Integrarea sistemelor informatice

Suport curs nr. 1

Programator >> Arhitect

Introducere

2024-2025

C1 – Introducere

Obiective

- Introducere în domeniul sistemelor informatice integrate
- Identificarea procesului de dezvoltare a proiectelor software
- Înțelegerea rolului integrării în dezvoltarea sistemelor informatice

Dezvoltarea sistemelor informatice

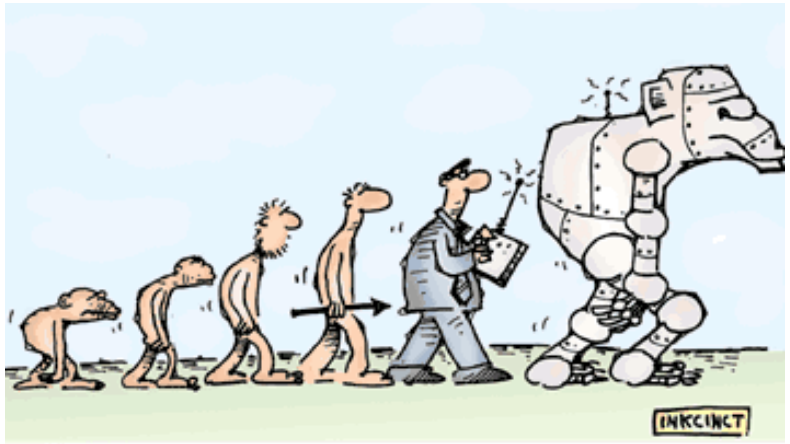
Punctele de bază

- Problema mea – Cerințele
  - Client și Utilizator
  - Cum reușim?
- Echipa mea – Comunicarea
  - Distribuția echipei
  - Cum lucrăm împreună?
- Punctul de plecare – Decizii
  - Ce este deja făcut? – Refolosire
  - Cu ce încep? – Instrumente

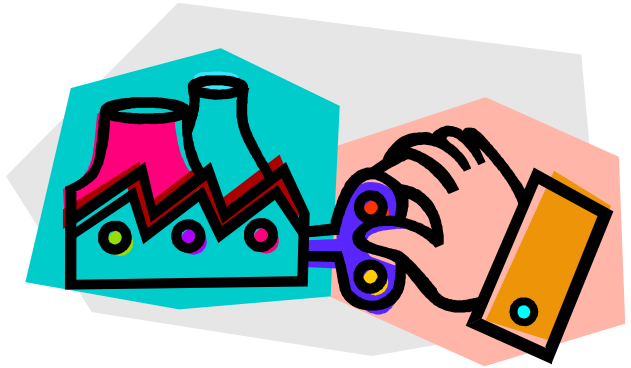
Image by Freepik

Evoluția dezvoltării software-ului

- Problema dezvoltării de software
  - Continua și constanta creștere în volum și complexitate
- Primele abordări de Software Engineering
  - Erau o replică a hardware-ului sau a altor discipline inginerești
- Cheia pentru un software bun …



The ascent of man.



Studiu de caz: legea lui Gall

“A complex system that works is invariably found to have evolved from a simple system that worked”, John Gall

Studiu de caz – Lansarea în 2013 a platformei healthcare.gov pentru tranzacționare asigurări de sănătate (Affordable Care Act)

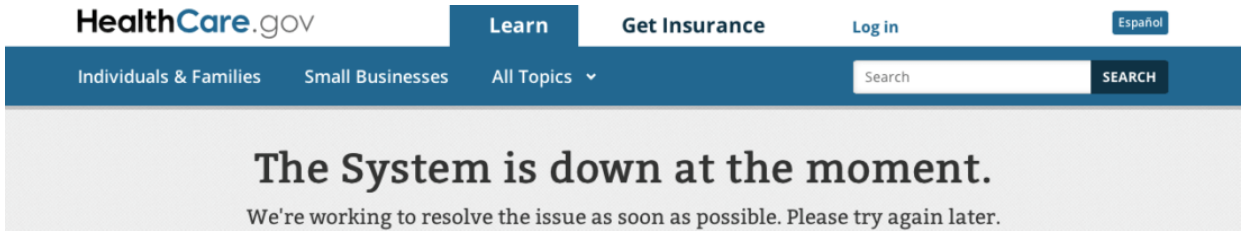
- **Integrări complexe** – comunicare cu baze de date guvernamentale, transmiterea datelor la sute de asiguratori
- Obiectiv general – o platformă complexă care să funcționeze **pentru toată lumea** din momentul lansării (inclusiv pentru multele cazuri excepționale) – complexitate ridicată din start

If you have a complex project, follow “Gall’s law” — or it will fail, Bigthink, 2023

Studiu de caz: legea lui Gall

Rezultat – platforma nu a funcționat pentru niciun utilizator – toți utilizatorii au apelat la call-center pentru soluționarea cererilor

- Cauze – Au existat mulți manageri de proiect dar niciun manager de produs (ar fi susținut că un astfel de produs este imposibil de realizat în forma prevăzută în proiect – cu toate funcționalitățile incluse din start)
- Totuși, se putea altfel – lansare iterativă? “That’s not how government works”, Clay Shirky (NYU)
- Buget inițial: 93.7 mil. USD ... costuri finale: 1.7 mld. USD



If you have a complex project, follow “Gall’s law” — or it will fail, Bigthink, 2023

Administrativ – notare

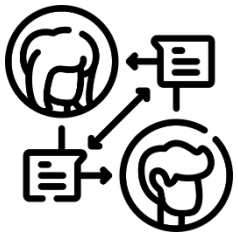
- Laborator: https://ocw.cs.pub.ro/courses/isi
- Notare
  - 40p – examen
  - 60p – semestru
    - 20p – activitate laborator
    - 40p – proiect laborator
      - 10p – prezentare inițială
      - 10p – prezentare intermediară (3 etape)
      - 20p – prezentare finală și demo
    - 5p bonus – evaluare echipe proiect
    - 5p bonus – teste curs
- Cerințe minime pentru promovare
  - 50% examen
  - 50% punctaj semestru (din fiecare activitate, exc. bonus)
  - 7 prezențe la laborator

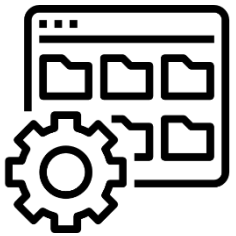
Administrativ – proiect

- Proiect
  - Tema generală: Aplicație GIS care să rezolve o problemă concretă în mod interactiv
  - Echipe de 3 studenți
    - Evaluarea echipei și a membrilor (punctaj diferențiat)
  - Etape
    - Specificarea temei – înregistrare în tabel
    - Prezentare inițială – obiective, schemă bloc, cazuri de utilizare, funcționalități, planificare
    - Prezentare intermediară
      - Setup – inițializare proiect, mediu de dezvoltare, resurse
      - Implementare – realizarea componentelor aplicației
      - Integrare – integrarea componentelor și testarea aplicației
    - Evaluare finală
      - Prezentare finală proiect
      - Demo aplicație

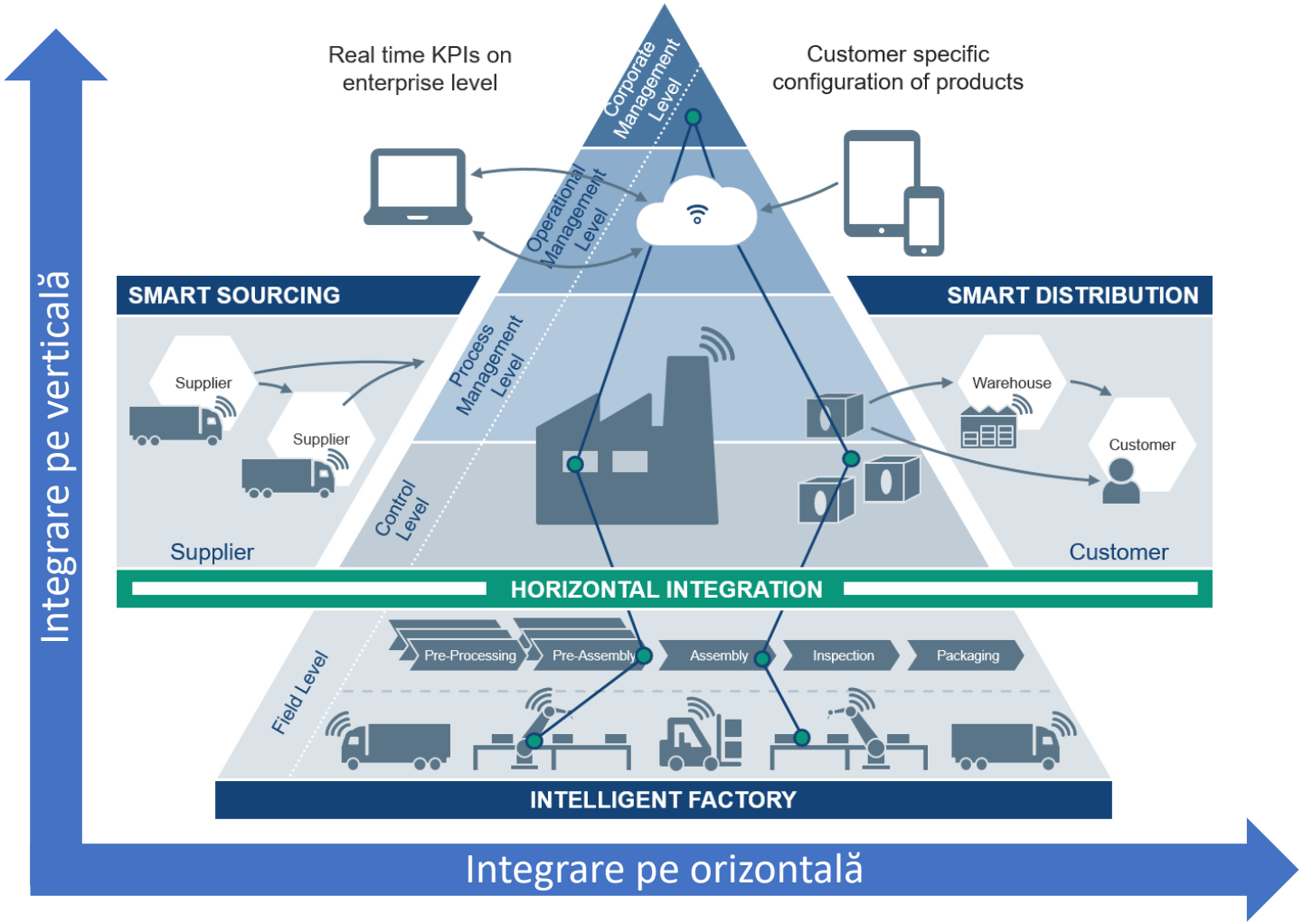
Sisteme informatice

- Un sistem informatic constă din oameni și mașini care produc și/sau folosesc informații care sunt unite prin sisteme de comunicații
- Un sistem informatic este integrat dacă:
  - Procesele de afaceri și procesele informatice care le susțin sunt corelate în profunzime
  - Legătura între diferitele programe este în mare măsură automatizată
  - Datele sunt achiziționate și disponibile pentru toate programele, fiind gestionate în mod centralizat
- Un sistem informatic redă atât procesele productive, interne cât și schimburile din interiorul firmei și dintre firmă și mediul înconjurător





Sisteme informatice integrate



Sisteme informatice integrate

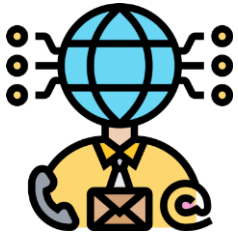
Metode de integrare la nivel de companie/industrie

- Integrarea pe verticală
  - Aceeași companie controlează produsul final dar și părțile componente
  - Exemplu: produsele Apple (ex. iPhone, iPad, Macbook) au hardware și software proiectat de Apple
- Integrarea pe orizontală
  - Exemplu: Google deține sistemul de operare Android și mai multe straturi de servicii, nu are control asupra nivelului hardware sau de marketing -> nu poate garanta succesul Android în viitor
  - Google face tranziția către abordarea verticală prin achiziționarea companiilor de hardware (ex. Motorola, HTC, FitBit), social media (ex. YouTube) și advertising (ex. DoubleClick)

Sisteme informatice

Rezolvarea problemei

- Ce fel de cerințe am?
  - scrise? verbale? complete?
- Cum pot aduna cerințele și cum le pot verifica?
- Cum obțin feedback pentru efortul meu?
  - Cum îl mențin?
- Cum reduc complexitatea integrării?
- Cum și când îmi testez produsul?
  - Când consider că este complet?



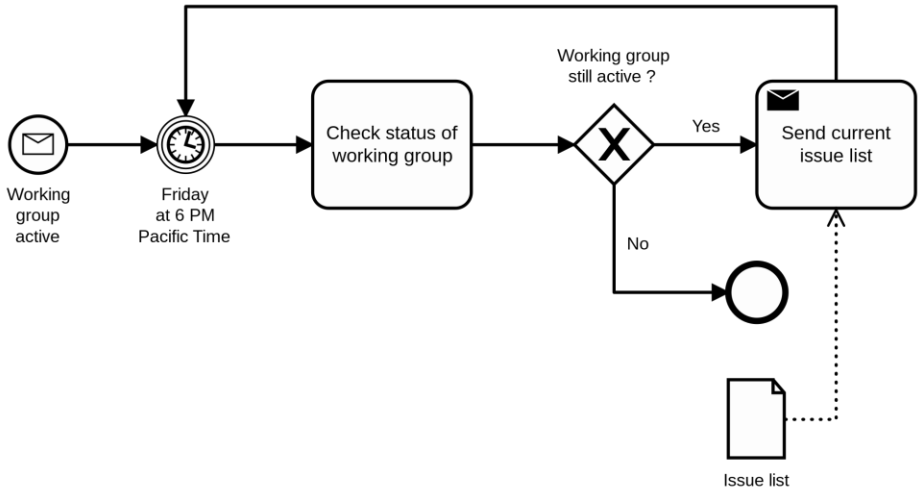
Sisteme informatice

Reutilizare și instrumente

- Ce este disponibil?
  - Comercial sau Open Source
- Ce pot folosi?
  - Buget, Complexitate, Familiaritate, Bariere legale
- Ce trebuie să folosesc?
  - Cerințe tehnice, Standarde
- Ce ajutor primesc la folosirea unor pachete?
- Cum evaluez un software Open Source?
- Ce riscuri sunt legate de reutilizare?

Procesul dezvoltării de software

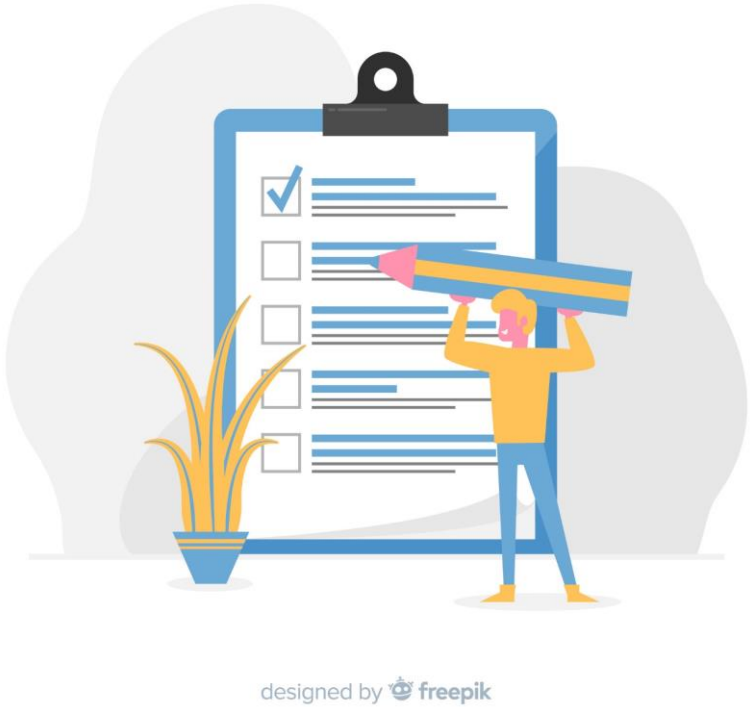
- Orice software este dezvoltat în cadrul unei structuri organizatorice și modelul proceselor – *process model* – descrie acest cadru
- Sunt descrise activitățile ce trebuie derulate și rezultatele – numite artefacte – ce trebuie realizate
- Pentru fiecare activitate se definesc roluri pentru angajați, care folosesc metode, directive, convenții, liste de verificare si modele



Procesul dezvoltării de software

Specificarea cerințelor

- Cerință (IEEE 729)
  - condiție/capabilitate necesară unui utilizator pentru a îndeplini un obiectiv
  - condiție/capabilitate care trebuie îndeplinită de sau incorporată într-un sistem /componentă pentru a satisface un contract, standard, specificație, sau alt document formal
  - reprezentare documentată a unei condiții /capabilități

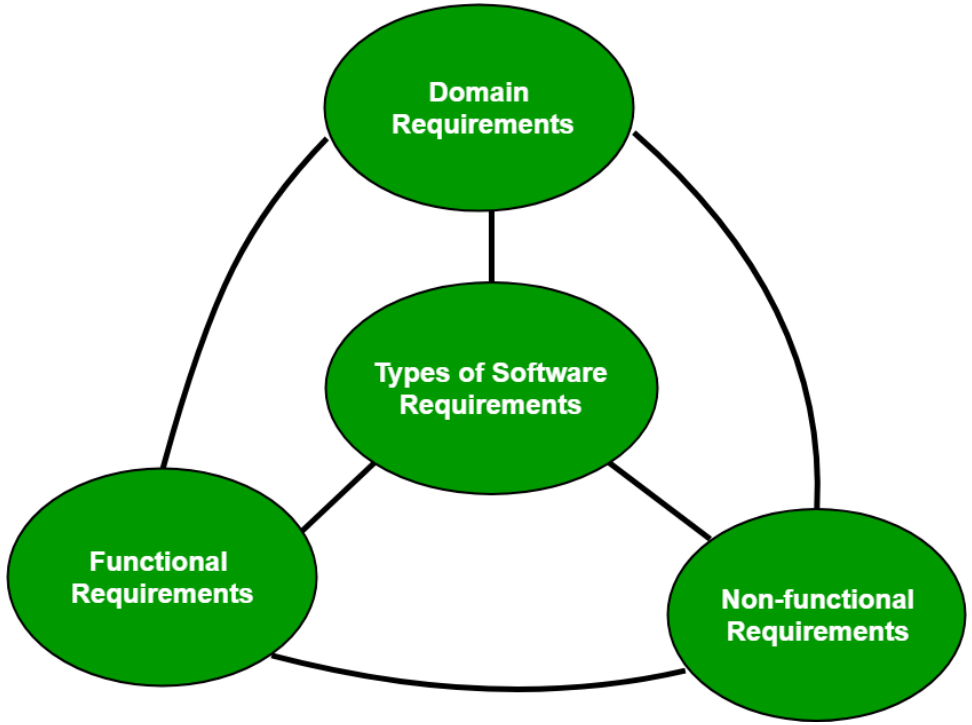


Software Engineering | Classification of Software Requirements, GeeksForGeeks

Procesul dezvoltării de software

Specificarea cerințelor

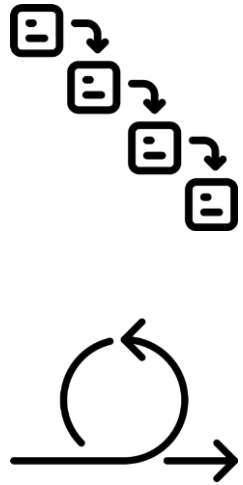
- **Funcționale** – cerințe care definesc funcționalitatea dorită de utilizatorul final
- **Non-funcționale** – cerințe de calitate ale proiectului, de ex.
  - Utilizare: stabilitate, performanță, scalabilitate, fiabilitate, securitate, portabilitate, flexibilitate
  - Dezvoltare: mentenabilitate, reutilizabilitate, extensibilitate
- **Specifice domeniului**
  - de ex. domeniu academic, medical, financiar, militar



Software Engineering | Classification of Software Requirements, GeeksForGeeks

Procesul dezvoltării de software

- Avem cerințele, cum facem cu implementarea?
- Procese de dezvoltare:
  - Waterfall – 1950, evoluat din ingineria “clasică”, preferat de organizații mari, fără implicarea directă a clientului
  - Agile – 2001, specific domeniului software, preferat de companii flexibile care dezvoltă produse noi, cu implicarea clientului încurajată la fiecare iterație



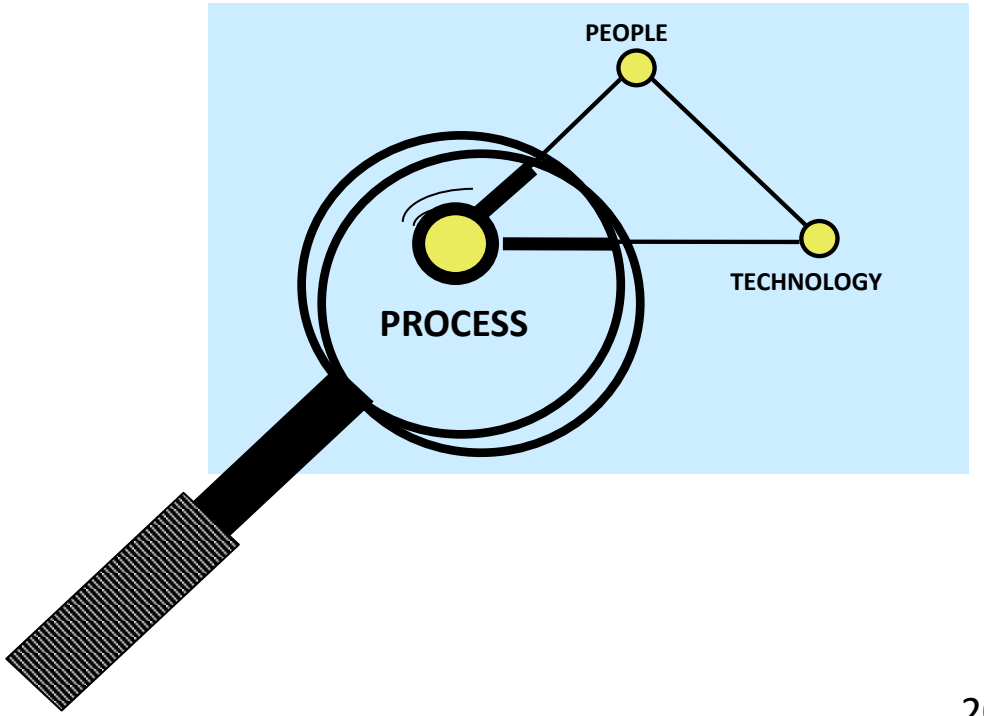
Procese de dezvoltare

Conform dicționarului Webster, un *proces* este *"a system of operations introducing something ... a series of actions, changes, or functions that achieve an end or result"*

În strânsă legătură cu metodologiile de management al proiectelor (click for more)

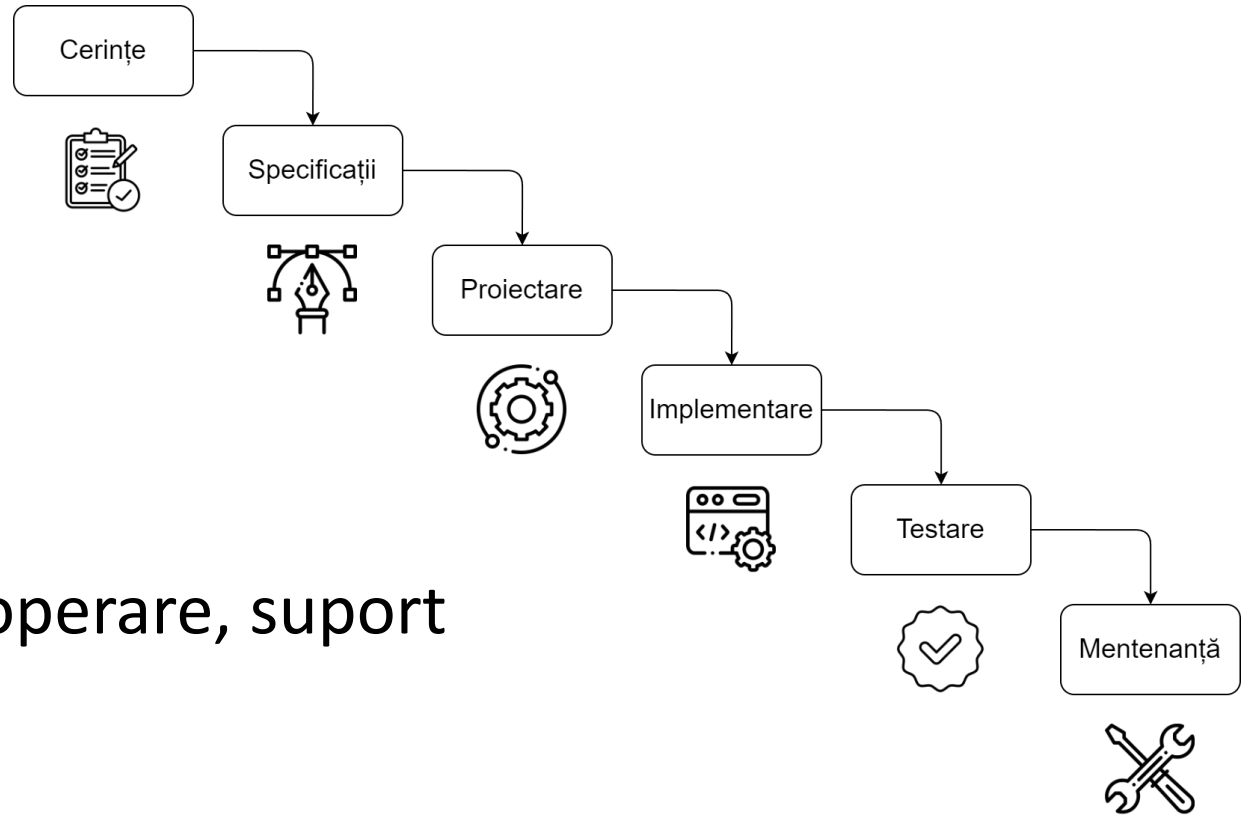
Procesul (de dezvoltare)

- Procesul este deseori descris în contextul triadei: **process – people – technology**
- Procesul este definit ca un element de legătură în cadrul unui sistem



Procese de dezvoltare – Modelul Waterfall

- Ce este Waterfall?
  - Etape succesive
    - Specificarea cerințelor
    - Proiectarea sistemului
    - Implementare software
    - Integrare și testare
    - Mentenanță – instalare, operare, suport

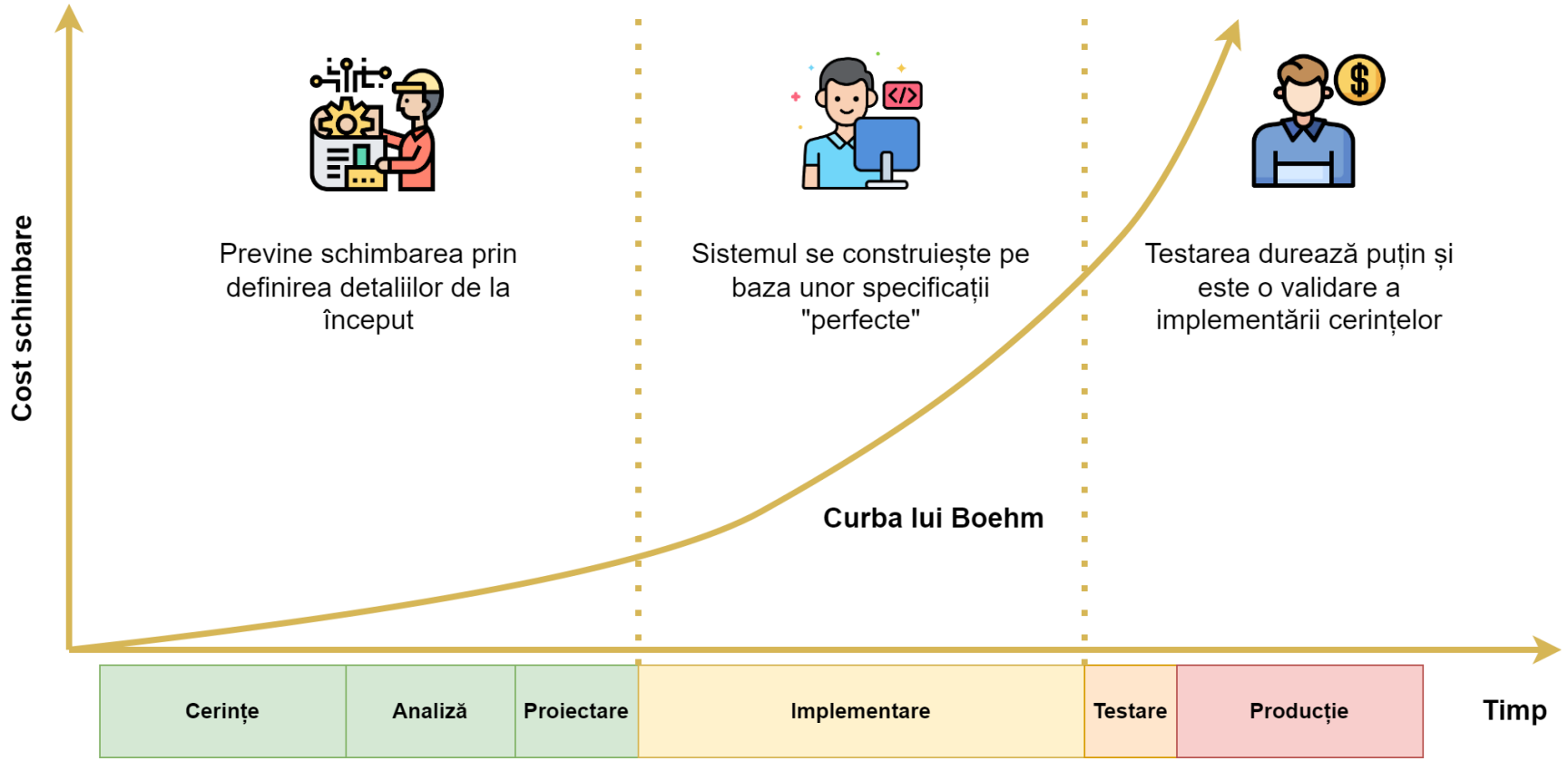


Procese de dezvoltare – Modelul Waterfall

- Se bazează pe un model clasic, ingineresc
  - Aplicabil la construcția de hardware, poduri, clădiri, mașini…
- Dezavantaje pentru software
  - Necesită specificații “complete”
    - Cerințele noi sunt penalizate
  - Integrare și testare târzie
    - Duce la soluții de ultim moment
  - Planuri, termene și estimări nerealiste
    - Nu au la bază date reale

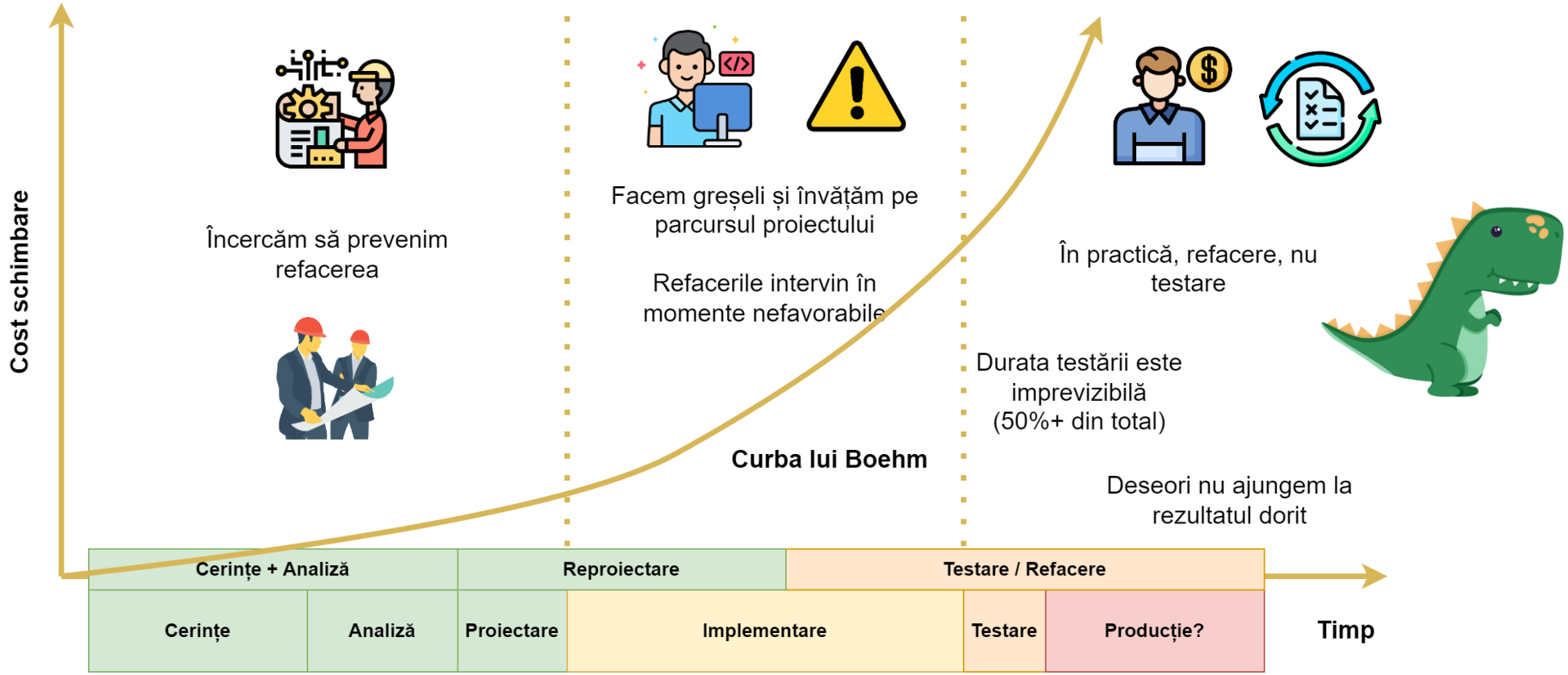
Procese de dezvoltare – Modelul Waterfall

Modelul Waterfall – *teoretic* previne schimbarea și defectele



Procese de dezvoltare – Modelul Waterfall

Modelul Waterfall – *în practică* generează multă muncă de refacere



De ce este folosit în continuare?

Modelul Waterfall

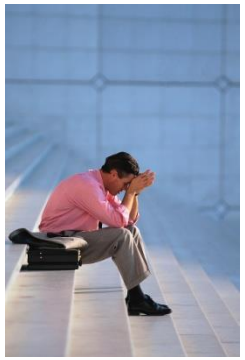
- Dă impresia că putem gestiona timpul și bugetul mai bine

dar

- Nu permite flexibilitate și creativitate pe parcursul dezvoltării
- În practică nu se folosește Waterfall în formă pură – metodele iterative sunt folosite în schimb, dar sunt la fel de vechi
- Deși pare mai simplu de gestionat și planificat – *“For every complex problem, there is a solution that is simple, neat, and wrong.”* – H. L. Mencken

Statistici privind dezvoltarea de software

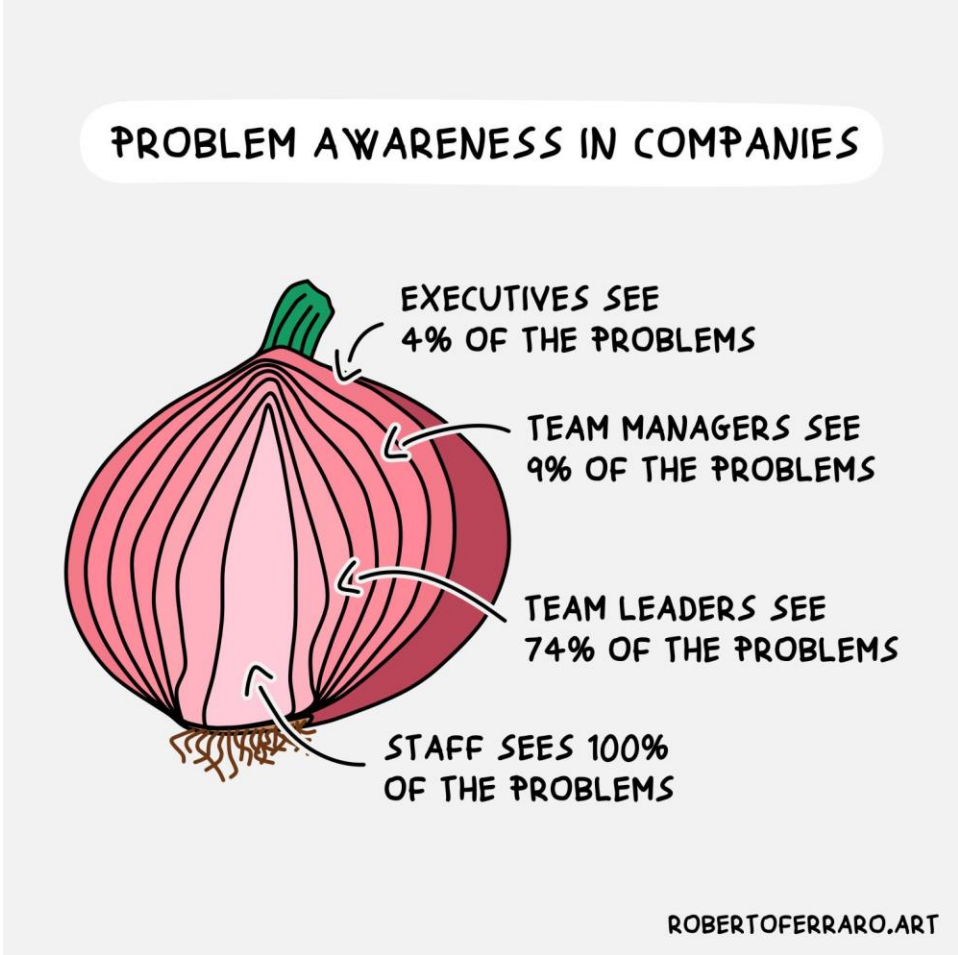
- În istoria proiectelor IT sunt multe nereușite
- 30 – 40% din proiectele de sistem eșuează înainte de finalizare [1]
- Jumătate din proiecte își depășesc bugetul sau termenul cu 200% sau mai mult [1]
- Proiectele eșuate sunt în valoare de mai mult de 100 miliarde USD/an, doar in SUA [2]
- 67% din proiectele CRM eșuează [3]



1 B.P. Lientz and K.P. Rea, *Breakthrough Technology Project Management*
2 *Computerworld*
3 *The Economist*

Unde să fie problema?

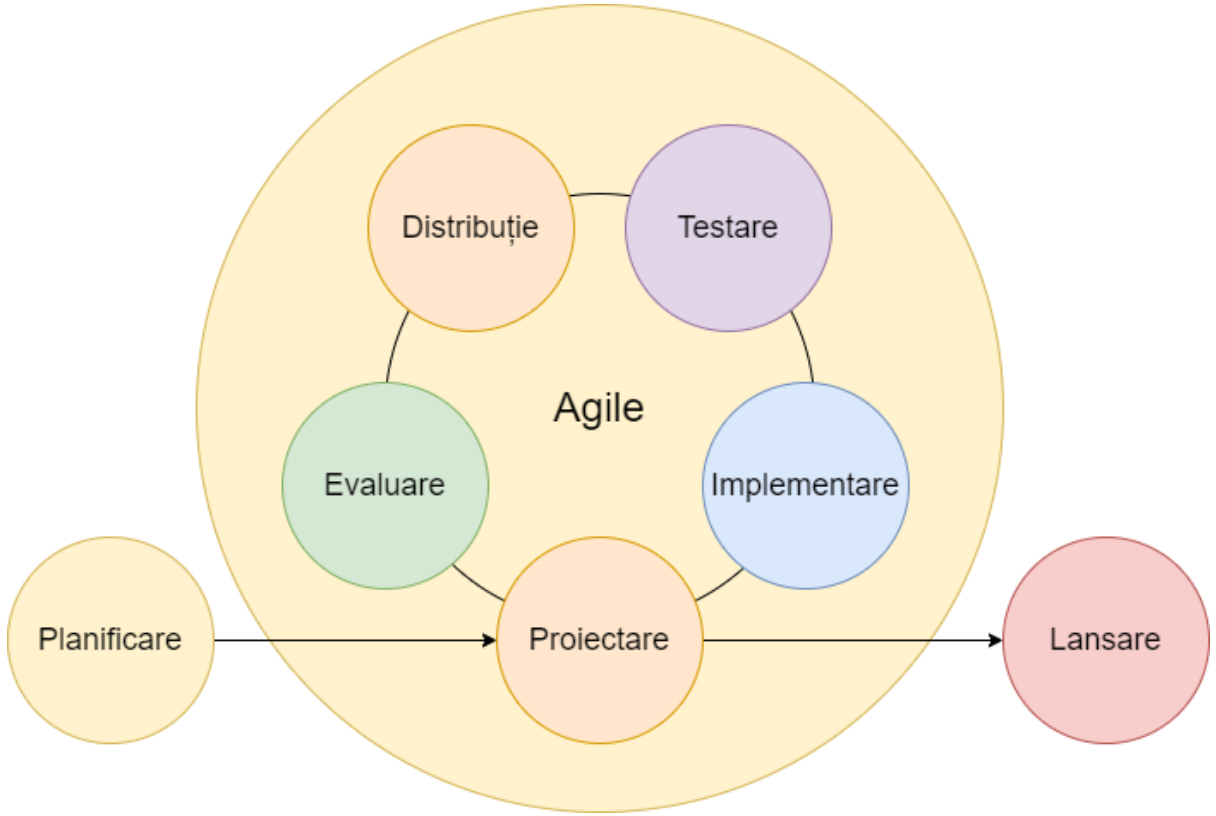
- Nu este doar despre partea tehnică/programare
- Modul de organizare a proiectului/companiei are un impact major asupra rezultatului (succes vs eșec)
- Este necesară o comunicare eficientă în cadrul organizației



Roberto Ferraro, LinkedIn, 2023

Procese de dezvoltare – Modelul Agile

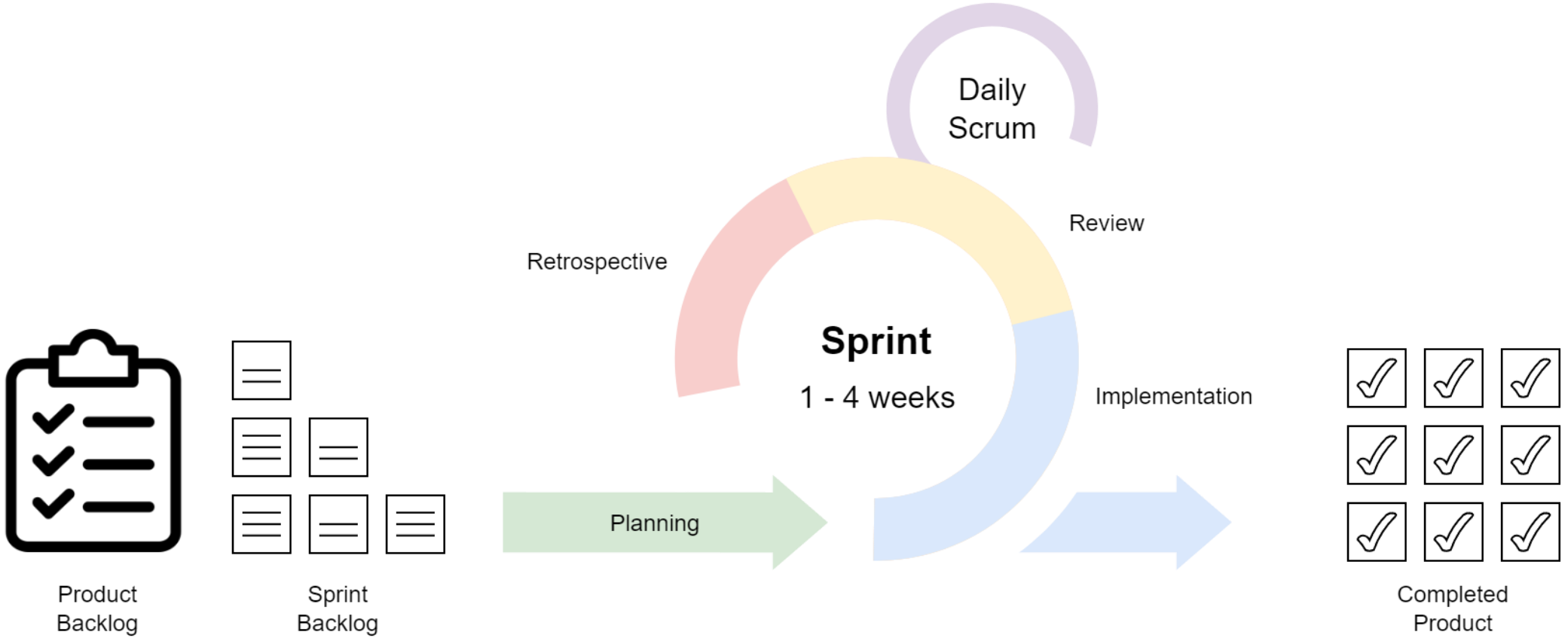
- Ce este Agile?
  - Etape succesive
    - Colectarea cerințelor
    - Proiectarea sistemului
    - Dezvoltare/iterație
    - Testare/QA
    - Distribuție/Deployment
    - Feedback



Procese de dezvoltare – Modelul Agile

- Se bazează pe un model iterativ
  - Aplicabil la dezvoltarea de software (în principal)
  - Un produs software nu este comparabil cu un obiect fizic, este mai curând comparabil cu un organism viu – necesită dezvoltare continuă
- Avantaje pentru software
  - Permite modificarea cerințelor (se întâmplă des în practică)
    - Cerințele noi sunt integrate în următoarea iterație
  - Integrare și testare continuă
    - Asigură un produs funcțional la fiecare iterație
  - Planuri, termene și estimări realiste
    - Au la bază date reale (ex. feedback) de la fiecare iterație

Procese de dezvoltare – Modelul Agile

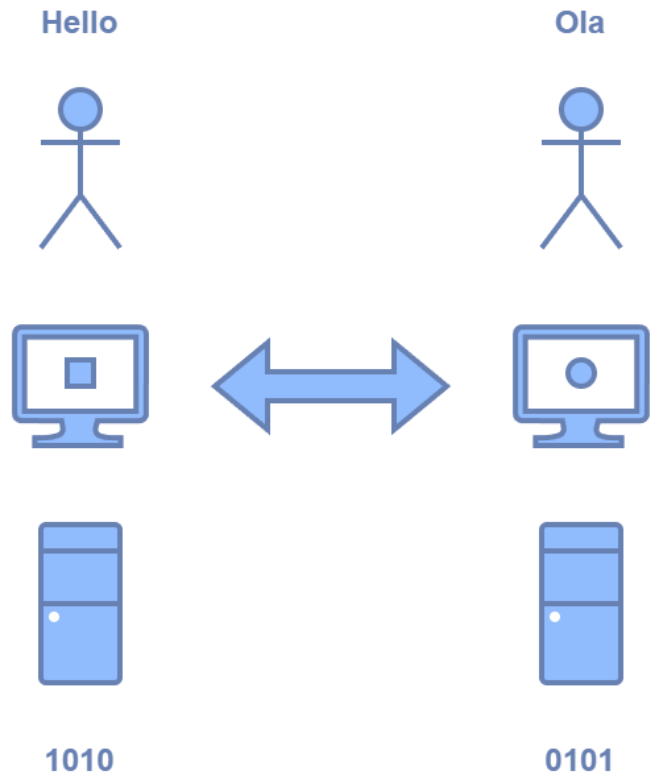


Agile project management methodology, Wrike, 2023

Concepte de bază – Integrare

Integrarea

- Proces de inginerie care creează sau îmbunătățește fluxul de informații între sisteme informaționale create pentru diferite scopuri
- Procesul de interconectare a unui sistem cu un altul, cu scopul de a asigura un schimb util de informații, date și/sau de control între sisteme.

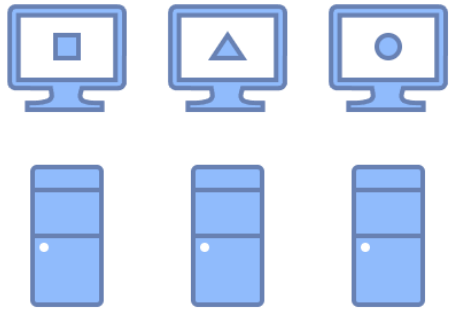


Concepte de bază – Integrare

Prima generație de software

pentru asistarea proceselor din organizații

- Soluții unicat pentru seturi particulare de funcții, asociate unor organizații individuale
- **1980-1990**: îmbunătățiri importante:
  - Creșterea capacității de calcul
  - Dezvoltarea comunicațiilor
  - Capacitate de memorare

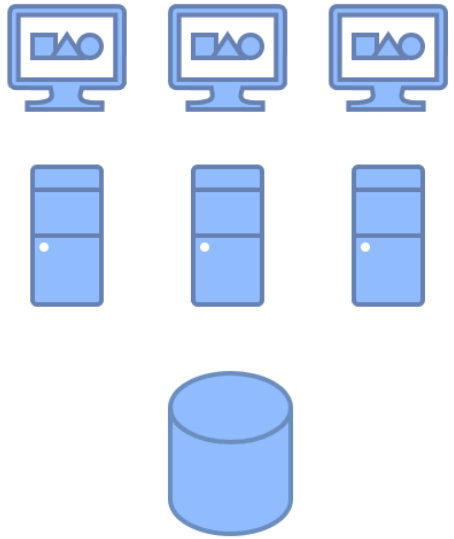


Concepte de bază – Integrare

A 2-a generație de software

pentru asistarea proceselor din organizații

- Set de sisteme software orientate pe funcții specifice
  - care operează pe o bază de date comună
  - folosind un model informațional de afaceri integrat
  - destinat unei game largi de organizații
  - adaptate într-o oarecare măsură procesului de afaceri
- **1990-2000**: dezvoltări notabile:
  - Sisteme ERP și CRM pentru întreprinderi
  - dezvoltate de companii mari (SAP, Oracle)

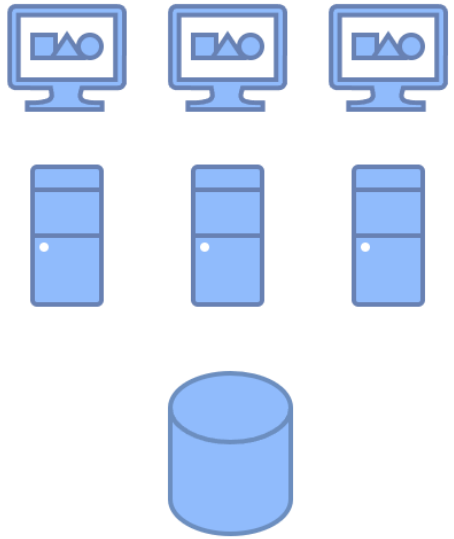


Concepte de bază – Integrare

A 2-a generație de software

pentru asistarea proceselor din organizații

- Clasificare (exemple)
  - ERP (Enterprise Resource Planning)
  - PDM (Product Data Management)
  - CRM (Customer Relations Management)
- Caracteristici
  - integrarea fluxurilor între sisteme pre-integrate
  - Integrări asistate de producător/vânzător
  - firme specializate în integrare
  - (de fapt personalizare/customizare)



Concepte de bază – Integrare

A 3-a generație de software

pentru asistarea proceselor din organizații

- Premise
  - Reducerea costurilor interne
  - Creșterea cotei de piață
  - Concentrarea pe produsele cheie/capabilitățile specializate
  - externalizarea funcțiunilor auxiliare (ex. proiectare/producție de subansamble, distribuție)
- **2000-prezent**: dezvoltări notabile:
  - Combinarea acestor atribute ale "Quality era" a dus la un model de afaceri nou pentru multe organizații de producție – “the virtual enterprise”

Concepte de bază – Integrare

Modelul conceptual descris în limbaj UML

- Diagrama de clase (abstractizare conceptuală)
- Sistemul integrat este format din componente și îndeplinește anumite funcții în cadrul unui proces bine definit
- etc.

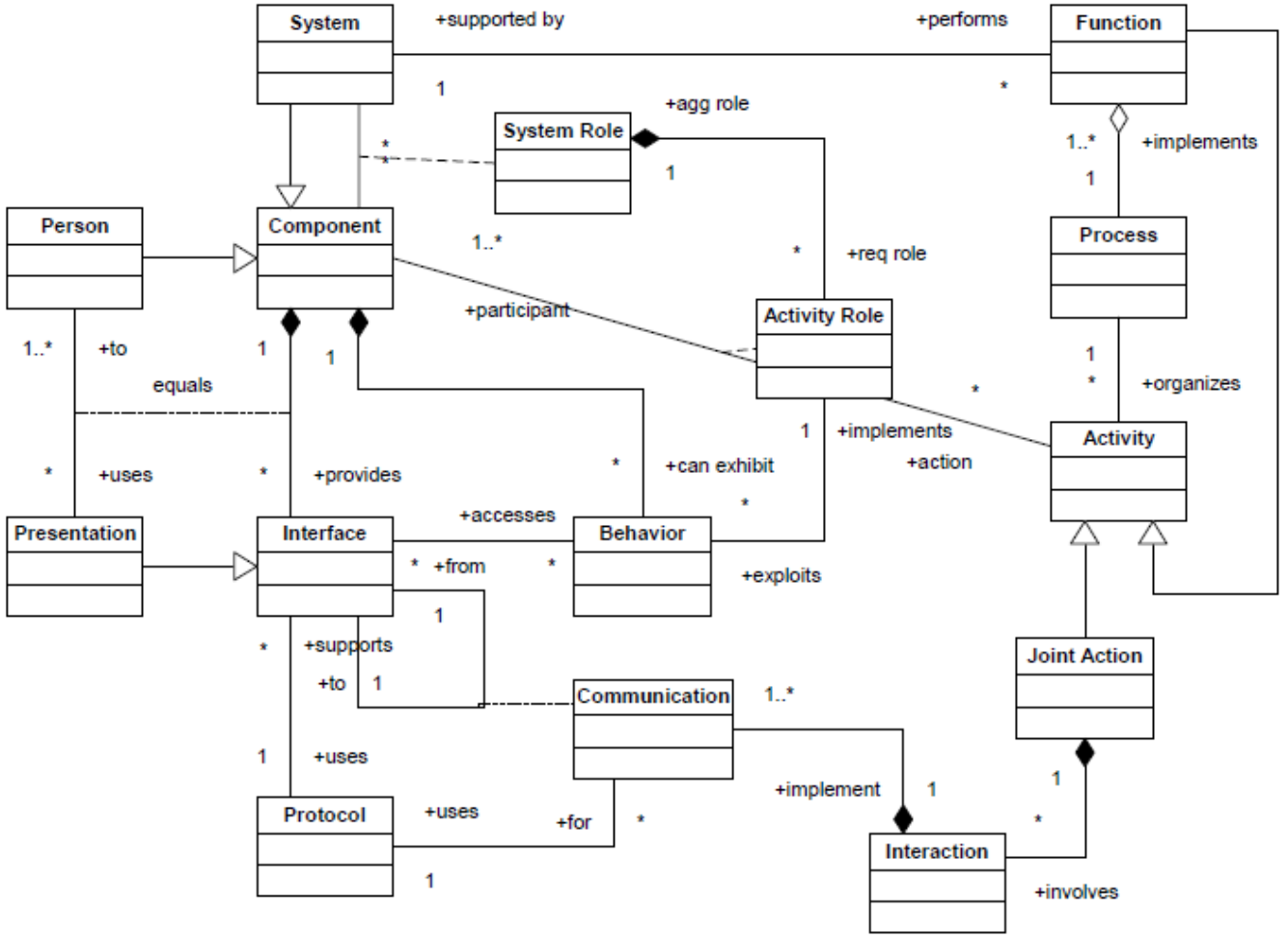


Figure 1 Fundamental integration concepts

Întrebări?