UML
Diagrame de clase -2

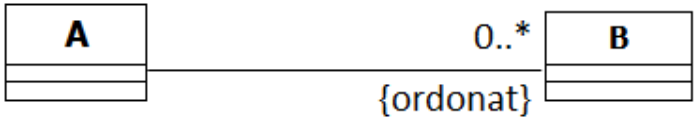
Prof. univ. dr. ing. Florica Moldoveanu

Curs *Ingineria programelor* – UPB, Automatică şi Calculatoare
2020-2021

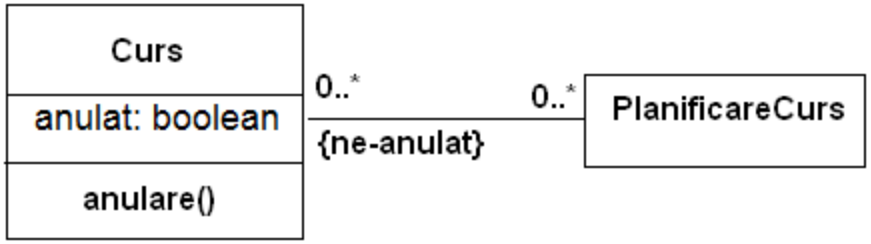
Relatia de asociere - continuare

Asociere constrânsă

O constrângere este o conditie care trebuie sa fie îndeplinita pe toata durata de existență a obiectelor claselor asociate.

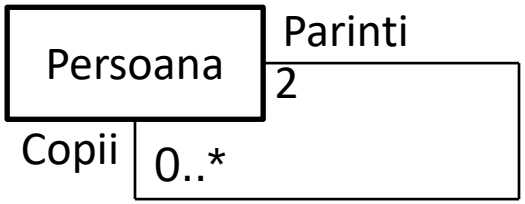


- Obiectele clasei B legate la un obiect din clasa A trebuie sa fie ordonate in orice moment.



- Se poate crea o legatura intre un obiect PlanificareCurs si un obiect Curs, numai daca valoarea atributului **anulat** este *ne-anulat*.

Asociere reflexivă: abstractizarea unor legaturi intre obiecte din aceeasi clasa

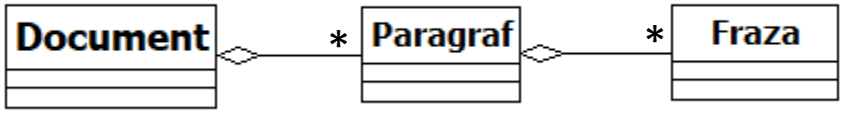


- Numirea rolurilor este în acest caz esentiala pentru claritatea diagramei.

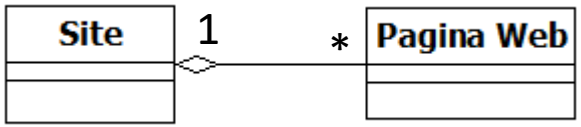
Asocierea prin agregare (1)

• Agregarea este o formă particulară de asociere, care exprimă un **cuplaj temporar de tipul "compus-componente", parte-întreg, parte-din.**

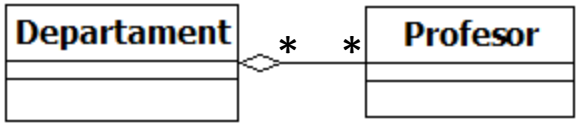
• Obiectele au propria lor existenta dar între agregat si componente exista o relatie de apartenenta la un moment dat.



O fraza F poate fi scoasa dintr-un paragraf si folosita in alt paragraf

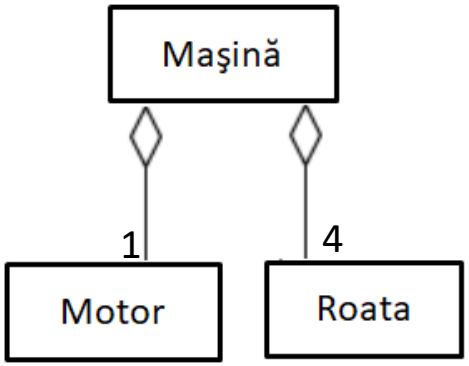


O pagina web, P, poate fi scoasa de pe un site, S, pentru a fi folosita în alt site. Site-ul S nu dispare dupa scoaterea paginii P. Distrugerea site-ului S nu afecteaza paginile pe care le avea.



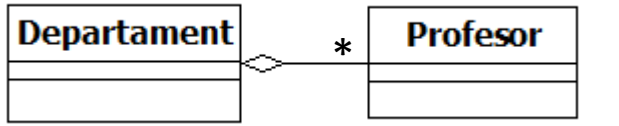
Un profesor, X, face parte dintr-un departament D, la un moment dat. La un alt moment de timp, X poate face parte din alt departament, din mai multe departamente sau din nici unul. Departamentul continua sa existe dupa plecarea unui profesor. Daca un departament dispare, profesorii care fac parte din departament se pot angaja in alte departamente.

Asocierea prin agregare (2)

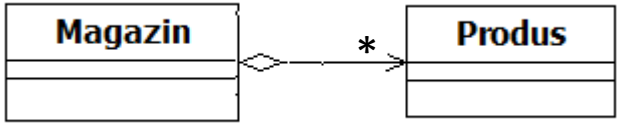


- O maşină are un motor si 4 roti.
- Atat motorul cât şi rotile pot fi inlocuite
- Distrugerea masinii nu are ca efect distrugerea motorului si a rotilor pe care le-a avut.

Asocierea prin agregare poate fi bidirectionala sau unidirectionala:



- Departamentul cunoaste profesorii sai.
- Un profesor stie din ce department face parte.



- Un magazin cunoaste produsele pe care le contine.
- Produsele nu cunosc magazinul în care se afla.

Agregarea mai este numita "Shared aggregation". Se caracterizeaza prin urmatoarele reguli:

- Agregatul si componentele au propria lor existenta, sunt create si distruse independent.
- Componentele unui agregat nu dispar atunci cand este distrus agregatul.
- O componenta a unui agregat poate face parte din mai multe agregate.

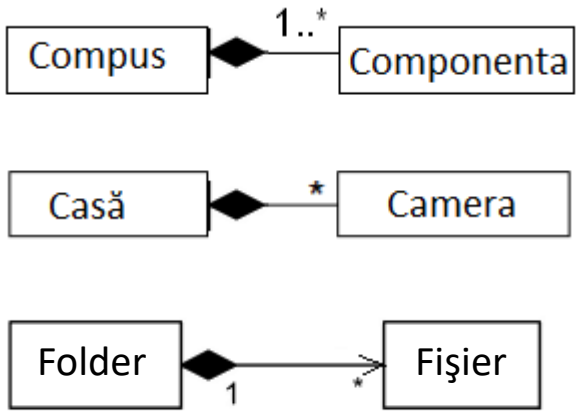
Este o relatie de tip compus-componente slabă.

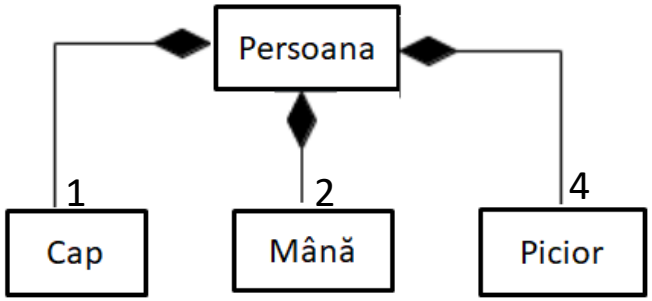
Asocierea prin compunere

Compunerea – numită şi “Composite aggregation”

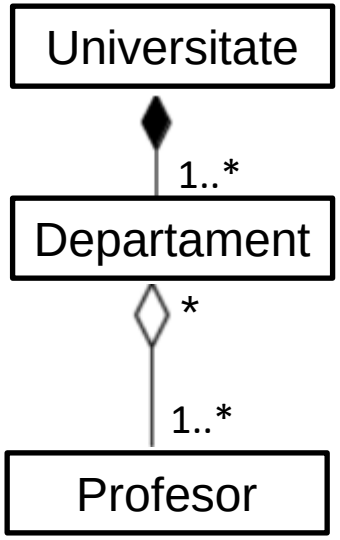
Este o agregare prin conţinere fizică, ce se caracterizeaza prin urmatoarele reguli:

- O componenta poate fi inclusa intr-un singur agregat la un moment dat
- Componentele nu pot exista de sine stătător, ele fac întotdeauna parte dintr-un agregat.
- Agregatul este responsabil de crearea și distrugerea componentelor sale, fie în mod direct, fie indirect (prin intermediul altor clase)
- Distrugerea agregatului are ca efect distrugerea tuturor componentelor sale.

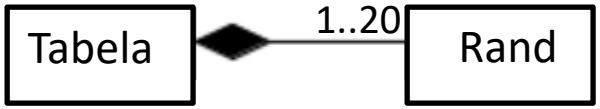




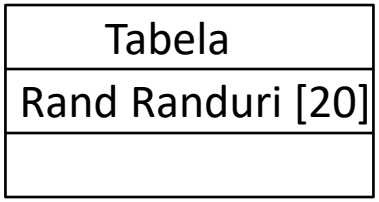
Diferente Compunere - Agregare



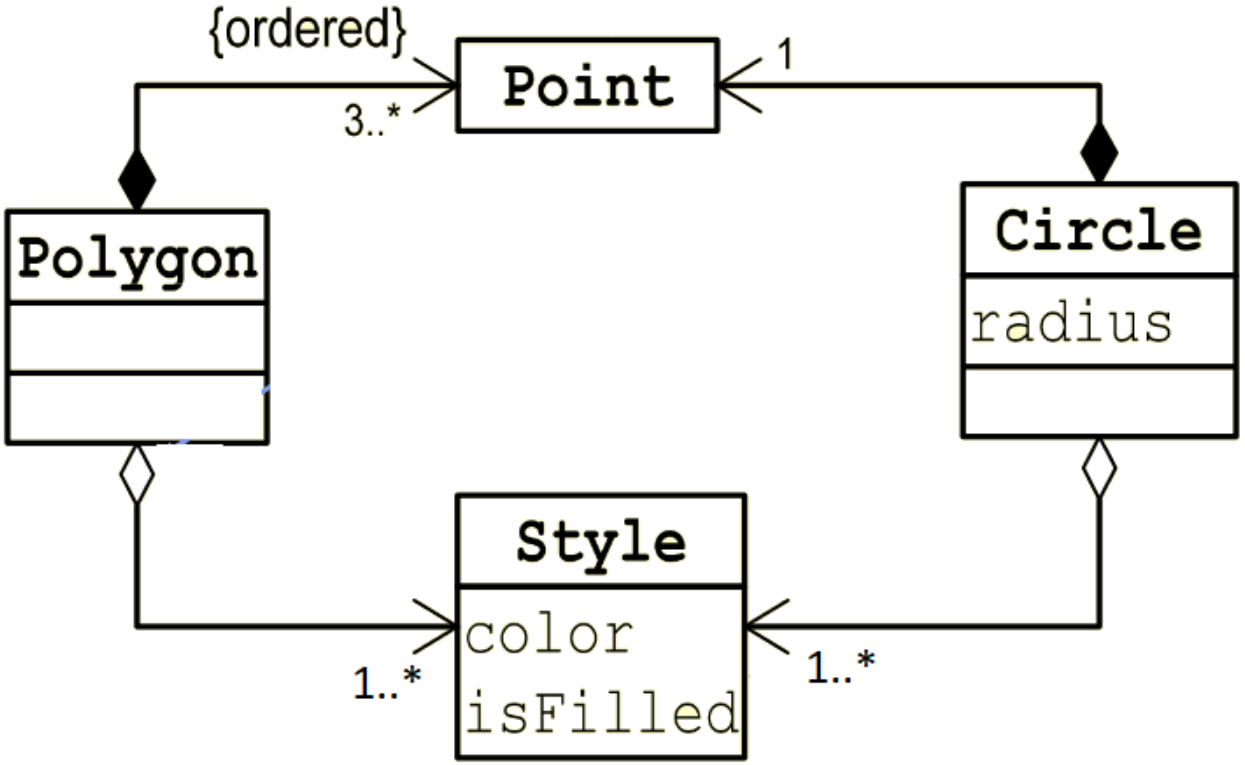
- O universitate este alcatuita din departamente (Calculatoare, Matematica, etc.)
- Fiecare department are mai multi pofesori.
- Daca universitatea se închide, departamentele dispar, dar profesorii din departamente continua sa existe.
- Un professor poate lucra in mai multe departamente
- Un department poate face parte dintr-o singura universitate



Echivalente



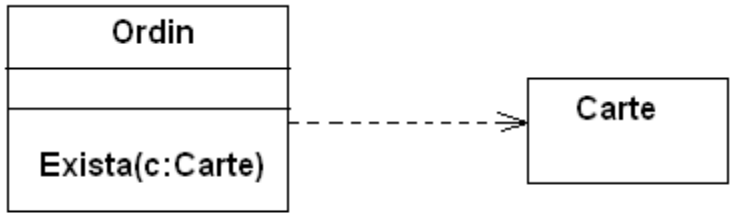
Diferente Compunere - Agregare



Relatia de dependenţă (1)

- Este cea mai slaba relatie între doua clase.
- Exprima de obicei o relatie temporara între obiectele a doua clase:

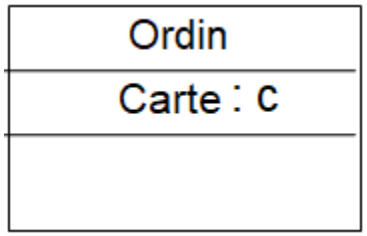
  obiectele clasei dependente interactioneaza pentru scurt timp cu obiectele clasei tinta.
- De ex., o clasa A depinde de o clasa B, daca o metoda a clasei A are un parametru de tip B.

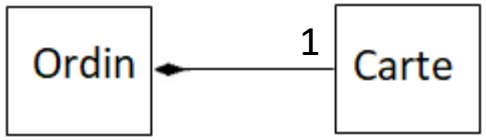


Client: elementul dependent Furnizor: elementul independent

Daca definitia clasei Carte se modifica, este posibil sa fie necesara modificarea functiei Exista().

- Se deosebeste de cazul asocierii dintre A si B, cand un atribut al clasei A este o instanta a clasei B, caz in care intre A si B exista o relatie de agregare prin compunere.

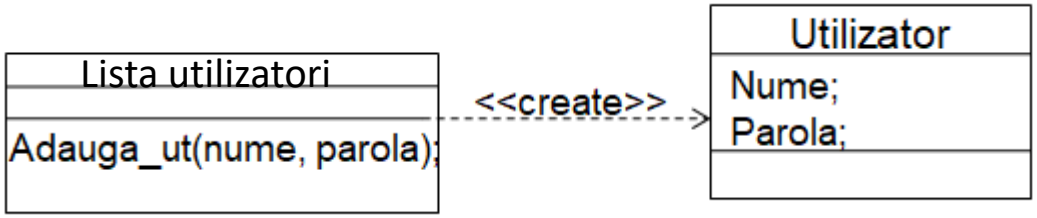




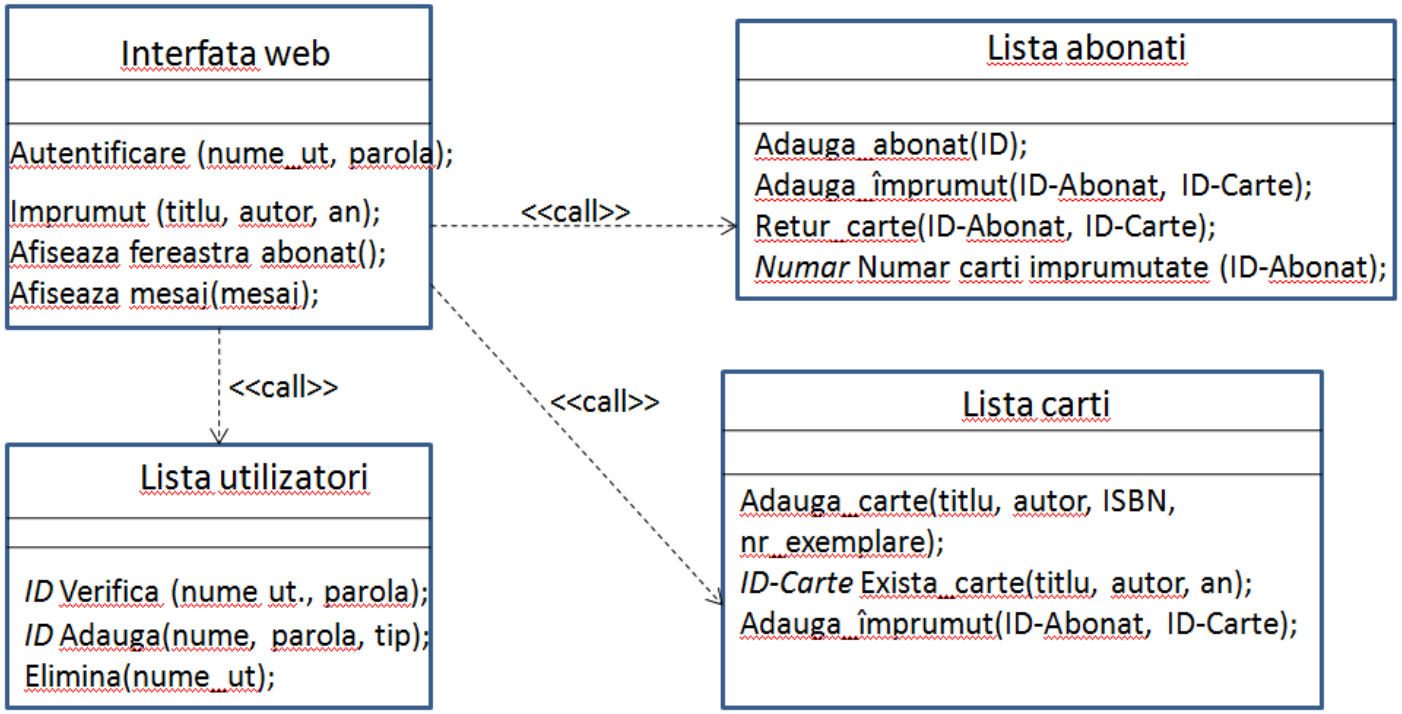
Un obiect Ordin contine un obiect Carte

Relatia de dependenţă (2)

Relația de dependență poate fi adnotată cu un stereotip care oferă informații despre natura relatiei. Exemple:



Obiectele clasei **Utilizator** sunt create de un obiect al clasei **Lista utilizatori** la apelul metodei Adauga_ut.



Interfata web apeleaza operatii din **Lista utilizatori**, **Lista abonati** si **Lista carti.**

De la diagrama de secventa la diagrama de clase

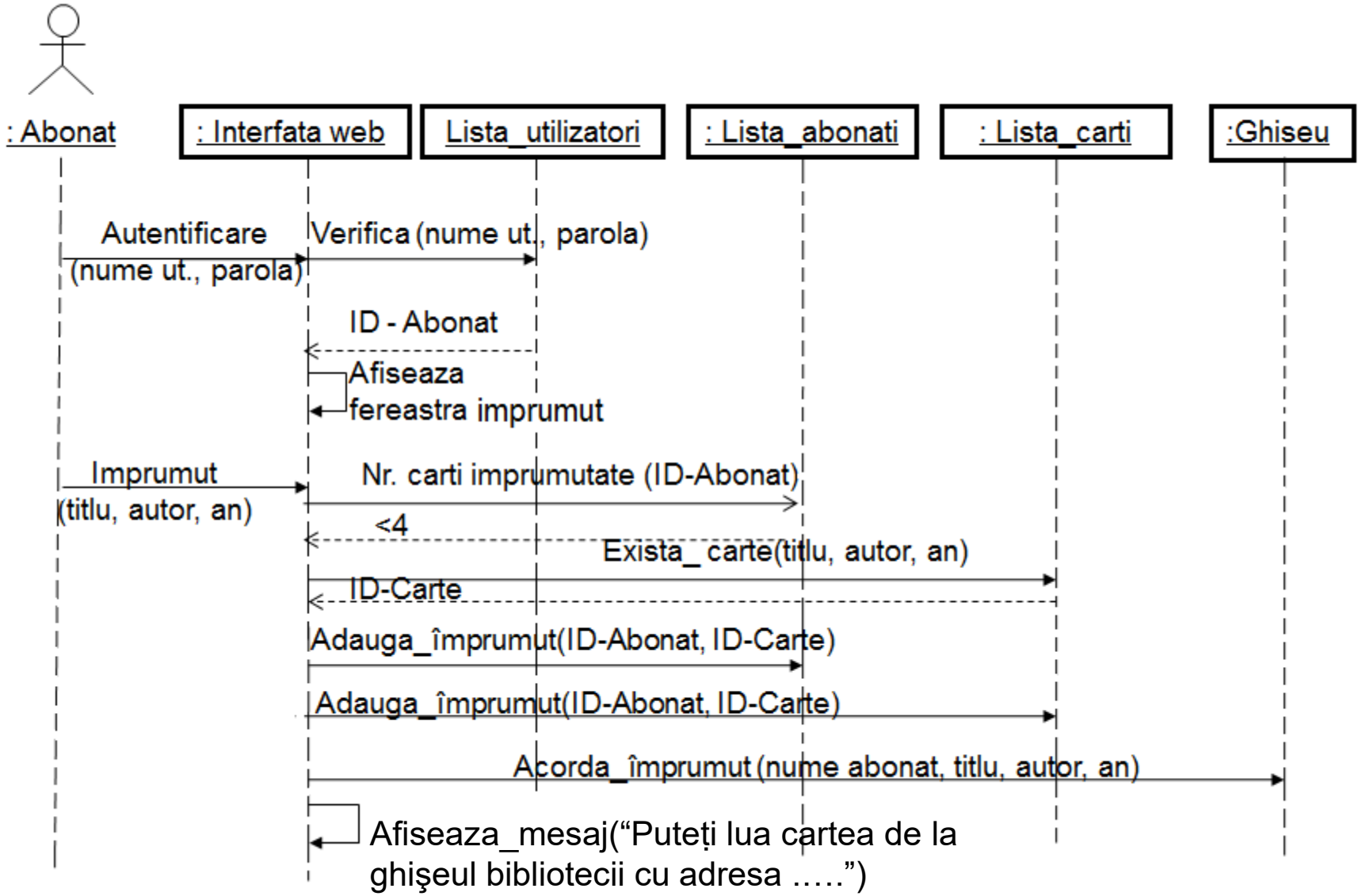
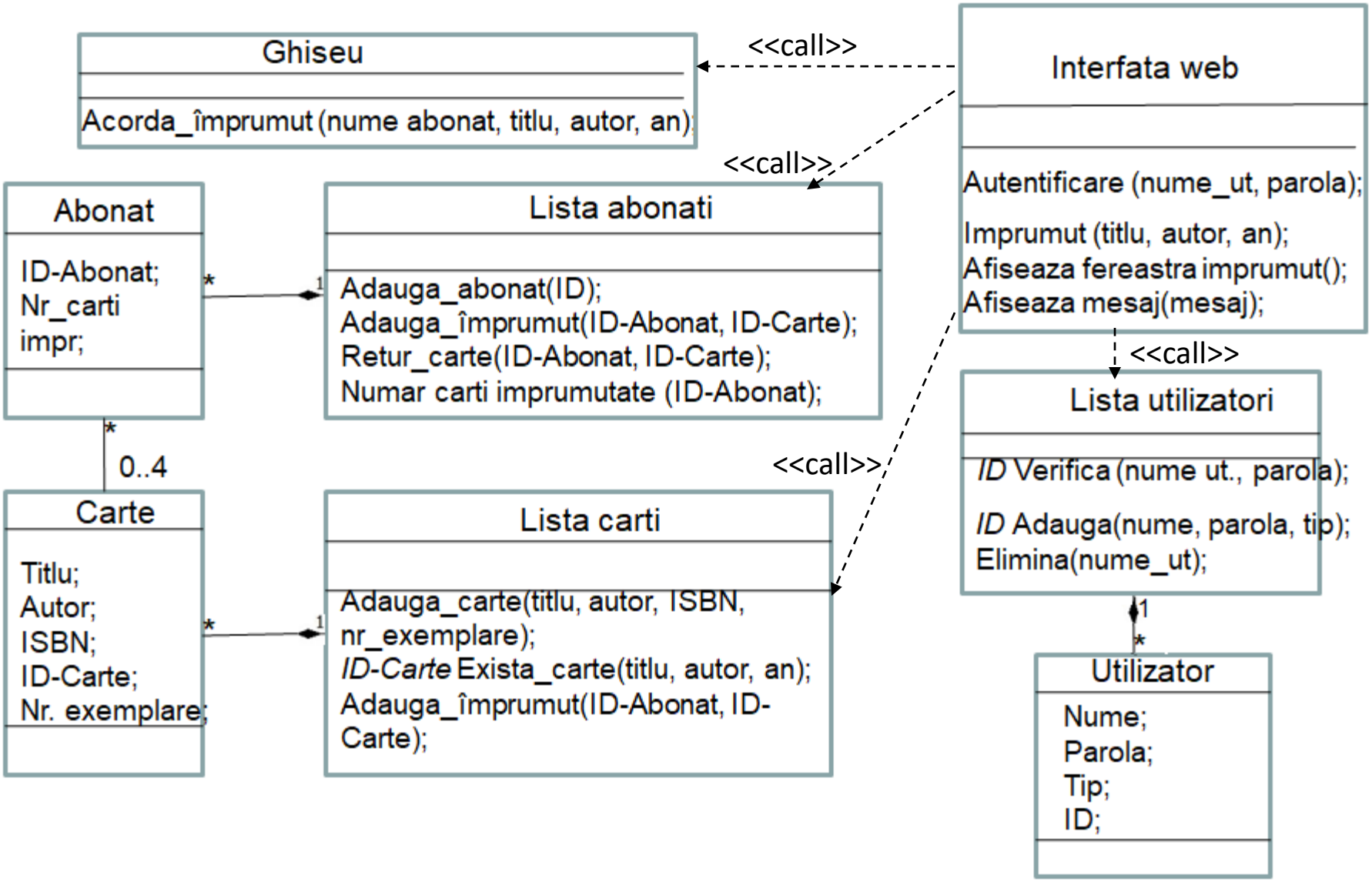


Diagrama de clase conceptuale a SGCB



Relatia de generalizare (1)

- Se bazeaza pe notiunile de: clasificare, generalizare, specializare si extindere.

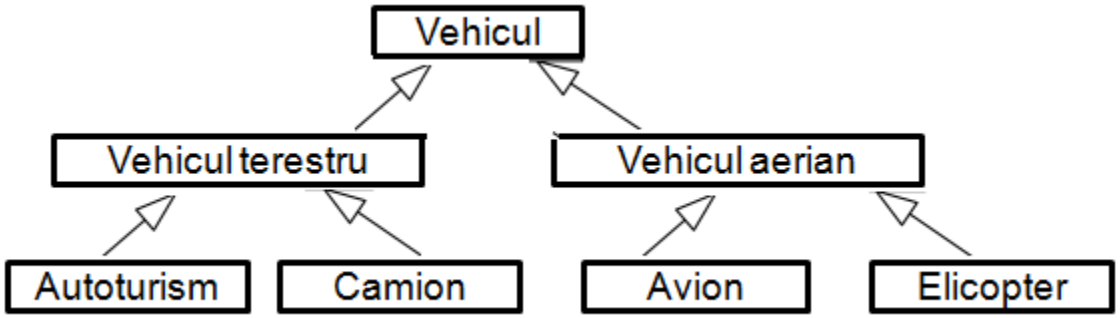
Generalizarea: factorizarea elementelor comune (atribute, operatii si constrangeri) ale unui ansamblu de clase într-o clasa mai generala, numită superclasă.

➢Relatia de generalizare din UML este mai abstracta decat relatia de mostenire din limbajele POO. Ea este mai adecvata etapei de analiza (există şi intre cazuri de utilizare!).

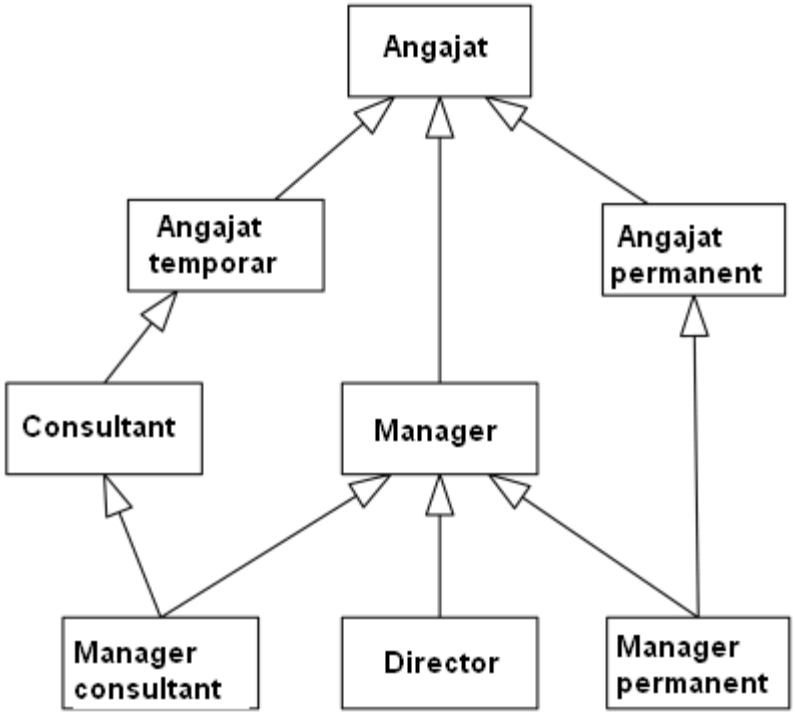
➢Prin clasificare si generalizare, universul problemei este divizat în parti independente care grupeaza obiecte prin afinitate. Modificarea unei parti antreneaza un minimum de modificari ale celorlalte → **arborele de mostenire**: **modificarea unui subarbore nu afecteaza ceilalti subarbori care au aceeasi radacina cu subarborele modificat.**

Specializare si extindere: capturarea particularitatilor unui ansamblu de obiecte, nereprezentate complet prin clasele existente intr-o structura ierarhica.

Relatia de generalizare (2)



Arborele de mostenire: fiecare clasa poate avea o singura super-clasa.

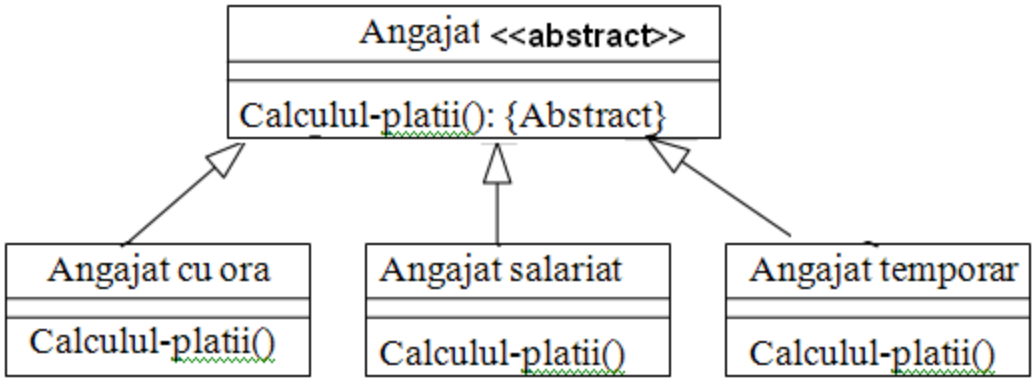


Graful de mostenire: clasele pot avea mai multe super-clase

Relatia de generalizare (3)

Clasa abstracta

- Clasa care nu este instantiabila, dar ale carei clase descendente pot produce instante.
- Clasele abstracte înglobeaza mecanisme generale, ce pot fi particularizate prin specializare si extindere, în clasele descendente.
- O clasa este desemnata ca abstracta prin **stereotipul <<abstract>>** sau folosind caractere italice pentru numele său.
- O operatie abstracta este desemnata prin **constrangerea {Abstract}** sau folosind caractere italice pentru numele său.

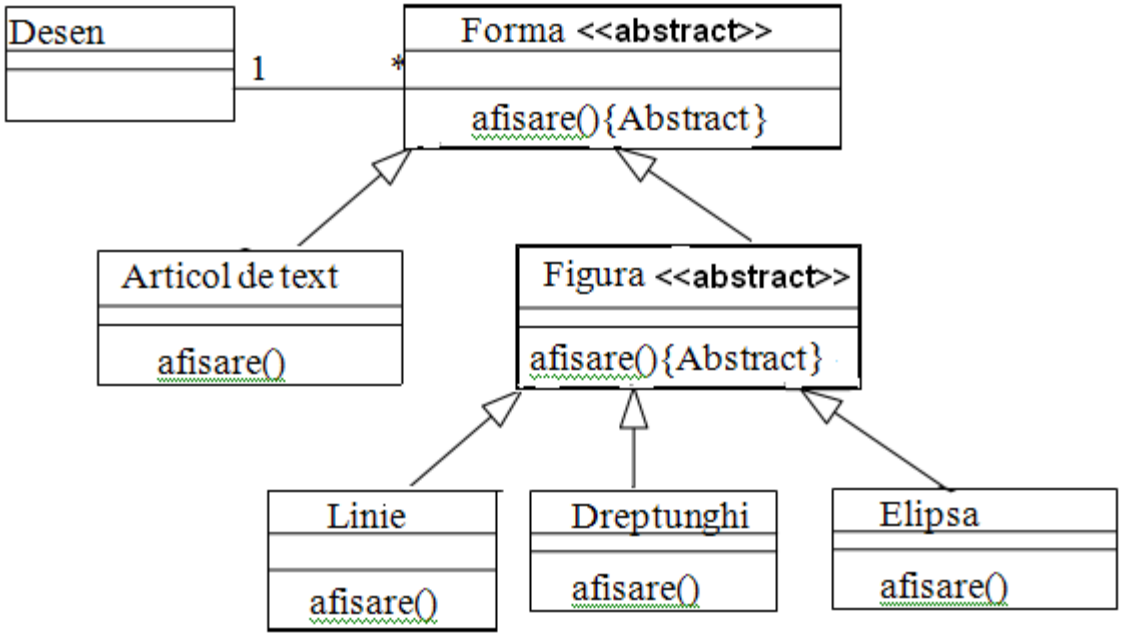


sau

Relatia de generalizare (4)

Polimorfismul

- Desemneaza proprietatea unui element de a lua mai multe forme.
- In Informatica, termenul desemneaza un concept al teoriei tipurilor, conform caruia un nume de obiect poate desemna instante ale unor clase diferite dar provenite din aceeasi arborescenta.
- Interactiunile dintre obiecte pot fi descrise folosind protocolul de comunicatie definit în super-clasele lor. Polimorfismul operatiilor desemneaza posibilitatea ca o operatie a unei superclase să declanseze operatii diferite ca raspuns la acelasi mesaj.

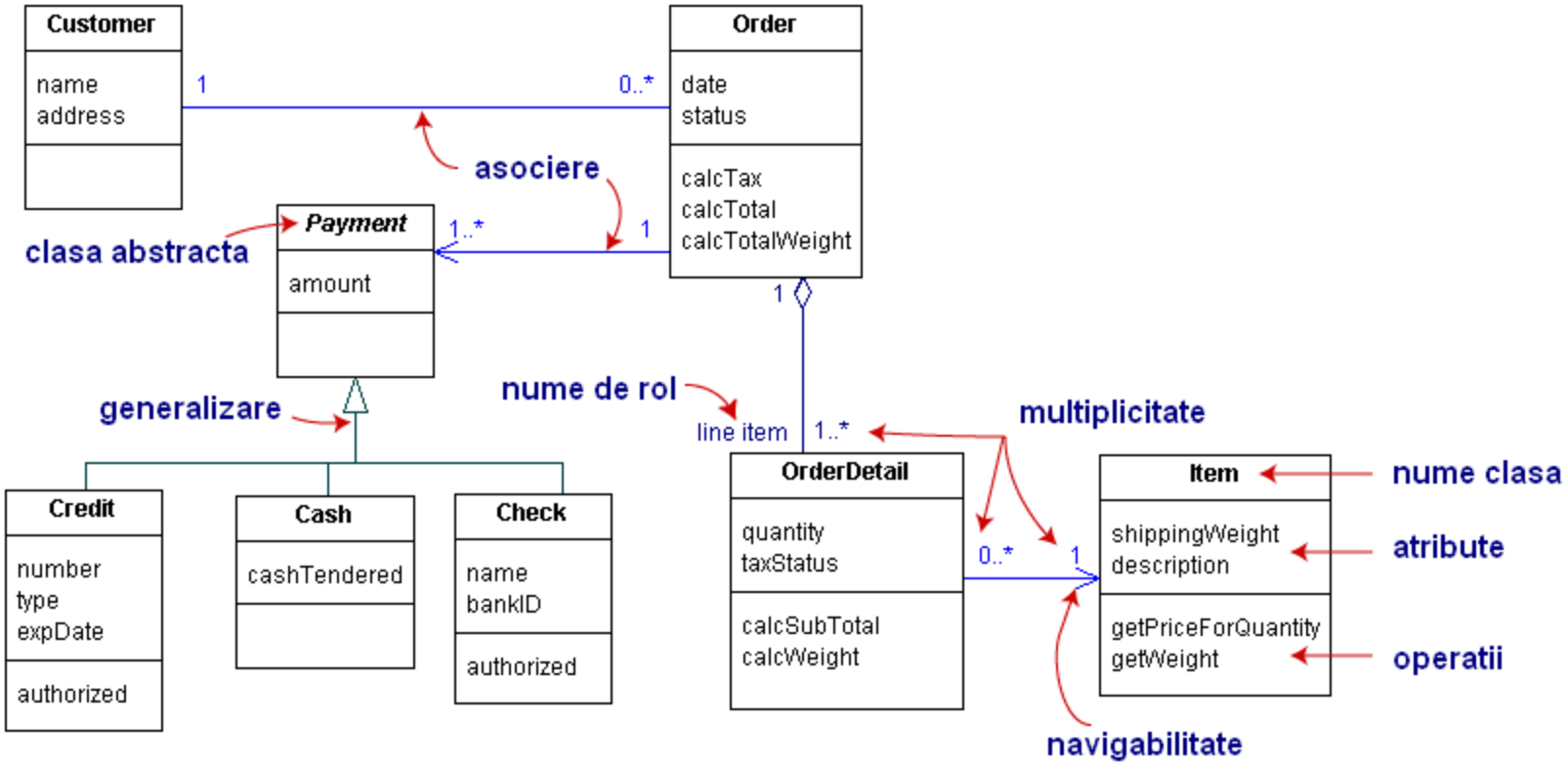


Forma * f;

f: pointer la un obiect Articol de text, Linie, Dreptunghi sau Elipsa

f-> afisare()
va fi executata operatia specifica obiectului punctat de f

Diagrama de clase conceptuale
- clase si relatii intre clase -



Diagrame de clase: utilizari

1) In modelarea conceptuala (analiza orientata obiect) si specificarea software

- Clasele corespund entitatilor (conceptelor / obiectelor) din domeniul aplicatiei.
- Nu exista neaparat o legatura directa cu clasele de obiecte utilizate in implementare

si deci diagrama de clase nu face parte din modelul structural (de proiectare) al sistemului.

- De regula, nu se definesc tipurile atributelor si nici cele ale parametrilor operatiilor.

2) In proiectarea de detaliu si implementare

- Diagramele contin clase de obiecte implementate intr-un limbaj de programare.
- Diagramele fac parte din modelul structural al sistemului.

Diagrame de obiecte conceptuale (1)

- O diagrama de obiecte este o instanță a unei diagrame de clase.
- Obiectele: instante ale claselor
- Legaturile dintre obiecte: instante ale relatiilor dintre clase

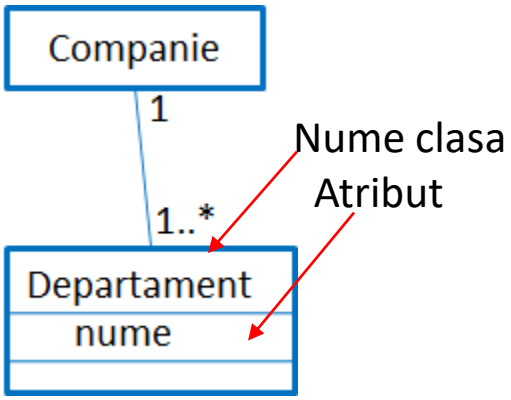
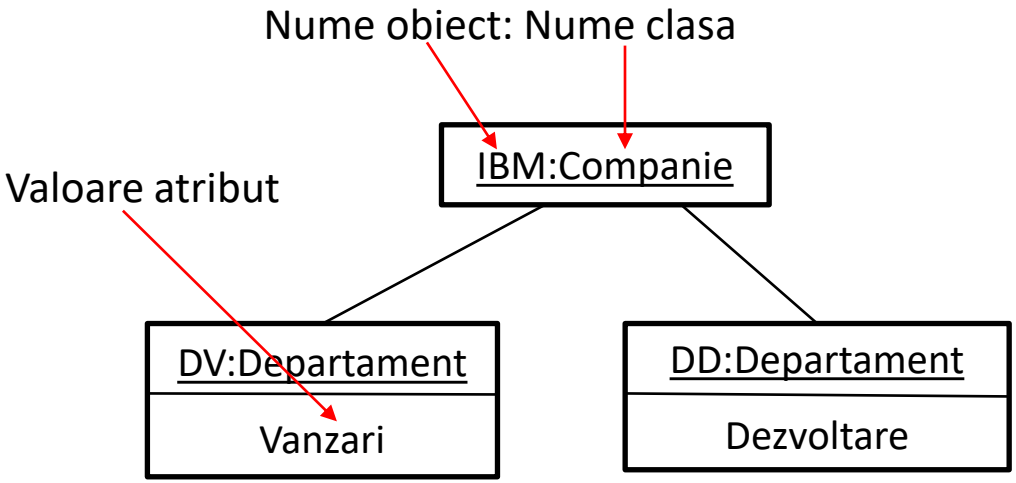


Diagrama de clase

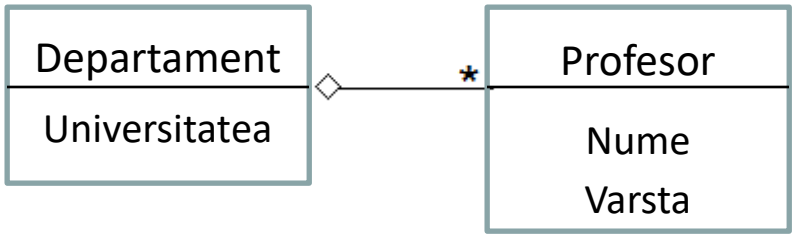


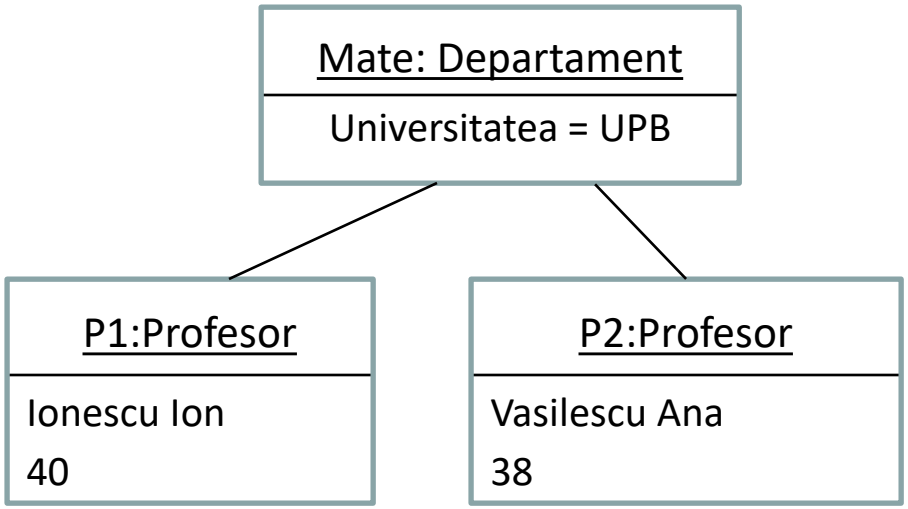
Diagramă de obiecte

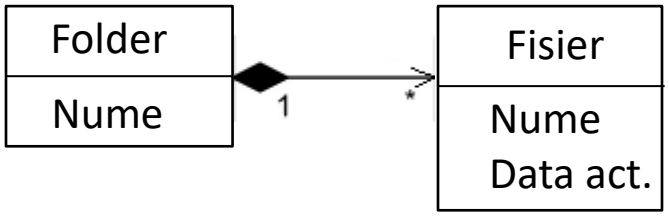
- Diagramele de obiecte conceptuale se folosesc pentru:

- a ilustra d.p.d.v. structural starile mai importante la executia unui sistem.
- a reda legaturile dintre obiecte la diferite momente ale executiei sistemului

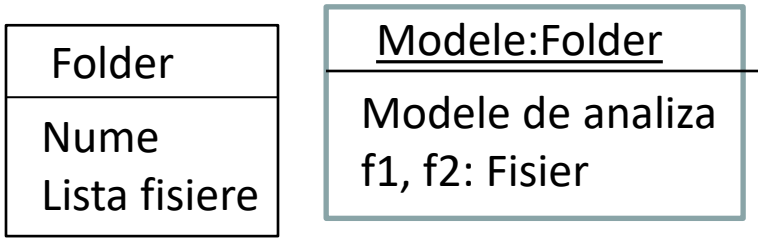
Diagrame de obiecte conceptuale (2)

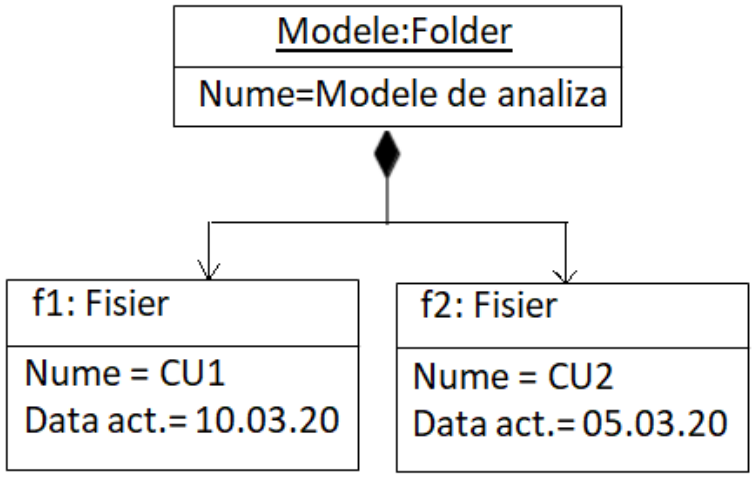






echivalent





Obiectul Modele contine obiectele f1, f2.