate în dezvoltarea orientata-obiect a aplicatiilor (precum O.O.S.E., O.M.T., Booch, Catalysis) si ca foloseste U.M.L. pentru documentarea activitatii de realizare a sistemelor software (considerat astazi ca standard industrial).

Etapele principale ale metodologiei sunt: extragerea cerintelor sistemului; analiza de sistem; proiectarea de sistem; proiectarea obiectuala; implementarea sistemului.

Aceste activitati vor avea ca rezultat descrierea detaliata a sistemului prin prisma a trei modele:

- *modelul functional* reprezentat în U.M.L. prin diagramele de situatii posibile care descriu sistemul din punctul de vedere al utilizatorului;
- *modelul obiectual* reprezentat în U.M.L. prin diagrame de clase care descriu structura sistemului în termeni de obiecte, atribute, asocieri si operatii;
- *modelul dinamic* reprezentat în U.M.L. prin diagrame de secventa, de stare si de activitate care descriu comportamentul intern al sistemului.

Extragerea cerintelor sistemului include identificarea actorilor, scenariilor, situatiilor posibile si relatiilor între actori si situatiile posibile.

Actorii reprezinta entitati externe care interactioneaza cu sistemul. În cazul unui sistem de management al documentelor actorii pot fi:

- angajatii firmei (utilizatori ai aplicatiei);
- clientii firmei (prin solicitari adresate sub forma de documente);
- gestionarul arhivei definitive (manipuleaza discurile magneto-optice);
- administratorii de sistem (care efectueaza operatii de întretinere curenta a aplicatiei).

Scenariile reprezinta descrieri în limbaj natural a ceea ce oamenii trebuie sa realizeze prin intermediul calculatorului pentru a-si îndeplini sarcinile ce le revin. Totodata, mai multe scenarii diferite sunt concentrate de dezvoltatorii de aplicatii, prin generalizarea functionalitatii lor, într-o ***situatie posibila***. În cazul unui sistem de management al documentelor situatiile posibile pot fi: înregistrarea unui document; înregistrarea unui dosar; transferul unui document; transferul unui dosar; arhivarea unui document; dezarhivarea unui document; arhivarea unui dosar; dezarhivarea unui dosar etc.

Relatiile ce pot fi identificate între mai multe ***situatii posibile*** sunt exemplificate în figura 1.

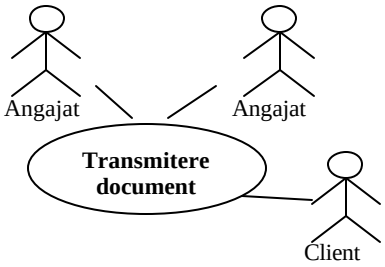

Fig. 1. Exemplu de descriere a relatiilor existente între mai multe situatii posibile

Analiza de sistem este etapa în care ***scenariile*** si ***situatiile posibile,*** identificate si rafinate în cadrul etapei "Extragerea cerintelor", sunt transformate într-un model de analiza. Activitatile desfasurate în cadrul analizei de sistem sunt: identificarea obiectelor entitate; identificarea obiectelor de interfata; identificarea obiectelor de control; modelarea interactiunilor între obiecte; identificarea asocierilor între obiecte; identificarea atributelor obiectelor; modelarea comportamentului cu ajutorul diagramelor de stare; modelarea generalizarii relatiilor între clase; revizuirea modelului de analiza.

Obiectele entitate ale sistemului vor fi identificate, printr-un algoritm euristic (Abbott, 1983), pe baza analizei limbajului natural în care au fost descrise ***situatiile posibile***. Astfel, ***obiectele entitate*** se regasesc printre substantivele ce apar în descrierea ***situatiilor posibile***, anume: document, dosar, destinatar, emitent, cont de utilizator, arhiva definitiva.

Obiectele de interfata reprezinta acele obiecte cu care actorii interactioneaza în scopul lansarii de actiuni sau introducerii de date ce vor fi utilizate de ***obiectele entitate*** sau de ***obiectele de control***.

În cazul unui sistem de management al documentelor au fost identificate urmatoarele ***obiecte de interfata***: Buton de înregistrare document; Buton de înregistrare dosar; Buton de vizualizare/modificare document; Aplicatie pentru vizualizare/modificare document; Fereastra de vizualizarea/modificarea proprietatilor documentului; Fereastra de vizualizarea/modificarea proprietatilor dosarului; Fereastra selectare fisiere de pe disc; Fereastra selectare utilizator; Fereastra cu dosarele din contul curent; Mesaj de dialog; Buton de transmitere document; Buton de transmitere dosar; Buton de arhivare document; Buton de arhivare dosar; Buton de dezarhivare document; Buton de dezarhivare dosar.

Obiectele de control sunt responsabile pentru coordonarea ***obiectelor interfata*** si a celor ***entitate***. ***Obiectele de control*** sunt foarte strâns legate de ***situatiile posibile,*** fiind create la initializarea ***situatiilor posibile*** si activând pâna la terminarea acestora.

În urma analizei ***situatiilor posibile*** au rezultat urmatoarele ***obiecte de control***: Controlul înregistrarii documentului; Controlul înregistrarii dosarului; Controlul vizualizarii/modificarii documentului; Controlul transmiterii documentului; Controlul transmiterii dosarului; Controlul arhivarii documentului; Controlul arhivarii dosarului; Controlul dezarhivarii documentului; Controlul dezarhivarii dosarului.

Modelarea interactiunilor între obiecte se realizeaza cu ajutorul diagramelor de secventa. Acestea arata cum evolutia situatiilor posibile se rasfrânge asupra obiectelor participante. În figura 2 este prezentat un exemple de diagrama de secventa care modeleaza interactiunile între obiecte sistemului.

Modelarea comportamentului cu ajutorul diagramelor de stare începe prin identificarea claselor de obiecte care au un comportament preponderent dinamic. În cazul unui sistem de management al documentelor clasele de obiecte cele mai dinamice sunt: document, dosar.

Pentru ***identificarea atributelor*** se examineaza adjectivele si frazele posesive. De obicei, pentru obiectele entitate, orice proprietate care trebuie memorata constituie un potential atribut. Astfel, atributele identificate vor fi prezentate în clase dupa modelul din figura 3.

Pentru fiecare din clasele identificate se construieste câte o diagrama de stare folosind U.M.L. (figura 4).

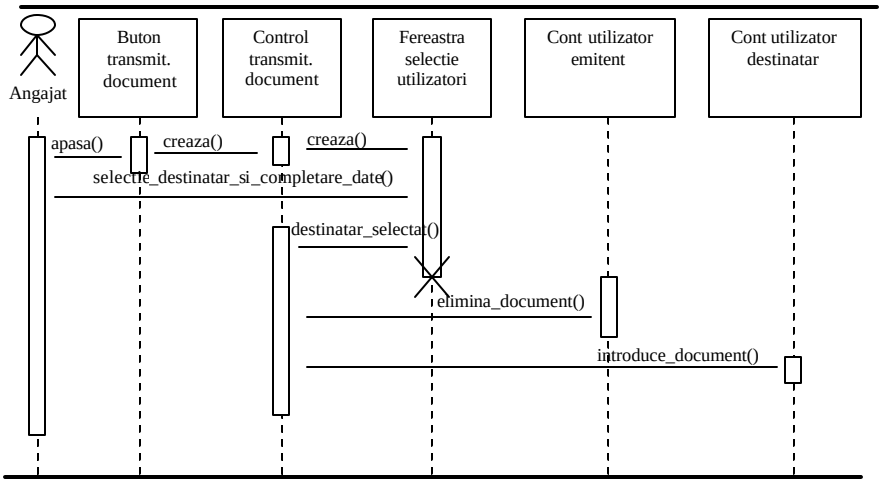

Fig. 2. Exemplu de diagrama de secventa pentru situatia posibila *Transmitere document*

Proiectarea de sistem constituie primul pas în procesul de construire efectiva a sistemului. În cadrul acestei etape atentia se focalizeaza asupra proceselor, structurilor de date, a componentelor software si hardware ce concura la realizarea sistemului.

DOCUMENT
nr. de înregistrare al documentului (unic) subiect: String stare: String nrpagini: Numeric dimensiune: Numeric extensie: String

Fig. 3. Descrierea clasei ***Document***

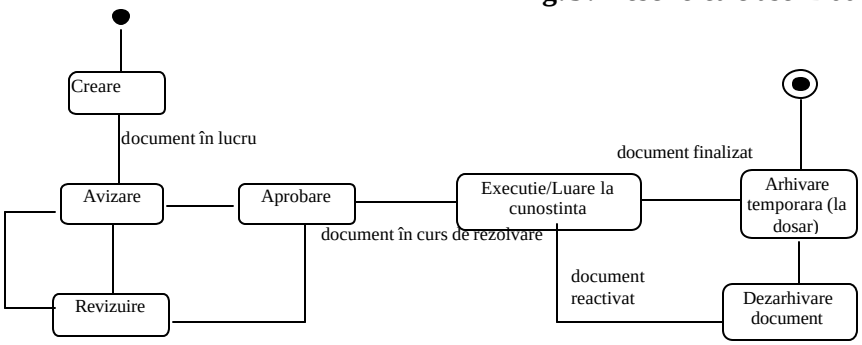

Fig. 4. Exemplu de diagrama de stare U.M.L. pentru clasa ***Document***

Modelarea generalizarii relatiilor între obiecte este utilizata pentru eliminarea redundantelor din modelul de analiza. Daca doua sau mai multe clase partajeaza o parte a atributelor sau comportamentul lor este similar, acestea se consolideaza într-o superclasa. De exemplu, prin generalizare poate rezulta clasa ***Utilizator*** (figura 5).

Identificarea obiectivului proiectarii înseamna identificarea calitatilor pe care sistemul trebuie sa se focalizeze. O parte a acestor calitati se regasesc în cerintele speciale obtinute în etapa **Extragerea cerintelor**. Celelalte vor fi obtinute din discutii cu beneficiarii sistemului.

Identificarea subsistemelor din cadrul **proiectarii de sistem** se aseamana foarte mult cu identificarea obiectelor efectuata în timpul **analizei de sistem**.

În acest moment al proiectarii dispunerea în spatiu a firmei are un rol foarte important. Daca luam ca ipoteza de lucru existenta mai multor filiale ale firmei dispuse la distante relativ mari între ele (figura 6), atunci vom fi nevoiti sa identificam o solutie ieftina pentru comunicarea între acestea. Deci, o prima descompunere în subsisteme se va realiza în

asa fel încât fiecare filiala sa contina un subsistem informatic. Pentru simplificarea proiectarii vom considera ca aceste subsisteme sunt identice din punct de vedere al dezvoltarii si difera doar prin configurarea fiecaruia în parte.

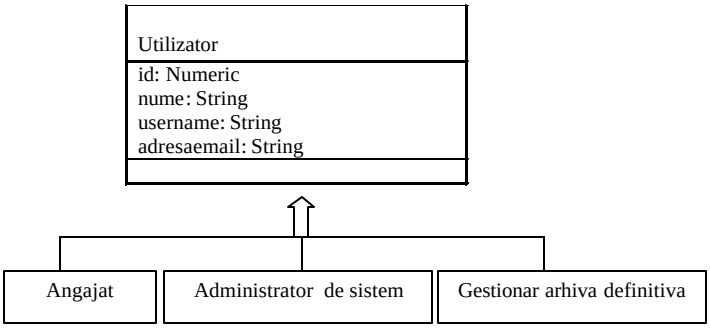

Fig. 5. Exemplu de generalizarea relatiilor între obiecte

Totodata, în cadrul unei filiale se pot identifica urmatoarele subsisteme:

- *subsistemul client* care este responsabil cu gestionarea interfetei utilizator;
- *subsistemul de gestiune a documentelor* care este responsabil cu manipularea documentelor de la creare pâna la arhivare temporara;
- *subsistemul de gestiune a dosarelor nearhivate* care este responsabil cu manipularea dosarelor de la creare pâna la arhivare definitiva;
- *subsistemul de gestiune a dosarelor arhivate* care este responsabil cu stocarea în conditii optime a dosarelor arhivate si cu regasirea acestora;
- *subsistemul de comunicatie cu alte filiale si cu clientii* care este responsabil cu transmiterea documentelor si dosarelor între filiale sau în cazul documentelor catre clienti.

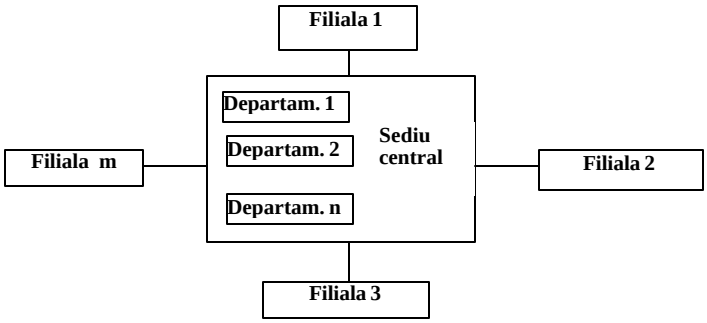

Fig. 6. Organizarea teritoriala a firmei (ipoteza de lucru)

Diagrama de clase rezultata în urma descompunerii în subsisteme este prezentata în figura 8.

Identificarea componentelor hardware si software necesare consta în alegerea platformei de lucru si a configuratiei hardware. Considerând ca structura sistemului de management trebuie sa fie modulara (ceea ce permite o usoara depanare si implica o adaptabilitate ridicata la schimbari) si alegând strategia de proiectare client-server a sistemului am identificat 3 tipuri de calculatoare (figura 7): **client** – folosite de utilizatori pentru accesarea functiilor sistemului; **server de management** – necesare gestionarii dosarelor si documentelor în cadrul unei filiale; **server de comunicatie** – cu rolul de a transporta documentele si dosarele între filiale.

Definirea datelor persistente consta în identificarea obiectelor care trebuie sa fie persistente. Dupa aceea se alege modalitatea de realizare a managementului stocarii obiectelor. În cadrul lucrarii de fata obiectele vor fi stocate în baze de date orientate-obiect cu exceptia continutului documentelor care vor fi stocate în *fisiere plate(gestionate prin intermediul sistemului de operare)* organizate în *magazii de documente.*

Controlul accesului la sistem se defineste prin descrierea modalitatilor prin care fiecare actor este autentificat înainte de utilizarea sistemului si a modalitatilor de criptare a datelor.

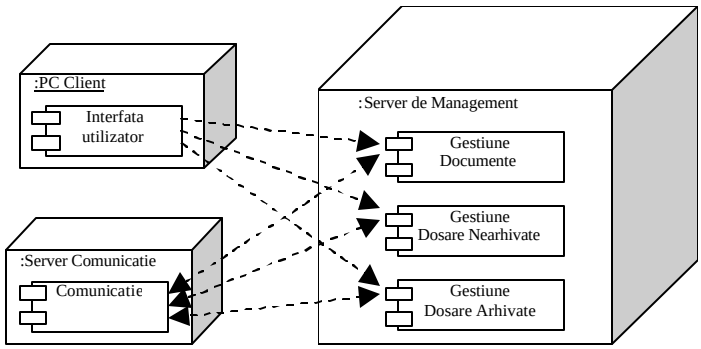

Fig. 7. Maparea subsistemelor la hardware

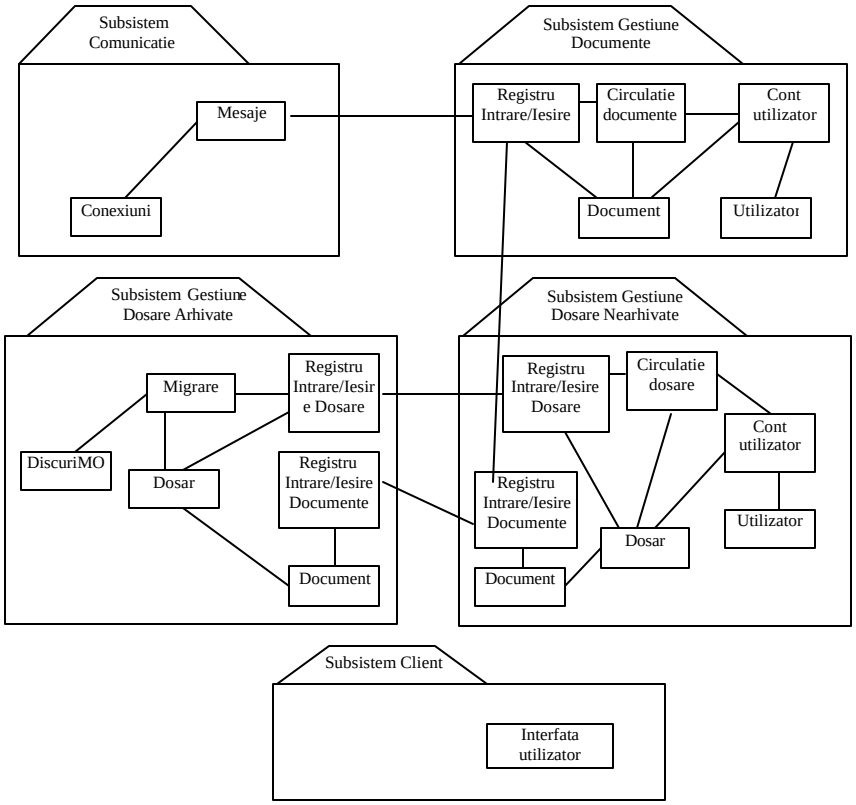

Fig. 8. Exemplu de descompunere în subsisteme

Pentru simplificare autentificarea utilizatorilor se va baza pe numele de utilizator si parola cu care acestia sunt definiti în cadrul server-ului de baze de date. Documentele care circula în cadrul unei filiale nu necesita o protectie mai mare decât cea oferita de un sistem de operare din clasa C2 (ex. Windows 2000 Server). Documentele care circula între filiale vor fi criptate cu ajutorul unui server de certificare, care va permite, în mod transparent pentru utilizatori, criptarea si decriptarea acestora.

Proiectarea controlului global al fluxurilor se refera la stabilirea ordinii de executie a operatiilor. Astfel, toate subsistemele vor fi bazate pe mecanismul *procedure-driven control* (caz în care operatiile asteapta pentru introducerea de date ori de câte ori este necesar) cu exceptia subsistemului de comunicatie care va folosi mecanismul event-driven

control (caz în care fiecare este tratat imediat ce devine disponibil).

Identificarea conditiilor de extrem se refera la determinarea situatiilor în care sistemul este pornit, initializat si închis, tinând cont de eventualele caderi pe care acesta le poate avea, precum coruperea datelor. Astfel, vor fi identificate noi situatii posibile care trebuiesc incluse în modelul de analiza. În general, actorul în astfel de cazuri este administratorul de sistem.

În cazul unui SGAED utilizarea servere-lor de baze de date si de comunicatie, componente software ce trebuiesc alese cu atentie, simplifica aceasta activitate datorita mecanismelor de tranzactionare a operatiilor pe care acestea le includ. Astfel, sarcinile administratorului de sistem în cazul unor caderi de tensiune se reduc la utilizarea instrumentelor de administrare oferite de cele doua servere.

Proiectarea obiectuala are ca scop umplerea golului dintre obiectele rezultate în faza de **analiza** si platforma hardware/software alease pe durata **proiectarii de sistem** Aceasta include identificarea obiectelor periferice, ajustarea componentelor obtinute în faza de **proiectare de sistem** si descrierea exacta a fiecarei interfete a subsistemelor si a claselor.

Proiectarea obiectuala include patru grupuri de activitati:

- ***specificarea exacta a claselor***, din care fac parte activitatile: identificarea atributelor si operatiilor lipsa; specificarea tipurilor si vizibilitatii acestora; specificarea restrictiilor; specificarea exceptiilor;
- ***selectia componentelor***, din care fac parte activitatile: identificarea si ajustarea bibliotecilor de clase; identificarea si ajustarea cadrului de lucru al aplicatiei;
- ***restructurarea***, care contine activitatile: marirea reutilizarii; eliminarea dependintelor implementarii;
- ***optimizarea***, din care fac parte activitatile: reanalizarea cailor de acces; eliminarea obiectelor prin transformarea lor în atribute; memorarea rezultatelor calculelor complexe; temporizarea calculelor complexe.

Identificarea atributelor si operatiilor lipsa se realizeaza prin examinarea serviciilor pe care trebuie sa le asigure fiecare subsistem în paralel cu reanalizarea diagramei claselor obtinuta în cadrul proiectarii de sistem.

Specificarea tipurilor si vizibilitatii proprietatilor si metodelor rafineaza modelul obiectual în doua moduri:

- adauga la model detalii legate de spatiul valorilor care pot fi atinse de proprietati;
- leaga clasele si atributele de tipurile de date furnizate de mediile de dezvoltare a programelor.

Specificarea restrictiilor are scopul de a limita ambiguitatea modelului obiectual cât mai mult posibil.

Specificarea exceptiilor se refera la conditiile pe care utilizatorul trebuie sa le respecte înainte de a apela o anumita operatie. În cadrul acestei activitati se descrie mecanismul de detectare si tratare a unei erori în cadrul fiecarei operatii.

Identificarea si ajustarea bibliotecilor de clase este activitatea în care dezvoltatorul, luând ca ipoteza limbajul de programare în care va lucra, trebuie sa identifice bibliotecile de functii si clase ale acestuia pentru o cât mai eficienta reutilizare a codului sursa.

Identificarea si ajustarea cadrului de lucru al aplicatiei este o activitate de care depinde rapiditatea implementarii sistemului, deoarece aici se identifica eventualele aplicatii distribuite si componente ce pot fi integrate în sistem (ex. ActiveX, DLL, Microsoft MFC, baze de date tranzactionale).

Marirea reutilizarii codului sursa pune accentul pe utilizarea, identificarea, acelor clase similare care pot fi agregate. O parte a acestora sunt oferite de mediile de programare si o alta parte poate fi obtinuta din bibliotecile de clase personale (create de-a lungul carierei de programator).

Reanalizarea cailor de acces are ca scop identificarea cailor de acces ineficiente (ocolite). Acestea trebuiesc modificate în concordanta cu frecventa de parcurgere a lor. Tot în aceasta activitate se elimina clasele aparute ca urmare a unei modelari excesive.

Eliminarea claselor prin transformarea lor în atribute este necesara în conditiile în care în faza de proiectare de sistem si proiectare obiectuala au fost facute modificari importante asupra modelului rezultat în faza de

analiza, caz în care anumite clase pot fi reduse la nivelul de atribute.

Memorarea rezultatelor calculelor complexe este activitatea în care sunt identificate operatiile consumatoare de putere de calcul si a caror frecventa de executie este ridicata. În cadrul acestei activitati rezolvarea consta în memorarea rezultatului operatiei si utilizarea acestuia pe perioada în care rezultatul operatiei nu se poate schimba.

O alta solutie pentru economisirea puterii de calcul a sistemului consta în ***temporizarea operatiilor care efectueaza calcule complexe***. Astfel, operatiile vor fi executate numai atunci când efectiv este nevoie de rezultatul lor.

Bibliografie

[1] Berndt Bruegge & Allen H. Dutoit, *Object-Oriented Software Engineering; Conquering Complex Changing Systems*, Pretince Hall, 2000

[2] Commonwealth of Australia, *Impro-ving Electronic Document Manag ement*, 1996

[3] Bruce Silver Associates, *Workflow, Document Management, and Customer Interaction Technology*, www.brsilver.com, noiembrie 1998

[4] James Boyle, *A good Electronic Document Management System can bring together document storage, workflow, and indexing*, 1997