



UNIVERSITATEA  
DIN BUCUREŞTI

# Metode de dezvoltare software

## Procese de dezvoltare software

21.02.2023

Alin Ștefănescu

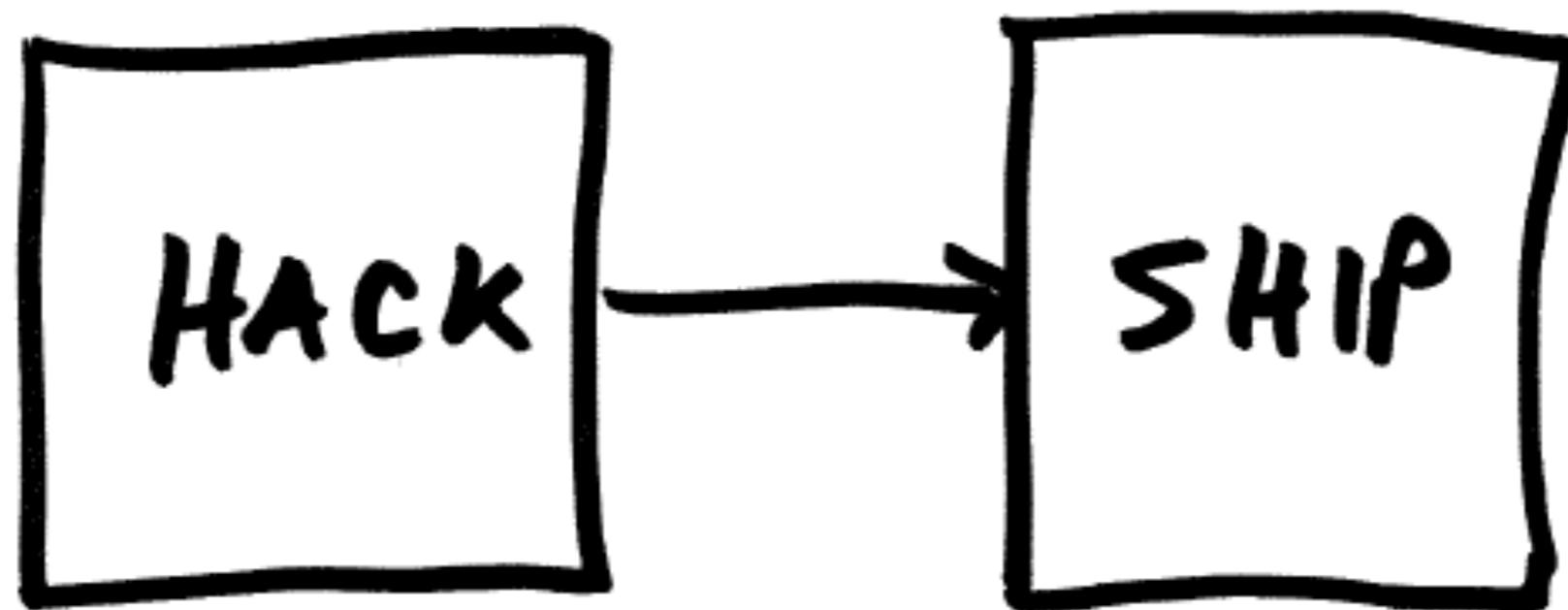




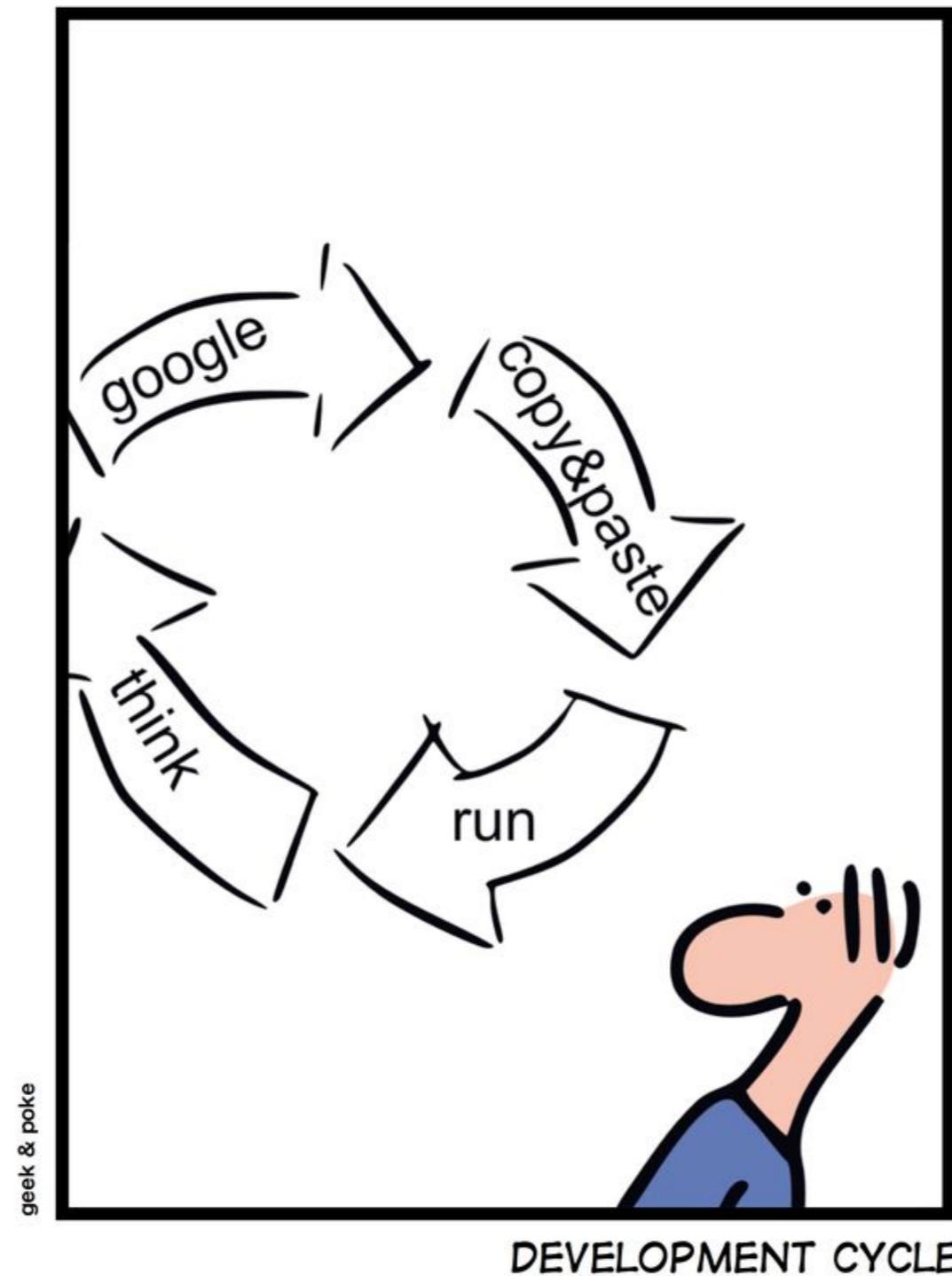
## Procese de dezvoltare software

Prezentare bazată pe materiale de: Florentin Ipate (FMI UniBuc) - note de curs  
și Ian Sommerville: Software Engineering 10th edition

# Procese de dezvoltare software... în practică



# Varianta detaliată



Cerinte

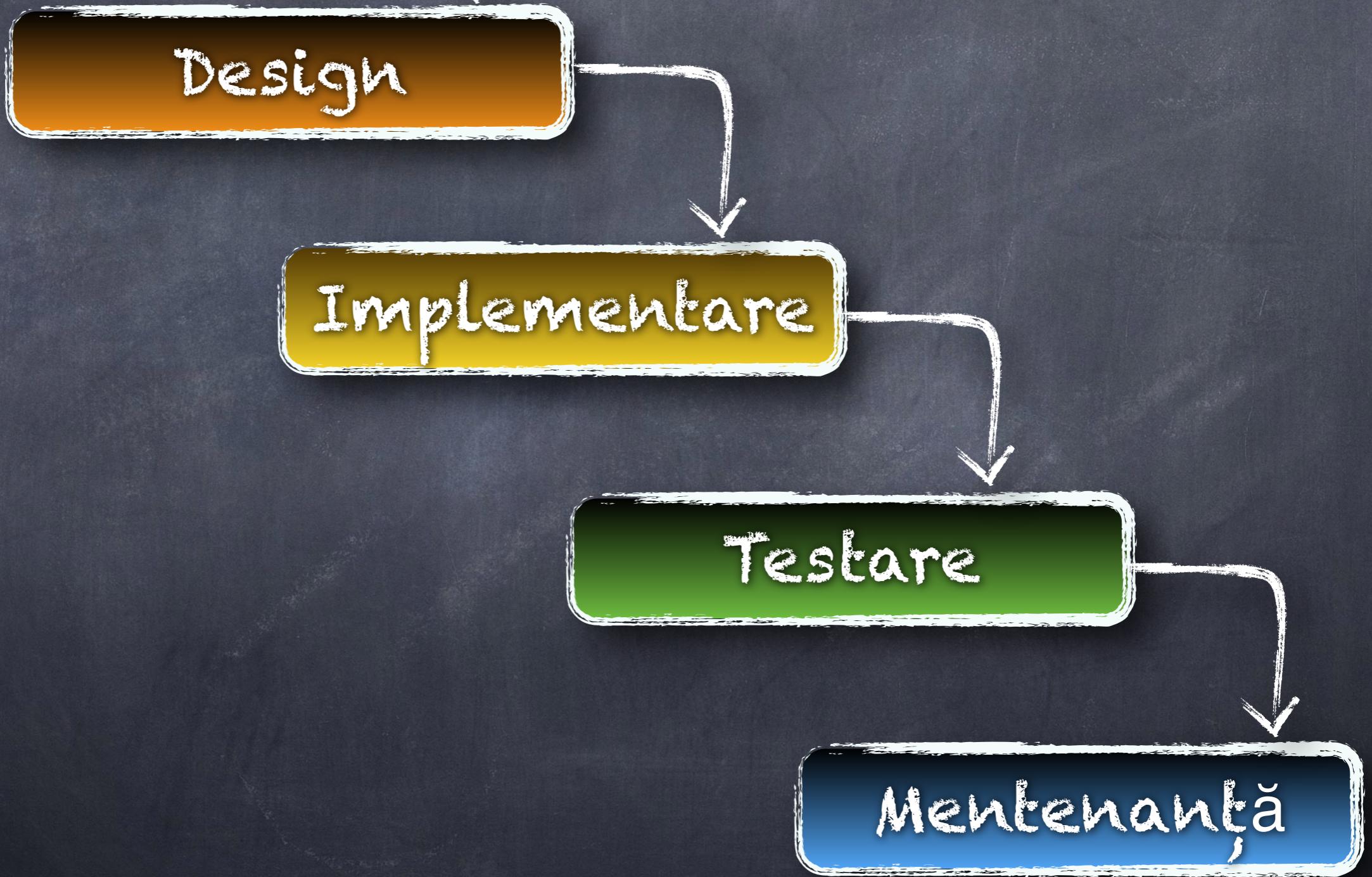
Procesul de dezvoltare  
"cascadă" (waterfall)

Design

Implementare

Testare

Mențenanță



# Etapele procesului “cascadă”

- **analiza și definirea cerințelor:** Sunt stabilite serviciile, constrângerile și scopurile sistemului prin consultare cu utilizatorul. (**ce** trebuie să facă sistemul).
- **design:** Se stabilește o arhitectură de ansamblu și funcțiile sistemului software pornind de la cerințe. (**cum** trebuie să se comporte sistemul).
- **implementare și testare unitară:** Designul sistemului este transformat într-o mulțime de programe (unități de program); testarea unităților de program verifică faptul că fiecare unitate de program este conformă cu specificația.
- **integrare și testare sistem.** Unitățile de program sunt integrate și testate ca un sistem complet; apoi acesta este livrat clientului.
- **operare și mențenanță.** Sistemul este folosit în practică; mențenanța include: corectarea erorilor, îmbunătățirea unor servicii, adăugarea de noi funcționalități.

# Avantaje și dezavantaje

- fiecare etapă nu trebuie să înceapă înainte ca precedenta să fie încheiată
- fiecare fază are ca rezultat unul sau mai multe documente care trebuie “aprobată”
- bazat pe modele de proces folosite pentru producția de hardware

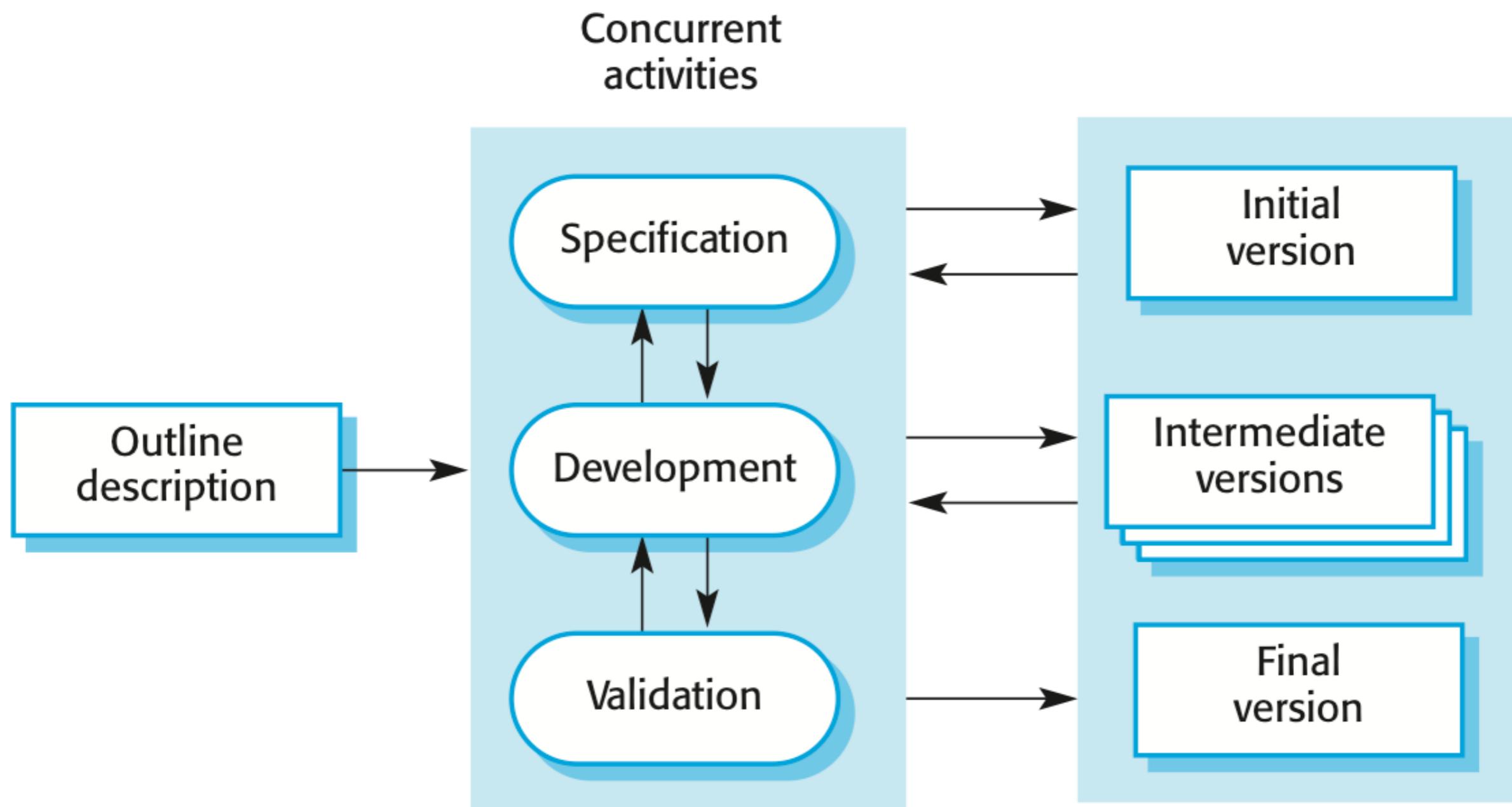
**Avantaj:** proces bine structurat, riguros, clar; produce sisteme robuste

- Probleme:
  - dezvoltarea unui sistem software nu este de obicei un proces liniar; etapele se întrepătrund
  - metoda oferă un punct de vedere **static** asupra cerințelor
  - schimbarile cerințelor nu pot fi luate în considerare după aprobarea specificației
  - nu permite implicarea utilizatorului după aprobarea specificației

**Concluzie:** Modelul cascadă trebuie folosit atunci când cerințele sunt bine înțelese și când este necesar un proces de dezvoltare clar și riguros.

# Procesul incremental

**Idee:** "un sistem de succes de mare dimensiune începe cu un sistem de succes de mică dimensiune care apoi crește puțin câte puțin" (Gilb, 1988)



# Procesul incremental

- sunt identificate cerințele sistemului la nivel înalt, dar, în loc de a dezvolta și livra un sistem dintr-o dată, dezvoltarea și livrarea este realizată în părți (**incremente**), fiecare increment încorporând o parte de funcționalitate.
- cerințele sunt ordonate după **priorități**, astfel încât cele cu prioritatea cea mai mare fac parte din primul increment, etc.
- după ce dezvoltarea unui increment a început, cerințele pentru acel increment sunt înghețate, dar cerințele pentru noile incremente pot fi modificate.

# Avantajele procesului incremental

## Avantaje

- clienții nu trebuie să aștepte până ce întreg sistemul a fost livrat pentru a beneficia de el. Primul increment include cele mai importante cerințe, deci sistemul poate fi folosit imediat.
- primele incremente pot fi prototipuri din care se pot stabili cerințele pentru următoarele incremente.
- se micșorează riscul ca proiectul să fie un eșec, deoarece părțile cele mai importante sunt livrate la început.
- deoarece cerințele cele mai importante fac parte din primele incremente, acestea vor fi testate cel mai mult.

## Probleme

- dificultăți în transformarea cerințelor utilizatorului în incremente de mărime potrivită.
- procesul nu este foarte vizibil pentru utilizator (nu e suficientă documentație între iterații).
- codul se poate degrada în decursul ciclurilor.

# Exemple de procese incrementale

Există multe variante de procese de dezvoltare incrementale:

- procese de dezvoltare agile. Exemple:
  - Scrum
  - programare extremă (XP - extreme programming)

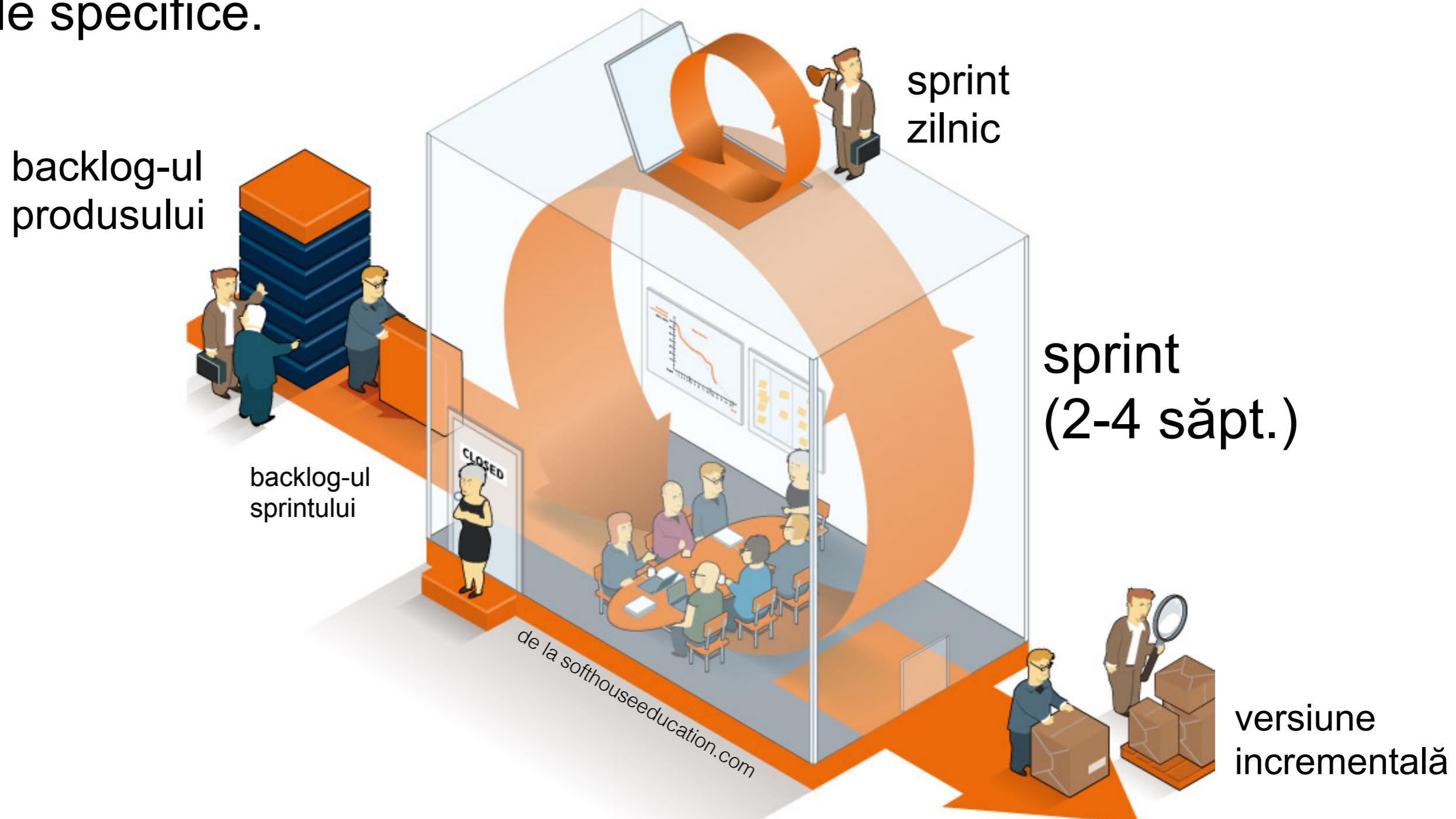


de la n-ix.com

# SCRUM

# Metoda Scrum

Scrum este o metodă “agilă”, care se axează mai mult pe **managementul dezvoltării incrementale**, decât pe practici agile specifice.



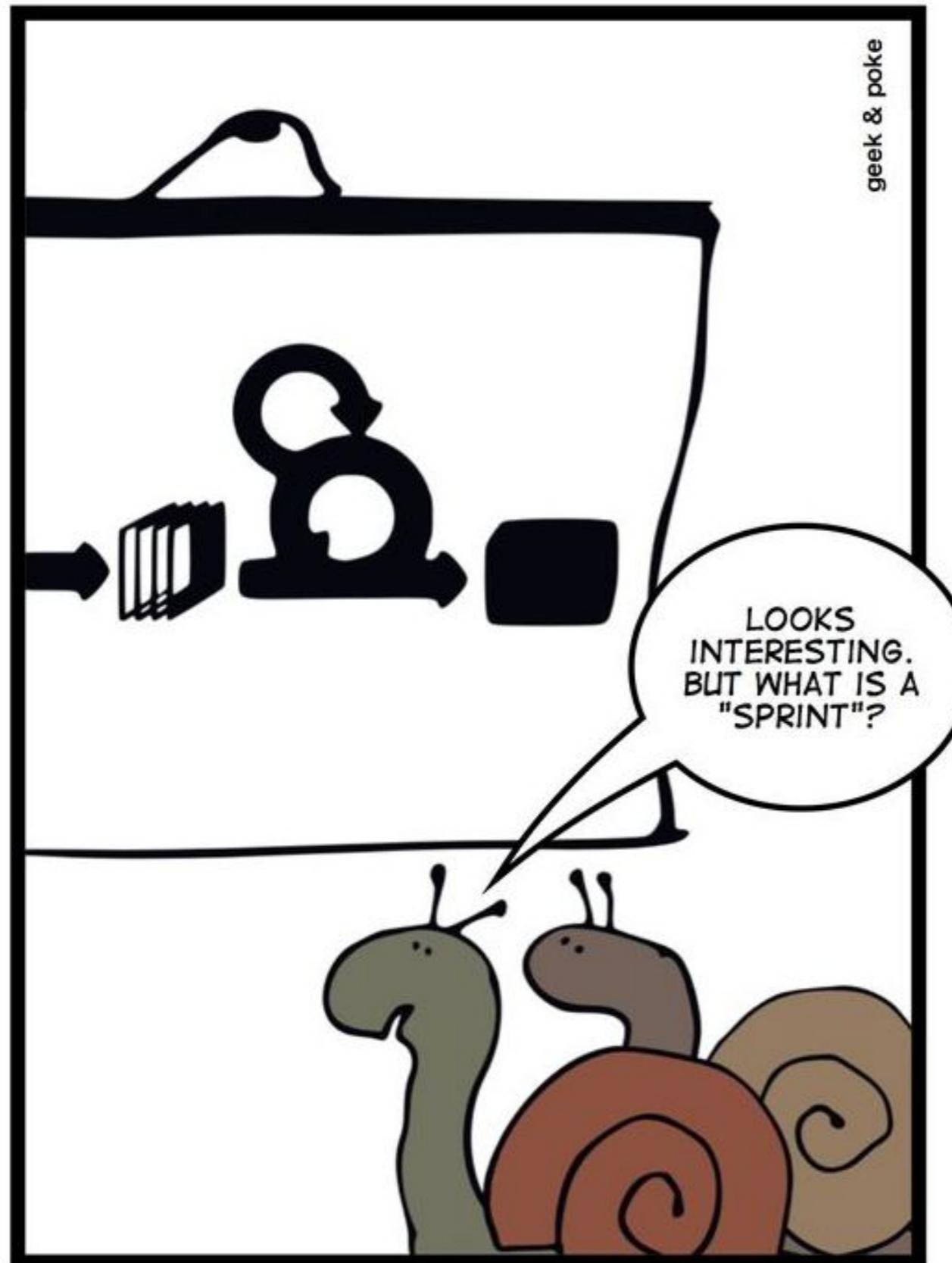
# Scrum pe scurt

- un proprietar de produs creează o listă de sarcini numită “**backlog**”.
- după aceasta, se planifică care dintre sarcini vor fi implementate în următoarea iteratie, numită “**sprint**”.
- această listă de sarcini se numește “**sprint backlog**”.
- sarcinile sunt rezolvate în decursul unui sprint care are rezervată o perioadă relativ scurtă de 2-4 săptămâni.
- echipa se întâlnește zilnic pentru a discuta progresul (“**daily scrum**”). Ceremoniile sunt conduse de un “**scrum master**”.
- la sfârșitului sprintului, rezultatul ar trebui să fie livrabil (adică folosit de client sau vândabil).
- după o analiză a sprintului, se reiterează.



ScrumAlliance®

# Scrum... în practică



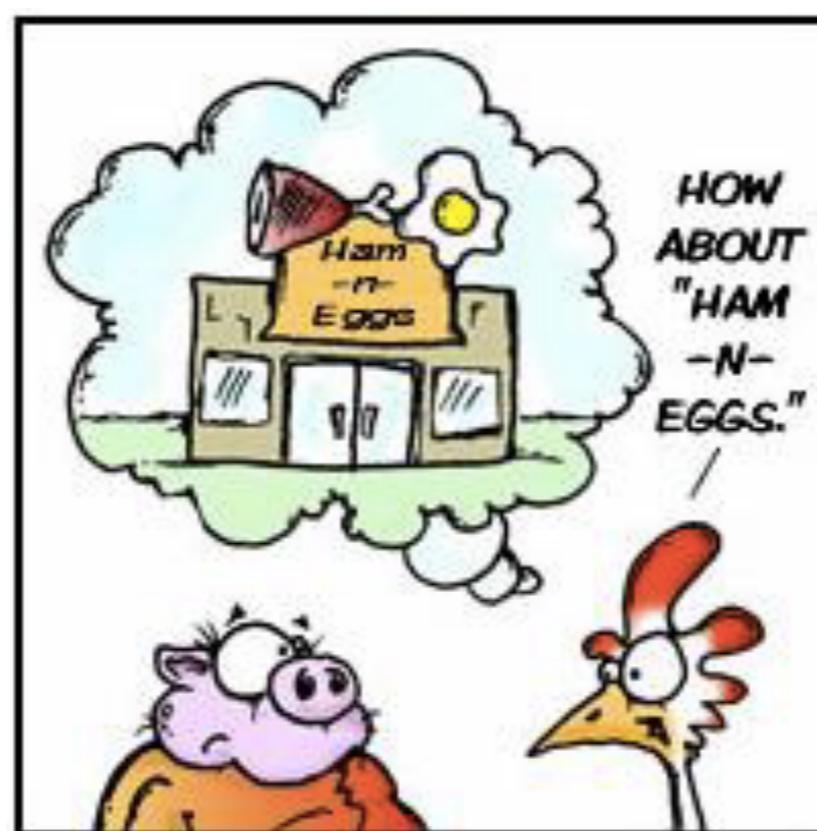
Learning Scrum

# Roluri în SCRUM

- product owner (proprietar de produs) - definește backlogul
- scrum master - are grija de întreg procesul
- membrii echipei
- utilizatori & stakeholders & manageri

pigs

← chickens



# Daily (deadly?)... Scrum

