

DIAGrame UML PENTRU MODELAREA COMPORTAMENTULUI CLASELOR, A INTERACȚIUNII DINTRE CLASE ȘI A IMPLEMENTĂRII ACESTORA ÎN CAZUL UNUI SISTEM DE GESTIUNE PRIN METODA ABC A COSTURILOR UNEI ORGANIZAȚII

Liana Anica – Popa

Academia de Studii Economice, București

Rezumat: Limbajul de modelare UML (Universal Modelling Language) este utilizat cu succes astăzi în domeniul modelării sistemelor informatice. Acest articol continuă prezentarea soluției de proiectare UML propuse în numărul 3/2004¹ al revistei. Diagramele UML realizate vizează trei aspecte:

- comportamentul claselor: diagrama de activități, diagrama de stări;
- interacțiunea între clase: diagrama de colaborări; diagrama de secvențe;
- implementarea acestora: diagrama componentelor; diagrama de amplasare.

Fiecare diagramă oferă o perspectivă distinctă a sistemului proiectat.

Cuvinte cheie: UML, proiectare obiectuală, modelare orientată obiect, diagrama de activități, diagrama de stări, diagrama de colaborări, diagrama de secvențe, diagrama componentelor, diagrama de amplasare, gestiune, costuri, metoda ABC

1. Diagrama de stări

În cadrul acestui tip de diagramă se reprezintă stările prin care trece fiecare obiect, evenimentele care determină tranzițiile de la o stare la alta, precum și condițiile în care se realizează aceste tranziții. Se poate realiza pentru fiecare tip de obiect sau numai pentru tipurile ale căror stări prezintă o importanță deosebită.

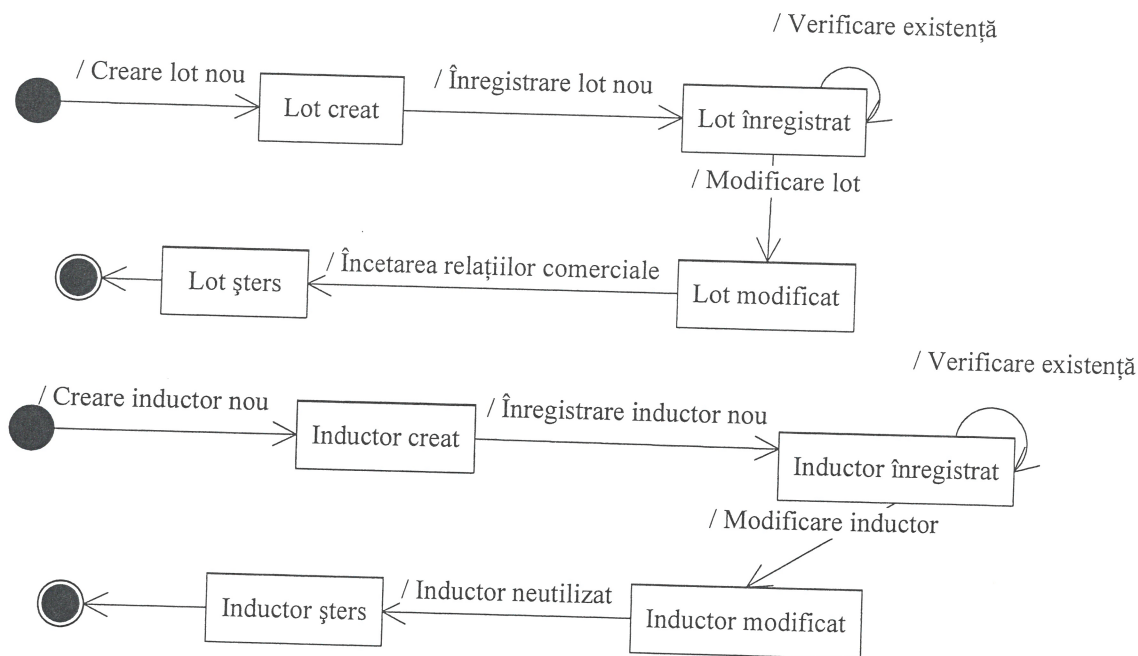


Figura 1 - Diagrama de stări a obiectelor din clasa Lot și Inductor

¹ Articolul din Revista Română de Informatică și Automatică, vol. 14, nr. 3/2004, se intitulează „Proiectare obiectuală UML a unui sistem de gestiune a costurilor unei organizații prin metoda ABC – diagramele cazurilor de utilizare și ale claselor”.

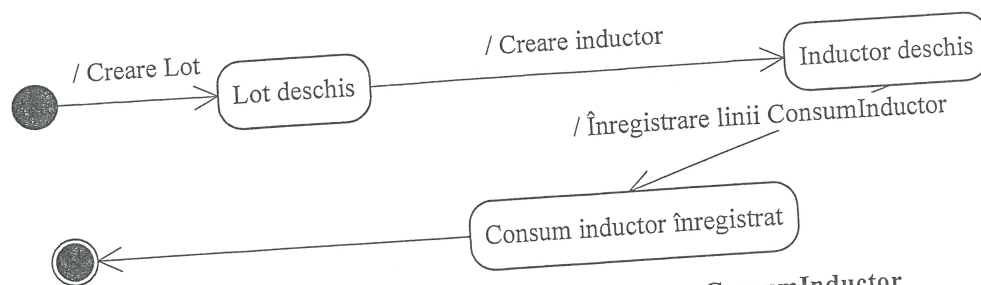


Figura 2 - Diagrama de stări a obiectelor din clasa ConsumInductor

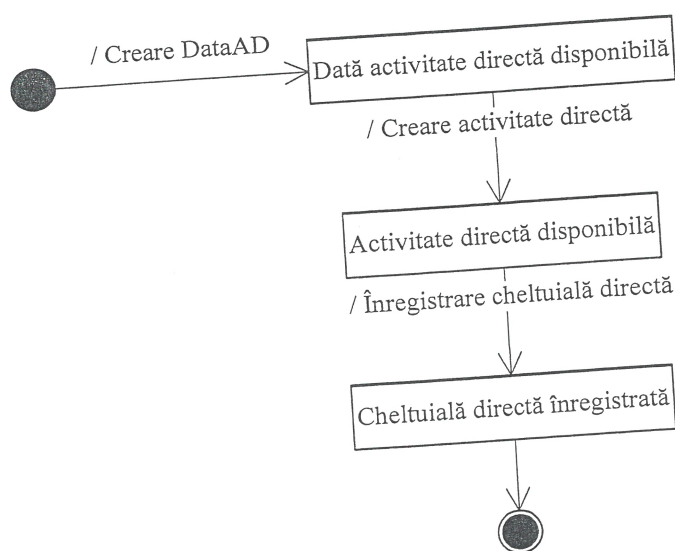
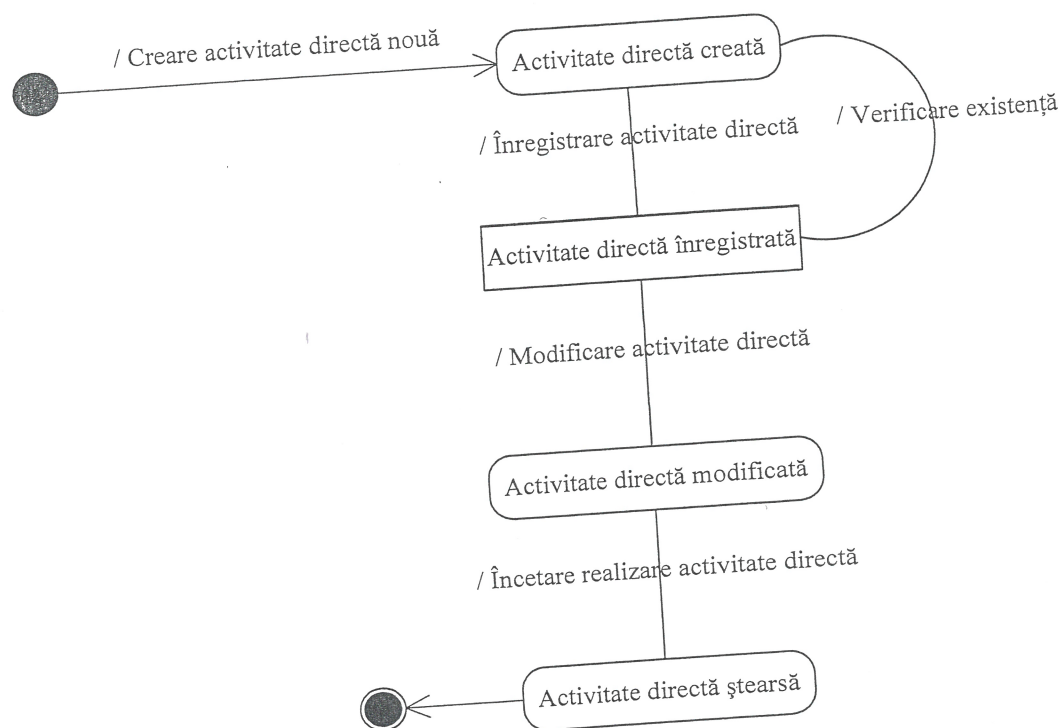


Figura 3 - Diagrame de stări ale obiectelor din clasele ActivitateDirecta și CheltuialaDirecta

2. Diagrama de activități

Diagrama de activități descrie modul de organizare a activităților, prezentând atât comportamente condiționale (puncte de decizie), cât și comportamente paralele (sincronizate cu ajutorul barelor de sincronizare). Un punct de decizie are o singură tranziție de intrare și mai multe tranziții de ieșire, condiționate astfel încât să se poată selecta una singură. Un punct de fuziune are o singură tranziție de ieșire și mai multe tranziții de intrare, indicând încheierea unui comportament condițional, determinat de un punct de decizie anterior.

Figura 11 prezintă o diagramă de activități pentru cazul de utilizare *Determinare cost total*, desfășurată pe două pagini conectate prin simbolul grafic .

3. Diagrama de colaborări

Între clase și obiecte au loc colaborări. O colaborare reprezintă mulțimea participanților și a relațiilor existente între aceștia, impusă de funcționarea sistemului și de scopurile sale finale. Comportamentul unei clase se definește prin integrarea rolurilor deținute de aceasta în toate colaborările și scenariile în care este implicată.

O primă formă de reprezentare grafică a colaborărilor și interacțiunilor dintre obiecte - diagrama de colaborări - se realizează cu scopul descrierii comportamentelor la care participă obiecte aparținând mai multor clase pentru realizarea unui caz de utilizare. O interacțiune se realizează prin intermediul unui mesaj și presupune existența unui emitenț și a unui destinatar. Interacțiunile dintre obiecte sunt evidențiate prin săgeți care pot reprezenta, după caz, stimuli sau mesaje, folosite de unele obiecte pentru a determina comportamentul altora. Forma săgeților indică tipul comunicației, ca în Figura 4. Se pot specifica în diagramă, de asemenea, sensul și ordinea cronologică a acestor interacțiuni.

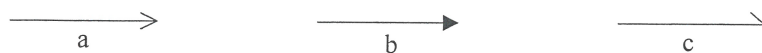


Figura 4 – a – flux simplu; b – comunicație sincronă; c – comunicație asincronă.

O comunicație sincronă constă într-un apel de procedură a cărei execuție trebuie să fie terminată pentru ca aplicația să poată continua, în timp ce, în cazul unei comunicații asincrone, emitențul (inițiatorul mesajului) își continuă secvența de prelucrări, independent de reacția destinatarului.

O diagramă de colaborări se poate realiza fie la nivel de specificație, fie la nivel de instanță.

Pentru a preciza efectul unui mesaj asupra unui obiect ori a unei legături, se folosesc notațiile standard:

- {new} – creare de obiecte ori de legătură în timpul execuției;
- {transient} – creare de obiect ori legătură, urmată de distrugere în timpul execuției;
- {destroyed} – suprimare de obiect ori legătură în timpul execuției.

Dialogul între eventualii actori implicați și sistem are loc prin intermediul unei interfețe (în diagrama realizată *frmLoturi* sau *Ecran CostABC*).

Modelul de implementare a claselor poate fi actualizat prin adăugarea aspectelor relevante de diagramele de colaborări.

Diagrama de colaborări pentru adăugarea unui lot este prezentată în Figura 5. Este asemănătoare, din punct de vedere al interacțiunilor și al comunicațiilor dintre obiectele implicate, cu cele pentru operații de adăugare de noi activități (generatoare de cheltuieli directe ori indirecte) ori de noi inductori sau de noi centre de regurare a inductorilor ș.a.m.d.

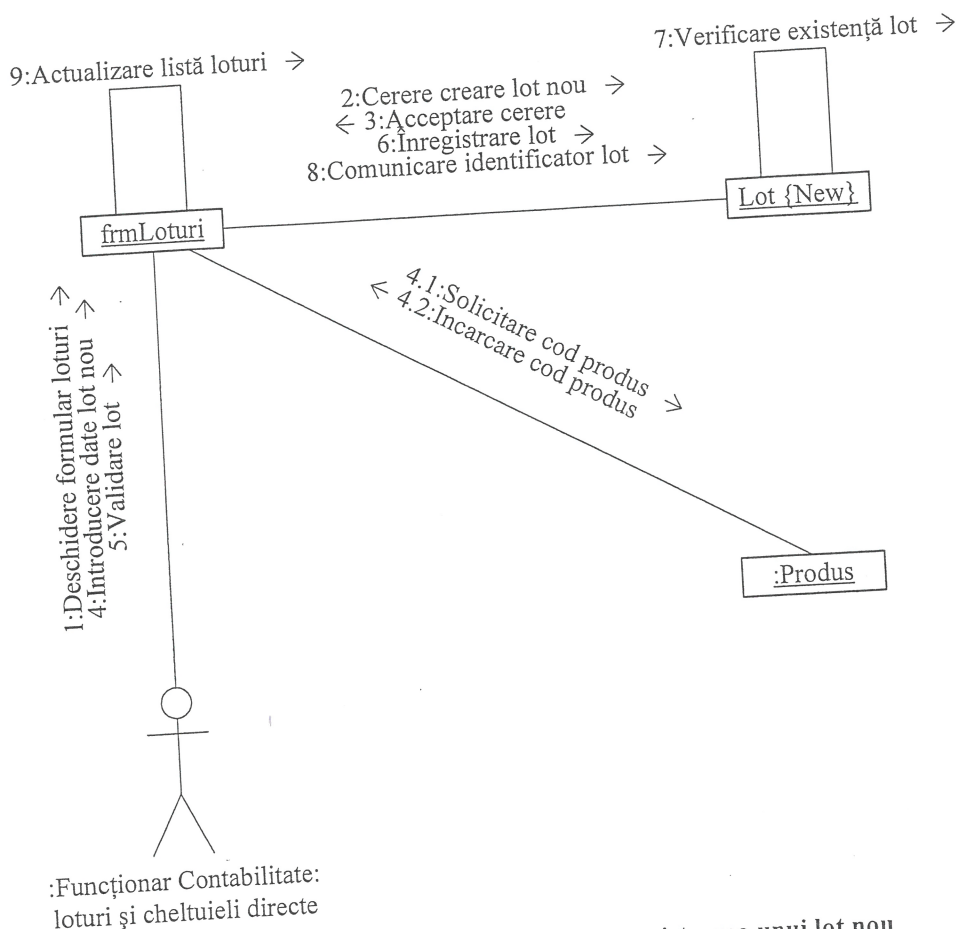


Figura 5 - Diagramă de colaborări pentru înregistrarea unui lot nou

Prin intermediul interfeței asigurate de formularul *Loturi* utilizatorul solicită crearea unui lot (:2). După crearea sa, sunt introduse datele privind noul lot (:4), inclusiv cele referitoare la produsul fabricat în lotul respectiv, verificate și confirmate ulterior (:5). Datele noului lot sunt transmise obiectului *Lot*, creat anterior pentru înregistrare (:6). Identificatorul lotului creat este transmis interfeței (:8), după ce, în prealabil, s-a verificat dacă lotul respectiv nu există deja în administrare. Datele nou înregistrate sunt utilizate pentru a actualiza situația loturilor lansate în fabricație (:9).

4. Diagrama de secvențe

Diagrama de secvențe aprofundează latura cronologică a interacțiunilor dintre obiecte și este foarte utilă, cu precădere, în proiectarea sistemelor cu funcționare în timp real. Fiecare obiect este reprezentat printr-un dreptunghi, de-a lungul unei linii verticale numită „linie de viață”, reflectând durata de existență a obiectului, între momentele creării și distrugerii sale. Săgețile prin care sunt reprezentate mesajele dintre obiecte sunt ordonate cronologic de sus în jos, specificându-se pentru fiecare cel puțin numele. Pot fi reprezentate mesaje transmise de un obiect către el însuși (autoapelare), precum și eventuale reguli care să condiționeze emiterea unui mesaj.

Pe orizontală, se vor reprezenta participanții la aceste interacțiuni. În diagramele de secvențe, distrugerea unui obiect este notată grafic cu X. Comunicația prin mesaje este reprezentată cu săgeți asemănătoare celor folosite în diagramele de colaborări (oblice, în cazul în care durata mesajului este mai mare).

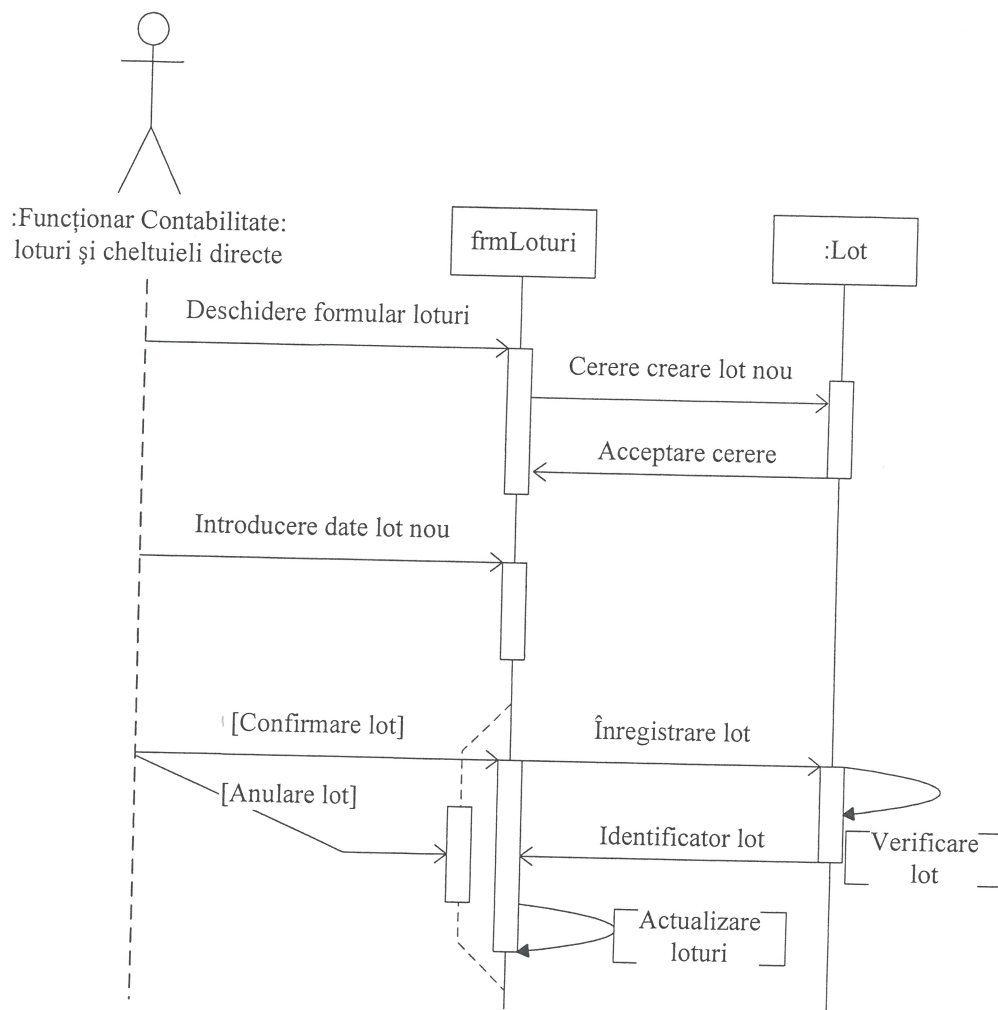


Figura 6 - Diagramă de secvențe pentru înregistrarea unui lot nou

5. Diagrama componentelor

O *componentă* desemnează o unitate fizică de implementare, compusă din cod de program în orice formă (sursă, binar sau executabil) sau un echivalent al acestuia (fișier de comenzi sau script).² Diagrama de componente servește la reprezentarea grafică a repartizării acestora pe nodurile unei rețele de lucru și a interacțiunilor dintre componente. În mediul integrat de dezvoltare Visual Basic, o componentă corespunde unui proiect.

² [17], p. 302;

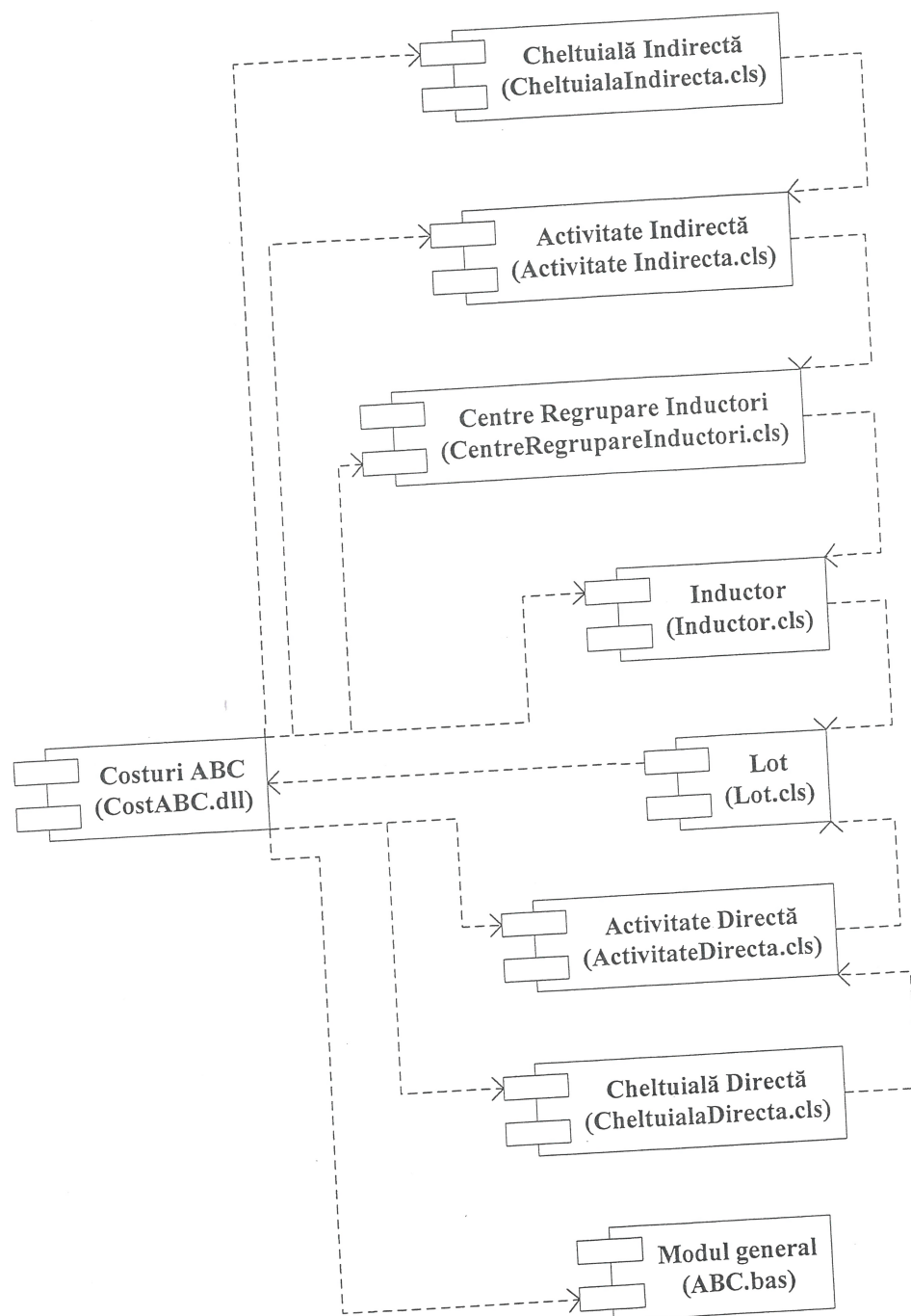


Figura 7 - Diagrama componentelor în format sursă pentru determinarea costului total prin metoda ABC

6. Diagrama de amplasare

Modul de organizare a lucrului în rețeaua pe care se va exploata sistemul este reprezentat într-o diagramă de amplasare (sau de distribuire), ca în Figura 8. Astfel, este specificată repartizarea fizică a bazei de date și a funcționalităților sistemului.

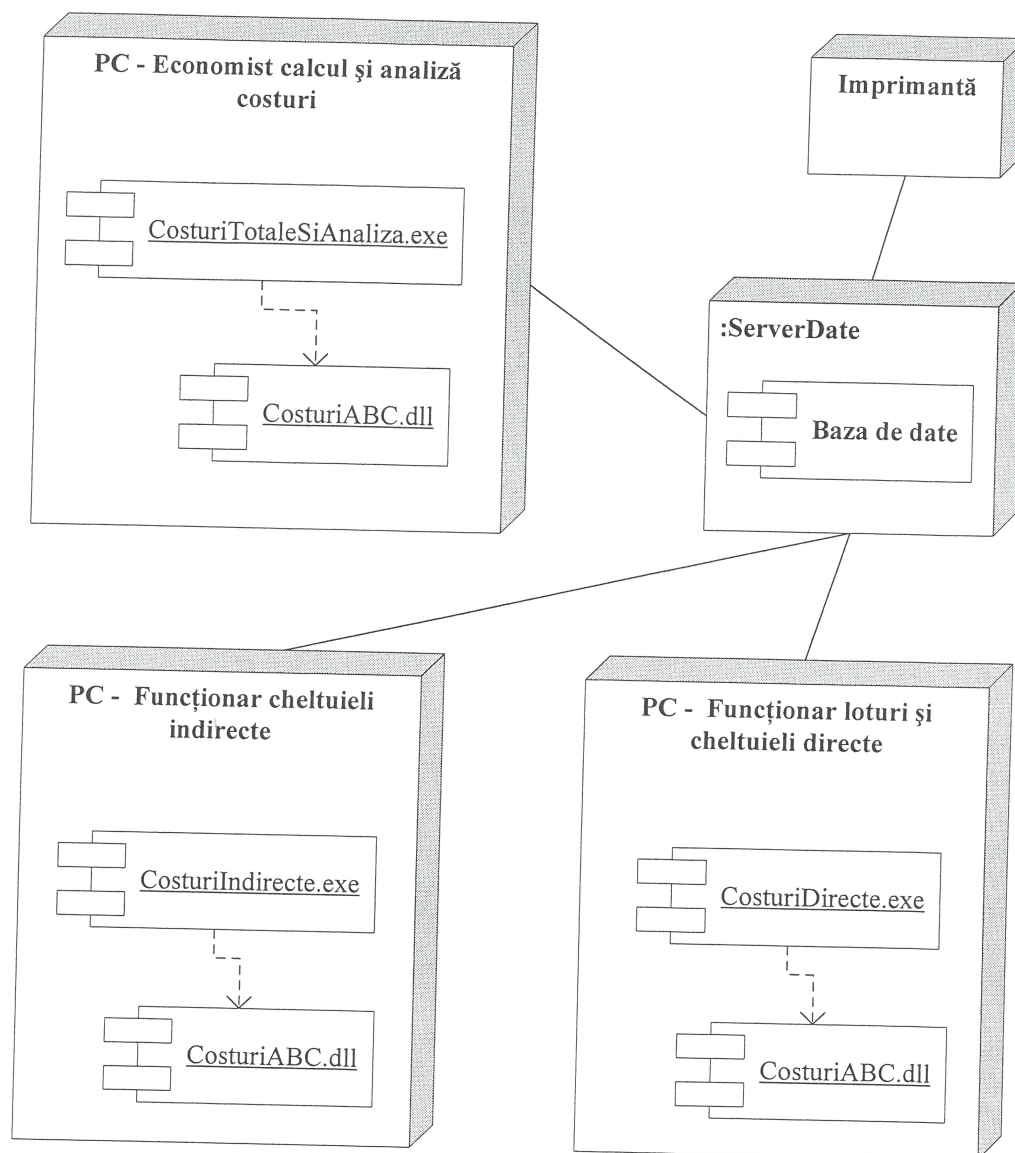


Figura 8 - Diagrama de amplasare pentru determinarea costului total utilizând sistemul ABC

7. Modelul de persistență

Modelul de persistență este un model relațional. Identificatorii obiectelor se transformă în chei primare ale tabelor modelului. Dintre atribute, sunt salvate doar cele a căror persistență trebuie asigurată; nu trebuie incluse în tabele atributele derivate (totaluri, medii etc.). Asocierile și agregările se concretizează în generarea de chei externe.

Modelul relațional (logic), obținut într-un demers sistemic de proiectare, poate constitui modelul de persistență al datelor, realizat în cadrul abordării obiectuale utilizând UML.

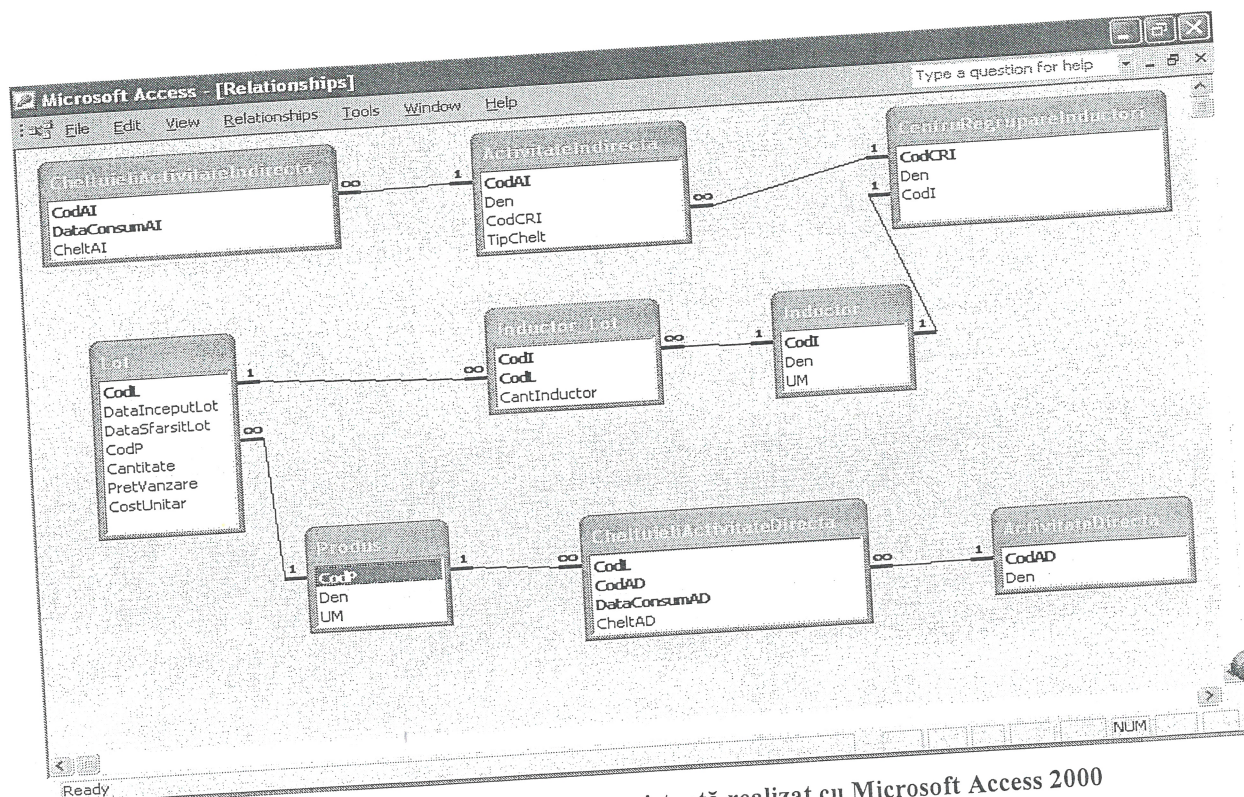


Figura 9 - Structura modelului de persistență realizat cu Microsoft Access 2000

Bibliografie

1. **ANICA-POPA, I., L. ANICA-POPA**: Programarea calculatoarelor în limbajul Visual Basic. Teorie și aplicații, Editura Infomega, București, 2002.
2. **AREITIO, M. Del C.**: Análisis de sistemas: fundamento y desarrollo de aplicaciones informáticas, Editura Areber, Bilbao, 1992.
3. **AREITIO, M. Del C.**: Ingeniería del Software. Guía de estudio, Editura G.C., Bilbao, 1995.
4. **CAGWIN, D., M. J. BOUWMAN**: The Association Between Activity-Based Costing And Improvement. În: Financial Performance, aprilie 2000.
5. **De AMESCUA SECO, A.**: Ingeniería del software de gestión: análisis y diseño de aplicaciones, Editura Paraninfo, Madrid, 1995.
6. **DRUCKER, P.F.**: The Information Executives Truly Need. În: Harvard Business Review, ianuarie/februarie, 1995.
7. **EMBLEMSVAG, J.**: Historic Overview of Modern Management Practices. În: SURF Study Trip, 2000.
8. **GARDARIN, G.**: Bases de données: object et relationnel. Maison d'Édition Eyrolles, Paris, 2001.
9. **GRANOF, M., D. PLATT, I. Vaysman**: Using Activity-Based Costing to manage more effectively, The Pricewaterhouse Coopers Endowment for The Business of Government, ianuarie, 2000.
10. **Grupul BDASEIG, V. FLORESCU (coordonator)**: Baze de date. Fundamente teoretice și practice, Editura Infomega, București, 2002.
11. **LEÓN SERRANU, G.**: Ingeniería de sistemas de software, Editura Isdefe, Madrid, 1996.
12. **OPREA, D.**: Analiza și proiectarea sistemelor informaționale economice, Editura Polirom, Iași, 1999.
13. **POPOVICI, D.M., I. M. POPOVICI, J. G. RICAN**: Proiectare și implementare software, Editura Teora, București, 1998.
14. **RAVIGNON, L., P.-L. BESCOS**: Gestion par activités. La méthode ABC/ABM. Piloter efficacement une PME, Edition d'Organisation, Paris, 1998.
15. **SARU, D., A. D. IONIȚĂ**: Dezvoltarea orientată pe obiecte a programelor medii și mari în limbajul C++", Editura Matrix Rom, București, 1998.
16. **WHITTEN, J., L. BENTLEY, V. BARLOW**: Análisis de sistemas y métodos de diseño: edición para el instructor, traducción, Diorki, Editura Irwin, Barcelona, 1996.
17. **ZAHARIE, D., I. ROȘCA**: Proiectarea obiectuală a sistemelor informatice, Editura Dual Tech, București, 2002.