Laborator 2

Diagrama de clase (*Class Diagram*)

Diagrama de clase este folosită pentru a modela structura (viziunea statică asupra) unui sistem. O astfel de diagramă conține clase / interfețe, obiecte şi relații care se stabilesc între acestea. Relațiile pot fi de tipul:

- asociere;
- agregare;
- generalizare;
- dependență;
- realizare.

Clasele sunt folosite pentru a surprinde vocabularul sistemului ce trebuie dezvoltat. Ele pot include:

- abstracții care fac parte din domeniul problemei;
- clase necesare la momentul implementării.

O clasă poate reprezenta entități software, entități hardware sau concepte. Modelarea unui sistem presupune identificarea elementelor importante din punctul de vedere al celui care modelează. Aceste elemente formează vocabularul sistemului. Fiecare dintre aceste elemente are o mulțime de proprietăți.

Clase

Elementele unei clase sunt:

Nume - prin care se distinge de alte clase - o clasă poate fi desenată arătându-i numai numele;

Atribute - reprezintă numele unor proprietăți ale clasei;

Operații (metode) - reprezintă implementarea unor servicii care pot fi cerute oricărui obiect al clasei.

Notația grafică pentru clasă poate fi observată în figura 1.

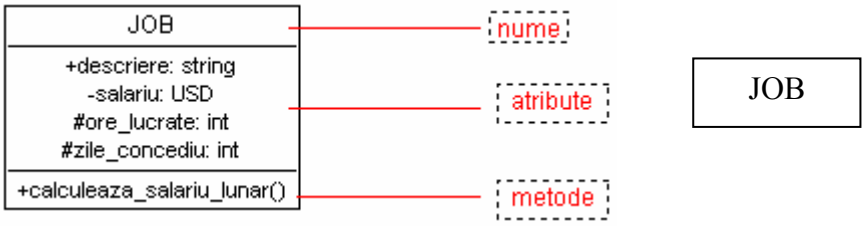

Figura 1. Notația grafică pentru clasă

Specificatorii de vizibilitate au următoarea semnificație:

- \+ public (dreptul de utilizare se acordă şi celorlalte clase);
- \- private (numai clasa dată poate folosi atributul sau operația);
- \# protected (posibilitatea de utilizare este accesibilă numai claselor succesoare)

O clasă poate avea oricâte atribute şi operații sau poate să nu aibă nici un atribut sau nici o operație. Modelarea vocabularului unui sistem presupune identificarea elementelor pe care utilizatorul sau programatorul le foloseşte pentru a descrie soluția problemei. Pentru fiecare element se identifică o mulțime de responsabilități (ce trebuie să facă acesta), după care se definesc atributele şi operațiile necesare îndeplinirii acestor responsabilități.

Obiecte

Obiectele sunt instanțe ale claselor. Obiectele au identitate şi valori ale atributelor. Notația grafică pentru obiect se poate observa în figura 2.

Figura 2. Notații grafice pentru obiecte

Interfețe

Interfața specifică un grup de operații caracteristice unei comportări determinate a unei clase. Se modelează cu acelaşi simbol ca şi clasele. Interfața are numai operații. Pentru a le putea deosebi de clase se plasează stereotipul <<interface>> sau caracterul "I" la începutul numelui interfeței respective.

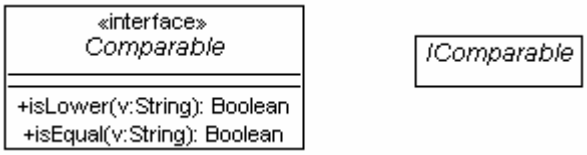

Figura 3. Notații grafice pentru interfețe

Clasă parametrizată (template)

Clasa parametrizată are unul sau mai mulți parametri formali. Prin intermediul acestei clase se poate defini o familie de clase dând valori parametrilor formali. De obicei parametrii reprezintă tipuri ale atributelor. Notația grafică se poate vedea în figura 4.

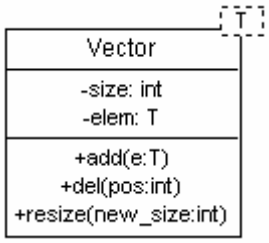

Figura 4. Exemplu de clasă parametrizată

Relații

O relație modelează o conexiune semantică sau o interacțiune între elementele pe care le conectează. În modelarea orientată obiect cele mai importante relații sunt relațiile de asociere, dependența, generalizare şi realizare. Un caz particular al relației de asociere îl constituie relația de agregare. Între clase se pot stabili relații de generalizare, dependență şi realizare. Relațiile de asociere şi agregare se stabilesc între instanțe ale claselor.

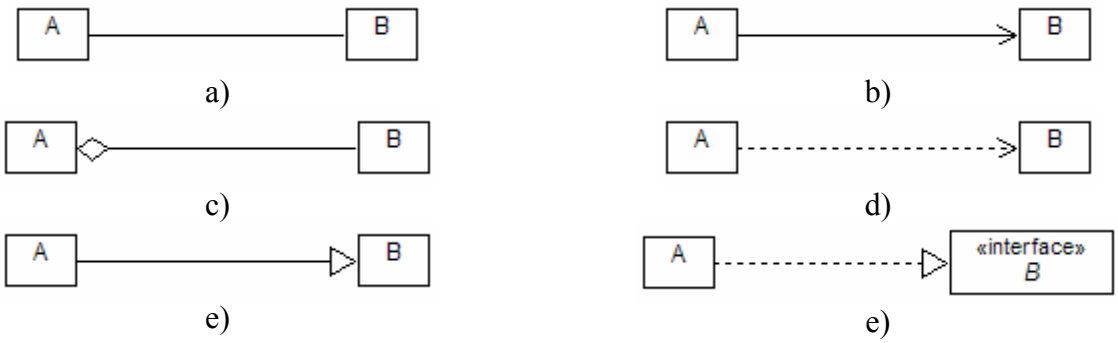

Figura 5. Notații grafice pentru diferite tipuri de relații: a)asociere bidirecțională; b)asociere unidirecțională; c) agregare; d) dependență; e) generalizare, f) realizare

Relația de asociere

Relația de asociere exprimă o conexiune semantică sau o interacțiune între obiecte aparținând unor anumite clase. Asocierea poartă informații despre legăturile dintre obiectele unui sistem. Pe măsură ce sistemul evoluează, pot fi create legături noi între obiecte sau pot fi distruse legăturile existente. Relația de asociere oferă baza arhitecturală pentru modelarea conlucrării şi interacțiunii dintre clase.

O asociere interacționează cu obiectele sale prin intermediul *capetelor de asociere*. Capetele de asociere pot avea *nume*, cunoscute sub denumirea de roluri, şi *vizibilitate*, ce specifică modul în care se poate naviga înspre respectivul capăt de asociere. Cea mai importantă caracteristică a lor este multiplicitatea, ce specifică câte instanțe ale unei clase corespund unei singure instanțe a altei clase. De obicei multiplicitatea este folosită pentru asociațiile binare.

Reprezentarea grafică a asocierii este o linie (sau drum) ce conectează clasele ce participă în asociere. Numele asocierii este plasat pe linie, iar multiplicitatea şi rolurile sunt plasate la extremitățile sale (figura 6).

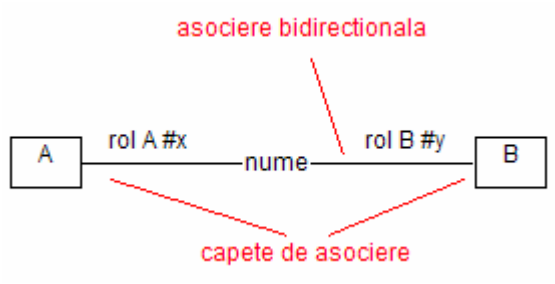

Figura 6: Notația grafică detaliată a relației de asociere

Observație. Numele rolurilor pot fi omise (eventual şi numele asocierii)

Este posibilă specificarea direcției unei asocieri, pentru a elimina legături redundante sau irelevante pentru un anumit model. În această situație notația grafică pentru relația de asociere este o linie cu o săgeată la unul din capete indicând direcția asocierii (figura 5b).

Relația de agregare

Relația de agregare este o asociere ce modelează o relație *parte-întreg*. Este reprezentată ca un romb gol ataşat la capătul asocierii de lângă clasa agregat (figura 5c). În figură o instanță a clasei A conține o instanță a clasei B (altfel spus un obiect B este o parte unui obiect A). Relația de agregare este deci un caz particular al relației de asociere. Ea poate avea toate elementele unei relații de asociere, însă în general se specifică numai multiplicitatea. Se foloseşte pentru a modela situațiile în care un obiect este format din mai multe componente.

Compoziția este o formă mai puternică a agregării. *Partea* are „timpul de viață" al *întregului*. *Întregul* poate avea responsabilitatea directă pentru crearea sau distrugerea *părții* sau poate accepta o altă *parte* creată şi mai apoi să paseze „responsabilitatea" altui *întreg*.

Obs. Instanțele nu pot avea relații de agregare ciclice (o *parte* nu poate conține *întregul*)

În figura 7 este prezentat un exemplu în care se foloseşte agregarea şi compoziția.

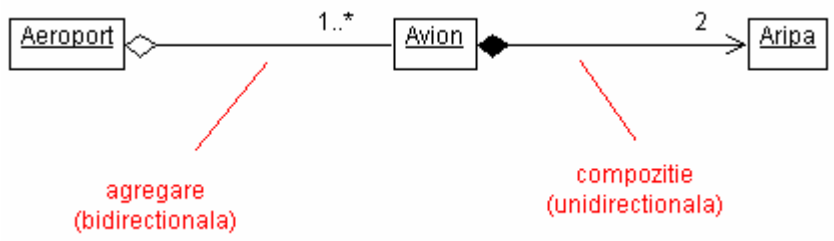

Figura 7: Exemplu de relații de agregare şi compoziție

Relația de dependență

O dependență (figura 5d) indică o relație semantică între două elemente ale modelului. Se foloseşte în situația în care o schimbare în elementul destinație (B) poate atrage după sine o schimbare în elementul sursă (A). Această relație modelează interdependențele ce apar la implementare.

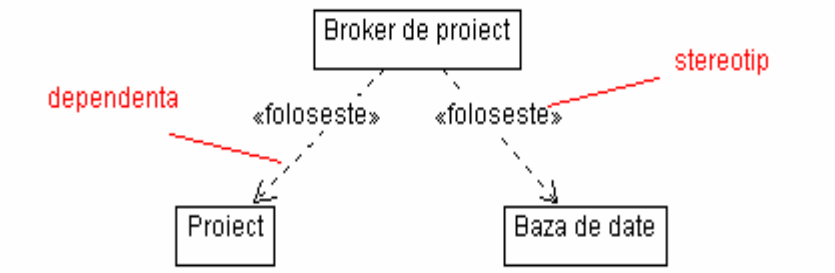

Figura 8. Exemplu de relații de dependență

Relația de generalizare

Relația de generalizare (figura 5e) se foloseşte pentru a modela conceptul de moştenire între clase. în figură clasa A este derivată din clasa B. Spunem că A este clasa derivată (sau subclasa, sau clasa copil), iar B este clasa de baza (sau superclasa, sau clasa părinte).

Relația de generalizare mai poartă denumirea de relație de tip *is a* (*este un fel de*), în sensul că o instanță a clasei derivate A este în acelaşi timp o instanță a clasei de bază B (clasa A este un caz particular al clasei B sau, altfel spus, clasa B este o generalizare a clasei A). Clasa A moşteneşte toate atributele şi metodele clasei B. Ea poate adăuga noi atribute sau metode sau le poate redefini pe cele existente.

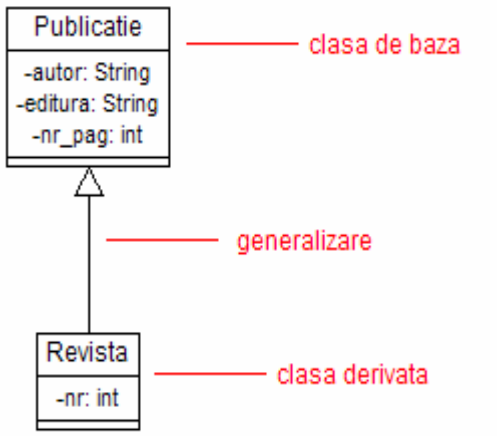

Figura 9: Exemplu de relație de generalizare

Relația de realizare

Relația de realizare (figura 5f) se foloseşte în cazul în care o clasă (A) implementează o interfață (B). Se spune că A realizează interfața specificată de B. O interfață este o specificare comportamentală fără o implementare asociată. O clasă include o implementare. Una sau mai multe clase pot realiza o interfață prin implementarea metodelor definite de acea interfață.

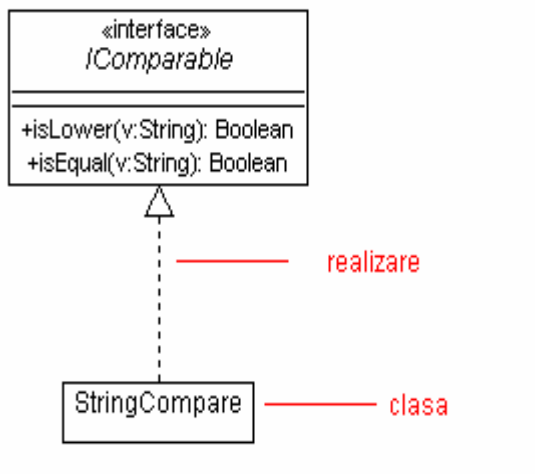

Figura 10. Exemplu de relație de realizare.

În figura 11 sunt prezentate două diagrame de clase în care se folosesc toate tipurile de relații prezentate mai sus.

Probleme propuse

Pentru fiecare din problemele de mai jos să se realizeze diagramele de clase:

1. Automat cafea (alegere tip cafea, introducere moneda, eliberare rest, preluare produs, etc)
2. ATM (verificare PIN, vizualizare suma din contul personal, extragere, tipărire chitanța etc.)
3. Ceas electronic (afişare ora curentă / data curentă, modificare oră / dată, cronometru etc.)

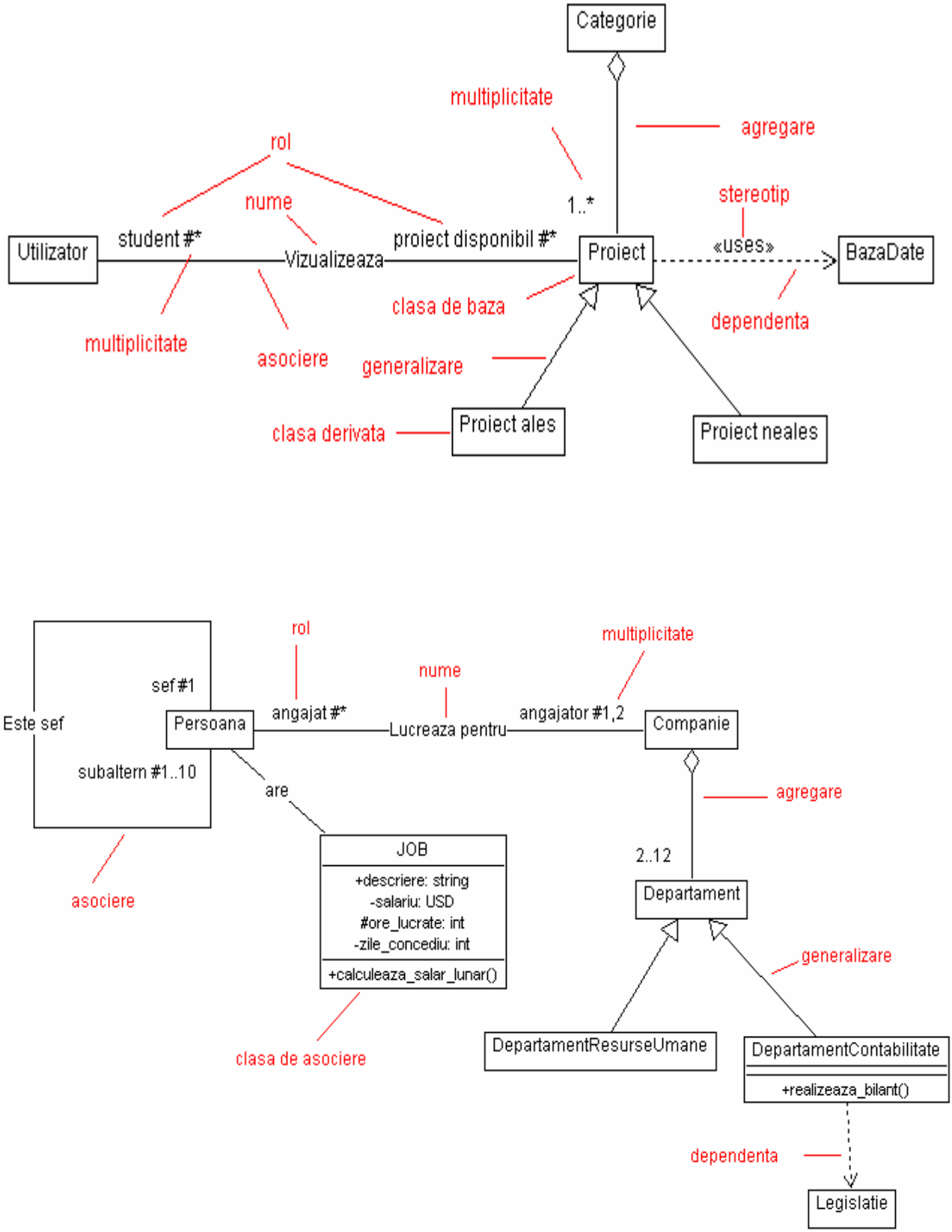

Figura 11. Exemple de diagrame de clase