Integrarea sistemelor informatice

Suport curs nr. 9

Programator >> Arhitect

Ingineria proiectării sistemelor

2024-2025

Obiective

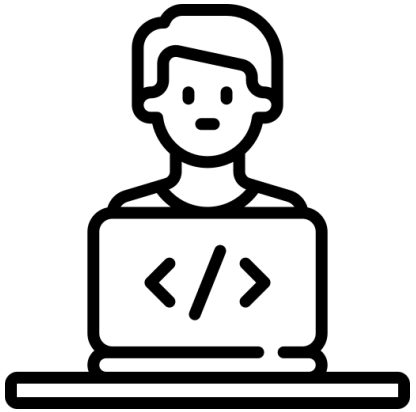
- Introducere în ingineria sistemelor
- Introducere în ingineria proiectării
- Specificarea și modelarea cerințelor

Ingineria sistemelor

Inginerie = aplicarea cunoașterii științifice, economice, sociale, practice asupra realității materiale și/sau sociale în vederea proiectării, executării, întreținerii, modificării unor structuri care să fie capabile să furnizeze/genereze rezultate, produse, procese și/sau efecte predefinite și/sau conforme unor așteptări predictibile și/sau controlabile

Etimologie: vine din latinescul "ingeniare"= a născoci

Persoanele ce se ocupă cu ingineria sunt numite **ingineri**

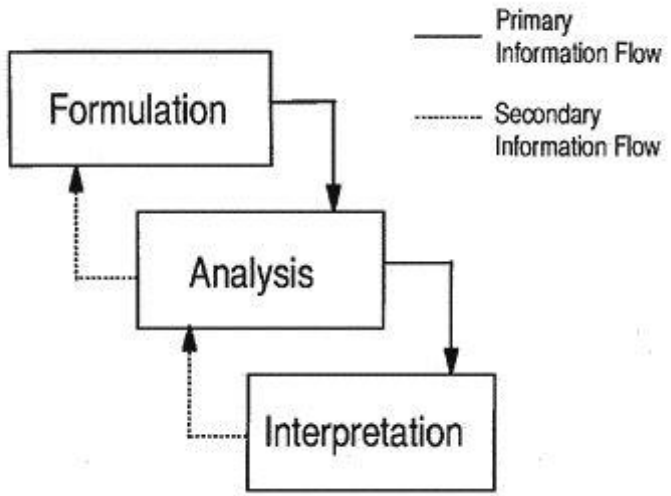


TRUST ME I'M AN ENGINEER

Ingineria sistemelor

Ingineria sistemelor este orice abordare metodică de sinteză a unui sistem complex care

- definește **perspectivele sistemului** care ajută la elaborarea **cerințelor**
- gestionează relația între cerințele de performanță, constrângeri, componente și perspective de sistem specifice domeniului



Ingineria sistemelor

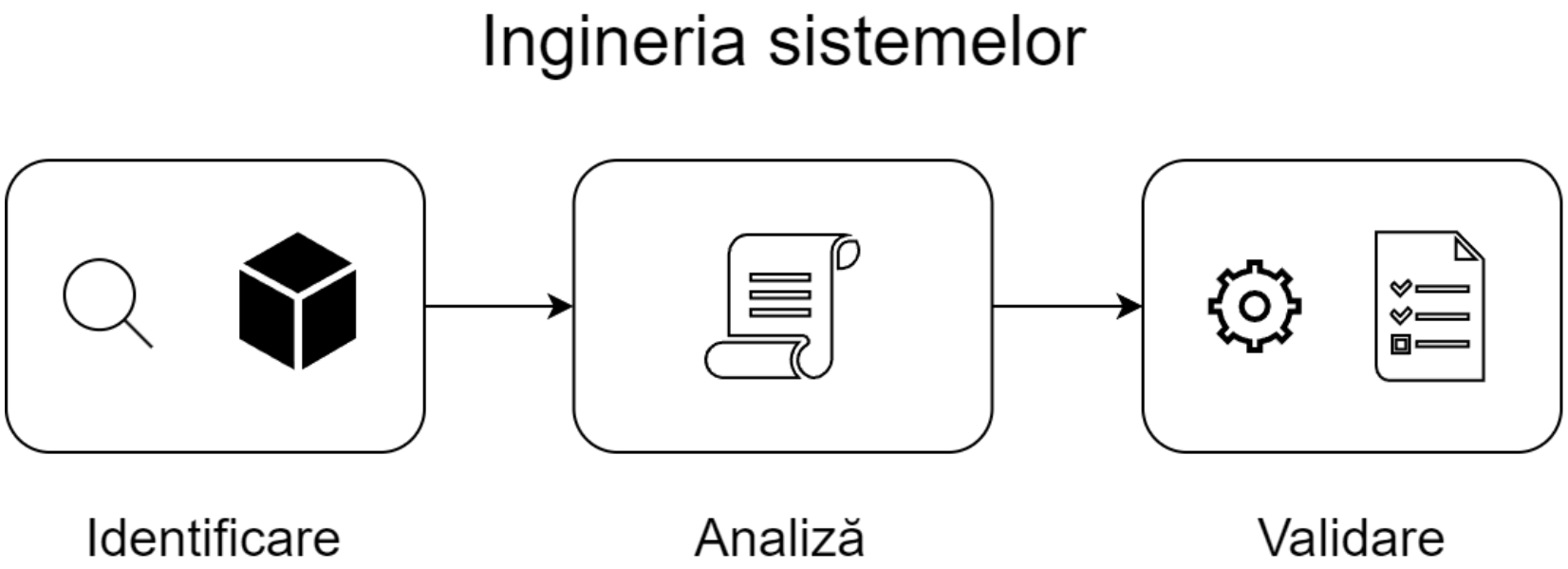
- este orientată în primul rând spre **specificarea problemei**, în timp ce alte discipline inginerești sunt orientate spre rezolvarea problemei
- oferă un plan de organizare a informației/resurselor și de direcționare a activităților spre realizarea sarcinii de integrare – acest plan este numit **procesul de inginerie a sistemelor**

Ingineria sistemelor

Procesul de inginerie a sistemelor

- identifică **punctele de vedere** relevante, capabilități și resurse de rezolvare a problemelor
- **analizează cerințele**, astfel încât să se identifice o corespondență dintre acestea și funcționalitățile sistemului, agenții de rezolvare a problemelor, precum și măsurile de asigurarea performanței; **atribuie task-urile** pentru rezolvarea cerințelor
- **evaluează metricile** și constrângerile de performanță, astfel încât succesul să poată fi recunoscut

Ingineria sistemelor

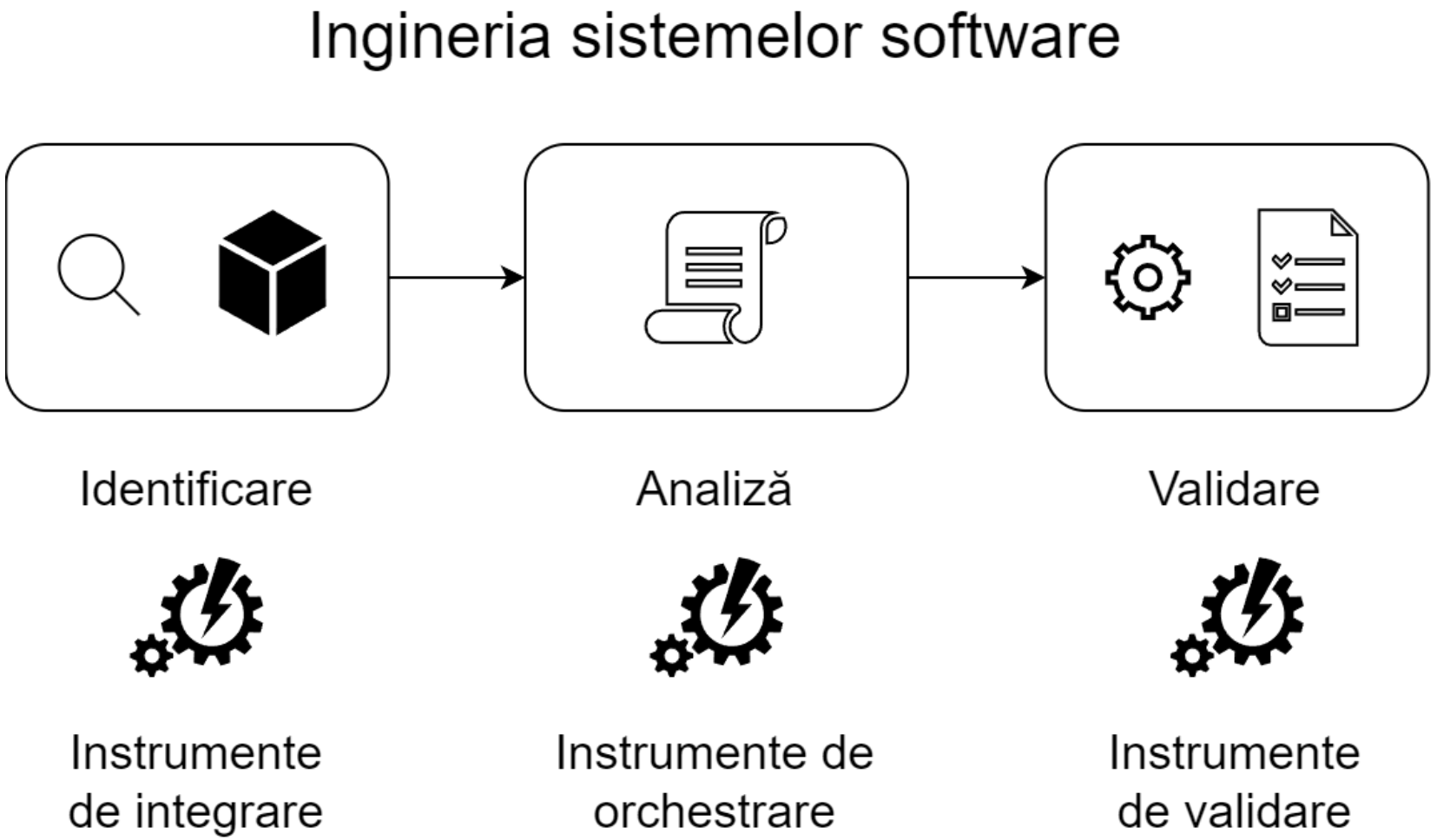


Ingineria sistemelor

Procesul de inginerie a sistemelor software

- identifică **punctele de vedere** relevante pentru aspectele de integrare și **modelele** sau instrumentele și tehnicile existente pentru dezvoltarea și reprezentarea modelelor; identifică **instrumente automate** care sunt capabile să rezolve problemele și conflictele relevante pentru cazul specific de integrare
- **analizează cerințele** de rol și de interacțiune și le reprezintă astfel încât să poată fi utilizate de toți agenții de integrare implicați => caiete de sarcini parțiale; **identifică task-urile** și orchestrează execuția tool-urilor
- dezvoltă sau utilizează instrumente de **validare**

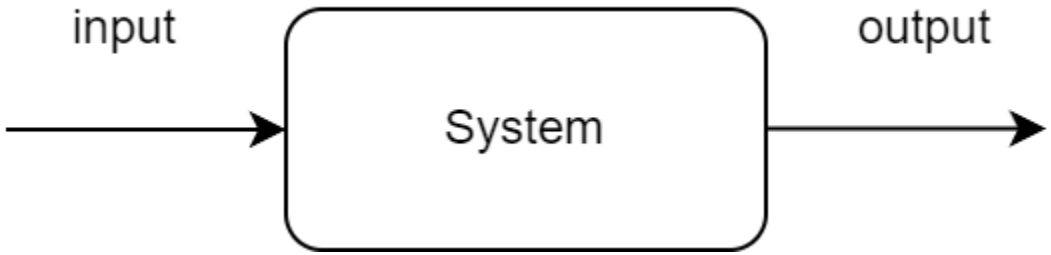
Ingineria sistemelor



Ingineria sistemelor

Procesul de inginerie a sistemelor software

- **Conceptul** sistemului = abstractizare
- **Soluția** sistemului = o instanță reprezentativă a conceptului sistemului



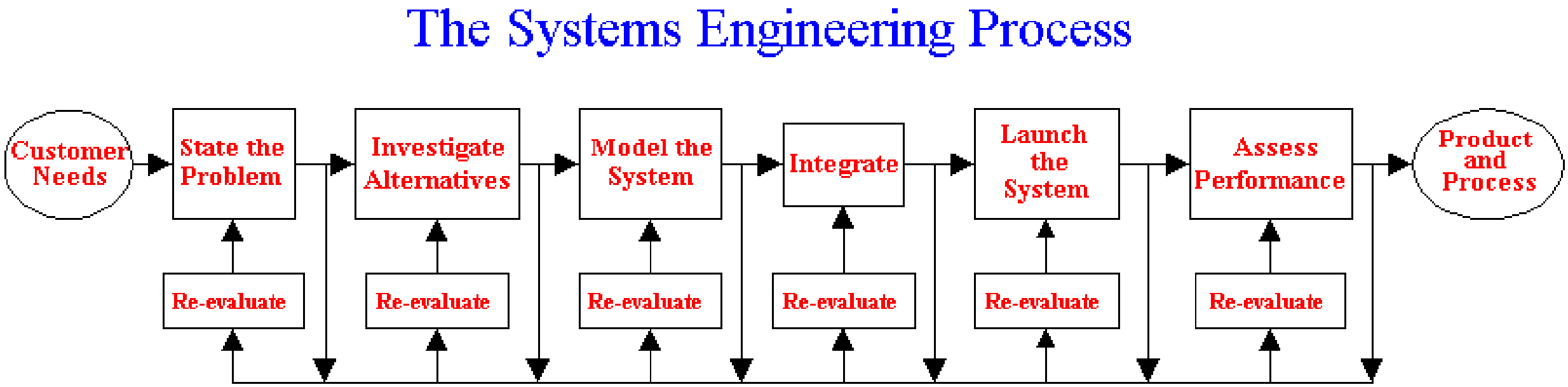
Ingineria sistemelor

Metode de inginerie a cerințelor sistemelor

- practici standard pentru **formularea cerințelor**
- standarde pentru **reprezentarea cerințelor**
- practici pentru **evaluarea și prioritizarea cerințelor**
- practici pentru **agregarea și rafinarea cerințelor**
- practici pentru **separarea cerințelor** pe diferite puncte de vedere
- instrumente pentru obținerea și **structurarea cerințelor**
- instrumente și practici pentru **analiza interacțiunilor**

Ingineria sistemelor

Procesul de inginerie a sistemelor



Specificarea și modelarea

- Modelarea **specificațiilor**
- Modelarea **funcțională**
- Specificarea **serviciilor** și **interfețelor**
- Modelarea **obiectelor** și **informației**
- Metamodele și integrarea **modelelor**

Modelarea specificațiilor

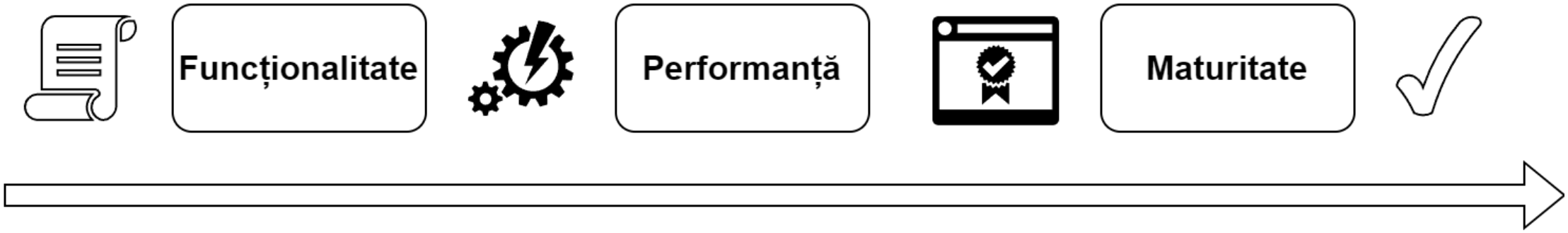
Ingineria specificațiilor

- Cerințe
  - **funcționale** – descriu funcționalitatea sistemului (use case)
  - **juridic-contractuale** – clauze de întreținere, drept de proprietate intelectuală, versiuni ulterioare – update-uri
  - **tehnologice** – scalabilitate, extensibilitate, etc.
  - **de calitate** – durata maximă de nefuncționare, etc.
  - **asistarea utilizatorului** – legătura dintre pagini, workflow
  - **modularizare** – ce pachete se livrează
  - **activități** – planificare

Modelarea specificațiilor

Caracteristici calitative ale sistemului

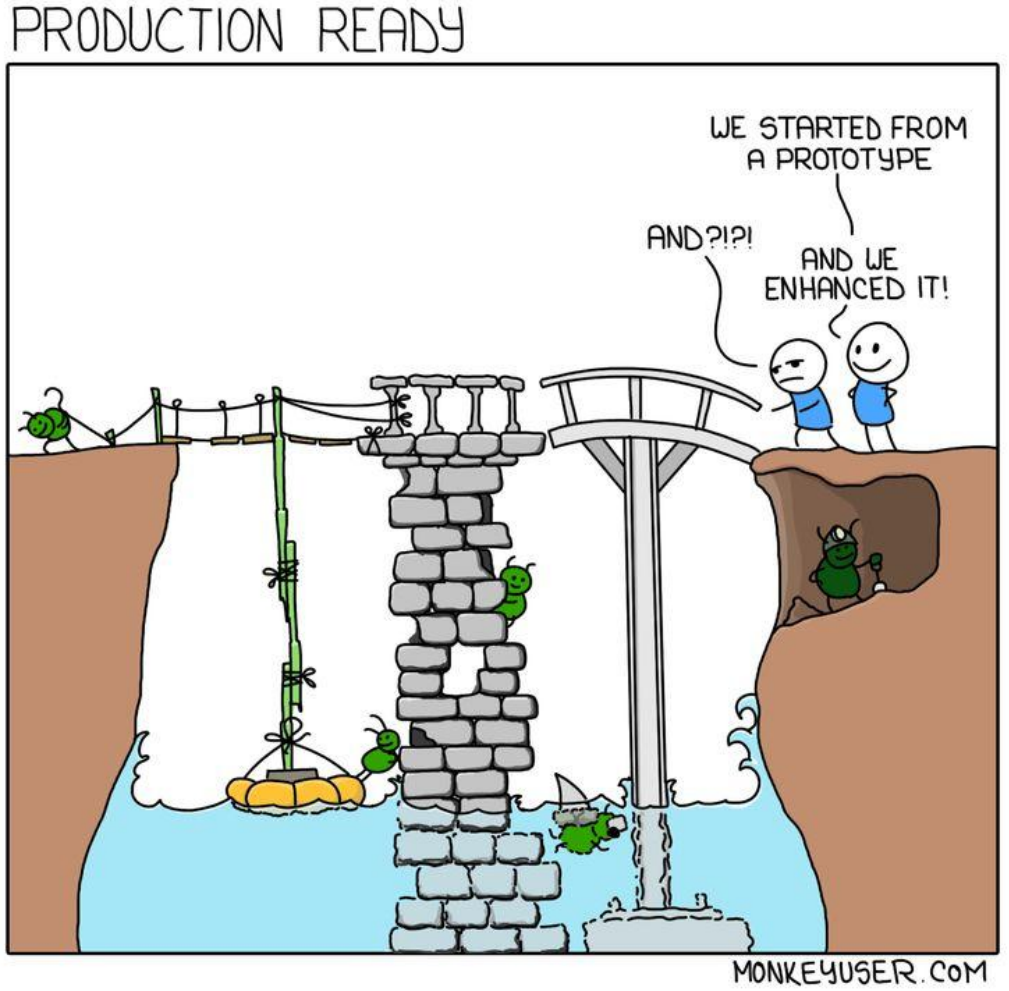
- Capabilitatea sistemului
  - acoperirea funcționalităților
  - acoperirea cerințelor tehnice / non-funcționale
- Maturitatea sistemului
  - respectarea bunelor practici din domeniu



Modelarea specificațiilor

Caracteristici calitative ale sistemului

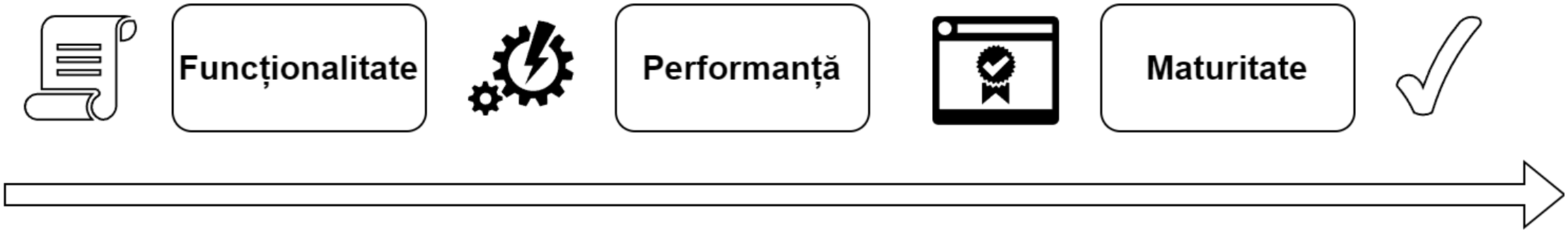
- Calitatea software-ului depinde mai puțin de limitările sale în condiții normale de funcționare, cât de situații excepționale (erori, nefuncționare / downtime), și durată de întreținere
- Când se produc cele mai mari daune în exploatarea sistemelor software? (vezi legea lui Murphy)
- Technical debt – “datorie tehnică” acumulată în timp



Modelarea specificațiilor

Cerințe tehnice / non-funcționale de sistem

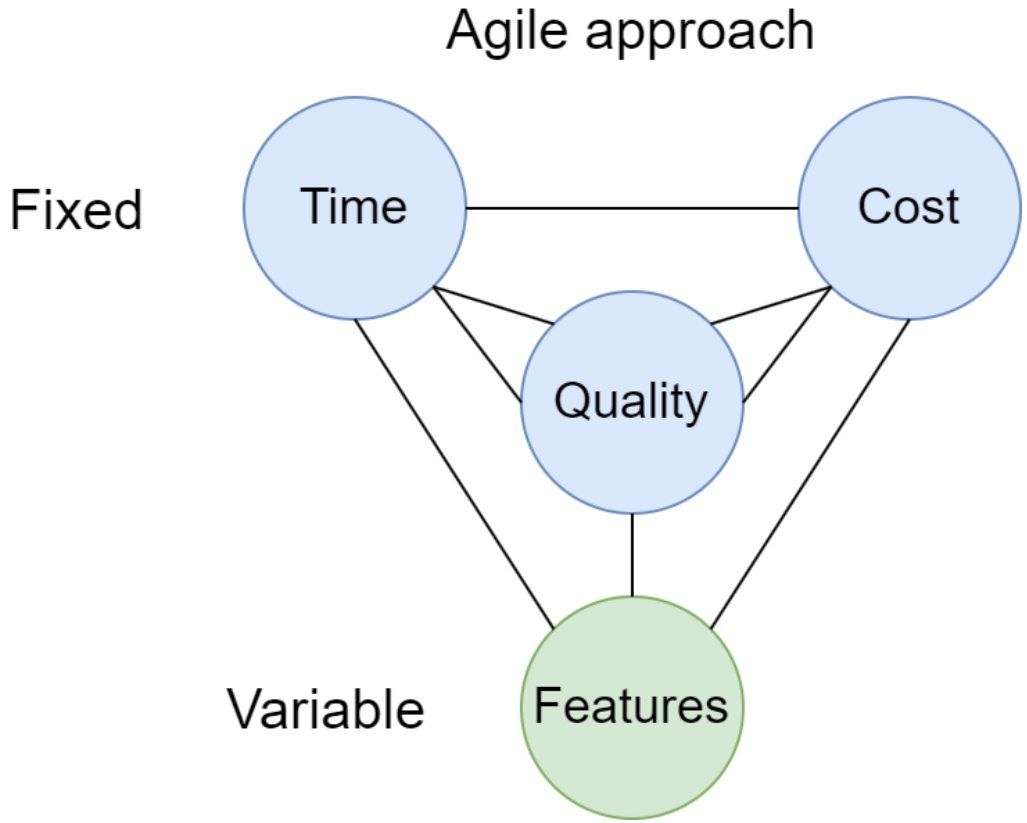
- Cerințe relevante din **perspectivă operațională**
  - Performanță, Securitate, Disponibilitate, Siguranța în funcționare
- Cerințe relevante din **perspectiva dezvoltării**
  - Extensibilitate, Scalabilitate, Testabilitate, Integrabilitate, Observabilitate, Controlabilitate



Modelarea specificațiilor

Gestionarea produsului software

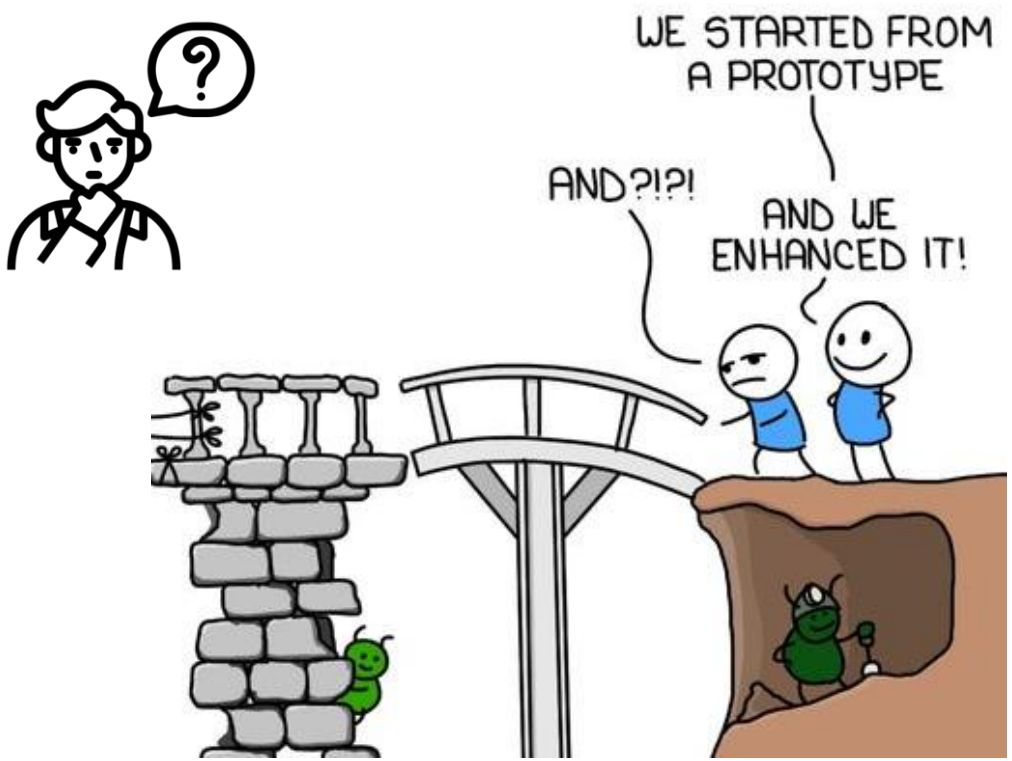
- Product management – cum ne raportăm la metodologia de dezvoltare a proiectului?
- Timp x Cost = Funcționalități x Calitate
- Agile: timp x cost bine delimitate (sprint-uri), funcționalități livrate incremental x calitate
- Waterfall: funcționalități definite de la început, gestionare timp, cost, calitate



Modelarea specificațiilor

Gestionarea produsului software

- De ce sunt atât de importante cerințele dacă tot le ajustăm pe parcurs? (Agile)
- Cât de mult timp trebuie să alocăm modelării specificațiilor?
- Chiar e necesar să avem toate specificațiile clare înainte de începerea dezvoltării produsului?

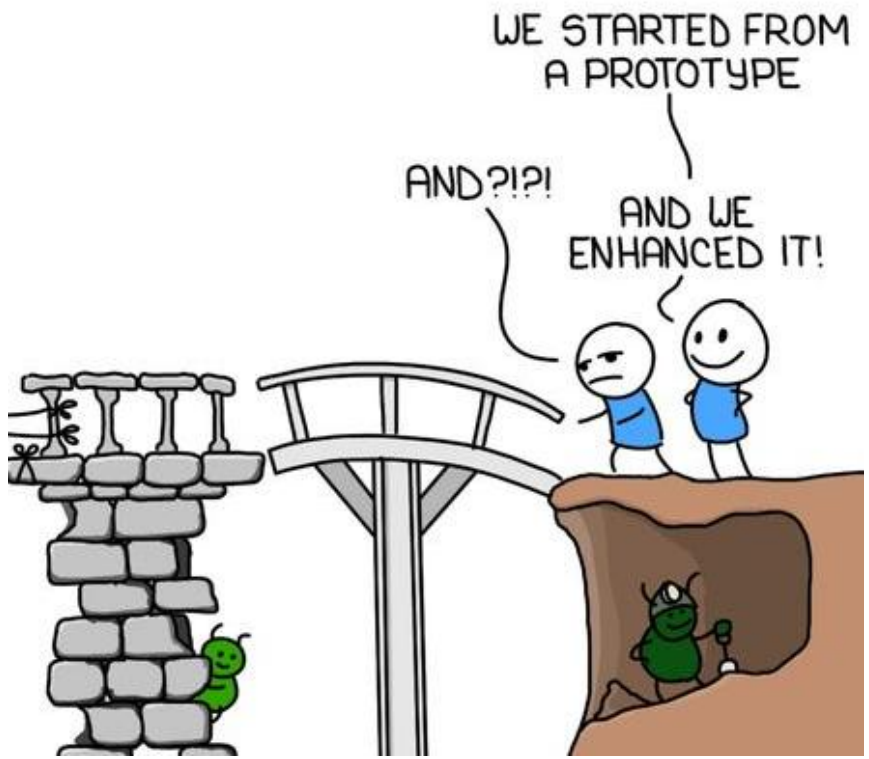


N. Alves et al, Towards an Ontology of Terms on Technical Debt, 2014, IEEE

Modelarea specificațiilor

Gestionarea produsului software

- Riscuri pe termen lung – technical debt
  - Arhitectură (necorespunzătoare)
  - Build (dependențe, procese ineficiente)
  - Cod (code style)
  - Bug-uri (neadresate)
  - Design (necorespunzător – d. patterns)
  - Documentație (incompletă / inexistentă)
  - Infrastructură (subdimensionată)
  - ...

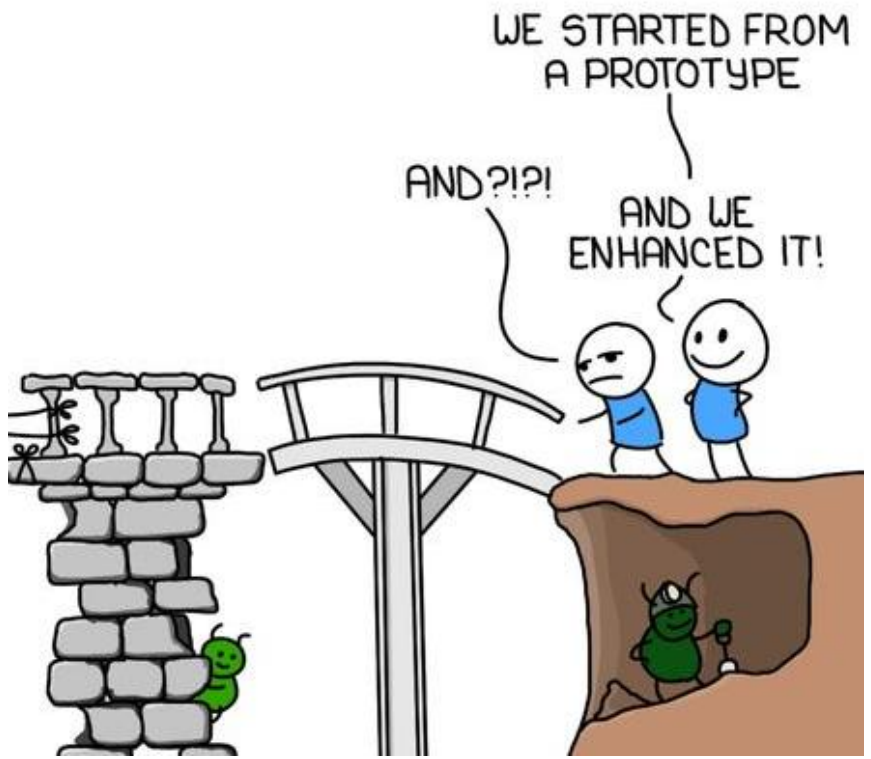


N. Alves et al, Towards an Ontology of Terms on Technical Debt, 2014, IEEE

Modelarea specificațiilor

Gestionarea produsului software

- Riscuri pe termen lung – technical debt
  - ...
  - Resurse umane (insuficiente)
  - Procese (neactualizate)
  - Cerințe (compromisuri)
  - Servicii (substituite)
  - Testare automată (CI/CD)
  - Testare (code coverage)



N. Alves et al, Towards an Ontology of Terms on Technical Debt, 2014, IEEE

Modelarea funcțională (R)

- Un **model funcțional** – o reprezentare structurată a funcțiilor (activități, acțiuni, procese, operații) în cadrul unui sistem
- Cuprinde modele comportamentale din UML
  - Exemplu: diagrama **cazurilor de utilizare** (use case)

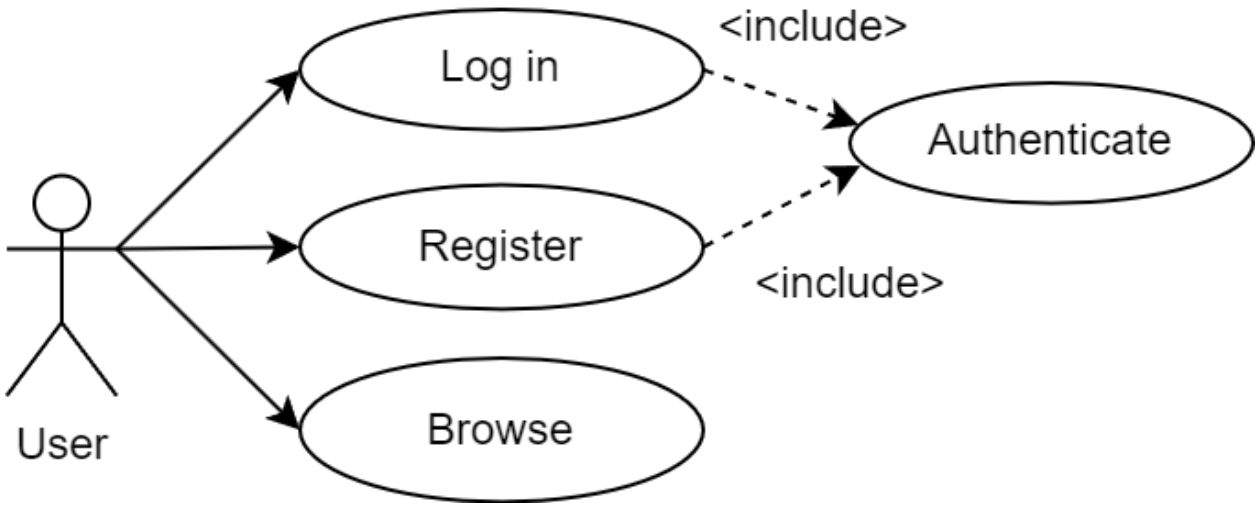


diagrama cazurilor de utilizare (use case)

Studiu de caz: Programarea funcțională

- Programare declarativă bazată pe logică matematică (algebră lambda / lambda calculus)
- Suportată nativ de limbajele de programare: Haskell, JavaScript, Python, Scala, Erlang, Lisp, Clojure
- Concepte de bază
  - Funcții pure (depind doar de intrări, nu au efecte secundare)
  - Recursivitate (nu există instrucțiuni repetitive)
  - Funcții tratate ca orice altă variabilă
  - Variabile imutabile
- Aplicații
  - Calcule matematice, transformări
  - Modelarea concurenței sau a paralelismului

Studiu de caz: Programarea funcțională

Problemă

Cod JavaScript de adunare a numerelor pare dintr-o listă, înmulțite cu 10.

Programare imperativă

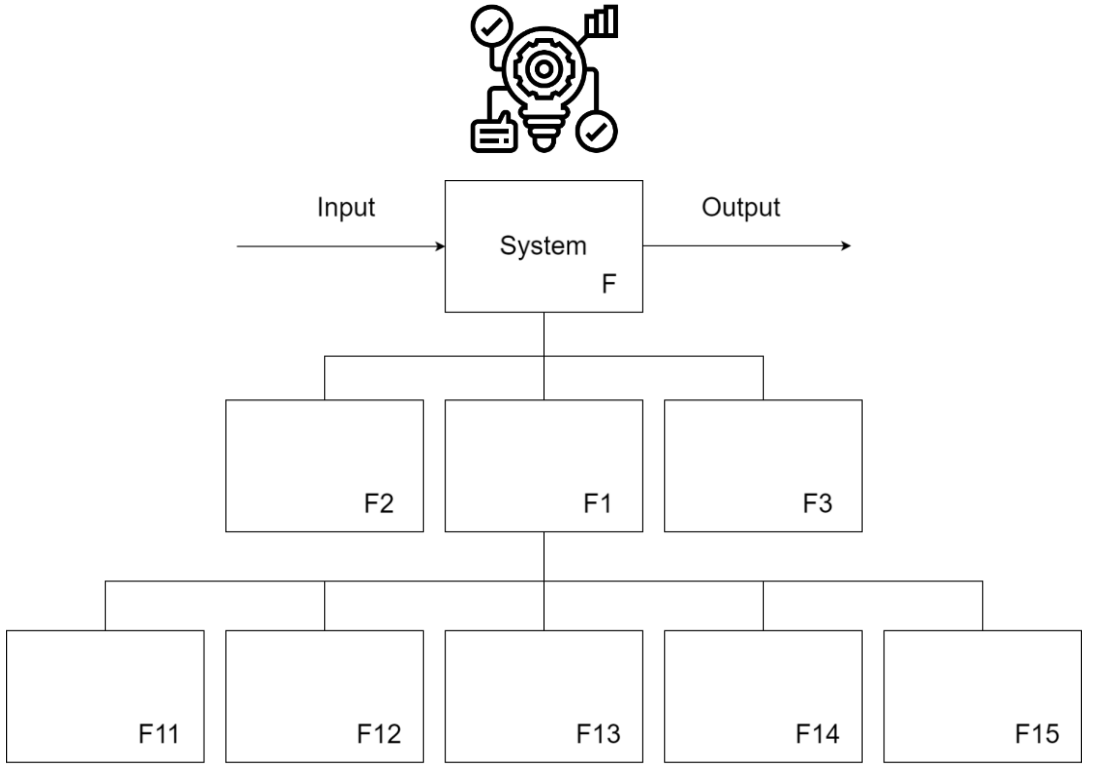
```
const numList = [1,2,3,4,5,6,7,8,9,10];
let result = 0;
for (let i = 0; i < numList.length; i++) {
  if (numList[i] % 2 === 0) {
    result += numList[i] * 10;
  }
}
```

Programare funcțională

```
const result = [1,2,3,4,5,6,7,8,9,10]
               .filter(n => n % 2 === 0)
               .map(a => a * 10)
               .reduce((a, b) => a + b, 0);
```

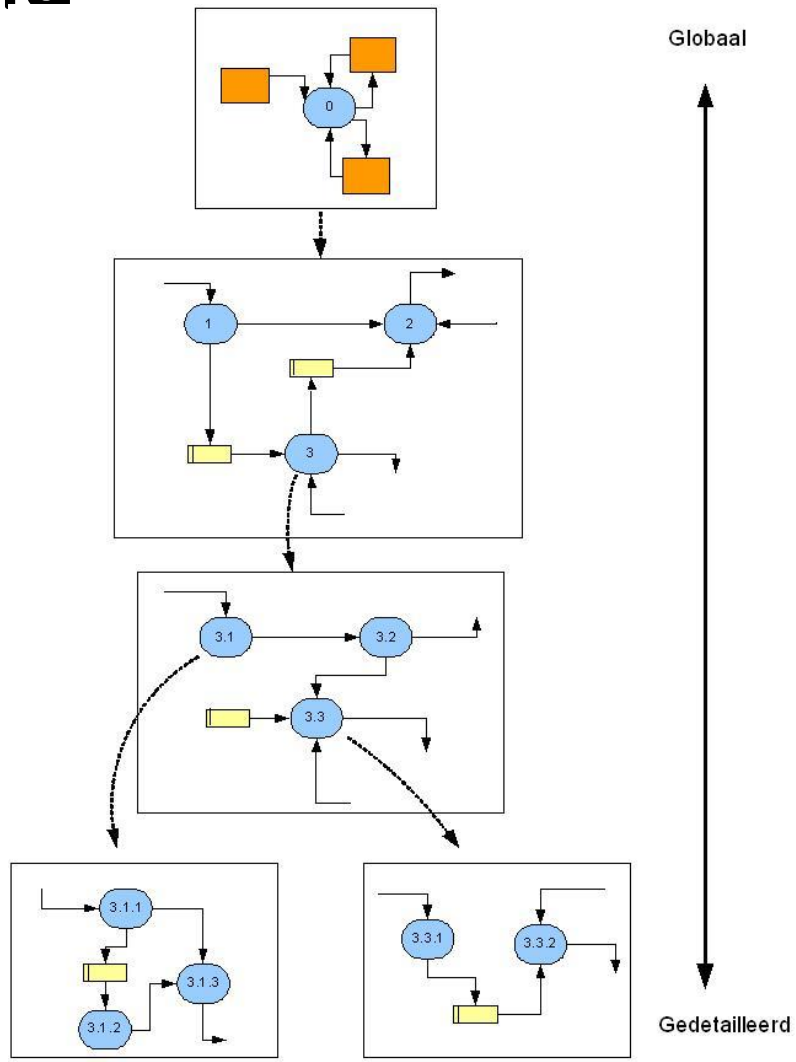
Modelarea funcțională

- Vizează
  - **funcționalitățile** pe care trebuie să le îndeplinească un sistem
  - date de **intrare**
  - date de **ieșire**
- Granularitate
  - funcționalități primare
  - funcționalități secundare
  - funcționalități elementare



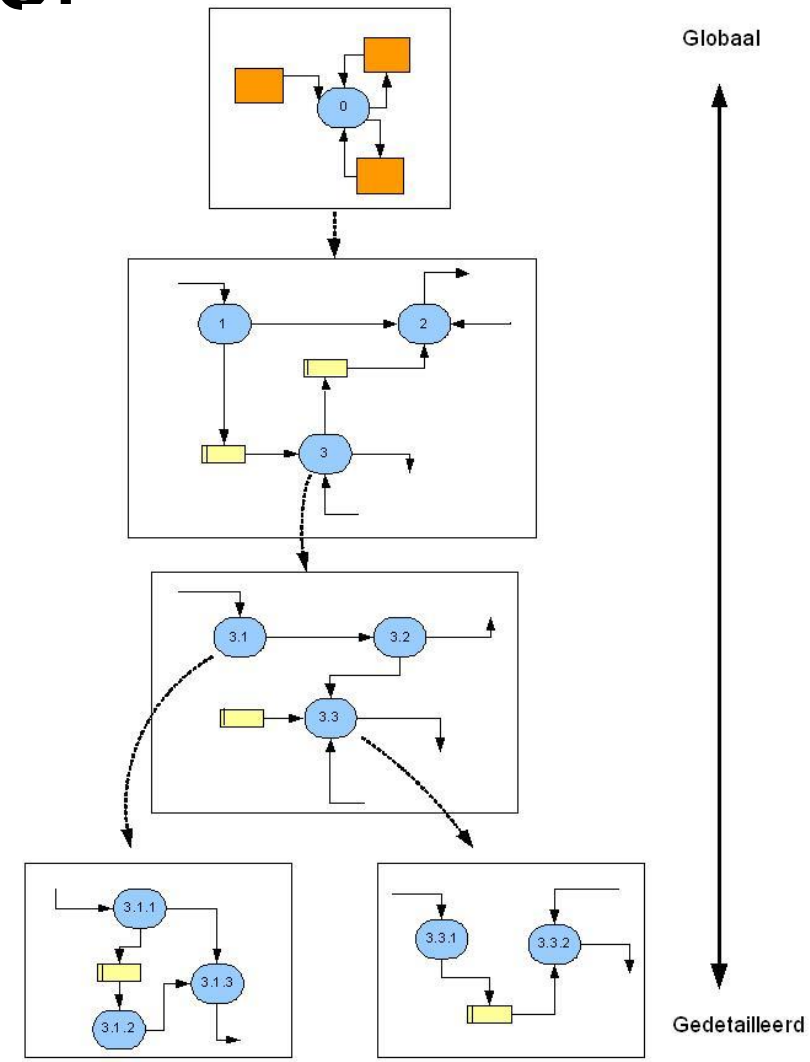
Modelarea funcțională

- Modelarea **activităților**
  - reprezintă succesiunea pașilor printr-un graf parțial ordonat
  - fiecare task este împărțit în subtask-uri interdependente
- Modelarea **proceselor**
  - formă specială de modelare a activităților
  - se axează pe interacțiuni – input-urile si output-urile părților componente ale sistemului



Modelarea activităților

- Modelarea **activităților**
- Perspective
  - Administrativă / de proiect (coordonarea dezvoltării proiectului)
  - Tehnică / de sistem (funcționarea sistemului proiectat)



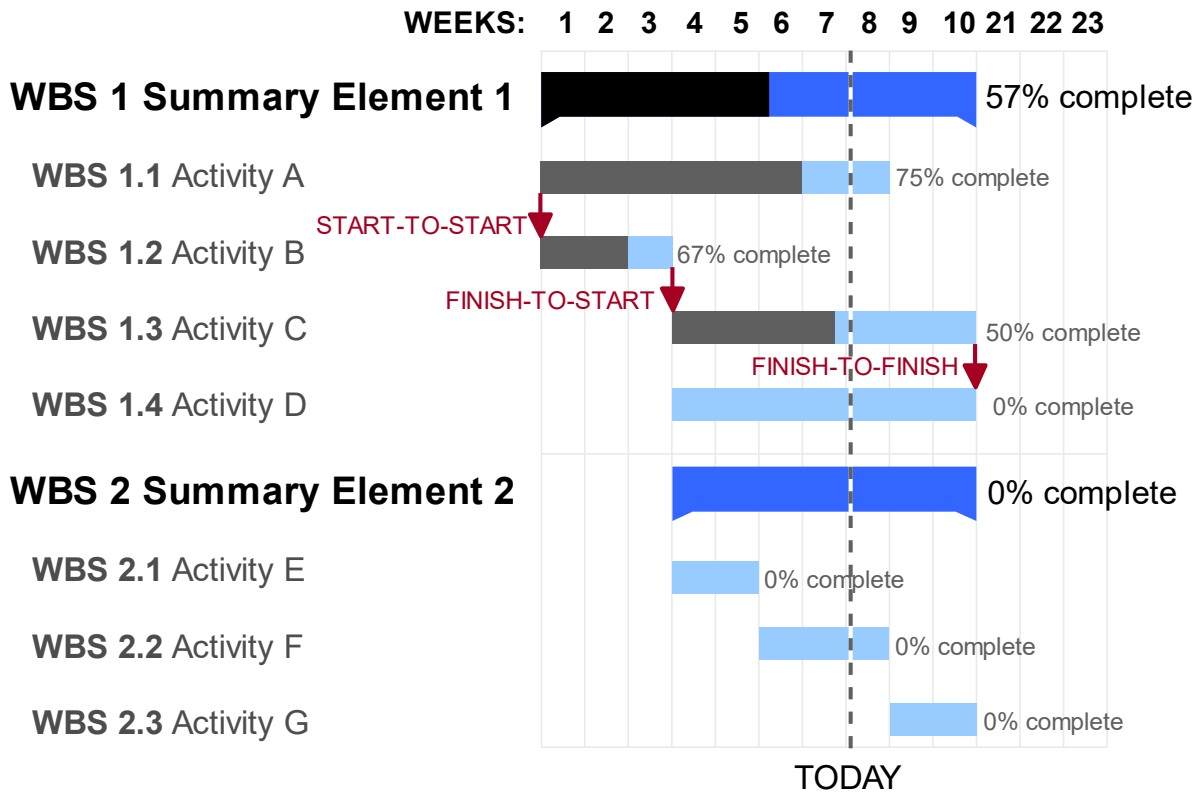
Modelarea activităților de proiect
WBS – Work Breakdown Structure

WBS – metodă de structurare a proiectelor mari în componente mai mici orientate pe livrabile

- Descompunere ierarhică

Exemple

- Spreadsheet
- Diagramă de tip flowchart
- Listă de activități
- Diagramă Gantt

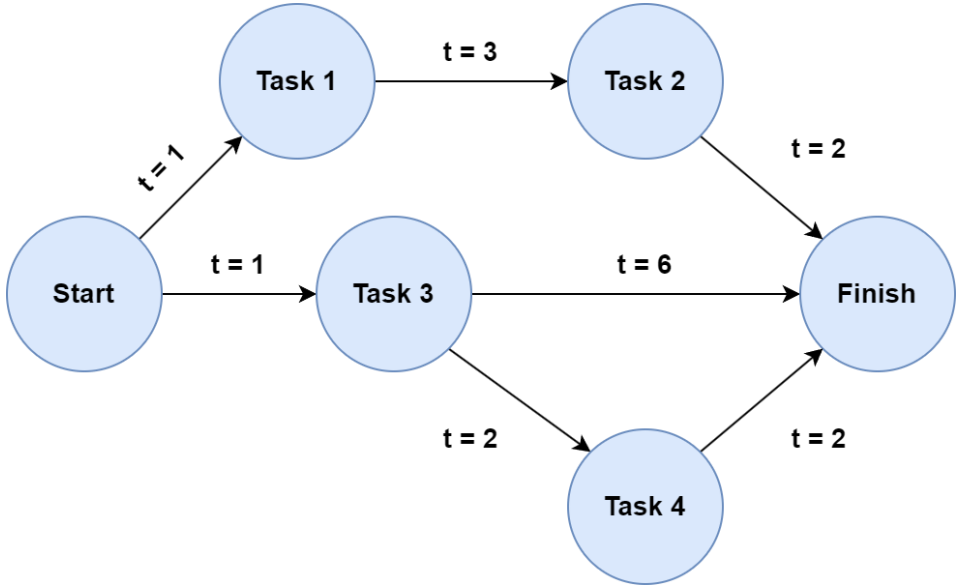


Exemplu: diagramă GANTT

Modelarea activităților de proiect
PERT – Program Evaluation and Review Technique

PERT – metodă de **structurare** a proiectelor (foarte) mari în activități mai mici astfel încât pentru fiecare dintre acestea să se poată **estima** rezultatele și resursele folosite

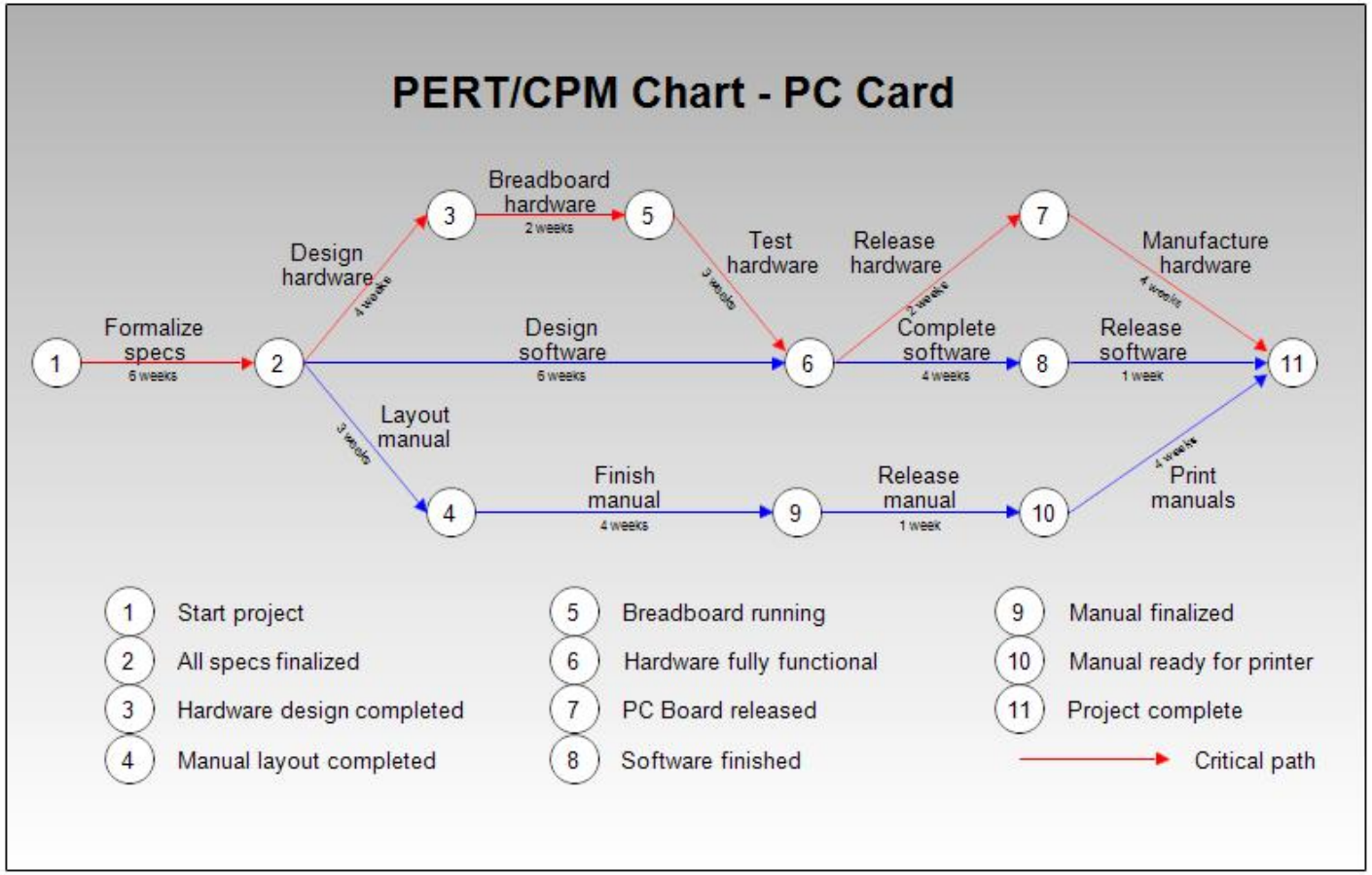
- Graf orientat
- Rădăcina = rezultatul (output-ul) final al activității
- Arcele = obiecte care pot fi input-ul sau output-ul unui subtask



Modelarea activităților de proiect
CPM – Critical Path Method

CPM – metodă de planificare a activităților din cadrul proiectului

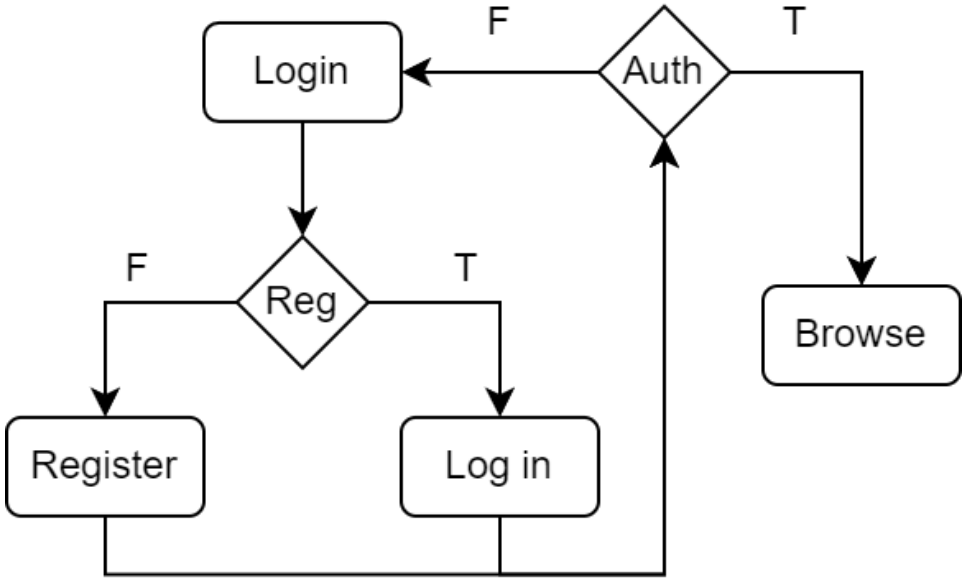
- Are la bază PERT
- Cea mai lungă secvență de activități succesive



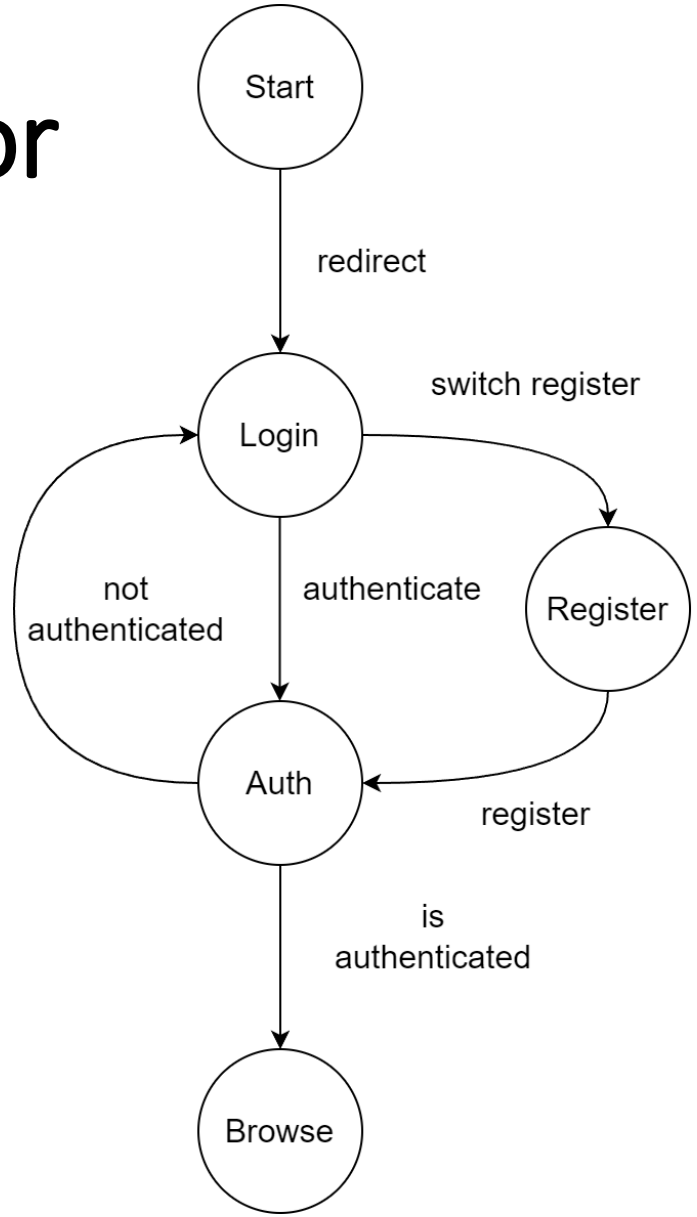
Modelarea activităților

- **Modelul de execuție** reflectă modificările sistemului datorate unor acțiuni (activități elementare)
  - Diagrame de secvență
  - Diagrame de interacțiune
  - Diagrame de activități
- **Modelul de stare** reflectă comportamentul sistemului ca răspuns al activității acestuia într-un context dat
  - Diagrame de stare (ASM)

Modelarea activităților



diagramă de activități (workflow)

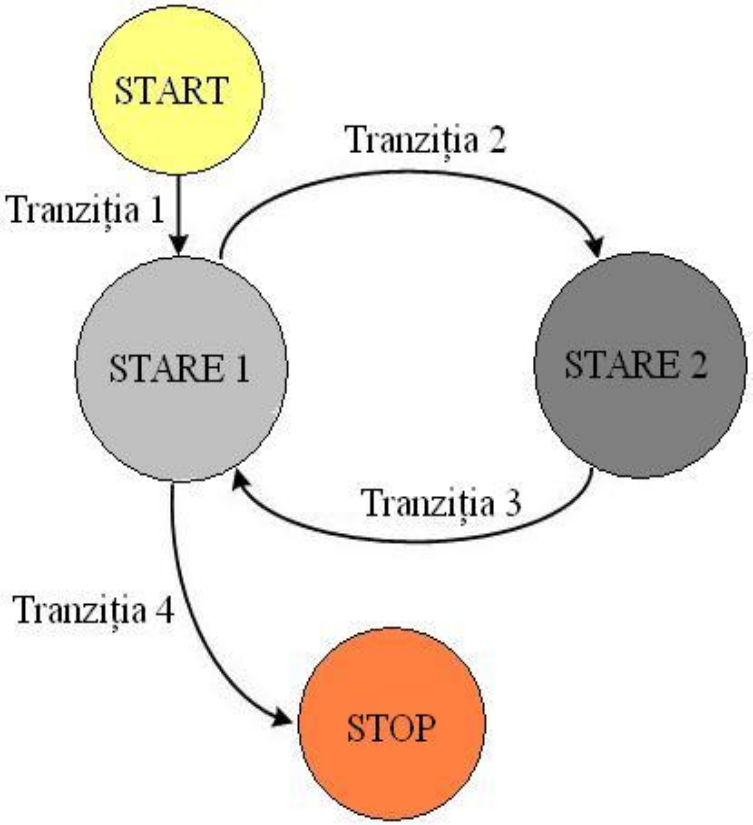


diagramă de stare

Modelarea activităților – Automate cu stări finite

Modelele ASM sunt deosebit de utile pentru a descrie formal răspunsul unui sistem de situații și evenimente imprevizibile / situațiile în care **secvența** corectă a acțiunilor nu poate fi descrisă cu ușurință

- Perspective
  - **Tehnică / de sistem** (funcționarea sistemului proiectat)
  - Administrativă / de proiect (coordonarea dezvoltării proiectului)

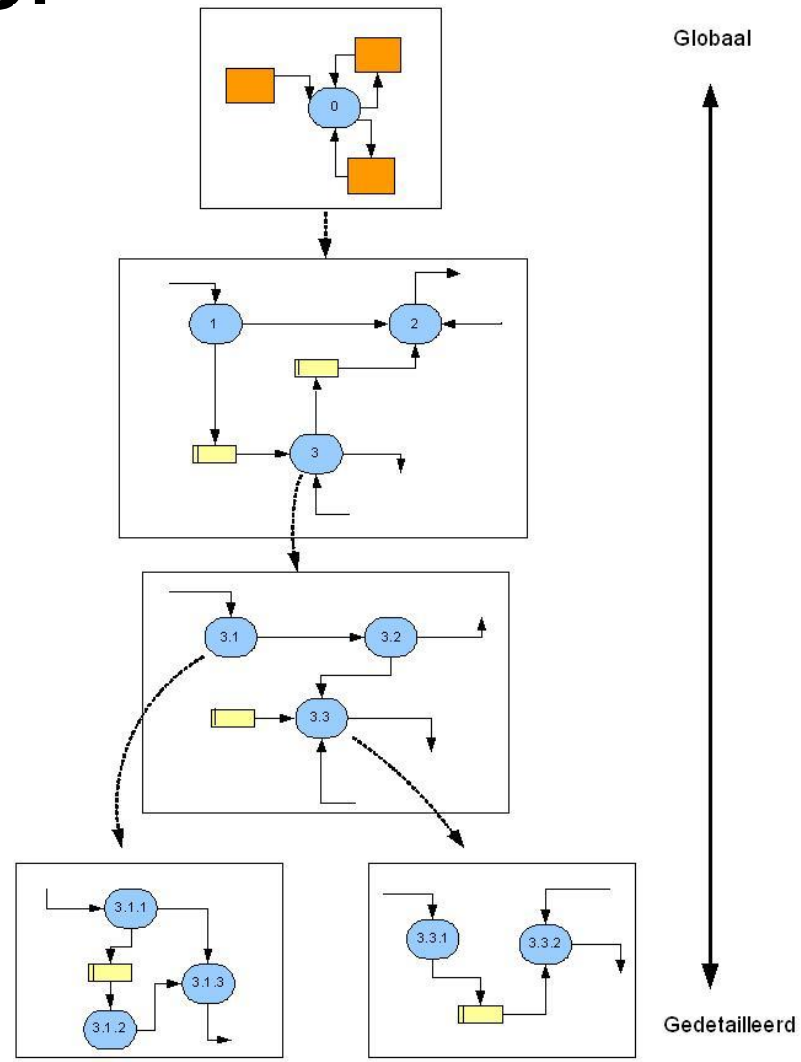


Modelarea proceselor

- **Modelul de proces** reprezintă descompunerea unei funcții (graf temporal de activități ale unei componente)
  - descrie **modul** în care un sistem îndeplinește o funcție
- Exemple
  - BPMN (Business Process Model and Notation)
  - IDEF0 – Integration Definition for Function Modeling
  - Rețele Petri

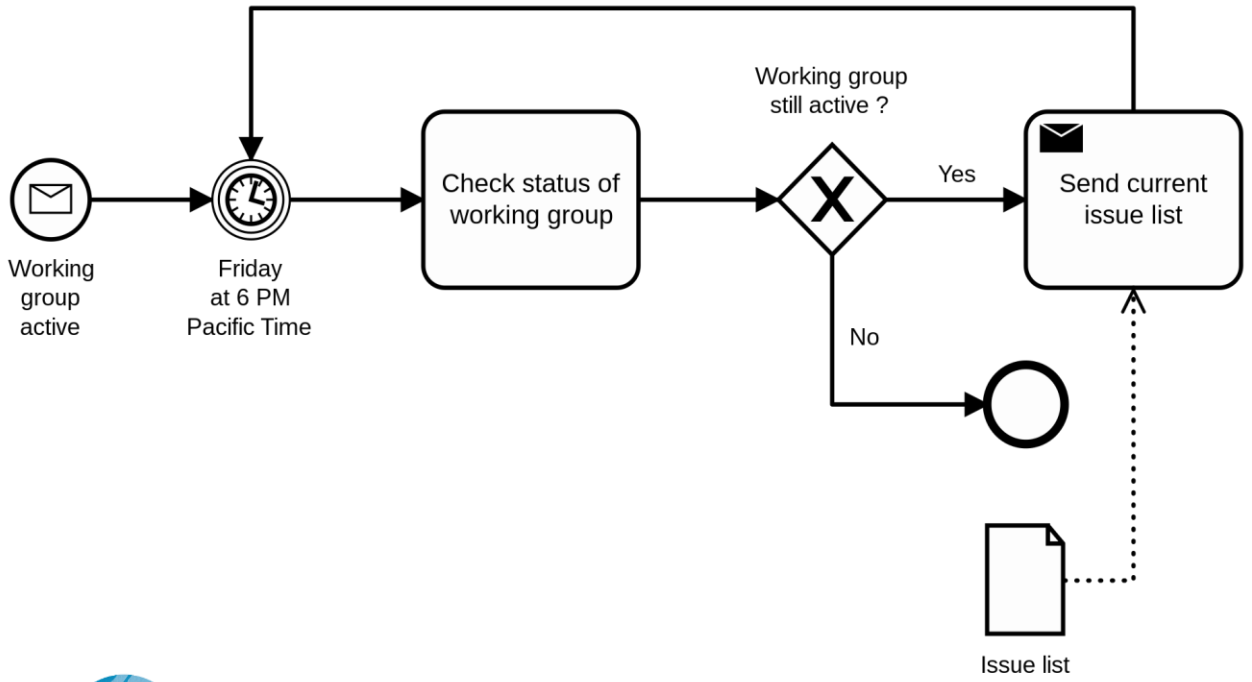
Modelarea proceselor

- Modelarea **proceselor**
- Perspective
  - **Tehnică / de sistem** (funcționarea sistemului proiectat)
  - Administrativă / de proiect (coordonarea dezvoltării proiectului)



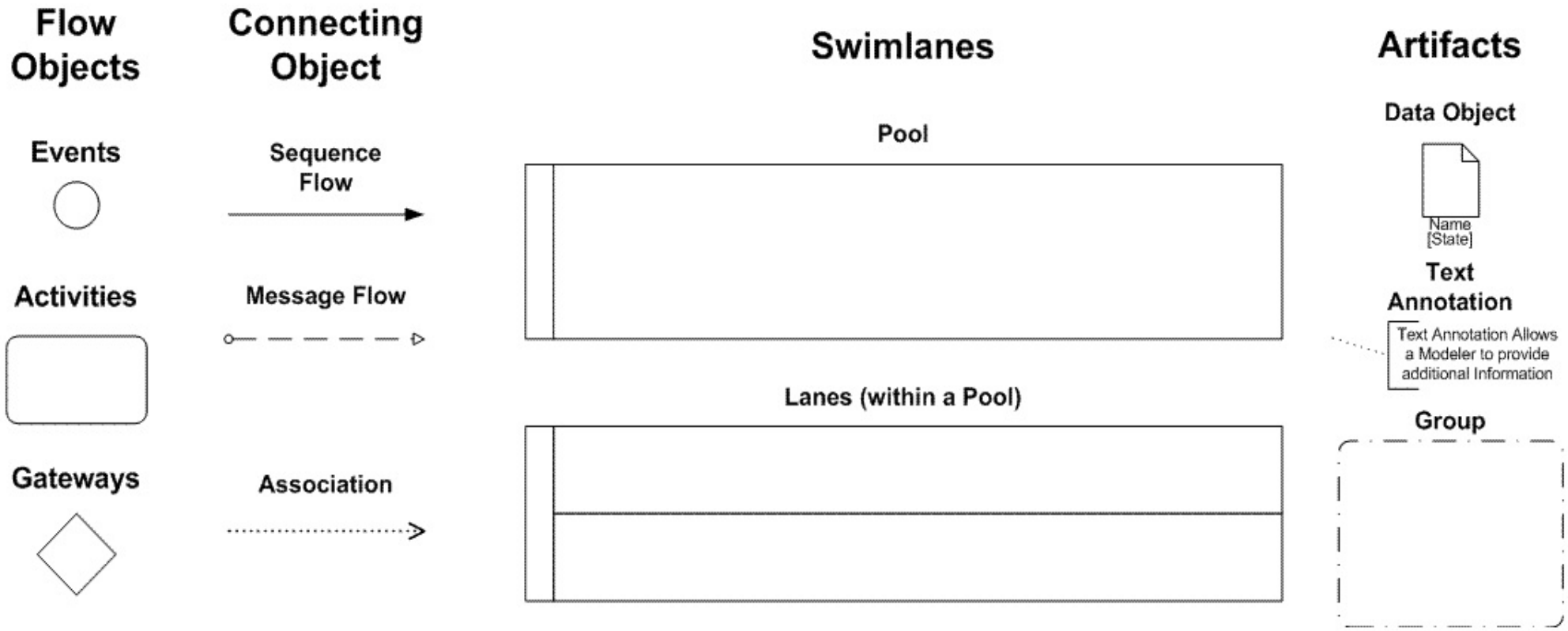
Modelarea proceselor – BPMN

- BPMN (Business Process Model and Notation) este un limbaj grafic de definire a proceselor / workflows
  - Reprezentare grafică
  - Mapare pe cerințele de business
  - Similar diagramei de activități (UML)



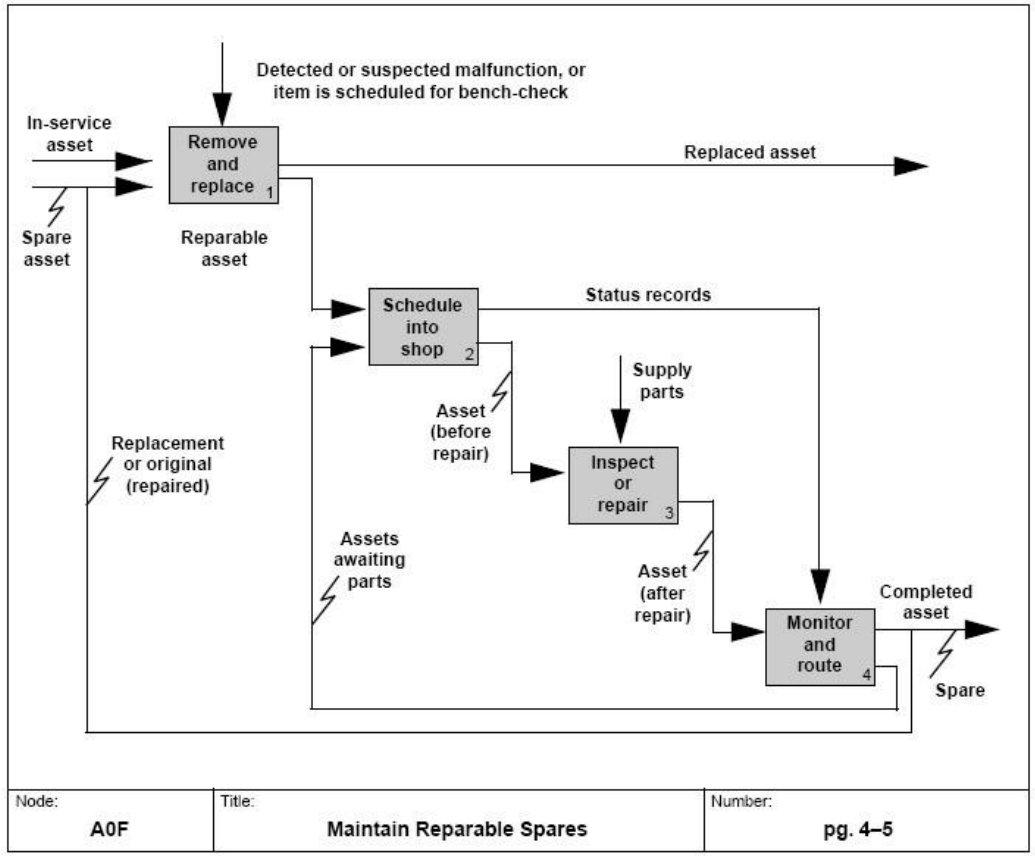
Sursa: Mikelo Skarabo - BPMN - A Process with normal flow

Modelarea proceselor – BPMN



Modelarea proceselor
IDEF0 – Integration Definition for Function Modeling

- IDEF0 – formă de modelare a activităților cu accent pe **input și output**
- Distingem trei tipuri de obiecte și informații într-o activitate:
  - de intrare
  - de control
  - resurse
- Permite descompunerea ierarhică

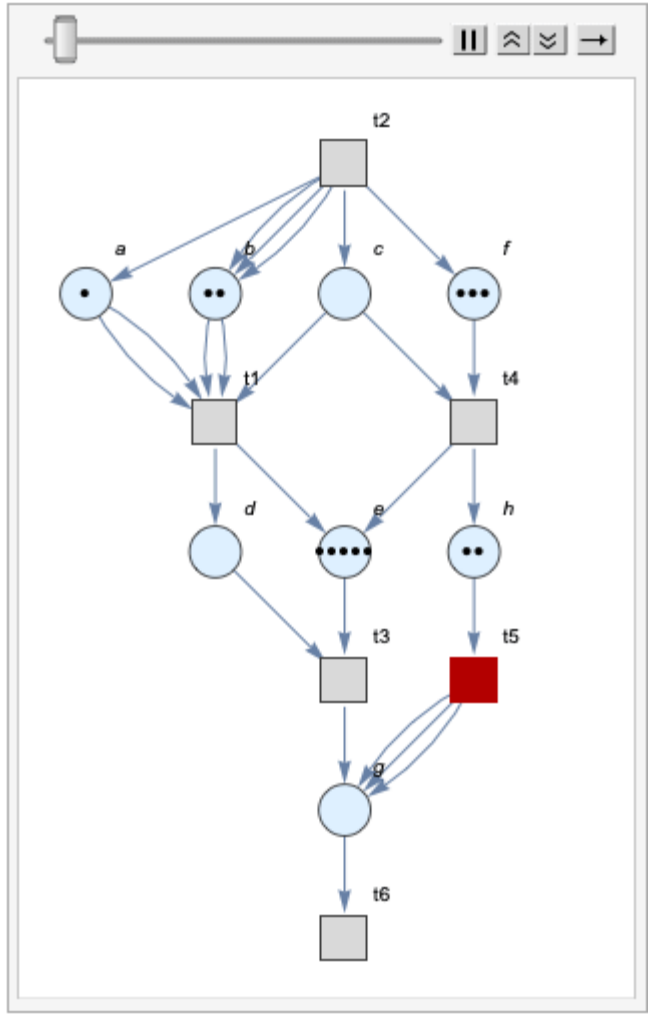


Modelarea proceselor – Rețele Petri

O metodă formală (matematică) folosită pentru modelarea și verificarea **sistemelor** (concurente/distribuite) **dinamice cu evenimente discrete**

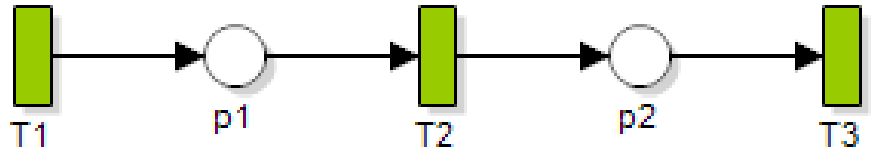
Sistemele

- alcătuite din componente care interacționează
- îndeplinesc o anumită funcționalitate
- evenimente și stări
- concurență, comunicare, sincronizare



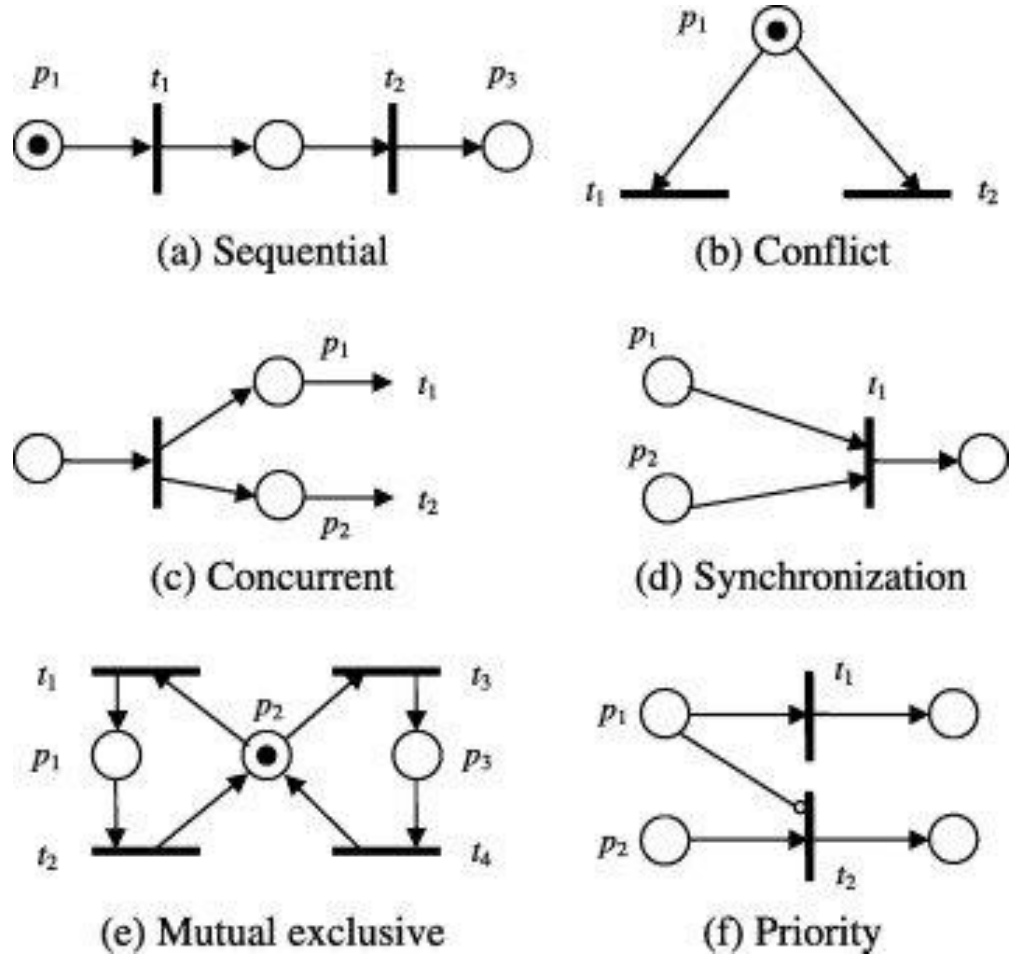
Modelarea proceselor – Rețele Petri

- **Rețele Petri – grafuri bipartite** proiectate special pentru a modela sisteme cu componente cu interacțiune concurentă
- au la bază studiile lui Carl Adam Petri (teza de doctorat)
- A.W. Holt extinde proiectul pentru teoria sistemelor informaționale (Information System Theory Project)
- 1970 Jack B. Dennis, Rețelele Petri și metodele aferente, Institutul de Tehnologie din Massachusetts (M.I.T.)



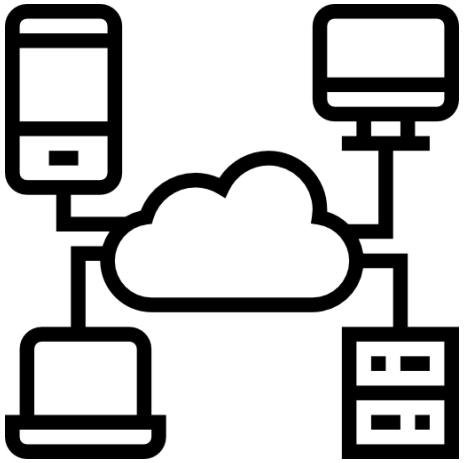
Modelarea proceselor – Rețele Petri

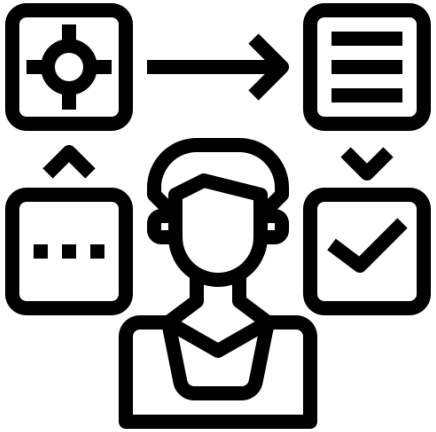
- Avantaje / utilizare
  - reprezentare grafică **intuitivă** semantică formală a stărilor și evenimentelor dintr-un sistem
  - **expresivitate** (concurență, nedeterminism, comunicare, sincronizare)
  - existența metodelor de **analiză** a proprietăților
  - există numeroase **instrumente** software pentru editarea/verificarea proprietăților rețelelor Petri



Modelarea proceselor – Rețele Petri

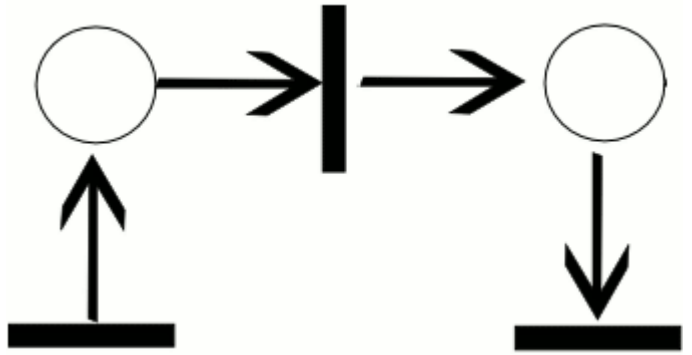
- Domenii de aplicabilitate
  - Protocoale de comunicare, rețele
  - Sisteme software și hardware
  - Algoritmi distribuiți
  - Protocoale de securitate
  - Biologie, Chimie, Medicină
  - Economie (fluxuri de lucru)



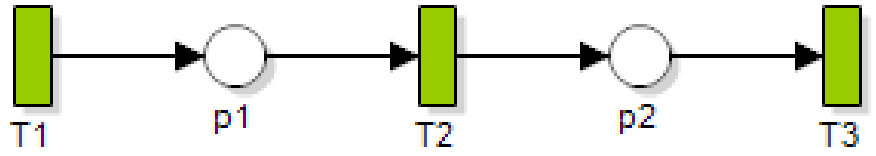


Modelarea proceselor – Rețele Petri

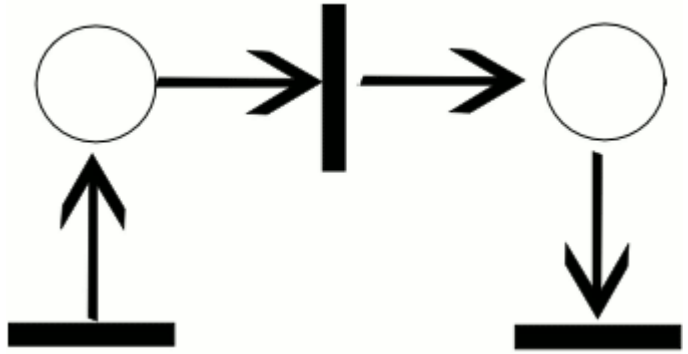
- O rețea Petri este compusă din:
  - o mulțime de poziții S
  - o mulțime de tranziții T
  - o funcție de intrare I
  - o funcție de ieșire O



- Funcțiile de intrare și ieșire sunt relații între T (mulțimea de tranziții) și S (mulțimea de poziții)
- Structura unei rețele Petri este definită de **pozițiile** și **tranzițiile** sale, de funcția sa de intrare și de cea de ieșire



Modelarea proceselor – Rețele Petri

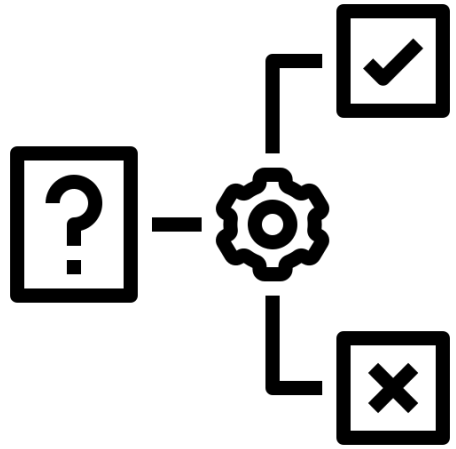


Structura unei rețele Petri

- **Tranziții** = acțiuni sau evenimente din sistemul modelat
- **Punctele** din poziții = resurse/valori booleene
- **Pozițiile** de intrare conțin resurse (reprezentate de punctele din poziție) care vor fi folosite de către acțiune, precondiții pentru producerea unui eveniment
- Ponderea unui **arc input** = câte resurse de un anumit tip sunt necesare producerii acțiunii
- Ponderea unui **arc output** = numărul de resurse de un anumit tip rezultate prin producerea acțiunii

Modelare bazată pe reguli

- Modele ce descriu **funcții și comportamente** ale sistemelor ca un set de secvențe **situație – răspuns**
- Reprezentate prin
  - **Pseudocod**
  - **Tabele de decizie**
  - **Arbori de decizie**
- Condiții
  - Completitudine
  - Consistență



Modelare bazată pe reguli

Exemplu

- O firmă de transport pentru elevi acordă reducere de 50% pentru elevii cu vârsta de până la 12 ani. Elevii cu vârsta de peste 12 ani au reducere de 50% numai pe mijloacele de transport fără regim de rezervare a locurilor. Pasagerii care nu sunt elevi nu beneficiază de reducere.

Modelare bazată pe reguli

Pseudocod

if elev
then
  if varsta < 12
  then ***50% reducere***
  else {necesita rezervare}
    if nu necesita rezervare
    then ***50% reducere***
    else ***fara reducere***
else {nu este elev} ***fara reducere***

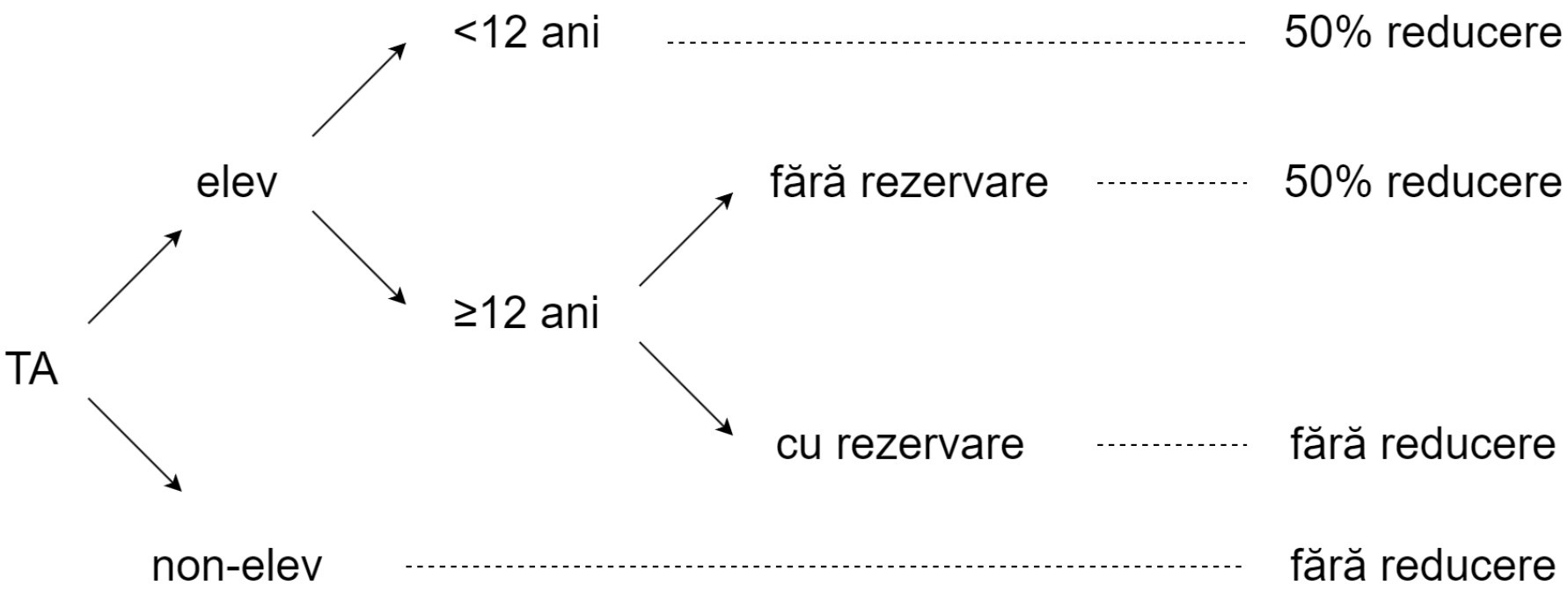
Modelare bazată pe reguli

Tabel de decizie

	Condiții			Acțiuni	
Condiție / Nr. regulă	**Elev?**	**Sub 12 ani?**	**Necesită rezervare?**	**50% reducere**	**Fără reducere**
1	0	0	0		1
2	0	0	1		1
3	0	1	0		1
4	0	1	1		1
5	1	0	0	1	
6	1	0	1		1
7	1	1	0	1	
8	1	1	1	1	

Modelare bazată pe reguli

Arbore de decizie



Modelare bazată pe reguli

Problemă

Se dorește proiectarea unui sistem care detectează știrile false și clasifică postările de pe rețelele sociale, pe baza mai multor factori relevanți:

- Credibilitatea sursei – Verificarea dacă sursa este una credibilă/verificată sau anonimă.
- Analiza conținutului – Identificarea unui limbaj inflamator sau a unor afirmații senzaționale.
- Prezența sau absența dovezilor factuale care susțin conținutul.
- Modelele de interacțiune ale utilizatorilor – Dacă interacțiunile provin din conturi suspecte (ex. roboți) sau din conturi credibile.
- Rezultatele verificării faptelor (fact-checking) – Confirmarea ca falsă sau autentică de către verificatori de fapte.

Modelați sistemul pe bază de reguli.

Întrebări?