Metode de învăţământ din perspectiva formării competenţei de programare orientată pe obiecte

Silviu GÎNCU,
doctorand,
Universitatea de Stat din Tiraspol

Andrian CRÎŞMARU,
profesor de informatică,
Liceul Nicolae Iorga

Rezumat

În lucrare sunt prezentate metode de învăţământ propuse pentru aplicare în procesul de formare şi dezvoltare a competenţei de programare orientată pe obiecte.

Abstract

This paper is presented some frequently teaching methods used for developing object-oriented programming competence.

Un rol important în formarea de competenţe îl au metodele de învăţământ. Selectarea, adaptarea şi îmbinarea acestora reprezintă o metodă eficientă de a ajuta instruitul (elev, student, etc.) să dobândească noi cunoştinţe, să-şi dezvolte noi abilităţi şi în final să dobândească noi competenţe. Metodele de învăţământ, după I. Cerghit [1] se clasifică în:

I. METODE DE COMUNICARE

1.1. ORALE:

1. *Expozitive*: descrierea, explicaţia, prelegerea, etc.

2. *Conversative*: conversaţia, discuţia didactică, problematizarea, etc.

1.2. SCRISE

1.3. ORAL – VIZUALE

1.4. COMUNICAREA INTERIOARĂ

II. METODE DE EXPLORARE:

2.1. DIRECTE: observaţia didactică, lucrări experimentale, studiul de caz, etc.

2.2. INDIRECTE (demonstrative): demonstraţia obiectelor reale, modelarea, etc.

III. METODE DE ACŢIUNE:

3.1. REALĂ: exerciţii, lucrări practice, elaborarea de proiecte, etc.

3.2. SIMULATĂ

IV. METODE DE RAŢIONALIZARE: instruirea programată, algoritmizarea, etc.

Practica demonstrează că studenţii întâmpină diverse dificultăţi în formarea competenţei de programare orientată pe obiecte. Aceste impedimente pot fi înlăturate prin aplicarea adecvată a metodelor de învăţământ în procesul educaţional. Ca şi oricare alt domeniu studiat, programarea orientată pe obiecte, are latura sa specifică, cât şi o latură care este comună tuturor disciplinelor ce ţin de domeniul informaticii. Aşa cum programarea orientată pe obiecte are un caracter aplicativ, în prezenta lucrare vom reliefa acele metode şi procedee didactice care facilitează participarea conştientă şi activă a studentului la învăţare.

Problematizarea mai poate fi denumită şi predare prin rezolvare de probleme.

Conform acestei metode, instruitul este pus în fața unor dificultăți create în mod deliberat și prin depășirea lor învață ceva nou. „Punctul forte" al metodei îl constituie *situația-problemă*. Din această cauză este necesar de a crea corect situația-problemă și de a ține cont de următoarele caracteristici:

A. Situația trebuie să prezinte o dificultate pentru instruit, iar pentru determinarea soluției, acesta se va confrunta cu efort mental;
B. Situația trebuie să prezinte interes, astfel încât, studentul să acționeze spre rezolvarea acesteia;
C. Situația trebuie să orienteze activitatea studentului spre a rezolva problema și de a-l cointeresa să dobândească noi cunoștințe;
D. Rezolvarea situației nu va fi posibilă fără a apela la resursele recent dobândite.

Prin intermediul situației create, studentul este cointeresat de a participa activ la rezolvarea problemei. Aplicarea acestei metode presupune parcurgerea a patru etape:

1. *Formularea problemei* – este descrisă situația-problemă, explicată, pentru a permite studentului să perceapă problema;
2. *Analiza problemei* – se lucrează în mod independent, sunt reactualizate anumite resurse;
3. *Determinarea soluției* – în cadrul acestei etape sunt selectate resursele necesare, se descoperă mijloacele care conduc la rezolvarea problemei și este analizat modul de aplicare a acestora în determinarea soluției;
4. *Obținerea rezultatului final* – se analizează rezultatul obținut și formulate anumite concluzii.

Exemplu. Această metodă poate fi aplicată cu succes la predarea temei „*Polimorfismul dinamic*". Pentru a utiliza situația-problemă, care urmează a fi creată este necesar a fi studiate noțiunile de moștenire și modalitățile de operare cu pointeri într-o ierarhie de clase care sunt aflate în relație de moștenire.

Formularea problemei: De la monitor se citesc datele despre *n* angajați din instituția *X*. În cadrul instituției activează angajați precum *administratori*, *contabili* și *operatori*. Elaborați un program prin intermediul căruia se vor efectua operații de:

– citire și extragere a datelor (despre fiecare categorie de angajați se cunosc date diferite);
– sortare a datelor după nume, funcție și salariu (fiecare categorie de salariat va fi determinat după o formulă specifică);
– determinare a sumei necesare pentru achitarea salariului tuturor angajaților;
– afișare a angajatului cu cel mai mare salariu.

Analiza problemei: conform datelor problemei se observă că aceasta poate fi rezolvată prin utilizarea a trei tablouri. Totuși, această cale nu este optimă, deoarece nu am putea efectua operații precum sortarea și totodată ar fi mai complicat de realizat și celelalte aspecte ale problemei. Acest exemplu creează o situație în care instruitul este nevoit să apeleze la propriile resurse. După o analiză a acestora se va ajunge la concluzia că este necesar de a crea o ierarhie de clase care să se afle în relație de moștenire.

Determinarea soluției: Soluția va apare atunci când va fi studiată noțiunea de polimorfism și modalitatea de operare cu metodele virtuale într-o relație de moștenire.

Obținerea rezultatului final: Rezultatul obținut va fi un program care operează cu o structură dinamică (de tip listă) sau statică (tablou unidimensional), unde în calitate de elemente sunt obiecte de tip *administrator*, *contabil* sau *operator*.

Printre avantajele problematizării vom menționa:

1. Mobilizează grupul de lucru, atenţia fiind concentrată spre a determina soluţia problemei.
2. Formează un stil activ de muncă.
3. Asigură motivaţia pentru învăţare.
4. Oferă încrederea în sine.

Modelarea este „o metodă de cercetare sau învăţare a obiectelor, fenomenelor (legi, principii, norme), care constă în folosirea unui model construit pe baza proprietăţilor esenţiale ale originalului. Modelul (care facilitează învăţarea) este un mod de materializare a generalului, a ansamblului de la care se pleacă în descoperirea elementelor particulare; reproduce simplificat trăsăturile şi caracteristicile obiectelor şi fenomenelor, dificil de perceput şi cercetat, în mod direct". [7, p.35]. Prin intermediul acestei metode se pot reda, prin analogie, diverse situaţii, raţionamente, care pot să reprezinte relaţii dintre obiecte, fenomene, procese, etc.

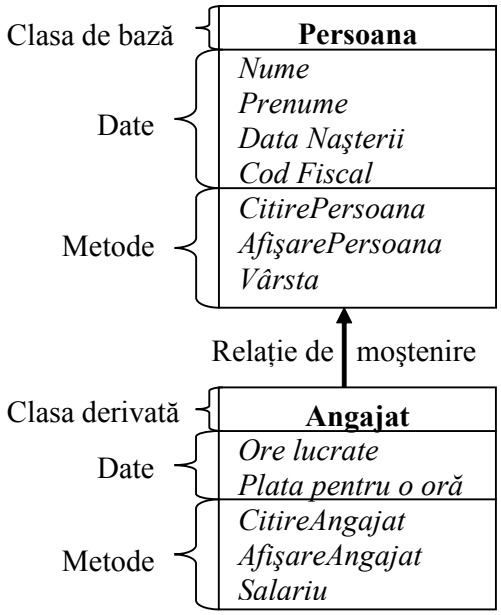
Metoda modelării didactice este cu succes utilizată în domeniul tehnologiei informaţionale. În 1997 G.Booch, I.Jacobson şi J.Rumbaugh au lansat un limbaj de modelare grafică UML (Unified Modeling Language). După Martin Fowler „UML reprezintă o familie de notaţii grafice, la baza cărora se află un singur meta-model. Acesta este predestinat pentru proiectarea şi elaborarea de produse software, în special a celor orientate pe obiecte" [6, p.27]. Conform definiţiei „UML este un limbaj de modelare bazat pe notaţii grafice folosit pentru a specifica, vizualiza, construi şi documenta componentele unui program".[4, p.11].

Limbajul de modelare grafică UML conţine un set de 9 tipuri de diagrame, printre care: *class diagram* (diagrame de clase) şi *object diagram* (diagrame de obiecte). Prin intermediul diagramelor pot fi uşor reprezentate relaţiile dintre clase, respectiv obiecte.

Exemplu. Această metodă poate fi aplicată cu succes la predarea temei *Moştenire*. Se poate utiliza modelarea pentru a reprezenta relaţiile dintre 2 clase: *Persoană* şi *Angajat*.

Pentru a reprezenta relaţiile dintre aceste două clase în limbajul C++, vom scrie:

```
class Persoana{
protected:
char *nume,*prenume;
char *cod_fiscal;
int anul_nasterii;
public:
void CitirePersoana();
void AfisarePersoana();
int Varsta();
};
class Angajat : public Persoana{
int ore,pl_ora;
public:
void CitireAngajat();
void AfisareAngajat();
double Salariu();
};
```



Relaţia dintre aceste două clase poate fi prezentată şi prin intermediul diagramelor UML

După cum observăm, prin intermediul diagramelor mesajul didactic este transmis

mai uşor şi mai rapid. Din punctul nostru de vedere este important ca mai întâi să fie înțeles conceptul de moştenire, apoi şi modul de implementare a conceptului prin intermediul unui limbaj de programare orientat pe obiecte.

Metoda **decompoziției** presupune a descompune o situație (problemă) complicată în mai multe situații mai simple, iar rezolvarea fiecărei situații în parte duce implicit la rezolvarea problemei în ansamblu. Această metodă este aplicată cu succes pentru rezolvarea de probleme în limbajul de programare orientat pe obiecte. Conform DEX „decompoziția este un procedeu de determinare prin analiză sau prin calcul a caracteristicilor unui obiect". În programare termenul de decompoziție este întâlnit în formele:

• decompoziția algoritmică;

• decompoziția pe obiecte. Acest proces presupune crearea mai multor obiecte, iar prin intermediul relațiilor dintre acestea se obțin o serie de avantaje caracteristice programării orientate pe obiecte.

Punerea în aplicare a decompoziției va permite studenților să se concentreze pe analiza şi crearea claselor, precum şi pe anumite relații dintre ele.

La aplicarea decompoziției se recomandă a ține cont de următoarele cerințe de definire a unei clase:

1. structura obiectului trebuie să fie clară;

2. obiectul nu trebuie să includă în structura sa (la nivel conceptual), mai multe obiecte;

3. obiectul trebuie să fie "complet", adică să conțină îndeajuns date şi metode pentru prelucrarea acestuia.

Decompoziția va fi efectuată astfel încât obiectele create vor avea în definirea lor o structură clară, în conformitate cu sarcina dată, şi totodată creată o relație bine definită dintre celelalte obiecte. Prin decompoziție, studenții învață a analiza şi a crea obiecte, în consecință vor justifica atât alegerea claselor, cât şi a relațiilor dintre acestea.

Crearea unui obiect începe prin determinarea tipului şi a entităților (datele şi metodele). Inițial nu ar trebui să stabilească anumite entități, ele vor fi stabilite în procesul de analiză a problemei. Astfel la stabilirea entităților unui obiect se va ține cont de următoarele cerințe:

1. Obiectul trebuie să aibă minimum caracteristicile unui tip de date predefinit:

a. domeniu de valori;

b. set de operații admisibile.

2. Obiectul va satisface o serie de cerințe:

a. va consta dintr-o serie de câmpuri, care corespund proprietăților datelor descrise, ce permit descrierea domeniului de valori;

b. va consta dintr-o serie de operații, care corespund operațiilor admise asupra datelor;

c. accesul la câmpuri va fi realizat doar prin intermediul unor operații, care vor forma interfața tipului.

Metoda decompoziției poate fi aplicată cu succes pentru integrarea resurselor specifice diferitor situații. Pentru a forma competențe vor fi analizate, rezolvate mai multe situații, iar prin intermediul acestei metode pot fi aduse exemple care ar oferi posibilitatea de a analiza noi situații care vor include în structura sa probleme deja analizate sau rezolvate. Pentru integrarea situațiilor specifice polimorfismului static şi a relației de agregare poate fi propusă următoarea problemă.

Exemplu. Să se creeze un tip de date, care ar permite operarea cu tipul de date matrice, elementele cărora sunt numere complexe. Prin intermediul acestui tip de date se vor determina elementele matricei D, dacă: D=(A+B)*(C-A*B), A;B;C – matrice de ordinul 3, iar elementele sunt numere complexe.

Analizând acest exemplu, se va ajunge la concluzia că este necesar de a crea o clasă prin

intermediul căreia să operăm cu numere complexe. Pentru crearea acestui tip, se vor supraîncărca operatorii necesari tipului complex.

```
class complex{
double x,y;//complex=x+yi
public:
friend complex operator+(complex, complex);
friend complex operator*(complex, complex);
friend complex operator-(complex,complex);
friend complex operator/(complex, complex);
complex operator= (complex);
};
```

Clasa Matrice:

```
class Matrice{
complex m[3][3];
public:
friend Matrice operator + (Matrice, Matrice);
friend Matrice operator* (Matrice, Matrice);
friend Matrice operator-(Matrice, Matrice);
Matrice operator = (Matrice);
};
```

Conceptul de agregare este prezent prin declararea unui tablou bidimensional de numere complexe, astfel clasa matrice este formată din mai multe obiecte de tip complex.

Metoda **studiul de caz** valorifică o situaţie reală care se analizează şi se rezolvă. „Ea reprezintă o modalitate de apropiere a procesului de învăţare de modelul vieţii, al practicii, având o mare valoare euristică şi aplicativă”[2, p.96]. Aşa cum problemele rezolvate în stilul orientat pe obiecte au un grad sporit de dificultate, sunt cazuri când este necesar de a prezenta studentului probleme deja rezolvate. Avantajul metodei, constă în faptul că fiecare student va contribui la analiza şi rezolvarea problemei. În utilizarea acestei metode se conturează câteva etape:

1. Selectarea şi prezentarea cazului;
2. Organizarea echipelor de lucru;
3. Prelucrarea şi conceptualizarea;
4. Structurarea finală a studiului.

Exemplu. Ca şi în cazul decompoziţiei această metodă va fi aplicată cu scopul integrării resurselor formate pentru diferite situaţii specifice. Pentru aceasta se va propune următoarea problemă:

1. Selectarea şi prezentarea cazului

Să se elaboreze un program prin intermediul căruia se va monitoriza evidenţa personalului dintr-o instituţie. În instituţia dată sunt următoarele tipuri de personal:

Angajat – despre care se cunosc datele: nume, prenume, codul fiscal, numărul de ore lucrate, plata pentru o oră. În calitate de metode vor fi aplicate: *scrierea* datelor de la tastatură, afişarea datelor la ecran, determinarea salariului, conform formulei: *salariu=ore_lucrate*plate_ora*.

Angajat de bază – despre care se cunosc datele: nume, prenume, codul fiscal, numărul de ore lucrate, plata pentru o oră, gradul (0,1,2,3), anul angajării. În calitate de metode vor fi aplicate: *scrierea* datelor de la tastatură, afişarea datelor la ecran, determinarea salariului, conform formulei: *salariu=ore_lucrate*plate_ora*. Angajatul va beneficia şi de un adaos la salariu. Salariaţii cu gradul 0 vor avea un adaos de 50%, gradul 1 – 40%, gradul 2 – 30%. În dependenţă de stagiu, adaosul va fi de 35% pentru cei cu un stagiu $\geq$ 10 ani, 25% pentru cei cu un stagiu $\geq$ 3 ani, şi 10 % pentru toţi ceilalţi.

Student – despre care se cunosc următoarele date: nume, prenume, codul fiscal, grupa, media. În calitate de metode vor fi utilizate: *scrierea* datelor de la tastatură, afişarea datelor la ecran, determinarea bursei. Studentul nu va avea bursă dacă va avea o medie mai mică ca 7.5, va primi 300 lei dacă bursa e mai mică ca 8.5, 400 lei dacă e mai mică ca 9.5 şi 500 în caz contrar.

Student Angajat – despre care se cunosc datele: nume, prenume, codul fiscal, grupa, media, numărul de ore lucrate, plata pentru o

oră. În calitate de metode vor fi aplicate: *scrierea* datelor de la tastatură, afişarea datelor la ecran, determinarea bursei şi a salariului. Metodele *bursa* şi *salariul* vor fi determinate conform formulelor din clasele precedente.

Referitor la personalul instituției vor fi efectuate operații de adăugare a unei persoane, excludere, afişare a tuturor persoanelor din instituție, precum şi determinarea tuturor banilor ce trebuie achitați pentru tot personalul. La ecran va fi afişat un meniu, prin intermediul căruia va fi prezentate informația despre personal.

2. Organizarea echipelor de lucru

Pentru rezolvarea acestei probleme va fi necesar de a crea şapte clase. Studenții se împart în şapte grupe. Grupurilor formate li se propune spre analiză câte o clasă. Ei analizează structura acestora pentru a o explica colegilor.

3. Prelucrarea şi conceptualizarea

Fiecare grup va explica structura fiecărei clase. Profesorul va monitoriza lucrul studenților, pentru însuşirea modului de rezolvare a problemei.

Programului propus pentru analiză (fără definirea metodelor clasei):

```
#define an 2010
class Persoana{
 protected:
 char nume[10],prenume[10],idnp[13];
 public:
 virtual void citire();
 virtual void afisare();
 virtual double salariu(){return 0;}
 virtual double bursa(){return 0;}
};
class Angajat:virtual public Persoana{
 public:
 int ore;
 double pl_ora;
 void citire();
 void afisare();
 double salariu();
};
class Ang_baza:public Angajat{
public:
int an_ang,grad;
 void citire();
 void afisare();
 double salariu();
};
class Student:virtual public Persoana{
public:
char grupa[8];
double media;
 void citire();
 void afisare();
 double bursa();
};
class Stud_Ang:public Student,public Angajat{
public:
 void citire();
 void afisare();
};
class celula{
public:
Persoana *p;
celula *next;
celula(){next=NULL;}
void cit();
void afis();
double consum();
};
class lista{
 celula *prim;
 public:
 lista(){prim=NULL;}
 public:
 void creare();
 void afisare();
 void inserare();
 void exclude();
 double bani();
 ~lista();
};
```

4. *Structurarea finală a studiului*

Programul este prezentat studenţilor. Se execută pentru vizualizarea rezultatului. Studenţii prezintă avantajele rezolvării acestei probleme prin stilul orientat pe obiecte comparativ cu alte stiluri cunoscute.

Instruirea prin proiecte reprezintă „o modalitate de instruire/autoinstruire graţie căreia elevii, dar mai ales studenţii, efectuează o cercetare orientată spre obiective practice şi finalizată într-un produs ce poate fi un obiect, un aparat, o instalaţie, o culegere tematică, un album, o lucrare ştiinţifică etc.”[3, p.191] Elaborarea unui proiect presupune parcurgerea mai multor etape. Diferiţi cercetători evidenţiază de la trei până la opt etape în realizarea proiectului. În [5] sunt prezentate diferite modele de elaborare a proiectelor (Tabelul 1).

Tabelul 1

Etapele de elaborare a unui proiect în varianta cercetătorilor din S.U.A.	**Etapele de elaborare a unui proiect în varianta cercetătorilor din Europa**
(a) evidenţierea problemei, formularea sarcinii; (b) discutarea variantelor cercetării, selectarea procedeelor; (c) autoinstruirea şi actualizarea cunoştinţelor; (d) repartizarea obligaţiilor; (e) cercetarea: rezolvarea unor sarcini; (f) generalizarea rezultatelor; (g) analiza succeselor şi a greşelilor; (h) corectarea sau trecerea la un alt proiect.	(a) pregătirea; (b) planificarea; (c) cercetarea; (d) rezultatele şi/sau concluziile; (e) prezentarea sau raportul; (f) aprecierea rezultatelor şi a procesului.

În varianta noastră etapele de realizare a unui proiect sunt:

1. Familiarizarea. Această etapă se caracterizează prin actualizarea cunoştinţelor necesare şi prezentarea cerinţelor faţă de proiect.

2. Structurarea. În cadrul acestei etape sunt prezentate exemple de proiecte, împreună cu studenţii sunt elaborate proiecte similare, astfel încât studentul să fie capabil să selecteze informaţii necesare pentru realizarea propriului proiect.

3. Aplicarea. Aceasta este etapa când studentul elaborează proiectul, îl prezintă şi este evaluat pentru munca depusă.

4. Reflecţia. Este o etapă în care sunt analizate proiectele elaborate, sunt precizate cazurile în care urmează a fi utilizat proiectul. Este foarte important ca orice proiect elaborat, să fie utilizat şi după evaluarea acestuia.

Exemplu. Această metodă poate fi aplicată cu succes la elaborarea de aplicaţii pentru gestiunea bazelor de date.

Familiarizare

- Elaboraţi un proiect prin intermediul căruia se automatizează procesul de lucru în întreprinderea X.
- Actualizare cunoştinţe: baze de date, Paleta ADO, Data Controls, Data Aces, etc.

Structurare

- Elaborarea unui proiect pentru evidenţa cărţilor unei biblioteci
- Selectarea informaţiei în domeniul proiectului

Aplicare

- Pentru elaborarea proiectului studentul va crea formulare de inserare, excludere, modificare, exagerare, căutare, etc.
- Prezentarea
- Evaluarea

Reflecţie

- Analiza
- Utilizarea proiectului pentru teza de diplomă/licenţă, utilizarea acestuia în întreprinderea X

Metoda proiectului ca formă de instruire este o soluție la noile cerințe din domeniul tehnologiilor informaționale cât şi a evoluției societății contemporane. Un pas important în formarea profesională a viitorului specialist îl realizăm prin intermediul proiectului. După realizarea acestuia, studentul obține o viziune mai clară despre viitoarea lui specialitate.

Referințe bibliografice:

1. Cerghit, I. *Metode de învățământ*, E.D.P., Bucureşti,1999
2. Miron Ionescu, Vasile C. *Strategii de predare şi învățare*, Editura Ştiințifică, Bucureşti, 1992
3. Stan Panțuru, Daniela Necşoi, *Teoria şi metodologia instruirii*, Brasov, 2007;
4. Грейди Буч, Джеймс Рамбо, Айвар Джекобсон, UML. *Руководство пользователя* Издательство: ДМК, 2001 г. Pag. 432;
5. Гузеев В.В. *Проектное обучение как одна из интегральных технологий*. În: Метод проектов. Выпуск 2 / Белорусский государственный университет. Центр проблем развития образования. Республиканский институт высшей школы БГУ. Минск: РИВШ БГУ, 2003. 240 с. ISBN: 985-6684-55-2;
6. Фаулер М. UML. Основы / Пер. с англ. – 3-е изд. – СПб: Символ-Плюс, 2004. – 192 pag.;
7. http://www.docstoc.com/docs/46539614/metode-de-instruire-si-mijloace-de-învatamânt

Recenzent: Ion ACHIRI, doctor în ştiințe fizico-matematice, conferențiar universitar, IŞE