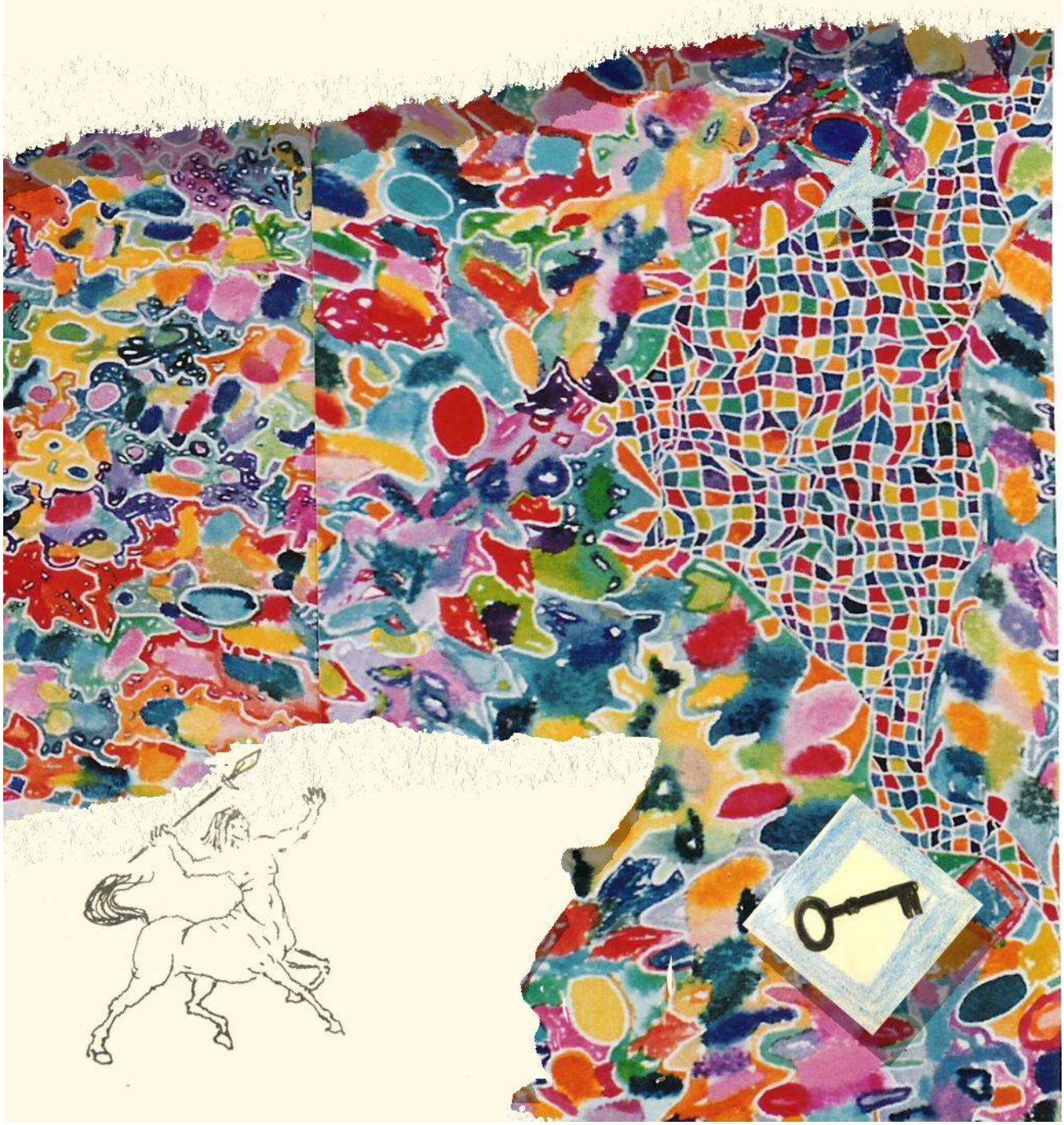
Slavko Marić Dražen Brđanin

# RELACIONE BAZE PODATAKA



# Glava 2

# Konceptualno modelovanje podataka

Kao što je navedeno u uvodnom dijelu, projektovanje baze podataka je jedan od najznačajnijih segmenata razvoja sistema sa bazama podataka. Cilj je da se dođe do detaljne specifikacije sveukupne strukture baze podataka (modela) za odgovarajući DBMS, odnosno do logičke šeme baze podataka. Logička šema mora odražavati prirodna svojstva objekata i pojava realnog svijeta i njihove stvarne veze. Loša organizacija baze podataka može biti ekstremno razorna i biti izvor loših aplikativnih karakteristika i potrebe za čestim, obimnim i skupim promjenama aplikacija. Poseban problem je razvoj i implementacija kompleksnih sistema sa bazama podataka, u kojima se vodi na hiljade elementarnih podataka koje je potrebno ispravno grupisati i uspostaviti odgovarajuće relacije između pojedinih grupa. Zbog toga je potreban sistematizovan pristup koji podrazumijeva korištenje odgovarajuće metodologije. Poželjno je da je metodologija bliska čovjeku i njegovoj percepciji realnog svijeta i da posjeduje odgovarajuće koncepte za izražavanje te percepcije u formi konceptualne šeme na visokom nivou apstrakcije. Dodatno, potrebno je da postoji mogućnost za jednostavnu transformaciju dobijene konceptualne šeme u logičku šemu za odnosni DBMS.

Najčešće korišteni model visokog nivoa je **Model Objekti-Veze**, iz kojeg je moguće jednostavnim pravilima generisati relacioni model podataka, na kojem su bazirani danas najaktuelniji DBMS. Pored ova dva modela, u ovom poglavlju je opisano i modelovanje podataka korištenjem UML-a, široko korištenog unificiranog jezika za modelovanje sistema.

## 2.1 Model objekti-veze

Ukoliko posmatramo neki realni sistem, možemo u takvom sistemu uočiti postojanje određenih entiteta (objekata): predmeta, osoba, pojava, koncepata, događaja itd. Svaki entitet predstavlja nešto što postoji (ne nužno materijalno) kao posebna cjelina, i što se razlikuje od drugih entiteta koji egzistiraju u sistemu od interesa. Na primjer, u nekoj kompaniji koja se bavi veletrgovinom, entiteti su konkretne organizacione jedinice, poslovni partneri, ugovori, fakture, zaposleni, itd. Entiteti koje možemo uočiti posmatrajući univerzitet kao sistem od interesa jesu, između ostalog, pojedini fakulteti, studenti, nastavnici, nastavni planovi, studijski programi, konkretni predmeti itd. Posmatrajući entitete sistema od interesa, možemo uočiti da između pojedinih entiteta postoje određene veze. Na primjer, neka faktura je povezana sa određenim poslovnim partnerom, ili, neki određeni predmet može biti povezan sa (izvoditi se na) više studijskih programa.

Koncept modela objekti-veze je jednostavan i intuitivan. Zasniva se na percepciji realnog svijeta, odnosno identifikaciji klasa sličnih entiteta i specifikaciji veza između njih putem odgovarajućih grafičkih koncepata. Modelom konkretnog sistema se specifikuje sveukupna logička struktura baze podataka sistema na visokom nivou apstrakcije, odnosno konceptualna šema. Konceptualna šema se reprezentuje grafičkom notacijom poznatom pod nazivom **MOV dijagram** (eng. *E-R diagram*). Model objekti-veze predstavlja **semantički model** podataka, jer omogućava ne samo reprezentaciju podataka (entiteta) i veza između njih, nego i razumijevanje i interpretaciju njihovog značenja.

### 2.1.1 Osnovni koncepti

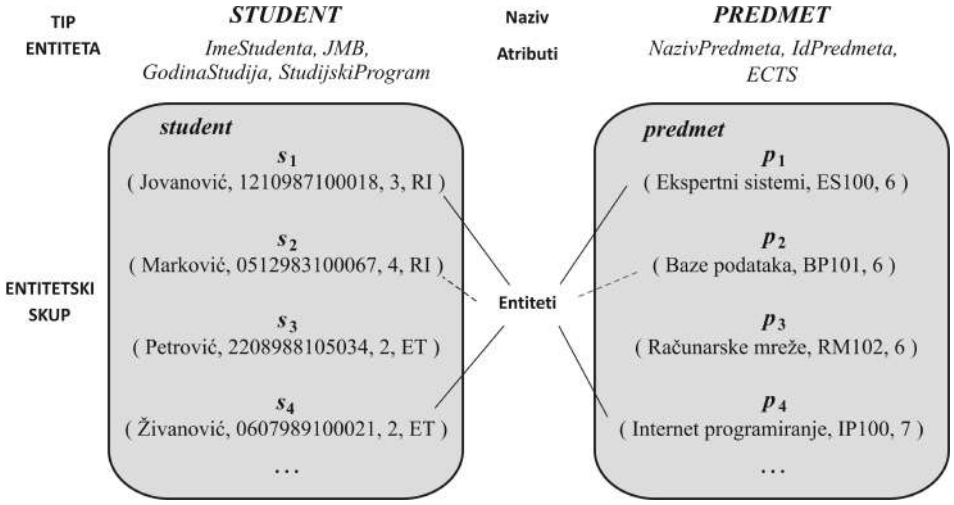
#### Entiteti, entitetski tipovi i skupovi

**Entitet** (objekat) je nešto što postoji u realnom svijetu i što se razlikuje od drugih postojećih objekata. Entitet može da egzistira fizički (određeni student, knjiga, ...) ili konceptualno (predmet studijskog programa, posao, praznik, i sl.). Svaki entitet realnog svijeta karakterišu određena svojstva koja se u MOV reprezentuju **atributima**. Određeno svojstvo entiteta se iskazuje odgovarajućom kvantitativnom vrijednošću atributa. Na primjer, određeni student (entitet realnog svijeta) ima konkretne vrijednosti za atribute *ImeStudenta*, *JMB*, *GodinaStudija*, *StudijskiProgram*. Dakle, entiteti se mogu opisati skupom parova: (*atribut*, *vrijednost*). Na primjer, student Jovanović može da se opiše skupom:

$$\{(ImeStudenta, \text{Jovanović}), (JMB, 1210987100018),$$
$$(GodinaStudija, 3), (StudijskiProgram, \text{RI})\}.$$

Entiteti na određenom podskupu atributa mogu da imaju jedinstvene vrijednosti. Na primjer, svaki student ima jedinstvenu vrijednost za atribut *JMB*.

Određene entitete realnog svijeta karakterišu ista svojstva, odnosno ti entiteti mogu da se reprezentuju istim skupom atributa. Za takve entitete kažemo da su slični, odnosno da su istog tipa. **Tip entiteta** je određen imenom tipa i skupom atributa koji reprezentuju svojstva svih entiteta tog tipa. Na primjer, entitetski tip *STUDENT* je određen atributima *ImeStudenta*, *JMB*, *GodinaStudija*, *StudijskiProgram*. Tip entiteta opisuje **šemu** (eng. *schema*[18]), odnosno strukturu entiteta koji pripadaju istom entitetskom skupu.




**Slika 2.1**: Entiteti, entitetski skupovi i tipovi

**Entitetski skup** je skup entiteta istog tipa u bazi podataka, u nekom trenutku. Na primjer, sve studente karakterišu svojstva definisana entitetskim tipom *STUDENT*. Entitetskim skupom *student* možemo predstaviti skup svih studenata u bazi podataka univerziteta, u nekom trenutku. Slično, entitetski skup *predmet* reprezentuje skup svih pojedinačnih predmeta predviđenih nastavnim planovima univerziteta (sl. 2.1). Slični entiteti se međusobno razlikuju u vrijednostima određenih atributa. Za entitetski skup

[18] Koristi se i termin *intension*.

se još kaže da predstavlja ekstenziju (eng. *extension*) entitetskog tipa. S obzirom na to da su svojstva (atributi) entiteta iz entitetskog skupa definisana entitetskim tipom, slijedi da važi i obrnuto, svojstva (atributi) entiteta/entitetskog skupa su i atributi odnosnog entitetskog tipa. Iz tog razloga, termini entitetski tip i entitetski skup se mogu koristiti alternativno. Entitetski skup se uobičajeno referencira istim imenom kao i entitetski tip, tako da se naziv *STUDENT* može koristiti i za referenciranje entitetskog tipa i za referenciranje na tekući entitetski skup svih studenata u bazi. U daljem tekstu, tamo gdje je to pogodnije, za eksplicitno referenciranje na entitetski skup koristiće se isti naziv kao i za entitetski tip, pisan malim slovima u kurzivu (na primjer entitetski skup *student*). Nazivi entitetskih tipova su imenice koje asociraju na vrstu i karakteristike entiteta koje reprezentuju.

Tip entiteta odgovara programerskom konceptu tipa promjenljivih. Promjenljiva određenog tipa ima određenu vrijednost u posmatranom vremenskom trenutku. Programerski koncept promjenljive odgovara konceptu entiteta u MOV.

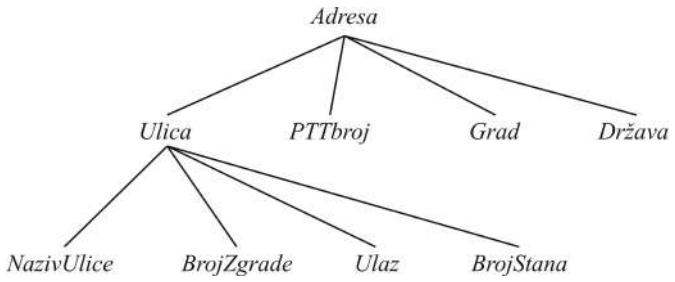
Entitetski skupovi ne moraju biti u potpunosti disjunktni. Neki entitet može pripadati različitim entitetskim skupovima. Na primjer, određeni entitet iz entitetskog skupa *osoba* može pripadati i entitetskom skupu *zaposleni* i entitetskom skupu *nastavnici*.

### Atributi

Entiteti se reprezentuju odgovarajućim skupom atributa. Skup dopuštenih vrijednosti nekog atributa predstavlja njegov **domen**. S obzirom na to da se svojstva entiteta iskazuju kvantitativnim vrijednostima odnosnih atributa, atribut se formalno može smatrati funkcijom koja vrši preslikavanje entitetskog skupa u domen atributa. Naziv atributa treba da bude imenica koja asocira na svojstvo entiteta koje se reprezentuje atributom.

U MOV mogu postojati različiti tipovi atributa:

- **Prosti (atomični) i složeni (kompozitni) atributi.** Prosti ili atomični atributi imaju vrijednosti koje su nedjeljive. Složeni ili kompozitni atributi se sastoje od više komponenata (elementarnijih dijelova/atributa) koje imaju posebno i nezavisno značenje. Na primjer, dok je *DatumRođenja* osobe prosti atribut, *Adresa* stanovanja je složeni atribut koji uključuje komponente: *Ulica*, *PTTbroj*, *Grad*, *Država*. Atribut *Ulica* može se takođe tretirati kao kompozitni, čiji su komponentni atributi *NazivUlice*, *BrojZgrade*, *BrojUlaza*, *BrojStana* (sl. 2.2).




**Slika 2.2**: Kompozitni atributi

- **Jednoznačni i višeznačni atributi.** Jednoznačni (eng. *single-valued*) atributi mogu imati samo jednu vrijednost, dok višeznačni (eng. *multivalued*) atributi mogu imati više od jedne vrijednosti za neki entitet. Na primjer, *DatumRođenja* je jednoznačni atribut (jer osoba može imati samo jedan datum rođenja). Pretpostavimo da entitetski tip *ZAPOSLENI* sadrži atribut *StraniJezik*, kojim se opisuje poznavanje stranog jezika zaposlenog. Atribut *StraniJezik* je višeznačan: neki zaposleni (entitet) može poznavati više stranih jezika, atribut će imati po jednu vrijednost za svaki strani jezik koji zaposleni poznaje.
- **Bazni i izvedeni atributi.** Izvedeni (eng. *derived*) atributi su atributi čije se vrijednosti mogu izvesti iz drugih atributa. Na primjer, ako se za osobe vodi atribut *StarosnaDob*, koji opisuje starosnu dob osobe, vrijednost atributa *StarosnaDob* za svaku osobu se može izvesti iz tekućeg datuma i atributa *DatumRođenja*. Tekući datum je vrijednost koja se uvijek može dobiti od računarskog sistema pozivom odgovarajuće funkcije. Drugi primjer: recimo da entitetski tip *STUDENT* sadrži atribut *BrojPoloženihIspita*, koji treba da sadrži podatak o broju ispita koji je student položio. Ova vrijednost se može dobiti prebrojavanjem zapisa o položenim ispitima odnosnog studenta.

  Vrijednosti izvedenih atributa se ne pohranjuju u bazu podataka, za razliku od baznih (eng. *base* ili *stored*) atributa. Prilikom referenciranja, vrijednost izvedenog atributa se sračunava korištenjem odgovarajućih parametara i baznih atributa. U određenim specifičnim situacijama, kada je sračunavanje vrijednosti izvedenih atributa vremenski zahtjevno, projektant baze podataka može odlučiti da se vrijednosti izvedenih atributa pohranjuju u bazi podataka.

Pojedini atributi nekih entiteta mogu imati **null** vrijednost. *Null* vrijednost je specijalna vrijednost, koja može biti član bilo kojeg domena i koja označava da vrijednost atributa nije poznata ili da konkretna vrijednost nije primjenljiva za odnosni entitet. Na primjer, ako bi entitetski tip *ZAPOSLENI* imao atribute *BrojTelefona* i *AkademskoZvanje*, onda bi vrijednost *null* za atribut *BrojTelefona* nekog entiteta (zaposlenog) mogla (najčešće) da znači da broj telefona za posmatranog zaposlenog nije poznat (uobičajeno je da svaki zaposleni ima broj telefona preko kojeg se sa zaposlenim može uspostaviti kontakt i komunicirati) ili nije raspoloživ (na primjer ako je zaposleni stavio zabranu na identifikaciju broja telefona). S druge strane, neki zaposleni nemaju akademsko zvanje, pa bi za njih atribut *AkademskoZvanje* imao vrijednost *null*, sa značenjem da druga, konkretna vrijednost iz domena atributa nije primjenljiva. Sama *null* vrijednost nam ne daje informacije o tome da li je vrijednost atributa nepoznata ili konkretna vrijednost atributa nije primjenljiva za odnosni entitet. Na primjer, za nekog zaposlenog vrijednost atributa *BrojTelefona* u određenom trenutku/periodu vremena može biti neprimjenljiva, ako zaposleni ne posjeduje broj telefona.

Primijetimo da višeznačni atribut može biti istovremeno i kompozitni, kao i da neka komponenta kompozitnog atributa može biti višeznačna. Na primjer, student može imati više adresa: adresu u mjestu studiranja i adresu u mjestu prebivališta. U tom slučaju atribut *Adresa* je višeznačni, kompozitni atribut.

### Veze, vezni tipovi i skupovi

Između pojedinih entiteta u sistemu od interesa može postojati određeni odnos. Na primjer, ako nastavnik Matić predaje predmet Baze podataka, onda možemo definisati vezu – asocijaciju između navedenih entiteta, koja reprezentuje dati odnos, tj. činjenicu da nastavnik Matić predaje predmet Baze podataka.

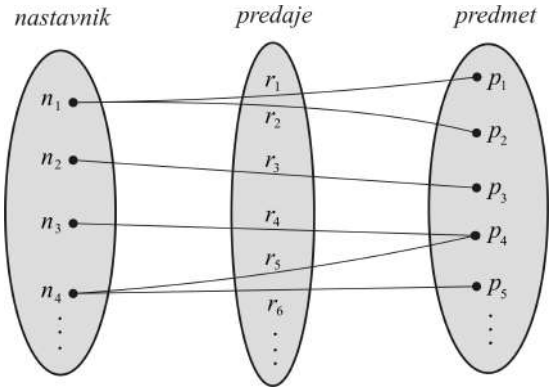
Formalno, **veza** (eng. *relationship*) je asocijacija $(e_1, e_2, \ldots, e_n)$ između entiteta $e_1, e_2, \ldots, e_n$, pri čemu je $e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n$. **Tip veze** $R$ između entitetskih tipova $E_1, E_2, \ldots, E_n$, definiše mogući **skup veza** između entiteta odnosnih entitetskih tipova. Zato se tip veze može definisati kao podskup Dekartovog proizvoda $E_1 \times E_2 \times \cdots \times E_n$, odnosno kao matematička relacija na $E_1, E_2, \ldots, E_n$. Za entitetske tipove $E_1, E_2, \ldots, E_n$ se kaže da participiraju u veznom tipu $R$.

**Vezni skup** je skup veza istog tipa. Matematički, vezni skup je podskup skupa

$$\{(e_1, e_2, \ldots, e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}.$$

Vezni skup je skup konkretnih instanci veze. **Instanca veze** $r_i$ reprezentuje asocijaciju između konkretnih entiteta $e_1, e_2, \ldots, e_n$ koji pripadaju entitetskim skupovima tipa $E_1, E_2, \ldots, E_n$, respektivno. Za entitetske skupove čiji entiteti učestvuju u vezama veznog skupa, kaže se da participiraju u veznom skupu. Za označavanje i referenciranje veznih skupova važi isto pravilo kao i za entitetske skupove. Vezni skupovi se referenciraju istim imenom kao i odnosni tip veze. U ovom tekstu distinkcija je napravljena tako što se za tip veze koristiti naziv pisan velikim slovima, a za referenciranje na odnosni vezni skup koristi se isti naziv, pisan malim slovima u kurzivu. Poželjno je da nazivi veza budu glagoli koji izražavaju tip asocijacije između entiteta odnosnih entitetskih skupova/tipova.

Posmatrajmo entitetske skupove *nastavnik* i *predmet* na sl. 2.3. Vezni skup *predaje* sadrži skup veza između nastavnika i predmeta, pri čemu svaka veza (tipa *PREDAJE*) reprezentuje činjenicu da određeni nastavnik predaje određeni predmet. Veza $(n_1, p_1)$, označena kao instanca $r_1$, specifikuje da nastavnik $n_1$ predaje predmet $p_1$, veza $r_2 = (n_1, p_2)$ specifikuje da nastavnik $n_1$ predaje i predmet $p_2$, veza $r_3 = (n_2, p_3)$ specifikuje da nastavnik $n_2$ predaje predmet $p_3$ i tako dalje.




**Slika 2.3**: Veze i vezni skup

Entiteti koji učestvuju u vezi imaju specifične **uloge** u vezi. S obzirom na to da su najčešće veze između entiteta različitih entitetskih skupova, uloge entiteta u vezi su implicitno određene i nije ih potrebno posebno naznačiti (nazivi entitetskih skupova se mogu koristiti kao nazivi uloga). Međutim, ukoliko imamo veze između entiteta istog entitetskog skupa (**rekurzivne**[19] veze i vezni skupovi), onda je potrebno eksplicitno naznačiti ulogu poje-

[19] U literaturi se koristi i termin **refleksivne**.

dinih entiteta u vezi putem odgovarajućeg naziva uloge. Na primjer, ukoliko imamo veze između zaposlenih radnika koje reprezentuju odnos nadređenog rukovodioca (menadžer) i izvršioca (radnik), onda je određena veza reprezentovana uređenim parom entiteta, pri čemu su oba entiteta tipa *ZAPOSLENI*. Prvom entitetu (zaposlenom) u vezi se može pridružiti uloga *radnik*, dok se drugom entitetu (zaposlenom) u vezi može dodijeliti uloga *menadžer*. Svaka veza $(z_1, z_2)$ je tipa (*radnik*, *menadžer*), odnosno označava da je $z_1$ zaposleni (radnik) čiji je rukovodilac (menadžer) zaposleni $z_2$.

Tipovi veza (pa samim tim i veze) mogu imati svoje vlastite, **opisne atribute**. Na primjer, tip veze *UPISAO*, koji povezuje entitetske tipove *STUDENT* i *PREDMET*, može imati opisni atribut *BrojPrisustava*. Svaka veza tipa *UPISAO* reprezentuje da je određeni student upisao određeni predmet, a atribut veze *BrojPrisustava* može da sadrži podatak koliko je puta student prisustvovao nastavi za upisani predmet.

Svaka veza mora biti jedinstveno identifikovana entitetima koji participiraju u vezi (bez korištenja opisnih atributa). To znači da ne mogu postojati dvije (ili više) veza istog tipa između istih entiteta. Dakle, ako bi prethodno pomenuti tip veze *UPISAO* imao atribut *DatumPrisustva*, ne bismo mogli pretpostaviti postojanje višestrukih instanci veze navedenog tipa između studenta i predmeta (jer postoji samo jedna veza između studenta i predmeta) za reprezentaciju više različitih datuma prisustva nastavi studenta na odnosnom predmetu. Ako bismo željeli putem atributa *DatumPrisustva* predstaviti sve datume kada je neki student prisustvovao nastavi za određeni upisani predmet, opisni atribut *DatumPrisustva* bi morao da bude višeznačan (kako bi mogao da sadrži više vrijednosti o datumima prisustva). Isti tipovi entiteta (odnosno entitetski skupovi) mogu da participiraju u više tipova veza (veznih skupova), pa samim tim između istih entiteta može da egzistira više veza različitih tipova. Tako, na primjer, između entitetskih tipova *STUDENT* i *PREDMET*, pored tipa veze *UPISAO*, može da egzistira i tip veze *POLOŽIO*, koji definiše mogući skup veza između studenata i predmeta, pri čemu bi svaka instanca veze reprezentovala činjenicu da je neki student položio određeni predmet.

**Stepen tipa veze** (i odgovarajućeg veznog skupa) je određen brojem entitetskih tipova (i analogno entitetskih skupova) koji participiraju u tipu veze (veznom skupu). Najčešće su **binarne veze**, tj. veze koje povezuju dva entitetska tipa (entitetska skupa). Prethodno pomenuti vezni tipovi *UPISAO* i *POLOŽIO* su binarni tipovi, jer povezuju dva entitetska tipa: *STUDENT* i *PREDMET*. U veznim tipovima mogu da participiraju i više od dva entitetska tipa (u opštem slučaju *n-arni* tipovi veza).

Ako posmatramo univerzitet kao sistem za koji želimo da projektujemo bazu podataka i odgovarajući informacioni sistem, pored entitetskih tipova *NASTAVNIK* i *PREDMET*, bilo bi potrebno definisati i entitetski tip *FAKULTET*. Ako bismo željeli da predstavimo činjenicu da nastavnik predaje određeni predmet na nekom fakultetu, onda bi tip veze *PREDAJE* bio **ternarni tip veze** (umjesto binarnog tipa veze *PREDAJE* na sl. 2.3), koji bi povezivao entitetske tipove *NASTAVNIK*, *PREDMET* i *FAKULTET*. Ternarni tip veze *POSAO_NA_UNIVERZITETU*, u kojem participiraju entitetski tipovi *ZAPOSLENI*, *POSAO* i *ORG_JEDINICA*, definisao bi veze zaposlenih, poslova i organizacionih jedinica na univerzitetu. Na primjer, određeni zaposleni radnik bi mogao da u organizacionoj jedinici rektorat obavlja poslove prorektora, dok bi u drugoj organizacionoj jedinici (fakultetu) mogao da bude nastavnik. Ternarni tipovi veze su stepena tri.

### 2.1.2 Ograničenja na tipovima veza

Tip veze ne samo da definiše koji tipovi entiteta učestvuju u vezama, nego može da specifikuje i dodatna ograničenja koja se odnose na veze između entiteta. Sadržaj implementirane baze podataka, u svakom trenutku, mora da bude u saglasnosti sa specifikovanim ograničenjima. Najznačajnija ograničenja koja se odnose na veze između entiteta odnosnih entitetskih tipova jesu **ograničenja kardinalnosti mapiranja** (eng. *mapping cardinality*) i **učešća entiteta u vezi** (eng. *participation constraint*).

#### Kardinalnost mapiranja

Kardinalnost mapiranja specifikuje, za određeni tip veze, maksimalni broj veza (instanci veze) u kojima neki entitet odnosnog tipa može da učestvuje. Na primjer, neki entitet tipa *STUDENT* može da učestvuje u više veza tipa *UPISAO* (neki student može da upiše više predmeta). Ili, drugim riječima, neki entitet iz entitetskog skupa *student* može da učestvuje u više veza iz veznog skupa *upisao*. Može, takođe, da se kaže da neki entitet (student) iz entitetskog skupa *student* može da bude povezan sa više entiteta (predmeta) iz entitetskog skupa *predmet* vezama tipa *UPISAO*. Analogno bi se moglo zaključiti i da neki entitet iz entitetskog skupa *predmet* može da bude povezan sa više entiteta iz entitetskog skupa *student* (vezama tipa *UPISAO*).

Specifikacija kardinalnosti mapiranja je vrlo pogodna i korisna u binarnim tipovima veze. Neka su dati entitetski skupovi $a$ i $b$ i neka su njihovi entitetski tipovi $A$ i $B$, respektivno. Neka je $R$ tip veze između entitetskih

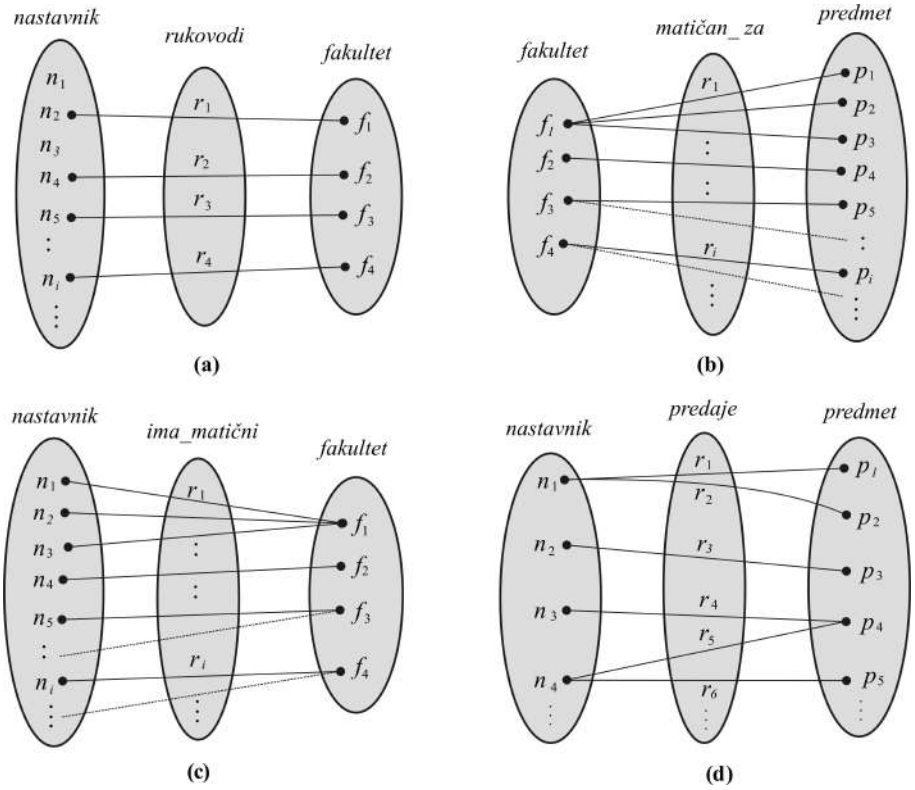
tipova $A$ i $B$. Moguće su slijedeće vrste ograničenja kardinalnosti mapiranja za binarni tip veze $R$ (predstavljene na sl. 2.4):

- **Jedan prema jedan** (eng. *one to one*), skraćeno **1:1**. Ovo ograničenje specifikuje da neki entitet iz entitetskog skupa $a$ može biti u vezi sa najviše jednim entitetom iz entitetskog skupa $b$, i obrnuto, neki entitet iz entitetskog skupa $b$ može biti u vezi sa najviše jednim entitetom iz entitetskog skupa $a$ (sl. 2.4a).
- **Jedan prema više** (eng. *one to many*), skraćeno **1:M**. Ovo ograničenje specifikuje da neki entitet iz entitetskog skupa $a$ može biti u vezi sa bilo kojim brojem (0 ili više) entiteta iz entitetskog skupa $b$, i da entitet iz entitetskog skupa $b$ može biti u vezi sa najviše jednim entitetom iz entitetskog skupa $a$ (sl. 2.4b).
- **Više prema jedan** (eng. *many to one*), skraćeno **M:1**. Ovo ograničenje specifikuje da neki entitet iz entitetskog skupa $a$ može biti u vezi sa najviše jednim entitetom iz entitetskog skupa $b$, i da entitet iz entitetskog skupa $b$ može biti u vezi sa bilo kojim brojem (0 ili više) entiteta iz entitetskog skupa $a$ (sl. 2.4c).
- **Više prema više** (eng. *many to many*), skraćeno **M:M**. Ovo ograničenje specifikuje da neki entitet iz entitetskog skupa $a$ može biti u vezi sa bilo kojim brojem (0 ili više) entiteta iz entitetskog skupa $b$ i obrnuto, da entitet iz entitetskog skupa $b$ može biti u vezi sa bilo kojim brojem (0 ili više) entiteta iz entitetskog skupa $a$ (sl. 2.4d).

Napomena: U svim prethodnim varijantama, ograničenja kardinalnosti mapiranja ne specifikuju da je učešće entiteta u vezi obavezno, tj. moguće je postojanje entiteta iz entitetskih skupova $a$ i $b$ koji ne participiraju ni u jednoj vezi tipa $R$.

Pretpostavimo da tip veze *RUKOVODI* u univerzitetskoj bazi podataka definiše veze nastavnika koji rukovode fakultetom (dekana) i fakulteta kojim oni rukovode (sl. 2.4a). Neki nastavnik može rukovoditi (biti dekan) najviše na jednom fakultetu, tj. može biti u vezi ovog tipa najviše sa jednim fakultetom. Slično, neki fakultet može imati samo jednog dekana, tj. može biti u vezi ovog tipa sa samo jednim nastavnikom. Iz prethodnog zaključujemo da je kardinalnost mapiranja veze tipa *RUKOVODI* (odnosno veznog skupa *rukovodi*) 1:1. Primijetimo da ima nastavnika iz entitetskog skupa *nastavnik* koji ne učestvuju u ovom tipu veze (rukovođenja fakultetom).

Posmatrajmo tip veze *MATIČAN_ZA*, koji definiše veze fakulteta i predmeta za koje je fakultet matičan (sl. 2.4b). Neki fakultet je matičan za više predmeta, a za neki predmet matičan je samo jedan fakultet. Znači, fakultet može biti u vezi sa više predmeta, dok neki predmet može biti u vezi




**Slika 2.4**: Kardinalnost mapiranja: a) jedan prema jedan, b) jedan prema više, c) više prema jedan, d) više prema više

samo sa jednim fakultetom. Kardinalnost mapiranja ovog tipa veze (veznog skupa) je 1:M, posmatrano od strana entitetskog tipa *FAKULTET* (odnosno entitetskog skupa *fakultet*) prema entitetskom tipu *PREDMET* (odnosno entitetskom skupu *predmet*).

Neka tip veze *IMA_MATIČNI* definiše veze nastavnika i fakulteta koji je matični fakultet nastavnika (sl. 2.4c). Nastavnik može predavati na više fakulteta univerziteta, ali mu je jedan fakultet matičan – fakultet na kojem ima radni raspored/radno mjesto. Neki nastavnik ima samo jedan matični fakultet (na kojem ima radno mjesto), dok je neki fakultet matičan za više nastavnika. Znači, nastavnik može biti u vezi tipa *IMA_MATIČNI* samo sa jednim fakultetom, dok jedan fakultet može biti u vezi sa više nastavnika (kojima je on matičan). Kardinalnost mapiranja ovog tipa veze (veznog skupa) je M:1, posmatrano od strane entitetskog tipa *NASTAVNIK* (odnosno entitetskog skupa *nastavnik*) prema entitetskom tipu *FAKULTET* (odnosno entitetskom skupu *fakultet*).

Neka je u univerzitetskoj bazi podataka specifikovan binarni tip veze *PREDAJE*, koji definiše veze nastavnika i predmeta koje oni predaju (sl. 2.4d). Neki nastavnik može predavati više (M) predmeta, a pretpostavimo i da jedan predmet može predavati više (M) nastavnika. Dakle, jedan nastavnik može biti u vezi sa više predmeta, a neki predmet može biti u vezi sa više nastavnika. Iz prethodnog zaključujemo da je kardinalnost mapiranja veze tipa *PREDAJE* (veznog skupa *predaje*) M:M.

### Ograničenje učešća entiteta u vezi

Ograničenje učešća entiteta u vezi specifikuje, za određeni tip veze, minimalni broj veza (relacionih instanci) u kojima neki entitet odnosnog entitetskog tipa (skupa) mora da učestvuje. To znači da i postojanje entiteta može biti uslovljeno učešćem entiteta u nekom, minimalnom broju veza. U MOV modelu moguće je specifikovati dvije vrste ograničenja na učešće entiteta u vezi: **totalno** i **parcijalno**. Učešće nekog entitetskog tipa (skupa) u veznom tipu (skupu) je totalno ako svaki entitet tog entitetskog tipa (skupa) mora učestvovati barem u jednoj vezi iz veznog skupa. Učešće nekog entitetskog tipa (skupa) u veznom tipu (skupu) je parcijalno ako samo neki entiteti tog entitetskog tipa (skupa) učestvuju u vezama iz veznog skupa. Ograničenje za totalnost učešća se naziva i **egzistencijalna zavisnost** (eng. *existence dependency*), jer entitet entitetskog tipa čije je učešće u vezi totalno, ne može da egzistira bez postojanja veze sa entitetom iz drugog entitetskog skupa, a time i samog tog drugog entiteta.

Primijetimo da je učešće entitetskog skupa *nastavnik* u veznom skupu *rukovodi* parcijalno, dok je učešće entitetskog skupa *fakultet* totalno (sl. 2.4a)[20].

U veznom skupu *predaje* učešće entitetskih skupova *nastavnik* i *predmet* je parcijalno sa obje strane (sl. 2.4d), ako usvojimo slijedeće pretpostavke o realnom sistemu:

- Neki predmet može da bude u nastavnom planu a da se ne predaje (npr. predmet je izborni i nijedan student ga nije izabrao).
- Neki nastavnik može u nekom periodu da bude bez zaduženja u nastavi, tj. da ne predaje (npr. trenutno se nalazi na postdoktorskom studiju ili usavršavanju u inostranstvu).

[20] Analogno, ograničenja učešća entitetskih tipova u tipu veze *RUKOVODI* su parcijalno učešće za entitetski tip *NASTAVNIK* i totalno za entitetski tip *FAKULTET*.

### 2.1.3 Ključevi

Entiteti entitetskog skupa se međusobno razlikuju, odnosno svaki entitet entitetskog skupa je jedinstven. To implicira da ne mogu da postoje dva entiteta koja imaju iste vrijednosti na svim atributima. Važno ograničenje na entitetskim skupovima, kojim se specifikuje jedinstvenost entiteta entitetskog skupa, jeste ključ. Koncept ključeva takođe omogućava specifikaciju jedinstvenosti veza veznog skupa.

#### Ključevi entitetskog tipa (skupa)

**Superključ** (eng. *superkey*) je skup atributa entitetskog tipa, na kojem svaki entitet odnosnog entitetskog skupa ima jedinstvenu vrijednost. Pri tome se skup atributa koji čine superključ, posmatra kao cjelina. Pošto na atributima koji čine superključ svaki entitet entitetskog skupa ima jedinstvenu vrijednost, superključ se može koristiti za jedinstvenu identifikaciju entiteta. Ako znamo vrijednost superključa, možemo poređenjem te vrijednosti sa vrijednostima odgovarajućih atributa entiteta, identifikovati entitet iz entitetskog skupa koji ima traženu vrijednost superključa.

Na primjer, superključ entitetskog tipa *STUDENT* je $\{JMB\}$. Dva entiteta entitetskog tipa *STUDENT* ne mogu imati istu vrijednost na tom atributu (isti Jedinstveni Matični Broj). Međutim, superključ je i kombinacija atributa *JMB*, *ImeStudenta*, kao i *JMB*, *GodinaStudija*, *StudijskiProgram*. Svaki nadskup superključa, je takođe superključ. Od interesa je **kandidatski ključ**[21] – superključ koji nema pravi podskup[22] atributa, koji je takođe superključ. Neki entitetski tip može da ima više kandidatskih ključeva. Na primjer, ako bi se kao atribut tipa *STUDENT* vodio i *BrojLK* (broj lične karte), onda bi kandidatski ključevi bili i $\{JMB\}$ i $\{BrojLK\}$. Nekad ključ/kandidatski ključ ne čini samo jedan atribut, već kombinacija atributa.

Jedan od kandidatskih ključeva projektant baze selektuje kao **primarni ključ**, odnosno kao osnovni ključ za jedinstvenu identifikaciju entiteta u entitetskom skupu. Superključ, kandidatski ključ i primarni ključ su svojstva entitetskih tipova i skupova, a ne pojedinih entiteta. Ova ograničenja specifikovana za određeni entitetski tip moraju da važe na svakom entitetskom skupu i za sve instance bilo kog entitetskog skupa odnosnog tipa. Za

[21] U literaturi se često koristi termin **ključ** (eng. *key*) ako postoji samo jedan minimalni superključ za entitetski tip. Ako postoji više različitih minimalnih superključeva, onda se koristi termin kandidatski ključ.

[22] $A$ je pravi podskup od $B$ ($A \subset B$), ako je $A \subseteq B$ i $A \neq B$.

određene entitetske tipove nije moguće specifikovati superključ, pa prema tome ni primarni ključ. O ovom, slabom entitetskom tipu, biće riječi u dijelu 2.1.5.

Specifikacija kandidatskih ključeva zahtijeva pažnju. Ovo ograničenje mora da odražava realna svojstva odgovarajućih entitetskih tipova/skupova. Na primjer, pretpostavka da ne postoje dva studenta sa istim imenom nije realna, pa u realnom sistemu atribute koji reprezentuju ime ne treba uzeti kao kandidatski ključ.

**Ključevi tipa veze (veznog skupa)**

Prethodno je pomenuto ograničenje da ne može postojati više veza istog tipa između istih entiteta, i da svaka veza mora biti jedinstveno identifikovana entitetima koji učestvuju u vezi. Koncept ključeva omogućava specifikaciju ovih ograničenja, kao i identifikaciju veza od interesa.

Neka je $R$ tip veze koji povezuje entitetske tipove $E_1, E_2, \ldots, E_n$. Pretpostavimo da su nazivi svih atributa koji čine primarne ključeve entitetskih tipova jedinstveni i da su svi entitetski tipovi koji učestvuju u vezi $R$ različiti. Neka $PK(E_i)$ označava skup atributa koji čine primarni ključ entitetskog tipa $E_i$. Ako vezni tip $R$ nema vlastite atribute, onda skup atributa $PK(E_1) \cup PK(E_2) \cup \cdots \cup PK(E_n)$ određuje individualne veze tipa $R$. Veza koja povezuje entitete $e_1 \in E_1$, $e_2 \in E_2$, ... , $e_n \in E_n$ je određena vrijednostima primarnih ključeva entiteta $e_1, e_2, \ldots, e_n$, jer primarni ključevi jedinstveno identifikuju entitete koji učestvuju u vezi. Pošto su vrijednosti entiteta na atributima primarnog ključa jedinstvene, i veza između odnosnih entiteta $e_1, e_2, \ldots, e_n$ je jedinstvena. Iz prethodnog slijedi da je $PK(E_1) \cup PK(E_2) \cup \cdots \cup PK(E_n)$ superključ tipa veze.

Ako tip veze $R$ ima svoje vlastite atribute $A_1, A_2, \ldots, A_n$, onda skup atributa $PK(E_1) \cup PK(E_2) \cup \cdots \cup PK(E_n) \cup \{A_1, A_2, \ldots, A_n\}$ određuje individualne veze tipa $R$. Slično prethodnom razmatranju, možemo zaključiti da je i u ovom slučaju $PK(E_1) \cup PK(E_2) \cup \cdots \cup PK(E_n)$ superključ tipa veze.

U slučajevima da nazivi atributa primarnih ključeva entitetskih tipova koji učestvuju u tipu veze $R$ nisu jedinstveni, potrebno ih je preimenovati. Preimenovanje se jednostavno može realizovati kombinovanjem naziva pripadajućeg entitetskog tipa i atributa koji nije jedinstven. U slučaju da entitetski tip u vezi učestvuje rekurzivno, onda se mogu koristiti nazivi uloga za jedinstveno imenovanje atributa.

Pošto je $PK(E_1) \cup PK(E_2) \cup \cdots \cup PK(E_n)$ superključ tipa veze, može se postaviti pitanje šta je kandidatski/primarni ključ tipa veze. Primarni

ključ zavisi od kardinalnosti mapiranja. Za binarne tipove veza, ukoliko je kardinalnost mapiranja 1:M ili M:1, onda je veza identifikovana primarnim ključem entitetskog tipa sa M strane veze. Na primjer, u vezi tipa *IMA_MATIČNI*, jedan nastavnik može biti u vezi samo sa jednim (matičnim) fakultetom. Superključ veze je $PK(NASTAVNIK) \cup PK(FAKULTET)$. Svaka veza je određena vrijednostima primarnih ključeva entiteta fakultet i nastavnik, koji učestvuju u vezi. Ako imamo tako određen skup veza, onda je vrijednost atributa $PK(NASTAVNIK)$ u skupu veza jedinstvena, jer nastavnik učestvuje samo u jednoj vezi. Iz navedenog slijedi da svaku vezu jedinstveno identifikuje atribut $PK(NASTAVNIK)$, pa je to i primarni ključ veze.

Slično možemo zaključiti da su kandidatski ključevi u tipu veze sa kardinalnošću mapiranja 1:1 primarni ključevi odnosnih entitetskih tipova. Bilo koji od njih može da bude selektovan kao primarni ključ tipa veze.

U vezama tipa M:M, primarni ključ veze je unija primarnih ključeva entitetskih tipova koji učestvuju u vezi. U tipu veze *PREDAJE* sa kardinalnošću mapiranja M:M, superključ veze je $PK(NASTAVNIK) \cup PK(PREDMET)$. Svaka veza je određena vrijednostima primarnih ključeva entiteta nastavnik i predmet, koji učestvuju u vezi. Bilo koji nastavnik ili predmet može se pojaviti u više veza. Iz tog razloga, u skupu veza vrijednosti na atributu $PK(NASTAVNIK)$ ili atributu $PK(PREDMET)$ nisu jedinstvene. Jedinstvena je kombinacija vrijednosti atributa $PK(NASTAVNIK)$ i $PK(PREDMET)$, pa ta kombinacija identifikuje vezu, odnosno primarni ključ tipa veze *PREDAJE* je $PK(NASTAVNIK) \cup PK(PREDMET)$.

U nebinarnim vezama, ukoliko svaki entitetski tip učestvuje u vezi sa kardinalnošću M, onda je primarni ključ veze unija primarnih ključeva entitetskih tipova koji učestvuju u vezi. Ako jedan entitetski tip učestvuje u vezi sa kardinalnošću 1, a ostali sa kardinalnošću M, onda je primarni ključ veze unija primarnih ključeva entitetskih tipova koji u vezi učestvuju sa kardinalnošću M. U suprotnom, postoji problem jednoznačne interpretacije kardinalnosti mapiranja. Na primjer, ako je veza tipa $R$ kvaternarna (uključuje 4 entitetska tipa), i ako uključuje entitetske tipove $A$, $B$, $C$ i $D$, pri čemu je kardinalnost učešća entitetskih tipova $A$ i $B$ u vezi M, a kardinalnost učešća entitetskih tipova $C$ i $D$ u vezi 1, moguće su dvije interpretacije: da jedan entitet tipa $C$ može biti u vezi samo sa jednom kombinacijom entiteta tipa $A$, $B$ i $D$ (primarni ključ je unija primarnih ključeva entitetskih tipova $A$, $B$ i $D$), ili da neka kombinacija entiteta tipa $A$ i $B$ može biti u vezi sa samo jednim parom entiteta tipa $C$ i $D$ (primarni ključ je unija primarnih ključeva entitetskih tipova $A$ i $B$).

S obzirom na to da su veze implicitno definisane primarnim ključevima entiteta koji učestvuju u vezi, primarne ključeve entitetskih tipova koji učestvuju u vezi ne treba modelovati kao atribute tipa veze. Slično, u MOV modelu ne treba modelovati primarni ključ nekog entitetskog tipa kao atribut drugog entitetskog tipa (za referenciranje na prvi entitetski tip)[23].

### 2.1.4 MOV dijagrami

MOV model konkretnog sistema, koji reprezentuje sveukupnu logičku strukturu baze podataka, predstavlja se u grafičkoj formi putem MOV dijagrama. Ovi dijagrami su intuitivni i jednostavni za razumijevanje. Na njima se grafički predstavljaju entiteti i njihovi atributi, veze između entiteta i ograničenja. Osnovne komponente MOV dijagrama su:

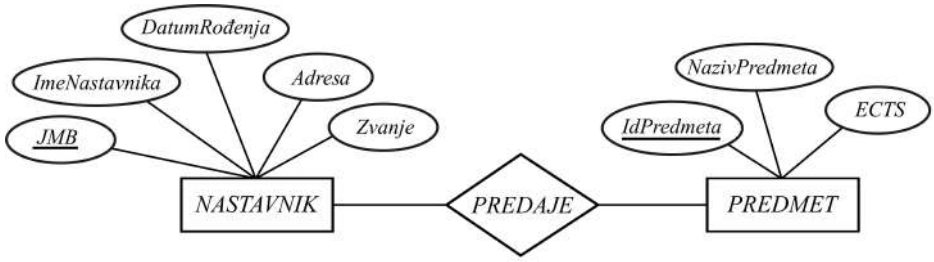
- **Pravougaonici**, reprezentuju tipove entiteta (odnosno entitetske skupove). U pravougaonik se upisuje naziv odnosnog tipa entiteta.
- **Elipse**, reprezentuju atribute. U elipsu se upisuje naziv odnosnog atributa.
- **Rombovi**, reprezentuju tipove veza (vezne skupove). U romb se upisuje naziv odnosnog tipa veze.
- **Linije**, povezuju atribute sa pripadajućim tipom entiteta ili tipom veze, i tipove entiteta sa tipovima veza.
- **Duple elipse** (elipse sa duplom ivicom), reprezentuju višeznačne atribute.
- **Elipse sa isprekidanom linijom**, reprezentuju izvedene atribute.
- **Duple linije**, reprezentuju totalnu participaciju tipa entiteta u tipu veze.

Posmatrajmo MOV dijagram na sl. 2.5 koji reprezentuje dva entitetska tipa *NASTAVNIK* i *PREDMET* povezana binarnom vezom *PREDAJE*.

Entitetski tip *NASTAVNIK* ima atribute *JMB* (jedinstveni matični broj), *ImeNastavnika*, *DatumRođenja*, *Adresa* i *Zvanje*. Entitetski tip *PREDMET* ima atribute *IdPredmeta*, *NazivPredmeta* i *ECTS*[24]. Primarni ključevi entitetskih tipova *NASTAVNIK* i *PREDMET* su {*JMB*} i {*IdPredmeta*}, respektivno. Nazivi atributi koji čine primarni ključ, na MOV dijagramima se ispisuju podvučeno. Kardinalnost mapiranja tipa veze *PREDAJE* sa sl. 2.5 je M:M (i odgovara predstavi veza entiteta tipa *NASTAVNIK* i

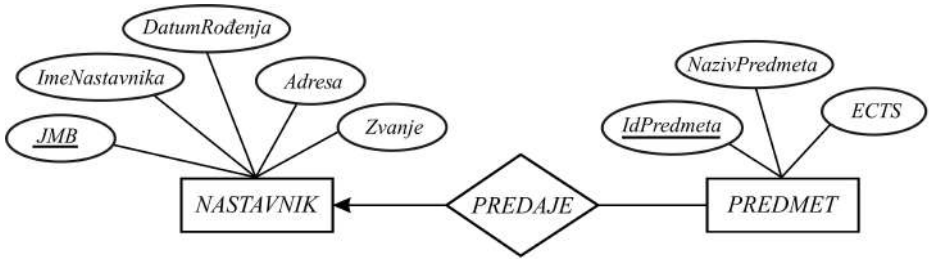
[23] U nekim komercijalnim alatima za modelovanje se odstupa od ovog pravila (npr. ER-win), pa se pri modelovanju 1:M ili 1:1 tipa veze automatski vrši migracija primarnog ključa *parent* entiteta u *child* entitet kao atribut stranog ključa.

[24] *European Credit Transfer and Accumulation System*




**Slika 2.5**: MOV dijagram veze entitetskih tipova *NASTAVNIK* i *PREDMET*

*PREDMET* na sl. 2.4d). Ukoliko bi pretpostavili da jedan nastavnik može da predaje više predmeta, a da jedan predmet predaje samo jedan nastavnik, onda bi kardinalnost mapiranja tog tipa veze bila 1:M (od *NASTAVNIK* ka *PREDMET*). U tom slučaju, predstava veze entitetskih tipova *NASTAVNIK* i *PREDMET*, preko tipa veze *PREDAJE*, bila bi kao na sl. 2.6.

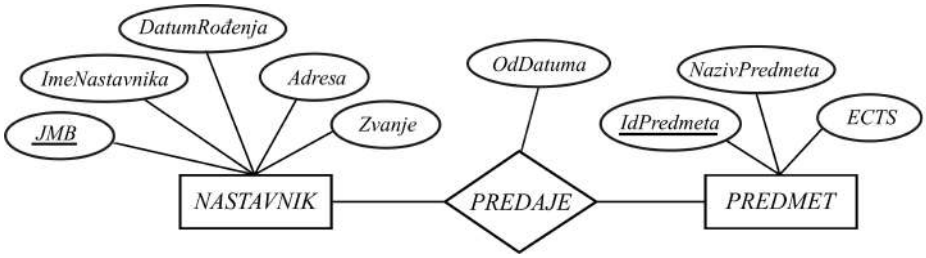



**Slika 2.6**: 1:M tip veze na MOV dijagramu

Na MOV dijagramima, za specifikaciju učešća entiteta nekog entitetskog tipa u vezi sa kardinalnošću M koristi se povezivanje entitetskog tipa sa tipom veze neusmjerenom linijom, dok se za specifikaciju učešća entiteta nekog entitetskog tipa u vezi sa kardinalnošću 1 koristi usmjerena linija od tipa veze ka tom entitetskom tipu.

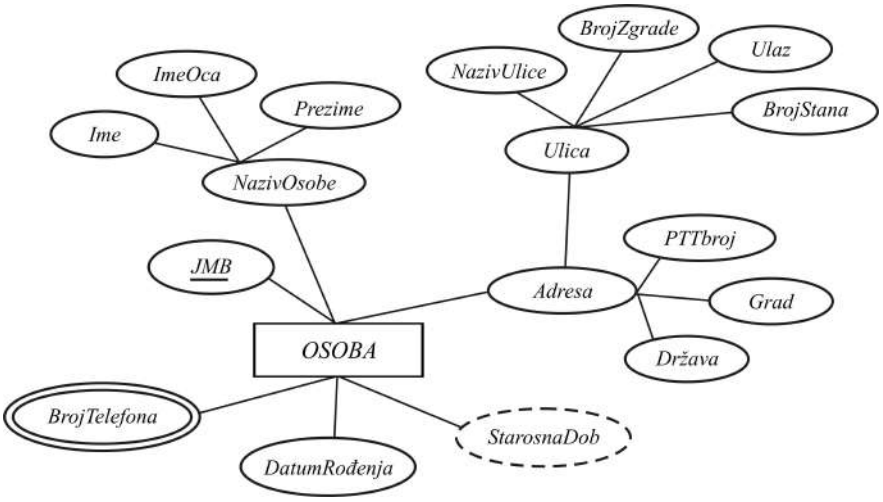
Ako bi kardinalnost mapiranja tipa veze *PREDAJE* bila 1:1, onda bi obje linije od tipa veze *PREDAJE* ka entitetskim tipovima *NASTAVNIK* i *PREDAJE* bile usmjerene (sa strelicom ka entitetskim tipovima). Ako bi kardinalnost mapiranja tipa veze *PREDAJE* bila M:1, onda bi na dijagramu entitetski tip *NASTAVNIK* bio povezan sa tipom veze *PREDAJE* neusmjerenom linijom, a entitetski tip *PREDMET* bi sa tipom veze *PREDAJE* bio povezan usmjerenom linijom (sa strelicom ka entitetskom tipu *PREDMET*).

Ako tip veze ima svoje vlastite – opisne atribute, onda se ti atributi na MOV dijagramu povezuju sa veznim tipom. Na primjer, ako bi tip veze *PREDAJE* imao atribut *OdDatuma*, koji bi sadržavao informaciju od kada nastavnik predaje predmet, onda bi MOV dijagram bio kao sl. 2.7.



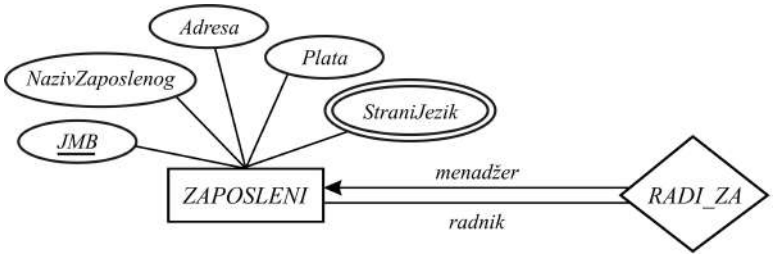

**Slika 2.7**: Tip veze sa vlastitim atributima

Slika 2.8 ilustruje predstavu kompozitnih, izvedenih i višeznačnih atributa na MOV dijagramu. Kompozitni atributi entitetskog tipa *OSOBA* jesu *Adresa* i *NazivOsobe*. Jedna komponenta kompozitnog atributa *Adresa* je i sama kompozitna (*Ulica* ima potkomponente *NazivUlice*, *BrojZgrade*, *Ulaz* i *BrojStana*). Atribut *StarosnaDob* je izvedeni atribut, jer se može izračunati iz tekućeg datuma i atributa *DatumRođenja*. Atribut *BrojTelefona* je višeznačni atribut (predstavljen duplom elipsom), jer jedna osoba može imati više vrijednosti ovog atributa (više brojeva telefona).



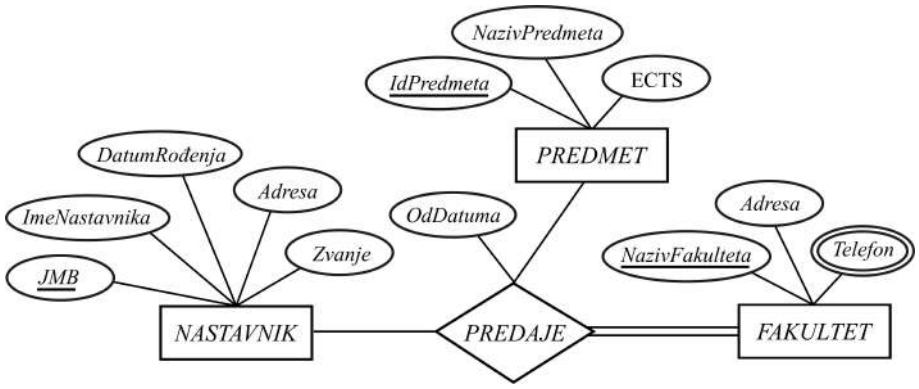

**Slika 2.8**: Kompozitni, izvedeni i višeznačni atributi

Uloge entitetskih tipova u vezi, na MOV dijagramima se specifikuju nazivom uloge uz liniju koja spaja entitetski tip i tip veze. Za primjer naveden u dijelu 2.1.1, koji se odnosi na rekurzivnu vezu između zaposlenih koja povezuje radnike i njihove rukovodioce (menadžere), MOV dijagram sa specifikacijom uloga zaposlenih u vezi predstavljen je na sl. 2.9. Dijagramom se specifikuje da jedan (zaposleni) radnik može da radi za samo jednog menadžera (kardinalnost 1 na strani uloge *menadžer*, indicirana strelicom), dok za jednog menadžera može da radi više radnika (kardinalnost M na strani uloge *radnik*, indicirana neusmjerenom linijom).




**Slika 2.9**: Specifikacija uloga u MOV dijagramima

Veze stepena višeg od 2 (nebinarne veze), na MOV dijagramima se predstavljaju istim grafičkim simbolom za tip veze (romb) povezanim sa grafičkim simbolima entitetskih tipova koji učestvuju u vezi. Ako želimo da predstavimo činjenicu da nastavnik predaje neki predmet na nekom fakultetu, onda možemo vezu *PREDAJE* na MOV dijagramu modelovati ternarnom vezom kao na sl. 2.10 (umjesto binarnom vezom na sl. 2.7).




**Slika 2.10**: Ternarni tip veze

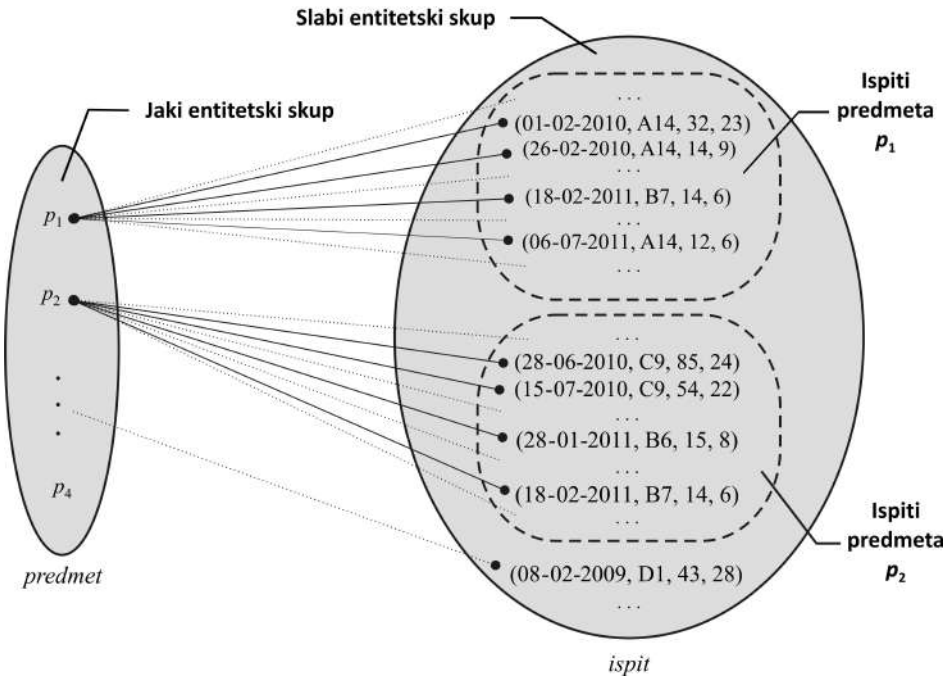
Dupla linija koja povezuje entitetski tip *FAKULTET* s tipom veze *PREDAJE* označava totalnu participaciju entiteta tipa *FAKULTET* u ternarnoj vezi *PREDAJE*. To znači da svaki fakultet učestvuje u barem jednoj vezi (fakultet ne može da se registruje i počne sa radom ako nema definisane predmete i nastavnike zadužene da izvode nastavu na tom fakultetu). Primijetimo da su veze na strani entitetskih tipova *NASTAVNIK* i *PREDMET* parcijalne. To znači da smo modelom predvidjeli da neki nastavnik u nekom trenutku može biti bez zaduženja u nastavi, ili da imamo predmet na nekom fakultetu za koji nije definisano zaduženje nastavnika.

### 2.1.5 Slabi entitetski tip

Neki entitetski tipovi nemaju atribute koji čine primarni ključ, odnosno ne mogu se identifikovati atributi na kojima će svaki entitet entitetskog skupa tog tipa imati jedinstvenu vrijednost. Takvi entitetski tipovi se nazivaju **slabi entitetski tipovi**, a entitetski skupovi tog tipa **slabi entitetski skupovi**. Entitetski tip koji ima atribute koji čine primarni ključ se naziva **jaki entitetski tip** (i analogno tome, odnosni entitetski skupovi su **jaki entitetski skupovi**).

Za ilustraciju, posmatrajmo entitetske tipove *PREDMET* i *ISPIT* u modelu univerzitetskog sistema. Prethodno smo definisali da su atributi entitetskog tipa *PREDMET*: *IdPredmeta*, *NazivPredmeta*, i *ECTS*. Za predmete se organizuju ispiti u određenim terminima. Za jedan predmet se organizuje više ispita u različitim terminima (sl. 2.11). *ISPIT* karakterišu atributi: *DatumIspita*, *Lokacija*, *Izašlo* i *Položilo*. *Lokacija* je oznaka za mjesto (sala) gdje se održava ispit, atribut *Izašlo* sadrži informaciju o broju studenata koji su izašli na ispit, dok *Položilo* sadrži informaciju o broju studenata koji su položili ispit.

Kao što se vidi sa slike, ispiti iz pojedinih predmeta su pridruženi odnosnom predmetu. Ako predvidimo mogućnost da se neki (pismeni) ispiti mogu održati iz više predmeta u jednoj sali, da na neke ispite iz tih predmeta može izaći i položiti isti broj studenata, onda u entitetskom skupu *ispit* mogu postojati entiteti koji imaju iste vrijednosti na svim atributima. To znači da entitetski tip *ISPIT* nema primarni ključ, pa je to slabi entitetski tip (nijedan skup atributa entitetskog tipa *ISPIT* ne identifikuje jedinstveno entitete iz entitetskog skupa tog tipa). U slabom entitetskom skupu *ispit* iste vrijednosti na svim atributima (18-02-2011, B7, 14, 6) imaju dva entiteta (u realnom sistemu ovakva situacija je rijetka, ali moguća).




**Slika 2.11**: Slabi i jaki entitetski skupovi

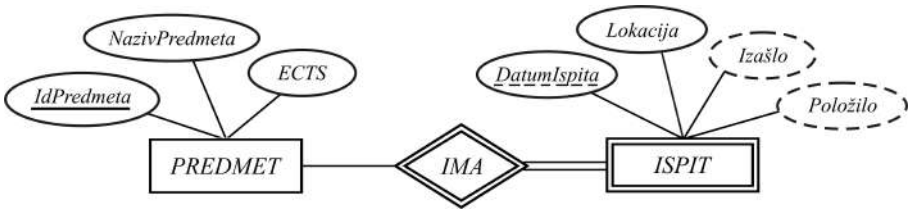
Entiteti slabog entitetskog skupa se mogu identifikovati na bazi njihove povezanosti sa entitetima jakog entitetskog skupa. Zato se takav jaki entitetski skup (tip) naziva i **identifikujući/vlasnički entitetski skup (tip)** (eng. *identifying/owner entity set* (*type*)),[25] a tip veze koji povezuje slabi i jaki entitetski tip se naziva **identifikujući tip veze**. Dakle, ako posmatramo samo entitete slabog entitetskog skupa *ispit*, koji su povezani sa predmetom $p_1$ (entitetom jakog entitetskog skupa), možemo uočiti da se oni međusobno razlikuju na određenom skupu atributa koji se naziva **diskriminator** ili **parcijalni ključ**. U prethodnom primjeru, diskriminator je atribut *DatumIspita*. Dakle, diskriminator je skup atributa koji jedinstveno identifikuju entitete slabog entitetskog skupa koji su povezani sa istim entitetom jakog entitetskog skupa. Specifično za slabe entitetske skupove, primarni ključ slabog entitetskog skupa se može formirati kombinacijom primarnog ključa jakog entitetskog skupa i diskriminatora.

[25] Za *identifikujući entitetski tip* koriste se i termini *roditeljski entitetski tip* (eng. *parent entity type*) i *dominantni entitetski tip* (eng. *dominant entity type*). Za *slabi entitetski tip* se koriste i termini *podređeni entitetski tip* (eng. *subordinate entity type*) i *entitetski tip potomak* (eng. *child entity type*).

Svaki entitet slabog entitetskog skupa mora biti povezan samo sa jednim entitetom iz jakog entitetskog skupa. Ne može postojati nijedan ispit ako nije povezan sa konkretnim predmetom (ako se ne odnosi na konkretan predmet). Kardinalnost mapiranja identifikujućeg tipa veze je više prema jedan od slabog ka jakom entitetskom tipu (više ispita se organizuje za jedan predmet). Participacija slabog entitetskog tipa u vezi je totalna (svaki ispit mora biti povezan sa nekim predmetom). S obzirom na prethodno navedeno, identifikujući tip veze ne treba da sadrži deskriptivne atribute, pošto se ti atributi mogu pridružiti slabom entitetskom tipu.

U skladu sa prethodno rečenim, slabi entitetski skup ne može ni da egzistira ukoliko ne postoji jaki entitetski skup sa kojim je u vezi, odnosno kaže se još i da je slabi entitetski skup (tip) egzistencijalno zavisan od jakog entitetskog skupa (tipa) sa kojim je povezan identifikujućim tipom veze. Egzistencijalna zavisnost nekog entitetskog tipa ne implicira automatski da je taj entitetski tip slab. Na primjer, entitetski tip *LICNAKARTA* je egzistencijalno zavisan od entitetskog tipa *OSOBA* (konkretna lična karta ne može da egzistira bez osobe kojoj je izdata), ali *LICNAKARTA* nije slabi entitetski tip, jer ima vlastiti primarni ključ (*BrojLK*).

Na MOV dijagramima se slabi entitetski tipovi predstavljaju pravougaonikom sa duplim linijama, a identifikujući tip veze rombom sa duplim linijama. Nazivi atributa koji čine diskriminator podvlače se isprekidanom linijom (sl. 2.12). Dupla linija koja povezuje slabi entitetski tip i identifikujući tip veze indicira totalnu participaciju slabog entitetskog tipa u vezi.




**Slika 2.12**: MOV dijagram sa slabim i jakim entitetskim tipovima

Slabi entitetski tip može biti povezan sa više jakih entitetskih tipova identifikujućim tipovima veze. Dalje, slabi entitetski tip može da učestvuje u drugoj vezi kao jaki entitetski tip. Identifikujući tip veze može takođe biti stepena većeg od dva. Pošto su disjunktni podskupovi slabih entitetskih skupova povezani samo sa jednim entitetom jakog entitetskog skupa, slabi entitetski tip se može modelovati kao kompozitni, višeznačni atribut jakog entitetskog tipa. O ovim opcijama više riječi biće u dijelu 2.1.8.

### 2.1.6 Prošireni MOV model

Osnovni MOV model [9], predložen 1976. godine (Peter P. Chen), omogućavao je pogodno i adekvatno modelovanje strukture i svojstava entiteta realnog svijeta za mnoge aplikativne sisteme. U kompleksnim sistemima postojala je potreba za predstavljanjem određenih specifičnih svojstava entiteta i odnosa između entiteta na pogodniji i semantički ekspresivniji način, za šta je bilo potrebno proširenje osnovnog modela dodatnim konceptima. Prošireni MOV model[26] uključuje, pored koncepata osnovnog modela, i dodatne koncepte entitetskih tipova višeg/nižeg nivoa (superklase/potklase), specijalizacije/generalizacije, nasljeđivanje atributa i veza, unije te agregacije.

#### Specijalizacija/Generalizacija

U nekim entitetskim skupovima možemo uočiti podgrupe entiteta koji se razlikuju po nekim specifičnim svojstvima u odnosu na druge entitete iz entitetskog skupa. Te specifičnosti je najčešće poželjno i potrebno prezentovati eksplicitno. Ako posmatramo entitetski skup *osoba* na univerzitetu, za koji želimo da vodimo određene podatke od interesa (atribute *JMB*, *NazivOsobe*, *Adresa* i *DatumRođenja*), možemo uočiti da postoje pojedine podgrupe (na primjer *zaposleni* i *studenti*) koje imaju neka specifična svojstva/karakteristike. Posebna svojstva navedenih podgrupa možemo reprezentovati **specifičnim (lokalnim) atributima**. Na primjer, za zaposlenu osobu relevantno svojstvo je *Plata*, dok su za studenta bitni *AdresaStudiranja*, *CiklusStudija*, *GodinaStudija*, *StudijskiProgram*. Pošto su uloga i interakcije ovih entitetskih podgrupa sa drugim entitetskim skupovima u mnogim situacijama specifični, potrebno je istaći i reprezentovati ove podgrupe entiteta posebno.

Proces identifikacije i reprezentacije podgrupa u okviru nekog entitetskog skupa naziva se **specijalizacija**. Specijalizacija predstavlja postupak koji se primjenjuje od 'vrha nadolje' (eng. *top-down*), jer se, polaženjem od nekog postojećeg, generalnijeg entitetskog skupa (tipa), analizom dolazi do specijalizovanih entitetskih skupova (tipova). U hijerarhijskoj predstavi, generalniji **entitetski skupovi (tipovi)** su **višeg nivoa**, dok su specijalizovani **entitetski skupovi (tipovi) nižeg nivoa**. Za entitetske skupove (tipove) višeg nivoa koristi se i termin **superklasa**, a za entitetske skupove (tipove) nižeg nivoa termin **potklasa**.

[26] Engleski termini koji se koriste su *Extended* ili *Enhanced E-R model* (*EER model*).

Potklase imaju sve atribute superklase, i dodatno svoje specifične atribute. Drugim riječima, potklase, pored svojih specifičnih atributa, nasljeđuju sve atribute superklase. Analogno nasljeđivanju atributa, potklase nasljeđuju i tipove veza u kojima superklasa učestvuje. Nasljeđivanje atributa i tipova veza superklase od strane potklase predstavlja **nasljeđivanje tipa** (eng. *type inheritance*). Entiteti koji pripadaju nekoj potklasi, istovremeno su članovi i odnosne superklase, dok obrnuto ne mora da važi. Potklasa, sa vlastitim i naslijeđenim atributima i tipovima veza, može se smatrati i novim entitetskim tipom. Primijetimo da je entitet potklase ujedno i član superklase, odnosno radi se o jednom objektu realnog svijeta koji kao član potklase ima *specijalizovanu ulogu* (na primjer, neka osoba može biti u ulozi *STUDENT* ili u ulozi *ZAPOSLENI*). Na taj način koncept specijalizacije implicitno dopušta da neki entiteti imaju višestruke tipove.

Za implementaciju specijalizacije moguće je koristiti više pristupa, što je detaljno objašnjeno u dijelu 3.4.1. Na primjer, jedan pristup koristi jedan zapis u bazi podataka koji reprezentuje entitet kao člana superklase, a drugi zapis reprezentuje specifična svojstva entiteta kao člana potklase (tzv. *vertikalni princip*). Takođe, može se koristiti implementaciona opcija gdje se koristi jedan zapis za reprezentaciju svojstava entiteta i kao člana superklase i kao člana potklase (tzv. *horizontalni princip*).

Pošto potklasa može da se posmatra kao poseban entitetski tip, specijalizacija se može primijeniti i na potklasu. Na primjer, u okviru entitetskog skupa *nastavnici* možemo identifikovati potklase (podgrupe) *nastavnik_ro* (nastavnik u punom radnom odnosu) i *hon_nastavnik* (honorarni nastavnik, po ugovoru). Specijalizacija se može primijeniti na isti entitetski skup po više kriterijuma. Tako se na univerzitetu *osobe* mogu grupisati po kriterijumu pozicije i uloge u nastavnom procesu na potklase *nastavnik*, *asistent* i *student*, a po kriterijumu radnog odnosa u potklase *zaposleni* i *honorarni*. Kada imamo specijalizaciju superklase po više kriterijuma, onda neki entitet iz superklase može pripadati i biti član više potklasa koje su identifikovane i kreirane po različitim kriterijumima. Za ilustraciju, neka osoba može istovremeno biti član i potklase *nastavnik* i potklase *zaposleni*.

Ukoliko neki entitetski skup kao potklasa može učestvovati u samo jednoj specijalizaciji, onda se specijalizacijom kreira hijerarhijska struktura entitetskih skupova. U suprotnom, ukoliko neki entitetski skup kao potklasa učestvuje u više specijalizacija[27], onda taj entitetski skup nasljeđuje više

[27] Ovakav entitetski skup/tip naziva se i *dijeljena potklasa* (eng. *shared subclass*).

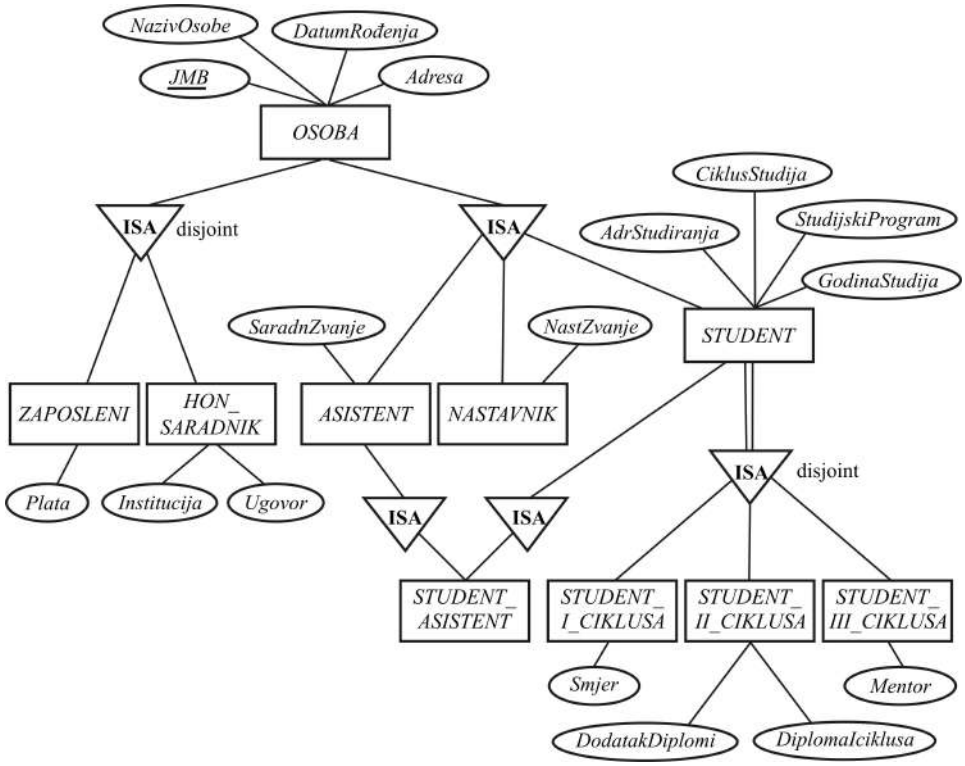
superklasa (eng. *multiple inheritance*), a rezultujuća struktura je mreža (eng. *lattice*). Na primjer, jedna osoba može biti i asistent i student, tako da entitetski skup *student_asistent* nasljeđuje atribute i veze i entitetskog skupa *asistent* i entitetskog skupa *student*, a time i atribute odgovarajućih entitetskih skupova viših nivoa (u ovom slučaju *osoba*). Entitet koji je član dijeljene potklase, mora biti istovremeno i član svake superklase koju dijeljena klasa nasljeđuje. Ako se neki atribut ili tip veze nasljeđuje višestruko u mrežnoj strukturi (na primjer, atributi entitetskog skupa *osoba* se u *student_asistent* nasljeđuju i preko *student* i preko *asistent*), takvi atributi i veze se uključuju samo jedanput u dijeljenoj potklasi (*student_asistent*), kako dijeljene potklase ne bi imale više identičnih atributa (to što je neko student-asistent ne znači da treba da ima dva imena ili dva JMB).

Kao što za MOV model nije standardizovana terminologija, nije standardizovana ni grafička notacija. U daljem tekstu za reprezentaciju specijalizacije u proširenim MOV dijagramima koristićemo jednu od čestih notacija, grafički simbol *trougao* koji se označava sa **ISA**[28] (sl. 2.13). Oznaka ISA potiče od skraćenice za '*is a*' (ili '*is an*'), sa značenjem da je član potklase ujedno i član superklase, kao na primjer, zaposleni 'je' (eng. '*is a*') osoba, ili nastavnik 'je' osoba.

Do identifikacije generalnijih entitetskih tipova i entitetskih tipova nižeg nivoa koji imaju određene specifičnosti (superklase i potklase) može se doći i **generalizacijom**, procesom obrnutim u odnosu na specijalizaciju. Ukoliko se u toku analize sistema prvo identifikuju neki entitetski tipovi (skupovi) a zatim utvrdi njihova sličnost, onda se može izvršiti generalizacija i definisanje entitetskog tipa (skupa) višeg nivoa (superklase), čiji bi atributi bili zajednički atributi polaznih entitetskih tipova.

Na primjer, ako bismo prvo identifikovali entitetske skupove *nastavnik*, *asistent* i *student*, mogli bismo uočiti zajedničke karakteristike entiteta tih entitetskih skupova, reprezentovane atributima *JMB*, *NazivNastavnika/Asistenta/Studenta*, *Adresa* i *DatumRođenja*. Na osnovu postojanja zajedničkih karakteristika, može se izvršiti sinteza ovih entitetskih skupova u entitetski skup višeg nivoa, superklasu *osoba*. Dakle, postupak generalizacije se primjenjuje 'odozdo prema gore' (eng. *bottom-up*), od postojećih, sličnih ali različitih entitetskih tipova, prema izvedenom generalnijem entitetskom tipu.

[28] Za ISA tip veze koristi se i termin veza tipa superklasa-potklasa (eng. *superclass-subclass relationship*). ISA grafički simbol – trougao koristi se i u varijanti sa vrhom trougla okrenutim nagore (obično ako se želi naglasiti da je kreiranje ISA veza rezultat procesa generalizacije).




**Slika 2.13**: Specijalizacija i generalizacija

Postupci specijalizacije i generalizacije su međusobno inverzni. Specijalizacija polazi od jednog entitetskog skupa u kojem se uočavaju i specifikuju razlike u svojstvima pojedinih podgrupa entiteta tako što se formiraju specifični entitetski skupovi nižeg nivoa. Generalizacija polazi od više sličnih entitetskih skupova čija se zajednička svojstva mogu reprezentovati kreiranjem entitetskog skupa višeg nivoa. Konceptualno, specijalizacija je preciziranje i dotjerivanje opisa, dok je generalizacija sinteza. Rezultat i generalizacije i specijalizacije je isti: slični entitetski tipovi se reprezentuju entitetskim tipom višeg nivoa (superklasom) i entitetskim tipovima nižeg nivoa (potklasama) koji su ISA vezom povezani sa superklasom.

Koncepti specijalizacije/generalizacije su potrebni i korisni iz slijedećih razloga:

- Ističu se specifičnosti pojedinih entiteta i entitetskih skupova, ali i njihova zajednička svojstva.
- Neke potklase mogu imati specifične veze sa drugim entitetskim tipovima, u odnosu na druge slične potklase. Specijalizacija/generalizacija

omogućava prezentaciju tih veza. Korištenjem ovog koncepta projektanti baza podataka mogu preciznije da specifikuju koji entiteti učestvuju u pojedinim vezama i kakva je priroda veza. Na primjer, samo nastavnici zaposleni u punom radnom odnosu mogu biti povezani sa fakultetom tipom veze *RUKOVODI*, tj. biti dekani. Veze entitetskih tipova nižeg nivoa (potklasa) sa drugim entitetskim tipovima na MOV dijagramima se predstavljaju na standardan način.

- Grafička predstava je jednostavnija i preglednija, jer se zajednički atributi specifikuju samo na jednom mjestu (kao atributi superklase).

**Ograničenja vezana za specijalizaciju/generalizaciju.** Kako bi se svojstva odnosa između entiteta realnog svijeta koja se opisuju kroz koncepte specijalizacije/generalizacije preciznije iskazala, definisana su različita ograničenja.

Prvi tip ograničenja se odnosi na način određivanja članstva entiteta iz superklase u jednoj ili više potklasa. Članstvo u potklasi može biti:

(1) **Definisano uslovom** (eng. *condition-defined*). Članstvo nekog entiteta iz superklase u nekoj potklasi zavisi od toga da li entitet zadovoljava određeni uslov ili predikat. Ako se članstvo entiteta iz superklase u bilo kojoj potklasi može utvrditi na bazi vrijednosti određenog atributa, onda se za takvu specijalizaciju/generalizaciju kaže da je **definisana atributom** (eng. *attribute-defined*).

(2) **Korisnički definisano** (eng. *user-defined*). Ako članstvo entiteta u potklasama nije definisano uslovom, onda se kaže da je članstvo korisnički definisano. Drugim riječima, podrazumijeva se da ovlašteni korisnik za svaki entitet posebno određuje njegovo članstvo u potklasama.

Drugi tip ograničenja se odnosi na specifikaciju da li entitet neke superklase može da bude član više potklasa. Ograničenje može biti:

(1) **Disjunktno** (eng. *disjoint*[29]). Ograničenje disjunktnosti specifikuje da neki entitet ne može biti član više od jedne potklase. Ovaj tip ograničenja se na MOV dijagramima specifikuje tekstom 'disjoint' pored ISA trougla koji označava specijalizaciju/generalizaciju.

(2) **Preklapajuće** (eng. *overlapping*[30]). Neki entitet može biti član više od jedne potklase. Podrazumijeva se preklapanje potklasa ukoliko ograničenje za disjunktnost entitetskih skupova nije eksplicitno specifikovano.

[29] Koristi se i termin eng. *exclusive*.

[30] Koristi se i termin eng. *inclusive*.

Treći tip ograničenja specifikuje kompletnost specijalizacije/generalizacije. Ograničenja ovog tipa mogu biti:

(1) **Totalna specijalizacija/generalizacija** (eng. *total specialization/generalization*). Specifikacija ovog ograničenja zahtijeva da svaki entitet superklase bude član najmanje jedne potklase. Ovaj tip ograničenja se na MOV dijagramima označava duplom linijom koja povezuje superklasu sa ISA trouglom kojim se reprezentuje specijalizacija/generalizacija.

(2) **Parcijalna specijalizacija/generalizacija** (eng. *partial specialization/generalization*). Neki entiteti superklase ne moraju biti članovi nijedne potklase (podrazumijevajuća opcija).

Na sl. 2.13 je predstavljena pojednostavljena specijalizacija osoba u informacionom sistemu fakulteta/univerziteta. Specijalizacija entitetskog tipa *STUDENT* po kriterijumu ciklusa studija je totalna, jer svaki student mora biti ili *STUDENT_I_CIKLUSA*, ili *STUDENT_II_CIKLUSA* ili *STUDENT_III_CIKLUSA* studija. Dodatno, takva specijalizacija je disjunktna, što znači da smo pretpostavili da neki student ne može da bude istovremeno student na dva ciklusa studija. Primijetimo da specijalizacija superklase *OSOBA* nije totalna, što znači da u bazi možemo imati registrovane osobe koje ne pripadaju nijednoj od potklasa (na primjer, penzionisane osobe i druge osobe koje više nisu na univerzitetu). Takođe, specijalizacija superklase *OSOBA* nije disjunktna (nego podrazumijevajuće preklapajuća), jer npr. osoba koja je student, istovremeno može biti i zaposlena. Ako želimo da registrujemo osobe koje su istovremeno studenti i asistenti, onda je potrebno kreirati potklasu *STUDENT_ASISTENT*, koja nasljeđuje atribute i veze i superklase *STUDENT* i superklase *ASISTENT*. Član dijeljene potklase *STUDENT_ASISTENT* je istovremeno član i superklase *STUDENT* i superklase *ASISTENT*, i slijedom puta nasljeđivanja takođe član superklase *OSOBA*.

### Potklase tipa unije ili kategorije

Za razliku od prethodnih primjera, u kojima je u svakoj vezi tipa superklasa/potklasa uključena samo jedna superklasa, u određenim situacijama postoji potreba da se modeluje veza koja uključuje potklasu sa više superklasa različitih tipova. U tom slučaju potklasa se naziva **unija** ili **kategorija**[31], i reprezentuje skup entiteta različitih tipova. Na primjer, ako

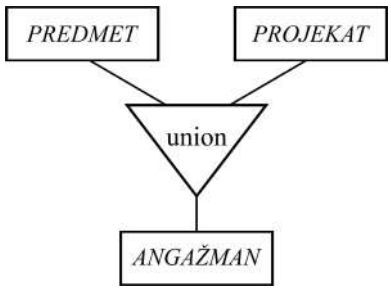
[31] Takođe se koriste engleski termini *union type*, *cluster type*, ili *cluster class*. Ovaj koncept je predložen 1985 (Elmasri, Weddreyer, Hevner), ali nije podržan u većini MOV dijagrama i alata. Za ilustraciju koncepta korištena je ISA notacija.

nastavnici i saradnici na univerzitetu mogu biti angažovani na predmetima i/ili na projektima, onda te različite vrste poslova možemo modelovati potklasom *ANGAŽMAN* (sl. 2.14). Entiteti potklase (kategorije) *ANGAŽMAN* jesu entiteti tipa *PREDMET* i entiteti tipa *PROJEKAT*, dakle kategorija je podskup unije entiteta različitih tipova, članova superklasa. Drugi primjer: ako bismo željeli da modelujemo tip *VLASNIK* koji bi reprezentovao vlasnike vozila koji mogu biti *OSOBA* ili *KOMPANIJA*, onda bi se *VLASNIK* mogao modelovati kao kategorija/potklasa povezana unijom sa supeklasama *OSOBA* i *KOMPANIJA*.

Neki entitet kategorije ne može biti član više od jedne superklase. Svaki entitet iz entitetskog skupa *angažman* je član ili entitetskog skupa *predmet* ili entitetskog skupa *projekat* (i ne može biti član i jednog i drugog entitetskog skupa). Svaki entitet entitetskog skupa *angažman* nasljeđuje ili atribute entitetskog skupa *predmet* ili atribute entitetskog skupa *projekat*, zavisno od superklase kojoj pripada. Superklase kategorije mogu imati neke zajedničke atribute, pa i atribute koji čine ključ.

Kategorija može biti totalna ili parcijalna. Totalna kategorija sadrži uniju svih entiteta iz superklasa, dok parcijalna kategorija sadrži podskup unije entiteta iz superklasa. Ako svaki predmet i svaki projekat predstavlja angažman nekog nastavnika ili saradnika, onda *ANGAŽMAN* predstavlja totalnu kategoriju. S druge strane, u primjeru za kategoriju *VLASNIK*, neke osobe (ili kompanije) ne moraju biti i vlasnici vozila, pa je kategorija *VLASNIK* parcijalna, jer ne uključuje sve entitete iz superklasa *OSOBA* i *KOMPANIJA*.

Ukoliko je kategorija totalna, onda se ona može modelovati i kao generalizacija. U tom slučaju, ukoliko entiteti superklasa imaju značajan broj zajedničkih atributa i ako se mogu okarakterisati kao entiteti suštinski istog tipa, onda je pogodnije koristiti generalizaciju; u suprotnom, kategorija je adekvatniji izbor.



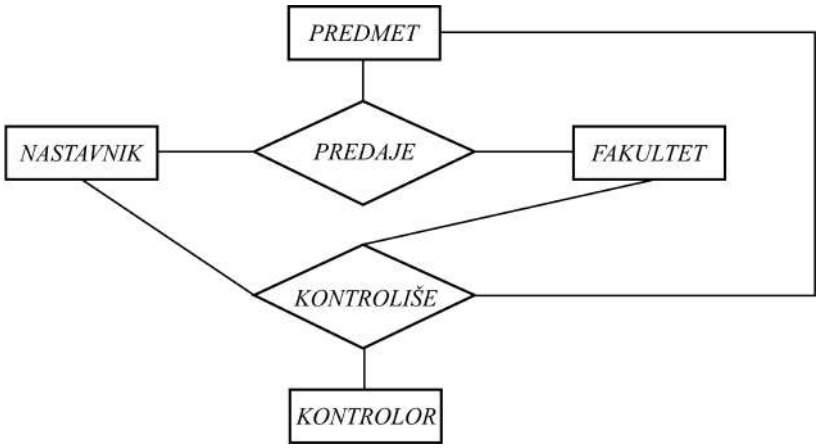

**Slika 2.14**: Unija

Modelovanje unija nije standardno uključeno u svim alatima i metodologijama modelovanja.

### Agregacija

Generalno, agregacija (eng. *aggregation*) je koncept kojim se kreira složeni (kompozitni) objekat od komponentnih objekata. Ilustrujmo primjenu koncepta agregacije u MOV modelu na slijedećem primjeru:

Pretpostavimo da rukovodstvo univerziteta želi da prati i kontroliše neka predavanja (koja su reprezentovana vezom *PREDAJE* na sl. 2.10) na univerzitetu. Pretpostavimo da entitetski tip *KONTROLOR* reprezentuje osobe koje bi bile zadužene za kontrolu. Takvu situaciju možemo modelovati MOV modelom na sl. 2.15.




**Slika 2.15**: MOV dijagram sa redundantnim vezama

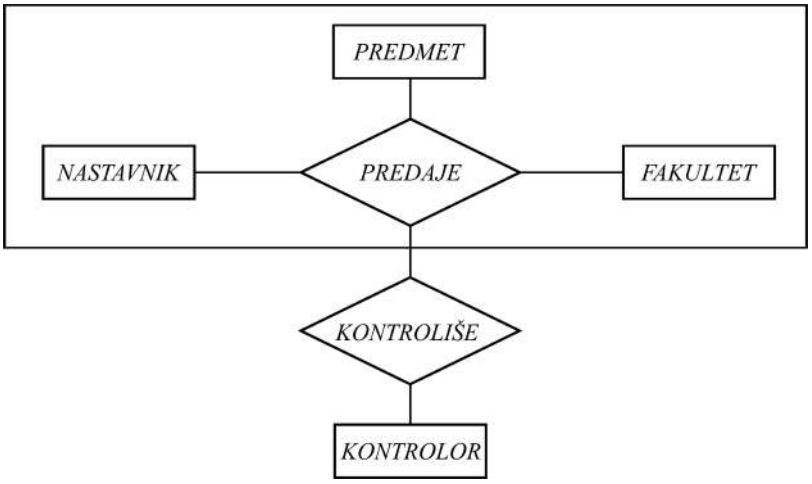
Tip veze *KONTROLIŠE* definiše skup veza $(k_i, n_i, p_i, f_i)$, gdje je $k_i$ kontrolor zadužen za kontrolu predavanja koje izvodi nastavnik $n_i$ na predmetu $p_i$ na fakultetu $f_i$. Primijetimo da, iako se u vezi $(k_i, n_i, p_i, f_i)$ pojavljuje informacija o vezi $(n_i, p_i, f_i)$, koja je sadržana u veznom skupu tipa *PREDAJE*, ne možemo objediniti ove tipove veze/vezne skupove u jedan, pošto u veznom skupu tipa *PREDAJE* može postojati veza $(n_j, p_j, f_j)$ koja nije sadržana u veznom skupu tipa *KONTROLIŠE*. Ipak, sam model sadrži redundantne informacije o vezama nastavnika, predmeta i fakulteta (i preko veznog tipa *KONTROLIŠE* i preko veznog tipa *PREDAJE*). Bilo bi pogodno da se prezentuje veza kontrolora i trojke (nastavnik, predmet, fakultet), ali zbog konvencije MOV modela nije dozvoljeno

direktno povezivati tipove veza (u prethodnom slučaju *KONTROLIŠE* i *PREDAJE*). Koncept kojim se može na pogodan način opisati ova situacija jeste **agregacija**.

Agregacija je apstrakcija kojom se veze tretiraju kao entiteti višeg nivoa. Za prethodni primjer, vezni tip *PREDAJE* se posmatra kao entitetski tip višeg nivoa, tako da je moguće kreirati binarni tip veze (*KONTROLIŠE*) između entitetskog tipa *KONTROLOR* i višeg entitetskog tipa *PREDAJE*. Na sl. 2.16 je predstavljena notacija za specifikaciju tipa veze kao entitetskog tipa višeg nivoa.

Prethodno prezentovani koncept agregacije koji se eksplicitno koristi u MOV modelu, razlikuje se od koncepta agregacije kojim se objekti kombinuju u agregatni (kompozitni) objekat višeg nivoa (koncept agregacije koji se koristi za modelovanje u drugim modelima, UML-u, na primjer)[32]. Po takvom konceptu agregacije, komponentni objekti su povezani sa agregatnim objektom tipom veze IS_PART_OF (JE_DIO_OD).

Dodatno, UML omogućava i modelovanje **kompozicije** (eng. *composition*), koja predstavlja snažniji oblik veze od agregacije. Naime, agregacija omogućava modelovanje veze tipa dio-cjelina u kojoj dijelovi mogu da egzistiraju bez cjeline, dok u kompoziciji dijelovi ne mogu da egzistiraju bez cjeline.

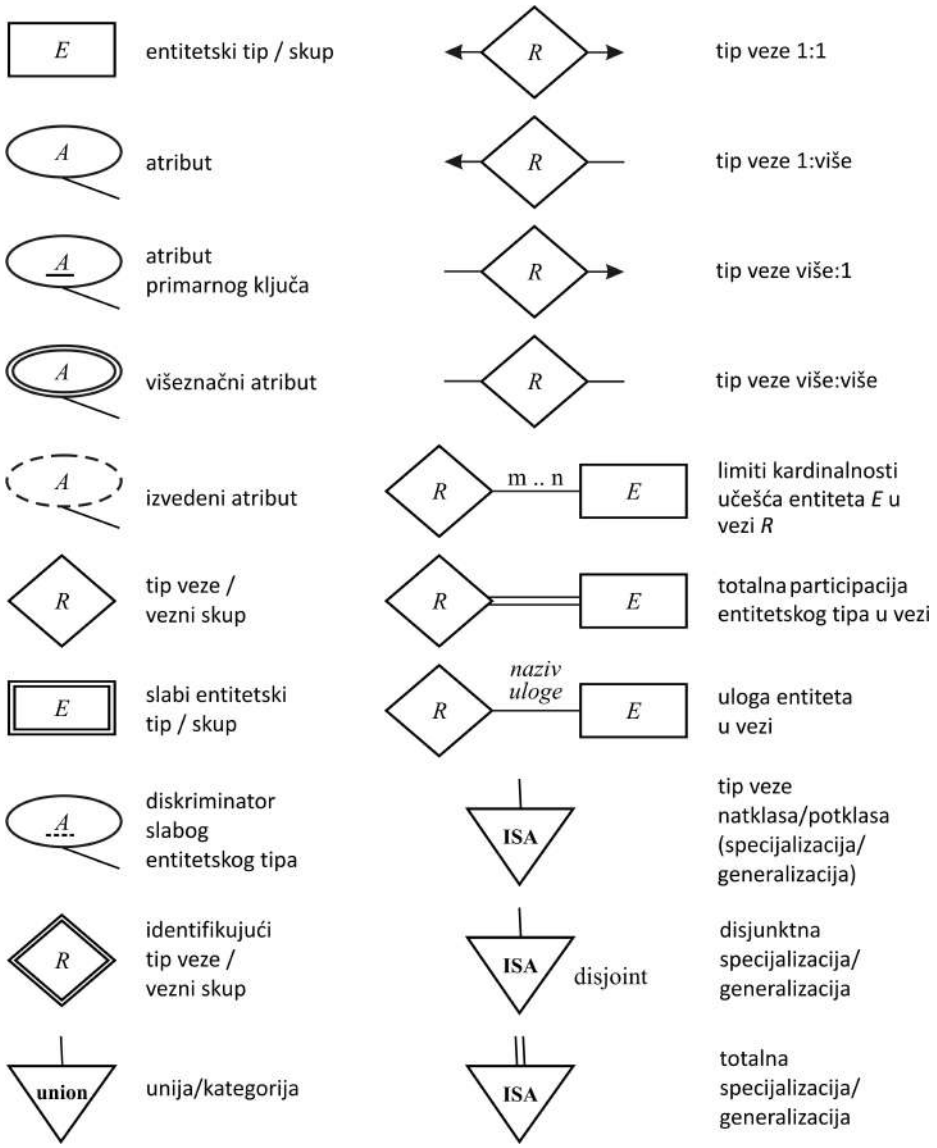



**Slika 2.16**: MOV dijagram sa agregacijom

[32] Ovim ističemo distinkciju između ovih koncepata agregacije.

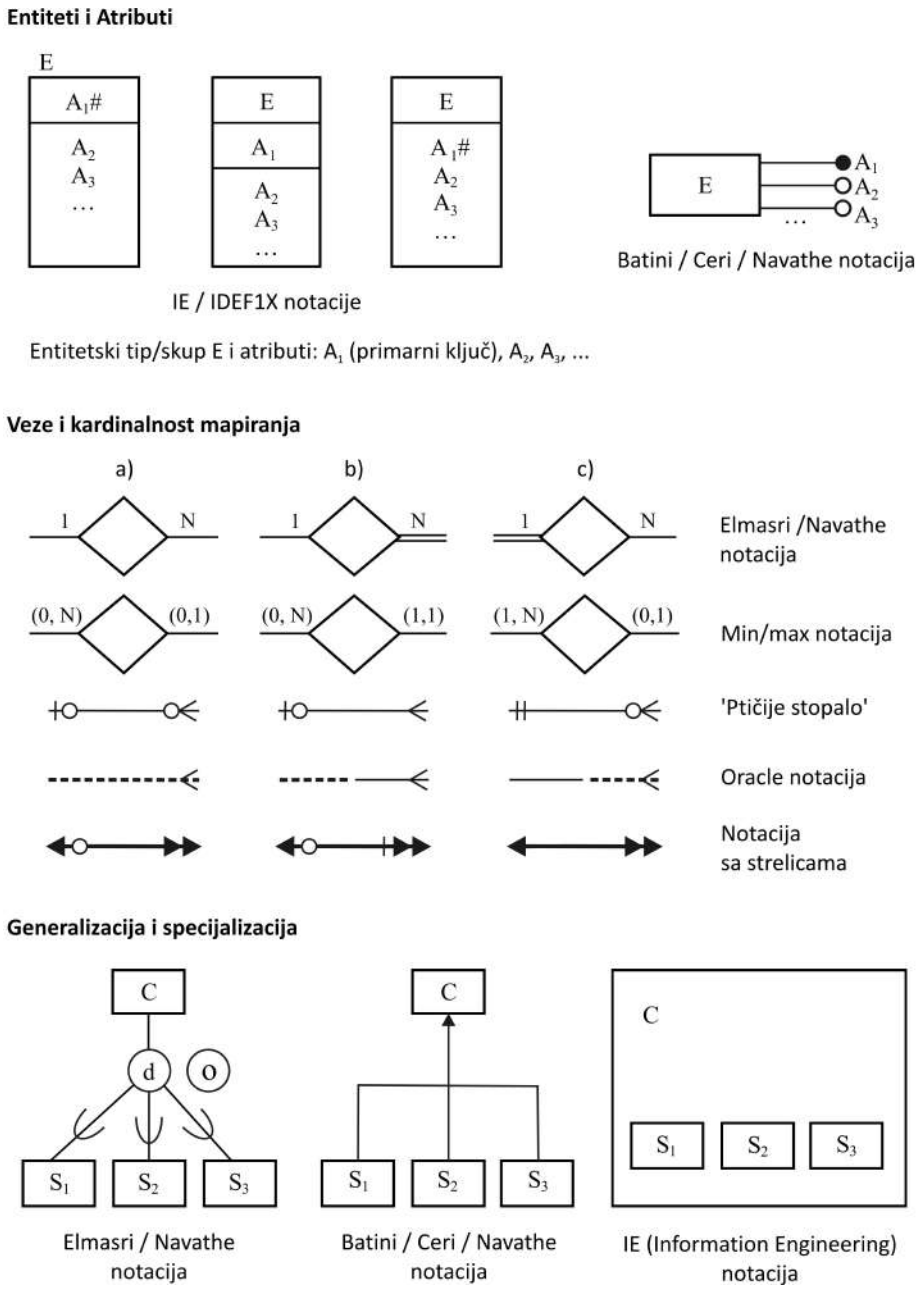
### 2.1.7 Sumarni pregled simbola koji se koriste u MOV modelu i alternativne notacije

Skup simbola koji se koriste za MOV modelovanje nije standardizovan. U ovoj knjizi korištena je Korth & Silberschatz [45] notacija (1991), bliska originalnoj Chenovoj notaciji za MOV dijagrame (sl. 2.17).




**Slika 2.17**: Sumarni pregled simbola MOV modela

Neki simboli koji se alternativno [47] koriste u različitim alatima i metodologijama prikazani su na sl. 2.18.




**Slika 2.18**: Alternativne notacije u MOV dijagramima

Kao što se vidi sa sl. 2.18, za predstavljanje entitetskog tipa i atributa alternativno se koristi pravougaonik, u okviru kojeg se navode atributi (primarni ključ je odijeljen posebnim separatorom u okviru pravougaonika), dok se naziv entitetskog tipa specifikuje iznad pravougaonog okvira (IDEF1X notacija). Varijante ove reprezentacije su sa specifikacijom naziva entitetskog tipa unutar okvira, sa i bez separacije atributa unutar okvira. U nekim alatima entitetski tipovi se predstavljaju zaobljenim ivicama okvira (Oracle). Atributi koji čine primarni ključ se mogu istaći dodatkom posebnog znaka (#, na primjer), dok se u nekim alatima posebnim znakom označavaju atributi koji su obavezni (eng. *mandatory attributes*).

Za predstavu veza, kardinalnosti mapiranja i ograničenja participacije entiteta u vezi, koriste se takođe različite alternativne notacije, od kojih su neke prikazane na sl. 2.18. Pored predstave veze simbolom romba, koristi se i linija. Prikazana je veza sa kardinalnošću mapiranja 1:više, gdje jedan entitet entitetskog skupa sa lijeve strane veze može biti u vezi sa više entiteta entitetskog skupa sa desne strane veze, dok neki entitet sa desne strane veze može biti u vezi samo sa jednim entitetom entitetskog skupa sa lijeve strane veze. Ograničenje totalnosti participacije entitetskih skupova u vezi je parcijalno sa obe strane za slučaj a), entitetski skup sa strane više veze totalno participira u veznom skupu za slučaj b), dok je participacija entitetskog skupa sa lijeve strane veze totalna za slučaj c).

Kardinalnost učešća entiteta nekog entitetskog skupa u vezi može biti specifikovana odgovarajućom notacijom kardinalnosti na strani do entitetskog skupa (eng. *look here notation*), kao što je to slučaj kod Min/max notacije, ili specifikacijom kardinalnosti na suprotnoj strani veze (eng. *look accross notation*), što je korišteno u ostalim notacijama na sl. 2.18. Sve navedene notacije za participaciju entiteta entitetskog skupa u vezi koriste *look here* notaciju, odnosno specifikuju participaciju entitetskog skupa na strani veze do entitetskog skupa.

Na kraju, na sl. 2.18 date su i neke od alternativno korištenih notacija za predstavu generalizacije/specijalizacije.

Treba napomenuti da neke notacije ne omogućavaju modelovanje veza čiji je stepen veći od dva, tj. podržavaju modelovanje samo binarnih veza.

### 2.1.8 Kreiranje MOV dijagrama i projektne opcije

Iako je u uvodnom dijelu poglavlja 2.1 MOV koncept opisan kao jednostavan i intuitivan, ne postoji jedinstvena metodologija koja propisuje precizne i rigidne korake koji vode ka jedinstvenom i jednoznačnom rješenju. Razlog tome je što ne postoje precizna pravila ni za jednoznačnu identi-

fikaciju i definisanje osnovnih komponenata modela (entitetskih i veznih tipova/skupova) realnog sistema. Najopštiji pristup je da se polazna identifikacija entitetskih skupova i veza vrši na osnovu specifikovanih zahtjeva: imenice koje se pojavljuju u specifikaciji zahtjeva i koje reprezentuju objekte, pojave, događaje realnog sistema jesu kandidati za nazive entitetskih tipova, dok su glagoli koji opisuju interakcije između objekata potencijalno nazivi tipova veza početnog modela. U toku ovog procesa identifikacije entitetskih skupova, veza i ograničenja projektanti se susreću sa različitim mogućnostima i dilemama. Selektovane opcije o ulozi pojedinih objekata i njihovim interakcijama treba da reflektuju realne situacije i funkcionalne zahtjeve. Različite percepcije realnog (mini) svijeta od interesa, rezultovaće različitim zaključcima i identifikacijom entitetskih i veznih skupova na različite načine i na kraju različitim modelima.

Sam proces izgradnje modela je najčešće iterativan, pri čemu se prvo kreira početni model, a zatim se u iterativnom procesu u svakoj iteraciji vrši analiza, otklanjanje nedostataka i poboljšanje tekućeg modela, sve dok se ne dođe do finalnog modela koji na adekvatan način opisuje realni sistem.

Neke od najčešćih dilema u toku procesa izgradnje MOV modela kao i orijentacione smjernice za izbor odgovarajuće opcije date su u tekstu koji slijedi.

### Korištenje atributa ili entitetskog tipa

Za ilustraciju ovih opcija modelovanja, posmatrajmo problem predstavljanja telefona zaposlenih u MOV modelu nekog poslovnog sistema. S jedne strane, od interesa može da bude registrovanje samo brojeva telefona zaposlenih osoba, kako bismo evidentirali telefonske kontakt informacije tih osoba. U tom slučaju telefon bi mogli posmatrati kao višeznačni atribut, koji reprezentuje jedno svojstvo (telefonske kontakt informacije) zaposlenih osoba. S druge strane, od interesa bi mogle biti i druge informacije vezane za telefon, na primjer tip telefona (mobilni/fiksni), prostorija u kojoj se telefon nalazi (ako je telefon fiksni), godina proizvodnje, marka, itd. U tom slučaju, telefon je adekvatno posmatrati kao entitetski tip koji karakterišu određena svojstva (atributi), i koji je povezan sa drugim entitetskim tipovima (zaposlenih, dobavljača, proizvođača, ...) odgovarajućim tipovima veze. Na primjer, veza telefona kao entiteta sa zaposlenim može reprezentovati telefonski broj/kontakt informaciju za komunikaciju sa osobom, zaduženje osobe za telefon itd.

Postojanje objekata, pojava, događaja kao posebnosti u realnom svijetu je potreban uslov za modelovanje tih *posebnosti* (entiteta) konceptom

entitetskog tipa. Na primjer, imena ili količine se ne mogu same za sebe smatrati posebnostima (objekti, pojave ili događaji realnog svijeta), pa se iz tog razloga ne modeluju konceptom entitetskog tipa. Navedeni pojmovi predstavljaju određene karakteristike pojedinih entitetskih tipova/skupova, pa se oni iz tog razloga modeluju atributima.

Ukoliko pak identifikujemo entitete/posebnosti realnog svijeta, u MOV modelu ih najčešće modelujemo entitetskim tipovima. U određenim situacijama možemo te posebnosti modelovati i kao atribute drugog entitetskog tipa (kao u prvoj varijanti modelovanja telefona). Ukoliko je od interesa da evidentiramo samo jednu (ili mali broj) karakteristiku neke posebnosti (entiteta) kao svojstvo entiteta nekog drugog tipa, onda se takva situacija može modelovati atributom, umjesto entitetskim tipom. Kao drugi primjer posmatrajmo studijske programe kao posebnosti/entitete univerziteta. Ukoliko bi za realni sistem bilo dovoljno da se naziv studijskog programa evidentira (samo) kao karakteristika entitetskog tipa *STUDENT*, onda bi *NazivSP* mogao da bude atribut entitetskog tipa *STUDENT*. Međutim, ukoliko bi bilo potrebno da se u realnom sistemu vode i druge informacije o studijskom programu (*OpisSP*, *BrojRješenjaSP*, *Ciklus*, *Trajanje*, ... , što je opcija koja je više realistična) i/ili ako bi neki studijski program bio u vezi i sa drugim entitetima različitih tipova (predmeti, fakultet, nastavnici, ...), onda je bolje reprezentovati studijske programe konceptom entitetskog tipa i ustanoviti odgovarajuće tipove veza sa drugim entitetskim tipovima.

Rješenje sa entitetskim tipom je generalnije, jer omogućava modelovanje svih potrebnih karakteristika određenog tipa objekata, pojava, događaja realnog svijeta samo na jednom mjestu i daje jasnu predstavu tih entiteta i njihovih interakcija sa drugim entitetima. Kao što je prethodno navedeno, u određenim situacijama, kada je od interesa evidentiranje jedne ili malog broja karakteristika neke posebnosti kao svojstva drugih entiteta (samo jednog tipa), pogodnija je predstava te posebnosti kao atributa (drugog) entitetskog tipa.

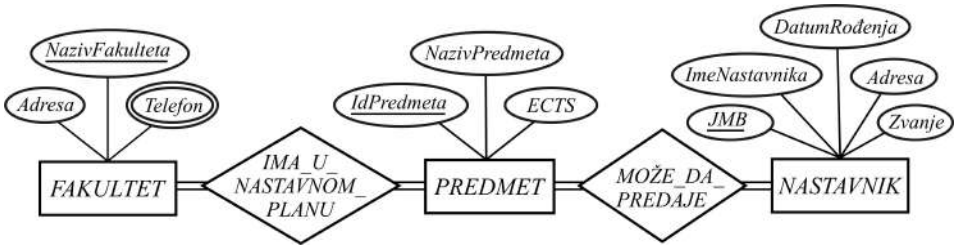
Odluka da li će se neke činjenice modelovati kao atribut ili kao entitet, često zavisi od konteksta. Na primjer, *Ulica* kao dio adresne informacije se može modelovati kao atribut ili kao entitet, zavisno od aplikativnog konteksta. Za specifikaciju adresne informacije poslovnih partnera ili zaposlenih najčešće je dovoljno modelovati adresne informacije o ulici i kućnom broju kao prosti atribut. Međutim, u aplikacijama koje će koristiti adresne informacije za optimizaciju transporta, ili u aplikacijama koje ih koriste za izborne i popisne liste i procese, prethodne (i dodatne) adresne informacije je potrebno voditi kao posebne atribute entitetskog tipa *ULICA*, sa

ciljem lakše ekstrakcije potrebnih adresnih detalja, grupisanja, selekcije i sortiranja podataka. Na primjer, u aplikacijama koje se odnose na izborne procese bitna je pripadnost adresa izbornoj jedinici, pa bi iz tog razloga bilo potrebno adresne informacije modelovati korištenjem entitetskog tipa.

U procesu modelovanja moguće je da se neka posebnost prvo modeluje kao atribut (npr. *NazivSP*) i da se naknadno zaključi da ta posebnost egzistira kao atribut u više različitih entitetskih tipova (na primjer, *NazivSP* može da se modeluje kao atribut u entitetskim tipovima *STUDENT*, *PREDMET*, *FAKULTET*, ...), te da se u nekoj od narednih iteracija modelovanja napravi modifikacija modela tako da se studijski program modeluje entitetskim tipom *STUD_PROGRAM*, i taj novi entitetski tip poveže odgovarajućim tipovima veza sa entitetskim tipovima *STUDENT*, *PREDMET*, *FAKULTET*. U ovom slučaju varijanta sa modelovanjem studijskog programa kao entitetskog tipa je bolja, jer, zbog njihove egzistencije samo na jednom mjestu, omogućava jednostavne izmjene podataka. Moguća je i obrnuta varijanta: ako bi se inicijalno modelovao entitetski tip *STUD_PROGRAM* sa npr. samo jednim atributom (*NazivSP*), i ako bi taj entitetski tip bio povezan samo sa entitetskim tipom *STUDENT* odgovarajućim tipom veze, onda bi se u nekoj od slijedećih iteracija prethodna situacija (da je student povezan sa studijskim programom na koji je upisan) mogla modelovati atributom *NazivSP* u entitetskom tipu *STUDENT* (pri čemu bi se iz modela eliminisao entitetski tip *STUD_PROGRAM* i odgovarajuća veza tog tipa sa entitetskim tipom *STUDENT*).

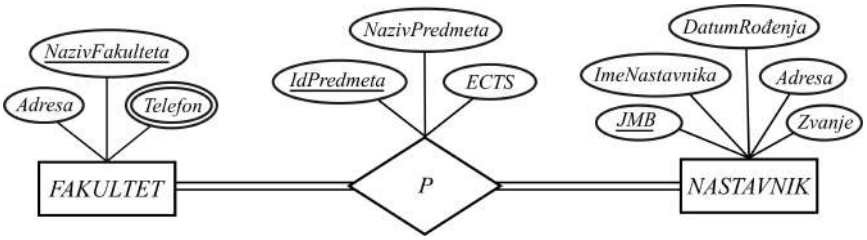
### Korištenje entitetskog tipa ili tipa veze

Pretpostavimo da smo analizirali realni univerzitetski sistem i utvrdili da postoje entitetski tipovi *FAKULTET*, *PREDMET* i *NASTAVNIK* i da želimo da MOV modelom reprezentujemo činjenice o mogućem izvođenju nastave iz predmeta na fakultetima i o nastavnicima koji mogu da izvode tu nastavu. Zavisno od konkretnog univerzitetskog sistema, mogli bismo utvrditi da jedan nastavnik može da drži nastavu samo na jednom ili na više predmeta, te da nastavu iz jednog predmeta može da izvodi jedan ili više nastavnika. Nastava iz nekog predmeta može da se izvodi na više (studijskih programa) različitih fakulteta, ili samo na jednom fakultetu. Svaki fakultet bi morao da organizuje nastavu iz više (ne obavezno svih) predmeta. Najopštiji slučaj (nastava iz nekog predmeta može da se izvodi na više (studijskih programa) različitih fakulteta, jedan nastavnik može da izvodi nastavu na više predmeta, nastavu na jednom predmetu može da izvodi više nastavnika) predstavljen je na sl. 2.19.




**Slika 2.19**: MOV model sa entitetskim tipovima *FAKULTET*, *PREDMET* i *NASTAVNIK*

Razmotrimo mogućnost da se navedene činjenice reprezentuju modelom u kojem se *PREDMET* modeluje tipom veze koji povezuje entitetske tipove *FAKULTET* i *NASTAVNIK*, kako je to predstavljeno na sl. 2.20.




**Slika 2.20**: MOV model u kojem se predmeti reprezentuju preko tipa veze

Analizirajmo prethodni segment modela podataka univerziteta za različite varijante ograničenja participacije entitetskih tipova u vezi. Ukoliko se neki predmet predaje samo na jednom fakultetu, i ukoliko nastavnik predaje samo jedan predmet na fakultetu (što nije realna opcija), onda predloženo rješenje može biti zadovoljavajuće: podaci o predmetu registrovaće se kao opisni atributi samo jedne veze. Ako želimo da reprezentujemo uobičajenu situaciju, gdje jedan nastavnik može da predaje više predmeta na jednom fakultetu, onda bi se podaci o predmetu morali modelovati kao višeznačni kompozitni atribut (neki nastavnik sa nekim fakultetom može biti povezan samo jednom vezom, iz čega slijedi da se za reprezentaciju predavanja više predmeta od strane jednog nastavnika na jednom fakultetu mora koristiti višeznačni kompozitni atribut). Ova situacija se mnogo jednostavnije reprezentuje i interpretira modelom na sl. 2.19, jer je specifikovana kardinalnost M na strani tipa veze *MOŽE_DA_PREDAJE* prema entitetskom tipu *PREDMET*.

Ako dalje pretpostavimo realnu i generalniju situaciju da se neki predmet može predavati od strane više nastavnika i na više različitih fakulteta,

onda to znači da bi opis istog predmeta (putem atributa veze $P$) trebao da postoji u okviru više veza veznog skupa $P$. Dakle, ako predmet $p_i$ predaje više nastavnika, i ako se taj predmet predaje na više fakulteta, onda će za svaki fakultet na kojem se $p_i$ predaje postojati veza sa nastavnikom (ili nastavnicima) koji drži nastavu na tom predmetu. Za svaku takvu vezu, atributi veze (kojima se opisuje predmet koji nastavnik drži na fakultetu) moraju imati istu vrijednost. Time bismo imali redundanciju podataka, tj. isti podaci (o predmetu) bi se pojavljivali kao opisni atributi u više veza. Pored zauzeća dodatnog prostora, problem sa redundancijom podataka je posebno aktuelan kod ažuriranja: svaka promjena vezana za predmet (npr. naziv predmeta, broj ECTS bodova) mora višestruko da se ažurira u svim redundatnim instancama ovih podataka.

U rješenju na sl. 2.19 ne postoji problem redundancije podataka (podaci postoje i registruju se samo na jednom mjestu), niti moramo da koristimo višeznačne atribute. Dalje, entitete tipa *PREDMET* možemo da dovedemo u vezu sa entitetima drugih entitetskih tipova, što nije moguće predstavom na sl. 2.20.

Kao uopšteno pravilo, koncept veza treba koristiti za modelovanje interakcije između entiteta. U prethodnom slučaju, predmeti su entiteti, pa ih je bolje modelovati entitetskim tipom nego tipom veze.

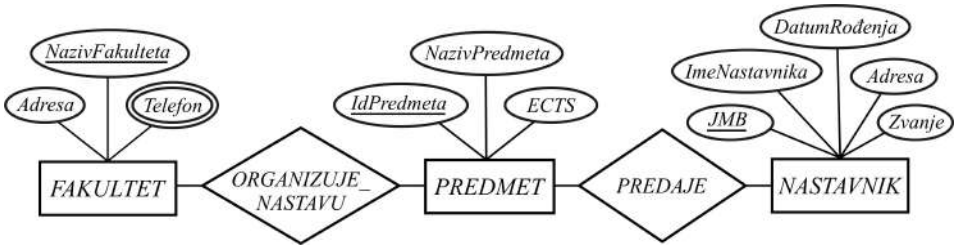
Primijetimo i slijedeću distinkciju dijagrama na sl. 2.19 i 2.20. Slika 2.20 reprezentuje činjenicu da nastavnik *na određenom fakultetu* može da izvodi nastavu iz određenog predmeta. Slika 2.19 reprezentuje činjenicu da nastavnik može da predaje određeni predmet, ali bez ograničenja na kojem fakultetu može da izvodi nastavu.

### Binarni ili *n-arni* tipovi veza

U procesu modelovanja, najčešće srećemo i koristimo binarni tip veze. U određenim situacijama, odnosi između entitetskih skupova/tipova mogu bolje da se reprezentuju *n-arnim* vezama. Dijagramom na sl. 2.10, ternarnom vezom modelovane su činjenice o izvođenju nastave na predmetima na fakultetima od strane profesora. Instanca veze $(n_i, p_i, f_i)$ specifikuje da nastavnik $n_i$ izvodi nastavu (predaje) na predmetu $p_i$ na fakultetu $f_i$. Model predstavljen na sl. 2.21, na kojem su isti entitetski tipovi povezani preko dva binarna tipa veze, ne reprezentuje istu stvar (Napomena: sličan model na sl. 2.19 reprezentuje mogućnosti držanja nastave na fakultetima iz pojedinih predmeta, i mogućnost profesora da izvodi nastavu iz predmeta, a ne činjenicu da se nastava izvodi). Učešće entitetskih tipova u vezama je

parcijalno. To znači da neki fakultet (koji je u osnivanju, npr.) može da egzistira a da još nije organizovao izvođenje nastave, da neki nastavnik u nekom trenutku može biti bez angažmana na izvođenju nastave (npr. zbog specijalizacije), ili da imamo predmet na kojem se ne izvodi nastava (npr. izborni predmet za koji nije bilo zainteresovanih studenata).

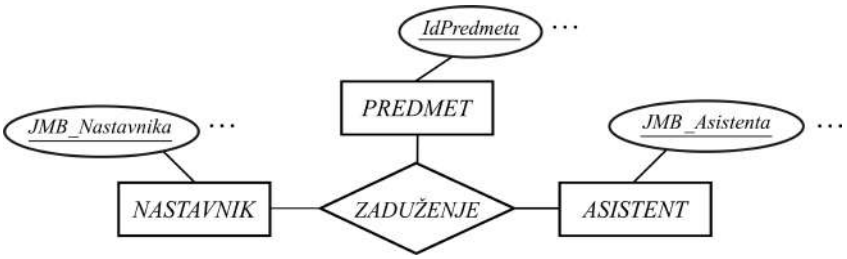
MOV dijagram sa sl. 2.21 reprezentuje činjenice da nastavnici izvode nastavu (predaju) na pojedinim predmetima, i takođe činjenicu da se nastava na nekom fakultetu izvodi (odvija) za određene predmete, ali se ne može zaključiti koji nastavnici izvode nastavu na kojim fakultetima. Na primjer, ako tri nastavnika izvode nastavu na nekom predmetu, i nastava iz tog predmeta se odvija na tri fakulteta, nije moguće zaključiti koji nastavnik drži nastavu na kojem fakultetu.




**Slika 2.21**: MOV model o izvođenju nastave sa dvije binarne veze

Za razliku od korištenja binarnih veza, ternarnom vezom se jasno i adekvatno modeluju činjenice o držanju predavanja iz nekog predmeta na nekom fakultetu od strane određenog nastavnika (sl. 2.10).

U nekim slučajevima, činjenice i situacije realnog svijeta bolje se modeluju binarnim nego ternarnim vezama. Za ilustraciju posmatrajmo primjer modelovanja zaduženja nastavnika i asistenata za izvođenje nastave na određenom predmetu, predstavljen na sl. 2.22.



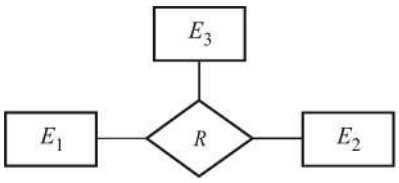

**Slika 2.22**: Zaduženje nastavnika i saradnika modelovano ternarnom vezom

Model na sl. 2.22 nije odgovarajući za moguće situacije kada na predmetu nema zaduženja asistenata (na primjer, ako nastavnim planom na predmetu nisu predviđene vježbe, ili ako nastavnik izvodi i predavanja i vježbe). Po definiciji, vrijednost atributa na primarnom ključu entitetskog tipa/tipa veze mora biti jedinstvena za svaki entitet u entitetskom skupu/vezu u veznom skupu. *Null* vrijednost nekog atributa primarnog ključa znači da je njegova vrijednost nepoznata ili da ne postoji, što znači da ne bismo mogli utvrditi jedinstvenost vrijednosti na primarnom ključu, što je u suprotnosti sa definicijom primarnog ključa. To implicira da vrijednosti atributa na primarnom ključu entitetskog tipa/tipa veze ne mogu biti *null*. S obzirom na to da je superključ (i primarni ključ) tipa veze *ZADUŽENJE* {*JMB_Nastavnika*, *JMB_Asistenta*, *IdPredmeta*}, i da vrijednost atributa na primarnom ključu ne može biti *null* vrijednost, navedenim modelom se ne bi moglo predstaviti zaduženje samo nastavnika na predmetu (vrijednost *JMB_Asistenta* na primarnom ključu za takvu vezu bi bila *null*).

Iz navedenog razloga, korektno bi bilo modelovati navedenu situaciju sa dvije binarne veze, kako je to predstavljeno MOV modelom na sl. 2.23.


**Slika 2.23**: Zaduženje nastavnika i saradnika modelovano binarnim vezama

Svaka *n-arna* veza ($n > 2$) se može zamijeniti modelom koji sadrži samo binarne veze. Na primjer, posmatrajmo ternarni tip veze $R$ koji povezuje entitetske tipove $E_1, E_2, E_3$ (sl. 2.24).



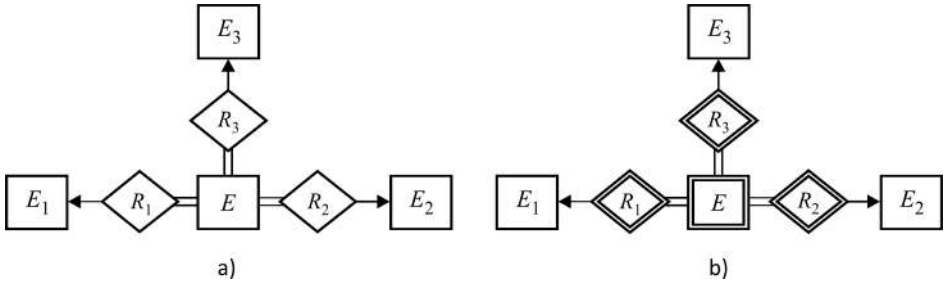

**Slika 2.24**: Ternarna veza entitetskih tipova $E_1, E_2, E_3$

Prethodni model se može predstaviti ekvivalentnim modelima sa binarnim vezama, predstavljenim na sl. 2.25.

U varijanti predstavljenoj na sl. 2.25a), entitetski tip $E$ je vještački entitetski tip koji kao atribute ima vještački kreiran primarni (surogat) ključ,

plus atribute tipa veze $R$, ukoliko postoje. Za svaku vezu $(e_1, e_2, e_3) \in R$, gdje je $e_1 \in E_1, e_2 \in E_2, e_3 \in E_3$, potrebno je formirati vještački entitet $e \in E$ i veze $(e, e1) \in R_1, (e, e2) \in R_2, (e, e3) \in R_3$.

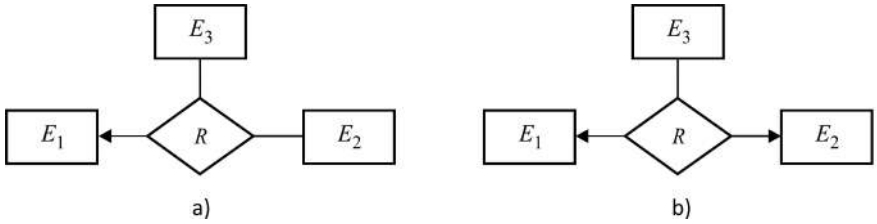
U varijanti predstavljenoj na sl. 2.25b), entitetski tip $E$ je slabi entitetski tip koji identifikuju jaki entitetski tipovi $E_1, E_2, E_3$. Primarni ključ slabog entitetskog tipa $E$ je kombinacija primarnih ključeva entitetskih tipova $E_1$, $E_2$, $E_3$. Ukoliko ternarni tip veze $R$ ima opisne atribute, oni postaju opisni atributi slabog entitetskog tipa $E$. Svaka veza $(e_1, e_2, e_3) \in R$ implicira postojanje entiteta u slabom entitetskom skupu, čiji se primarni ključ formira kao $PK(e_1) \cup PK(e_2) \cup PK(e_3)$.




**Slika 2.25**: Model sa binarnim vezama, ekvivalentan modelu sa ternarnom vezom

Često nije jednostavno donijeti zaključak da li je bolje neki odnos između entitetskih tipova predstaviti *n-arnom* vezom ($n > 2$) ili određenim brojem veza nižeg stepena. Zaključak treba da bude rezultat analize koja treba da dâ odgovor na pitanje da li određeni model adekvatno izražava odnose i ograničenja između entitetskih tipova i da li se datom predstavom mogu specifikovati svi mogući realni slučajevi. Odluka treba da uzme u obzir i ekspresivnost i jasnoću pojedinih opcija.

Semantika kardinalnosti mapiranja učešća entitetskih tipova u *n-arnom* tipu veze nije jednoznačna u svim slučajevima gdje imamo kardinalnost 1 na strani više od jednog entitetskog tipa (i barem jednu kardinalnost tipa M). Posmatrajmo kardinalnost mapiranja za ternarnu vezu $R$ koja povezuje entitetske tipove $E_1$, $E_2$ i $E_3$. Tipovi kardinalnosti mapiranja za ove veze mogu biti: 1:1:1, 1:1:M, 1:M:M i M:M:M. Kardinalnost M:M:M znači da su sve (jednostruke instance) veze između entiteta entitetskih tipova povezanih ternarnom vezom dopuštene. Kardinalnost 1:1:1 ograničava moguće asocijacije između entiteta entitetskih tipova $E_1$, $E_2$ i $E_3$: jedan par entiteta iz dva entitetska tipa, može biti u vezi samo sa jednim entitetom iz trećeg entitetskog tipa. Na sl. 2.26 su prikazani MOV dijagrami za ternarne veze sa kardinalnošću mapiranja 1:M:M i 1:1:M.




**Slika 2.26**: Kardinalnost mapiranja u ternarnoj vezi: a) 1:M:M i b) 1:1:M

U varijanti predstavljenoj na sl. 2.26a), kardinalnost je tipa M od $E_2$ i $E_3$ prema $E_1$ i može se jednoznačno interpretirati na slijedeći način: svaki par entiteta entitetskih tipova $E_2$ i $E_3$ je povezan sa najviše jednim entitetom entitetskog tipa $E_1$. Ovo implicira da vezu jednoznačno identifikuje $PK(E_2) \cup PK(E_3)$, pa je to primarni ključ veze.

U varijanti predstavljenoj na sl. 2.26b), kardinalnost je tipa M od $E_3$ prema $E_1$ i $E_2$, koji participiraju u tipu veze sa kardinalnošću 1. Interpretacija može biti dvojaka:

(1) Jedan entitet entitetskog tipa $E_3$ može biti povezan najviše sa jednim parom entiteta entitetskih tipova $E_1$ i $E_2$. Drugim riječima, entitet tipa $E_3$, ako učestvuje u vezi, identifikuje druga dva entiteta tipa $E_1$ i $E_3$ respektivno, sa kojima je u vezi. Implikacija je da je u ovom slučaju $PK(E_3)$ primarni ključ veze.

(2) Samo jedan entitet entitetskog tipa $E_1$ može biti povezan sa nekim parom entiteta tipa $E_2$ i $E_3$, respektivno. Analogno prethodnom, samo jedan entitet entitetskog tipa $E_2$ može biti povezan sa nekim parom entiteta tipa $E_1$ i $E_3$, respektivno. Drugim riječima, par entiteta tipa $E_2$ i $E_3$ respektivno, identifikuje entitet tipa $E_1$ sa kojima je u vezi. Analogno tome, par entiteta tipa $E_1$ i $E_3$ respektivno, identifikuje entitet tipa $E_2$ sa kojima je u vezi. Implikacija je da su kandidatski ključevi tipa veze $PK(E_2) \cup PK(E_3)$ i $PK(E_1) \cup PK(E_3)$. Ova interpretacija je češće korištena u literaturi.

Jednoznačna specifikacija ograničenja kardinalnosti mapiranja u *n-arnim* vezama je moguća korištenjem teorije funkcionalnih zavisnosti koja je izložena u poglavlju 5.2.

### Slabi entitetski skupovi ili višeznačni kompozitni atributi

Jedna od projektnih opcija prilikom izgradnje MOV dijagrama jeste modelovanje neke činjenice/objekta realnog svijeta slabim entitetskim tipom ili višeznačnim atributom. Kao što je prethodno opisano, preko identifikujuće

veze jedan ili više entiteta slabog entitetskog skupa/tipa se pridružuju jednom jakom entitetu jakog entitetskog skupa/tipa i svaki slabi entitet učestvuje u vezi sa najviše jednim jakim entitetom. Višeznačnim atributom se takođe reprezentuje pridruživanje više vrijednosti nekog svojstva nekom entitetu. Po analogiji, pridruživanje činjenica reprezentovanih slabim entitetima jakom entitetu se može reprezentovati kompozitnim višeznačnim atributom jakog entitetskog tipa, gdje kompozitni atribut reprezentuje svojstva slabog entitetskog tipa. Korištenje slabog entitetskog tipa ili višeznačnog kompozitnog atributa u procesu MOV modelovanja je rezultat odluke koju donosi projektant na bazi više orijentacionih i neformalnih pravila.

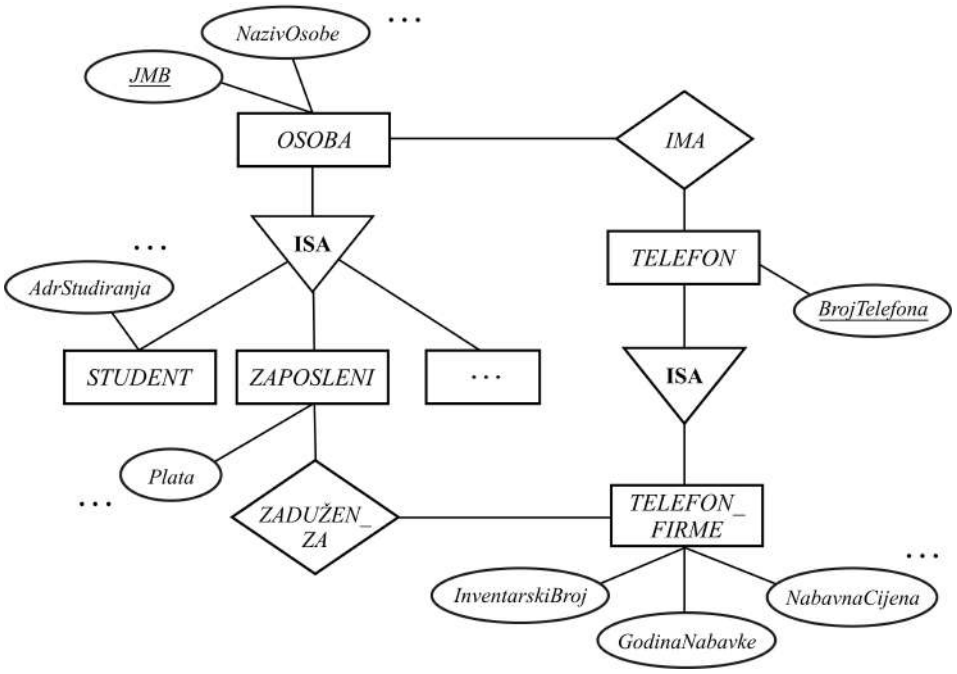
Ukoliko u modelu egzistira slabi entitetski tip sa jednim ili malim brojem atributa koji učestvuje u samo jednoj identifikujućoj vezi, i ako se činjenice reprezentovane atributima mogu posmatrati i kao svojstvo jakog, identifikujućeg entitetskog tipa, onda je adekvatnija predstava preko višeznačnog (kompozitnog) atributa jakog entitetskog tipa umjesto slabog entitetskog tipa i identifikujuće veze. Komponente kompozitnog atributa su atributi slabog entitetskog tipa (ukoliko slabi entitetski tip sadrži više atributa).

Ukoliko slabi entitetski tip sadrži veći broj atributa, ili učestvuje u više veza sa drugim entitetskim tipovima, onda je za reprezentaciju takve situacije u modelu adekvatno koristititi slabi entitetski tip.

### Korištenje specijalizacije

Korištenje specijalizacije omogućava da se zajednički atributi sličnih entitetskih tipova predstave na jednom mjestu, kao i da se predstave specifične veze pojedinih specijalizovanih entitetskih tipova nižeg nivoa sa drugim entitetskim tipovima. Za ilustraciju, posmatrajmo pomenuti primjer modelovanja podataka o telefonu i vezu tog podatka sa osobama. Kao što je prethodno navedeno, može biti od interesa da se telefon vodi kao višeznačni atribut koji reprezentuje kontakt informacije za osobu. S druge strane, od interesa mogu biti i drugi podaci o telefonu: inventarski broj, godina nabavke, nabavna vrijednost, tip telefona (mobilni/fiksni), prostorija u kojoj se telefon nalazi (ako je telefon fiksni), dobavljač, itd. Dodatne informacije o telefonu su od interesa samo ako je telefon vlasništvo firme za koju se radi model podataka. Postavlja se pitanje kako modelovati podatke o telefonu kada je za neke situacije dovoljna informacija samo o broju telefona (telefoni studenata, privatni telefoni zaposlenih i telefoni drugih osoba), dok je u situacijama gdje su telefoni vlasništvo firme potrebno modelovati pored broja telefona i druge informacije.

Jedno rješenje jeste da se broj telefona vodi kao višeznačni atribut osobe, a da se posebnim entitetskim tipom (sa dodatnim atributima) reprezentuju telefoni koji se vode kao vlasništvo – osnovno sredstvo firme. Problem sa ovakvim rješenjem je što bi se neki brojevi telefona vodili na više mjesta (u višeznačnom atributu osobe *BrojTelefona* i u posebnom entitetskom skupu za reprezentaciju telefona), što komplikuje ažuriranje i predstavlja potencijalnu opasnost za integritet podataka u bazi. Odgovarajuće rješenje kojim se eliminiše navedeni problem, jeste MOV model sa korištenjem specijalizacije, kao na sl. 2.27.



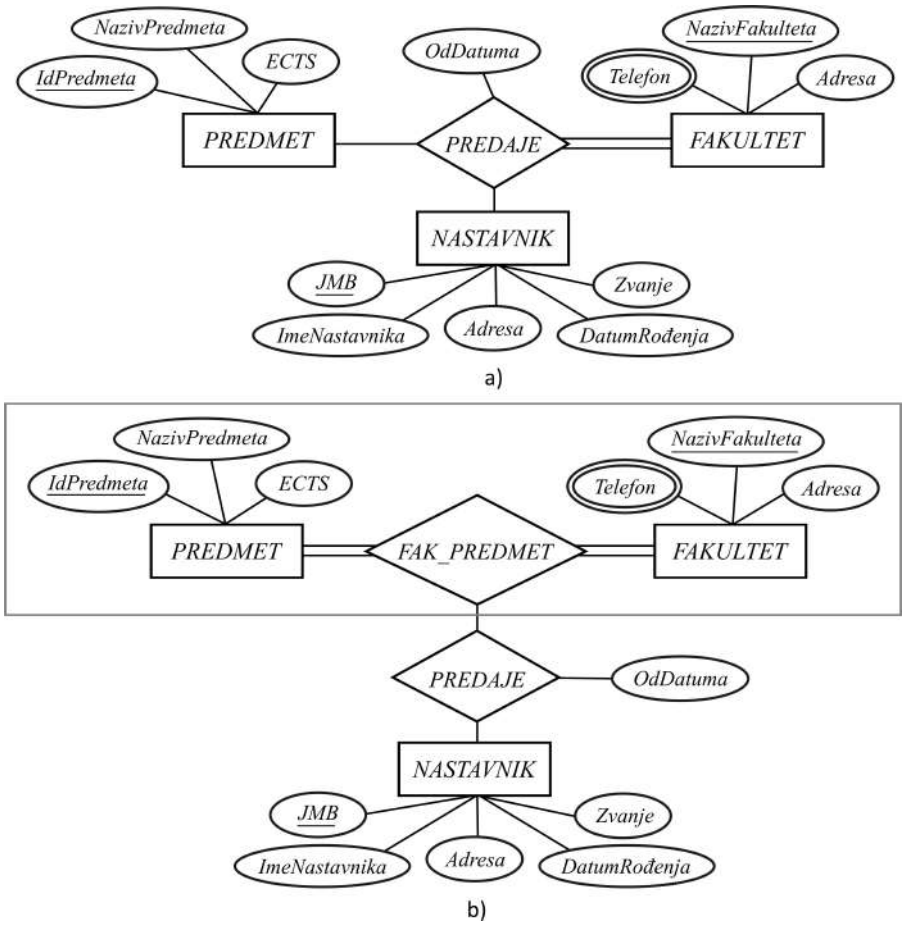

**Slika 2.27**: Projektno rješenje bazirano na konceptu specijalizacije

Zajednički podatak o telefonu (*BrojTelefona*) se vodi na jednom mjestu za sve opcije – u superklasi *TELEFON*. Dodatni podaci koji su od interesa za telefone u vlasništvu firme se vode u potklasi *TELEFON_FIRME*. Ovakav model omogućava da se preko veze *IMA* specifikuju informacije o kontaktima – brojevima telefona osobe, dok se vezom *ZADUŽEN_ZA* specifikuju zaduženja zaposlenih osoba za telefone firme. Ujedno, navedenim rješenjem definišemo da samo zaposlena osoba može biti zadužena za telefon u vlasništvu firme.

### Korištenje agregacije

Korištenje agregacije ima svoje specifičnosti u odnosu na *n-arne* veze. Koju opciju izabrati zavisi od konteksta i semantike koju želimo da modelujemo. Za ilustraciju posmatrajmo ponovo ternarnu vezu koja povezuje entitetske tipove *NASTAVNIK*, *PREDMET* i *FAKULTET* (sl. 2.28a) i alternativni model koji koristi agragaciju (sl. 2.28b).

Ternarna veza jasno reprezentuje činjenicu da neki nastavnik predaje određeni predmet na nekom fakultetu (od nekog datuma). Ukoliko se na nekom predmetu ne izvodi nastava (na primjer iz razloga što za izborni predmet nije bio dovoljan broj prijavljenih kandidata), onda ternarnom vezom ne bismo mogli predstaviti činjenicu da je neki predmet povezan sa




**Slika 2.28**: Ternarna veza (a) u poređenju sa konceptom agregacije (b)

fakultetom (da je u nastavnom planu studijskih programa fakulteta), ali da se ne izvodi nastava na tom predmetu. Ukoliko želimo da predstavimo i te informacije, treba paralelno sa ternarnom vezom ustanoviti posebnu binarnu vezu *FAK_PREDMET* između entitetskih tipova *PREDMET* i *FAKULTET*.

Za navedeni primjer, agregacijom se može odvojeno predstaviti činjenica o pridruženosti predmeta fakultetima (vezom *FAK_PREDMET*, da je predmet u nastavnom planu studijskih programa fakulteta), i činjenica o pridruženosti nastavnika predmetu na fakultetu (vezom *PREDAJE*). Obje veze mogu imati svoje vlastite opisne atribute. U varijanti sa agregacijom možemo dodatno da predstavimo i ograničenje totalnosti participacije *PREDMET*-a u vezi sa *FAKULTET*-om. Ukoliko bismo htjeli, primjera radi, da modelujemo situaciju da određeni predmet na nekom fakultetu može da predaje samo jedan profesor, u varijanti sa korištenjem agregacije bi se to izrazilo kardinalnošću 1 (na strani veze *PREDAJE* prema *NASTAVNIK*-u) i jednostavnije mentalno interpretiralo nego u varijanti sa ternarnom vezom.

Iz prethodnih razmatranja se vidi da, iako se na prvi pogled agregacijom i *n-arnom* vezom predstavlja ista stvar, postoje suptilne distinkcije koje treba imati u vidu prilikom izbora odgovarajuće opcije u procesu modelovanja realnog sistema.

### 2.1.9 Primjer kreiranja MOV dijagrama

Ilustrujmo proces kreiranja MOV dijagrama na primjeru izrade pojednostavljenog konceptualnog modela univerzitetskog informacionog sistema. Pretpostavimo da smo u procesu analize identifikovali slijedeće informacione potrebe hipotetičkog univerzitetskog sistema.

- Univerzitet je obrazovna institucija koja ima naziv, adresu (sjedište) i određen broj telefonskih brojeva za kontakt, u čiji sastav ulazi određeni broj fakulteta. Svaki fakultet ima naziv, adresu i određen broj telefonskih brojeva za kontakt. Na fakultetima se izvodi nastava na jednom ili više studijskih programa. Studijski program karakteriše naziv studijskog programa, ciklus studija (I ciklus – dodiplomski, II ciklus – master/postdiplomski, III ciklus – doktorski studij), trajanje (broj godina/semestara), ukupan broj ECTS kredita koji se stiče studiranjem na studijskom programu, kao i naziv zvanja koji se stiče završetkom studijskog programa. Nastavnim planom studijskog programa se definišu predmeti koje sadrži studijski program, semestar u kojem se po nastavnom planu izvodi nastava na predmetu i tip predmeta (obavezni/izborni).

- Predmete karakteriše jedinstven identifikator/kod, naziv predmeta i broj ECTS bodova. Svi predmeti su jednosemestralni. Jedan predmet može da se izvodi na više studijskih programa, jednog ili više fakulteta. Za svaki predmet je matičan određeni fakultet.
- Na univerzitetu je neophodno voditi evidenciju o različitim osobama. Svaka osoba ima jedinstven identifikator (matični broj), ime, datum rođenja i adresu. Za odvijanje osnovnih procesa na univerzitetu – nastavnih i naučnoistraživačkih, ključne grupe osoba su nastavnici, asistenti/saradnici i studenti. Ostale osobe su u funkciji podrške za odvijanje osnovnih procesa na univerzitetu. Nastavu na fakultetu izvode zaduženi nastavnici i saradnici (asistenti). Na izvođenju nastave moraju biti angažovani nastavnici, dok na nekim (teorijskim) predmetima vježbe nisu obavezne. Nastavnici i asistenti su osobe koje mogu biti zaposlene na univerzitetu, ili spoljni saradnici. Svaki zaposleni ima odgovarajuću platu. Spoljne saradnike karakteriše institucija u kojoj su zaposleni. Za angažovanje na univerzitetu, spoljni saradnici zaključuju ugovor. Nastavnici i asistenti imaju nastavnička odnosno asistentska zvanja. Nastavnici i asistenti mogu izvoditi nastavu na više predmeta i na više studijskih programa/fakulteta univerziteta. Dekan koji rukovodi fakultetom i rektor koji rukovodi univerzitetom, biraju se iz redova zaposlenih nastavnika univerziteta.
- Student je osoba koja je upisana na neki studijski program nekog fakulteta. Prilikom upisa na fakultet studenti dobijaju indeks sa jedinstvenim brojem indeksa za studijski program koji upisuju. Studenti nekog studijskog programa upisuju odgovarajući semestar i predmete (i tokom semestra prisustvuju nastavi) i polažu ispite koji se organizuju u terminima ispitnih rokova i na određenim lokacijama. Nakon izlaska na ispit, student dobija odgovarajuću ocjenu. Za svaki ispit treba da postoji mogućnost registrovanja broja studenata koji su izašli na ispit, i broja studenata koji su položili ispit. Studenti na nekom studijskom programu (II i III ciklusa) mogu biti ujedno i asistenti na nižim ciklusima studija.

Koraci u procesu kreiranja MOV dijagrama kojim reprezentujemo konceptualni model univerzitetskog informacionog sistema prema prethodnim zahtjevima, jesu:

**1. Identifikacija entitetskih tipova.** Na osnovu analize opisa i zahtjeva za univerzitetski sistem, možemo identifikovati slijedeće entitetske tipove i njihove atribute:

- Entitetski tip *UNIVERZITET* ima atribute *NazivUni*, *AdresaUni* i *TelefonUni*. S obzirom na to da se zahtijeva evidentiranje samo telefonskih kontakt brojeva, telefon je modelovan kao višeznačni atribut. *NazivUni* selektujemo kao primarni ključ, iako u bazi podataka (jednog) univerziteta imamo samo jedan entitet tog entitetskog tipa.
- Entitetski tip *FAKULTET* ima atribute *NazivFak* (naziv fakulteta), *AdresaFak* (adresa fakulteta) i *TelefonFak*. Za atribut *TelefonFak* važi ista napomena kao i za atribut *TelefonUni*. *NazivFak* je jedinstven u okviru univerziteta, pa možemo da ga selektujemo kao primarni ključ entitetskog tipa.
- Entitetski tip *STUD_PROGRAM* ima atribute *NazivSP*, *Ciklus*, *Trajanje*, *UkupnoECTS* i *Zvanje*. Napomena: opciono, ciklus studija bi mogao da se modeluje i kao entitetski tip, ako bi se za ciklus studija vodili dodatni podaci. Iako možemo da pretpostavimo da u okviru univerziteta nemamo dva studijska programa sa istim imenom, što omogućava da se *NazivSP* izabere kao primarni ključ, ipak ćemo uvesti atribut *IdSP* kao jedinstveni identifikator studijskog programa, odnosno kao primarni ključ, jer uobičajeno studijski programi imaju svoj kod.
- Entitetski tip *PREDMET* ima atribute *IdPredmeta*, *NazivPredmeta* i *ECTS*. *IdPredmeta* jedinstveno identifikuje predmet u okviru univerziteta, pa ga selektujemo kao primarni ključ.
- Entitetski tip *OSOBA* ima atribute *JMB*, *Ime*, *DatumRođenja* i *AdrOsobe*. Jedinstven identifikator (primarni ključ) je *JMB*.
- Po kriterijumu radnog odnosa, osobe možemo podijeliti u dvije grupe reprezentovane potklasama *ZAPOSLENI* i *HONORARNI*. Tip *ZAPOSLENI* ima, pored atributa entitetskog tipa *OSOBA* i dodatni atribut *Plata*. Tip *HONORARNI* ima, pored atributa entitetskog tipa *OSOBA*, i dodatne atribute *Institucija* (iz koje osoba dolazi) i *Ugovor* (po kojem je angažovana). Po kriterijumu uloge osoba u nastavnim procesima na univerzitetu, osobe (od interesa) možemo svrstati u grupe reprezentovane potklasama *NASTAVNIK*, *ASISTENT* i *STUDENT*. Dalje, nastavnik koji je zaposlen (sa punim radnim vremenom) je tipa *NASTAVNIK_RO*, dok je nastavnik u dopunskom radnom odnosu tipa *HON_NASTAVNIK* (slično za *ASISTENT_RO* i *HON_ASISTENT*). Asistent koji studira na nekom od viših ciklusa studija je tipa *STUDENT_ASISTENT*.
- Entitetski tip *NASTAVNIK* karakterišu atributi entitetskog tipa *OSOBA* i dodatno atribut *NastZvanje*.

- Entitetski tip *ASISTENT* dodatno karakteriše, osim atributa entitetskog tipa *OSOBA*, atribut *SaradnZvanje*.
- Zaposleni nastavnici (*NASTAVNIK_RO*) i asistenti (*ASISTENT_RO*) imaju dodatni atribut *Plata* (u odnosu na entitetske tipove *NASTAVNIK*, odnosno *ASISTENT*), koji karakteriše zaposlene osobe. *NASTAVNIK_RO* je potklasa i entitetskog tipa *NASTAVNIK* i entitetskog tipa *ZAPOSLENI* (višestruko nasljeđivanje) i ima dodatni specifični atribut *Kabinet*, koji specifikuje lokaciju njegovog radnog mjesta. Slično, *ASISTENT_RO* je potklasa i entitetskog tipa *ASISTENT* i entitetskog tipa *ZAPOSLENI*, i ima dodatni atribut *Laboratorija*, koji specifikuje laboratoriju za koju je asistent zadužen.
- Honorarno zaposleni nastavnici i asistenti (*HON_NASTAVNIK*, *HON_ASISTENT*) nisu u punom radnom odnosu i imaju dodatne atribute *Institucija* (u kojoj su zaposleni) i *Ugovor* (o djelu/povremenim poslovima) u odnosu na entitetske tipove *NASTAVNIK* odnosno *ASISTENT*. *HON_NASTAVNIK* je potklasa i entitetskog tipa NASTAVNIK i entitetskog tipa *HONORARNI* (višestruko nasljeđivanje). Slično, *HON_ASISTENT* je potklasa i entitetskog tipa *ASISTENT* i entitetskog tipa *HONORARNI*.
- Entitetski tip *STUDENT* ima dodatni atribut *AdrStudiranja* u odnosu na tip *OSOBA*, koji sadrži informaciju o adresi studenta tokom studija. Student može biti upisan na više različitih studijskih programa (I, II, III ciklusa) u određeni semestar. Pri upisu nekog studijskog programa, student dobija indeks.
- Studenti koji su istovremeno i asistenti reprezentuju se tipom *STUDENT_ASISTENT*, koji ima sve atribute koje imaju entitetski tipovi *STUDENT* i *ASISTENT* (dobija ih višestrukim nasljeđivanjem iz entitetskih tipova *STUDENT* i *ASISTENT*).
- Entitetski tip *ISPIT* ima atribute *DatumIspita* i *Lokacija*. Atributi *Izašlo* (broj studenata koji su izašli na ispit) i *Položilo* (broj studenata koji su položili ispit) jesu izvedeni atributi, jer se mogu izračunati na osnovu broja veza studenata sa ispitom, i analizom dobijenih ocjena. Napomena: *ISPIT* nema primarni ključ (neki ispiti iz različitih predmeta se mogu održati istog datuma na istoj lokaciji). Ispit ne može postojati bez predmeta, pa se mora modelovati kao slabi entitetski tip. Diskriminator entitetskog tipa *ISPIT* je *DatumIspita*, jer se ispiti koji se odnose na neki predmet razlikuju po datumu održavanja ispita.

**2. Identifikacija tipova veze.** Na osnovu analize opisa i zahtjeva možemo identifikovati slijedeće tipove veza:

- Na osnovu opisa možemo identifikovati više specijalizacija tipa *OSOBA* koje su od interesa. Kao što je prethodno navedeno, po kriterijumu radnog odnosa imamo specijalizaciju na *ZAPOSLENI* i *HONORARNI*, a po kriterijumu uloge osoba u procesima na univerzitetu imamo specijalizacije tipa *OSOBA* na: *NASTAVNIK*, *ASISTENT* i *STUDENT*. Navedene potklase su povezane sa superklasom *OSOBA* ISA vezom. Ostavljamo mogućnost da imamo registrovane osobe koje ne pripadaju nijednoj navedenoj potklasi, pa je specijalizacija natklase *OSOBA* parcijalna. S obzirom na to da entiteti tipa *OSOBA* ne mogu biti istovremeno i *ZAPOSLENI* i *HONORARNI*, odnosna specijalizacija je disjunktna. Neka osoba može biti istovremeno i tipa *ASISTENT* i tipa *STUDENT*, pa je specijalizacija po kriterijumu uloge osoba preklapajuća.
- Slično, u okviru entitetskog tipa *NASTAVNIK* (slično *ASISTENT*) možemo uočiti grupe entiteta sa određenim specifičnostima. Neki nastavnici i asistenti su stalno zaposleni (u radnom odnosu), dok su drugi nastavnici i asistenti angažovani po ugovoru kao honorarni nastavnici i saradnici. Specijalizaciju nastavnika na podgrupe *NASTAVNIK_RO* i *HON_NASTAVNIK* modelujemo disjunktnom (*disjoint*) ISA vezom. Slično je i sa specijalizacijom entitetskog tipa *ASISTENT*, kao i sa specijalizacijom tipova *ZAPOSLENI* i *HONORARNI*.
- *IMA_U_SASTAVU* je veza koja povezuje (jedan) univerzitet sa više entiteta tipa *FAKULTET* koji su u sastavu univerziteta, pa je veza tipa 1:M, a participacija entitetskih skupova sa obje strane veze je totalna. Svaki fakultet mora biti u sastavu univerziteta, a univerzitet mora biti u vezi sa fakultetima koji su u njegovom sastavu.
- Svaki fakultet organizuje nastavu na jednom ili više studijskih programa, dok je svaki studijski program vezan za jedan fakultet. Veza *IMA* koja povezuje entitetske tipove *FAKULTET* i *STUD_PROGRAM* je zbog toga tipa 1:M. S obzirom na to da svaki fakultet mora biti u vezi sa (barem jednim) studijskim programom i da studijski program ne može da egzistira bez fakulteta na kojem se studijski program organizuje, participacija oba entitetska tipa u vezi je totalna.
- S obzirom na to da jedan predmet može da bude u nastavnom planu više studijskih programa univerziteta i da normalno studijski program sadrži više predmeta, tip veze *SADRŽI* koji povezuje entitetske tipove *PREDMET* i *STUD_PROGRAM* ima kardinalnost M:M. S obzirom na to da svaki studijski program i predmet moraju participirati u vezi (ne postoji studijski program bez predmeta, niti predmet može

egzistirati samostalno bez povezanosti sa studijskim programom), participacija entitetskih tipova *PREDMET* i *STUD_PROGRAM* u vezi *SADRŽI* je totalna.

- Studijski program upisuje više studenata. Student tipično upisuje jedan studijski program tokom studiranja. Ako ostavimo mogućnost da student tokom vremena može da upiše više od jednog studijskog programa na univerzitetu (na različitim ciklusima), onda je kardinalnost mapiranja tipa veze *UPISAN_NA* između entitetskih tipova *STUDENT* i *STUD_PROGRAM* M:M.

  Analizirajući procese na univerzitetu, modelom je ostavljena mogućnost da na nekom studijskom programu nema upisanih studenata. Ovo je moguće ako se organizuje novi studijski program za koji je dobijeno odgovarajuće odobrenje (licenca), pri čemu još nije počeo upis na studijski program, ili ako za neki odobreni studijski program nema dovoljno zainteresovanih kandidata. Zbog prethodno navedenog, participacija tipa *STUDENT* u vezi je totalna, a entitetskog tipa *STUD_PROGRAM* parcijalna.

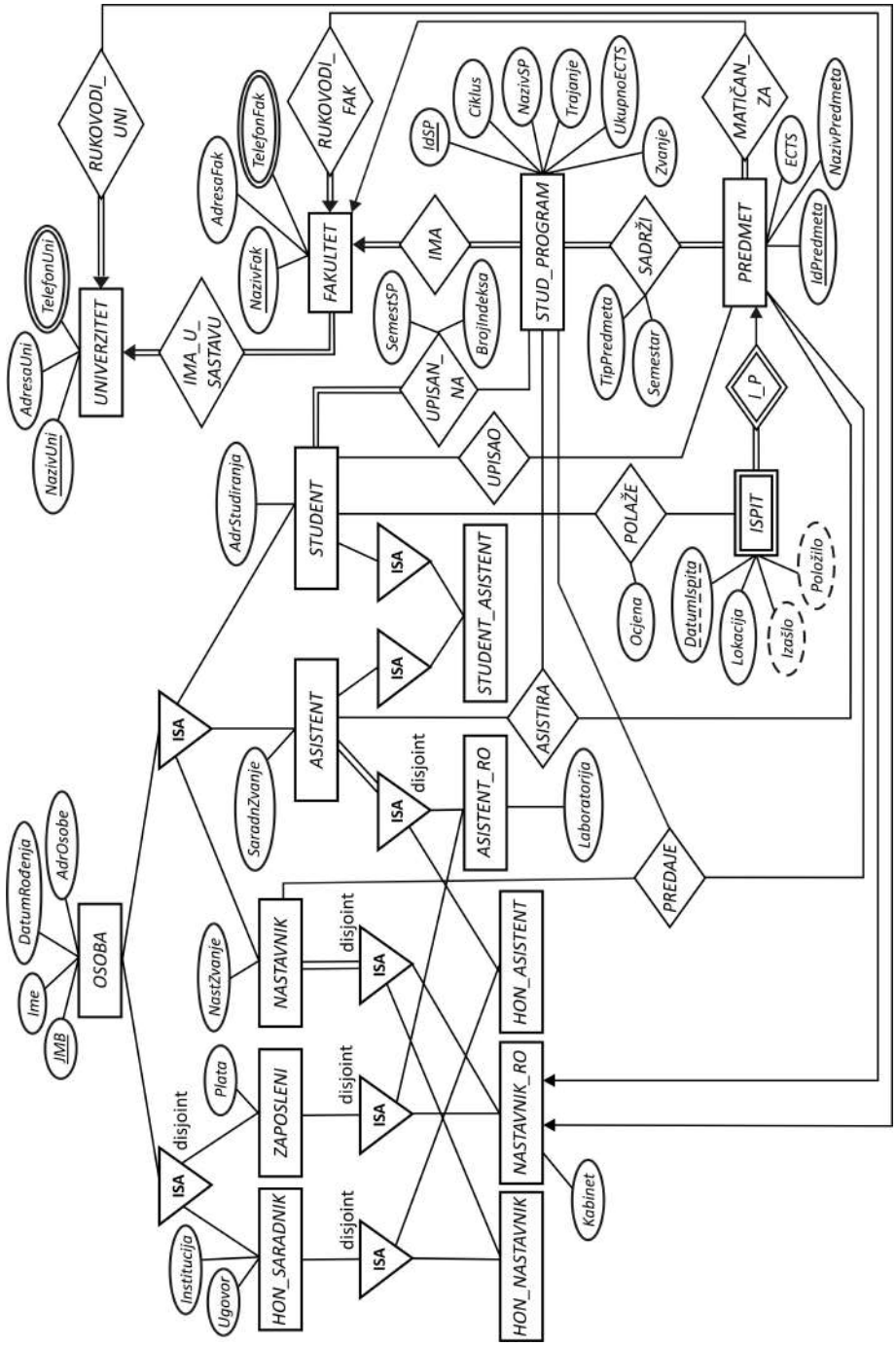
  Studentu se dodjeljuje indeks za svaki studijski program koji je upisao. Ispunjavanjem uslova (praćenjem nastave i vježbi te izradom zadaća i polaganjem testova i ispita) student stiče uslove za upis slijedećeg semestra, odnosno godine studija. Zbog pojednostavljenja, u ovom modelu je predviđeno vođenje evidencije samo o posljednjem upisanom semestru, iako bi u realnom modelu bilo potrebno voditi istoriju upisa semestara uz neke dodatne atribute (npr. datum upisa semestra).
- Tip veze *UPISAO* definiše moguće veze između studenata i predmeta. Jedan student može upisati više predmeta, a jedan predmet može da upiše više studenata. Neki izborni predmet može egzistirati u studijskom programu, a da ga nije upisao nijedan student. Ako pretpostavimo da student pri upisu na fakultet prvo upisuje studijski program, a naknadno upisuje semestar i bira predmete, možemo zaključiti da u određenom periodu mogu postojati studenti koji (još) nisu upisali nijedan predmet, pa je participacija entitetskih tipova *STUDENT* i *PREDMET* u tipu veze *UPISAO* parcijalna sa obje strane.

  Primijetimo da MOV modelom na sl. 2.29 nije specifikovano ograničenje da student može da upiše samo predmet sa studijskog programa koji je upisao.
- Pretpostavimo da je za svaki predmet matičan jedan fakultet, a da je fakultet (tipično) matičan za više predmeta. Pretpostavimo da mogu postojati i fakulteti za multidisciplinarne studije koji nisu matični ni za

jedan predmet. Tip veze *MATIČAN_ZA* onda ima kardinalnost mapiranja M:1 od entitetskog tipa *PREDMET* prema entitetskom tipu *FAKULTET*, pri čemu je participacija entitetskog tipa *PREDMET* u vezi totalna, a *FAKULTET*-a parcijalna.

- *ISPIT* je slabi entitetski tip, egzistencijalno zavisan od entitetskog tipa *PREDMET*, pa je sa njim povezan identifikujućom vezom sa mapiranjem M:1 od *ISPIT*-a ka *PREDMET*-u, pri čemu je participacija *ISPIT*-a u vezi totalna.
- Student može da polaže više ispita, neki ispit (tipično) polaže više studenata, pa je kardinalnost mapiranja tipa veze *POLAŽE* M:M. S obzirom na to da studenti koji su tek upisali studijski program i predmete, i koji su tek počeli da slušaju nastavu nisu polagali ispite, participacija entitetskog tipa *STUDENT* u vezi *POLAŽE* je parcijalna. Na neki ispit ne mora izaći nijedan student, pa je participacija entitetskog tipa *ISPIT* u vezi takođe parcijalna.
- Situaciju da predmet (na jednom ili više studijskih programa) predaje jedan ili više nastavnika, kao i da nastavnik predaje određen broj predmeta (uključujući i 0) na različitim studijskim programima, predstavili smo ternarnom vezom *PREDAJE*, sa kardinalnošću mapiranja M:M:M, koja definiše moguće veze entiteta tipa *NASTAVNIK*, *PREDMET* i *STUD_PROGRAM*. Ova veza reprezentuje trenutnu situaciju o zaduženjima i izvođenju nastave od strane nastavnika na predmetima studijskih programa.

  Ako bismo željeli da modelujemo vezu nastavnika sa predmetima koje oni mogu da predaju (tj. za koje posjeduju odgovarajuće kompetencije), onda treba da modelujemo posebnu binarnu vezu *KOMPETENTAN_ZA* tipa M:M između *NASTAVNIK*-a i *PREDMET*-a.
- Analogno prethodnom je i opis, zapažanja i diskusija za ternarnu vezu *ASISTIRA*.
- S obzirom na to da nekim fakultetom rukovodi (kao dekan) samo jedan nastavnik koji je zaposlen na univerzitetu, i da neki nastavnik zaposlen na univerzitetu može da rukovodi (da bude dekan) samo jednim fakultetom, veza *RUKOVODI_FAK* između entitetskih tipova *NASTAVNIK_RO* i *FAKULTET* ima kardinalnost mapiranja 1:1. Sa pretpostavkom da fakultet uvijek mora da ima rukovodioca (dekana), a da svi nastavnici zaposleni na univerzitetu ne rukovode fakultetima (nisu dekani), participacija entitetskog tipa *FAKULTET* u vezi je totalna, a participacija entitetskog tipa *NASTAVNIK_RO* je parcijalna.
- Slično prethodnom je i diskusija za vezu *RUKOVODI_UNI*.




**Slika 2.29**: Pojednostavljeni MOV dijagram baze podataka univerziteta

MOV dijagram kreiran na osnovu prethodne analize prikazan je na sl. 2.29. Dijagram nije kreiran u jednosmjernom procesu, nego u više iterativno-inkrementalnih koraka, tokom kojih su rješavane projektne dileme i vršen izbor između raspoloživih opcija. Na primjer, postoji više načina na koje se specijalizacijom mogu predstaviti različite grupe osoba od interesa za univerzitetski sistem. Posebno su analizirana i u iteracijama precizirana i korigovana ograničenja kardinalnosti/participacije entitetskih tipova u vezama. Može se primijetiti da neke konstatacije i specifikacije vezane za kardinalnosti/participacije entitetskih tipova u vezama nisu direktno navedene u zahtjevima, pa se mogu smatrati rezultatom naknadne analize i pojašnjenja zahtjeva.

Potrebno je napomenuti da je realni sistem, naravno, mnogo kompleksniji, i da bi konceptualni model realnog sistema morao da reprezentuje mnoge druge činjenice i odnose između entiteta. Izloženi konceptualni model se odnosi na uprošteni univerzitetski sistem, koji je razumljiv velikom broju potencijalnih čitalaca, i dat je sa ciljem ilustracije procesa kreiranja MOV modela i primjene većine relevantnih MOV koncepata.

## 2.2 Modelovanje primjenom UML

UML (*Unified Modeling Language*) je grafički jezik za vizuelizaciju, specifikaciju, projektovanje i dokumentovanje softverskih sistema. Postao je vodeći standardizovani jezik za modelovanje softvera, koji je podržan u većini CASE (*Computer Aided Software Engineering*) alata. Jedan od razloga za široku upotrebu UML-a za modelovanje softverskih, ali i drugih 'nesoftverskih' sistema, leži u činjenici da je to samo bogata notacija[33] za modelovanje koja je nezavisna od procesa modelovanja. Ta karakteristika UML-a omogućava definisanje različitih pristupa modelovanju koji koriste raspoložive mogućnosti jezika u većoj ili manjoj mjeri za različite namjene. Drugi razlog proističe iz otvorenosti koncepta, jer je standardni UML otvoreni sistem koji može da se proširuje, odnosno specijalizuje za određenu oblast uvođenjem profila (eng. *profile*).

Početak razvoja UML-a veže se za zajednički rad trojice vodećih teoretičara O-O (objektno-orijentisane) metodologije (Grady Booch, Ivar Jacobson, James Rumbaugh). Inicijalna verzija (UML 0.9) nastala je 1996. godine kao rezultat unifikacije različitih pristupa i notacija (Booch, OMT

[33] UML je nastao objedinjavanjem više različitih notacija, pa se često u prevodima koriste odrednice kao što su: *objedinjeni*, *jedinstveni* ili *unificirani* jezik modelovanja.

i OOSE). Potom je, uz široku podršku partnera, značajno unaprijeđena arhitektura i formalizam, pa je početkom 1997. godine UML 1.0 predložen OMG[34]-u kao standard za modelovanje. Nakon značajnih unapređenja, krajem 1997. godine OMG je zvanično prihvatila UML kao standard za modelovanje (UML 1.1). Nakon toga je pod okriljem OMG-a uslijedio niz dopunjenih i unaprijeđenih specifikacija (1.2 – 1.5), koje se često označavaju kao **UML 1.x**. UML 1.4.2 specifikacija prihvaćena je kao ISO standard (ISO/IEC 19501).

Glavne zamjerke koje se odnose na mehanizme za proširenje standardnog jezika i upravljanje dokumentima, rezultovale su razvojem standarda poznatog kao **UML 2**. Superstruktura (novi tipovi dijagrama) je definisana 2004. godine, od kada se radi na razvoju[35] i unapređenju infrastrukture (odgovarajuće klase za superstrukturu), mehanizama za efikasniju razmjenu dijagrama itd.

### 2.2.1 Arhitektura UML-a

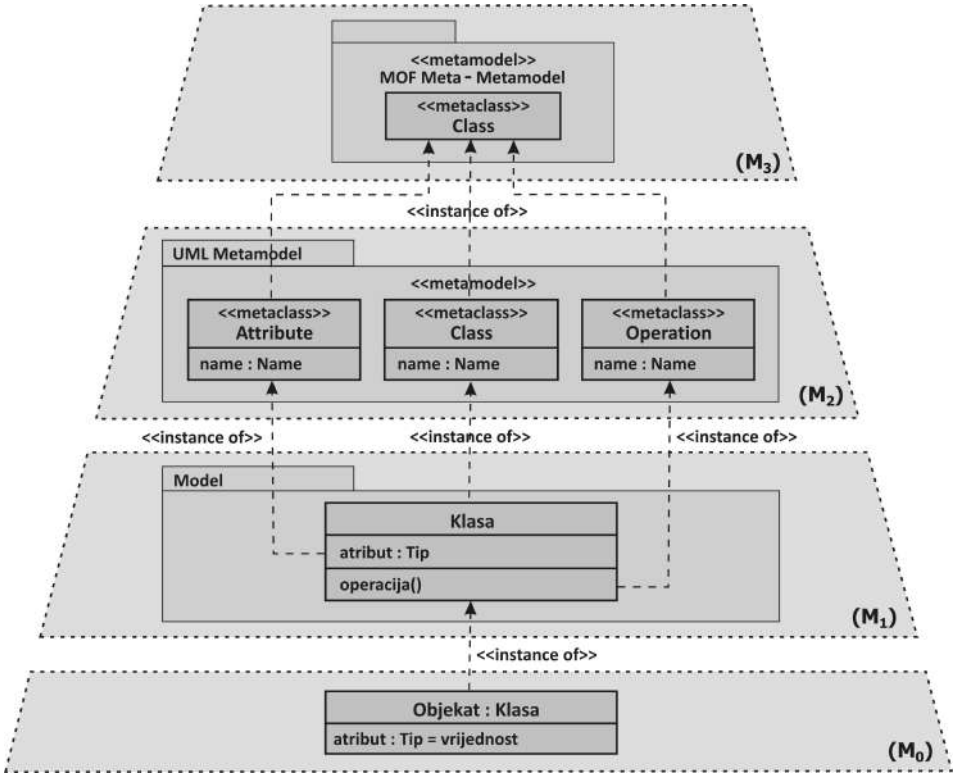
UML je definisan u okviru četvoroslojne arhitekture za modelovanje koja je definisana od strane OMG. Ovaj konceptualni okvir karakterišu četiri različita nivoa apstrakcije u modelovanju (sl. 2.30).

Najniži nivo (M0) je nivo konkretnih objekata, pa se modeli, koji se realizuju na ovom nivou, obično nazivaju **modeli objekata** ili **modeli instanci**. Svaki koncept na ovom nivou je instanca nekog koncepta sa viših nivoa (M1 ili M2). Na primjer, objekat *student1*, čiji atribut *ime* ima vrijednost 'Marko', jeste konkretan objekat na nivou M0. Objekat *student1* je instanca klase *Student* koja predstavlja koncept na višem nivou apstrakcije, tj. na nivou M1.

Nivo M1 je **nivo modela**. Ovdje se kreiraju UML modeli (npr. dijagram klasa) koji predstavljaju apstrakciju domena koji se modeluje. Koncepti koji se koriste na ovom nivou predstavljaju instance koncepata sa višeg nivoa (M2). Na primjer, prethodno pomenuta klasa *Student* i njena roditeljska klasa *Osoba*, te veza generalizacije kojom su te dvije klase povezane, predstavljaju koncepte na nivou M1, koji su instance tzv. *metaklasa* sa nivoa M2.

[34] OMG (*Object Management Group*) je neprofitabilni internacionalni IT konzorcijum koji predstavlja vodeću grupaciju za objektne tehnologije.

[35] Ova knjiga usklađena je sa specifikacijom 2.3 [39, 40] (maj 2010). Tokom pisanja knjige objavljena je i unaprijeđena verzija 2.4.1 (avgust 2011). UML specifikacije dostupne su na `www.omg.org/spec/UML/`.




**Slika 2.30**: Četvoroslojna OMG arhitektura za modelovanje

Nivo M2 je **nivo metamodela**, odnosno **nivo notacije**. Na ovom nivou apstrakcije formalizuju se koncepti karakteristični za podržane paradigme i definiše jezik za kreiranje modela. U konkretnom slučaju to je **UML metamodel**, koji sadrži definiciju metaklasa čije se instance koriste prilikom kreiranja konkretnih UML modela (dijagrama) na nivoima M0 i M1. Kao što je ranije navedeno, UML je otvoreni sistem, pa postoji mogućnost proširivanja UML metamodela (npr. specijalizacijom postojećih metaklasa).

Najviši nivo (M3) je **meta-metamodel**. To je nivo na kojem se definiše jezik za specifikaciju metamodela, tj. najosnovniji koncepti, tzv. stvari (eng. *things*) ili *meta-metaklase*, pomoću kojih se definišu gradivni elementi jezika za modelovanje.

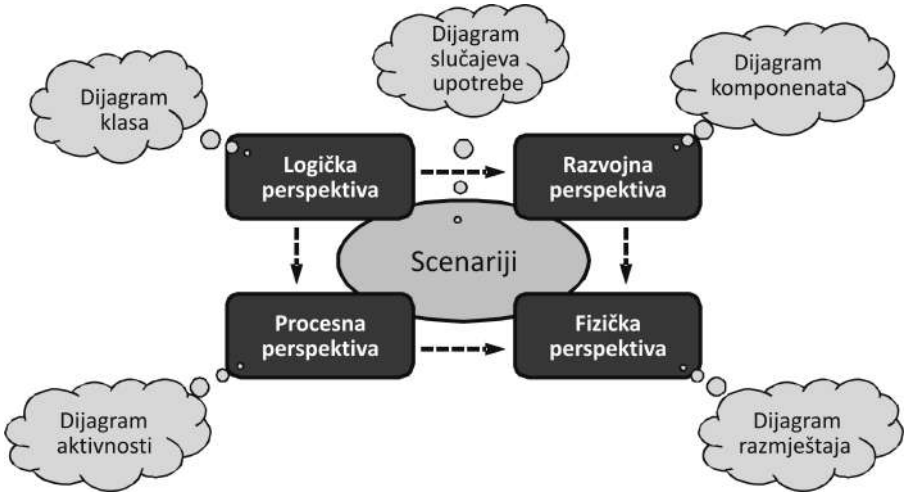
Detaljan opis UML metamodela, odnosno arhitekture samog jezika, prevazilazi ciljeve ovog poglavlja, pa u nastavku slijedi samo kratak opis **infrastrukture** i **superstrukture**, koje komplementarno predstavljaju aktuelnu UML specifikaciju.

**Infrastruktura** (eng. *infrastructure*) predstavlja jezgro (eng. *core*) jezika koje obezbjeđuje usaglašenost sa višim hijerarhijskim nivoom (MOF), a sadrži definicije primarnih tipova i osnovnih koncepata kao što su metaklase neophodne za formiranje UML metamodela i mehanizmi za proširenje metamodela, odnosno definisanje profila.

**Superstruktura** (eng. *superstructure*) je znatno obimnija i sadrži specifikaciju statičkih, odnosno strukturnih elemenata koji se koriste u kreiranju strukturnih dijagrama (kao što su dijagram klasa i dijagram komponenata), zatim specifikaciju dinamičkih elemenata, odnosno koncepata koji omogućavaju modelovanje ponašanja i interakcije u odgovarajućim dijagramima (kao što su dijagram aktivnosti, dijagram slučajeva upotrebe, itd.), te još neke dodatne koncepte, kao što su informacioni tokovi, šabloni, itd.

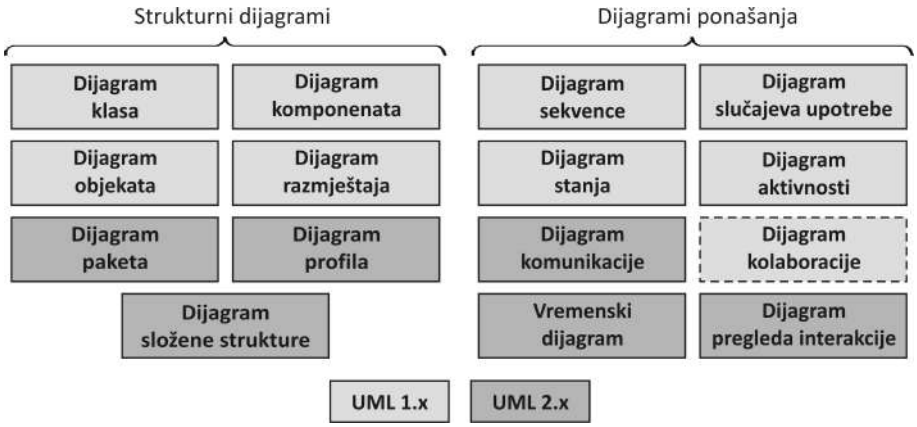
### 2.2.2 Standardni UML dijagrami

UML posjeduje bogatu (prevashodno) objektno-orijentisanu notaciju. Bogat skup od devet (UML 1.x), odnosno četrnaest (UML 2.x) različitih tipova standardnih dijagrama, omogućava primjenu UML-a u svim fazama procesa razvoja softverskog sistema, od specifikacije zahtjeva do implementacije u realnom sistemu. Inicijalne verzije (UML 1.x) uglavnom su imale podršku za tzv. "4+1" model arhitekture softverskih sistema [33], koji omogućava opis sistema iz četiri različite perspektive (logička, razvojna, procesna i fizička) i scenarije koji, kao peti (+1) pogled na sistem, služe za identifikaciju funkcionalnosti sistema i validaciju prototipova (sl. 2.31).




**Slika 2.31**: "4+1" model arhitekture softverskih sistema

Standardni UML dijagrami mogu da se svrstaju u dvije grupe (sl. 2.32). Prvu grupu čine **dijagrami za modelovanje strukture** (eng. *structure diagrams*), a drugu grupu **dijagrami za modelovanje ponašanja** (eng. *behaviour diagrams*). Strukturni dijagrami omogućavaju modelovanje statičkih aspekata, odnosno strukture objekata i sistema, dok dijagrami ponašanja omogućavaju modelovanje dinamičkih aspekata, odnosno ponašanje objekata i sistema tokom vremena. Treba imati u vidu da je, osim standardnih tipova dijagrama, moguće kreirati i druge tipove dijagrama koji mogu istovremeno da kombinuju i statičke i dinamičke elemente.




**Slika 2.32**: Standardni UML dijagrami

Većina UML 1.x dijagrama, koji su na sl. 2.32 prikazani svjetlijom nijansom, predstavlja sastavni dio UML 2 notacije. Dijagrami koji su dodatno uvedeni u UML 2 prikazani su tamnijom nijansom. Treba uočiti da je na slici prikazano ukupno 15 različitih tipova dijagrama. Naime, dijagram kolaboracije (UML 1.x) ne postoji u UML 2, jer je uveden tzv. *dijagram komunikacije* koji ima istu funkcionalnost.

Detaljnija specifikacija svih UML dijagrama prevazilazi pretenzije ove knjige. U nastavku će biti dati samo kratki opisi funkcionalnosti svih dijagrama, dok će, zbog uloge koju ima u projektovanju baze podataka, detaljno biti prikazan samo dijagram klasa.

### Strukturni dijagrami

**Dijagram klasa** (eng. *class diagram*) opisuje statičku strukturu sistema prikazujući klase i njihove međusobne veze. Dijagram klasa predstavlja notaciju koja se, s intenzivnim razvojem UML-a, sve više koristi za modelovanje podataka i biće detaljnije prikazana u nastavku ovog poglavlja.

**Dijagram objekata** (eng. *object diagram*) predstavlja konkretnu instancu dijagrama klasa, koja prikazuje konfiguraciju objekata u sistemu u nekom trenutku (eng. *snapshot*).

**Dijagram komponenata** (eng. *component diagram*) prikazuje softverske komponente sistema i njihove međusobne veze.

**Dijagram razmještaja** (eng. *deployment diagram*) omogućava modelovanje hardverskih resursa u implementaciji softverskog sistema i razmještaj softverskih komponenata na tim hardverskim resursima.

**Dijagram složene strukture** (eng. *composite structure diagram*) služi za opis unutrašnje strukture klase i kolaboracija koje klasa omogućava.

**Dijagram paketa** (eng. *package diagram*) omogućava prikaz logičkog grupisanja elemenata sistema u pakete.

**Dijagram profila** (eng. *profile diagram*) služi za modelovanje profila, odnosno specijalizaciju standardnog jezika.

### Dijagrami ponašanja

**Dijagram slučajeva upotrebe** (eng. *use case diagram*) omogućava prezentaciju funkcionalnosti sistema na visokom nivou apstrakcije. Dijagramom slučajeva upotrebe prikazuju su slučajevi upotrebe i učesnici. Slučaj upotrebe (eng. *use case*) predstavlja jednu funkcionalnost sistema, tj. niz aktivnosti koji stvara neku konkretnu prepoznatljivu vrijednost za krajnjeg korisnika. Subjekti kojima dati sistem obezbjeđuje konkretnu funkcionalnost, bez obzira na njihovu prirodu (ljudi, sistemi ili stvari), nazivaju se učesnici (eng. *actors*). Na primjer, neki karakteristični slučajevi upotrebe studentske službe jesu *PrijavaIspita* i *OvjeraSemestra*, jer su to tipične funkcionalnosti koje stvaraju konkretne prepoznatljive vrijednosti za *Studente* (učesnike).

**Dijagram aktivnosti** (eng. *activity diagram*) predstavlja jednu od najčešće korišćenih notacija za modelovanje toka aktivnosti (eng. *workflow*) u nekom procesu. Bogata notacija omogućava da se, osim toka kontrole, modeluju i tokovi objekata, pa se dijagram aktivnosti koristi za modelovanje procesa na različitim nivoima apstrakcije, i to kao: **dijagram makroaktivnosti** u inicijalnom poslovnom modelovanju za grubu specifikaciju aktivnosti, te kao **detaljni dijagram aktivnosti** koji detaljno specifikuje aktivnosti, sve učesnike u realizaciji poslovnog procesa (spoljni sistemi, radnici, organizacione jedinice, itd.) i sve objekte koji se koriste ili stvaraju u tom procesu.

**Dijagram stanja** (eng. *state machine diagram*) omogućava modelovanje stanja i mogućih promjena stanja nekog objekta tokom vremena, odnosno modelovanje životnog ciklusa instance neke klase.

**Dijagram sekvence** (eng. *sequence diagram*) modeluje komunikacije između objekata kroz razmjenu poruka tokom životnog vijeka objekata, fokusirajući se na vremensku dimenziju (redoslijed) komunikacije.

**Dijagram komunikacije** (eng. *communication diagram*) omogućava modelovanje interakcije objekata istovremeno prikazujući statičku strukturu i dinamičko ponašanje sistema, jer prikazuje dijagram objekata (statički aspekt) upotpunjen porukama koje razmjenjuju objekti (dinamički aspekt).

**Vremenski dijagram** (eng. *timing diagram*) predstavlja kombinaciju dijagrama sekvence i dijagrama stanja, koja prikazuje stanja objekta tokom vremena i poruke koje modifikuju stanja.

**Pregledni dijagram interakcije** (eng. *interaction overview diagram*) je posebna vrsta dijagrama aktivnosti u kojem čvorovi predstavljaju dijagrame interakcije (npr. dijagram sekvence, dijagram komunikacije i sl.).

### 2.2.3 Dijagram klasa

Dijagram klasa je strukturni UML dijagram koji omogućava modelovanje klasa i njihovih međusobnih veza, i predstavlja jednu od najčešće korišćenih notacija u O-O softverskom inženjerstvu. S obzirom na intenzivan razvoj i primjenu UML-a u svim fazama životnog ciklusa sofverskih sistema, dijagram klasa se često koristi i u projektovanju baza podataka, iako standardna notacija nije baš u potpunosti prilagođena toj namjeni. Zato se, osim standardnog dijagrama klasa, u praksi koriste i profili kojima se standardna notacija specijalizuje i prilagođava bilo za modelovanje na konceptualnom nivou, odnosno za razvoj platformski nezavisnih modela (eng. *Platform Independent Model* – PIM), bilo za modelovanje implementacionih detalja, odnosno razvoj platformski specifičnih modela (eng. *Platform Specific Model* – PSM). U nastavku je dat detaljniji opis standardnog dijagrama klasa i njegova primjena u konceptualnom modelovanju baze podataka.

#### Klase

Klasa (eng. *class*) je deskriptor za skup entiteta (objekata) sa sličnim svojstvima, ponašanjem i vezama sa drugim entitetima. Prema principu *inkapsulacije*[36], klase imaju dvije vrste članova: **atribute** koji reprezen-

[36] Inkapsulacija (eng. *encapsulation*) je princip koji podrazumijeva da se atributi i operacije posmatraju objedinjeno za istu vrstu objekata.

tuju svojstva objekata i **operacije** koje su dozvoljene (definisane) nad objektima. Klase bez definisanih operacija su analogne entitetskim tipovima koji su uvedeni u dijelu 2.1.1.

Klase se na dijagramu klasa prikazuju u obliku trodjelnog pravougaonika (sl. 2.33a), pri čemu gornja sekcija sadrži naziv klase, srednja atribute, a donja operacije. Kad detalji nisu neophodni, atributi i operacije ne moraju da se prikazuju (sl. 2.33b). Naziv klase tipično počinje velikim slovom (npr. *Osoba*, *Student*, itd.).

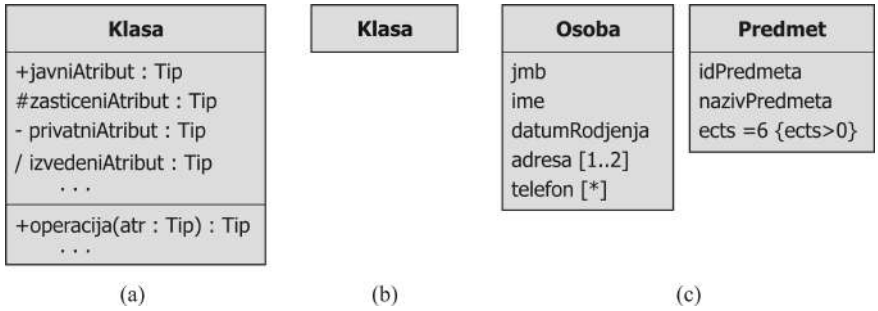
Specifikacija, odnosno definicija atributa ima opšti oblik:

$$[vidljivost]\ ime\ [:\ tip]\ [multiplikativnost]\ [=\ vrijednost]\ [uslovi]$$

gdje elementi unutar srednjih zagrada nisu obavezni. To omogućava da se dijagram klasa koristi za modelovanje statičke strukture sistema na različitim nivoima apstrakcije, odnosno sa različitim stepenom implementacionih detalja. U specifikaciji atributa jedino je obavezno ime, koje tipično počinje malim slovom (npr. *jmb*, *ime*, *datumRodjenja*, itd.).

Prema *principu skrivanja informacija* (eng. *information hiding*), članovi klase mogu da budu **javni** (eng. *public*), **zaštićeni** (eng. *protected*) i **privatni** (eng. *private*). To svojstvo članova klase se naziva *vidljivost* (eng. *visibility*) i na dijagramu klasa se prikazuje odgovarajućim simbolom, redom: +, #, **-** (sl. 2.33a). Međutim, iz perspektive konceptualnog modelovanja ova distinkcija nije značajna, pa će u nastavku teksta atributi biti prikazani bez oznake vidljivosti (standardom je dozvoljeno da se u inicijalnim fazama razvoja softverskog sistema klase prikazuju bez implementacionih detalja), kao što je prikazano na sl. 2.33c.

Izvedeni atributi (koncept uveden u dijelu 2.1.1) prikazuju se oznakom '/' ispred naziva atributa, kao što je ilustrovano na sl. 2.33a.




**Slika 2.33**: UML notacija za klase: (a) sa implementacionim detaljima, (b) bez atributa i operacija, (c) uobičajena za konceptualno modelovanje

Višeznačni atributi (koncept uveden u dijelu 2.1.1) definišu se tako što se za dati atribut specifikuje *multiplikativnost* (eng. *multiplicity*), odnosno *kardinalnost*, tj. minimalan i maksimalan broj vrijednosti datog višeznačnog atributa. Multiplikativnost višeznačnih atributa se specifikuje unutar srednjih zagrada kao interval koji započinje donjom nenegativnom cjelobrojnom granicom ($\geq 0$) i završava gornjom granicom, koja može biti neki konačan nenegativan cijeli broj ($\geq 0$) ili beskonačnost koja se označava zvjezdicom ($*$). Neke tipične multiplikativnosti su: [0..1], [1..1] ili kraće [1], [0..$*$] ili kraće [$*$], [1..$*$], itd. Podrazumijevana multiplikativnost je [1], odnosno [1..1], i ona se tipično ne prikazuje. Takvu multiplikativnost u klasi *Osoba* (sl. 2.33c) imaju atributi *jmb*, *ime* i *datumRodjenja*, dok su atributi *adresa* i *telefon* višeznačni. Atribut *adresa* ima multiplikativnost [1..2], što znači da svaka osoba mora da ima bar jednu adresu (npr. adresa prebivališta), ali i da osoba može da ima dvije adrese (osim adrese prebivališta, još i adresu boravišta). Atribut *telefon* ima multiplikativnost [$*$], odnosno [0..$*$], što znači da svaka osoba može da ima više telefona, ali mogu da postoje i osobe koje nemaju telefon. Multiplikativnost [0..1] znači da dati atribut može da ima maksimalno jednu vrijednost, ili da nema vrijednost (*null*).

Svakom atributu može da se definiše podrazumijevana vrijednost, koja se unutar definicije atributa specifikuje kao izraz čijim se izračunavanjem dobija podrazumijevana vrijednost datog atributa. Podrazumijevana vrijednost atributa *ects* u klasi *Predmet* iznosi 6 (sl. 2.33c).

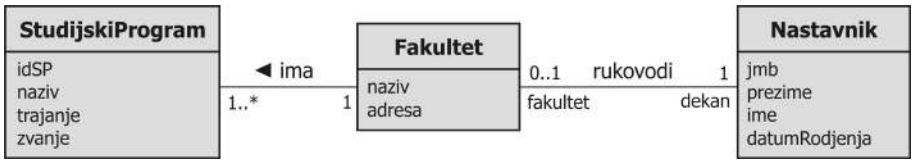
Atributima mogu da se specifikuju *uslovi*, odnosno *ograničenja*. Ograničenja mogu da se odnose na dozvoljene vrijednosti koje dati atribut može da ima, ili mogu da reprezentuju neke druge specifičnosti atributa, kao što su jedinstvenost (*unique*) ili uređenost (*ordered*) skupa vrijednosti kojima se reprezentuje dati višeznačni atribut, itd. Za svaki atribut može da se definiše zaseban skup ograničenja. Ograničenja se navode unutar velikih zagrada, a međusobno se odvajaju zapetama, npr. $\{ordered, unique\}$. Atribut *ects* (sl. 2.33c) ima ograničenje po vrijednosti $\{ects > 0\}$, što znači da svaki predmet mora da ima neki broj ECTS kredita.

Standardni UML nema skup simbola koji bi npr. omogućavao da se posebno označe atributi koji čine primarni ključ. Zato neki alati koriste specijalizovanu notaciju koja ima te mogućnosti. U literaturi se primarni ključ najčešće specifikuje korišćenjem stereotipova ili $\{PK\}$ ograničenja. Ne postoji ni standardna UML notacija za neke druge specifične koncepte koji postoje u MOV, kao što su slabi entitetski tipovi, složeni ili kompozitni atributi, itd. Međutim, postoje semantički ekvivalentni koncepti, o čemu će više riječi biti u nastavku ovog poglavlja.

### Asocijacije

**Asocijacija** (eng. *association*) je najopštija vrsta veze u UML-u. Pod pojmom *asocijacija* podrazumijeva se binarna veza, tj. veza između dvije i samo dvije klase. Prateći analogiju sa MOV, koncept UML asocijacije je pandan binarnom tipu veze između dva entitetska tipa u MOV, jer svaka asocijacija u UML dijagramu klasa predstavlja tip veze između objekata odnosnih klasa.

Asocijacija se u UML dijagramu klasa prikazuje punom linijom. Svaka asocijacija ima *naziv* koji se prikazuje na sredini (i najčešće iznad) linije. Dodatno se pored naziva asocijacije može prikazati i mali crni trougao, čija orijentacija ukazuje na redoslijed čitanja asocijacije. To semantički ne obogaćuje dijagram, ali olakšava čitanje asocijacije, posebno u nekim specifičnim domenima. Na primjer, iako inženjeri znaju da je rotor dio motora, tj. da motor ima rotor, laici ponekad teško mogu da odrede da li se asocijacija *ima* između klasa *Motor* i *Rotor* čita kao *Motor ima Rotor* ili *Rotor ima Motor*. Na dijagramu klasa sa sl. 2.34 asocijacija *ima* između klasa *StudijskiProgram* i *Fakultet* dodatno je označena simbolom za čitanje (*Fakultet ima StudijskiProgram*).




**Slika 2.34**: Primjeri binarnih asocijacija

*Krajevi* ili *završeci* asocijacije (eng. *association ends*), koji predstavljaju tačke konekcije asocijacije sa klasama, imaju veoma važnu ulogu u izražavanju bogate strukturalne semantike koju podržava UML. Završetak neke asocijacije tipično prikazuje *ulogu* koju objekti date klase imaju u datom tipu veze, *multiplikativnost* i *karakter asocijacije*.

*Uloga*, koju objekti date klase imaju u konkretnom tipu veze, jeste string koji se tipično prikazuje neposredno uz odgovarajući završetak asocijacije. Na dijagramu klasa sa sl. 2.34, u asocijaciji *rukovodi* između klasa *Nastavnik* i *Fakultet*, nastavnik ima ulogu *dekana* (dekan je nastavnik koji rukovodi fakultetom). Uloge se tipično prikazuju kad između klasa postoje različite, odnosno višestruke asocijacije, u kojima objekti iste klase imaju različite uloge (postoje i druge asocijacije između klasa *Nastavnik* i *Fakultet*, koje nisu prikazane na ovom dijagramu, a u kojima nastavnici

imaju drugačije uloge od uloge dekana), kod *rekurzivnih* ili *refleksivnih* asocijacija, tj. asocijacija koje vezuju klasu sa samom sobom (tada sigurno objekti iste klase imaju različite uloge), ili kad uloga ne može direktno da se prepozna iz samog naziva asocijacije. Kada je uloga objekata u nekoj asocijaciji očigledna (kao npr. u slučaju asocijacije *ima* između klasa *Fakultet* i *StudijskiProgram*), uloge ne moraju da se specifikuju.

*Multiplikativnost*[37] krajeva asocijacije služi za specifikaciju ograničenja učešća objekata odnosnih klasa u vezama datog tipa. Tipično se jedan kraj asocijacije naziva *source*, a drugi *target*. Multiplikativnost *source* kraja specifikuje minimalan i maksimalan broj objekata *source* klase (klasa na *source* strani) koji učestvuju u vezi datog tipa sa jednim objektom klase na *target* strani, i obrnuto, multiplikativnost *target* kraja specifikuje minimalan i maksimalan broj objekata *target* klase (klasa na *target* strani) koji učestvuju u vezi datog tipa sa jednim objektom *source* klase. Bogat skup mogućih ograničenja za multiplikativnost krajeva, kojima raspolaže UML, prikazan je na sl. 2.35.

| simbol | značenje |
|---|---|
| 1 | jedan |
| *m* | neki cijeli broj |
| 0..1 | nijedan ili jedan |

| simbol | značenje |
|---|---|
| 0..* ili * | nijedan ili više (proizvoljno) |
| 1..* | jedan ili više |
| *m..n* | najmanje *m*, a najviše *n* |

**Slika 2.35**: Multiplikativnosti krajeva asocijacije u dijagramu klasa

Dijagram klasa sa sl. 2.34 ilustruje tipičnu situaciju na univerzitetu koja podrazumijeva da svakim fakultetom rukovodi dekan iz reda nastavnika. Multiplikativnost kraja asocijacije *rukovodi* na *Nastavnik* strani je '1'. To znači da svaki fakultet ima jednog i samo jednog dekana, dok multiplikativnost '0..1' na strani *Fakultet* znači da ima nastavnika koji jesu dekani, ali i nastavnika koji nisu dekani, jer svaki nastavnik, po osnovu asocijacije *rukovodi*, može da bude u vezi sa jednim ili nijednim fakultetom. Druga asocijacija modeluje činjenicu da svaki fakultet može da ima jedan ili više studijskih programa, a da je za organizaciju svakog studijskog programa nadležan samo jedan fakultet.

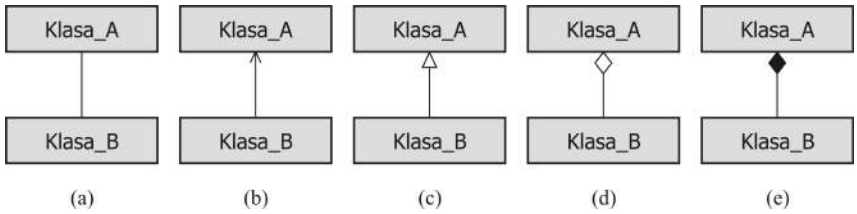
Treba uočiti da multiplikativnosti krajeva asocijacije istovremeno specifikuju i *kardinalnost mapiranja* i *totalnost učešća* objekata odnosnih klasa u vezi datog tipa. S obzirom na to da se ograničenja specifikuju jedinstveno, u

[37] Multiplikativnost je doslovan prevod termina *multiplicity* koji se koristi u UML specifikaciji. U literaturi se često koriste i drugi termini (npr. *kardinalnost*).

dijagramu klasa nema posebne grafičke notacije za totalnu participaciju kao u MOV dijagramu. Na dijagramu klasa sa sl. 2.34 multiplikativnosti specifikuju da svaki fakultet ima dekana, što znači da svi fakulteti učestvuju u vezama, tj. da je učešće objekata klase *Fakultet* u asocijaciji *rukovodi* totalno. S druge strane, ne mora svaki nastavnik da bude dekan, tj. učešće objekata klase *Nastavnik* u asocijaciji *rukovodi* je parcijalno. Analogno dolazimo do zaključka da je participacija objekata obje klase u asocijaciji *ima* totalna (svaki fakultet mora da ima bar jedan studijski program, a svaki studijski program pripada nekom fakultetu). Može se zaključiti da je totalnost učešća objekata *source* klase određena multiplikativnošću na *target* strani asocijacije i obrnuto, totalnost učešća objekata *target* klase je određena multiplikativnošću *source* strane asocijacije.

Karakter asocijacije (npr. generalizacija, agregacija, kompozicija, itd.) ilustruje se određenim grafičkim simbolom kojim se završava veza (npr. prazni trougao za generalizaciju, prazni romb za agregaciju, puni romb za kompoziciju, itd.), kao što je prikazano na sl. 2.36, a o čemu će više riječi biti u nastavku.

Asocijacije mogu da budu *jednosmjerne* (eng. *unidirectional*) ili *asimetrične* (eng. *asymmetrical*), te *dvosmjerne* (eng. *bidirectional*) ili *simetrične* (eng. *symmetrical*). Završeci simetrične asocijacije prikazuju se bez strelica (sl. 2.36a), dok se kod asimetrične asocijacije jedan završetak predstavlja strelicom (eng. *navigation arrow*) koja ukazuje na usmjerenost asocijacije (sl. 2.36b). Na nivou analize sistema, odnosno konceptualnog projektovanja, usmjerenost[38] asocijacije nema značaj, pa su sve asocijacije u ovom tekstu simetrične.

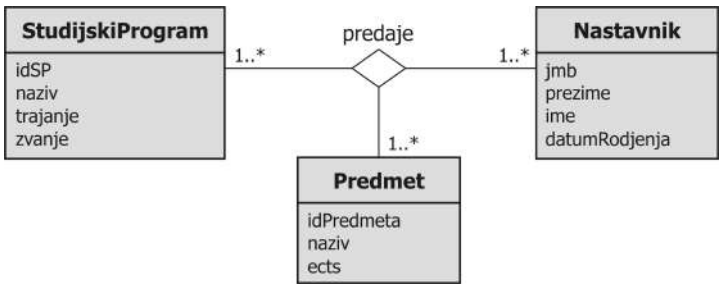



**Slika 2.36**: Ilustracije različitih asocijacija: a) simetrična asocijacija, b) asimetrična asocijacija, c) generalizacija, d) agregacija, e) kompozicija

[38] Usmjerenost asocijacije je bitan implementacioni detalj, jer usmjerenost od *source* ka *target* strani asocijacije pokazuje da objekti *source* klase pokazuju na objekte *target* klase, tj. da svaki objekat *source* klase ima atribut koji je pokazivač na *target* klasu.

Pod ***n-arnom* asocijacijom** (eng. *n-ary association*) podrazumijeva se veza tri ili više klasa. Slično MOV notaciji, *n-arna* asocijacija u dijagramu klasa prikazuje se kao prazni romb koji je punim linijima spojen sa odnosnim klasama, kao što je prikazano na sl. 2.37. U *n-arnim* asocijacijama uobičajena multiplikativnost krajeva je '$*$', tj. '0..$*$', jer obezbjeđuje veću fleksibilnost, odnosno postojanje veza u kojima ne učestvuju objekti svih $n$ klasa. Sva tri završetka ternarne asocijacije *predaje* na sl. 2.37 imaju multiplikativnost '1..$*$', što znači da ne može da postoji nijedna veza tipa *predaje* u kojoj ne učestvuju barem po jedan objekat svake od tri odnosne klase, tj. barem jedan studijski program, barem jedan nastavnik i barem jedan predmet. Dakle, ne može da postoji veza tipa *predaje* između nekog studijskog programa i nekog predmeta na tom studijskom programu, a da pri tome ne postoji bar jedan nastavnik koji predaje taj predmet na tom studijskom programu. Slično, ne može ni da postoji veza između nekog nastavnika i nekog predmeta, a da to nije na nekom studijskom programu, kao ni veza između nekog nastavnika i studijskog programa, a da ne postoji predmet koji taj nastavnik predaje u okviru tog studijskog programa. S druge strane, na svakom studijskom programu može da bude više predmeta, a svaki od tih predmeta može da predaje više nastavnika. Isto tako, svaki nastavnik može da predaje više predmeta na više studijskih programa.

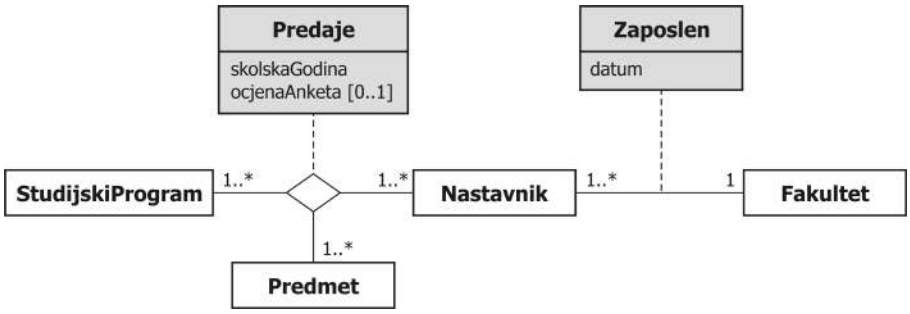
U *n-arnim* asocijacijama ne mogu da se izvlače zaključci o totalnosti učešća objekata odnosnih klasa. Tako se, za model na sl. 2.37, ne može tvrditi da li svaki nastavnik predaje neki predmet na nekom od studijskih programa ili postoje i nastavnici koji ne participiraju u vezama tipa *predaje*, tj. nastavnici koji ne predaju nijedan predmet.




**Slika 2.37**: Primjer *n-arne* (ternarna) asocijacije

Asocijacija koja ima *sopstvene atribute* modeluje se kao **vezna klasa** (eng. *association class*), koja se na dijagramu klasa prikazuje kao klasa vezana za odnosnu asocijaciju isprekidanom linijom (sl. 2.38). Tipično se takva asocijacija ne imenuje, jer vezna klasa ima odgovarajući naziv.

Obje asocijacije na sl. 2.38 imaju pridružene vezne klase kojima se modeluju atributi odnosnih asocijacija. Asocijaciji klasa *Nastavnik* i *Fakultet* pridružena je vezna klasa *Zaposlen*, koja ima jedan atribut (*datum*), koji predstavlja datum zaposlenja nastavnika na nekom fakultetu. Druga vezna klasa *Predaje* opisuje ternarnu asocijaciju sa sl. 2.38. Svaki angažman nekog nastavnika na nekom predmetu u okviru nekog studijskog programa vezan je za neku školsku godinu (*skolskaGodina*), a dodatni atribut svakog angažmana je ocjena koju je nastavnik dobio u studentskoj anketi (*ocjenaAnketa*).




**Slika 2.38**: Primjeri veznih klasa

### Agregacija i kompozicija

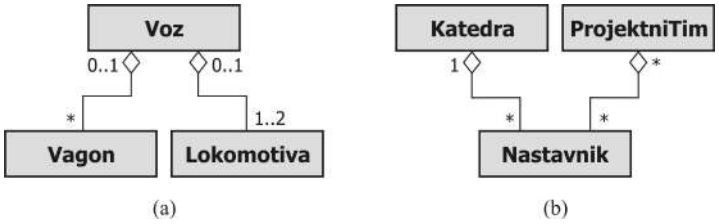
**Agregacija** je specifična asocijacija između dvije klase kojom se modeluje veza tipa *dio-cjelina* (eng. *part-whole*), pri čemu *dio* predstavlja entitet sa vlastitim identitetom, što mu omogućava da egzistira nezavisno od *cjeline* (*cjelina* se obično naziva i *agregat*). Ovakav koncept agregacije razlikuje se od koncepta agregacije u MOV (dio 2.1.6).

Na dijagramu klasa agregacija se prikazuje kao asocijacija čiji završetak na strani cjeline ima simbol praznog romba, kao što je ilustrovano na sl. 2.39. Prvi primjer (sl. 2.39a) modeluje situaciju koju često vidimo na željezničkoj stanici u trenutku kad se od voza otkači jedan ili više vagona ili jedna od dvije lokomotive, koji više ne pripadaju vozu ali i dalje postoje.

Nezavisnost od cjeline omogućava dijelovima da istovremeno participiraju u više različitih agregacija. Drugi primjer (sl. 2.39b) pokazuje da nastavnici, osim što pripadaju nekoj katedri na univerzitetu, mogu da pripadaju i nekim drugim organizacionim cjelinama, npr. nekom projektnom timu ili većem broju njih, itd.

Agregaciju tipično karakterišu glagolske fraze kao što su: *ima*, *pripada*, *sastoji se*, *sadrži*, itd. Zbog toga se često u dijagramu klasa naziv agregacije izostavlja. Treba imati u vidu da svaka veza dvije klase, koju karakteriše

neka od prethodno navedenih glagolskih fraza, ne mora nužno da se modeluje agregacijom. Na primjer, veza klasa *Profesor* i *Asistent* nije agregacija, nego asocijacija, jer iako profesor *ima* asistenta, ipak se ne radi o vezi tipa *dio-cjelina*.




**Slika 2.39**: Primjeri agregacije

**Kompozicija**[39] je stroži tip asocijacije u odnosu na agregaciju, u kojoj *dio* (koji se obično naziva *komponenta*) mora da pripada *cjelini* i ne može da egzistira nezavisno od *cjeline*. Na dijagramu klasa kompozicija se prikazuje kao asocijacija čiji završetak na strani *cjeline* ima simbol punog romba.

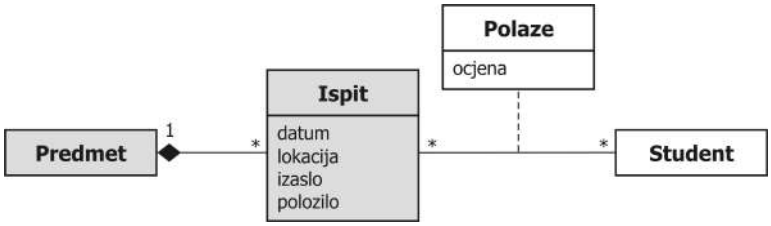
Kompozicija prikazana na sl. 2.40 modeluje veze računa i stavki koje mu pripadaju. Svaka stavka pripada isključivo jednom računu i egzistencijalno zavisi od tog računa, tj. ne može da postoji ako ne postoji i račun kojem pripada.


**Slika 2.40**: Primjer kompozicije

Kao što smo ranije napomenuli, UML nema posebnu notaciju za slabe entitetske skupove. S obzirom na to da kompozicija (iako se radi o drugačijem konceptu) omogućava modelovanje egzistencijalne zavisnosti, veza jakog i slabog entitetskog skupa, odnosno odgovarajućih klasa, može da se modeluje kao kompozicija, kao što je ilustrovano na sl. 2.41. Prikazani dijagram klasa modeluje situaciju opisanu u dijelu 2.1.4, pa nema potrebe za ponovnim opisom. Identifikujuća veza entitetskih tipova *Predmet* i *Ispit* modelovana je kompozicijom istoimenih klasa, pri čemu se slabi entitetski tip (*Ispit*) predstavlja komponentom.

Kompozicija omogućava modelovanje i kompozitnih (složenih) atributa. Naime, atributi neke klase ne mogu direktno da se označe kao kompozitni

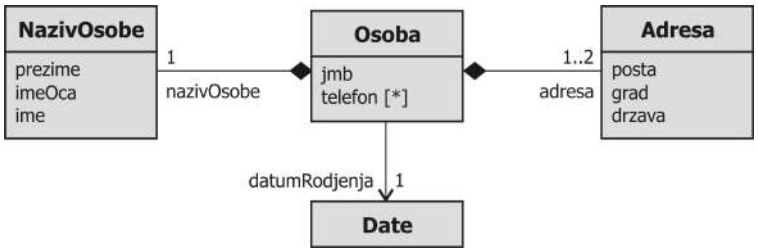
[39] U literaturi se koristi i termin *kompozitna agregacija* (eng. *composite aggregation*).




**Slika 2.41**: Modelovanje slabog entitetskog skupa kompozicijom

(kao što je to bilo moguće u MOV), nego se data klasa povezuje kompozicijom sa klasom koja reprezentuje tip kompozitnog atributa. Tako će klasa *Osoba* (sl. 2.42), osim atributa *jmb* i višeznačnog atributa *telefon*, imati još dva kompozitna atributa: *nazivOsobe* i *adresa*, pri čemu je *adresa* višeznačni kompozitni atribut.

Pripadnost bilo kojeg atributa nekoj klasi ne mora se prikazati na ranije opisani način, već se, slično modelovanju kompozitnih atributa, može modelovati usmjerenom asocijacijom date klase i klase koja predstavlja tip datog atributa. Ovo se ne odnosi samo na klase karakteristične za neki domen, nego čak i na standardne tipove (*Boolean*, *Date*, itd.). Tako će klasa *Osoba* (sl. 2.42), osim prethodno pobrojanih atributa, imati još i atribut *datumRodjenja*, koji je tipa *Date*. Ovaj primjer više navodimo kao ilustraciju semantičkog bogatstva UML-a, nego kao dobru praksu koju treba primjenjivati u konceptualnom projektovanju baze podataka.




**Slika 2.42**: Modelovanje kompozitnih atributa kompozicijom

### Kvalifikacija

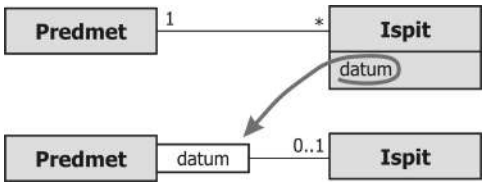
Koncept kvalifikacije ilustrovaćemo na primjeru slabog entitetskog tipa *Ispit*, koji je prikazan na sl. 2.41. Za svaki predmet postoji više ispita. U istom danu može da bude više ispita iz različitih predmeta, ali u jednom danu ne mogu da budu dva ispita iz istog predmeta. Pošto za svaki predmet ne postoje dva ispita sa istim datumom, atribut *datum* predstavlja

*diskriminator* objekata klase *Ispit*, tj. unutar grupe ispita za neki predmet, svaki ispit je jedinstveno identifikovan datumom održavanja. Tako se iz grupe ispita za neki predmet izdvaja samo jedan ispit, i to onaj koji je određen datumom održavanja ispita. Drugim riječima, diskriminator omogućava da se multiplikativnost '$*$' svede na '1'. Ovakav mehanizam za redukciju multiplikativnosti asocijacije u UML-u se naziva **kvalifikacija** (eng. *qualification*) i modeluje **kvalifikovanom asocijacijom** (eng. *qualified association*).

Da bi asocijacija bila kvalifikovana, treba da se specifikuje *kvalifikator* (eng. *qualifier*), tj. atribut (ili skup atributa) koji omogućava jedinstvenu identifikaciju objekta unutar grupe objekata jedne klase koji je u vezi sa konkretnim objektom druge klase. Kvalifikator se navodi unutar dodatnog pravougaonika na suprotnom kraju asocijacije u odnosu na klasu čija se multiplikativnost redukuje, kao što je ilustrovano na sl. 2.43. Nakon kvalifikovanja asocijacije, atribut koji predstavlja kvalifikator više se ne prikazuje kao atribut klase.

U primjeru na sl. 2.43 pretpostavljeno je da svakom predmetu odgovara '0..$*$' ispita, tj. mogu da postoje predmeti za koje još nije održan nijedan ispit. Zato se nakon kvalifikovanja asocijacije multiplikativnost redukuje na '0..1', tj. za neki datum može da postoji najviše jedan ispit iz nekog predmeta. Ovdje se multiplikativnost '0..$*$' (tj. '$*$') redukuje na '0..1'. Pored toga, moguće je redukovati i multiplikativnost '1..$*$', ali tada redukovana multiplikativnost iznosi tačno '1'. Tako bi se redukovala multiplikativnost asocijacije klasa *Fakultet* i *Nastavnik* sa sl. 2.38.

Iako je moguće specifikovati kvalifikator za oba završetka asocijacije, uobičajeno se kvalifikator specifikuje samo na jednoj strani asocijacije.




**Slika 2.43**: Redukcija multiplikativnosti primjenom kvalifikacije

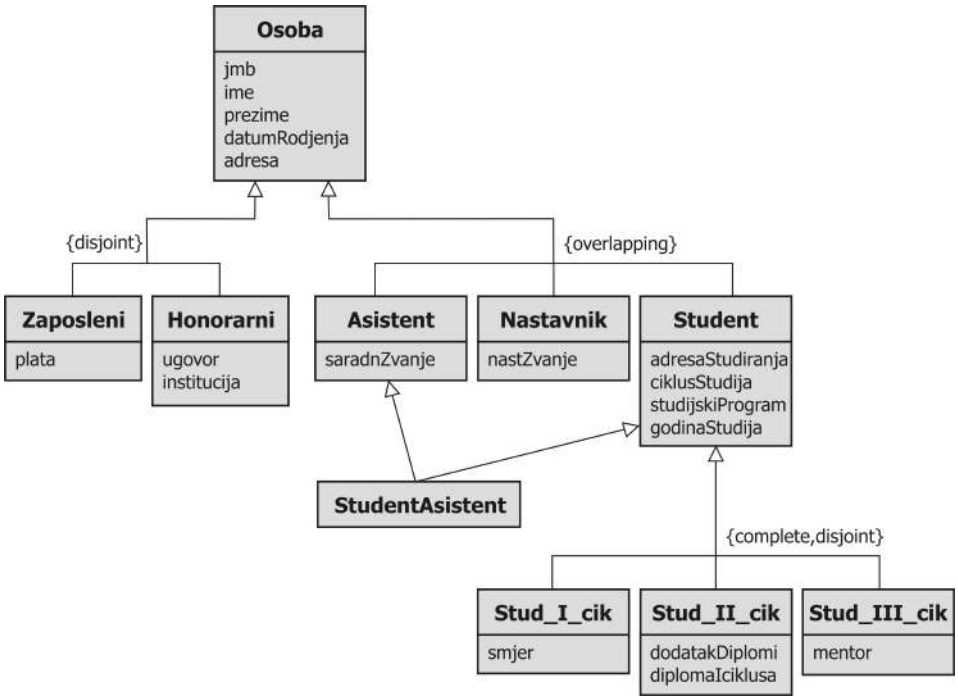
### Nasljeđivanje

**Nasljeđivanje** (eng. *inheritance*), kao jedan od najvažnijih O-O koncepata, podrazumijeva da neka klasa nasljeđuje (preuzima) sadržaj (članove) jedne ili više drugih klasa, uz mogućnost modifikacije preuzetih i dodavanja novih članova.

Posmatrano u kontekstu konceptualnog modelovanja baze podataka, nasljeđivanje omogućava modelovanje veza **generalizacije** i **specijalizacije**. S obzirom na to da su ove vrste veza detaljno opisane u dijelu 2.1.6, ovdje navodimo samo specifičnosti UML notacije.

Nasljeđivanje se modeluje strelicom sa praznim trougaonim vrhom, usmjerenom od *nasljednika* prema *pretku*[40], kao što je već ilustrovano na sl. 2.36c.

Na sl. 2.44 prikazan je dijagram klasa koji je ekvivalentan MOV dijagramu sa sl. 2.13 i koji ilustruje tipične slučajeve uvedene u dijelu 2.1.6. Klasa *Osoba* je osnovna klasa. Iz nje je izvedeno, odnosno nju specijalizuje pet potklasa (*Zaposleni*, *Honorarni*, *Asistent*, *Nastavnik*, *Student*) po osnovu dva različita kriterijuma. Za klasu *Osoba* se još kaže i da je *direktna osnovna klasa* tih potklasa.




**Slika 2.44**: Veze generalizacije i specijalizacije

[40] Uobičajeni termini za *nasljednika* su još i: *potklasa*, *podtip*, *izvedena* ili *specijalizovana klasa*. Uobičajeni termini za pretka su još i: *natklasa*, *nadtip*, *osnovna* ili *generalizovana klasa*.

Po osnovu prvog kriterijuma (radnopravni status), izvedene su potklase *Zaposleni* i *Honorarni*, dok su po osnovu drugog kriterijuma (pozicija i uloga u nastavnom procesu) izvedene preostale tri potklase. Specijalizacija po osnovu radnopravnog statusa je *disjunktna* (eng. *disjoint*), jer neka osoba ne može istovremeno da bude stalno zaposlena i honorarni saradnik, dok je specijalizacija po osnovu drugog kriterijuma *preklapajuća* (eng. *overlapping*). Ne postoji posebna notacija kojom se naglašava razlika između disjunktne i preklapajuće specijalizacije, pa se karakter specijalizacije specifikuje odgovarajućim ograničenjem, $\{disjoint\}$ ili $\{overlapping\}$.

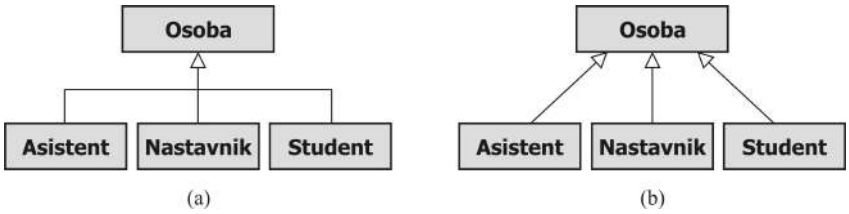
Očigledno je da osnovna klasa sadrži samo atribute koji karakterišu svaku osobu, dok potklase sadrže samo dodatne atribute koji predstavljaju specifičnost specijalizovanih, odnosno izvedenih klasa. Tako svaka zaposlena osoba na univerzitetu, osim atributa koji karakterišu osobu (*jmb*, *ime*, *prezime*, *datumRodjenja*, *adresa*), ima još dodatni atribut *plata* koji je član u odnosnoj izvedenoj klasi *Zaposleni*. Slično, svaki honorarni saradnik na univerzitetu ima (osim atributa koji karakterišu osobu) dodatne atribute *ugovor* i *institucija*, koji su modelovani kao članovi odnosne izvedene klase *Honorarni*. Po istom principu su modelovane i preostale tri klase (*Asistent*, *Nastavnik*, *Student*) izvedene iz klase *Osoba*.

Klasa *Student* dodatno je specijalizovana i predstavlja direktnu osnovnu klasu za izvedene klase *Stud_I_cik*, *Stud_II_cik* i *Stud_III_cik*. Za klasu *Osoba*, iz koje je izvedena klasa *Student*, kaže se da predstavlja *indirektnu osnovnu klasu* za potklase *Stud_I_cik*, *Stud_II_cik* i *Stud_III_cik*. Specijalizacija klase *Student* je disjunktna i *totalna* (eng. *complete*), jer svaki student *mora* biti student *samo jednog* ciklusa studija. Ne postoji ni posebna notacija za specifikaciju totalnosti specijalizacije, pa se koristi odgovarajuće ograničenje, $\{complete\}$ ili $\{incomplete\}$. U konkretnom slučaju, ograničenje specijalizacije je $\{complete, disjoint\}$[41].

Sve potklase po osnovu istog kriterijuma specijalizacije, u navedenom primjeru, modelovane su primjenom tzv. *dijeljenog* obrasca za prikaz nasljeđivanja (eng. *shared target style*). Ovaj način modelovanja nasljeđivanja dodatno je ilustrovan na sl. 2.45a). Drugi način za modelovanje nasljeđivanja (sl. 2.45b) podrazumijeva da se veze natklase sa potklasama prikazuju odvojeno (eng. *separate target style*). Iako po standardu nema semantičke

[41] Ovako definisana ograničenja omogućavaju da ista osoba bude student samo prvog, drugog ili trećeg ciklusa. Ako bi model trebao da omogući da student istovremeno može da bude student na različitim ciklusima studija (na različitim fakultetima), tada bi specijalizacija trebala da bude preklapajuća, tj. ograničenje bi trebalo da bude $\{complete, overlapping\}$.

razlike između ova dva stila, često se u praksi dijeljeni obrazac koristi za modelovanje specijalizacije po jednom kriterijumu, dok se drugi stil koristi za modelovanje specijalizacije po različitim kriterijima. Takav princip je primijenjen i u konkretnom primjeru na sl. 2.44.




**Slika 2.45**: Modelovanje nasljeđivanja primjenom *shared* (a) i *separate* (b) stila

Za klasu *StudentAsistent* (sl. 2.44) se kaže da je *višestruko izvedena*, jer ima dvije direktne osnovne klase, *Student* i *Asistent*. To znači da svaki objekat klase *StudentAsistent* nasljeđuje karakteristike obje roditeljske klase, što mu omogućava da se istovremeno ponaša i kao student i kao asistent[42].

Ako se koncept nasljeđivanja ne bi posmatrao samo u kontekstu konceptualnog modelovanja relacione baze podataka, tada bi od interesa bili i drugi O-O aspekti nasljeđivanja, kao što su *apstraktne* klase, *virtuelno* nasljeđivanje, itd. S obzirom na to da ovi koncepti nisu u uskoj vezi sa predmetom ove knjige, izostavljamo njihov detaljniji opis.

## 2.3 Sažetak

**Projektovanje baze podataka** je jedan od najznačajnijih segmenata razvoja sistema sa bazama podataka. Cilj je da se dođe do detaljne specifikacije sveukupne strukture baze podataka (modela) za odgovarajući DBMS, odnosno do **logičke šeme baze podataka** na implementacionom nivou. Zbog kompleksnosti problema potreban je sistematizovan pristup koji podrazumijeva korištenje odgovarajuće metodologije. Početna faza projektovanja baze podataka jeste **konceptualno modelovanje**, koje omogućava specifikaciju sveukupne strukture baze podataka na visokom nivou apstrakcije (**konceptualna šema**), za šta se najčešće koristi **Model**

[42] Striktno govoreći u duhu O-O paradigme, svaki objekat klase *StudentAsistent* sadrži jedan podobjekat klase *Student* (naslijeđen iz klase *Student*) i jedan podobjekat klase *Asistent* (naslijeđen iz klase *Asistent*).

**Objekti-Veze**. U MOV modelu, konceptualna šema se reprezentuje grafičkom notacijom poznatom pod nazivom **MOV dijagram**, koja nije standardizovana. Koncept modela objekti-veze je jednostavan i intuitivan, zasniva se na percepciji realnog svijeta, identifikaciji klasa sličnih entiteta i specifikaciji veza između njih putem odgovarajućih grafičkih koncepata. Model objekti-veze predstavlja **semantički model** podataka, jer omogućava ne samo reprezentaciju strukture podataka i veza između njih, nego i specifikaciju njihovog značenja kao i različitih ograničenja. Iz MOV modela je moguće, primjenom jednostavnih pravila, generisati logičku šemu za odnosni DBMS. Osnovni koncepti MOV dijagrama su entiteti i veze.

**Entitet** (objekat) je nešto što postoji u realnom svijetu i što se razlikuje od drugih postojećih objekata. Svaki entitet realnog svijeta karakterišu određena svojstva koja se u MOV modelu reprezentuju **atributima**. Određeno svojstvo entiteta se iskazuje odgovarajućom kvantitativnom vrijednošću odnosnog atributa. Skup dopuštenih vrijednosti nekog atributa, predstavlja njegov **domen**. *Null* vrijednost je specijalna vrijednost, koja može biti član bilo kojeg domena, i koja označava da vrijednost atributa nije poznata, ili da konkretna vrijednost nije primjenljiva za odnosni entitet. U MOV modelu mogu postojati različiti tipovi atributa: prosti ili složeni, jednoznačni ili višeznačni, bazni ili izvedeni.

Određene entitete realnog svijeta karakterišu ista svojstva, odnosno ti entiteti se mogu reprezentovati istim skupom atributa. Za takve entitete kažemo da su slični, odnosno da su istog tipa. **Tip entiteta** je određen imenom tipa i skupom atributa koji reprezentuju svojstva entiteta tog tipa. Tip entiteta opisuje **šemu**, odnosno strukturu i karakteristike entiteta koji pripadaju istom entitetskom skupu. **Entitetski skup** je skup entiteta istog tipa u bazi podataka, u nekom određenom trenutku.

**Veza** je asocijacija između određenog broja entiteta. **Tip veze** definiše karakteristike i mogući skup veza između entiteta odnosnih entitetskih tipova. **Vezni skup** je skup veza istog tipa. **Uloge** entiteta u vezi su najčešće implicitno određene. U (rekurzivnim) vezama između entiteta istog entitetskog skupa potrebno je eksplicitno naznačiti ulogu pojedinih entiteta u vezi, putem odgovarajućeg naziva uloge. Veze mogu imati svoje vlastite, **opisne** atribute.

**Stepen tipa veze** (i odgovarajućeg veznog skupa) određen je brojem entitetskih tipova (i analogno entitetskih skupova) koji participiraju u tipu veze (veznom skupu). Najčešće su **binarne veze**, koje povezuju dva entitetska tipa (entitetska skupa). Veze koje povezuju više od dva entiteta jesu *n-arne* veze.

Tip veze ne samo da definiše koji tipovi entiteta učestvuju u vezama, nego specifikuje i dodatna ograničenja na učešće entiteta u vezama. **Kardinalnosti mapiranja** određenog tipa veze, specifikuje maksimalan broj veza u kojima neki entitet odnosnog entitetskog tipa (skupa) može da učestvuje. Kardinalnost mapiranja za binarni tip veze može biti *jedan:jedan*, *jedan:više*, *više:jedan* i *više:više*. **Ograničenje učešća entiteta u vezi** specifikuje, za određeni tip veze, mininalni broj veza u kojima neki entitet odnosnog entitetskog tipa (skupa) mora da učestvuje. Učešće entiteta u vezi može biti *totalno* i *parcijalno*.

Značajna forma ograničenja u MOV modelu su ključevi. **Superključ** je skup atributa entitetskog tipa na kojem svaki entitet entitetskog skupa (tog tipa) ima jedinstvenu vrijednost. **Kandidatski ključ** je minimalni superključ, tj. superključ koji nema netrivijalni podskup atributa koji je takođe superključ. Neki entitetski tip može sadržavati više kandidatskih ključeva. Za entitetski tip (skup) jedan od kandidatskih ključeva se selektuje kao **primarni ključ**, odnosno kao osnovni ključ za jedinstvenu identifikaciju entiteta u entitetskom skupu. Slično, superključ tipa veze je skup atributa koji jedinstveno identifikuje svaku vezu u veznom skupu odnosnog tipa. Kandidatski/primarni ključ tipa veze je minimalni superključ tipa veze.

Neki entitetski tipovi nemaju atribute koji čine primarni ključ. Takvi entitetski tipovi se nazivaju **slabi entitetski tipovi**, a entitetski skupovi tog tipa **slabi entitetski skupovi**. Entitetski tip koji ima atribute koji čine primarni ključ se naziva jaki entitetski tip (i analogno tome, odnosni entitetski skupovi su **jaki entitetski skupovi**). Tip veze koji povezuje slabi i jaki entitetski tip naziva se **identifikujući tip veze**.

Osnovni MOV model nije dovoljan za pogodno predstavljanje specifičnih svojstava entiteta i odnosa između entiteta, pa je MOV proširen dodatnim konceptima.

U nekim entitetskim skupovima mogu se uočiti podgrupe entiteteta koje se razlikuju po nekim specifičnim svojstvima u odnosu na druge entitete iz entitetskog skupa. Proces identifikacije i reprezentacije ovih podgrupa u okviru nekog entitetskog skupa, naziva se **specijalizacija**. Podgrupe entiteta sa specifičnim svojstvima unutar neke grupe modeluju se **entitetskim tipovima nižeg nivoa** koji su sa generalnijim **entitetskim tipom višeg nivoa** povezani ISA vezom. ISA veza reprezentuje činjenice da su (specijalizovani) entiteti entitetskog tipa nižeg nivoa ujedno i entiteti višeg entitetskog tipa. Za entitetske tipove višeg nivoa koristi se i termin **superklasa**, a za entitetske tipove nižeg nivoa termin **potklasa**. Entitetski

tipovi nižeg nivoa (potklase) nasljeđuju sve atribute entitetskog tipa višeg nivoa (superklase) sa kojim su povezani ISA vezom, i imaju, dodatno, svoje specifične atribute. Analogno nasljeđivanju atributa, nasljeđuju se i tipovi veza u kojima entitetski tip višeg nivoa (superklasa) učestvuje.

Do identifikacije generalnijih entitetskih tipova višeg nivoa i entitetskih tipova nižeg nivoa može se doći i **generalizacijom**, obrnutim procesom u odnosu na specijalizaciju. Ukoliko se u toku analize sistema prvo identifikuju neki entitetski tipovi (skupovi) a zatim utvrdi njihova sličnost, onda se može izvršiti generalizacija i definisanje entitetskog tipa (skupa) višeg nivoa (superklase), čiji bi atributi bili zajednički atributi polaznih entitetskih tipova. Za određenu specijalizaciju/generalizaciju se mogu specifikovati slijedeća ograničenja: uslov članstva entiteta entitetskog skupa višeg nivoa u entitetskom skupu nižeg nivoa (*definisano uslovom* ili *korisnički definisano*), ograničenje na participaciju entiteta višeg entitetskog skupa u entitetskim skupovima nižeg nivoa (*disjunktno* i *preklapajuće*), i ograničenje na kompletnost učešća entiteta entitetskog skupa višeg nivoa u entitetskim skupovima nižeg nivoa (*totalno* i *parcijalno*).

Koncept **unije** ili **kategorije** omogućuje modelovanje veza tipa superklasa/potklasa, pri čemu veza povezuje više superklasa različitog tipa sa jednom potklasom. Potklasa (kategorija/unija) može da sadrži entitete različitih tipova, koji su elementi superklasa sa kojima je kategorija/unija povezana, odnosno kategorija(unija) je podskup unije entiteta različitih tipova, članova superklasa.

**Agregacija** je apstrakcija kojom se veze tretiraju kao entiteti višeg nivoa, i koje, kao takvi apstraktni entiteti višeg nivoa, mogu učestvovati u drugim vezama. Modelovanje unija i agregacija nije standardno uključeno u svim alatima i MOV modelima.

Ne postoji metodologija za konceptualno modelovanje bazirana na MOV, koja propisuje precizne korake koji bi uvijek vodili ka jedinstvenom i jednoznačnom modelu nekog realnog sistema. U procesu projektovanja projektanti se susreću sa različitim mogućnostima i dilemama, i specifične percepcije realnog sistema mogu rezultovati različitim zaključcima i identifikacijom entitetskih i veznih tipova na različite načine. Najčešće **projektne dileme** uključuju izbor za reprezentovanje određenih činjenica konceptom atributa ili entitetskog tipa, konceptom atributa ili slabog entitetskog tipa, izbor između entitetskog ili veznog tipa, binarnih ili *n-arnih* tipova veza itd. Sam proces izgradnje modela je gotovo uvijek iterativan, pri čemu se prvo kreira početni model, a zatim se u iterativnom procesu u svakoj iteraciji vrši analiza, otklanjaju nedostaci, rješavaju projektne dileme i poboljšava model.

**UML dijagram klasa** predstavlja notaciju koja se često koristi u projektovanju baza podataka, iako standardna notacija nije baš u potpunosti prilagođena za tu namjenu. Dijagram klasa je strukturni UML dijagram koji omogućava modelovanje klasa i njihovih međusobnih veza.

**Klasa** je deskriptor za skup entiteta (objekata) sa sličnim svojstvima, ponašanjem i vezama sa drugim entitetima. Klase imaju dvije vrste članova: **atribute** koji reprezentuju svojstva objekata i **operacije** koje su definisane nad objektima. Klase bez operacija su analogne entitetskim tipovima u MOV.

UML omogućava modelovanje osnovnih, izvedenih i višeznačnih atributa u klasi. Atributima je moguće definisati podrazumijevanu vrijednost, kao i ograničenja u pogledu dozvoljenih vrijednosti, te jedinstvenosti i uređenosti skupa vrijednosti ako je atribut višeznačan. Višeznačnost atributa se definiše pomoću multiplikativnosti koja specifikuje minimalan i maksimalan broj vrijednosti višeznačnog atributa. Ako je minimalna multiplikativnost jednaka nuli, tada je dozvoljeno da atribut nema vrijednost, tj. da je vrijednost *null*.

Standardni UML nema posebnu notaciju za atribute koji čine primarni ključ. Zato neki alati koriste specijalizovanu notaciju koja ima te mogućnosti. Ne postoji ni standardna UML notacija za neke druge specifične koncepte koji postoje u MOV, kao što su slabi entitetski skupovi, složeni (kompozitni) atributi, itd. Međutim, postoje semantički ekvivalentni koncepti.

**Asocijacija** je veza između dvije klase i predstavlja pandan binarnom tipu veze između dva entitetska tipa u MOV. Veza između tri ili više klasa modeluje se ***n-arnom* asocijacijom**. *Završetak* asocijacije tipično prikazuje *ulogu* koju objekti date klase imaju u datom tipu veze, *multiplikativnost* i *karakter asocijacije*. *Multiplikativnost* krajeva asocijacije služi za specifikaciju ograničenja učešća objekata odnosnih klasa u datom tipu veze, a specifikuje minimalan i maksimalan broj objekata odnosne klase koji učestvuju u vezi datog tipa sa jednim objektom druge klase, i obrnuto. Multiplikativnosti krajeva asocijacije istovremeno specifikuju i *totalnost participacije* objekata odnosnih klasa u vezi datog tipa. S obzirom na to da se ograničenja specifikuju jedinstveno, u dijagramu klasa ne postoji posebna grafička notacija za totalnu participaciju kao u MOV dijagramu. Karakter asocijacije (npr. agregacija, kompozicija, generalizacija itd.) ilustruje se određenim grafičkim simbolom kojim završava veza. Asocijacija koja ima *sopstvene atribute* modeluje se **veznom klasom**, koja se prikazuje kao klasa vezana za odnosnu asocijaciju. Asocijacije mogu biti *jednosmjerne* ili *asimetrične* te *dvosmjerne* ili *simetrične*. U konceptualnom modelovanju relacione baze podataka tipično se koriste simetrične asocijacije.

**Agregacija** je specifična asocijacija dvije klase kojom se modeluje veza tipa *dio-cjelina*, pri čemu *dio* predstavlja entitet sa vlastitim identitetom, što mu omogućava da egzistira nezavisno od *cjeline*. Ovakav koncept agregacije razlikuje se od koncepta agregacije u MOV. Nezavisnost od cjeline omogućava dijelovima da istovremeno participiraju u više različitih agregacija.

**Kompozicija** je stroži tip asocijacije u odnosu na agregaciju, u kojoj *dio* mora da pripada *cjelini* i ne može da egzistira nezavisno od *cjeline*. Kompozicija omogućava modelovanje egzistencijalne zavisnosti, odnosno veze jakog i slabog entitetskog tipa. Kompozicija omogućava modelovanje i kompozitnih (složenih) atributa.

**Nasljeđivanje** omogućava modelovanje veza **generalizacije** i **specijalizacije**. Ne postoji posebna notacija kojom se naglašava razlika između *disjunktne* i *preklapajuće*, odnosno *totalne* i *parcijalne* specijalizacije, pa se karakter specijalizacije specifikuje odgovarajućim ograničenjem.

## Pitanja za ponavljanje i zadaci

2.1 Opišite i definišite slijedeće pojmove:

- Model Objekti-Veze
- Semantički model
- Entiteti
- Atributi
- Entitetski tip i entitetski skup
- Veze
- Vezni tip i vezni skup
- Opisni atributi veze/veznog tipa
- Stepen tipa veze
- Kardinalnost mapiranja
- Ograničenje učešća entiteta u vezi
- Ključevi
- Superključ
- Kandidatski ključ
- Primarni ključ
- MOV dijagrami
- Slabi i jaki entitetski tip
- Specijalizacija/generalizacija
- Ograničenja vezana za specijalizaciju/generalizaciju
- Unija/kategorija
- Agregacija
- UML
  - Arhitektura
  - Standardni dijagrami
  - Dijagram klasa
  - Klase i asocijacije
  - Agregacija
  - Kompozicija
  - Kvalifikacija
  - Nasljeđivanje

2.2 Objasnite distinkciju pojmova entitetski tip i entitetski skup, te pojmova vezni tip i vezni skup.

2.3 Koliko može biti primarnih ključeva, kandidatskih ključeva i superključeva za neki entitetski tip? Objasnite.

2.4 Diskutujte mogućnost modelovanja činjenica o polaganju ispita kao višeznačnog atributa veze *UPISAO* sa sl. 2.29.

2.5 Dopunite model na sl. 2.29 tako da model reprezentuje i katedre kao organizacione jedinice fakulteta, informacije o šefovima pojedinih katedri (iz reda nastavnika u stalnom radnom odnosu), te specifikaciju veze svakog nastavnika (iz reda nastavnika u stalnom radnom odnosu) sa šefom katedre.

2.6 Ako se procesom generalizacije kreira natklasa, kakve su karakteristike ograničenja totalnosti i preklapanja učešća entiteta superklase u potklasama (totalnost, disjunktnost)?

2.7 Ukoliko je kategorija totalna, da li se ona može modelovati i kao generalizacija i zašto? Da li se parcijalna kategorija može modelovati generalizacijom i zašto?

2.8 Pretpostavimo da imamo entitetske tipove *PUTNIČKO_VOZILO* i *KAMION*. Pretpostavimo da imamo kategoriju *REGISTROVANO_VOZILO* čije su superklase *PUTNIČKO_VOZILO* i *KAMION*. Objasniti razliku između kategorije *REGISTROVANO_VOZILO* i entitetskog tipa *VOZILO* koje predstavlja generalizaciju svih vozila.

2.9 Da li bi se kvaternarna veza na sl. 2.15 mogla zamijeniti binarnim vezama entitetskih tipova *KONTROLOR* i *NASTAVNIK* i *KONTROLOR* i *PREDMET*?

2.10 Da li bi povezivanje entitetskog tipa *KONTROLOR* sa entitetskim tipovima *NASTAVNIK*, *PREDMET* i *FAKULTET* direktno preko veze *PREDAJE* bilo ekvivalentno MOV modelu na sl. 2.15? Zašto?

2.11 Objasnite detaljno zašto se ne mogu objediniti tipovi veza/vezni skupovi *KONTROLIŠE* i *PREDAJE* u jedan vezni tip/vezni skup.

2.12 Šta je primarni ključ veznog tipa *KONTROLIŠE*, ako:

a) Za kontrolu predavanja nastavnika na nekom predmetu na nekom fakultetu može da bude određen samo jedan kontrolor?

b) Za kontrolu predavanja nastavnika na nekom predmetu na nekom fakultetu može da bude određeno više kontrolora?

2.13 Kreirajte UML dijagram klasa koji je ekvivalentan MOV dijagramu sa sl. 2.29. Diskutujte razlike.

2.14 Specifikujte informacione zahtjeve za informacioni sistem biblioteke i kreirajte konceptualni model podataka koristeći:

a) MOV notaciju.

b) UML dijagram klasa.

2.15 Projektovati konceptualni model podataka koji zadovoljava informacione potrebe hotela. Ciljna baza podataka treba da sadrži podatke o sobama, gostima, rezervacijama, računima, angažovanim licima itd.

2.16 Prilagoditi model iz prethodnog zadatka tako da zadovoljava informacione potrebe hotelskog preduzeća u čijem se vlasništvu nalazi lanac hotela distribuiranih u različitim državama.

2.17 Predmet projektnog zadatka je baza podataka auto-škole. Projektovati konceptualni model podataka tako da ciljna baza sadrži podatke o instruktorima, vozilima, kandidatima, upisu i obuci kandidata, terminima za teoretsku i praktičnu nastavu, terminima i rezultatima polaganja ispita, finansijskim obavezama i dr.

2.18 Projektovati konceptualni model podataka apoteke. Ciljna baza podataka treba da sadrži podatke o dobavljačima, nabavkama lijekova i drugih medicinskih stvari (aparati, preparati, materijali i sl.), stanju robe i materijala, izdavanjima na recept i bez recepta, angažovanim licima, finansijskim efektima, itd.

2.19 Prilagoditi model iz prethodnog zadatka tako da zadovoljava informacione potrebe lanca apoteka na teritoriji države.

2.20 Predmet projektnog zadatka je baza podataka primarne zdravstvene zaštite organizovane na principu porodičnog ljekara na teritoriji opštine. Projektovati konceptualni model podataka tako da ciljna baza sadrži podatke o: teritorijalnoj organizaciji i mreži zdravstvenih ambulanti te zaposlenim licima (ljekarima, sestrama, administrativnim licima i pomoćnim radnicima); stanovništvu i pripadnosti ambulanti (ljekaru); pacijentima i istoriji medicinskih tretmana (termini, dijagnoze, upute, recepti, intervencije...); finansijama (obračunate usluge, participacija...).

2.21 Predmet projektnog zadatka je baza podataka neophodna za organizaciju finalne utakmice na Svjetskom fudbalskom prvenstvu. Projektovati konceptualni model podataka tako da ciljna baza sadrži podatke o ekipama, igračima i sudijama, kao i karakteristične statističke podatke (golovi, strijelci, prekršaji, žuti/crveni kartoni, korneri, "ofsajdi", udarci u okvir i van okvira gola).

2.22 Prilagoditi model iz prethodnog zadatka tako da zadovoljava informacione potrebe organizatora fudbalskog turnira. Ciljna baza treba da sadrži podatke o: ekipama i takmičarima, sudijama i drugim angažovanim licima (obezbjeđenje, medicinsko osoblje, ...), rasporedu mečeva, odigranim mečevima, rezultatima, konačnom plasmanu, osvojenim nagradama (materijalnim, finansijskim), prodatim ulaznicama, isplatama i dr.

2.23 Predmet projektnog zadatka je baza podataka multipleks-bioskopa. Projektovati konceptualni model podataka tako da ciljna baza sadrži podatke o: filmovima i distributerima filmova, bioskopskim salama i planu projekcija, prodatim kartama i ostvarenim prihodima, zaposlenim licima, itd.

2.24 Projektovati konceptualni model podataka koji zadovoljava informacione potrebe sistema za rezervacije karata u avio sabraćaju. Sistem treba da omogući komunikaciju sa različitim vrstama klijenata (putnici koji direktno mogu da rezervišu karte i otkazuju rezervacije te agencije koje vrše i otkazuju rezervacije u ime trećih lica), avio prevoznicima (dostavljaju podatke o detaljima letova, a preuzimaju podatke o rezervacijama) i aerodromima (dostavljaju podatke o vanrednim situacijama, npr. otkazivanje leta, a preuzimaju podatke o rezervacijama).

2.31 Projektovati konceptualni model podataka koji zadovoljava informacione potrebe turističke agencije. Ciljna baza podataka treba da omogući manipulaciju podacima vezanim za klijente (korisnici usluga) i provajdere (oni koji pružaju usluge), zaposlene, ponudu turističkih aranžmana, rezervacije, finansijske efekte, itd.

2.32 Predmet projektnog zadatka je baza podataka neophodna za sprovođenje upisa osnovnih studija na univerzitetu. Projektovati konceptualni model podataka tako da ciljna baza sadrži podatke o: organizacionoj strukturi univerziteta, studijskim programima i kriterijumima za upis (oblasti – predmeti koji se polažu i prag bodova, drugi kriterijumi kao što su prosječna ocjena, ocjena iz predmeta, način finansiranja – budžet/samofinansiranje), kandidatima, prijemnim ispitima, upisnim rokovima, finansijskim efektima, angažovanim licima, itd.

2.24 Projektovati konceptualni model podataka koji zadovoljava informacione potrebe opštinske izborne komisije. Ciljna baza treba da sadrži podatke o: političkim partijama, koalicijama političkih partija i njihovim kandidatima (izbornim listama) kao i nezavisnim kandidatima, te biračkim mjestima, biračima i rezultatima izbora na biračkim mjestima po različitim nivoima vlasti.

2.25 Predmet projektnog zadatka je baza podataka studentskog doma. Projektovati konceptualni model podataka tako da ciljna baza sadrži podatke o smještajnim kapacitetima, konkursima, stanarima, finansijskim obavezama i dr.

2.26 Prilagoditi model iz prethodnog zadatka tako da zadovoljava informacione potrebe univerzitetskog studentskog centra koji u svom sastavu ima više studentskih domova.

2.27 Projektovati konceptualni model podataka koji zadovoljava informacione potrebe autobuske stanice. Ciljna baza treba da sadrži podatke o: redu vožnje i prevoznicima, prodanim kartama i finansijskim efektima, rezervacijama, angažovanim licima, itd.

2.28 Prilagoditi model iz prethodnog zadatka tako da zadovoljava informacione potrebe preduzeća koje posjeduje lanac autobuskih stanica na teritoriji države.

– § –