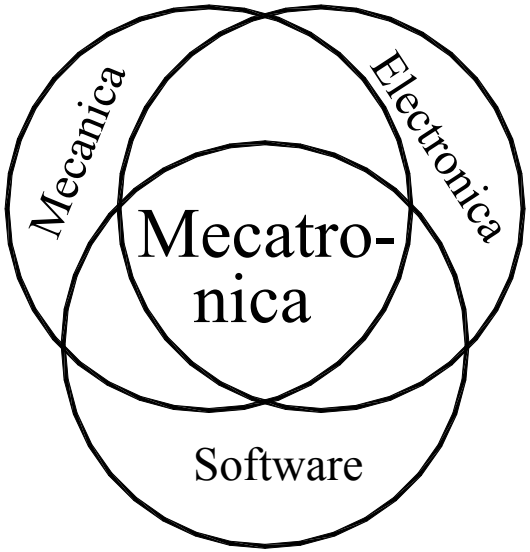
Departamentul de MECATRONICĂ
Facultatea de MECANICĂ
UNIVERSITATEA POLITHENICA TIMIŞOARA

PROIECTAREA SISTEMELOR MECATRONICE

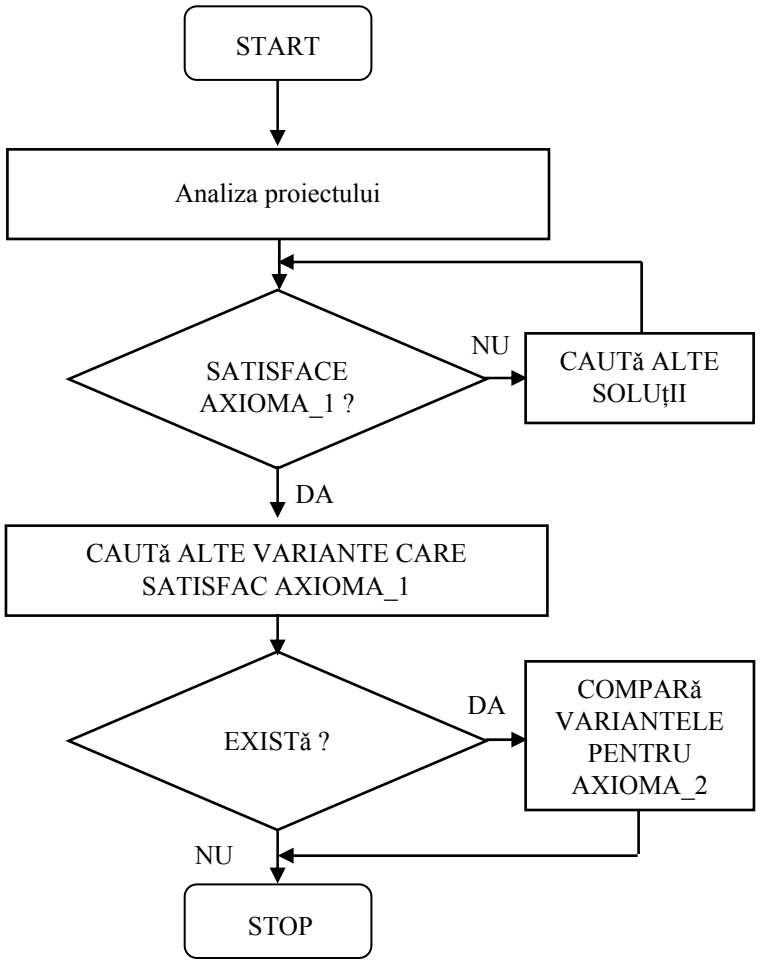


Prof. dr. ing. Valer DOLGA,

Cuprins
CAD

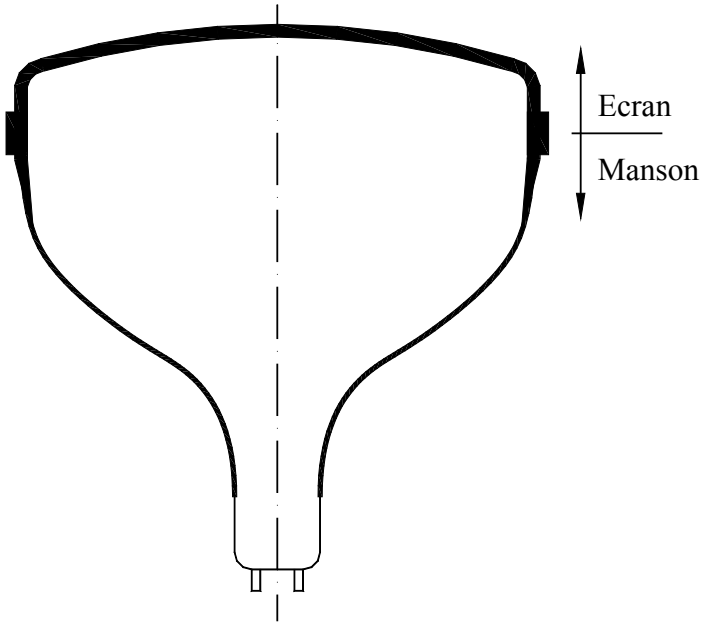
- Proiectarea axiomatica
- Metoda TRIZ
- Proiectarea orientata obiect

Proiectare axiomatica – algoritmul de proiectare

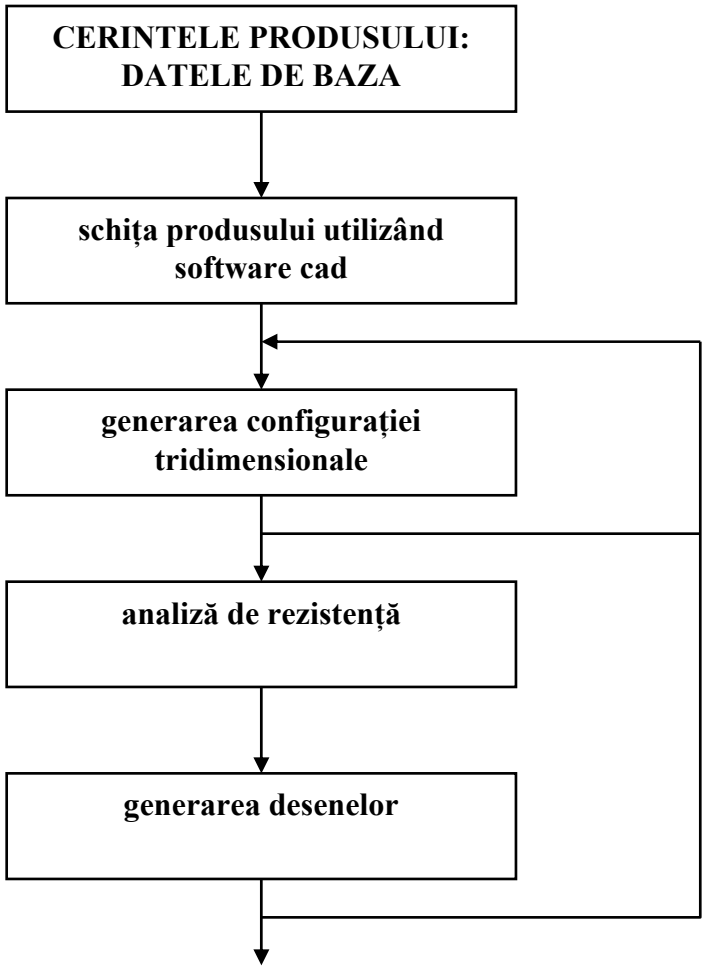


- Determinarea atributelor consumatorului / beneficiarului;
- Descrierea cerințelor funcționale FRs şi parametrii de proiectare DPs pentru procesul de proiectare convențional;
- Construirea matricii de proiectare;
- Căutarea surselor de cuplare;
- Stabilirea direcției de eliminare a cuplajelor;
- Determinarea FRs şi DPs şi a matricii de cuplare pentru noul proces de proiectare;
- Descompunerea în mod ierarhic a setului FRs – DPs;
- Utilizarea noi stări în proiectare.

Proiectare axiomatica
Exemplu



Tub cinescop



Stabilirea bazei de date

Procesul de proiectare convențional - 5 etape conform algoritmului:

1. ***stabilirea bazei de date***: date referitoare la curbură (raza, profilul, ecuația polinomială), specificațiile componentei, dimensiunea ecranului, desene. ⟶ cerințele funcționale:
   - FR1 – construirea bazei de date pentru noul produs;
   - FR2 – stabilirea configurației pentru produs;
   - FR3 – verificarea caracteristicilor produsului;
   - FR4 – generarea documentației grafice pentru produs.

⟶ parametrii de proiectare care să satisfacă FRs:

- DP1 – setul de date pentru noul produs;
- DP2 – configurația tridimensională;
- DP3 – condiții de lucru;
- DP4 – un set de date suplimentar pentru desen.

$$FR_1 \longrightarrow \begin{cases} FR_{11} - \text{alocarea unui număr pentru noul produs} \\ FR_{12} - \text{construirea unui set de date pentru produs} \end{cases}$$

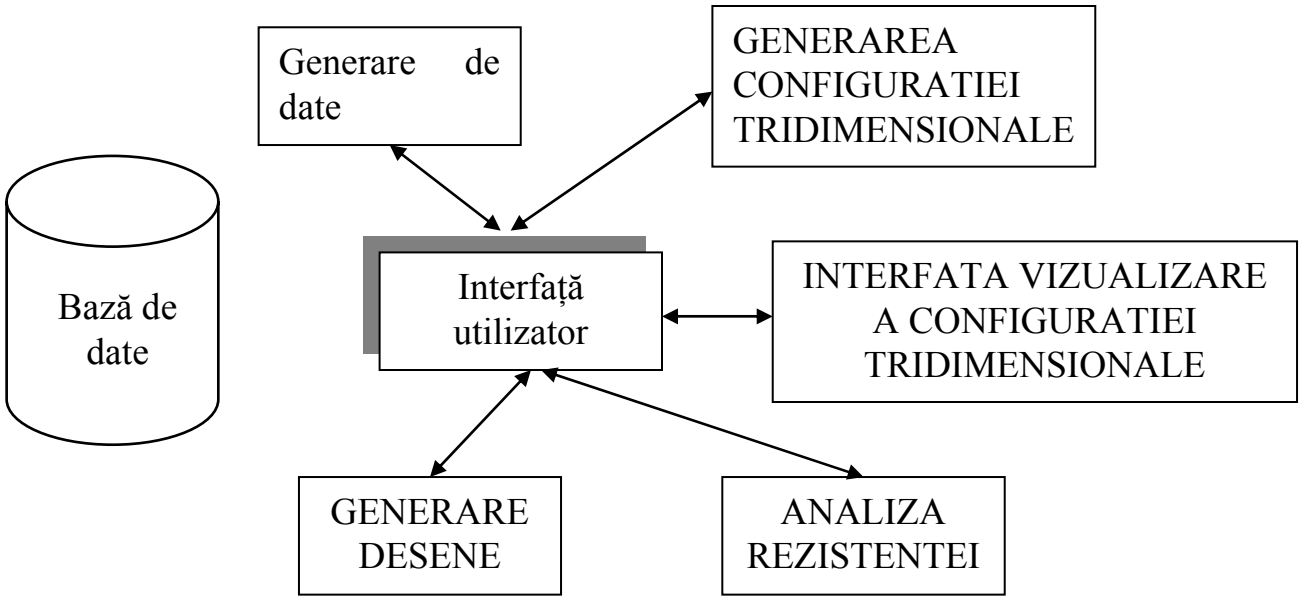
Schema bloc a software-lui

$FR_2 \rightarrow$
- FR_{21} – controlul curburii
- FR_{22} – calculul configurației tridimensionale
- FR_{21} – consideră tehnologia de fabricație

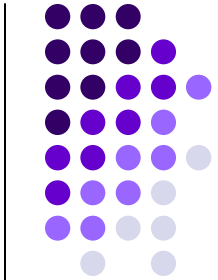
- **DP11 -** codul pentru noul produs
- **DP12** - un set specific de date

$$\begin{Bmatrix} FR_{11} \\ FR_{12} \end{Bmatrix} = \begin{Bmatrix} x & 0 \\ x & x \end{Bmatrix} \cdot \begin{Bmatrix} DP_{11} \\ DP_{12} \end{Bmatrix}$$

→ structura schemelor bloc si schema bloc a software-lui



Metoda TRIZ



• TRIZ – acronimul in limba rusa pentru - theory of inventive problem solving; **Genrikn Altshuller** - 1975 – 1986

• "Soluțiile standard 76" - bază de date - 5 clase

- ❖ clasa 1 în 2 grupe;
- ❖ clasa 2 în 4 grupe;
- ❖ clasa 3 în 2 grupe;
- ❖ clasa 4 în 5 grupe;
- ❖ clasa 5 în 5 grupe

Clasa	*Categoria*	*Număr de soluții*
1	**Îmbunătățirea sistemului cu sau fără uşoare schimbări**	**13 soluții standard**
2	**Îmbunătățirea sistemului prin schimbarea acestuia**	**23 soluții standard**
3	**Tranziții ale sistemului**	**6 soluții standard**
4	**Detecție şi măsurare**	**17 soluții standard**
5	**Strategii pentru simplificare şi îmbunătățire**	**17 soluții standard**

Solutii standard si metode de lucru

Grupa 4-1	**În loc de măsurare şi detecţie – schimbă sistemul**
Grupa 4-2	**Sinteza sistemului de masurare**
Grupa 4-3	**Extinderea sistemului de măsurare**
Grupa 4-4	**Tranziţia spre sistemele de măsurare feromagnetice**
Grupa 4-5	**Evoluţia sistemelor de măsurare**

Metologia TRIZ se bazează pe un set de ***metode şi strategii*** de lucru:

• ***Conceptul idealist***. Orice sistem prin funcţiile sale are *efecte utile şi efecte nocive*. Scop primordial - maximizarea efectelor utile. Două direcţii scop:

- ❖sistemele tehnice evoluează în sensul creşterii proporţiei de ideal;
- ❖direcţionarea rezolvitorului de probleme spre conceptualizarea perfecţiunii şi ruperea inerţiei psihologice sau a paradigmelor.

• ***ARIZ*** – un algoritm necomputaţional pentru rezolvarea problemelor de inventică;

• ***Tabelul contradicţiilor*** – o baza de date cu 1263 de contradicţii inginereşti. Contradicţia = efectul negativ asupra unui parametru datorită unui efect pozitiv asupra altui parametru.

Metode si strategii

- ***Principiile inventicii*** – un instrument de lucru cu 40 de principii şi aproximativ 50 de subdomenii; ***studiu pe 40.000 patente*** (initiat pe 400.000 patente); → clasificare ierarhică pe nivele inovative şi selecția celor mai bune (~ 21 %);
- ***Principiul separației***
- ***Legile evoluției sistemelor ingineresti***
- ***Analiza funcțională***

- Masura idealitatii: $$ideal = \frac{\sum beneficii}{\sum cost + \sum efecte\_nocive}$$

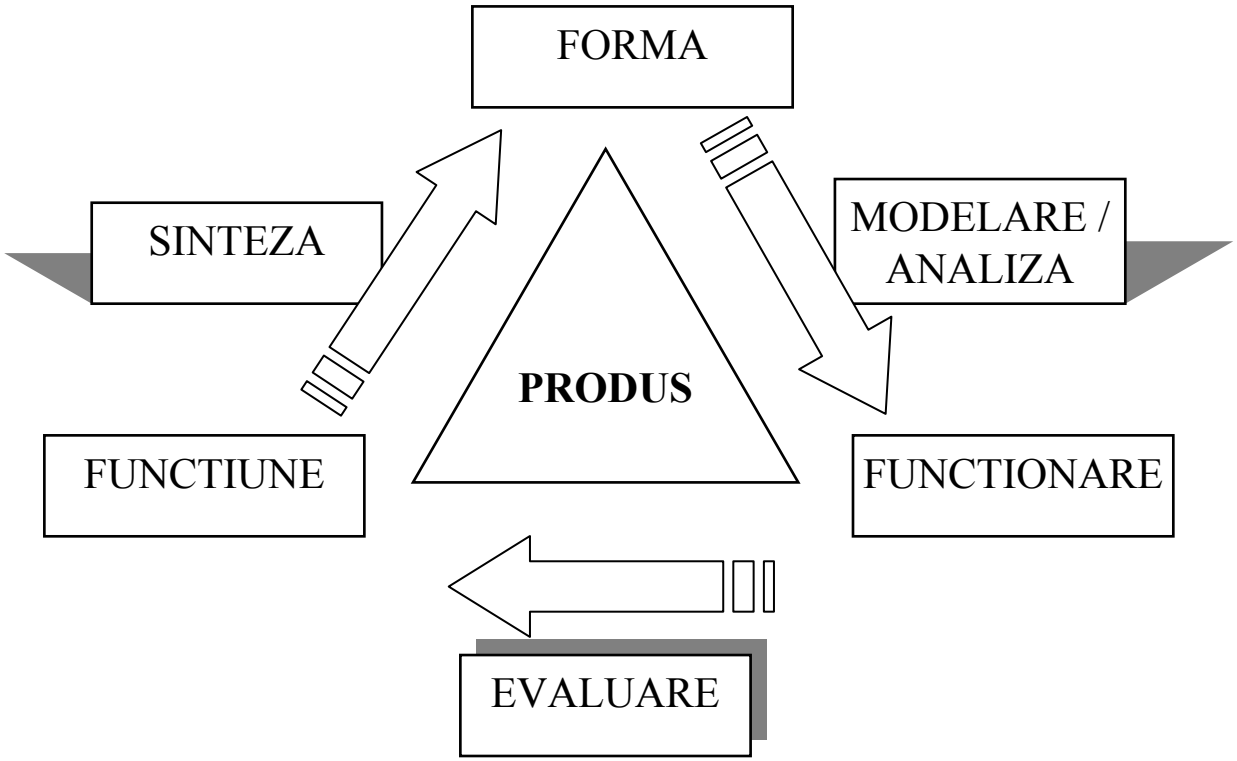
$\sum beneficii$ - cuantifica functiile utile ale sistemului;

$\sum cost$ - cuantifica costurile directe şi cele pentru societate

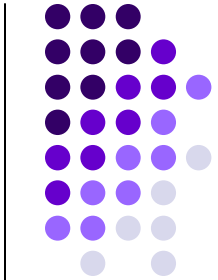
$\sum efecte\_nocive$ - Cuantifica moduri de defectare, funcții nocive, aspecte nedorite din ieşirile sistemului proiectat

CAD - inteligent

- ***instrument*** util pentru generarea produselor complexe
- ***Forma, funcţiile şi funcţionarea*** unui produs interacţionează şi se condiţionează reciproc.



Programarea orientata obiect



• ***Abordarea orientată spre obiecte*** - inițiată in 1957; proiectul american pentru racheta balistică Minuteman.

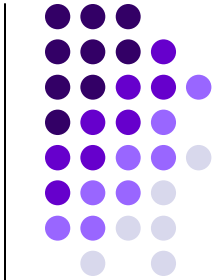
• Abordarea orientată spre obiecte este specifică ***lucrului în echipă***:

- ❖ se asigură o comunicare eficientă între membrii diverselor colective;
- ❖ împărțirea aplicațiilor în mai multe module; cel ce dezvoltă un modul nu trebuie să cunoască detaliile de implementare a altor module.

• ***programarea orientată pe obiecte*** - un *concept natural*:

- ❖ zi de zi avem de-a face în activitatea obişnuită cu *obiecte conectate* între ele, *comunicând* unele cu altele într-un anumit mod;
- ❖ În plus în natură o entitate este caracterizată atât prin structura sa cât şi printr-un anumit comportament. În natură obiectele evoluează în timp, adeseori modificându-şi structura şi funcționalitatea.

Programare orientata obiect / incapsularea



• ***Proiectarea orientată obiect*** - ***o strategie*** în care sistemul se gândeşte în termeni de ***"obiecte",*** în loc de ***operații şi funcții***.

• Un ansamblu = *obiect* sau clasă;

• Proiectarea de programe utilizând clase = programare orientată pe obiecte (OO).

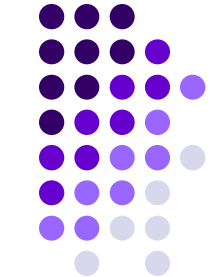
• ***Programul*** = mulțime de obiecte care interacționează, oferă servicii altor obiecte şi îsi gestionează starea internă.

• ***Conceptele fundamentale:***

- ❖ ***Incapsularea:***
  - ❑ structurile de date = *date membre*
  - ❑ procedurile = *functii membre* sau *metode*

$$date + metode = obiect$$

Sintaxa - clasa



• *categoria de "clasă"* - se asociază unui obiect real sau virtual - înglobând o serie de subcategorii:

- ❖ *Structură informațională* - un set de atribute asociat clasei;
- ❖ *Funcții de acces autorizat* - permit operații de manipulare a obiectului respectiv;
- ❖ *Funcții suplimentare* (ascunse utilizatorului clasic) - definesc comportamentul obiectului în mediul său de evoluție.

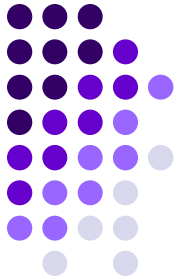
• sintaxă simplificată a declarării unei clase:

```
class NumeClasă
{
......
declarații variabile membre
......
declarații funcții membre
..
}
```

• asemănător limbajului C - trebuie să definim o variabilă de acel tip:

NumeClasă variabilă

Programarea OO / Mostenirea



❖ ***mostenirea*** - permite construirea de *clase derivate* pentru care:

- ❑ se păstrează unele structuri informaționale
- ❑ se pastreaza funcții de acces la structurile informatice,
- ❑ preluarea - din una / mai multe clase de referință = clasă de bază;
- ❑ o ierarhizare a obiectelor bazată pe criterii logice şi funcționale

❖ ***poliformismul -*** atribuirea aceluiaşi nume unei funcții de tratare a unor structuri informaționale; implementarea în mod dinamic în momentul rulării programului; recunoaşterea automată a tipului clasei obiectului referit.

• obiectul proiectat = incapsulat = proprietati + comportament;

• proprietățile - modelate prin parametrii (atributul nodului):

❑ geometria şi proprietățile fizice, forțe generalizate, poziții, viteze etc.; forma fizică = *date*:

➢ *proprietăți permanente:* asemănări geometrice, masă, coeficient de frecare etc.;

➢ *stare temporară* a obiectului: forță, poziție, viteză.

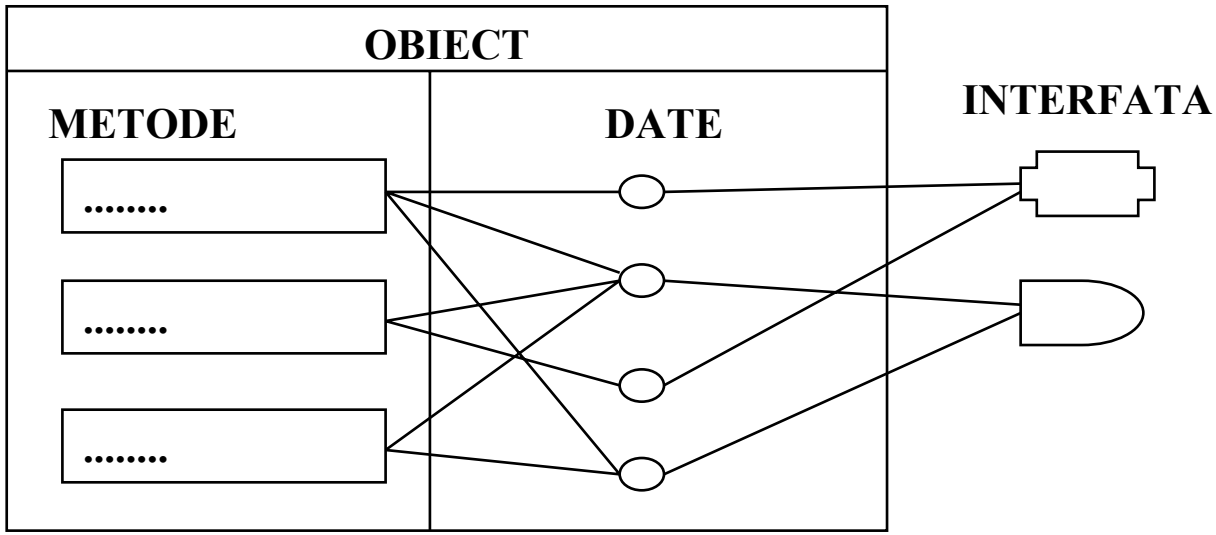
Proiectare OO – domeniul electromecanic

• prima categorie poate fi modificată doar prin metode de dimensionare relativ la proce;

•a doua categorie corespunde unor metode ce descriu o funcționare dinamică.

• funcționarea *obiectului* = interdependențele parametrilor (de ex.: masă, accelerație şi forță);

• instrucțiunile care descriu combinațiile = *metode* sau *constrângeri.*



Interfata

Interfeța:

• *Interfață externa* care descrie o interacțiune fizică energetică, masă, informație a obiectului cu mediu;

• *Interfață internă – implementare* – care pune în vedere funcționarea internă

• ***obiectele interactioneaza intre ele***;

• interacțiune – noțiunea de *port* sau *conector* care intră în componența interfeței obiect; reprezentare –

• **Portul** = punctele sau conturul prin care componentele interacționează interacționează **între ele** prin *schimb de energie* şi **cu mediul exterior**;

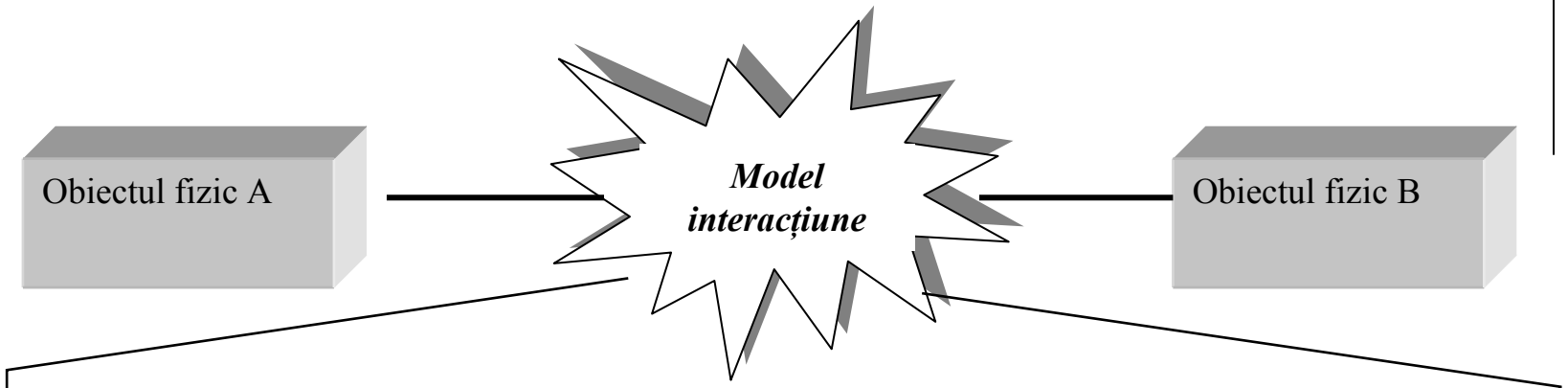
• grafic interacțiunea se reprezintă prin linii trasate între porturi.

Energia vehiculată printr-un port - descrisă de două variable:

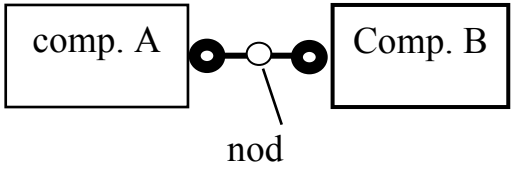
• o variabilă peste / transversală (*across*);

• o variabilă de transfer / trecere (*through*).

Modul de interactiune a obiectelor



Domeniu electric

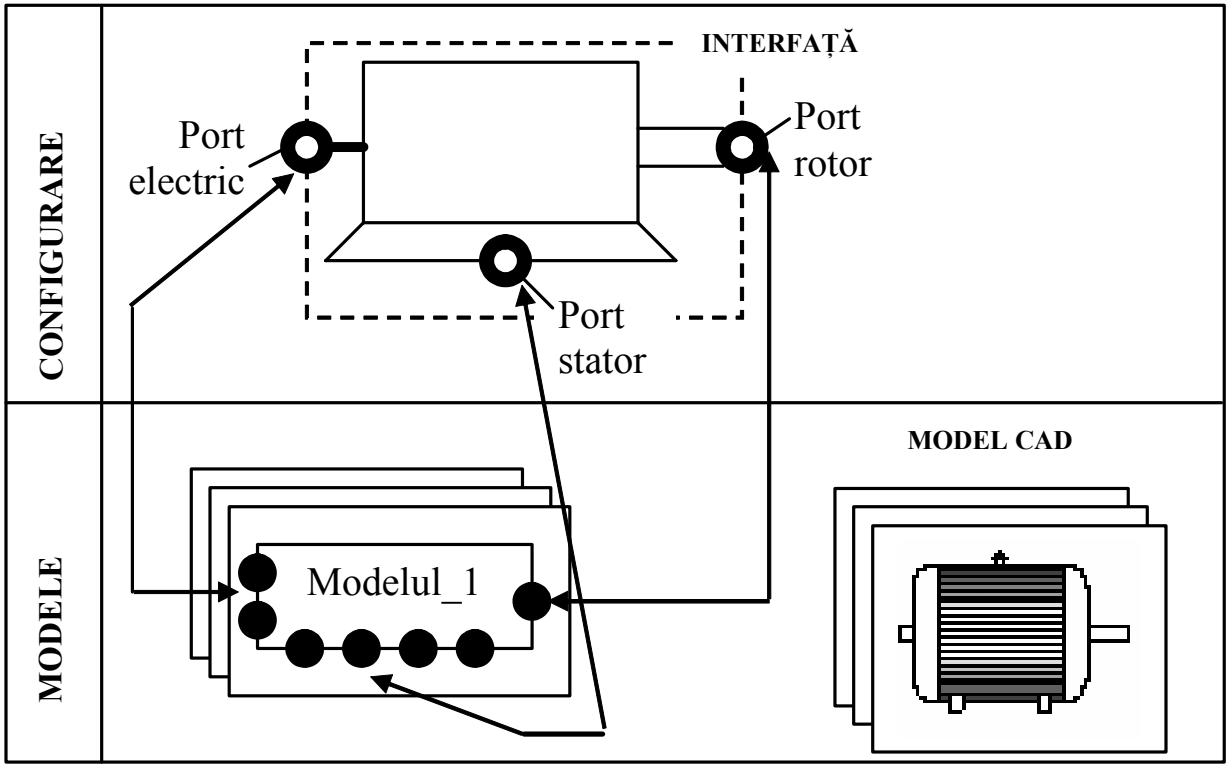


$$\begin{cases} va\_acr_A = va\_acr_B \\ \sum va\_tr = 0 \end{cases}$$

Domeniu Mecanic

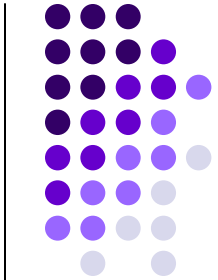
- Modelul cuplei cinematice
- Modelul transmisiei
- Modelul frecării

Porturile unui motor electric



• Portul 1 – modul de fixare a statorului față de mediu şi se concretizează printr-un număr de orificii pentru fixare;
• Portul 2 – calea prin care motorul se conectează în sistem vehiculând energie mecanică şi se concretizează prin arborele rotorului;
• Portul 3 – calea prin care motorul primeşte energie electrică dinspre exterior şi este concretizat prin conectorul electric.

Modele – configurarea motorului electric



- ***Modelul funcțional*** –
  - ❖modelul de tip interfață (funcționarea porturilor şi parametrii modelului)
  - ❖componenta de implementare. Implementarea este realizată sub forma seturilor de ecuații referitoare la funcționare***.***
- ***Modelul CAD*** - două scopuri majore:
  - ❖o reprezentare cu specificații referitoare la forma componentei (dimensiuni nominale, toleranțe, materiale);
  - ❖ o reprezentare matematică a geometriei obiectului (de ex.: vizualizare componente).

Domeniul electric: cele două variabile = ***tensiunea*** şi respectiv ***curentul***

- ❖class voltage = Real
- ❖class current = Real

Variabila reală = set inițial de atribute referitoare la:

- ❑unitatea de măsură;
- ❑valoarea inițială;
- ❑valoarea minimă şi maximă.

Exemple de clase

- De ex.: tensiune alternativă:

```
class Voltage = Real (unit="V", min=-220.0, max=220.0);
```

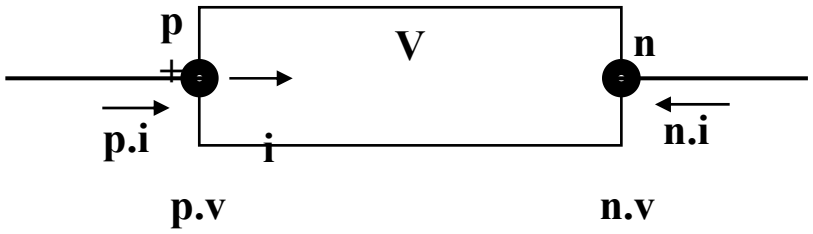
- Clasa port / conector:

```
connector Pin
Voltage v;
Flow Current  i;
end Pin
```

- definirea în raport cu doi conectori: clasa model *Doi_pini* (TwoPin)

```
class TwoPin
        Pin p, n;
        voltage v;
        Current  i;
  Equation
        v = p.v – n.v;
        0 = p.i + n.i;
        i = p.i;
end TwoPin;
```

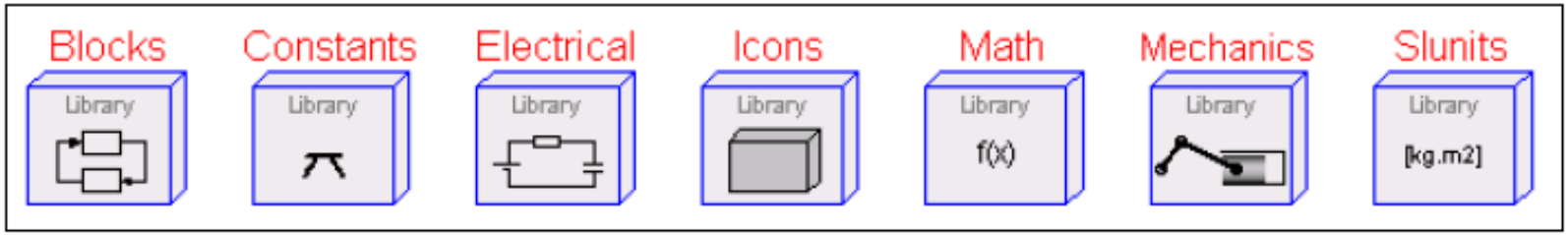
Rezistor, capacitate;



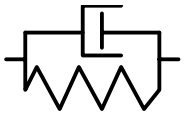
Exemplul_2

```
class Rezistor "rezistorul electric ideal"
        extends TwoPin
        parameter Real R (unit = "Ohm");
Equation
        R * i = v;
end Resistor;
```

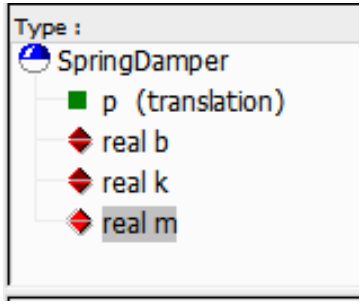
Biblioteca de modele (Dymola):



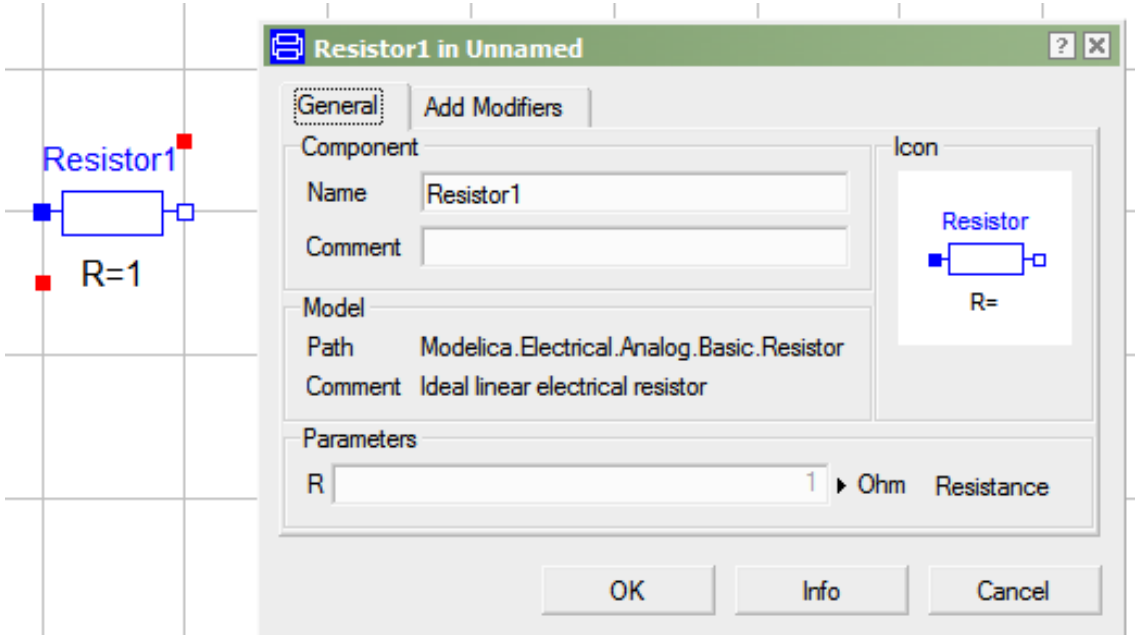
20_SIM:



20-sim3.6 Viewer (c) CL



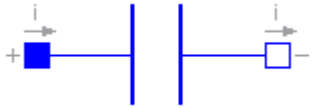
Rezistor – definirea parametrilor



model Capacitor "Ideal linear electrical capacitor“

extends Interfaces.OnePort;

parameter SI.Capacitance C=1 "Capacitance";



equation

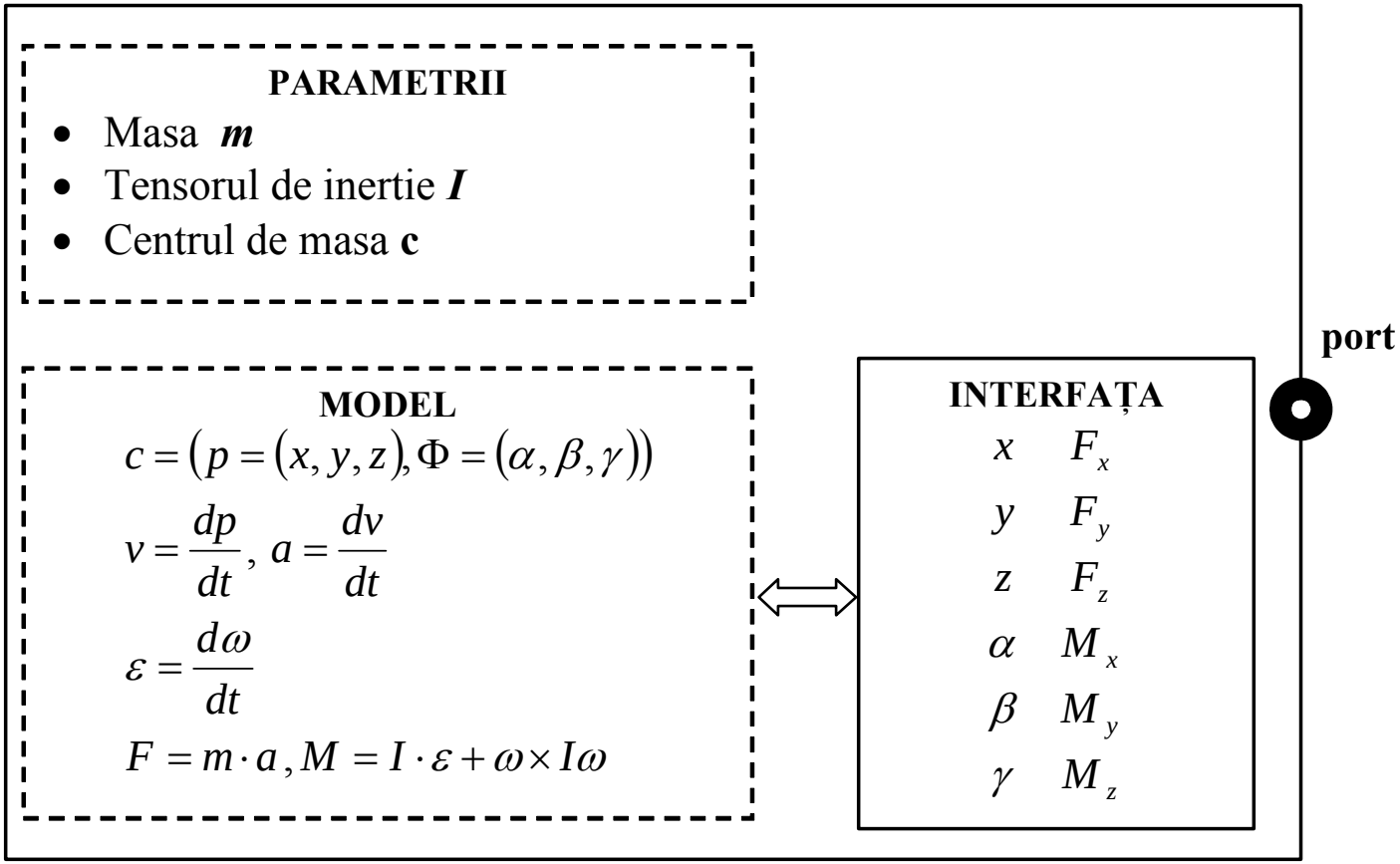
i = C*der(v);

end Capacitor;

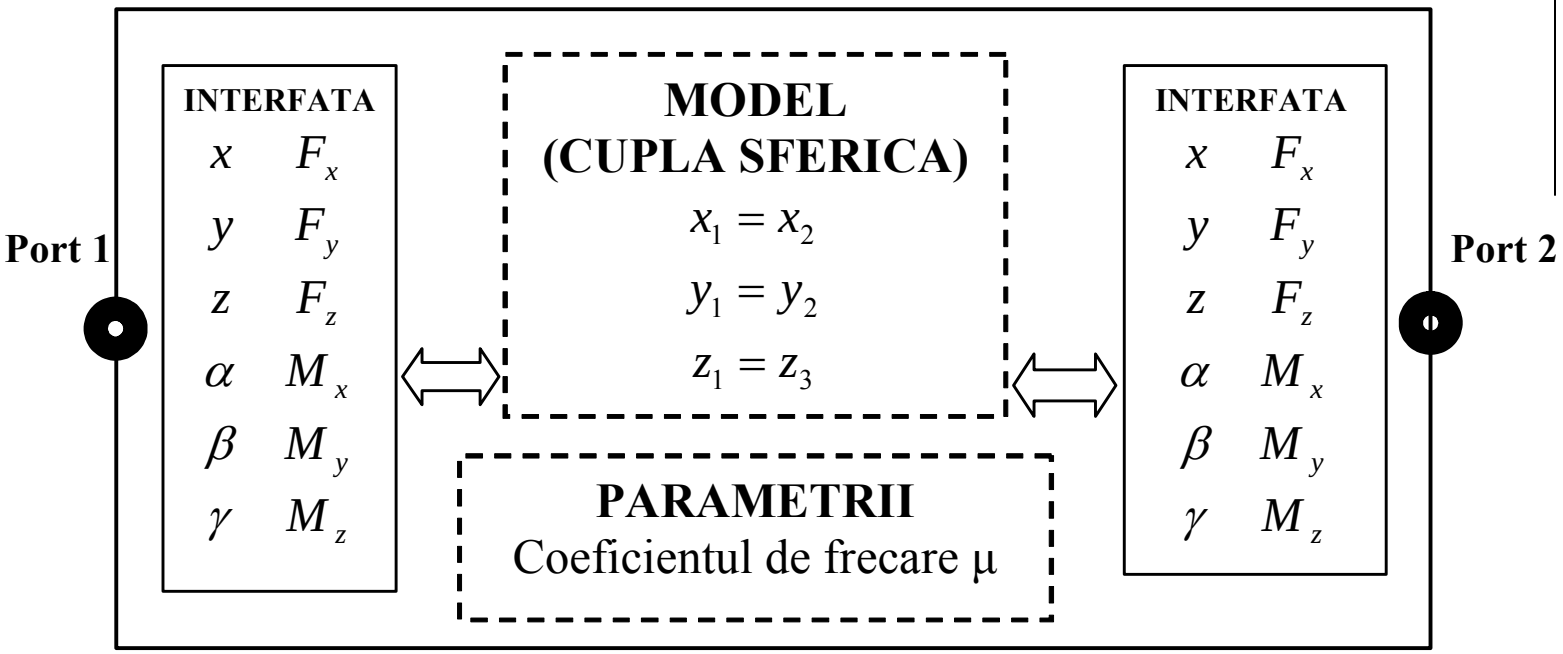
OO si elementul rigid

- **un element rigid este descris:**
  - ❖ **printr-un punct corespunzător centrului de masă**
  - ❖ **tensorul de inerție;**
  - ❖ **situarea acestuia – poziția şi orientarea – se exprimă relative la un sistem de referință global;**
  - ❖ **modelul de funcționare constă dintr-un set de ecuații referitoare la variabilele portului {p, Φ, F, M} unde:**
    - ❑ **perechea de variabile *across:***
      - variabila *p* (un vector) defineşte poziția centrului de masă;
      - variabila Φ defineşte orientarea corpului în raport cu sistemul de axe global
    - ❑ **perechea de variabile *through***
      - variabila *F* defineşte forța aplicată asupra elementului;
      - variabila M defineşte momentul aplicat asupra elementului

Descrierea elementului rigid



Modelul cuplei cinematice



conector / port de natură mecanică echivalent unei piese de conexiune:

connector Flange

Angle phi;
Flow Torque;

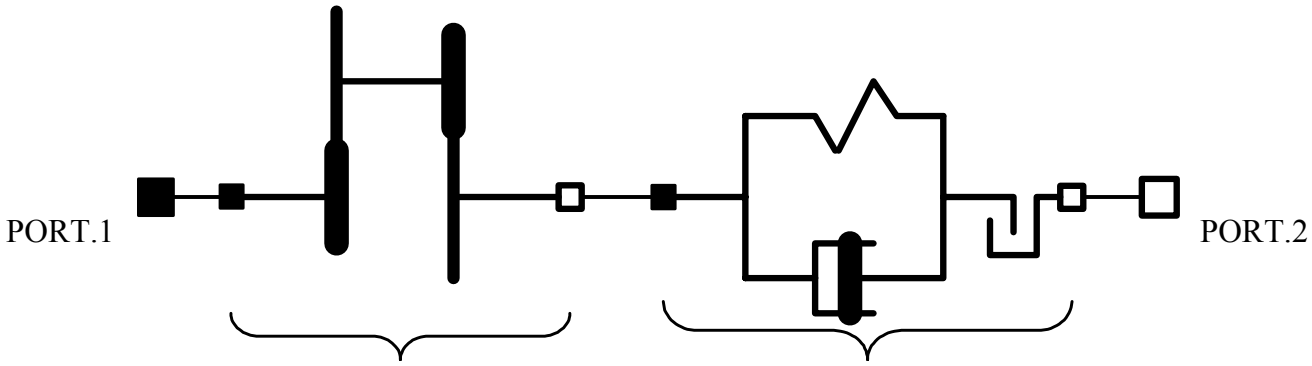
end Flange

Modelica – domeniul mecanic

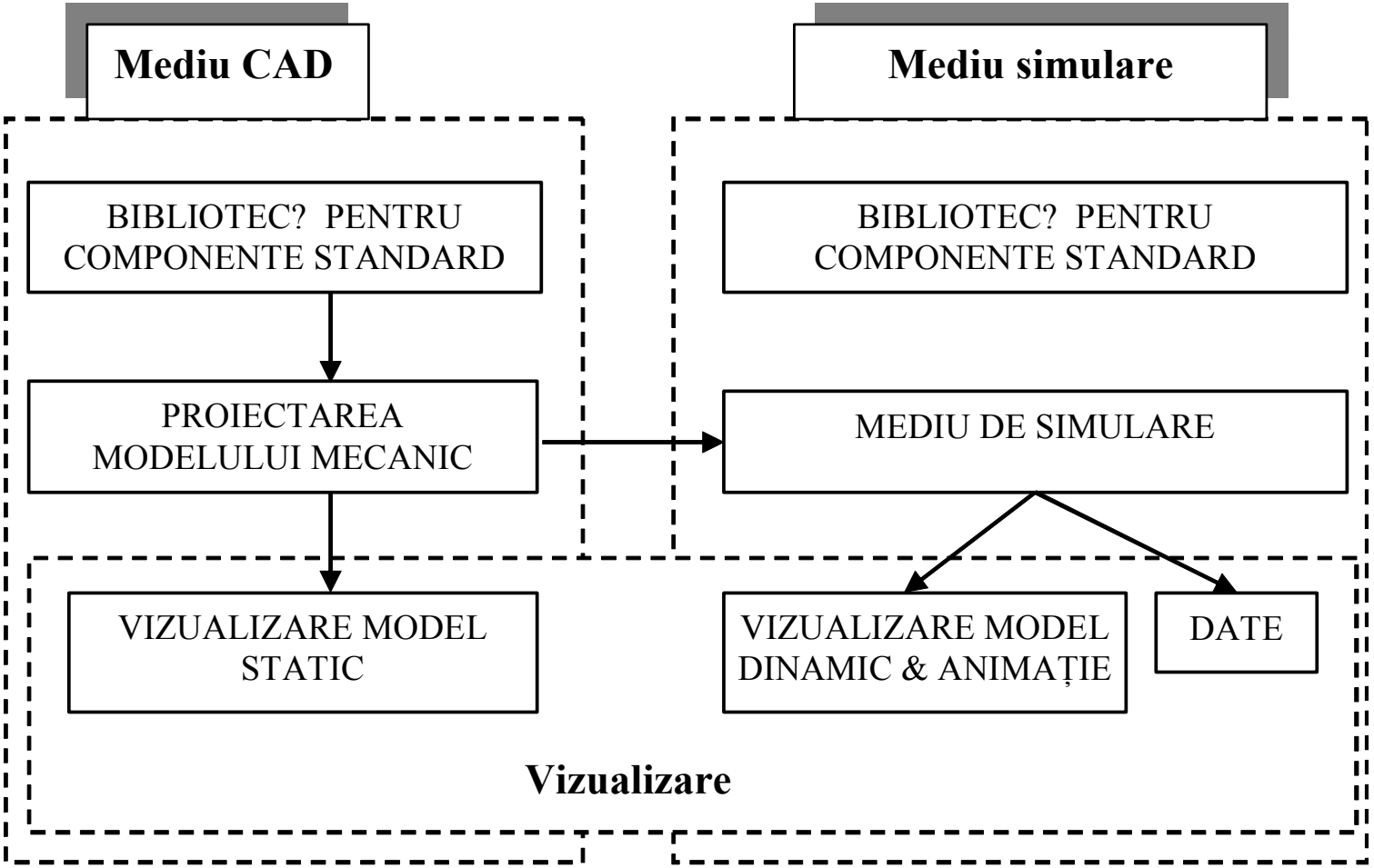
Modelica - domeniul mecanic - structurat în mod ierarhic:

- **modelica_mecanică_mişcarea de rotație:**
  - ❖ **cu subclase: componetă rotațională cu inerție, transmisie ideală R-R, transmisie ideală planetară, element elastic liniar, element amortizor liniar,…etc.**
- **modelica_mecanică_mişcarea de translație:**
  - ❖ **cu o serie de subclase specifice: componentă inerțială în translație, sensor de forță, senzor de poziție, …etc.**

EX: Transmisie reala



Mediu de lucru CAD



Mediu integrat CAD

Exemple de medii integrate CAD:

- **Visual Nastran 4D**
- **ADAMS**
- **CATIA**
- **PRO-E**
- **etc.**