Management banca
Tehnici de programare
Tema 4

Alexandru Nistor Botolan
An 2 Seria B Grupa 30227
Facultatea de Automatica si Calculatoare
Sectia Calculatoare si Tehnologia Informatiei

1.Obiectivul temei

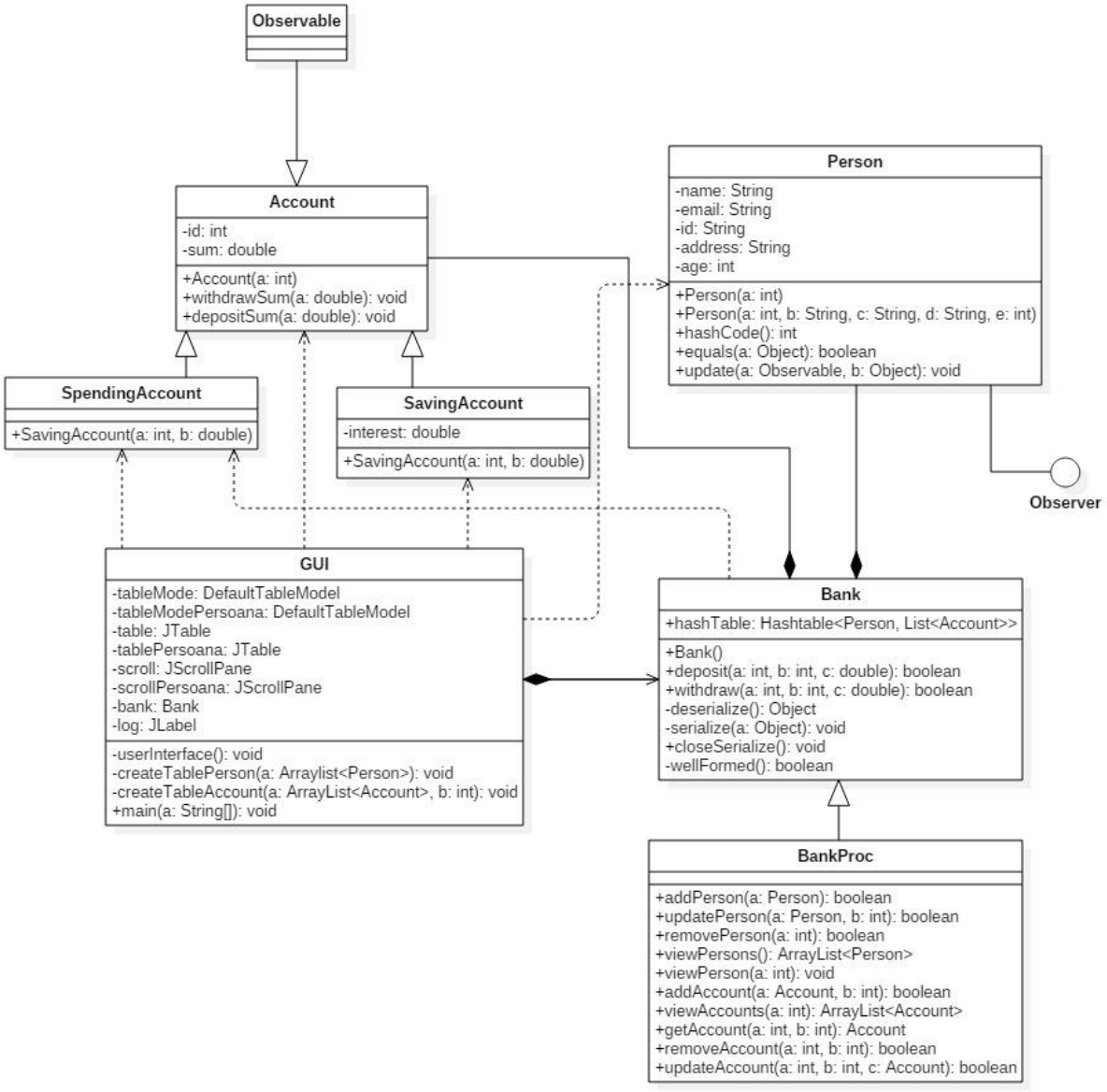
Obiectivul principal al temei este de a crea o aplicatie care se ocupa cu managementul unei banci, care este capabila de a memora persoane dar si conturile acestora, memorand date despre persoane si detalii despre conturi, cum ar fi tipul contului, suma si dobanda daca este cazul.

Obiectivele secundare sunt reprezentate de realizarea unei interfete grafice pentru a executa operatiile dorite, afisarea detaliilor persoanelor inregistrate la banca, afisarea detalii depspre conturile unei persoane dar si de a realiza extrageri si depuneri in conturile dorite.

2. Analiza problemei, asumptii, modelare, scenarii, cazuri de utilizare, erori

Aceasta aplicatie trebuie sa fie capabila sa simuleze cat mai bine o banca. Astfel, ea trebuie sa fie in stare sa prelucreze date despre persoanele inregistrate la banca, dar si despre conturile pe care aceasta le detine. Trebuie sa adauge sau sa elimine conturi si persoane, sa fie in stare sa depuna sau sa scoata banii din conturi si sa proceseze suma scoasa daca extragerea se face dintr-un cont de economii.

3. Proiectare (decizii de proiectare, diagrame UML, structuri de date, proiectare clase, interfete, relatii, packages, algoritmi, interfata utilizator, modul de tratare a erorilor)



Diagramade clase de mai sus descrie impelemntarea aplicatiei. Pentru realizarea aplicatiei am folosit mai multe patternuri si anume: Observer care are rolul de a notifica detinatorul unui cont bancar atunci cand s-a realizat o actiune asupra contului, adica o operatiune de extragere sau depunere a unor banii intr-un anumit cont. De asemenea, pe parcursul dezoltarii aplicatiei am folosit Design

by Contract care este folosit pentru a detecta anumite probleme care presupun folositea unor metode gata implementate corect, dar care depind de niste parametrii care se presupun a fi corecti din punctul de vedere al metodei. Pentru a face diferenta intre contul de economii si contul de cheltuieli, am folosit o clasa abstracta Account care retine suma si id-ul contului, iar cele doua tipuri de conturi extind clasa Account, acestea diferentiindu-se prin existenta dobanzii la SavingAccount. Pentru stocarea datelor am folosit tehnica de serializare care se bazeaza pe citirea, la inceputul aplicatiei a datelor deja existente, dintr-un fisier, iar la inchiderea aplicatiei salvarea datelor deja existente in acelasi fisier.

4. Implementare

Clasa UI va construi o fereastra compusa din mai multe campuri. Primul camp care este prezent tot timpul pe fereastra se refera la selectarea categoriei cu care se doreste a se lucra: Panoul de persoane care va deschide panoul cu optiuni care se refera la persoane, panoul de conturi care va deschide panoul cu optiuni care se refera la conturi si panoul de operatii, care se refera la optiuniile care se pot efectua asupra conturilor, si anume depunerea de bani sau extragerea de bani. In momentul in care una din optiuniile de mai sus este selectata va aparea pe fereastra un nou set de operatiuni care se pot efectua asupra categoriei selectata, iar in functie de ce operatiune este selectata, vor aparea sau nu campuri de introducere a datelor, dar si un tabel care va prezenta utilizatorului rezultatul unei cautari. Atunci cand se detecteaza iesirea din aplicatie, toate datele rezultate in urma operatiuniilor vor fi salvate intr-un fisier „Banca” care este scris cu ajutorul serializarii.

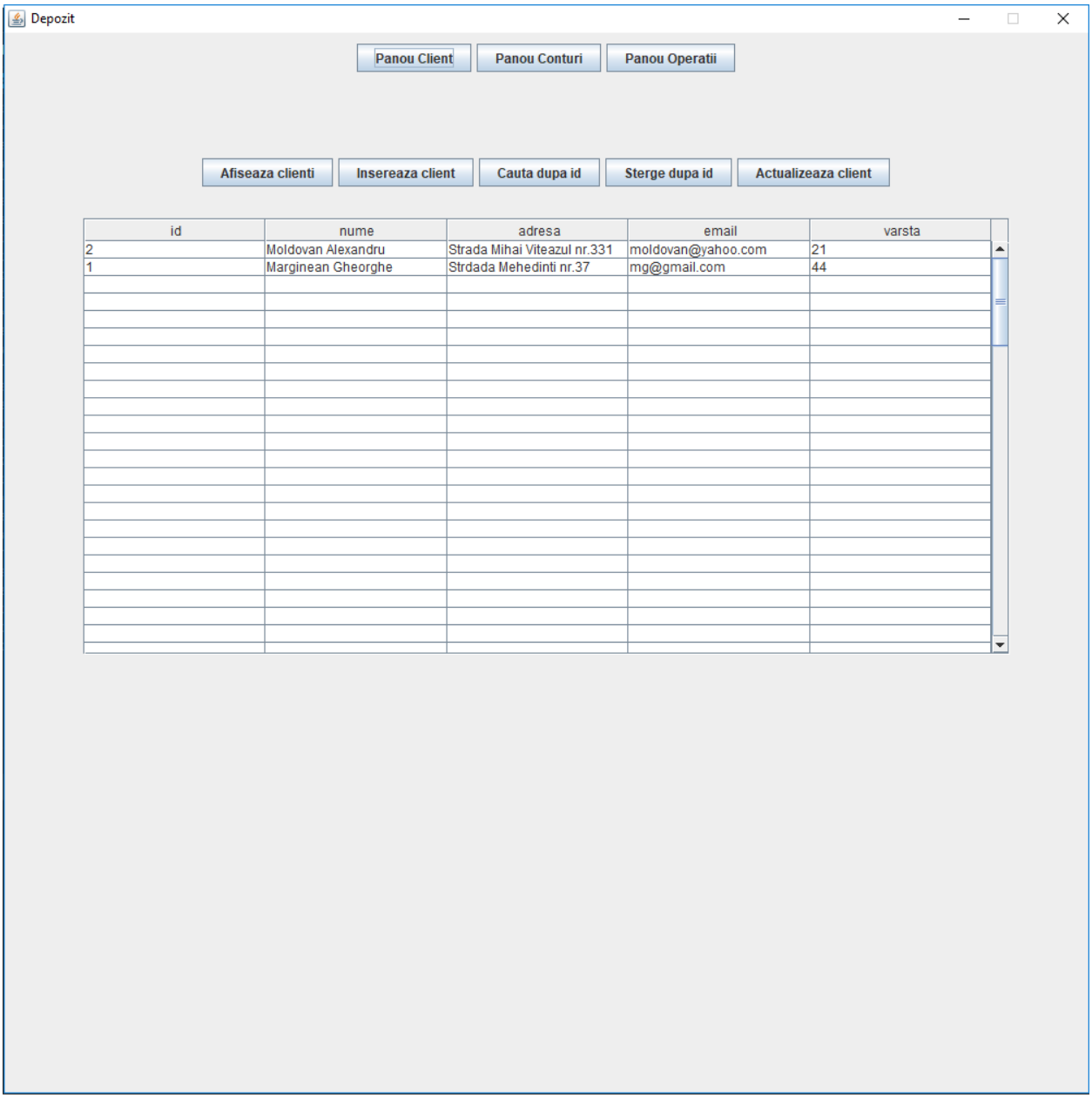
Clasa Person reprezinta, in aplicatia prezentata detinatorul unuia sau mai multor conturi. Acesta este identificat cu ajutorul unui id unic, iar campurile name, address, email si age descriu mai multe detalii despre persoana facand identificarea acesteia mai usoara. Aceasta clasa implementeaza interfata Observer, un lucru care ne permite sa identificam usor atunci cand se efectueaza o operatiune asupra unuia dintre conturile unei anumite persoane.

Clasele de conturi, SpendingAccount si SavingAccount reprezinta, in aplicatia prezentata, contul care apartine unui singur detinator. Acestea doua extind clasa abstracta Account, astfel identificarea se face cu ajutorul unui id unic pentru fiecare persoana, ceea ce inseamna ca pot exista in banca 2 conturi cu

acelasi id, dar conturile trebuie sa apartina unor persoane diferite. Clasa account extinde clasa Observable, ceea ce ne permite ca atunci cand se detecteaza o actiune asupra unuia dintre conturi, detinatorul contului sa fie anuntat despre acest eveniment.

Clasa Bank implementeaza practic banca care este simulata in aceasta aplicatie. Pentru a retine toate aceste date despre persoane si conturile lor, am optat pentru folosirea unui Hashtable, cheia unei intrari fiind reprezentata de o persoana, astfel clasa Person are suprascrisa metoda de hashCode, hashCode-ul fiind reprezentat acum de id-ul persoanei, iar pentru a identifica daca exista doua persoane sunt identificate la fel, desi ele sunt diferite, a trebuit suprascrisa metoda equals, astfel cand avem doua persoane cu acelasi id, acestea sa fie identificate ca fiind aceleasi. Clasa Bank permite adaugarea, stergerea sau actualizarea de persoane si conturi, de a afisa persoanele inregistrate si a conturilor lor, de a depozita sau de a retrage bani din conturi.

5. Testare/Rezultate



Pentru testarea programului am folosit o unitate de testare Junit in care am folosit folosit metodele de adaugare, stergere si actualizare a conturilor si a persoanelor si de a retrage sau depune bani in conturi. Din screenshot-ul de mai sus se observa interfata grafica. Aceasta este compusa dintr-o fereastra compusa la randul ei din mai multe campuri. Primul camp care este prezent tot timpul pe fereastra se refera la selectarea categoriei cu care se doreste a se lucra: Panoul de persoane care va deschide panoul cu optiuni care se refera la persoane, panoul de conturi care va deschide panoul cu optiuni care se refera la conturi si panoul de operatii, care se refera la optiuniile care se pot efectua asupra conturilor, si anume depunerea de bani sau extragerea de bani. In momentul in care una din optiuniile de mai sus este selectata va aparea pe fereastra un nou set de operatiuni care se pot efectua asupra categoriei selectata, iar in functie de ce operatiune este selectata, vor aparea sau nu campuri de introducere a datelor, dar si un tabel care va prezenta utilizatorului rezultatul unei cautari. Atunci cand se detecteaza iesirea din aplicatie, toate datele rezultate in urma operatiuniilor vor fi salvate intr-un fisier „Banca” care este scris cu ajutorul serializarii. Atunci cand este selectata optiunea Afiseaza clienti, toate persoanele inregistrate la banca vor fi afisate, impreune cu id, nume, adresa, email si varsta. Daca este selectata optiunea Insereaza clienti, vor aparea niste textfield-uri in care se vor introduce datele pentru un nou client. In cazul in care datele furnizate de catre utilizator sunt invalide, se va afisa un mesaj de eroare „Date invalide”. In cazul in care datele furnizate sunt corecte din punct de vedere al tipului si respecta constrangerile impuse de aplicatie, dar din cauza ca elementul descris exista deja in tabela de dispersie, se va afisa un mesaj care il va informa pe utilizator ca operatiunea nu a putut fi efectuata. In cazul unui succes, un mesaj de succes va fi prezentat pe interfata, iar persoana descrisa de utilizator va fi inserata in tabela de dispersie dupa apasarea butonului executa. Pentru cautarea dupa id va aparea un textfield in care se va introduce un id. In cazul in care datele furnizate de catre utilizator sunt invalide, se va afisa un mesaj de eroare „Date invalide”. In cazul in care datele furnizate sunt corecte din punct de vedere al tipului si respecta constrangerile impuse de aplicatie, Pe fereastra se va descrie un tabel care va prezenta persoana cauta cu toate detaliile sale. Stergerea dupa id este la fel ca partea descrisa mai sus. Se introduce un id, iar daca persoana cu id-ul respectiv exista in tabela, aceasta intrare in tabela de dispersie va fi eliminata, iar in caz de succes se va afisa un mesaj pe ecran. Pentru actualizarea unui client, se introduc

datele aferente clientului cu id-ul prezentat in campul corespunzator. Id-ul din camp trebuie sa corespunda cu unul deja prezent in tabela, altfel actualizarea nu va putea fi realizata.

6. Concluzii

Aceasta aplicatie este potrivita pentru o banca care vrea sa tina evidenta asupra clientilor sai dar si asupra conturilor aferente. Se poate tine evidenta asupra operatiunilor pe care detinatorul contului le savarste asupra contului, tinand minte suma de bani extrasa dar si id-ul persoanei care a realizat operatiunea. Din aceasta tema am invatat sa folosesc serializarea, o metoda de a citi si scrie date intr-un fisier dupa tipul de date pe care aplicatia le foloseste, Design by Contract care m-a invatat sa dezvolt mai sigur aplicatiie si de a elimina erorile unor secvente de cod care folosesc niste metode care functioneaza pe anumite seturi de date si care elimina tactica de a programa defenisv. Pe langa aceasta am invatat sa folosesc si tipul de date observer, care ne ajuta sa identificam in aplicatie evenimente care au loc asupra unor obiecte si de a fi in stare sa tratam aceste evenimente.