

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
3. E. Gamma et al., *Design Patterns. Elements of Reusable Object-Oriented Design* - State Pattern
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 (și cele de activități), diagramele UML de tranziție a stărilor sunt instrumente folosite pentru a reprezenta modele dinamice ale unui sistem.

Dacă diagramele de secvență/comunicare surprind comportamentul dinamic prin prisma obiectelor care iau parte la interacțiune și a schimbului de mesaje între acestea, diagramele de tranziție a stărilor surprind comportamentul prin prisma stărilor prin care trece un obiect (subsistem/grup de obiecte relate) de-a lungul ciclului sau de viață și a tranzițiilor posibile între aceste stări. Astfel de tranziții pot fi declanșate de apariția unor evenimente externe, de îndeplinirea anumitor condiții sau de trecerea unui interval de timp.

O diagrama de interacțiune se realizează, de regulă, pentru un scenariu al unui caz de utilizare. Diagramele de tranziție a stărilor se asociază 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: Considerăm exemplul unui ceas electronic simplu, care oferă funcționalități 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 – secunde nu se setează manual, ci se resetează la fiecare schimbare a minutului).



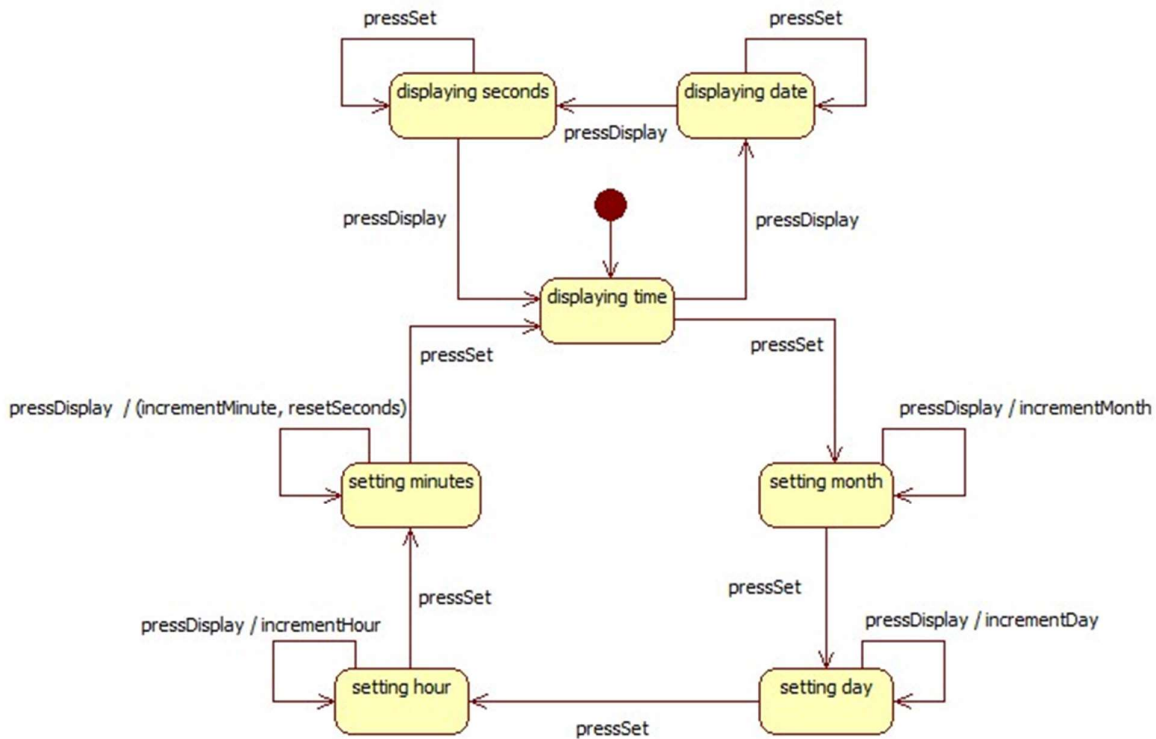
În interfața, ceasul oferă utilizatorului două butoane, unul pentru afișaj (Display button) și unul pentru reglaje (Set button).

După inserarea bateriei, ceasul intră în modul de afișaj al timpului curent (în formatul *ora : minut*). Din această stare, prin acționarea repetată a butonului Display, se permite ciclarea între stările de afișaj: afișare ora -> afișare data (format *zi : luna*) -> afișare secunde (format *_ : secunde*) -> afișare ora ... s.a.m.d. În stările de afișaj a date și a secundelor, acționarea butonului Set nu are nici un efect.

Din starea de afișaj a timpului curent, prin acționarea repetată a butonului Set, se ciclează între stările de reglaj: reglaj luna -> reglaj zi -> reglaj ora -> reglaj minut -> afișare 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 intermiment.

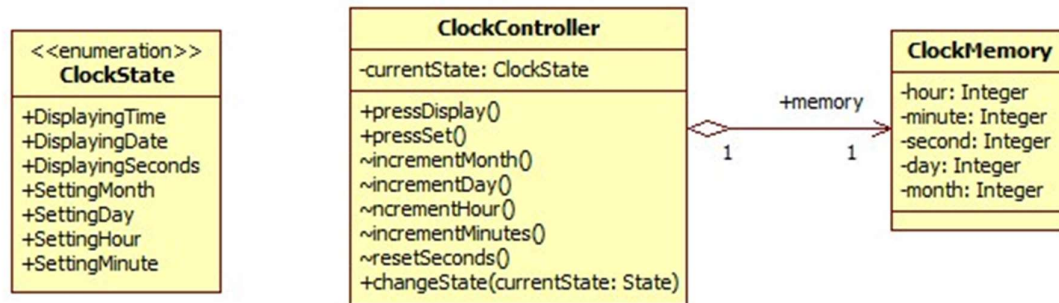
Diagrama de tranzitie a starilor corespunzatoare acestui comportament este următoarea (acțiunile utilizatorului de apasare a celor doua butoane corespund evenimentelor care declanseaza tranzitiile intre stari; în starile de reglaj, tranzitiile generate de apasarea butonului Display au acțiuni asociate):



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 referință către obiectul memorie asociat și un atribut de tip enumerare/întreg, corespunzător stării sale curente. Evenimentele și acțiunile de pe diagrama de



tranzitie a starilor corespund unor operatii la nivelul clasei.

În acest caz, logica operațiilor `pressDisplay()` și `pressSet()` va folosi structuri conditionale (if/switch/case) pentru a gestiona schimbările de stare și eventualele acțiuni asociate. Necesitatea de a lua în considerare o stare nouă ulterior finalizării sistemului va determina modificări la nivelul acelor structuri conditionale, în toate metodele corespunzătoare evenimentelor.

2. ClockController reține o referință către obiectul stare asociat (instantiere a sablonului de proiectare State, vezi [3] pentru detalii). Obiectul control delegă acestui obiect stare gestionarea evenimentelor (prin metodele de tip handler). Pentru a putea fi gestionată 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 încapsulată la nivelul obiectelor stare concrete. Luarea în considerare a unei stări noi presupune crearea unui nou obiect stare concret și implementarea handlerelor corespunzătoare.

