UDK 347.235.11:681.332.2
Originalni znanstveni članak

OBJEKTNO ORIJENTISANI SISTEM U UPRAVLJANJU PODACIMA KATASTRA NEKRETNINA

Zdravko GALIĆ — Sarajevo*

SAŽETAK: U radu je prikazan pristup projektovanju i implementaciji prototipa sistema za upravljanje podacima katastra nekretnina, zasnovanog na objektno orijentisanoj tehnologiji baza podataka. Prikazane su i neke osnovne karakteristike objektno orijentisane programske okoline baza podataka MODULEX. Samu aplikacionu okolinu karakterišu objekti kompleksne strukture, reprezentovani kao veliki perzistentni skupovi, i akcije čija je kompleksnost određena kompleksnošću samih objekata. Pored toga, prikazan je značaj objektno orijentisanog pristupa u projektovanju i realizaciji ovako složenog softverskog sistema (apstrakcija podataka, ekstenzibilnost, nasljeđivanje i polimorfizam).

1. UVOD

Jedan od očekivanih praktičnih rezultata tematske oblasti »Projektovanje proizvoda i tehnologija uz pomoć računara«, Društvenog cilja IX »Široka distribucija i primjena sistema produktike u različitim oblastima (Produktika)« jeste i razvoj prototipa objektno orijentisanog sistema za upravljanje podacima katastra nekretnina. Pod ovim se pojmom podrazumijeva računarska podrška u upravljanju podacima katastra nekretnina zasnovana na objektno orijentisanom sistemu za upravljanje bazama podataka. Naime, većinu raspoloživog softvera koji se koristi za potrebe izrade i održavanja katastra nekretnina u Bosni i Hercegovini karakteriše neki od sljedećih nedostataka:

— potpuno ignorisanje tehnologije baza podataka
— upotreba konvencionalnih programskih jezika (COBOL, PL/I, itd.)
— problemi očuvanja podataka u konzistentnom stanju (narušavanje uslova integriteta, iznenadni prestanak rada sistema, itd.)
— nemogućnost uporednog pristupa podacima od strane više korisnika
— zanemarivanje grafičke manipulacije i reprezentacije podataka.

* Mr. Zdravko Galić, Građevinski fakultet, Sarajevo, Hasana Brkića 24.

Ovaj rad je rezultat istraživanja koje se odvija u okviru DC—IX, TO—1, NP—2, finansiranog od strane SIZ-a nauke BiH.

Da bi jedan takav sistem prevazišao ovakve nedostatke i što bolje odgovarao ne samo trenutnim zahtjevima već omogućavao integraciju ovih podataka u informacione sisteme o prostoru, odlučeno je da se implementira:

— iskorištavanjem objektno orijentisane tehnologije baza podataka
— na radnoj stanici pod operativnim sistemom UNIX
— efikasna podrška grafičkoj manipulaciji i reprezentaciji podataka.

1.1. Zašto objektno orijentisani sistem baza podataka?

Dosadašnji razvoj tehnologije za upravljanje bazama podataka nije bio vezan za potrebe inženjerskih i prostornih aplikacionih okolina, tako da primjena ovih novih tehnologija nije na onom nivou koji se ostvaruje u konvencionalnim aplikacijama. Postojeći model podataka, upitni jezici, kao i fizička reprezentacija samog modela, projektovani su i realizovani sa ciljem da se omogući modeliranje, odnosno reprezentacija aplikacionih okolina čiji su relevantni objekti u strukturnom smislu jednostavni. Ta strukturna jednostavnost podrazumijeva da atributi tih objekata imaju kao svoje domene neke od prostih tipova podataka podržanih samim modelom. Zbog toga, postojeći modeli podataka (kao što je npr. relacioni) omogućavaju adekvatno i efikasno upravljanje velikim skupovima podataka u standardnim aplikacionim okolinama (Guting 1989).

Međutim, prostorni objekti su često složene strukture, i pored nekih standardnih atributa mogu npr. da imaju i geometrijske atribute koji nisu prosti, i nisu podržani samim modelom. (Parcela je ipak prostorni objekt, čiji je geometrijski oblik predstavljen poligonom.) Ukoliko je i moguće izvršiti strukturno modeliranje takvih objekata u okviru relacionog modela, sam strukturni dio modela u principu je kompleksan i teško razumljiv za korisnike takvih objekata. Jer u procesu modeliranja neminovno dolazi do dekompozicije takvih objekata u velik broj relacija, a to opet ima za posljedicu i drastičan pad vremenskih performansi, zbog toga što je za rekompoziciju takvih objekata neophodno izvršiti velik broj spajanja relacija (Alagić, Galić 1990).

Objektno orijentisani modeli već zauzimaju značajno mjesto u razvoju novih generacija modela pogodnih za modeliranje kompleksnih aplikacionih okolina. Takvi su pristupi zasnovani na njihovoj pogodnosti za reprezentaciju kompleksnih objekata, konceptualnoj jednostavnosti, kao i usklađenosti sa jasno izraženim trendovima u programskim jezicima i softverskom inženjerstvu. I kao što se objektno orijentisano programiranje u programskim jezicima omogućava njihovim proširenjima (Object Pasal (Tesler 1985), C^{++} (Stroustrup 1986), Objective-C (Cox 1986)), veoma je značajno da se u evoluciji objektnih sistema iskoriste sve prednosti relacionog modela (Beech 1988).

Iako ne postoji univerzalna usaglašenost o tome šta zapravo predstavlja pojam *'objektno orijentisan'*, takav sistem baza podataka trebalo bi da karakterišu (Bancilhon 1990):

— kompleksni objekti
— identifikatori objekata
— učaurenje (encapsulation)
— tipovi ili klase
— nasljeđivanje (inheritance)
— ekstenzibilnost.

Zbog svega toga, kao i zbog raspoloživosti objektno orijentisanog sistema baza podataka MODULEX* proširenog relacionog tipa, u realizaciji sistema za upravljanje podacima katastra nekretnina upotrijebljen je upravo taj sistem.

2. MODULEX OBJEKTNO ORIJENTISANA PROGRAMSKA OKOLINA BAZA PODATAKA

Programska okolina MODULEX (Alagić 1989) rezultat je usklađivanja relacionog i objektno orijentisanog pristupa u sjedinjenoj multiparadigmatskoj okolini, koja podržava oba pristupa na konzistentan način. Tu okolinu čine objektno orijentisani definicioni, upitni i manipulacioni jezik visokog nivoa, kao i programski jezik baza podataka zasnovan na objektno orijentisanom proširenju programskog jezika Modula-2 (Wirth 1983). U okviru proceduralnog jezika, proširenje relacionog modela ostvareno je uvođenjem koncepta modula (kao što je definisan u Moduli-2) na vrh tog modela i obezbjeđenjem drugih standardnih apstrakcija iz tog programskog jezika.

Osnovno proširenje u odnosu na jezik Modula-2 jeste uvođenje koncepta velikih skupova, čiji elementi ne moraju biti prosti (ENTITY SET), odnosno relacija sa pridruženim akcijama koje omogućavaju unošenje i brisanje elemenata tih skupova. U skladu sa takvim pristupom, pravila za komponovanje akcija su proširena kompozicijom FOREACH, koja omogućava specifikaciju upita relacionog tipa i akcije za ažuriranje nad skupovima.

Objekti se uniformno reprezentuju kao moduli, bez obzira na njihovu prirodu. Sa aspekta notacije programskog jezika, nema razlike između objekata koji su skupovi slogova (relacija) i drugih standardnih i nestandardnih objekata kao što su I/O, poligon, itd. Kompleksni objekti, posebno oni dobiveni nekim od standardnih apstrakcija, kao što su agregacija, generalizacija, pokrivanje i rekurzija, reprezentovani su sa više skupova entiteta (relacija), međutim ta reprezentacija je skrivena od korisnika. Korisnicima stoji na raspolaganju samo specifikacija osobina objekata i akcija nad njima.

Odvajanje specifikacije objekta od njegove implementacije (reprezentacije), što predstavlja fundamentalnu osobinu objektno orijentisanog pristupa (Atkinson 1989), ostvareno je definicionim i implementacionim modulom. Pored toga, konceptom modula se ostvaruje još jedna osobina objektno orijentisanog pristupa: učaurenje (encapsulation) osobina objekta i njemu pridruženih akcija. U definicionom modulu su specificirani samo njegovi atributi i akcije koje je moguće izvršiti nad primjercima tog tipa objekta. Akcije se specificiraju navođenjem njihovih imena i parametara tih akcija, što se ostvaruje specifikacijom naziva procedure i njenih parametara, s obzirom na to da je to sve što korisnik mora da zna u cilju pozivanja tih procedura. Tako je definicioni modul objekta zapravo interfejs za sve korisnike tog objekta. Definicioni modul ima sljedeću formu:

```
DEFINITION MODULE naziv_objekta;
    specifikacija atributa objekta;
    specifikacija naziva i parametara akcija (procedura);
END naziv_objekta.
```

* MODULEX je praktički rezultat naučnoistraživačkog projekta finansiranog od strane Američko-jugoslovenskog Savjeta za naučnu i tehnološku suradnju (NSF ugovor JFP 708).

Drugi, niži nivo apstrakcije specificiran je u implementacionom modulu objekta. Ovaj modul sadrži aktuelnu reprezentaciju skupa objekata i dekompoziciju akcija, kao procedura koje operišu nad tom reprezentacijom:

```
IMPLEMENTATION MODULE naziv_objekta;
    aktuelna reprezentacija skupa objekata;
    dekompozicija akcija nad izabranom reprezentacijom;
END naziv_objekta.
```

Tip skupa entiteta je skup slogova čiji su atributi prosti tipovi ili predstavljaju nizove znakova, skalarne tipove i podopsege prostih tipova. Ovaj tip se uvodi na sljedeći način:

```
TYPE E = ENTITY SET OF Z,
```

gdje je Z tip sloga. Nad tipom skupa entiteta definisane su i operacije unošenja i brisanja primjeraka tog tipa, kao i relacija pripadnosti primjerka skupu:

```
Insert(e, E);
Delete(e, E);
Member(e, E);
```

Pravilo komponovanja akcija FOREACH ima sljedeću formu:

```
FOREACH lista_opsega
    WHERE logički_izraz (AND logički_izraz)
    DO sekvenca_iskaza
END
```

Pored navedenih proširenja objektno orijentisane prirode programskog jezika Modula-2, MODULEX posjeduje identifikatorski tip i tip proširenog sloga (Wirth 1988). Vrijednosti identifikatorskog tipa su sistemski upravljani identifikatori primjeraka perzistentnih objekata, kojima se pored ostalog rješavaju problemi referentnih uslova integriteta. Uopštenost i fleksibilnost hijerarhije tipova objekata, kao jedne od fundamentalnih prednosti objektno orijentisanog pristupa, realizovana je u ograničenoj formi uvođenjem tipa proširenog sloga, čime je omogućeno vođenje koncepta nasljeđivanja i ograničena forma polimorfizma tipova.

Generalno govoreći, objektno orijentisana programska okolina baza podataka MODULEX posjeduje sljedeće osobine:

(i) učaurenje osobina objekta zajedno sa pridruženim akcijama
(ii) skrivanje informacija, tj. odvajanje specifikacije objekta od njegove implementacije
(iii) identifikatori objekata i njihovo sistemski orijentisano upravljanje
(iv) kompleksni objekti, posebno oni modelirani primjenom standardnih apstrakcija
(v) apstraktni tipovi podataka i tipovi objekata reprezentovani u smislu modula
(vi) hijerarhija tipova objekata i nasljeđivanje kroz ekstenzibilne tipove slogova

(vii) perzistencija objekata reprezentovanih kao veliki skupovi slogova

(viii) ekstenzibilnost programske okoline kao ekstenzibilne kolekcije modula

(ix) karakteristike sistema baza podataka: upravljanje eksternom memorijom (dinamički indeksi), kontrola konkurentnosti i oporavak.

Zapravo, sve ove osobine omogućavaju primjenu objektno orijentisanih principa projektovanja složenih i efikasnih softverskih sistema (Meyer 1989), (Pomberger 1989).

Jednostavnost ovako neformalno opisanog objektno orijentisanog programskog jezika jedna je od njegovih osnovnih karakteristika u odnosu na velik broj netrivijalnih proširenja koji posjeduje većina predloženih programskih jezika baza podataka. To je i glavni razlog što je MODULEX već implementiran. S druge strane, ostvarena integracija relacionog modela, proceduralnih apstrakcija i apstrakcija tipova podataka potrebnih za konceptualno modeliranje, kao i karakteristika objektno orijentisanih jezika, kao što su moduli, generalno je veoma interesantna za primjenu u kompleksnim aplikacionim okolinama.

3. APLIKACIONA OKOLINA

Radi zadovoljavanja informacionih potreba korisnika, tehnologija baza podataka zahtijeva projektovanje odgovarajuće reprezentacije aplikacione okoline. Ova se reprezentacija naziva konceptualni model, koji predstavlja apstraktnu reprezentaciju te okoline i sadrži samo one osobine okoline koje su relevantne za informacione zahtjeve korisnika (Alagić 1989). U projektovanju konceptualnog modela neophodno je klasificirati njene relevantne objekte u kolekciju tipova objekata. Skup objekata je tip objekta, ako svi objekti u tom skupu imaju iste relevante osobine (atribute). Sama aplikaciona okolina je reprezentovana:

— relevantnim objektima i njihovim osobinama, kao i odnosima među njima

— akcijama nad objektima i njihovim odnosima, koje odražavaju aktivnosti koje se javljaju i odnose se na osobine reprezentovanih objekata i odnosa među njima

— uslovima integriteta koji specificiraju one statičke i dinamičke osobine okoline koje nisu konvencionalno izražene u reprezentaciji objekata i akcija nad njima.

Katastar nekretnina sadrži tri međusobno povezane i uslovljene cjeline: katastarske planove, katastarski operat i zbirku isprava. Katastarski planovi sadrže podatke o obliku i položaju katastarskih parcela i zgrada, broju katastarske parcele i druge oznake, kojima se iskazuje karakteristika i sadržaj objekata i rastinja na zemljištu. Katastarski operat predstavlja skup spiskova i pregled u kojima su evidentirani podaci o nekretninama, podaci o vlasniku, odnosno korisniku i posjedniku, i vrstama prava i tereta. Zbirka isprava sastoji se od isprava na osnovu kojih su izvršeni upisi u katastar nekretnina.

3.1. *Konceptualni model*

Jedan od fundamentalnih tipova entiteta u ovoj aplikacionoj okolini je KatastarskaOpština. Jednoj katastarskoj opštini pripada skup katastarsko-

knjižnih uložaka i svaki uložak pripada jednoj i samo jednoj katastarskoj opštini. Ovaj 1 : N odnos između tipova entiteta KatastarskaOpština i KatastarskoknjižniUložak odgovara funkciji

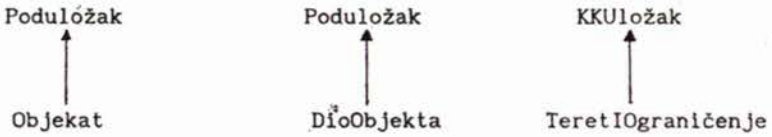
Svaki katastarskoknjižni uložak sadrži skup parcela (list A), skup nosilaca prava (vlasnika, posjednika) koji imaju ista prava nad tim skupom parcela (list B), skup objekata i njihovih dijelova (list A1), nosilaca prava nad njima (list B1) i skup tereta i ograničenja (list C). Naravno, neki od ovih skupova mogu biti prazni. Jedan vlasnik (posjednik) može imati prava nad parcelama koje se nalaze u različitim katastarskoknjižnim ulošcima. Dakle, ovdje postoji odnos M : N između tipova entiteta Vlasnik i KKUložak. To znači ne samo da je konačan skup od N vlasnika (posjednika) pridružen skupu parcela u jednom katastarskoknjižnom ulošku, nego da vlasnik (posjednik) može da ima M takvih skupova parcela (u M katastarskoknjižnih uložaka). Isti odnos postoji između tipova entiteta Vlasnik i Poduložak, što znači ne samo da je skup od N vlasnika (posjednika) pridružen skupu objekata, odnosno dijelova objekata u jednom podulošku, nego da jedan vlasnik (posjednik) može da ima M takvih poduložaka. Ovi tipovi odnosa reprezentuju se agregacijom, uvođenjem asocijativnih tipova entiteta VlasnikKKUložak i VlasnikPoduložak i dva odnosa 1 : N koja odgovaraju projekcijama u sljedećim dijagramima:

Kako između tipova entiteta KKUložak i Poduložak postoji odnos 1 : N (svakom katastarskoknjižnom ulošku pridružen je konačan skup (moguće i prazan) od N poduložaka), njega možemo predstaviti funkcijom:

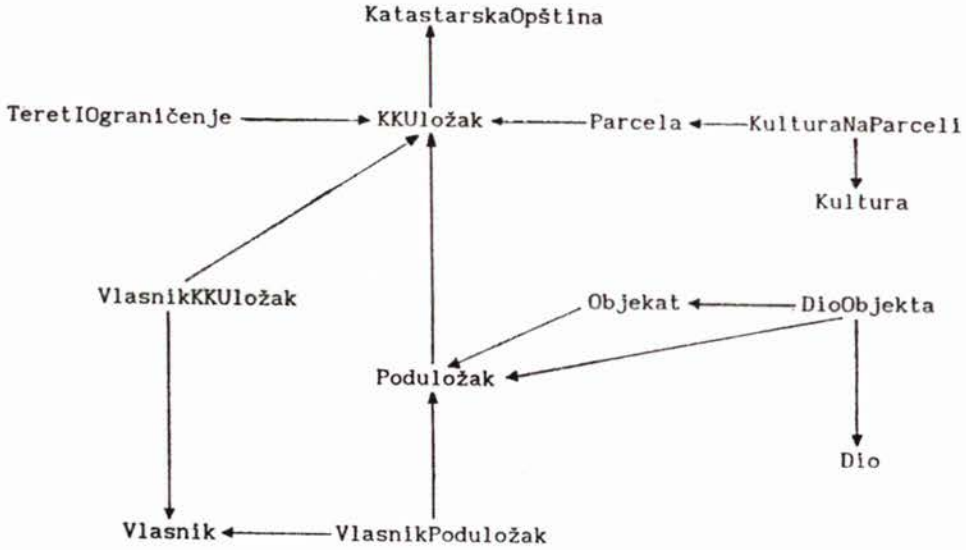
Među tipovima entiteta Parcela i Kultura, kao i između tipova entiteta Objekat i Dio postoji odnos M : N. Ovaj tip odnosa se, kao što smo već vidjeli, reprezentuje agregacijom tipa entiteta i dva odnosa 1 : N, koji odgovaraju projekcijama u sljedećim dijagramima:

Kako jedna parcela može da pripada jednom i samo jednom katastarskoknjižnom ulošku, a već smo rekli da je jednom ulošku pridružen skup parcela, između tipova entiteta KKUložak i Parcela postoji odnos 1 : N.

Isti odnos postoji između tipova entiteta Poduložak i Objekat, Poduložak i DioObjekta, i KKUložak i TeretIOgraničenje:



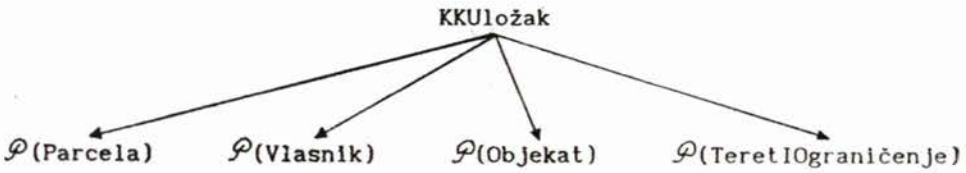
Cjelokupan rezultat projektovanja naše aplikacione okoline prikazan je na slici 1.



Slika 1.

3.2. *Kompleksni objekti; pokrivanje i apstraktni tipovi podataka*

Za korisnike podataka katastra nekretnina fundamentalan objekt je katastarskoknjižni uložak. Sve relevantne akcije korisnika zapravo su akcije nad tim objektom (uvidi u cijeli uložak ili neke njegove dijelove, kao i promjene i ažuriranja podataka koje on sadrži), ili skupom tih objekata (u cilju generisanja raznih statističkih izvještaja). Imamo li u vidu činjenicu da je i za korisnika katastarskoknjižni uložak kompleksan objekt, sa aspekta strukturnog modeliranja naše aplikacione okoline, objekt KKUložak definisan je primjenom dvije standardne apstrakcije: agregacije i pokrivanja (Codd 1979) (asocijacija u (Brodie 1984)), kao agregacija prostih i skupovnih atributa:



Naša reprezentacija zasnovana je na apstraktnim skupovnim tipom podataka atributa objekta specificiranim kao skriveni tipovi u definicionom modulu (Wirth 1983) tog objekta, zajedno sa odgovarajućim akcijama pridruženim takvim tipovima:

SkupObjekataIDijelova
SkupTeretaIOgraničenja

```
DEFINITION MODULE KKUlozak;

FROM KatastarskaOpstina IMPORT SifraKO;

TYPE  SkupParcela,
      SkupVlasnika,
      SkupObjekataIDijelova,
      SkupTeretaIOgranicenja; (* apstraktni tipovi podataka *)

      KKUlozakID = IDENTIFIER OF TipKKUlozak; (* identifika
                                                 torski tip *)

      TipKKUlozak = RECORD
                       ko     : SifraKO;
                       broj   : CARDINAL;
                       Alist  : SkupParcela;
                       Blist  : SkupVlasnika;
                       A1list : SkupObjekataIDijelova;
                       B1list : SkupVlasnika;
                       Clist  : SkupTeretaIOgranicenja
                    END;

PROCEDURE NadjiKKUlozak (ko:SifraKO; broj:CARDINAL) : KKulo-
          zakID;
PROCEDURE PrikaziKKUlozak (kku : KKUlozakID);
```

```
PROCEDURE UnesiKKUlozak (kkulozak : TipKKUlozak);
PROCEDURE BrisiKKUlozak (kku : KKulozakID);
...
PROCEDURE DodajTeret   (kku:KKUlozakID; clist:SkupTeretaIOgrani
                        cenja);
PROCEDURE BrisiTeret (kku:KKUlozakID; clist:SkupTeretaIOgranicenja);
...
(* izostavljeno rukovanje izuzetnim situacijama *)
...
END KKUlozak.
```

Značenje tipova SkupParcela, SkupVlasnika, SkupObjekataIDijelova i SkupTeretaIOgraničenja jasno je za korisnike modula KKUložak, kao i činjenica da je definisan ograničen skup akcija koje se mogu izvršiti nad primjercima tih tipova (npr. BrisiTeret). Na taj način, prošireni relacioni model nije narušen jer je eksplicitna specifikacija ovih apstraktnih tipova podataka sakrivena u implementacionom modulu KKUložak, gdje je data aktuelna reprezentacija pokrivanja u smislu agregacije.

Sve je to moguće zahvaljujući činjenici da su skriveni tipovi u programskom jeziku Modula-2 ograničeni na pokazivačke tipove, a u MODULEX-u su svi tipovi skupova entiteta reprezentovani predefinisanim skrivenim tipom koji je ustvari pokazivački tip.

Na taj način ostvareno je nekoliko važnih ciljeva:

— korisnik objekta KKUložak može da koristi samo predefinisani skup akcija nad njim. Kako je njihova implementacija zaštićena od vanjskog pristupa, njihova korektnost može biti zagarantovana.

— korisnici vide bazu podataka sa apstraktnog nivoa, gdje struktura objekata postaje vidljiva samo indirektno, preko akcija nad objektima.

— interna reprezentacija može u određenom smislu biti i promijenjena, bez uticaja na korisničke uvezene module, tako da njihova rekompilacija nije neophodna.

Dekompozicija akcija definisanih na konceptualnom nivou i pridruženih tipu objekta KKUložak ilustrovana je za akciju UnesiKKUložak. Funkcija *NewId*, koja se pojavljuje u toj proceduri, omogućava unošenje primjerka objekta u specificirani skup perzistentnih objekata i vraća kao svoj rezultat identifikator tog primjerka.

```
IMPLEMENTATION MODULE KKUlozak;

IMPORT Vlasnik,Parcela,Pravo,VideoTerminal,InOut;
...
TYPE SkupVlasnika = Vlasnik.TipSkupVlasnika;
     SkupParcela  = Parcela.TipSkupParcela;
     ...
     KKUlozakSlog = RECORD
                       ko   : SifraKO;
                       broj : CARDINAL
                    END;
```

```
        VlasnikKKUlozak = RECORD
                                    vlasnik      : Vlasnik.VlasnikID;
                                    kkulozak     : KKUlozakID;
                                    vrstaprava   : Pravo.SifraPrava;
                                    obimpravab   : CARDINAL;
                                    obimpravan   : CARDINAL
                                END;

        ...
        TipSkupKKUlozak       = ENTITY SET OF KKUlozakSlog;
        TipSkupVlasnikKKUlozak = ENTITY SET OF VlasnikKKUlozak;
        ...
VAR     SkupKKUlozak       : TipSkupKKUlozak;
        SkupVlasnikKKUlozak : TipSkupVlasnikKKUlozak;
        ...
PROCEDURE ZavrsenUnos();
...
PROCEDURE UnesiKKUlozak (kkulozak : TipKKUlozak);
VAR kku     : KKulozakSlog;
    vl      : Vlasnik.TipVlasnik;
    vlkku   : VlasnikKKUlozak;
BEGIN
      kku.ko:=kkulozak.ko;
      kku.broj:=kkulozak.broj;
      vlkku.kkulozak:=NewId(kku,SkupKKUlozak);
      REPEAT
            InOut.WriteString("Matični broj : ");
            InOut.ReadString(vl.matbroj); InOut.WriteLn;
            IF Ispravan(vl.matbroj)
            THEN
                  InOut.WriteString("Adresa       : ");
                  InOut.ReadString(vl.adresa); InOut.WriteLn;
                  vlkku.vlasnik:=Vlasnik.PostojiVlasnik(vl.matbroj);
                  IF vlkku.vlasnik = NIL
                  THEN vlkku.vlasnik:=NewId(vl,Vlasnik.SkupVlas
                        nik)
                  END;
                  InOut.WriteString("Obim prava : ");
                  InOut.ReadCard(vlkku.obimpravab);
                  InOut.WriteString(" / ");
                  InOut.ReadCard(vlkku.obimpravan); InOut.WriteLn;
                  Insert(vlkku,SkupVlasnikKKUlozak)
            ELSE InOut.WriteString("Neispravan matični broj !")
            END;
            VideoTerminal.ClearDisplay
       UNTIL ZavrsenUnos();
       ...
END UnesiKKUlozak;
...
END KKUlozak.
```

3.3. *Podrška grafičkoj reprezentaciji podataka; ekstenzibilnost, nasljeđivanje i polimorfizam*

Očekuje se da grafička reprezentacija i manipulacija podataka predstavljaju jednu od glavnih odlika novih sistema za upravljanje podacima katastra nekretnina. U tom smislu, posebna je pažnja posvećena razvoju efikasne podrške za grafičku manipulaciju i reprezentaciju podataka, odnosno izgradnji biblioteke za manipulaciju geometrijskim objektima (tačke, segmenti, linije, poligoni, itd.). Tako npr. definicioni modul geometrijskog objekta Poligon ima sljedeću formu:

```
DEFINITION MODULE Poligon;

FROM Tacka IMPORT TipTacka;
FROM KartografskiKljuc IMPORT TipBoja;

TYPE  NizTjemena;
      TipSkupPoligona; (* apstraktni tipovi *)
      PoligonID = IDENTIFIER OF TipPoligon;
      TipPoligon = RECORD
                        id      : PoligonID;
                        tjemena : NizTjemena
                   END;

VAR SkupPoligona : TipSkupPoligona;

PROCEDURE UnesiPoligon (VAR p : TipPoligon);
PROCEDURE NacrtajPoligon (p : TipPoligon);
PROCEDURE ObojiPoligon (p:TipPoligon; R:CARDINAL; boja:TipBoja);
PROCEDURE Povrsina (p : TipPoligonID) : REAL;
PROCEDURE Obim (p : TipPoligonID) : REAL;
PROCEDURE Centroid (p : TipPoligon; VAR c:TipTacka);
PROCEDURE Susjedni (p,q : TipPoligonID) : BOOLEAN;
PROCEDURE Unija (p,q : TipPoligonID; VAR r : TipPoligon);
PROCEDURE Presjek (p,q : TipPoligonID; VAR r : TipPoligon);
PROCEDURE Razlika (p,q : TipPoligonID; VAR r : TipPoligon);

END Poligon.
```

U našoj aplikacionoj okolini važan atribut objekta Parcela je njen oblik, koji se u grafičkom smislu reprezentuje poligonom, pa ih je u cilju grafičke manipulacije ovim objektima neophodno i reprezentovati poligonima. U standardnim modelima podataka zastupljeni su prosti tipovi podataka (znakovni, logički, realni i cjelobrojni), pa atributi objekata mogu da imaju vrijednosti iz određenog dijapazona vrijednosti pomenutih tipova podataka. Naravno, nijedan standardni model ne posjeduje poligon kao tip. Upravo zbog takvih potreba, koje su inače karakteristične za nestandardne aplikacione okoline, potrebno je obezbijediti odgovarajuću reprezentaciju relevantnih objekata.

U MODULEX-u je ovakve probleme moguće prevazići korištenjem koncepta nasljeđivanja i polimorfizma tipova, fundamentalnih osobina objektno orijentisanog pristupa (Meyer 1988). U našem slučaju, objekt Parcela posmatra se kao podtip objekta Poligon. Tip sloga koji reprezentuje primjerke podtipova objekta Poligon (TipParcela) jeste proširenje tipa sloga koji odgovara

tipu objekta (TipPoligon). Prošireni slog nasljeđuje sve atribute generičkog tipa objekta, kao i sve akcije definisane nad njim. Pored toga, podtipu mogu biti pridružene nove akcije, a generičke akcije mogu biti redefinisane. Ovakav pristup uvodi specifičnu formu polimorfizma, koji omogućava da se tip proširenog kao i na zamjenu parametara procedura, identifikatora tipova objekata i njegovih podobjekata, kao i na tipove skupova objekata. Sve ove karakteristike omogućavaju dodatnu opštost i fleksipibnost.

U definicionom modulu Poligon specificirana je generička struktura i odgovarajuće akcije nad tom strukturom. Za ovaj generički tip moguće je konstruisati različite podtipove objekata, kao što je i Parcela. Sve generičke akcije pridružene poligonu primjenljive su i na parcelu, a naravno, definisane su i neke specifične:

```
DEFINITION MODULE Parcela;

IMPORT Poligon;
FROM Kultura IMPORT SifraKulture,KulturaID;

TYPE  SkupKultura;
      ...
      ParcelaID = IDENTIFIER OF TipParcela;

      TipParcela = SUBTYPE OF Poligon.TipPoligon
                   RECORD
                          kkulozak: KKUlozakID;
                          broj    : CARDINAL;
                          podbroj : CARDINAL;
                          brplana : CARDINAL;
                          brskice : CARDINAL;
                          naziv   : STRING[15];
                          sektor  : CHAR;
                          namjena : CHAR;
                          teret   : BOOLEAN;
                          brdnevn : STRING[5];
                          datum   : STRING[8];
                          izkkul  : CARDINAL;
                          kulture : SkupKultura
                   END;

PROCEDURE NadjiParcelu (ko:SifraKO; br,podbr:CARDINAL):ParcelaID;
PROCEDURE UnesiParcelu (VAR p : TipParcela);
PROCEDURE BrisiParcelu (pid : ParcelaID);
PROCEDURE CijepajParcelu (pid : ParcelaID);
PROCEDURE PrikaziParcelu (p : TipParcela; R : CARDINAL);
PROCEDURE PrimijeniKulturu (pid:ParcelaID; kulture:SkupKultura);
PROCEDURE PostojiKultura (p:TipParcela; s:SifraKulture):KulturaID;
...
END Parcela.
```

U implementacionom modulu ilustrovana je primjena generičke (nasljeđene) procedure NacrtajPoligon nad primjerkom podtipa Parcela:

```
IMPLEMENTATION MODULE Parcela;

FROM Kultura IMPORT TipKultura,TipSkupKultura,KulturaID,Kulture;
FROM KartografskiKljuc IMPORT KartografskiZnak;

TYPE  SkupKultura : TipSkupKultura;

      ParcelaSlogID = IDENTIFIER OF ParcelaSlog;

      ParcelaSlog   = RECORD
                        id       : ParcelaSlogID;
                        kkulozak : KKUlozakID;
                        broj     : CARDINAL;
                        podbroj  : CARDINAL;
                        brplana  : CARDINAL;
                        brskice  : CARDINAL;
                        naziv    : STRING[15];
                        sektor   : CHAR;
                        namjena  : CHAR;
                        teret    : BOOLEAN;
                        brdnevn  : STRING[5];
                        datum    : STRING[8];
                        izkkul   : CARDINAL
                      END;

      KulturaParcela = RECORD
                        kultura  : KulturaID;
                        parcela  : ParcelaID;
                        boja     : KartografskiZnak;
                        poligon  : PoligonID
                      END;

      TipSkupParcela        = ENTITY SET OF ParcelaSlog;
      TipSkupParcelaKultura = ENTITY SET OF KulturaParcela;

VAR   SkupParcela      : TipSkupParcela;
      KultureNaParceli : TipSkupKulturaParcela;
      ...
PROCEDURE PrikaziParcelu(p : TipParcela; R : CARDINAL);
VAR kp   : KulturaParcela;
    parc : ParcelaSlog;
    pol  : Poligon.TipPoligon
BEGIN
      FOREACH parc IN SkupParcela
      WHERE (parc.kkulozak = p.kkulozak) AND
            (parc.broj = p.broj) AND
            (parc.podbroj = p.podbroj)
      DO NacrtajPoligon(p,R);
         Exit
      END;
```

```
            FOREACH kp IN KultureNaParceli
            WHERE (kp.parcela = parc.id)
            DO
               FOREACH pol IN SkupPoligona
               WHERE (pol.id = kp.poligon)
               DO
                  NacrtajPoligon(pol,R);
                  ObojiPoligon(pol,R,kp.boja)
               END
            END
      END PrikaziParcelu;
      ...
      END Parcela.
```

Primjer korisničkog modula za grafički prikaz svih parcela u razmjeri 1:1000, koje zadovoljavaju sljedeće uslove:

— nalaze se u društvenoj svojini
— namijenjene su za izgradnju
— površina parcele je veća od 100 m²
— na parceli ne postoji nikakav stambeni objekat

djelomično ilustruje na koji način je ostvarena podrška grafičkoj manipulaciji i prikazu podataka katastra nekretnina:

```
MODULE ParceleZaGradnju;
IMPORT Parcela;
VAR p : Parcela.TipParcela;
    R : CARDINAL;
BEGIN
      R:=1000;
      FOREACH p IN SkupParcela
      WHERE (p.sektor = "1)" AND (p.namjena = "1") AND
            (Poligon.Povrsina(p) > 100.0) AND
            (Parcela.PostojiKultura(p,"510") = NIL)
      DO Parcela.PrikaziParcelu(p,R)
END ParceleZaGradnju.
```

Očigledno je da je u korisnički modul neophodno uvesti jedino modul Parcela u cilju realizacije specificiranog upita.

4. ZAKLJUČAK

U radu su prikazani osnovni koncepti projektovanja i implementacije prototipa sistema za upravljanje podacima katastra nekretnina. Glavne karakteristike tog sistema ogledaju se u primjeni najnovijih informacionih tehnologija, novih metodologija u razvoju složenih softverskih sistema, kao i efikasnoj grafičkoj manipulaciji podataka. Za implementaciju sistema upotrijebljena je objektno orijentisana programska okolina baza podataka MODULEX, a posebno je prikazan programski jezik baza podataka.

Reprezentacija kompleksnih objekata ostvarena je primjenom apstraktnih tipova atributa, i dostupan je samo skup akcija nad njima, dok je njihova

aktuelna reprezentacija skrivena u odgovarajućem implementacionom modulu. Prikazanu aplikacionu okolinu karakterišu objekti sa kompleksnom strukturom, reprezentovani kao veliki perzistentni skupovi.

Zahvaljujući opštosti i fleksibilnosti, koje zajedno omogućavaju moduli i ekstenzibilni tipovi slogova, moguće je ostvariti modeliranje hijerarhije tipova, nasljeđivanje i specifičnu formu polimorfizma, a samim tim i primjenu objektno orijentisanog pristupa u projektovanju i efikasnoj realizaciji prikazanog sistema.

Iskustva u primjeni MODULEX-a ispunila su naša očekivanja, a glavni rezultati primjene objektno orijentisane tehnologije baza podataka su:

— ostvarivanje mogućnosti primjene razvijenog softvera za potrebe razvoja drugih softverskih sistema (reusability); naime, definicioni moduli mogu biti dostupni drugim korisnicima, u razvoju drugih sistema (npr. informacioni sistemi o prostoru)
— efikasno upravljanje i rukovanje objektima, kako za neformalnog korisnika, tako i za aplikacionog programera
— korisnici su u potpunosti oslobođeni poznavanja dekompozicije akcija nad relevantnim objektima
— zadovoljavajuće performanse baze podataka.

Prototip sistema za upravljanje podacima katastra nekretnina, čije su osnovne karakteristike prikazane u radu, implementira se na radnoj stanici Sun pod operativnim sistemom UNIX.

LITERATURA

Alagić, S.: Object-Oriented Database Programming, Springer-Verlag, New York 1989.

Alagić, S.: MODULEX: A Multiparadigm Approach to Database Programming, Technical Report, Software Development Laboratory, Faculty of Electrical Engineering, University of Sarajevo 1989.

Alagić, S., Galić Z.: Object-Oriented Geo-Information Processing in MODULEX, Proceedings of the Second International Conference on Technology of Object-Oriented Languages and Systems, pp. 393—406, Paris 1990.

Alagić, S., Galić, Z.: Advanced Database Programming Languages : A Geo-Information Processing Prospective, Proceedings of the Information Technology Conference CIPS '90, Edmonton 1990.

Atkinson, Bancilhon, F., DeWitt, D., Dittrich, D., Maier, D., and Zdonik, S.: The Object-Oriented Database System Manifesto, Technical Report 30—89, GIP ALTAIR, INRIA, 1989.

Bancilhon, F.: Object-Oriented Database Systems, Tutorials of TOOLS '90, Paris 1990.

Beech, D.: A Foundation for Evolution from Relational to Object Databases, Advances in Database Technology — EDBT 88, Proceedings of the International Conference, Venice 1988.

Brodie, M. L., Riđanović, Dž.: On the Design and Specification of Database Transactions, in M. L. Brodie, J. Mylopoulos and J. W. Schmidt (eds.), On Conceptual Modelling, Springer-Verlag, New York 1984, 277—312.

Codd, E. F.: Extending the Database Relational Model to Capture More Meaning, ACM TODS, 1979, 4, 397—434.

Cox, B.: Object-Oriented Programming — An Evolutionary Approach, Addison-Wesley, 1986.

Guting, R. H.: GRAL : An Extensible Relational Database System for Geometric Applications, Universitat Dortmund, Fachbereich Informatik, Report 293, 1989.

Meyer, B.: Object-Oritented Software Construction, Prentice-Hall 1988.
Pomberger, G.: Modula-2 and Software Engineering, First International Modula-2 Conference, Bled 1989.
Stroustrup, B.: The C++ Programming Language, Addison-Wesley Publishing Company, Reading, Massachusetts 1986.
Tesler, L.: Object Pascal Report, Structured Language World, 1985, 9, 10—14.
Wirth, N.: Programming in Modula-2, Springer-Verlag, Berlin 1983.
Wirth, N.: Type Extensions, ACM Transactions on Programming Languages and Systems, 1988, 2, 204—214.
Wirth, N.: From Modula to Oberon, Software Practice and Experience, 1988, 18, 661—670.

OBJECT-ORIENTED SYSTEM IN MANAGEMENT OF REAL ESTATES CADASTRE DATA

This paper reports on a project to implement a prototype data management system for real estates cadastre based on object-oriented database technology. Some of the basic features of MODULEX, an object-oriented database programming environment are also reported. The application environment is characterized by objects with complex structure represented as large, persistent sets, and actions whose complexity is determined with the complexity of objects. The imporance of object-oriented paradigm (data abstraction, extensibility, inheritance, and polymorphism) is also demonstrated.
Key words: Object-oriented, database management, database programming, real estates cadastre.

Primljeno: 1990-11-12