# ReadsProfiler

Popescu Georgian Ștefan, A7
georgianspopescu@gmail.com

Facultatea de Informatică Iași

**Rezumat** ReadesProfiler reprezinta o aplicatie minimalista ce ofera acces la cautarea si descarcarea simpla de carti gratuite.

## 1 Introducere

ReadsProfiler este o aplicatie client-server care ofera acces la o librarie online. Clientul va avea un cont la care se autentifica prin nume de utilizator si parola. Utilizatorii aplcatiei vor fi de doua tipuri: obisnuiti si administratori. Libraria va contine carti din diverse genuri si subgenuri.

Aplicatia ofera functionalitatea cautarii unei carti dupa atributele acesteia: titlu, autori, gen, an, ISBN, rating, fiind posibila specificarea mai multe filtre de cautare simultan. De asemenea, clientii vor putea oferi un rating unei carti(cuprins intre 1-foarte slab si 5-excelent), descarca cartea si vizualizarea istoricului cautarilor si descarcarilor.

Serverul ii va oferi recomandari de carti clientului pe baza gusturilor acestuia si a preferintelor altor clienti cu aceleasi gusturi.

## 2 Tehnologii utilizate

Aplicatia va utiliza tehnologia TCP din urmatoarele motive:

1. stabilirea unei conexiuni intre client si server: autentificarea utilizatorului;
2. conexiune full-duplex: comunicarea dintre client(utilizator) si server(librarie) se bazeaza pe un schimb de mesaje de tipul cerere-raspuns
3. transmiterea sigura a pachetelor intre clienti si server: siguranta ca utilizatorul va primi in intregime raspunsul de la server, descarcarea cartilor.

Servirea concurenta si rapida a clientilor va fi efectuata de un server care foloseste thread-uri.

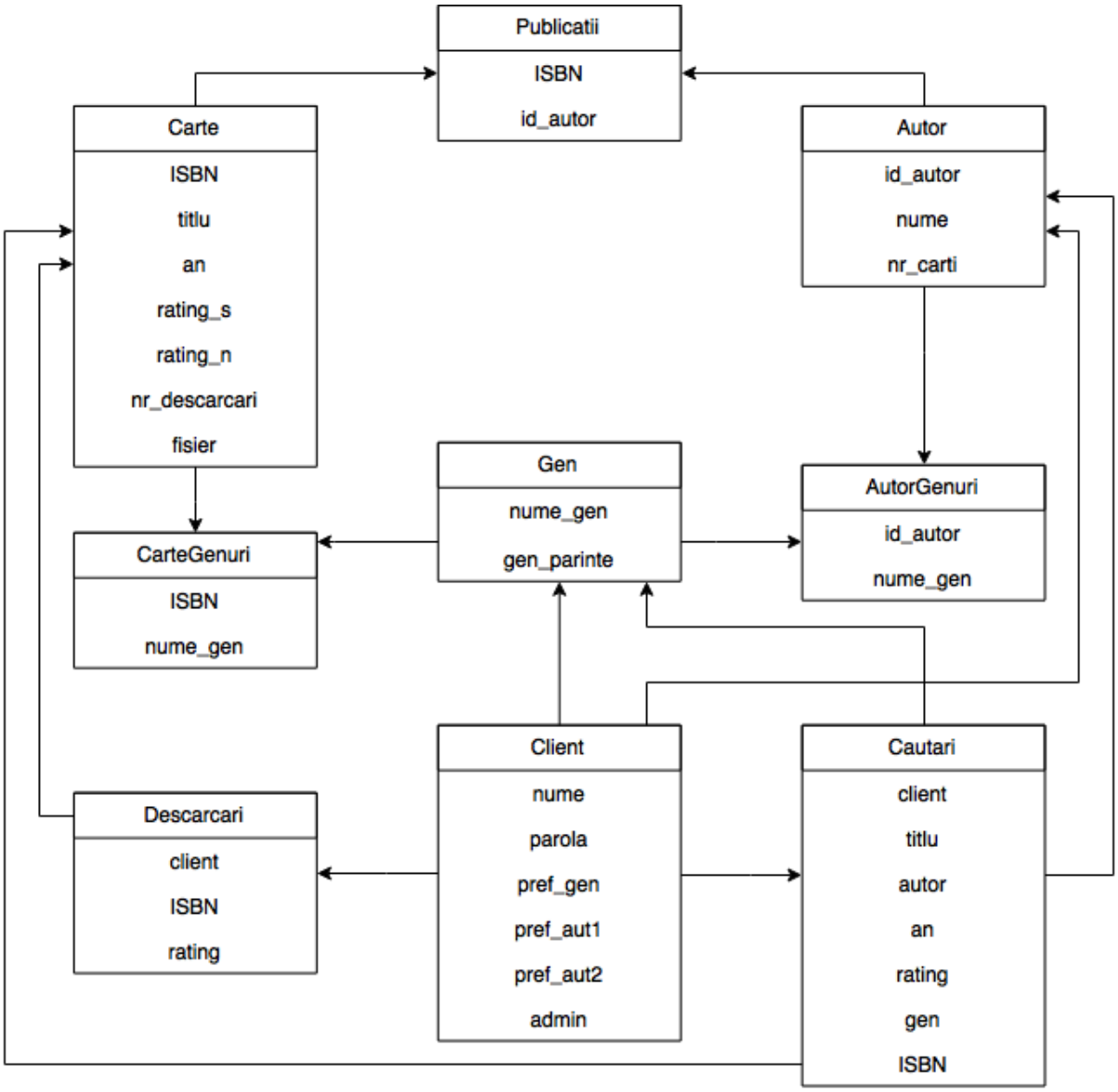
Toate informatiile pe care le utilizeaza serverul vor fi stocate intr-o baza de date relationala. Aceasta are rolul de a simplifica cautarile.

## 3 Arhitectura aplicatiei

### 3.1 Stocarea datelor

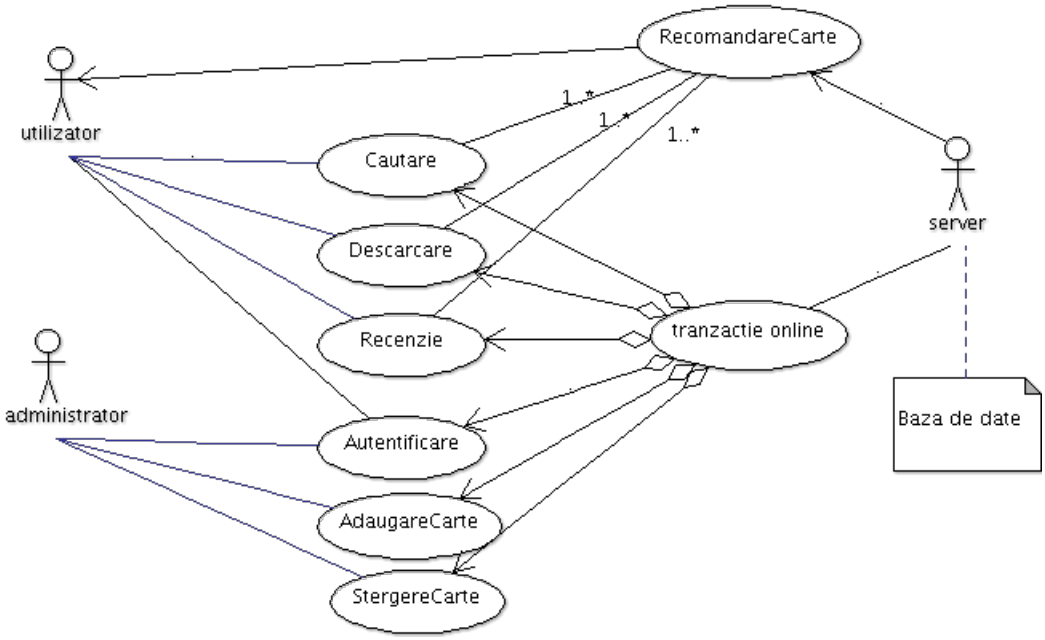
Baza de date contine tabelele *Carte*, *Autor* si *Genuri*, care vor avea ca tebele intermediare de legatura *Publicatii*, *CarteGenuri* si, respectiv, *AutorGenuri*.

In doua tabele se vor retine informatii despre cautarile si descarcarile clientilor, iar in tabela *Client* vom stoca datele de autentificare, 2 genuri preferate si 2 autori preferati pentru fiecare utilizator, informatii ce ne sunt utile pentru gestiunea sistemului de recomandari.



### 3.2 Scenariu de utilizare

Comunicarea dintre client si server se va desfasura dupa urmatorul scenariu:




### 3.3 Formatul liniei de comanda

1. nume_comanda // login |cauta |descarca |review |istoric |exit
2. argument_comanda

Exemple corecte:

```
[input] cauta
[output] Introdu ISBN.
[input] ALL
[output] Introdu titlu sau 'ALL'/ENTER pentru toate titlurile.
[input] ALL
[output] Introdu autorii despartiti prin virgula sau 'ALL'/ENTER pentru toti autorii.
[input] Liviu Rebreanu
[output]  Introdu genurile despartite prin virgula sau 'ALL'/ENTER pentru toate.
[input] universala,istorie
[output] Introdu subgenurile despartite prin virgula sau 'ALL'/ENTER pentru toate.
[input] beletrictica
[output] Introduceti anul aparitiei sau 'ALL'/ENTER pentru toti anii.
[input] ALL
[output] Introduceti ratingul minim sau 'ALL'/ENTER.
[input] ALL
```

```
[output] Ion - Liviu Rebreanu,universala,beletristica,2010,9786064301277
         Ciuleandra - Liviu Rebreanu,universala,beletristica,2012,4786064301279
```

### 3.4 Formatul pachetului

```
unsigned int       length
char               type
char[length - 1]   data payload
```

Unde type este:

```
l = login
u = adauga_user
c = cauta
d = descarca
r = review
a = adauga_carte
s = sterge_carte
h = istoric
p = recomanda
n = raspuns normal de la server
e = eroare
x = logout
t = exit
```

Continutul payload-ului difera in functie de tipul de serviciu:

```
type            payload
l               nume_utilizator parola
u               nume_utilizator parola 0|1
c               ISBN|ALL$titlu|ALL$nume_autor1[,nume_autor2,...]|ALL$
                gen1[,gen2,...]|ALL$subgen1[,subge2,...]|ALL$an|ALL$rating|ALL
d               ISBN
r               ISBN rating
a               ISBN$titlu$an$nume_autor1,[nume_autor2,...]$id_aut1,[id_aut2,...]$
                gen1[,gen2,...]$subgen1[subgen2,...]$fisier
s               ISBN
h               -
p               -
n               raspuns
e               descriere_eroare
x               -
```

### 3.5 Proiectarea si implementarea modelului client/server TCP

Serverul va fi implementat folosind tehnologia TCP – prethreaded, cu blocare pentru protectia *accept*(). Serverul creeaza un numar de thread-uri cand este pornit si aceastea vor servi clientii. Pentru ca doar un singur thread sa apeleze *accept* la un moment dat, se va folosi *mutex lock* pe apelul primitivei *accept*. Serverul va tine evidenta numarului de thread-uri active(care deservesc un client) si va creea noi thread-uri cand numarul clientilor se apropie de numarul total al thread-uilor create. In momentul cand un astfel de thread nou creat termina de deservit clientul, el isi va termina executia, anuntand procesul principal.

Clientul va fi implementat folosind tehnologia TCP. Aplicatia-client ii cere utilizatorului sa se autentifice si apoi, in cazul unei acceptari din partea serverului, clientul va putea introduce comenzie la care are acces. Utilizatorul va putea introduce alta comanda dupa primirea raspunsului de la server. Incheierea sesiunii se va face prin comanda *exit*.

### 3.6 Proiectarea sistemului de recomandari

```
    void Recomandare::recomanda(Client *client, char rezultat[2000])
{
    char rezultat1[1500];
    char gen1[20], gen2[20], aut1[20], aut2[20];
    client->getPrefGen1(gen1);  //genul preferat din tabela Cautari
    client->getPrefGen2(gen2);  //genul preferat din tabela Descarcari
    client->getPrefAut1(aut1);  //autorul preferat din tabela Cautari
    client->getPrefAut2(aut2);  //autorul preferat din tabela Descarcari

    char *sql = new char[1500];
    /*crearea selectului pentru gasirea cartilor scrise de unul din autorii
    preferati, ce apartin unuia din genurile preferate ce au rating > 2 si
    grupate si ordonate dupa titlu,autor,gen,numar de descarcari
    (primele 6 rezultate)*/
    CreareSelectPentruPreferinte(sql);

    BazaDate *gg = new BazaDate();
    gg->cautareCarte(sql,rezultat1);

    char alt_client[300];
    /*crearea selectului pentru gasirea altor clienti care au in comun cu
    utilizatorul un autor si un gen preferat*/
    CreareSelectPentruAltClient(alt_client);

    char *sql2 = new char[1500], rezultat2;
    /*crearea selectului pentru gasirea cartilor descarcate de 'alt_client'
    cu rating > 2 si ordonarea dupa numarul de descarcari
```

```
    (primele 6 rezultate)*/
    CreareSelectPentruPreferinteComune(sql2,alt_client);

    gg->cautareCarte(sql2,rezultat2);

    strcpy(rezultat,"Carti recomandate in functie de preferintele dvs. :");
    strcat(rezultat,rezultat1);
    strcat(rezultat,"Carti descarcate si apreciate de utilizatori" \
            "cu gusturi similare:");
    strcat(rezultat,rezultat2);
}
```

## 4 Concluzii

Aplicatia ofera functionalitatile de baza ale unei librarii online. Modul de implementare a serverului ofera servirea rapida, concurenta si dinamica a clientilor, iar folosirea tehnologiei TCP asigura siguranta transferului de date.

Odata cu cresterea numarului de utilizatori, numarul de cautari poate creste exponential. Folosirea unor *functii de hash* pentru gestionarea cautarilor ar reprezenta o metoda mai eficienta de implementare in acest caz.

## Bibliografie


[1.] https://profs.info.uaic.ro/ computernetworks/files/7rc_ProgramareaInReteaIII_Ro.pdf
[2.] https://www.ietf.org/rfc/rfc4253.txt
[3.] https://www.tutorialspoint.com/sqlite/sqlite_c_cpp.htm