Organizare:
1h (prima) – studiu individual, pe baza materialului de seminar atașat
1h (a doua) – consultatii pe baza materialului atașat
– mail: vladi@cs.ubbcluj.ro, skype id: vladiela.petrascu

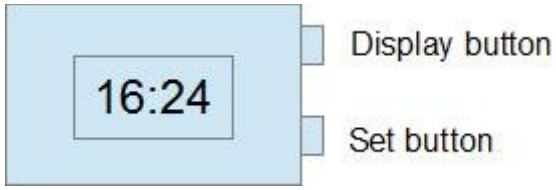
I. Proiectarea obiectuala (detaliata). Modele dinamice ale unui sistem Diagrame UML de tranzitie a starilor

Biblio:
1. Curs 2
2. Martina Seidl et al., *UML@Classroom* – pp. 85 (atașat)
3. E. Gamma et al., *Design Patterns. Elements of Reusable Object-Oriented Design* - State Pattern (atasat)
4. *OMG UML Spec 2.5.1* *https://www.omg.org/spec/UML/About-UML/* - pp.305 (pentru aspecte punctuale)

La fel ca și diagramele de interacțiune (si cele de activități), diagramele UML de tranzitie a starilor sunt instrumente folosite pentru a reprezenta modele dinamice ale unui sistem.
Dacă diagramele de secvența/comunicare surprind comportamentul dinamic prin prisma obiectelor care iau parte la interacțiune și a schimbului de mesaje intre acestea, diagramele de tranzitie a starilor surprind comportamentul prin prisma starilor prin care trece un obiect (subsistem/grup de obiecte relationate) de-a lungul ciclului sau de viața și a tranzitiilor posibile intre aceste stari. Astfel de tranzitii pot fi declansate de aparitia unor evenimente externe, de indeplinirea anumitor condiții sau de trecerea unui interval de timp.

O diagrama de interacțiune se realizeaza, de regula, pentru un scenariu al unui caz de utilizare. Diagramele de tranzitie a starilor se asociaza claselor (grupurilor de clase) din sistem, însă nu tuturor, ci doar celor caracterizate de un comportament dinamic semnificativ (răspund diferit la stimulii din mediu funcție de starea în care se afla).

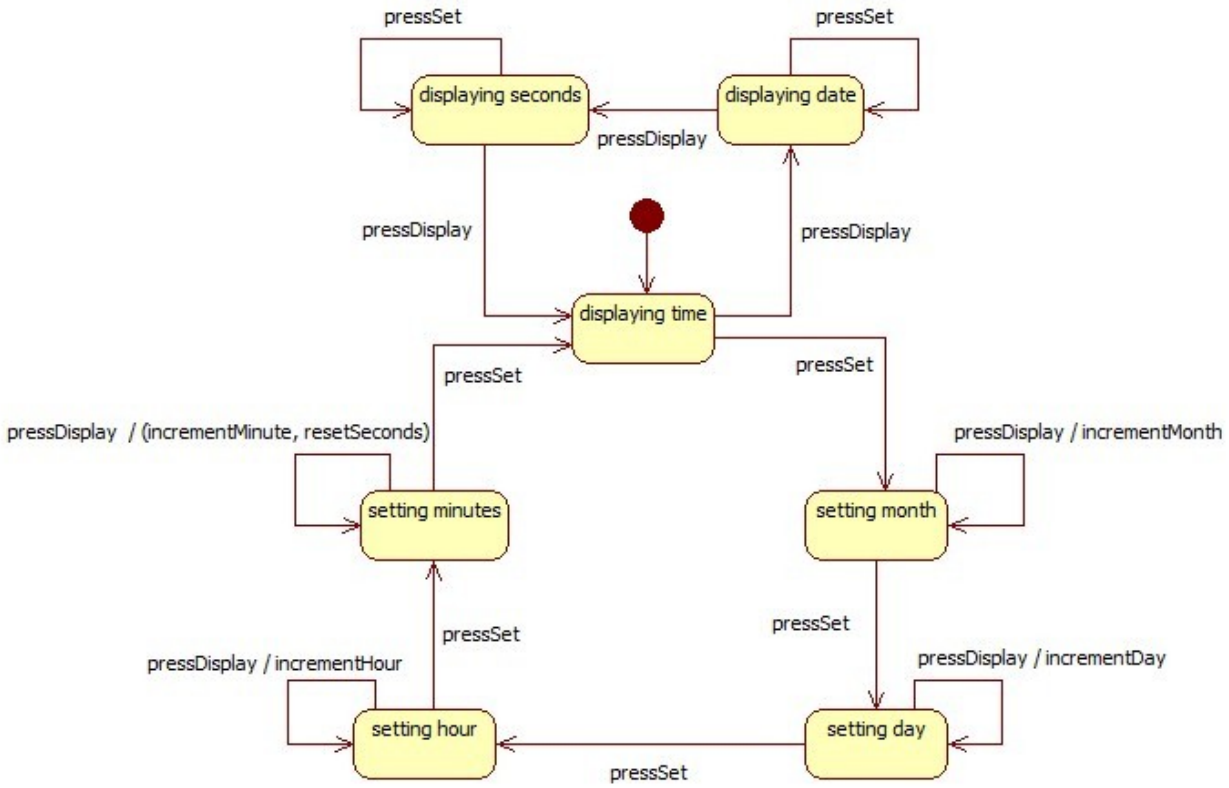
Exemplu: Consideram exemplul unui ceas electronic simplu, care ofera functionalitati de vizualizare a timpului curent (ora, minut, secunde) și a datei curente (zi, luna) și de setare a diferitelor sale componente (ora, minut, luna, zi – secundele nu se seteaza manual, ci se reseteaza la fiecare schimbare a minutului).



În interfata, ceasul ofera utilizatorului doua butoane, unul pentru afisaj (Display button) și unul pentru reglaje (Set button).
După inserarea bateriei, ceasul intra în modul de afisaj al timpului curent (în formatul *ora* :

minut). Din aceasta stare, prin actionarea repetata a butonului Display, se permite ciclarea intre starile de afisaj: afisare ora -> afisare data (format *zi* : *luna*) -> afisare secunde (format _ : *secunde*) -> afisare ora ... s.a.m.d. În starile de afisaj a date și a secundelor, actionarea butonului Set nu are nici un efect.
Din starea de afisaj a timpului curent, prin actionarea repetata a butonului Set, se cicleaza între starile de reglaj: reglaj luna -> reglaj zi -> reglaj ora -> reglaj minut -> afisare timp -> reglaj luna ... s.a.m.d. Într-o stare de reglaj, actionarea butonului Display permite incrementarea componentei curente cu o unitate (pana la capătul intervalului aferent de valori și apoi de la capăt). Într-o stare de reglaj, componenta curenta reglata este afisata intermintent.

Diagrama de tranzitie a starilor corespunzatoare acestui comportament este următoarea (acțiunile utilizatorului de apasare a celor doua butoane corespund evenimentelor care declanseaza tranzitiile intre stari; în starile de reglaj, tranzitiile generate de apasarea butonului Display au acțiuni asociate):



Un astfel de ceas prezinta un comportament dinamic dependent de stare: un același eveniment (apasarea unui buton) are efecte diferite, funcție de starea curenta în care se afla obiectul.
Considerand o structura a ceasului care conține o parte de interfata, un control (ClockController) și o memorie asociata (ClockMemory), diagrama de tranzitie a starilor anterioara va corespunde clasei de tip control.

Variante de proiectare a clasei ClockController:

1. ClockController reține o referinta catre obiectul memorie asociat și un atribut de tip enumerare/intreg, corespunzător starii sale curente. Evenimentele și acțiunile de pe

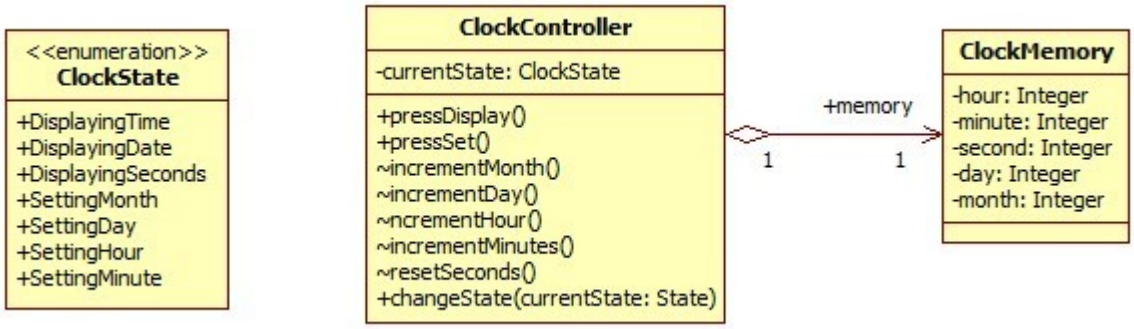
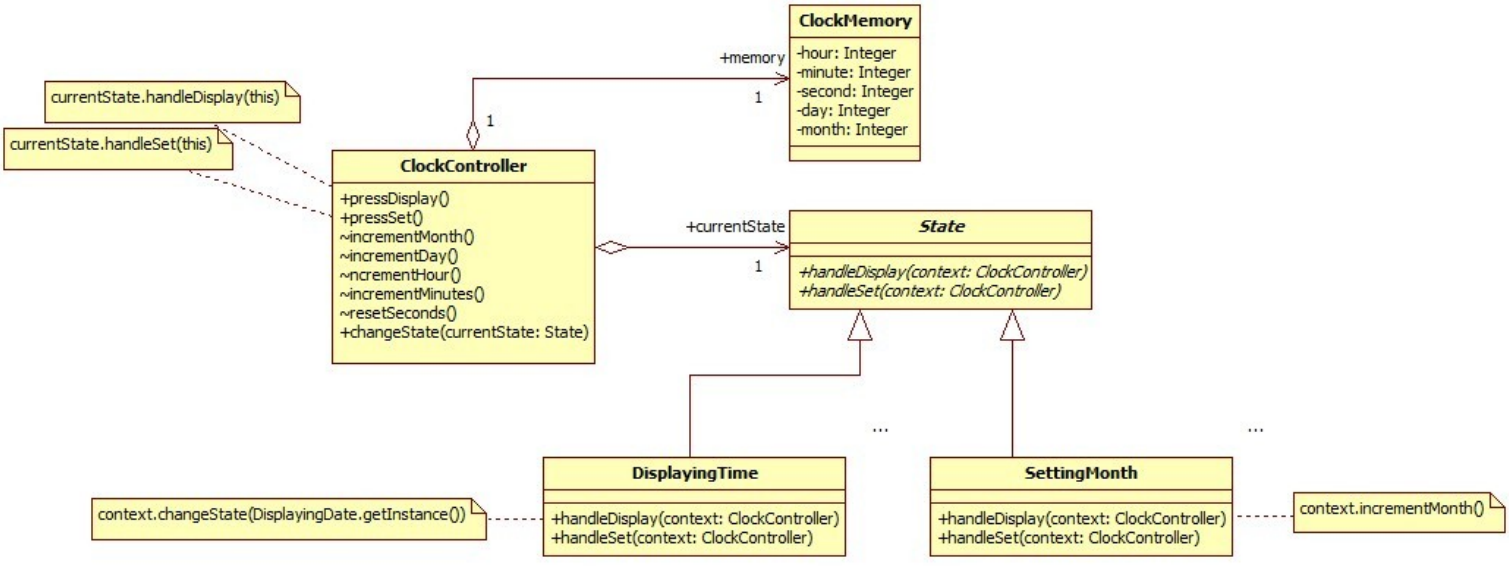


diagrama de tranzitie a starilor corespund unor operatii la nivelul clasei.
În acest caz, logica operatiilor pressDisplay() și pressSet() va folosi structuri conditionale (if/switch/case) pentru a gestiona schimbarile de stare și eventualele acțiuni asociate. Necesitatea de a lua în considerare o stare noua ulterior fnalizarii sistemului va determina modificari la nivelul acelor structuri conditionale, în toate metodele corespunzatoare evenimentelor.

? Ce pricipiu de proiectare enuntat la curs încalca aceasta abordare? Pentru bonus de seminar :), trimiteți răspunsul la aceasta întrebare și explicatiile aferente pe adresa vladi@cs.ubbcluj.ro, pana la finalul seminarului grupei de care apartineti.

2. ClockController reține o referinta catre obiectul stare asociat (instantiere a sablonului de proiectare State, vezi [3] pentru detalii). Obiectul control delega acestui obiect stare gestionarea evenimentelor (prin metodele de tip handler). Pentru a putea fi gestionata schimbarea de stare, obiectul context (ClockController) se transmite pe sine ca și parametru al metodelor handler.
În acest mod, logica de gestionare a evenimentelor este incapsulata la nivelul obiectelor stare concrete. Luarea în considerare a unei stari noi presupune crearea unui nou obiect stare concret și implementarea handlerelor corespunzatoare.



II. Specificarea constrangerilor pe modelele UML folosind limbajul OCL

Biblio:
1. Curs 9
2. *OMG OCL Spec 2.4* *https://www.omg.org/spec/OCL* (pentru aspecte punctuale)

Vezi cerintele și soluția din directorul *ocl* atașat.