Aplicații Integrate pentru Întreprinderi

Tema 2

Distribuirea pe componente a unei aplicații folosind tehnologia RMI

Lansare	Termen de predare	Pondere
08.11.2011, 23:55	22.11.2011, 23:55	10 / 70

Obiective

Scopul laboratorului constă în realizarea unui sistem informatic performant pentru gestiunea resurselor unei întreprinderi de dimensiuni mari. În scop didactic dar şi pentru a vă ajuta în procesul de analiză a sistemului ERP, instituția pentru care se va dezvolta aplicația nu face parte din sectorul comercial ci din spațiul academic – cu care se presupune că sunteți mai familiari.

Aplicația va fi implementată gradual prin intermediul unor teme de casă, pe parcursul a patru etape.

În cadrul acestei etape, sistemul informatic va fi modularizat în mai multe componente specializate pe anumite operații, fiind capabile să comunice în cadrul unei rețele de calculatoare prin care sunt conectate.

După rezolvarea temei de casă, studentul va fi capabil să:

- proiecteze o aplicație având o structură distribuită pe componente putând rula pe maşini diferite;
- modeleze comunicația între componentele distribuite ale aplicației.

Cunoştințele necesare[1] pentru rezolvarea temei de casă sunt:

- programarea în limbajul Java;
- manipularea bazelor de date folosind MySQL;
- folosirea API-ului Java DataBase Connectivity;
- utilizarea tehnologiei RMI pentru apelarea unor metode aparținând unor obiecte existente în altă maşină virtuală.

Enunț

Funcționalitatea sistemului informatic pentru facilitarea interacțiunii dintre personalul Universității „Politehnica” din Bucureşti şi studenți a fost distribuită în mai multe componente:

- componente de tip server
  - Arhiva
  - RegistruActivitățiDidactice
- componente de tip client
  - secretar
  - cadru didactic
    - şef de catedră
  - student

[1] Se poate utiliza orice limbaj de programare, sistem de gestiune pentru baze de date şi tehnologie pentru apelul de metode la distanță.

Accesul la tabelele din baza de date va fi restricționat pentru fiecare componentă în parte astfel:

- **Arhiva** va putea accesa doar datele din tabelele `catalog` şi `disciplina`;
- **RegistruActivităţiDidactice** va putea accesa doar datele din tabelele `utilizatori`, `orar` şi `programare_examene`;
- celelalte componente nu vor avea acces la informații decât prin metodele implementate de componentele **Arhiva** şi **RegistruActivităţiDidactice**.

1. **Arhiva** gestionează informațiile referitoare la disciplinele de învățământ cât şi la notele obținute de studenți la examene. Astfel, vor fi implementate metode care vor permite specificarea unui plan de învățământ de către un şef de catedră, stabilirea notei pentru o disciplină pentru un student (de către un cadru didactic) dar şi obținerea ei (de către un student), cât şi determinarea situației şcolare prin calculul mediei generale şi al numărului de puncte credit.

2. **RegistruActivităţiDidactice** gestionează informațiile despre calendarul activităților didactice de pe parcursul unui an universitar. Acesta va permite şefului de catedră să stabilească repartizarea unui cadru didactic pentru un tip de activitate didactică aferente unei discipline de învățământ, secretarului stabilirea formațiilor de studiu, specificarea orarului şi a programării examenelor iar studentului afişarea informațiilor cu privire la activitățile didactice la care trebuie să participe.

3. Un **şef de catedră**

a. va stabili planul de învățământ, accesând tabela `disciplina` prin intermediul componentei **Arhiva**:

```
public int adaugareDisciplina(ArrayList<Disciplina> discipline);
public int editareDisciplina(ArrayList<Disciplina> discipline);
public int stergereDisciplina(ArrayList<Disciplina> discipline);
```

Metodele vor întoarce valori de tip int prin care se va specifica numărul operațiilor realizate cu succes.

Clasa `Disciplina` va trebui implementată având toate câmpurile precizate în tabela corespunzătoare din baza de date, conținând metode de tip set/get. Datorită faptului că obiecte de acest tip vor fi transmise între componente rulând în maşini virtuale diferite, clasa `Disciplina` trebuie să fie **serializabilă**.

b. realizează repartizarea unui cadru didactic pentru un tip de activitate corespunzător unei discipline de învățământ, prin intermediul componentei **RegistruActivităţiDidactice**:

```
public boolean stabilesteRepartizareCadruDidactic
                (int CNPCadruDidactic,
                 int codDisciplina,
                 int tipActivitateDidactica);
```

Metoda va întoarce o valoare de tip `boolean` prin care se va specifica dacă operația a fost realizată sau nu cu succes.

4. Un **cadru didactic** va specifica nota obținută de un student la un examen prin intermediul componentei **Arhiva**.

```
public boolean stabilesteNota(int CNPCadruDidactic,
                              int codDisciplina,
                              int CNPStudent,
                              int nota,
                              Calendar data);
```

Componenta **Arhiva** verifică dacă există permisiuni pentru cadrul didactic să stabilească nota pentru respectivul student. În cazul în care mai există o nota specificată pentru perechea `(CNPStudent, codDisciplină)`, aceasta va fi suprascrisă doar dacă nota existentă era mai mică decât 5 (restanță) sau dacă valoarea nouă a notei este mai mare decât valoarea veche a notei (mărire).

Operația va eşua în cazul când nu există drepturi de acces, sau se încearcă suprascrierea unei note având o valoare mai mică.

5. Un **secretar** foloseşte serviciile componentei **RegistruActivitățiDidactice**:

a. stabileşte grupa din care face parte un grup de studenți (formații de studiu):

```
public int stabilesteFormatieDeStudiu
           (ArrayList<Integer> CNPStudent,
            int grupa);
```

Rezultatul arată numărul de studenți pentru operația de stabilire a grupei s-a realizat cu succes. Operația poate eşua dacă nu există un student cu CNP-ul specificat în baza de date.

b. stabileşte orarul:

```
public int adaugareActivitateDidactica
           (ArrayList<Orar> activitatiDidactice);
public int editareActivitateDidactica
           (ArrayList<Orar> activitatiDidactice);
public int stergereActivitateDidactica
           (ArrayList<Orar> activitatiDidactice);
```

Clasa `Orar` va trebui implementată având toate câmpurile precizate în tabela corespunzătoare din baza de date, conținând metode de tip set/get. Perechile `(codDisciplina, grupa, tipActivitate)` sunt unice în tabela `orar`. Datorită faptului că obiecte de acest tip vor fi transmise între componente rulând în maşini virtuale diferite, clasa `Orar` trebuie să fie **serializabilă**.

c. stabileşte calendarul de examene:

```
public int adaugareExamen(ArrayList<Examen> examene);
public int editareExamen(ArrayList<Examen> examene);
public int stergereExamen(ArrayList<Examen> examene);
```

Clasa `Examen` va trebui implementată având toate câmpurile precizate în tabela corespunzătoare din baza de date, conținând metode de tip set/get. Perechile `(codDisciplina, grupa)` sunt unice în tabela `programare_examene`. Datorită faptului că obiecte de acest tip vor fi transmise între componente rulând în maşini virtuale diferite, clasa `Examen` trebuie să fie **serializabilă**.

6. Un **student**:

a. obține nota pentru una sau mai multe discipline prin intermediul componentei **Arhiva**:

```
public ArrayList<Integer> obtineNota
                          (int CNPStudent,
                           ArrayList<Integer> codDisciplina);
```

În situația în care pentru o disciplină nu există notă trecută în catalog, rezultatul întors va fi -1.

b. obține situația şcolară pentru anul universitar curent prin intermediul componentei **Arhiva**:

```
public SituatieScolara obtineSituatieScolara(int CNPStudent);
```

Clasa `SituatieScolara` va trebui implementată, conținând informații despre media generală (calculată ca medie aritmetică, respectiv ponderata), numărul de puncte credit şi numărul de examene care nu au fost promovate[2] (restanțe). Se vor implementa metode setter/getter. Clasa va fi serializabilă întrucât instanțe ale ei vor fi transmise între maşini virtuale diferite.

c. obține orarul activităților didactice pentru semestrul în curs prin intermediul componentei **RegistruActivitățiDidactice**:

```
public ArrayList<Orar> obtineOrar(int CNPStudent);
```

d. obține calendarul examenelor pentru sesiunea în curs prin intermediul componentei **RegistruActivitățiDidactive:**

```
public ArrayList<Examen> obtineProgramareExamene(int CNPStudent);
```

Nu uitați că trebuie stabilite disciplinele opționale şi facultative specificate prin contractul de studii, atât orarul cât şi programarea examenelor incluzând şi aceste informații.

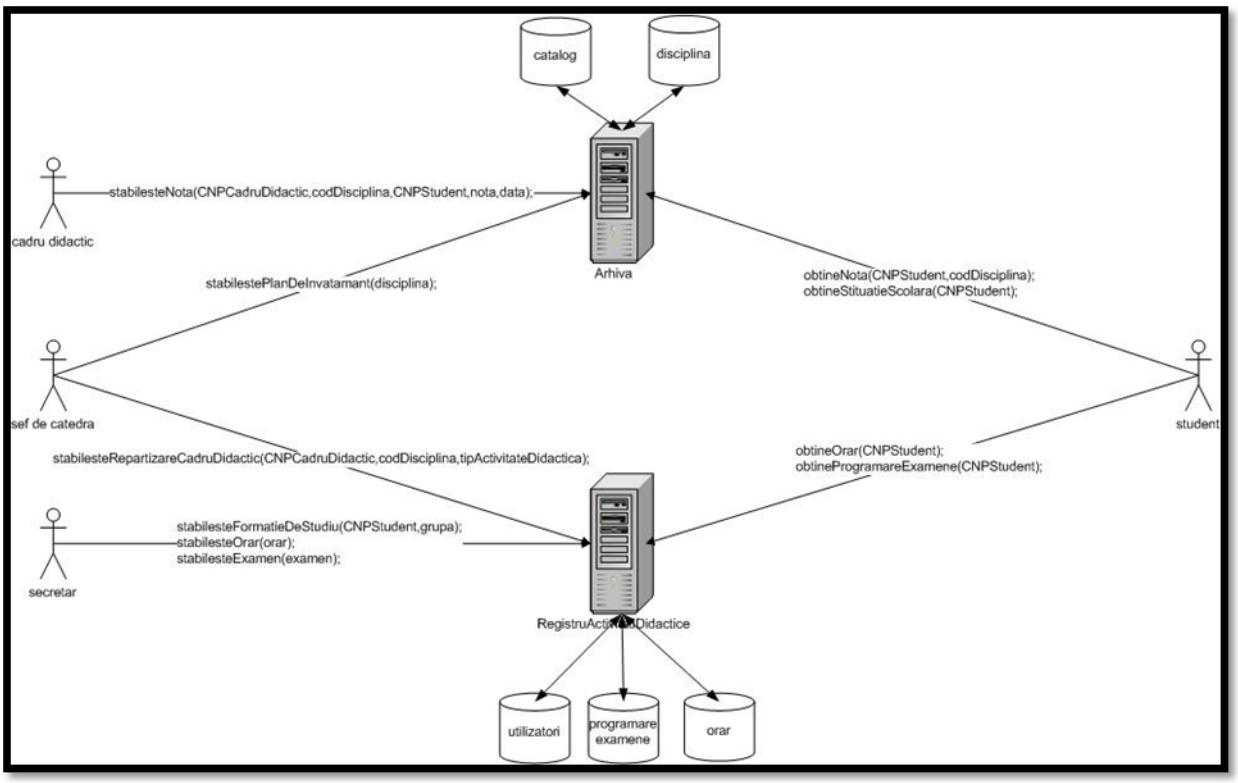
Implementarea metodelor puse la dispoziție de componentele de tip server trebuie să mențină proprietatea de integritate pentru baza de date. În acest sens se vor realiza toate verificările necesare[3]. Dacă accesul la tabele nu este permis din cadrul unei componente, se vor implementa metode pentru obținerea informațiilor necesare prin metode apelate din contextul acelor componete care dețin drepturi asupra tabelei respective.

Astfel, apelul unei metode la distanță poate implica alte apeluri la distanță încât fluxul de informații se poate face inclusiv între componentele de tip server.

[2] În situația în care contractul de studii nu a fost reținut prin tabela catalog, trebuie identificate disciplinele de învățământ existente în contractul de studii pentru care nu există note trecute, acestea fiind considerate restanțe.

[3] Spre exemplu, pentru metoda `stabilesteNota` apelată de un cadru didactic prin intermediul componentei Arhiva, se va verifica dacă există `CNPCadruDidactic`, `CNPStudent` şi `codDisciplina` şi dacă acel cadru didactic predă disciplina indicată la grupa din care face parte studentul.

O perspectivă asupra aplicației distribuite pe componente se poate observa în figura de mai jos:



Arhitectura aplicației distribuite pe componente

Precizări suplimentare

Se consideră că baza de date a fost populată corespunzător cu utilizatori, astfel încât nu mai este necesară implementarea unei funcționalități referitoare la gestiunea acestora (adăugare / editare / ştergere utilizatori de către administrator / super-administrator), respectiv implementarea unor operații referitoare la datele personale. Fiecare utilizator (componentă de tip client) specifică atunci când aplicația este lansată în execuție utilizatorul şi parola pentru autentificarea în sistemul informatic[4].

O componentă de tip server NU va stabili prin politica de securitate contextul din care este apelată o metodă pe care o implementează[5].

După cum veți observa pe parcursul implementării temei, arhitectura descrisă nu cuprinde toate metodele necesare interacțiunii dintre componente. Proiectarea altor metode puse la dispoziție de componentele de tip server, nespecificate în enunț, se consideră decizie de implementare.

[4] Unele metode transmit ca parametru utilizatorul pentru a verifica drepturile pentru realizarea unei anumite operații.

[5] Astfel, pentru componenta **Arhiva** nu este obligatoriu să se realizeze verificarea faptului că metoda `stabilesteNota` este apelată de către un cadru didactic, `stabilestePlanDeInvatamant` este apelată de către un şef de catedră sau `obtineNota/obţineSituaţieŞcolară` sunt apelate de student; la fel, pentru componenta **RegistruActivităţiDidactice** nu este obligatoriu să se realizeze verificarea faptului că metoda `stabilesteRepartizareCadreDidactice` e realizată de şef de catedră, `stabilesteOrar/stabilesteExamen/stabilesteFormatieDeStudiu` sunt apelate de către un secretar iar `obţineOrar/obţineProgramareExamene` sunt apelate de către un student.

Metodele descrise mai sus NU trebuie să aibă în mod obligatoriu semnătura care a fost precizată, aceasta putând fi adaptată conform implementării de la tema precedentă însă având aceeaşi funcționalitate[6].

Nu este necesar să implementați o interfață grafică pentru această temă, parametrii putând fi preluați din linia de comandă, din fişier sau de la tastatură.

Distribuirea bazei de date se va face doar în sensul restricționării accesului la tabele de către unele componente şi NU folosind un sistem de gestiune pentru baze de date distribuite sau prin stabilirea unor profile cu drepturi de acces asupra unor tabele.

Barem de corectare

Punctaj	Criterii de acordare
1,5 p	proiectarea interfețelor corespunzătoare componentelor de tip server • componenta Arhiva • componenta RegistruActivitățiDidactice
3 p	implementarea metodelor corespunzătoare componentei Arhiva • stabilesteNota [cadru didactic] • stabilestePlanDeInvatamant [şef de catedră] • obtineNota, obtineSituatieScolara [student]
3 p	implementarea metodelor corespunzătoare componentei RegistruActivitățiDidactice • stabilesteRepartizareCadruDidactic [sef de catedra] • stabilesteFormatieDeStudiu, stabilesteOrar, stabilesteExamen [secretar] • obțineOrar, obțineProgramareExamene [student]
1 p	apelarea corespunzătoare a metodelor din componentele de tip server din componentele de tip client
0,5 p	respectarea proprietății de integritate a bazei de date • asigurarea consistenței informațiilor reținute în tabele / mesaje de eroare
1 p	modularizare • structura aplicației • comentarii, README

BONUS. Se pot obține punctaje suplimentare (care se vor cumula cu punctajele obținute la celelalte teme), astfel:

- 0,25 p – predarea temei cu cinci zile mai devreme (17.11.2011, 23:55);
- 0,50 p – dezvoltarea unei intefețe grafice pentru componentele de tip client ținând cont de restricțiile asupra accesului la tabelele din baza de date;
- 0,25 p – studenții în an terminal care au promovat toate examenele pot solicita foaia matricolă, ce conține lista tuturor disciplinelor de învățământ ordonate pe ani de studiu, precizându-se mediile (aritmetică şi ponderată) şi numărul de puncte credit pentru fiecare an universitar; disciplinele sunt descrise prin toate detaliile (cod, denumire, tip, numar ore, puncte credit, forma examinare, an studiu, semestru).

[6] Spre exemplu, metoda `stabileşteNota` poate avea şi următoarea semnătură:

```
public int stabilesteNota(int CNPCadruDidactic,
                          int codDisciplina,
                          ArrayList<Integer> CNPStudent,
                          ArrayList<Integer> nota,
                          Calendar data);
```

specificând printr-un singur apel notele tuturor studenților care au participat la examenul aferent unei discipline, susținut cu cadrul didactic respectic, la o anumită dată.

Condiții de realizare şi predare

Tema va fi realizată individual. Dacă nu ați implementat tema precedentă, puteți folosi cod sursă dezvoltat de un coleg (cu permisiunea acestuia), precizând acest lucru în README. În această situație, prezentarea temei e obligatorie. Trebuie să puteți explica şi funcționalitatea codului sursă folosit de la coleg.

La fiecare laborator puteți realiza o anumită parte a temei:

- Laborator 06 – proiectarea interfețelor din componentelor tip server după identificarea operațiilor ce trebuie puse la dispoziție de acestea;
- Laborator 07 – implementarea metodelor specificate în interfețele componentelor tip server (Arhiva şi RegistruActivitățiDidactice);
- Laborator 08 – utilizarea corespunzătoare a metodelor la distanță din componentele tip client (şef de catedră, cadru didactic, secretar, student).

După prezentarea temei, va trebui să încărcați pe site o arhivă de tip .zip (cu denumirea Grupa34XCX_NumePrenume_Tema2.zip) care să conțină script-ul cu operațiile asupra bazei de date[7], sursele aplicației cu exemple de utilizare (lansări în execuție a componentelor de tip server şi a componentelor de tip client cu diferite operații, specificând proprietățile Java necesare[8] şi fişierele conținând politica de securitate), precum şi un README în care să prezentați succint atât deciziile de implementare şi tehnologiile folosite.

Temele vor fi comparate prin diferite aplicații de verificare a plagiatului spre a se depista fraudele. În această situație, punctajul pe parcursul semestrului va fi anulat, urmând să fiți obligați să repetați disciplina anul universitar următor.

[7] Acest lucru este obligatoriu chiar dacă a constituit o cerință şi pentru tema precedentă.

[8] Este vorba despre `java.rmi.server.codebase`, `java.rmi.server.hostname`, `java.security.policy`.