UNIVERSITATEA BABEŞ-BOLYAI
Facultatea de Matematică şi Informatică

ALGORITMI
ŞI
PROGRAMARE

Cursul 6

Programare Orientată Obiect 2

Arhitectură Stratificată

Şabloane GRASP

Ionescu Vlad
vlad.ionescu@ubbcluj.ro

Feedback săptămâna 5

- Întrebări
- Feedback

OOP – Încapsulare – **Exemplu**

- Ascunderea informației
- Reprezentarea internă a unui obiect (starea sa) trebuie protejată de restul aplicației
  - Acest lucru protejează integritatea datelor
  - Nu permite modificarea directă a stării din afara clasei => asigurarea consistenței
  - Starea se modifică prin interfața publică: toate metodele vizibile în exterior
  - Codul client trebuie să depindă doar de interfața publică

- Python nu impune sintactic aceste lucruri: **we are all adults here.**

OOP – Agregare

- O relație de tipul **parte – întreg** sau **parte din**
  - Mașina are un motor
  - Motorul are pistoane
  - Clădirea are camere

- Care dintre exemplele de mai sus e diferit de celelalte?

Arhitectura Stratificată

- Layered Architecture
- Strat / layer
  - Mecanism de structurare logică a elementelor ce compun un sistem software
  - Au responsabilități separate
  - Grup de clase / module cu același set de dependențe și refolosibile în circumstanțe similare

Arhitectura Stratificată

- User Interface
  - Interacțiunea cu utilizatorul
- Service
  - Business Logic
- Domain
  - Business Objects
- Infrastructure / Repository
  - Data access

Numărarea voturilor

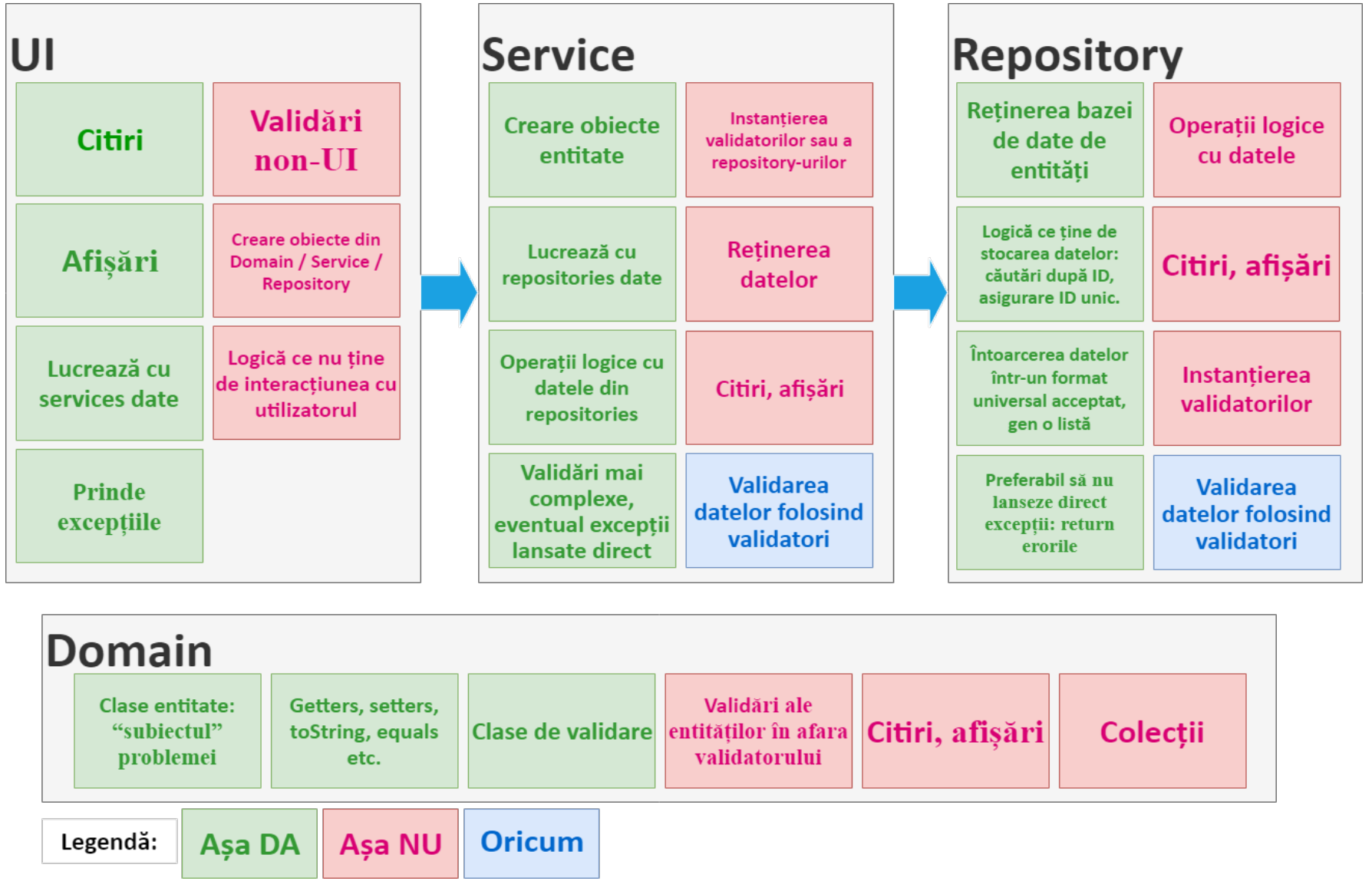
- Scrieți un program pentru gestiunea votului cu bile în parlament
  - UI
    - Add vote, count votes, clear votes
  - Service
    - Logica pentru operațiunile din UI
  - Domain
    - Vot: locație bilă albă, locație bilă neagră
  - Repository:
    - O listă sau un fișier

Numărarea voturilor

- Funcționalități:
  - F1: adăugare vot
  - F2: numărare voturi
  - F3: ștergere voturi
- Scenariu pentru F1:

#	User	Program	Explicație
1	add		Selectate adaugare
2		ID:	Programul cere un id
3	1		Userul dă id-ul 1
4		Urna bilei albe:	
5	a		Bila albă in urna albă
6		Urna bilei negre:	
7	a		Bila neagră în urna albă => vot nul

Arhitectura Stratificată – un ghid



Șabloane GRASP

- GRASP: General Responsibility Assignment Software Principles – principii de asignare a responsabilităților claselor
  - High Cohesion – discutat curs 5
  - Low Coupling – discutat curs 5
  - Information Expert
  - Controller
  - Protected Variations
  - Creator
  - Pure Fabrication

Șabloane GRASP

- High Cohesion
  - Asigurat de împărțirea în clase și straturi
  - Coeziune slabă:
    - Elementele au prea multe responsabilități, din arii diferite.
    - Greu de înțeles, reutilizat, întreținut, modificat.

Șabloane GRASP

- Low Coupling:
  - Minimizarea dependențelor între clase
  - X depinde de Y dacă:
    - X are un câmp de tip Y
    - X are o metodă care folosește tipul Y
    - X este derivat din Y

Șabloane GRASP

- Information Expert:
  - Responsabilitatea pentru un lucru îi revine clasei care are toate informațiile necesare pentru îndeplinirea sarcinii
  - Ne ajută să determinăm unde trebuie plasată o metodă, un câmp, un calcul
  - Răspunde la întrebarea **”Unde?”**

Șabloane GRASP

- Creator:
  - Descrie modul în care sunt create obiectele în aplicație
  - O clasă X ar trebui să aibă responsabilitatea de a crea obiecte de tip Y dacă sunt adevărate cât mai multe dintre:
    - X conține instanțe de tip Y
    - X gestionează instanțe de tip Y
    - X folosește des instanțe de tip Y
    - X are informațiile necesare pentru inițializarea instanțelor de tip Y

Șabloane GRASP

- Protected Variations:
  - Cum alocăm responsabilitățile astfel încât versiunile viitoare să nu necesite schimbări majore de organizare?
    - Creăm o clasă care încapsulează partea potențial instabilă

Șabloane GRASP

- Pure Fabrication:
  - Dezavantaj Information Expert: clase foarte mari
  - Când un element încalcă Low Coupling, High Cohesion:
    - Creăm o clasă artificială, care nu reprezintă ceva din domeniul problemei
    - Dar care crește coeziunea și scade cuplarea
- Exemplu:
  - Votul conține tot ce are nevoie pentru a fi reținut în memorie sau în fișier.
  - Dacă punem persistența în clasa Vot, aceasta va fi slab coezivă, cu potențial redus de refolosire.
  - Soluție: Repository se va ocupa de stocarea datelor.

Șabloane GRASP

- Grasp Controller:
  - Decuplează stratul de prezentare de restul aplicației
  - Coordonează operațiile necesare pentru a realiza acțiunea cerută de utilizator
  - În general doar coordonează, folosește alte obiecte care execută efectiv
  - Îi vom spune Service
    - pentru a nu se confunda cu Controller-ul din alte șabloane arhitecturale (de exemplu MVC – Model View Controller).
    - Mai apropiat de conceptul general de Service

Dependency Injection

- Un principiu care reduce cuplarea între componente
- Dacă un obiect X depinde de rezultatele produse de un obiect Y:
  - X nu are nevoie să știe cum e Y implementat.
  - X are nevoie să știe ce face Y, nu cum. Y trebuie să respecte un contract / protocol.
  - Y poate fi injectat în X (oferit).

Implementare – **Exemplu**

- Implementarea aplicației pentru voturile cu bile din parlament

Questions and Answers

Q & A