**Tema 4**

Tehnici de programare

Aplicatie Bancara

Toderean Alexandru

**An academic :2014 – 2015**

CUPRINS

1. Obiectivul temei . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .3

2. Analiza problemei. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . . .3

3.Modelare. . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .3

5. Cazuri de utilizare. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

6. Proiectare (diagrame UML, structure de date, proiectare clase, interfete, relatii, packages, algoritmi, interfata utilizator). . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

7. Implementare si testare . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .9

8. Rezultate. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

9. Concluzii, ce s-a invatat din tema, dezvoltari ulterioare . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

10. Bibliografie. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . . . . . . . . ..11

# 1.Obiectivul temei

Obiectivul acestei teme este proiectarea si implementarea unei aplicatii pentru o managementul unei banci. Aceasta aplicatie implemeteaza toate functiile pe care le are un client al unei banci, cum ar fi crearea unui anumit cont, de economii sau de cheltuieli.

Pentru a nu mai folosi baze de date, de data aceasta vom folosi un fisier pentru a stoca doate datele. Astfel cand rulam aplicatia preluam toate datele din fisier , iar la finalul ei toate datele din aplicatie le vom scrie inapoi in fisier. Un alt obiectiv al acestei teme este intelegerea conceptelor de serializare si deserializare a obiectelor. Un alt obiectiv este si acela de a ne familializa cu tabela de dispersie.

# 2.Analiza problemei

In primul rand, pentru rezolvarea temei avem nevoie de o buna intelegere a principiului serializare/deserializare. Serializarea obiectelor inseamna salvarea si restaurarea starii obiectelor. Obiectele oricarei clase care implementeaza interfata Serializable, pot fi salvate  intr-un stream(fluxde date) si restaurate  din acesta.  Pachetul java.io contine doua clase speciale ObjectInputStream respectiv ObjectOutputStream pentru serializarea tipurilor primitive.

Desing by Contract  reprezinta un "contract" care specifica restrictiile la care trebuie sa se supuna datele de intrare ale unei metode, valorile posibile de iesire si starile in care se poate afla programul - aceste restrictii sunt date sub forma unor:

a) preconditii: reprezinta obligatiile pe care datele de intrare ale unei metode trebuie sa le respecte pentru ca metoda sa functioneze corect

b) postconditii: reprezinta garantiile pe care datele de iesire ale unei metode le ofera

c) invarianti: reprezinta conditii impuse starilor in care programul se poate afla la un moment dat

Aplicatia trebuie sa realizeze o interfata cu utlilizatorul si banca. Utilizatorul trebuie

sa poata sa isi creeze conturi noi, sa . Astfel toate datele din aplicatie trebuie retinute intr-un fisier , iar la fiecare deschidere a aplicatiei toate datele din fisier vor fi preluate in aplicatia noastra, la fel si la iesirea din aplicatie toate datele vor fi scrie inapoi in fisier.

Clientul poate sa isi creeze conturi de tipul „Saving Acoount” sau „Spending Account”, depinzand de nevoile acestuia, sa depuna sau sa scoata bani din cont. Administratorul trebuie sa poata sa vizualizeze, sa stearga, sa caute anumite conturi, dupa nume, id, etc.

# 3.Modelare:

In realizarea modelarii aplicatiei am folosit conceptul de Desing by Contract dar si Serializable si Deserializable. Pentru retinerea conturilor, am folosit un obiect de tip LinkedHashMap, obiecte ce asociaza fiecarui element o cheie unica (perechi de tipul cheie – valoare). Nu pot contine asadar chei duplicate si fiecare chei este asociata la un singur element.

Astfel, acest Hashmap are ca si cheie un String, reprezentand CNP-ul clientului si o lista de intregi de tipul Account. Am folosit conceptul Design by Contract prin implmentarea Intervetei BankProc, clasa Bank implementand aceasta interfata. In ceea ce priveste paradigmele OOP, toate atributele din toate clasele sunt **private.** Astfel am realizat incapsularea la fiecare clasa. Am folosit mostenirea si Design by Contract.

# 5.Cazuri de utilizare:

Pentru a putea folosi acest program utilizatorul trebuie sa urmeze urmatorul sir de evenimente:

1. Utilizatorul ruleaza aplicatia

2.Alege operatia corespunzatoare din cele 2 Jtable.

3. Utilizatorul introduce datele clientului

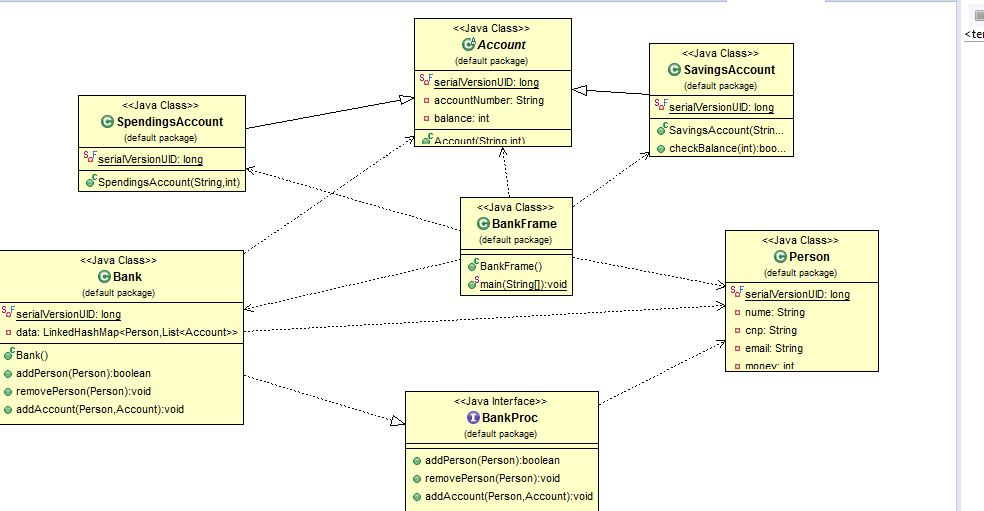
4. Ulilizatorul apasa butonul corespunzator operatiei pe care vrea sa o executesi asteapta ca operatia sa se efectueze.

5. Programul executa operatia si afiseaza rezultatul .

6. Pentru a iesi din aplicatie se apasa butonul de Exit. (X)

6.Proiectare:

Diagrama UML:



Cu ajutorul diagramei UML, de mai sus, a avut loc proiectarea, fiind evidentiate pachetele, clasele, metodele, dar si legaturile dintre ele, respecandu-se modelul OOP.

UML este un limbaj vizual de modelare, el nu este încă un limbaj de programare, deoarece nu dispune de întreg sprijinul semantic și vizual pentru a înlocui limbajele de programare. Limbajul este destinat vizualizării, specificării, construirii și documentării sistemelor de aplicații, dar are limitări în ceea ce privește generarea codului. UML reunește cele mai bune tehnici și practici din domeniul ingineriei progamării, care și-au dovedit eficiența în construirea sistemelor complexe.

Diagramele sunt grafuri care prezintă simboluri ale elementelor de modelare (model element) aranjate astfel încât să ilustreze o anumită parte sau un anumit aspect al sistemului. Un model are de obicei mai multe diagrame de același tip. O digramă este o parte a unui view specific, dar există posibilitatea ca o diagramă să facă parte din mai multe view-uri, în funcție de conținutul ei.

Interfata BankProc reprezinta interfata pe care clasa Bank o va implementa, astfel toate metodele pe care le dorim sa le realizam asupra conturilor in aceata aplicatie trebuie sa fie definite in aceasta interfata. Astfel orice dezvoltare ulterioara va fi usor de realizat. In aceasta interfata am realizat pentru fiecare metoda pre si post conditii, pentru a fi mai usor de inteles pentru cineva care doreste sa realizeze o dezvoltare ulterioara a aplicatiei.

Clasa Bank implementeaza interfata BankProc, si Serializable reprezentand clasa in care stocam conturile. Aceasta clasa este implementata cu un LinkedHashMap in care cheia este CNP-ull, de tipul String, si valoarea, o lista de conturi. Am implementat Serializable deoarece obiectul de tip Bank va fi serializat si deserializat. BankProc si cateva metode care ne ajuta sa realizam aceste metode. Toate metodele de aici sunt folosite in clasa GUI si anume in interfata cu utilizatorul.

Clasa Person este clasa care are ca si atribute toate atributelele unei persoane. Si anume : nume,email si CNP In aceasta clasa am avut nevoie doare de metode pentru get si o metoda isWellFormed.

Interfata grafica a utilizatorului (BankFrame) este un termen cu inteles lar care se refera la toate tipurile de comunicare vizuala intre un program si utilizatorii sai, Aceasta este o particularizare a interfetei cu utilizatorul (UI), prin care vom intelege conceptul generic de interactiune dintre program si utilizatori. Limbajul Java pune la dispozitie numeroase clase pentru implementarea diverselor functionalitati UI.

In principiu, crearea unei aplicatii grafuce presupune:

-Un Design: crearea unei suprafere de afisare pe care vor fi asezate obiecte grafice (componente) care servesc la comunicarea cu utilizatorul butoane, controale pentru editarea textelor, liste, etc), crearea si asezarea componentelor pe suprafata de afisare a pozitiile corespunsatoare.

-Functionalitatea: definirea unor actiuni care trebuie sa se execute in momentul cand utilizatorul interactioneaza cu obiectele grafice alea aplicatiei.

Dupa cum observam in imaginea de mai sus, interfata aplicatiei este foarte usor de inteles si navigat prin aplicatia deoarece am incercat sa o fac cat mai vizibila si nu prea complicate. Am reusit acest lucru datoria Meniului din bara de Meniu. Aici am introdus toate comenziile pe care le doresti sa le faci

8.Rezultate:

In momentul rulari aplicatie, aceasta prea toate datele dintr-un fisier. Astfel mereu aplicatia va avea mereu toate datele actualizate. Dupa ce am salvat toate datele deja putem efectua diferite operatii pe conturile din banca. Dupa ce am realizat operatiile dorite (stergere, inserare cont sau adaugare) toate datele vor fi scrise inapoi in fisier, astfel ele ramn sigure. Orice operatie realizata in momentul rularii aplicatiei se va salva intr-un fisier, astfel e ca si cand am folosi o baza de date insa folosim un fisier in care depozitam toate conturile din banca . In momentul rulari obiectul de tipul HashTable unde este depozitat conturile va prelua toate datele din acel fisier.

# 9.Concluzii, dezvoltari ulterioare:

Scopul acestei teme a fost atins, astfel incat programul ruleaza .Operatiile au fost implementate cat mai simplu, dorindu-se o intelegere si liniaritate a codului.Cu siguranta acestea nu sunt singurele metode, dar au fost cele mai bine intelese de catre mine si m-au ajutat sa realizez functionalitatea aplicatiei.

Ce am invatat?

In primul rand, ce e mai important e ca mi-am imbogatit cunostintele programarii orientate pe obiect, am invatat sa pun mai bine in practica paradigmele OOP. Am invatat sa folosesc concept noi cum ar fi : Desing by Contract si Serializable si Deserializable

Am realizat ca primul lucru la un proiect, dar mai ales la proietele mai mari e intelegerea cerintei si punerea acesteia in practica prin structurarea ideilor si analiza lor.

Dezvoltari ulterioare:

Depinzand de scopul pentru care se va folosi, aplicatia are numeroase posibilitati de extindere, de dezvoltare si de imbunatarire, atat a algoritmilor cat si a design-ului si a operatiilor pe care le efectueaza. Unele dintre aceste dezvoltari pot si:

1. Interfata mai atractiva
2. Mai multe tipuri de tranzactii dintr-un cont
3. Conturi in functie de varsata, de exemplu daca clientul are varsta mai mica decat 18 ani sa poate scoate bani doar la 18 ani, sau sa aiba un imputernicit pana la carsta de 18 ani.
4. Useri diferiti pentru anumite modificari aduse conturilor
5. Cod pin pentru fiecare cont

Lucrurile acestea se pot remedia prin cateva linii de cod puse la locul potrivit .

# 7.Bibliografie

Java de la 0 la expert ,Autor (i) : Ștefan Tănasa, Cristian Olaru, Ștefan Andrei, Editura : Polirom, An apariție : 2007

Introducere în universul JAVA, Autor (i) : Horia Georgescu, Editura : Tehnica

<http://en.wikipedia.org/wiki/Swing_%28Java%29>

<http://docs.oracle.com/javase/tutorial/uiswing/>

<https://docs.oracle.com/javase/tutorial/collections/interfaces/map.html>

<https://docs.oracle.com/javase/8/docs/api/java/util/HashMap.html>

<http://bytes.com/topic/java/answers/609266-buliding-english-dictionary-using-hashmap>

<https://docs.oracle.com/javase/8/docs/technotes/guides/language/assert.html#compiling>

<http://stackoverflow.com/questions/8962459/java-collections-keyset-vs-entryset-in-map>

<http://stackoverflow.com/questions/5922956/java-dictionary-searcher>

<http://docs.oracle.com/javase/7/docs/api/java/lang/String.html>

<http://www.ms.sapientia.ro/~manyi/teaching/oop/oop_romanian/curs15/curs15.html>

<http://loose.upt.ro/~oose/pmwiki.php/GOOD/Laborator3>

http://control.aut.utcluj.ro/isp/lab5/colectiiobiecte.htm