Usporedba IDEF1X i MIRIS metoda za modeliranje baza podataka kroz praktični primjer

Miljanić, Korina

Sveučilište u Rijeci – Odjel za informatiku

Preddiplomski jednopredmetni studij informatike

Korina Miljanić

Usporedba IDEF1X i MIRIS metoda za modeliranje baza podataka kroz praktični primjer

Završni rad

Mentor: prof. dr. sc. Patrizia Poščić

Rijeka, rujan 2021.

ODJEL ZA INFORMATIKU
SVEUČILIŠTE U RIJECI

Rijeka, 10. lipnja 2021.

Zadatak za završni rad

Pristupnik: Korina Miljanić

Naziv završnog rada: Usporedba IDEF1X i MIRIS metoda za modeliranje baza podataka kroz praktični primjer

Naziv završnog rada na eng. jeziku: Comparison of IDEF1X and MIRIS data modeling methods with an example

Sadržaj zadatka:

U ovom završnom radu potrebno je opisati metode koje se u metodologijama MIRIS i IDEF koriste za modeliranje baze podataka. Naglasak prilikom njihove obrade i usporedbe bit će na metodi entiteta i veza (MIRIS metodologija) i metodi IDEF1X (IDEF metodologija). Cilj rada je objasniti osnovne koncepte u odabranim metodama. Kroz vlastiti praktični primjer potrebno je usporediti metode i dati svoj osvrt.

Mentor

Prof. dr.sc. Patrizia Poščić

Voditelj za završne radove

Dr. sc. Miran Pobar

Zadatak preuzet: 10. lipnja 2021.

(potpis pristupnika)

Sadržaj

Sažetak

Cilj ovog rada je opisati metodologiju MIRIS i metodologiju IDEF koje se koriste za modeliranje baza podataka. Prilikom opisivanja metodologije MIRIS naglasak je na njenoj metodi entiteti – veze, dok je naglasak prilikom obrade IDEF metodologije na IDEF1X metodi. U nastavku su detaljnije opisani koncepti metode entiteti - veze i metode IDEF1X. Usporedbom koncepata ove dvije metode istaknule su se njihove sličnosti. Zatim je prikazan model podataka modeliran metodom entiteti – veze, a potom i IDEF1X metodom. Primjeri su detaljno opisani i uspoređeni.

Ključne riječi: MIRIS, IDEF, metoda entiteti – veze, IDEF1X, modeliranje podataka

Uvod

Kako bi sustav normalno funkcionirao i upravljao procesima na ispravan način mora prikupiti i obraditi golemu količinu informacija. Čovjek samostalno ne može obraditi toliku količinu informacija, te se iz tog razloga koristi računalom koje mu omogućuje lakše skladištenje i obradu informacija o sustavu. Za modeliranje nekog sustava nužne su nam informacije, a njih najlakše prenosimo putem podataka koji mogu biti alfanumerički, grafički, zvukovni i dr. Međusobno povezani segmenti koji teže pribavljanju i prenošenju informacija čine informacijski sustav. Kako bi informacijski sustav automatizirali, s vremenom je stvoreno mnoštvo metoda temeljenih na različitim teorijama tijekom razvoja metodologija za razvoj i izgradnju informacijskih sustava. U ovom radu upoznat ćemo metodologiju MIRIS i metodologiju IDEF, u kojima ćemo detaljnije obraditi metodu entiteti-veze metodologije MIRIS i metodu IDEF1X metodologije IDEF. Usporedit ćemo njihove koncepte i kroz praktični primjer usporediti ove dvije metode.

MIRIS

U ovom dijelu bavit ćemo se jednom od specijaliziranih metodologija, a to je Metodologija za Razvoj Informacijskog Sustava, u daljnjem tekstu MIRIS. Njezin cilj je projektirati i izgraditi informacijski sustav na temelju zadanih uputa i metoda. Njeno osmišljavanje i kreiranje trajalo je 11 godina, od 1984-1995 kada je i objavljena. Ona također propisuje faze razvoja i aktivnosti pojedine faze sve do detaljnijih razina IS-a. MIRIS se sastoji od dvije skupine faza životnog ciklusa prilikom razvoja informacijskog sustava. Prva faza razvoja je logičko oblikovanje gdje se projektira IS, a druga faza je fizičko oblikovanje gdje se IS izgrađuje . Obje faze sastoje se od još tri faze koje se u daljnjem razvoju dijele na aktivnosti. (Pavlić, 2009).

Logičko oblikovanje dijeli se na sljedeće faze:

1. Strateško planiranje informacijskog sustava
2. Glavni projekt
3. Izvedbeni projekt

Izrada IS-a počinje kreiranjem strateškog plana u kojem se mora definirati i obučiti tim, odraditi dekompozicija procesa te prikupiti popis dokumentacije i kretanja kroz sustav. Moraju se odrediti podsustavi, njihove veze i prioriteti. Nakon definiranja cjelovite infrastrukture i pregleda potrebnih resursa, može se krenuti na planiranje glavnog projekta.

Glavni projekt započinje izradom projektnog zadatka. Kroz dijagram toka podataka, u daljnjem tekstu DTP , vrši se intervjuiranje, raščlanjivanje i modeliranje procesa. Nadalje je potrebno analizirati procese, uvidjeti probleme, te moguća poboljšanja sustava. Nakon čega slijedi opisivanje podataka, izrada plana izvedbenog projekta, te definiranje modela resursa potrebnih za glavni projekt.

Posljednja faza logičkog oblikovanja je izvedbeni projekt, u kojem se kroz dijagram entiteta i veza, nadalje DEV, obavlja intervjuiranje, apstrakcija i modeliranje podataka. Zatim se dobiveni podaci trebaju prevesti u shemu baze podataka (Relacijski model), te nakon definiranja arhitekture programskog proizvoda slijedi projektiranje operacija nad shemom baze podataka (Pavlić, 2011).

Fizičko oblikovanje dijeli se na sljedeće faze:

1. Proizvodnja softvera
2. Uvođenje
3. Primjena i održavanje

Sama izgradnja IS-a u fazi fizičkog oblikovanja kreće planiranjem proizvodnje softvera. Planiranje slijedi oblikovanje baze podataka i razvoj programskog proizvoda kojeg je moguće izraditi od početka ili doraditi već postojećeg. Potom se razvijeni programski proizvod ispituje u testnoj okolini. Nakon uspješno odrađenog ispitivanja softvera u testnoj okolini, testiranje se

nastavlja u radnoj okolini, te se nakon njega softver prezentira korisniku i on ga testira. Izrađuje se popis primjedbi korisnika, ažurira se softver te se faza testiranja završava.

Nakon same proizvodnje, softver je potrebno uvesti u korisnikovu radnu okolinu i upoznati ga sa značajkama novog softvera, stoga se odvija faza uvođenja prilikom koje se instalira gotov softver kod korisnika, izrađuju se upute za korištenje, prezentira se gotov softver i obučavaju se korisnici za rad na njemu, vrši se posljednje testiranje i nova aplikacija se počinje svakodnevno koristiti.

Posljednja faza razvoja IS-a je primjena i održavanje istoga u kojoj se podešava novi aplikacijski sustav, izrađuje izvješće o procjeni novog projekta i raspodjeljuje odgovornost korisnika i programera. Povremeno se održavaju i unaprjeđuju segmenti informacijskog sustava kako bi se udovoljilo zahtjevima korisnika. (Pavlić, 2011).

Entiteti i veze

Metoda entiteti – veze poznatija pod imenom **model entiteti – veze** (engl. Entity – relationship model ili ER) jedna je od prvih semantički bogatijih metoda za modeliranje podataka prvi puta objavljena u Chenovu članku 1976. (Chen 1976.). Mnoštvo knjiga o razvoju informacijskih sustava i baza podataka sadrži opisanu metodu entiteti – veze, te je upravo ova metoda prihvaćena gotovo na svim svjetskim sveučilištima, dok se u praksi prilikom oblikovanja baze podataka većinom ona i koristi.

Američki znanstvenik i profesor informatike na fakultetu Louisiana State University Dr. ***Peter Pin – Shan Chen*** zaslužan je za razvitak metode entiteti – veze. Nakon studija elektrotehnike na Tajvanu kao stipendista odlazi na Harvard gdje upisuje postdiplomski studij računarstva. 1975. godine na internacionalnoj konferenciji u Las Vegasu „1st International Conference on Very Large Database“ predstavlja svoj rad o metodi entiteti – veze. Njegov znanstveni rad i razvoj ER modela ističe se kao jedan od najutjecajnijih radova u povijesti informatike.

Nakon predstavljanja modela, utemeljitelj relacijske metode, Codd oštro reagira na njegov rad pronalazeći mnoštvo zamjeraka. Codd 1979. objavljuje novu verziju relacijskog modela gdje su preuzeti neki koncepti iz ER modela. Tijekom devedesetih godina Chen je nekoliko puta uz Codda glavni govornik na mnogim simpozijima, što se može shvatiti kao prihvaćenost njegovog ER modela. Od tada, Chen je često glavni govornik na međunarodnim konferencijama. (Pavlić, 2011).

Metoda entiteti – veze su međusobno povezane skupine podataka promatranog sustava prikazanih grafički. Metoda se služi konceptima bliskim ljudima i stoga se smatra izrazito semantički bogatom metodom za modeliranje podataka. Metoda entiteti – veze i model dijagrama entiteti – veze izgrađen pomoću nje koristi se za razvoj najmanje dva dijela informacijskog sustava: shema baze podataka i arhitektura programskog proizvoda.

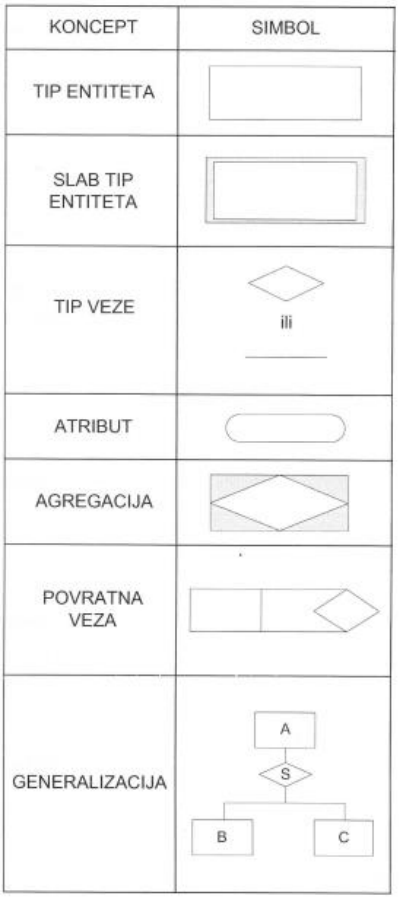
Struktura modela entiteti – veze izgrađena je od različitih osnovnih koncepata:

- Entitet i tip entiteta
- Veza i tip veze
- Atribut tipa entiteta
- Slab tip entiteta i specijalni tipovi veza
- Agregirani tip entiteta
- Povratni tip veze
- Generalizacijski tip veze

Metodom EV uporabom grafičkih simbola gradi se model podataka. DEV ili dijagram entiteta i veza (*Entity – Relationship Diagram*, ERD) nazivamo grafički prikaz modela podataka sustava uporabom metode entiteti – veze.

Definicije i objašnjenja koncepata za izgradnju modela entiteti – veze opisanih u daljnjem tekstu uzeti su iz Pavlić, 2011.

Za izgradnju DEV-a potrebni su osnovni grafički simboli prikazani na slici 1:



Slika 1- osnovni grafički simboli za izgradnju DEV-a (Pavlić, 2011.)

Entitet

Entitet (engl. *entity)* označava neki pojam u poslovnoj organizaciji o kojemu se zahtjeva čuvanje podataka. Dakle, možemo tumačiti entitet kao bilo što što je nedvosmisleno tj. jednoznačno imenovano. On može biti apstraktni pojam, osoba, stvaran predmet, događaj, dokument i slično. Na primjer entitet može biti: student, konkretna škola, račun, država i dr.

Slični entiteti mogu se klasificirati u tipove entiteta po zajedničkim svojstvima. **Klasifikacija se** može opisati kao apstrakcija u kojoj se entiteti sličnih svojstava grupiraju i definiraju zajedničkim nazivom tip entiteta. Ona se koristi za kategorizaciju na polju modeliranja podataka, a kao znanstvena metoda priznata je i u drugim znanostima poput biologije ili bibliografije. Smatramo dva entiteta sličnima ukoliko su njihove osobine jednake (svojstva, atributi), ako mogu biti u istim vezama sa drugim entitetima i ako se nad njima mogu provesti iste operacije bilo da entiteti izvode operacije ili se operacije izvode nad entitetima. Na primjer: slične entitete Veljko, Josip i Petar možemo klasificirati kao tip entiteta Osoba jer svi imaju zajedničke osobine poput imena, datuma rođenja, mjesta rođenja, i dr. Iz ovoga zaključujemo da je **Tip entiteta** (engl. *entity type*) skupina pojedinačnih, sličnih entiteta, dobivenih procesom klasifikacije. Dakle skup entiteta istog tipa. On se grafički prikazuje pravokutnikom. (Pavlić,2011)

Za imenovanje tipa entiteta najprikladnije su imenice u jednini. Ukoliko je potrebno ispred imenice može se dodati riječ koja pobliže opisuje tip entiteta. Naziv mora biti kratak i jasan, te je najbolje koristiti naziv koji se koristi u poslovanju. (Varga, 2020)

Autor u (Pavlić, 2011) razlikuje pojmove **klasa entiteta** i tip entiteta. Klasom entiteta smatra se skup pojedinačnih pojavljivanja entiteta, dok je tip entiteta imenovana klasa entiteta. Iako je poželjno pravilno definirati pojmove često se u praksi klasa i tip entiteta ne razlikuju, te se radi lakše komunikacije koristi naziv entitet iako se misli na tip entiteta.

Suprotno apstrakciji klasifikacije je **uzorkovanje**, tj. izdvajanje jedno konkretno pojavljivanje entiteta iz tipa entiteta kako bi ga mogli detaljnije promotriti. Također entitete možemo klasificirati u stvarne poput Kupca, Vozila ili Robe i apstraktne poput Aktivnosti, Seminar i Projekt.

Tip vrijednosti

Vrijednost je nekakav podatak ili element koji sadrži informaciju o njemu tj., karakteristiku entiteta. Na primjer: plav, Croatia banka, 5, Mara Palijan i dr. Moguće se postaviti ograničenje na podatke poput dužine zapisa podatka u boju znakova. Na primjer: ime može biti duljine do 13 znakova.

Tip vrijednosti čini skup pojedinačnih vrijednosti. Na primjer: Datumi rođenja, Imena, Nazivi države. Grafički se tip vrijednosti prikazuje ovalom.

Tip podataka (engl. *data type*) označava vrstu podataka u tipu vrijednosti. Svaki pojedini tip vrijednosti ima podatke jednog tipa. Tako je tip podataka „Slova“ za tip vrijednosti „Ime“. Ukoliko smo definirali tip podataka za tip vrijednosti, nameće se ograničenje na tip vrijednosti.

Kombinirajući ove vrste ograničenje mogu se definirati ograničenja na dopuštene vrijednosti u tipu vrijednosti:

- Eksplicitno navođenje svih dopuštenih vrijednosti
- Definiranjem intervala iz kojih se vrijednosti mogu, ali i ne moraju pojaviti
- Određivanjem tipa podataka koji su dopušteni u tipu vrijednosti.
- Određivanjem maksimalne i minimalne dužine zapisa podataka
- Određivanjem oblika zapisivanja podataka

Atribut

Atribut tipa entiteta smatramo funkciju koja u tip vrijednosti ili kartezijev produkt preslikava tip entiteta. Atribut (engl. *Attribute*) je svojstvo određenog entiteta. To svojstvo je imenovana funkcija koja vrijednost iz tipa vrijednosti pridružuje entitetu. Atributi mogu biti: boja, godina, šifra dobavljača i sl. Atribut prevodi podatak u informaciju. On se grafički prikazuje pomoću imenovane linije koja spaja pravokutnik i oval, tj. entitet i tip vrijednosti. Pomoću EV metode možemo uočiti razliku između podatka i informacije. Podatak tumačimo kao vrijednost u tipu vrijednosti, dok informaciju smatramo kao model koji povezuje tip entiteta i tip vrijednosti atributom. Na primjer: broj 31 je podatak, dok atribut Starost radniku pridružuje 31 godinu iz tipa vrijednosti Broj godina. U modelu podataka potrebno je jedinstveno imenovati atribute.

Ključevi

Autor u (Pavlić, 2011) navodi da ključ tipa entiteta (engl. entity type key) čini skup atributa koji moraju zadovoljiti dva uvjeta neovisno o vremenu. Ukoliko neki skup atributa u relaciji nije ključ ali je ključ u nekoj drugoj relaciji, njega nazivamo vanjskim ključem (engl. foreign key). Da skup atributa postane kandidat za ključ, on mora zadovoljiti uvjet jedinstvenosti i uvjet neredundantnosti.

Uvjet jedinstvenosti:

Ne smiju postojati dva individualna pojavljivanja entiteta u tipu entiteta sa jednakom vrijednošću atributa koji čini ključ i ne smiju postojati dva tipa entiteta koji posjeduju jednak skup atributa za ključ.

Uvjet neredundantnosti:

Ukoliko se ijedan atribut iz ključa izostavi , uvjet jedinstvenosti se gubi. Ne smije postojati niti jedan atribut kao dio ključa koji se može izostaviti iz njega, a da uvjet jedinstvenosti pritom ne bude izgubljen.

U jednom tipu entiteta može postojati više kandidata za ključ jer zadovoljavaju uvjet jedinstvenosti i neredundantnosti, no u tom slučaju projektant modela podataka će odlučiti kojeg kandidata za ključ izabrati. Najčešće će to biti jednostavniji i dostupniji atribut. (Pavlić, 2011.).

Primarni ključ

Primarni ili glavni ključ (engl. *primary key*) naziva se jedan od odabranih kandidata za ključ tipa entiteta radi nekog značajnijeg razloga kao zamjenika entiteta tog tipa, najčešće radi njegovog čestog korištenja. Za svaki tip entiteta u EV metodi postoji samo jedan ključ koji nazivamo primarnim (glavnim) ključem, koji će nam služiti kao osnova za konkretno zapisivanje podataka u BP. Grafički ćemo primarni ključ označiti: podvlačenjem linije ispod imena ključnog atributa ili zapisivanjem dvostruke linije između tipa entiteta i tipa vrijednosti ili ga istaknuti od drugih atributa najčešće na način da se postavi oznaka „*“ iza naziva atributa.

Vrste atributa

Postoji više vrsta atributa poput: jednostavnog, složenog, viševrijednosnog i izvedenog atributa.

Jednostavni atribut (engl. *simple attribute*) još nazivamo i atomski (engl. *atomic attribute*) su obični atributi.

Složeni atributi (engl. *composite attribute*) su atributi složeni (agregirani) od više jednostavnih atributa, a ponekad se nazivaju i grupnim atributima. Dva ili više atributa se spajaju u jedan grupni atribut. Novonastali atribut dobiva svoj naziv i vezu sa njegovim atributima. Grafički se prikazuje spajanjem atomskih atributa sa složenim atributom.

Viševrijednosni atribut (engl. *multi-valued*) smatra se atribut ako jednom tipu entiteta odgovara više vrijednosti iz tipa vrijednosti. Grafički ga prikazujemo dvostrukim ovalom.

Kaže se da je **atribut izveden** (engl. *derived*) ako se do njegove vrijednosti može pristupiti operacijama preko drugih atributa u bazi podataka. Grafički ga prikazujemo isprekidanom linijom u obliku ovala. (Pavlić, 2011).

Tip veze

Veza (engl. *relationship*) je pridruživanje između entiteta. Ona prikazuje odnos između entiteta neovisno je li u stvarnosti ili mislima. Na primjer: veza Brak je odnos između entiteta Igor i entiteta Biserka. Vezu možemo tumačiti kao jedno pojedinačno pojavljivanje tipa veze. Ona je pojedinačna struktura povezivanja pojedinačnih entiteta poput jedne niti užeta.

Dva ista tipa entiteta mogu imati više pojedinačnih veza koje onda nazivamo **tip veze** (engl. *relationship type*). On se grafički prikazuje rombom ili linijom, a njegov naziv se upisuje ili u romb ili na liniju. Tip veze je opća struktura povezivanja tipova entiteta poput užeta s puno niti.

Red tipa veze je broj tipova entiteta povezanih tim tipom veze. Veza koja je između dva tipa entiteta je binarna. Veza uvijek mora imati dva kraja inače ona ne postoji jer nema smisla. Postoje i višestruke veze kao što je **trostruki** i **N-arni tip veze** gdje je n bilo koji prirodni broj veći od jedan.

Naziv tipa veze čini riječ ili skupina riječi koje označavaju nekakvu radnju (glagolske imenice, glagoli), operaciju ili odnos između entiteta (imenice) gdje nam jasno definiraju odnos između tipova entiteta.

Uloga (engl. *role*) entiteta u vezi je imenovana funkcija koja spaja (pridružuje) jedan entitet drugome. Ukoliko se za prikaz veze koristi romb, onda se uloga grafički prikazuje linijom koja povezuje tip entiteta i tip veze. Ona se imenuje i upisuje na DEV. U vezi se uloga zapisuje na liniju između tipova entiteta.

Ključ tipa veze ne postoji. Kako tip veze u ovoj verziji EV metode nema atribute tako ne postoji niti ključ tipa veze.

Brojnost tipa veze

U tipu veze postoje dvije uloge (preslikavanja) između tipova entiteta. Brojnost preslikavanja je ograničavanje tih uloga (preslikavanja).

Brojnosti tipa veze smatramo broj koji je eksplicitno upisan na DEV i govori nam koliko se entiteta pojedinog tipa entiteta pojavljuje u tipu veze s entitetom iz drugog entiteta.

Na DEV upisujemo brojnost na način: (1,1), (0,M), (1,M), (M,M) na liniju između tipova entiteta. Za obje uloge možemo odrediti donju (DG) i gornju granicu (GG) brojnosti.

Brojnost preslikavanja uloge je uređeni par (DG,GG), a Brojnost preslikavanja tipa veze su dva uređena para (DG, GG) : (DG, GG) gdje je prvi uređeni par određen za prvu ulogu, a drugi za drugu ulogu.

Karakteristični tipovi brojnosti:

- (1,1) : (1,1) – točno jedan entitet prvog tipa entiteta pridružuje se najmanje niti jednom ili najviše jednom drugog tipa entiteta i obratno.
- (1,1) : (0,M) - točno jedan entitet prvog tipa entiteta pridružuje se najmanje niti jednom i najviše jednom drugog tipa entiteta, a jedan entitet drugog tipa pridružuje se najmanje niti jednom i najviše većem broju entiteta prvog tipa entiteta.
- (0,M) : (0,M) – točno jedan entitet prvog tipa entiteta pridružuje se najmanje niti jednom i najviše većem broju entiteta drugog tipa i obratno.

(Pavlić, 2011.).

Jak i slabi tip entiteta

Autor u (Pavlić, 2011) razlikuje **jak tip entiteta** (engl. *strong entity*) od slabog tipa entiteta. Jak tip entiteta smatramo tip entiteta koji posjeduje vlastiti primarni ključ i ne ovisi o ostalim tipovima entiteta u modelu podataka. Na primjer: za tip entiteta Škola primarni ključ je Šifra škole.

Slabi tip entiteta (engl. *weak entity*) smatramo tip entiteta koji ovisi o drugom tipu entiteta, gdje se takva ovisnost prikazuje posebnim tipom veze između jakog i slabog tipa entiteta. Na primjer: tip entiteta Djeca, jer ovise o tipu entiteta Roditelji. Grafički prikaz slabog entiteta je dvostruki pravokutnik koji se nalazi jedan unutar drugoga.

Specijalni tip veze nazivamo vezu koja je između dva povezana tipa entiteta kod kojih postoji međusobna ovisnost. Prema ovisnostima možemo definirati različite vrste specijalnih tipa veza.

Vrste specijalnih tipova veza:

- Egzistencijalna (E)
- Identifikacijska (I)
- Egzistencijalna i identifikacijska (E&I)
- Prethođenja (P)
- Generalizacijska (G)
- Hijerarhijska (H)
- Agregacijska (sastoji se od dvije E&I između triju tipova entiteta)

Specijalni tip veze grafički se prikazuje poput običnog tipa veze s time da se dodaje strelica na liniji od jakog prema slabom entitetu.

Egzistencijalni tip veze (E) povezuje dva tipa entiteta gdje jedan od entiteta egzistencijalno ovisi o drugome. Tip entiteta koji je egzistencijalno ovisan nazivamo slabi tip entiteta, a tip entiteta o kojem ovisi nazivamo jaki tip entiteta. Preslikavanje sa slabog tipa entiteta k jakom tipu entiteta je uvijek brojnosti (1,1) i zove se **trivijalno preslikavanje**. U obratnom slučaju, preslikavanja sa jakog tipa entiteta ka slabom tipu možemo imati bilo kakav slučaj po brojnostima.

Identifikacijski tip veze (I) prikazuje tip veze gdje se jedan tip entiteta ne može jedinstveno identificirati sa vlastitim ključem, već mora koristiti ključ (identifikator) drugog tipa entiteta kako bi se ostvarila jedinstvena identifikacija.

Egzistencijalni i identifikacijski (E&I) je veza gdje je jedan tip entiteta istovremeno egzistencijalno i identifikacijski ovisan o drugom tipu entiteta.

Tip veze prethođenja (P) je veza gdje jednom tipu entiteta, ne nužno slabom tipu entiteta, prethodi postojanje drugog tipa entiteta. (Pavlić, 2011)

Generalizacija i specijalizacija

Postoje tipovi entiteta koji su nadtipovi drugih entiteta koje zovemo podtipovima. Autor u (Varga, 2020) opisuje **generalizacijski tip veze** kao uopćavanje tipa entiteta niže razine (podtipa) tipom entiteta više razine (nadtipa). Generalizacija opisuje vezu tj. odnost *„jest"*. Dok autor u (Pavlić, 2011) smatra **generalizacijsko stablo** skup tipova entiteta (podtipova i nadtipova) skupa sa generalizacijskim tipom veze. Kod generalizacije postoji pravilo da tipovi entiteta podtipa nasljeđuju osobine i operacije tipova entiteta nadtipa, ali ne i obratno.

Kod specijalizacije se tipovi entiteta nadtipa preslikavaju u tipove entiteta podtipa te se ona grafički prikazuje rombom sa nazivom „S". Kod generalizacije se tipovi entiteta podtipa preslikavaju u tipove entiteta nadtipa te se ona grafički prikazuje rombom sa nazivom „G".

Povratni tip veze tumačimo kao vezu koja povezuje jedan tip entiteta sa samim sobom. On se grafički prikazuje rombom i imenovanjem uloga povratnoga tipa veze. Postoji razlika prilikom crtanja povratne veze:

- Ukoliko je brojnost M:M po GG povratna veza crta se kao agregacija, te se uloge i brojnosti ne moraju pisati.
- Ukoliko je brojnost 1:M po GG povratna veza crta se kao specijalna veza, te se uloge i brojnosti ne moraju pisati.
- Ukoliko je brojnost 1:1 po GG povratna veza crta se kao obična veza, te se uloge i brojnosti mogu, ali i ne moraju pisati.

Agregacija često nazivana i mješovitim tipom entiteta u kojoj se veza između dva ili više tipa entiteta smatra kao novi tip entiteta.

Tip veze će postati agregirani tip entiteta u slučaju:

- Pojavljivanja atributa tipa veze
- Potrebe za povezivanjem dvaju tipova veze međusobno
- Rastavljanjem višestrukih tipova veza u binarne tipove
- Utvrđivanje brojnosti tipa veze u vrijednosti M po GG sa obje strane

Ona se grafički prikazuje pravokutnikom unutar kojega se nalazi romb. Iz njega se može zaključiti da je koncept prvo bio tip veze a potom postao tip entiteta. Uloge se crtaju poput vektora iz tipa entiteta prema agregaciji i to bez naziva. Uloga sada ima zadaću specijalnog tipa veze između jakog tipa entiteta i agregacije.

Komponente agregacije su tipovi entiteta koji su povezani s agregacijom.

Slaba agregacija je apstrakcija koja prikazuje jedan tip slabe agregacije i slabog tipa entiteta. Grafički prikaz slabe agregacije je skraćeni zapis sasatavljen od simbola slabog tipa entiteta i simbola agregacije istodobno.

Ključ agregacije čini skup ključnih atributa komponenti agregacije. Ona dobiva ključ od komponenti agregacije, te je brojnost ključa tipa agregacije uvijek (1,1) : (1,1).

Ključ slabe agregacije čini skup ključnih atributa komponenti agregacije i ključ slabog tipa entiteta u slaboj agregaciji. Ona dobiva ključ od komponenti agregacije i najčešće redni broj stavke.

IDEF

Američko zrakoplovstvo je već sedamdesetih godina pokušavalo pronaći način kako pomoću sustavne primjene računalne tehnologije povećati produktivnost. Rezultat istraživanja njihovog programa za kompjutorski podržanu integriranu proizvodnju (engl. Integrated Computer Aided Manufacturing, skraćeno: ICAM) propisana je serija modela ICAM Definition a, 1999. je preimenovana u Integration Definition, poznatija pod kraticom IDEF. Iako američko zrakoplovstvo, druge vojske i ministarstvo obrane Sjedinjenih Država (United States Department of Defense, skraćeno DoD) i dalje najčešće koristi ovu metodu ona je javno vlasništvo, te je dostupna svima. Ona je specijalizirana metodologija koja se sastoji od brojnih metoda specijaliziranih za određeno područje. Pokriva širok spektar uporabe, od funkcionalnog modeliranja do modeliranja podataka, simulacija, objektno orijentiranih analiza i dizajna. Ne postoji jedinstvena metoda koja bi obuhvaćala sve faze od specifikacije do oblikovanja, stoga je potrebno povezati različite metode u jednu, suvislu cjelinu, tj. potrebna je integracija. Takva povezanost nastojala se postići prilikom razvoja IDEF metoda. (Wikipedia 2021)

S vremenom IDEF metode definirane su do IDEF14:

- IDEF0 – Function modeling
- IDEF1 – Information modelin
- IDEF1X – Data modeling
- IDEF2 – Simulation model design
- IDEF3 – Process description capture
- IDEF4 – Object-oriented design
- IDEF5 – Ontology description capture
- IDEF6 – Design rationale capture
- IDEF7 – Information system auditing
- IDEF8 – User interface modeling
- IDEF9 – Business constraint discovery
- IDEF10 – Implementation artifact modeling
- IDEF12 – Organization modeling
- IDEF13 – Three schema mapping design
- IDEF14 – Network design

Najčešće korištene i priznate metode IDEF metodologije su IDEF0 i IDEF1X. Metoda na koju ćemo se fokusirati je IDEF1X metoda koja se bavi informacijskim modelima i dizajnom baze podataka. (Wikipedia 2021)

Vrste IDEF metoda

IDEF0

Ova metoda je dizajnirana za modeliranje odluka, radnji i aktivnosti nekog sustava ili organizacije. Izrađena je na temelju etabliranog jezika za grafičko modeliranje Strukturna analiza i tehnika dizajna (Structured Analysis and Design Technique (SADT)). U svom originalnom obliku IDEF0 uključuje i definicije jezika za grafičko modeliranje (njena sintaksa i semantika) i opis cjelovite metodologije za razvoj modela. Američko zračno zrakoplovstvo dodijelio je zadatak SADT programerima razvitak metode modeliranja funkcija za analizu i komunikaciju strategije sustava. IDEF0 kao alat za analizu pomaže moderatoru u pronalaženju informacija koje se funkcije izvode i što je potrebno kako bi se one izvele, te što konkretni sustav radi ispravno, a što pogrešno. Prilikom razvoja sustava jedan od prvih zadataka su IDEF0 modeli. Cilj IDEF0 je pomoći pri analizi sustava i promovirati učinkovite komunikacije između analitičara i korisnika putem pojednostavljenih grafičkih uređaja.

Laboratorij računalnih sustava nacionalnog instituta za standarde i tehnologiju u Americi (Systems Laboratory of the National Institute if Standards and Technology (NIST)) objavio je IDEF0 kao standard za funkcijsko modeliranje u FIBS Publication 183 (Federal Information Processing Standards su javno objavljeni standardi računalne tehnologije izrađeni na institutu za standarde i tehnologiju koji se ne koriste u vojne svrhe.) (Knowledge Based Systems, I. idef.com, 2021)

IDEF1

Metoda IDEF1 razvijena je za modeliranje informacija. Ona je dizajnirana kao metoda za analizu i komunikaciju prilikom definiranja zahtjeva. Generalno se IDEF1 koristi za:

1. Utvrđivanje informacija kojima se trenutno raspolaže u nekoj organizaciji.
2. Utvrđivanje koji od problema nastalih prilikom analize su uzrokovani pogrešnim upravljanjem informacijama.
3. Određivanje kojim će se informacijama upravljati u budućoj implementaciji u sustav.

Ova metoda bilježi postojeće informacije o objektima unutar poduzeća. Njena perspektiva informacijskog sustava uključuje ne samo komponente automatiziranog sustava(računalna oprema), već i druge objekte poput ljudi, telefona, i dr. Razvijena je kao metoda koja bi pomogla organizacijama analizirati i izraziti svoje zahtjeve i potrebe upravljanja informacijskim resursima.

Autori u (Knowledge Based Systems, I. idef.com, 2021) definiraju IDEF1 umjesto metode za oblikovanje baza podataka kao metodu za analizu, te se ona koristi za:

1. Prikupljanje, pohranu i upravljanje informacijama poduzeća
2. Pravila koja uređuju upravljanje informacijama
3. Logičke veze unutar poduzeća koje se odražavaju u informacijama
4. Problemi koji su rezultat lošeg upravljanja informacijama

IDEF2

Izvorno je zamišljena kao metoda za modeliranje korisničkog sučelja, no ICAM-u je bio potreban alat za simulacijsko modeliranje i tako je nastala metoda dizajna simulacijskog modela za predstavljanje vremenski promjenjivog ponašanja resursa u proizvodnom sustavu koji je pružao razvojno okruženje za specifikaciju simulacija zasnovanih na matematičkom modelu. Metodološki program unutar ICAM-a imao je namjeru ispraviti ovakvu situaciju, ali financije to nisu dopuštale. Rezultat toga je nedostatak metode koja bi podržavala strukturiranje opisa korisničkog pregleda sustava koji se pokazao kao veliki nedostatak IDEF sustava. Sa metodološkog stajališta osnovni problem je razlikovanje između opisa što bi sustav (postojeći ili predloženi) trebao raditi i predstavljenog simulacijskog modela koji predviđa što će sustav učiniti. (Wikipedia 2021).

IDEF3

Metoda prikupljanja opisa koja omogućava sustav za prikupljanje i dokumentiranje procesa. IDEF3 sakuplja informacije o relacijama između situacija i događaja i pruža strukturiranu metodu o tome kako sustav, procesi ili organizacija radi.

Najčešće se IDEF3 opisi koriste za:

1. Snimanje neobrađenih podataka dobivenih kao rezultat intervjua za utvrđivanje činjenica aktivnosti analize sustava.
2. Određivanje utjecaja informacijskih izvora organizacije na ključne zadaće poslovanja.
3. Bilježenje postupaka odluka koji imaju utjecaj na stanje i životni ciklus kritičnih zajedničkih podataka, naročito podataka o proizvodnji, inženjerstvu i održavanju proizvoda.
4. Analizu i dizajn sustava
5. Upravljanje konfiguracijom podataka i promjenom kontrolnih pravila
6. Pružanje generiranja simulacijskog modela

IDEF4

Intuitivnost objektno-orijentiranog programiranja olakšava pisanje programskog koda, no nažalost ta lakoća kojom se stvara softver omogućuje proizvodnju softvera lošeg dizajna, što rezultira nedostatkom ponovne uporabljivosti, modularnosti i održavanja sustava. Kako bi se tehnologija ispravno koristila i kako bi pomogla riješiti navedene probleme, osmišljena je metoda IDEF4. Između više od trideset objektno-orijentiranih metoda projektiranja IDEF4 se ističe jer promatra objektno-orijentirani dizajn kao dio šireg razvojnog okvira sustava, a ne kao objektno-orijentiranu metodu analize i dizajna koja je često dvosmislena. On se značajno razlikuje od drugih objektno-orijentiranih metoda projektiranja prvenstveno radi pridavanja važnosti procjenjivanja utjecaja dizajna interakcija između nasljeđivanja klasa, sastava objekta, funkcijske dekompozicije i polimorfizma. (Knowledge Based Systems, I. idef.com, 2021)

IDEF5

Metoda IDEF5 ili ontologija prikupljanja opisa omogućuje teorijski i empirijski dobro utemeljenu metodu prvenstveno dizajniranu za pomoć pri stvaranju, održavanju i mijenjanju ontologija. (Knowledge Based Systems, I. idef.com, 2021)

U IDEF5 metodi ontologija se konstruira dohvaćanjem sadržaja određenih tvrdnji o objektima u stvarnome svijetu, njihovim svojstvima i međusobnim odnosima, te predstavljanje tog sadržaja u intuitivnom i prirodnom obliku. Ona se sastoji od tri glavne komponente: grafički jezik koji podržava konceptualnu onkološku analizu, strukturirani tekstualni jezik za detaljnu karakterizaciju ontologije i sustavni postupak koji daje smjernice za učinkovito obuhvaćanje ontologije. (Wikipedia, 2021)

Njene standardizirane procedure, mogućnost zastupanja ontologija u intuitivnom i prirodnom obliku, te kvalitetniji rezultati omogućeni primjenom IDEF5 alata također služe za smanjenje troškova izvođenja aktivnosti. (Knowledge Based Systems, I. idef.com, 2021)

IDEF6

Metoda za oblikovanje logičke podloge. Razvijena je s ciljem olakšavanja, stjecanja, predstavljanja i manipulacijom obrazloženjima dizajna koje se koristi u razvoju poslovnih sustava. Obrazloženje se tumači kao razlog, opravdanje, temeljna motivacija ili izgovor koji je dizajnera potaknuo na odabir određene strategije ili značajke dizajna. IDEF6 je metoda koja sadrži resurse i potrebne jezične mogućnosti za prikaz prirode i strukture informacija kod obrazloženja dizajna unutar konkretnog sustava, i povezivanje tog obrazloženja sa specifikacijama dizajna, modelima i dokumentacijom sustava. Od početne konceptualizacije, pa sve do posljednje faze, IDEF6 metoda je primjenjiva na sve dijelove razvoja informacijskog sustava. Preporučljivo je koristiti ju i tijekom procesa izgradnje softvera. (Wikipedia 2021).

IDEF8

Metoda za modeliranje korisničkog sučelja. Specijalizirana je za izradu kvalitetnog dizajna interakcija između korisnika i sustava kojim upravljaju. Sustavi predstavljaju skup objekata koji vrše funkcije za ostvarivanje odgovarajućeg cilja. Korisnik može stupiti u interakciju sa različitim sustavima, ne nužno samo sa računalnim programom. Definirane su tri razine interakcije između korisnika i sustava: 1) Definira kako sustav radi te proizvodi skupinu modela i opisa svih procesa u sustavu. 2) Definira scenarije uporabe usredotočene na uloge u sustavu. 3) Definira detalje dizajna sustava gdje IDEF8 pruža mogućnost korištenja biblioteke metafora, tj. već poznatih objekata i iskustava kao pomoć dizajnerima i korisnicima pri određivanju željenog ponašanja. (Wikipedia 2021).

IDEF9

Metoda za specifikaciju informacijskog sustava vođena scenarijima osmišljena je kako bi pomogla u otkrivanju i analizi ograničenja (pravila) u poslovnom sustavu. Na razvoj IDEF9 metode uvelike je utjecala činjenica da je skup pravila koje kreiraju poslovnu organizaciju generalno loše definiran. Loše poznavanje postojećih pravila, te u kojem su međusobnom odnosu ta pravila nije potpuno definirano, često nepovezano, raspodijeljeno pa čak i potpuno nepoznato. Kako bi poslovna organizacija funkcionirala na siguran i predvidljiv način potrebno je dobro poznavanje ovih pravila (Wikipedia 2021).

IDEF14

Metoda za oblikovanje mreže. IDEF14 je metoda kojoj je cilj modeliranje i dizajn komunikacijskih i računalnih mreža. Ona se koristiti prilikom modeliranja postojećih ili novih mreža. Pomoću IDEF4 metode dizajner mreže može istražiti potencijalne dizajne mreže i dokumentirati pojašnjenje svog dizajna. IDEF4 se razvila iz potrebe za pouzdanim mrežnim dizajnom koji se može provesti točno i u najkraćem mogućem roku (Wikipedia 2021).

Manje korištene IDEF metode:

- IDEF7 - Metoda koja se koristi za ocjenjivanje informacijskog sustava
- IDEF10 - Metoda za modeliranje arhitekturne implementacije
- IDEF11 - Metoda za modeliranje izradbe informacija
- IDEF12 - Metoda za modeliranje organizacije
- IDEF13 - Metoda za oblikovanje sheme triju pridruživanja
  (Pavlić, 2009)

IDEF1X

Općenito o IDEF1X

Autor u (National Institute of Standards and Technology, 1993) opisuje **Integration DEFinition for information modeling** ili **IDEF1X** kao metodu za modeliranja podataka koja definira odgovarajuća pravila i tehnike za razvoj logičkog modela podataka. IDEF1X se koristi za prikaz informacijskih modela koji predstavljaju strukturu i semantiku informacija unutar nekog sustava pomoću određenih grafičkih simbola. Ova metoda dozvoljava izgradnju semantičkih modela podataka koji se mogu koristiti kao podrška upravljanja podacima kao resursima, integracijom informacijskih sustava i izgradnjom baza podataka.

Koncepti IDEF1X metode:

- ❖ Entiteti
  - Entiteti neovisni o identifikatoru
  - Entitet ovisni o identifikatoru
- ❖ Veze
  - Veze koje ovise o identifikatoru
  - Odnosi između veza koji se ne identificiraju
  - Veze kategorizacije
  - Nespecifične veze
- ❖ Atributi
  - Atributi
- ❖ Ključevi
  - Primarni ključevi
  - Sekundarni ključevi
  - Strani/Vanjski ključevi

Definicije i objašnjenja koncepata za izgradnju korištenjem IDEF1X metode opisanih u daljnjem tekstu uzeti su iz National Institute of Standards and Technology, 1993.

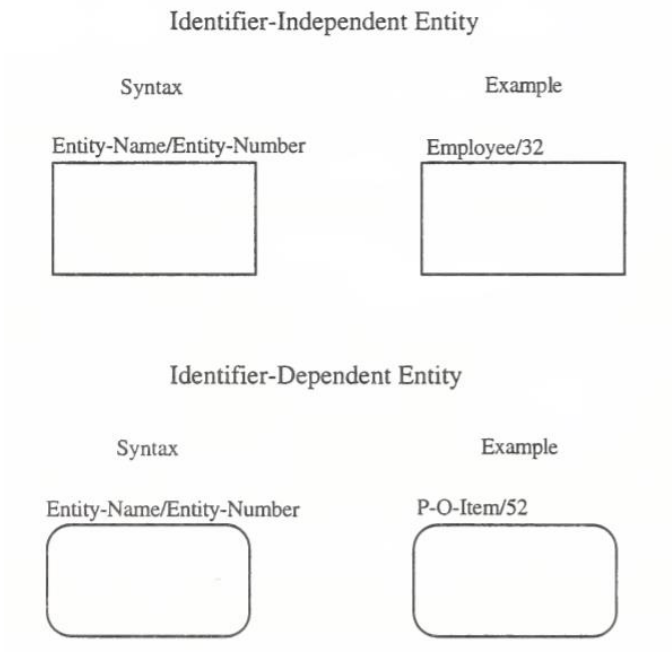
Entiteti

Entitet predstavlja skup realnih stvari ili apstrakciju poput ljudi, objekata, ideja, mjesta i dr. koji imaju zajedničke atribute ili karakteristike. **Instanca entiteta** je pojedinačno pojavljivanje entiteta iz skupa. Unutar pogleda stvarni objekt ili stvar može se prikazati sa više entiteta. Na primjer: Ana Anić može biti instanca entiteta STUDENT i instanca entiteta PROFESOR. Instanca entiteta može prikazivati i kombinaciju objekata iz stvarnoga svijeta. Na primjer: Mateo i Ivan mogu biti instanca entiteta BRAĆA.

Autori u (Halpin & Morgan, 2008) smatraju da je **entitet neovisan o svom identifikatoru** (engl. *identifier-independent*) ili jednostavnije neovisan (engl. *independent*) ukoliko se svaka instanca entiteta može jedinstveno identificirati bez potrebe za utvrđivanjem njegove veze sa drugim entitetom. Tada se zaključuje da je **entitet ovisan o svom identifikatoru** (engl. *identifier-dependent*) ili jednostavnije ovisan (engl. *dependent*) ukoliko je jedinstvena identifikacija instance entiteta ovisna o vezi sa drugim entitetom.

Za grafički prikaz entiteta koristi se pravokutnik kao što se vidi na slici 2. Ukoliko je entitet ovisan o identifikatoru onda su kutovi pravokutnika zaobljeni, u suprotnome se kutovi ne zaobljavaju. Naziv entiteta mora biti jedinstven i mora se smjestiti iznad pravokutnika. Entitetu možemo dodijeliti i pozitivan cijeli broj kao dio naziva, no on mora biti odvojen od naziva kosom crtom („/“). Za naziv entiteta koristimo imenice u jednini koje opisuju skup stvari koje taj entitet prikazuje. Kratice i akronimi dopušteni su pri imenovanju entiteta, međutim naziv mora biti smislen i dosljedan kroz cijeli model. Sve formalne definicije entiteta i popisi sinonima ili

pseudonima moraju se definirati u rječniku. Entitet se može pojavljivati kroz više dijagrama, no samo se jednom smije pojaviti u jednom dijagramu.



Slika 2- Primjer entiteta, (Technology, 1993)

Pravila entiteta:

- Svaki entitet mora imati naziv koji je jedinstven i koji uvijek ima isto značenje. Različiti nazivi ne smiju imati isto značenje osim ako su nazivi pseudonimi.
- U ključnom ili potpuno atribuiranom slučaju, entitet ima jedan ili više atributa koji su vlastiti ili dodijeljeni entitetu kroz vezu.
- U ključnom ili potpuno atribuiranom slučaju, entitet ima jedan ili više atributa čije vrijednosti jedinstveno identificiraju svaku instancu entiteta.
- Entitet može imati nebrojeno veza sa drugim entitetima u pogledu.
- Ukoliko cijeli vanjski ključ postane primarni ključ drugog entiteta onda je taj entitet ovisan o identifikatoru, no ako niti jedan ili samo jedan dio vanjskog ključa postane primarni ključ nekog drugog entiteta onda je taj entitet neovisan o identifikatoru.
- U pogledu, entitet je označen nazivom ili jednim od pseudonima. Može biti označen pseudonimom u drugom pogledu istog modela.
- Niti jedan pogled ne smije sadržavati dva entiteta čija su imena sinonimi.

Domena

Iz **domene** (engl. *domain*) atributi dobivaju svoje vrijednosti. Domenu možemo tumačiti kao skup vrijednosti. Kod IDEF1X metode domena je definirana odvojeno od entiteta i pogleda kako bi se omogućila njihova ponovna uporaba kroz sustav. Smatra se klasom koja ima fiksnu, a moguće i neograničenu skupinu instanci. Na primjer: *JMBAG studenta*

(**JMBAG** (**J**edinstveni **M**atični **B**roj **A**kademskog **G**rađana) jedinstveno određuje svaku osobu akademske zajednice) se sastoji od 10 znamenki. Za domene kažemo da su nepromjenjive klase čije se vrijednosti ne mijenjaju kroz vrijeme, kao što je slučaj kod entiteta.

Domene dijelimo na:

- Osnovne domene (engl. *Base domain*)
- Zapisane domene (engl. *Typed domain*)

Osnovne domene mogu sadržavati tipove podataka Znakovnog tipa, Numeričkog tipa ili Boolean. Uz ova tri osnovna tipa podataka u IDEF1X mogu se koristiti još i datumi, vrijeme, binarni zapis i dr. Osnovnoj domeni mogu se definirati pravila, od kojih su najčešći lista vrijednosti (engl. *value list*) i pravila raspona (engl. *range rules*), koja propisuju prihvatljive vrijednosti domene.

Lista vrijednosti opisuje sve prihvatljive instance vrijednosti neke domene. Atributi domene kojima je definirano pravilo liste vrijednosti su valjani ako i samo ako se njihova instanca vrijednosti nalazi na listi vrijednosti. Najčešće se koristi prilikom definiranja popisa šifriranih vrijednosti poput JMBAG-a studenta.

Pravilo raspona opisuje sve prihvatljive instance vrijednosti neke domene, gdje se ta instanca smatra donjom ili gornjom granicom. Na primjer: ocjena na ispitu mora biti između broja 1 i broja 5. Domena ne mora biti definirana pravilima, no u tom slučaju se ona ograničava sa tipom podataka koji se u njoj nalazi.

Klasificirana domena je podtip (engl. *sub-type*) osnovne domene koja može definirati prihvatljive vrijednosti domene. Ona postoji ukoliko je ograničena tipom podataka i zadovoljava pravilo nadređene ili nad-tipa (engl. *supertype domain*) domene. Radi toga se može definirati hijerarhija domena koja se prilikom spuštanja na niže nivoe sužava. Domena ima **generalizacijsku hijerarhiju.** Podtipovi domena u hijerarhiji se međusobno ne isključuju, kao što je to slučaj kod entiteta. Sve informacije o domeni potrebno je zapisati u rječnik jer ne postoje konkretna sintaksa za domene u IDEF1X metodi.

Pravila domene:

- Domena mora imati naziv koji je jedinstven i koji uvijek ima isto značenje. Različiti nazivi ne smiju imati isto značenje osim ako nazivi su pseudonimi.
- Domena je ili osnovna ili klasificirana.
- Domena može imati pravila.
- Pravila domene definiraju se rasponom ili listom vrijednosti.
- Pravilo raspona definira valjane instance sa donjom i/ili gornjom granicom vrijednosti
- Lista vrijednosti definira valjane instance ako se nalaze na listi vrijednosti.
- Klasificirana domena je podtip osnovne domene ili druge klasificirane domene.
- Niti jedna domena ne može biti direktno ili indirektno podtip same sebe.

Pogled

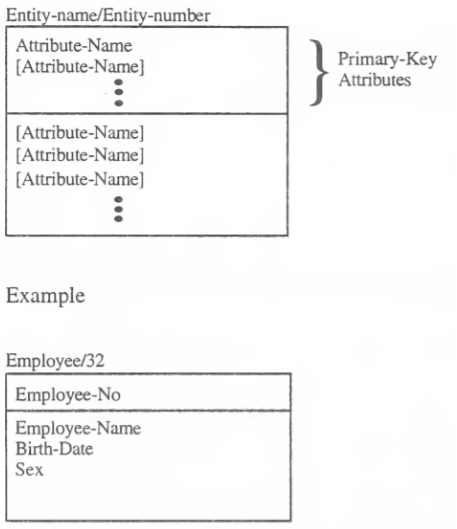
Pogled (engl. *view*) je skup kolekcija entiteta i dodijeljenih atributa raspoređenih radi neke svrhe. Pomoću pogleda možemo promatrati cijeli model ili samo dio modela. Entiteti i atributi su definirani u rječniku i povezani u pogledima. Tada na primjer entitet STUDENT se može pojaviti u više pogleda i modela i može imati različite atribute povezane s njim u svakom pogledu. Potrebno je da entitet STUDENT uvijek znači istu stvar. Pogled se mora imenovati i ponekad dodatno opisati . Opcionalno se na njega može napisati ime autora, datum izrade i zadnjeg posjećivanja, razinu obrade (ER, key-based, fully-attributed) i dr. Također se može dodati i tekstualni opis veza, entiteta, atributa i pravila.

Pravila pogleda:

- Svaki pogled mora imati jedinstveno ime
- Autor moraju biti dosljedan kroz sve poglede svog modela
- Model može sadržavati poglede različitih razina

Atributi

Atributom entiteta se naziva domena u vezi sa entitetom. U IDEF1X metodi atributi su povezani sa određenim entitetima. Atribut predstavlja svojstvo ili stvar koja je povezana sa realnom ili apstraktnim objektom poput ljudi, mjesta, ideja i dr. Instanca atributa (engl. *attribute instance*) je određena karakteristika pojedinca skupa. Instanca atributa se definira svojstvom i vrijednošću tog svojstva (engl. *attribute value*). Instanca entiteta ima jednu specifičnu vrijednost za svaki njoj dodijeljeni atribut. Na primjer: IME-STUDENTA i DATUM-ROĐENJA su atributi u vezi sa entitetom STUDENT. Instanca entiteta STUDENT može imati atribute vrijednosti „Ana Anić“ i „20.05.1997.“. Entitet mora imati atribut ili kombinaciju atributa koji imaju jedinstvenu identifikacijsku vrijednost entiteta. Takvi atributi postaju primarni ključ (engl. *primary-key*) entiteta. Na primjer: atribut STUDENT-JMBAG može biti primarni ključ, dok atributi STUDENT-IME i DATUM-ROĐENJA nemaju jedinstvenu identifikacijsku oznaku i ne mogu biti ključevi. Entitet može dobiti atribute koji pripadaju drugim atributima kroz nasljeđivanjem kroz specifičnu vezu ili vezu kategorizacije. Atributi sa primarnim ključem su jedini koji mogu preći iz jednog entiteta u drugi. Svaki atribut mora imati jedinstven naziv svoje ishodišne domene. Za imenovanje se koristi imenica u jednini koja opisuje svojstvo koje atribut predstavlja. Kao i kod entiteta, kratice i akronimi su dopušteni, ali autor mora biti dosljedan kroz cijeli model. Atributi se grafički prikazuju listom unutar pravokutnika (entiteta) proizvoljnih veličina kao što je prikazano na slici 3. Atributi koji čine primarni ključ svrstavaju se na vrh liste i odvajaju horizontalnom linijom od ostalih. Ključni atribut mora imati neku vrijednost, dakle ne smije biti null. Ukoliko atribut nije dio primarnog ključa, on smije biti null vrijednosti, ali mora biti jasno označen simbolom „O“ nakon naziva atributa. Niti jedan pogled ne smije sadržavati dva atributa čija su imena sinonimi.



Slika 3 - Primjer atributa i primarnog ključa, (Technology, 1993)

Veze

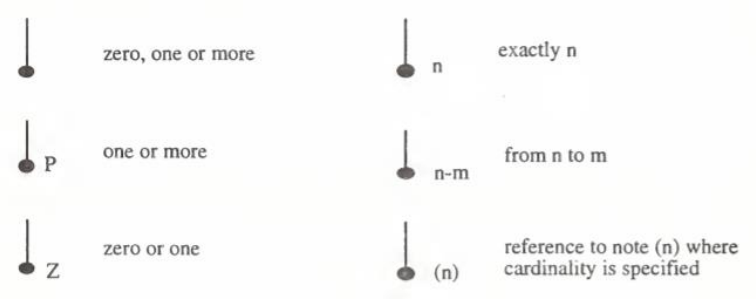
Veza specijalnih odnosa (engl. *specific connection relationship*) često se naziva i veza roditelj-dijete (engl. *parent-child relationship*) je veza između entiteta u kojemu je svaka instanca entiteta roditelj povezana s niti jednom, jednom ili više instanci drugog entiteta dijete i svaka instanca entiteta dijete povezana je sa niti jednom ili jednom instancom entiteta roditelj.

IDEF1X ima brojnosti veze:

- Svaka instanca entiteta roditelj može imati nula ili više instanci entiteta dijete
- Svaka instanca entiteta roditelj mora imati barem jednu pridruženu instancu entiteta dijete
- Svaka instanca entiteta roditelj može imati nula ili više instanci dijete
- Svaka instanca entiteta roditelj je u vezi sa točno određenim brojem instanci entiteta dijete
- Svaka instanca entiteta roditelj je u vezi sa određenim rasponom instanci entiteta dijete.

Kako postoji brojnost iz perspektive entiteta roditelj, tako postoji i brojnost veze iz perspektive entiteta dijete. Veza specijalnih odnosa prikazuje se kao linija između entiteta roditelj i entiteta dijete i točkom na kraju linije kod entiteta dijete.

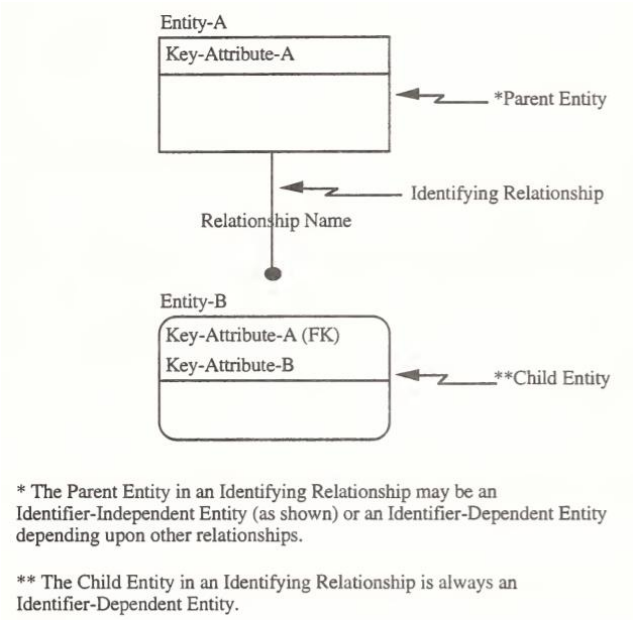
Slika 4 opisuje brojnost veze entiteta. Entitet dijete može kao početnu brojnost imati nula, jedan ili više. Oznaka „P“ se stavlja pored točke kao oznaka brojnosti jedan ili više, a oznaka „Z“ kao oznaka nula ili jedan. Ukoliko je brojnost neki pozitivan cijeli broj tada se pored točke zapisuje taj broj. Kada se treba označiti raspon on se također zapisuje pored točke. Ostale brojnosti se zapisuju u obliku bilješki pored točke. Naziv veze je u obliku glagola ili neki glagolski izraz koji se zapisuje pored linije veze. Naziv veze se najčešće specificira u smjeru roditelj-dijete, a kada se veza imenuje iz obje perspektive onda se imenuje prvo iz roditeljske perspektive, pa slijedi znak „/“ i onda iz perspektive djeteta.



Slika 4 - Primjer brojnosti veza, (Technology, 1993)

Identifikacijska veza

Kada je instanca entiteta dijete identificirana u odnosu sa entitetom roditelj kao što je prikazano na slici 5, onda tu vezu možemo nazvati **identifikacijska veza** (engl. *identifying relationship*). Svaka instanca entiteta dijete mora biti povezana sa točno jednom instancom entiteta roditelj. Entitet dijete u identifikacijskoj vezi uvijek egzistencijalno ovisi o entitetu roditelj. Identifikacijska veza grafički se označava punom linijom između entiteta dijete i roditelj sa točkom kod entiteta dijete. Ukoliko ova veza postoji, entitet dijete je uvijek egzistencijalno ovisan entitet grafički prikazan zaobljenim pravokutnikom sa primarnim ključem entiteta roditelj.



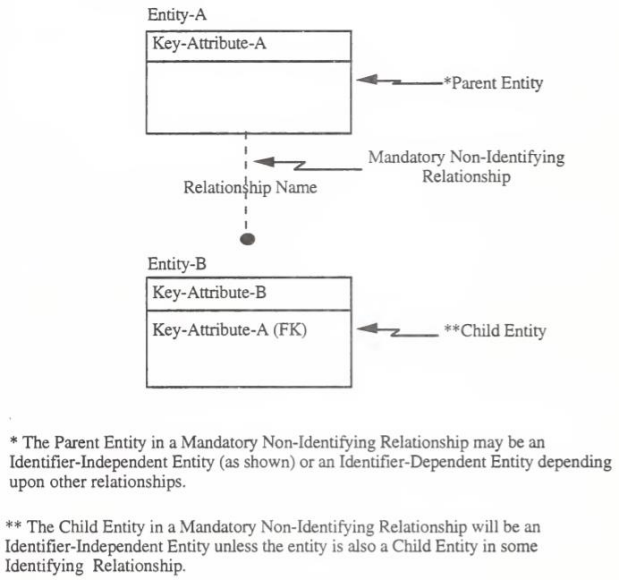
Slika 5- Primjer identifikacijske veze, (Technology, 1993)

Neidentifikacijska veza

Neidentifikacijskom vezom (engl. *non-identifying relationship*) nazivamo vezu kod koje se svaka instanca entiteta dijete može jedinstveno identificirati bez da znamo s kojom je instancom entiteta roditelj povezana. Neidentifikacijska veza grafički se prikazuje isprekidanom linijom između entiteta roditelj i entiteta dijete. Entitet roditelj i entitet dijete će oboje biti identifikacijski neovisni jedino ako jedan ili oboje u nekoj drugoj vezi koja je identifikacijska. (Halpin & Morgan, 2008)

Obvezna ne identifikacijska veza

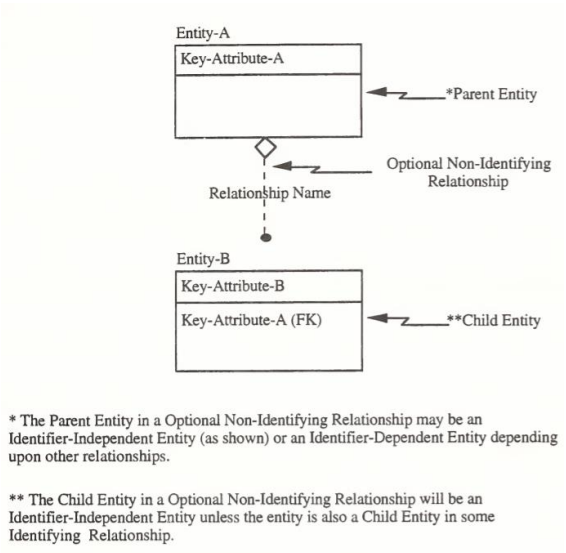
U **obveznoj neidentifikacijskoj vezi** (engl. *mandatory non-identify relationship*) svaka instanca entiteta dijete je u vezi sa točno jednom instancom entiteta roditelj kao što je prikazano na slici 6. (National Institute of Standards and Technology, 1993.).



Slika 6 - Primjer obavezne neidentifikacijske veze, (Technology, 1993)

Opcionalna neidentifikacijska veza

Opcionalna neidentifikacijska veza (engl. *optional non-identifying relationship*) grafički se prikazuje isprekidanom linijom sa rombom na kraju entiteta roditelj između entiteta roditelj i entiteta dijete. Kod opcionalne ne identifikacijske veze svaka instanca entiteta dijete je u relaciji sa nula ili jednom instancom entiteta roditelj.



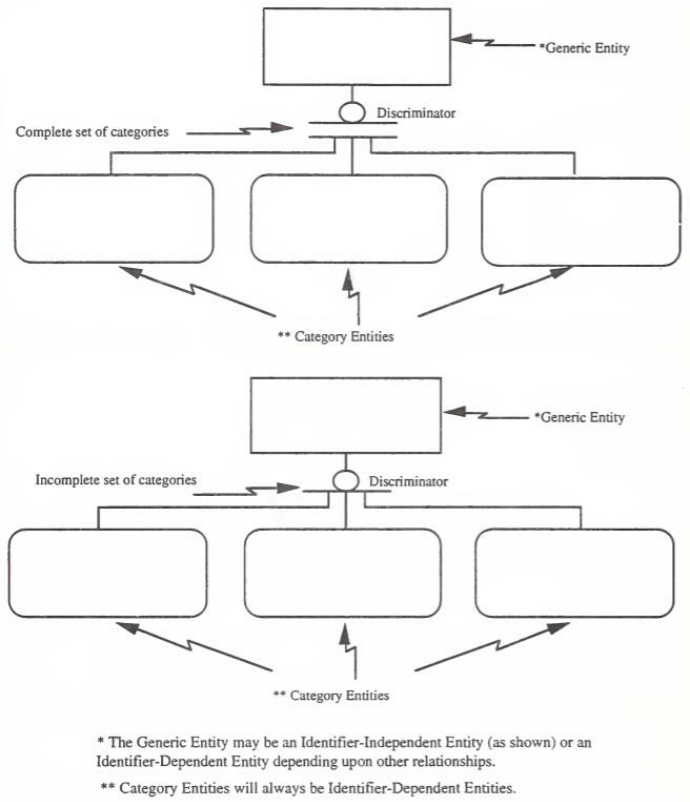
Slika 7 - Primjer opcionalne ne identifikacijske veze, (Technology, 1993)

Veze kategorizacije

Veze kategorizacije predstavljaju strukture u kojima je jedan entitet kategorija tj. „tip“ drugog entiteta. U stvarnome svijetu pojedine stvari čine kategorije nekih drugih stvari, zato imamo potrebu pojedine entitete smatrati kategorijama nekih drugih entiteta. Na primjer: iako imamo već neke informacije o entitetu ZAPOSLENIK kroz informacije o svim zaposlenicima, dodatne i različite informacije poput njihovih mjesečnih plaća i plaće po satu morati ćemo pronaći kroz vezu kategorizacije. Iz ovoga slijedi da su entiteti ZAPOSLENIK-PLAĆEN-MJESEČNO i entitet ZAPOSLENIK-PLAĆEN-PO-SATU kategorije entiteta ZAPOSLENIK povezani vezom kategorizacije. Ovakva vrsta veze može se koristiti i u slučaju da entitet neke određene kategorije mora imati vezu koja podržava samo jednu specifičnu kategoriju između mnogo drugih kategorija. Na primjer: ZAPOSLENIK-PUNO-RADNO-VRIJEME može koristiti NAKNADU dok ZAPOSLENIK-POLA-RADNOG-VREMENA to ne može.

Veza kategorizacije tumači se kao veza između jednog entiteta zvanog „generički entitet“ i drugog entiteta zvanog „entitet kategorije“. Skupinu od jedne ili više veza kategorizacije nazivamo „skupinom kategorija“ (engl. *category cluster*). Instanca jednog generičkog entiteta može biti u vezi samo sa jednom instancom entiteta iz skupa kategorija. Budući da se instanca generičkog entiteta ne može biti u vezi sa više od jednom instancom entiteta iz skupa kategorija, entiteti kategorija se međusobno isključuju, što znači da na primjer: zaposlenik plaćen mjesečno ne može biti plaćen i po satu. No, entitet može biti generički u više skupina kategorija i kategorizirani entiteti u skupini se međusobno ne isključuju sa drugima. Na primjer: generički entitet je ZAPOSLENIK i može biti prikazan u skupini kategorija ŽENSKI-ZAPOSLENIK i MUŠKI-ZAPOSLENIK. ZAPOSLENIK se može povezati instancom entiteta ZAPOSLENIK-PLAĆEN-MJESEČNO ili ZAPOSLENIK-PLAĆEN-PO-SATU te istovremeno sa instancom entiteta ŽENSKI-ZAPOSLENIK ili MUŠKI-ZAPOSLENIK. U **cjelovitom skupu kategorija** (engl. *complete category cluster*) svaka instanca generičkog entiteta je povezana s instancom entiteta kategorije. U **necjelovitom skupu kategorija** (engl. *incomplete category cluster*) instanca generičkih entiteta postoj i bez povezivanja sa instancama drugih kategorija.

Kao što je prikazano na slici 8 skupina kategorija grafički se prikazuje linijom koja povezuje generički entitet i podcrtani krug. Postoje dvije linije ispod kruga koje predstavljaju broj veza kategorija u skupu. Brojnost veza nije definirana jer je ona uvijek nula ili jedan. Entiteti kategorija su uvijek neovisni o identifikatoru. Generički entitet je neovisan o identifikatoru osim ako je njegov identifikator već prešao kroz neku drugu vezu.

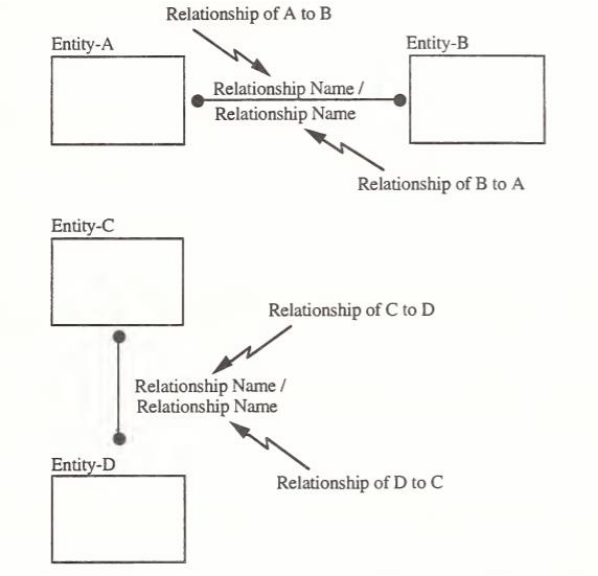


Slika 8 - Primjer veza kategorizacije, (Technology, 1993)

Ukoliko je krug dvostruko podcrtan to znači da je skupina kategorija potpuna, dok jedna crta predstavlja nepotpunu skupinu. Svaki generički entitet i svaka kategorija mora imati isti primarni ključ. (National Institute of Standards and Technology, 1993.).

Nespecifične veze

Nespecifične veze (engl. *non-specific relationship*)se koriste u visokim razinama pogleda ER modela kako bi prikazale *many-to-many* veze između entiteta. Ona je veza između dva entiteta u kojoj svaka instanca prvog entiteta je povezana sa niti jednom, jednom ili više instanci drugog entiteta i obratno. Kasnije se ona može definirati brojnošću iz oba smjera veze. Grafički se prikazuje pomoću linije koja na oba kraja ima točku kao što je prikazano slikom 9. Njena brojnost upisuje se pored točke simbolom „P“ za jedan ili više i simbolom „Z“ za nula ili jedan. Za naziv veze koriste se glagoli ili glagolske fraze, te se veza može imenovati u oba smjera gdje se pritom koristi simbol „/“. Ukoliko je pozicija entiteta horizontalna onda se naziv čita s lijeva na desno, a ukoliko je entitet u vertikalnoj poziciji onda se naziv čita od gore prema dole.



Slika 9 - Primjer nespecifične veze, (Technology, 1993)

Ključevi

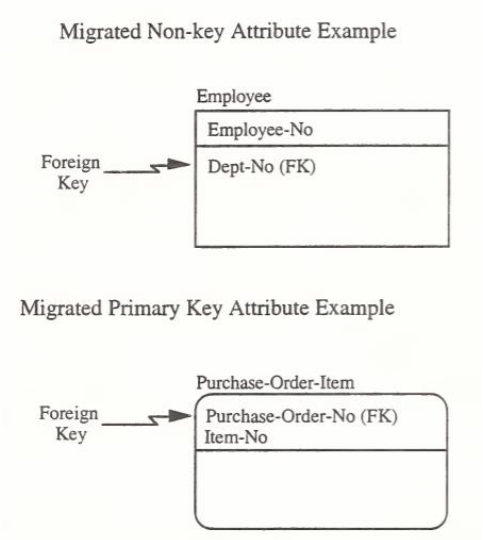
Primarni ključ

Primarni ključ (engl. *primary key*) predstavljaju ograničenja jedinstvenosti nad vrijednostima atributa entiteta. Kandidat za ključ (engl. *candidate key*) čine jedan ili više atributa čije vrijednosti jedinstveno identificiraju svaki instancu entiteta. Ukoliko postoji više kandidata za ključ, biramo jednoga za primarni ključ a ostale kandidate za ključ nazivamo **zamjenskim ključevima** (engl. *alternate key*). Ako postoji samo jedan kandidat za ključ, tada upravo on postaje primarni ključ. Ovakva definicija ključeva jednaka je definiciji ključeva kod EV metode.

Grafički primarni ključ predstavlja atribut koji je zapisan na vrhu liste atributa u entitetu i odvojen horizontalnom linijom od drugih atributa. Svakom zamjenskom ključu se dodjeljuje jedinstveni cijeli broj koji se smješta unutar zagrada pored simbola „AK“ desno od atributa. Na primjer: „(AK1)“ . Pojedini atribut može biti identificiran kao dio ili više od jednog zamjenskog ključa. Svaki entitet mora imati samo jedan primarni ključ i bilo koji broj zamjenskih ključeva. Ključevi se mogu sastojati od jednog atributa ili kombinacije više atributa. Primarni i zamjenski ključevi moraju sadržavati samo one atribute koji pridonose jedinstvenoj identifikaciji, to pravilo se naziva i pravilo najmanjeg ključa (engl. *Smallest-Key Rule*). **Pravilo potpune funkcijske ovisnosti** (engl. *Full-Functional-Dependency Rule*) govori ako je primarni ključ napravljen od više atributa, svi ne-ključni atributi moraju funkcionalno ovisiti o cijelom primarnom ključu. **Pravilo o neprelaznoj ovisnosti** (engl. *No-Transitive-Dependency Rule*) govori da svaki atribut koji nije dio primarnog ili zamjenskog ključa mora funkcionalno ovisiti samo ključnim atributima. (National Institute of Standards and Technology, 1993.).

Vanjski ključ

Vanjski ključ čine atributi u entitetima koji određuju instance povezanih entiteta. Ukoliko između dva entiteta postoji specijalna veza ili veza kategorije onda atributi koji tvore primarni ključ entiteta roditelj ili generičkog entiteta prelaze kao atributi entiteta dijete ili kategorije. Vanjski ključ može biti dio ili cijeli primarni ključ, zamjenski ključ ili ne ključni atribut entiteta. Ako su svi atributi primarnog ključa entiteta roditelj postali primarni ključevi entiteta dijete, onda tu vezu smatramo identifikacijskom. Ako je neki od prelaznih atributa nije dio primarnog ključa entiteta dijete onda tu vezu nazivamo ne identifikacijskom vezom. Kod veze kategorizacije generički entitet i entiteti kategorije predstavljaju istu stvar u stvarnom svijetu, stoga se primarni ključ entiteta svih entiteta kategorija nasljeđuje od primarnog ključa generičkog entiteta. Vanjski ključ se grafički prikazuje postavljanjem naziva prelaznih atributa unutar entiteta i zapisivanjem „(FK)“ pored njega kao što je prikazano na slici 10. Ukoliko su svi prijelazni atributi dio primarnog ključa entiteta dijete onda se svi zapisuju iznad horizontalne linije, a kutovi entiteta se zaobljavaju kako bi se naglasilo da identifikator entiteta dijete ovisi o primarnom ključu koji je naslijeđen. Kada prijelazni atribut ne pripada primarnom ključu entiteta dijete onda ga smještamo ispod horizontalne linije, a oblik entiteta može biti pravokutan ili zaobljenih vrhova ovisno o prisutnosti identifikacijskih veza entiteta. Prijelazni atributi mogu činiti dio zamjenskog ključa. (National Institute of Standards and Technology, 1993.).



Slika 10 - Primjer vanjskog ključa, (Technology, 1993)

Usporedba koncepata metode entiteti-veze metodologije MIRIS i metode IDEF1X metodologije IDEF

Entitet

Entitet u metodi entiteti-veze i metodi IDEF1X su slični pojmovi. U obje metode entiteti se karakteriziraju kao stvarne stvari ili apstraktni pojmovi. Kod obje metode pojedinačno pojavljivanje entiteta nazivamo instancom entiteta. Kod EV metode grupiranjem entiteta sličnih svojstava procesom klasifikacije dobivamo tipove entiteta. IDEF1X metoda ima dvije vrste entiteta, oni entiteti koji su ovisni o identifikatoru i oni koji nisu ovisni o identifikatoru. U obje metode za grafički prikaz entiteta koristi se pravokutnik, no kod IDEF1X kutovi pravokutnika će biti zaobljeni ako su njegovi entiteti ovisni o identifikatoru. Naziv u metodi EV biti će smješten unutar pravokutnika, a kod IDEF1X naziv će biti iznad pravokutnika. Za nazive prilikom imenovanja entiteta u ovim metodama koristimo imenice u jednini. Zadani naziv mora biti smislen i dosljedan kroz cijeli model podataka. Kod metode EV, pa tako i kod metode IDEF1X entitet može imati nebrojeno veza sa drugim entitetima u dijagramu.

Tip vrijednosti i domena

Tip vrijednosti u EV metodi i domena u IDEF1X metodi su slični. Obje metode smatraju vrijednošću podatak koji sadrži informaciju o entitetu. Kod metode EV tip vrijednosti podatka čini skup pojedinačnih vrijednosti. Tip vrijednosti se u ovom slučaju grafički prikazuje ovalom. Također, u IDEF1X metodi, domenu možemo tumačiti kao skup vrijednosti. Razlika je u tome što domenu u IDEF1X metodi možemo podijeliti na osnovne i zapisane domene. U osnovnoj domeni se mogu definirati pravila raspona i pravilo liste vrijednosti, slično kao i ograničenja na dopuštene vrijednosti u metodi EV. IDEF1X metoda sadrži generalizacijsku hijerarhiju, gdje podtipovi domena u hijerarhiji se međusobno ne isključuju, što nije slučaj kod EV metode.

Atributi

Atributi u obje metode predstavljaju neko svojstvo ili karakteristiku određenog entiteta. Atribut je funkcija koja pridružuje entitetu vrijednost iz tipa vrijednosti tj. domene. U metodi EV on se grafički prikazuje unutar ovala, a kod metode IDEF1X prikazuje se pomoću liste unutar pravokutnika (entiteta). Atributi u obje metode moraju biti jedinstveni kako bi mogli biti kandidati za ključ i naposlijetku tvoriti primarni ključ. Atributi u IDEF1X metodi koji čine primarni ključ su odvojeni horizontalnom linijom od ostalih atributa, a kod EV metode atributi primarnog ključa su podcrtani ili podebljani i zapisani su prije ostalih atributa. Za imenovanje atributa u obje metode moraju biti korištene imenice u jednini.

Veza

Veza kategorizacije u metodi IDEF1X slična je generalizaciji u metodi EV. Veza kategorije predstavlja strukture u kojima je jedan entitet kategorija nekog drugog entiteta. Ona je veza između jednog entiteta zvanog „generički entitet“ i drugog entiteta „skup kategorija“. Instanca jednog generičkog entiteta može biti u vezi sa samo jednom instancom entiteta iz skupa kategorija i zato se entiteti različitih kategorija međusobno isključuju. Skupina kategorija se grafički prikazuje linijom i podcrtanim krugom. Generalizacijskim tipom veze u metodi EV predstavlja se odnos između nadtipa i podtipova entiteta. Ovakav tip veze skupa sa skupom tipova entiteta nazivamo generalizacijsko stablo. Kod generalizacije vrijedi pravilo da tipovi entiteta podtipa nasljeđuju osobine i operacije entiteta nadtipa, ali ne i obratno. Grafički ju prikazujemo rombom i simbolom „G“.

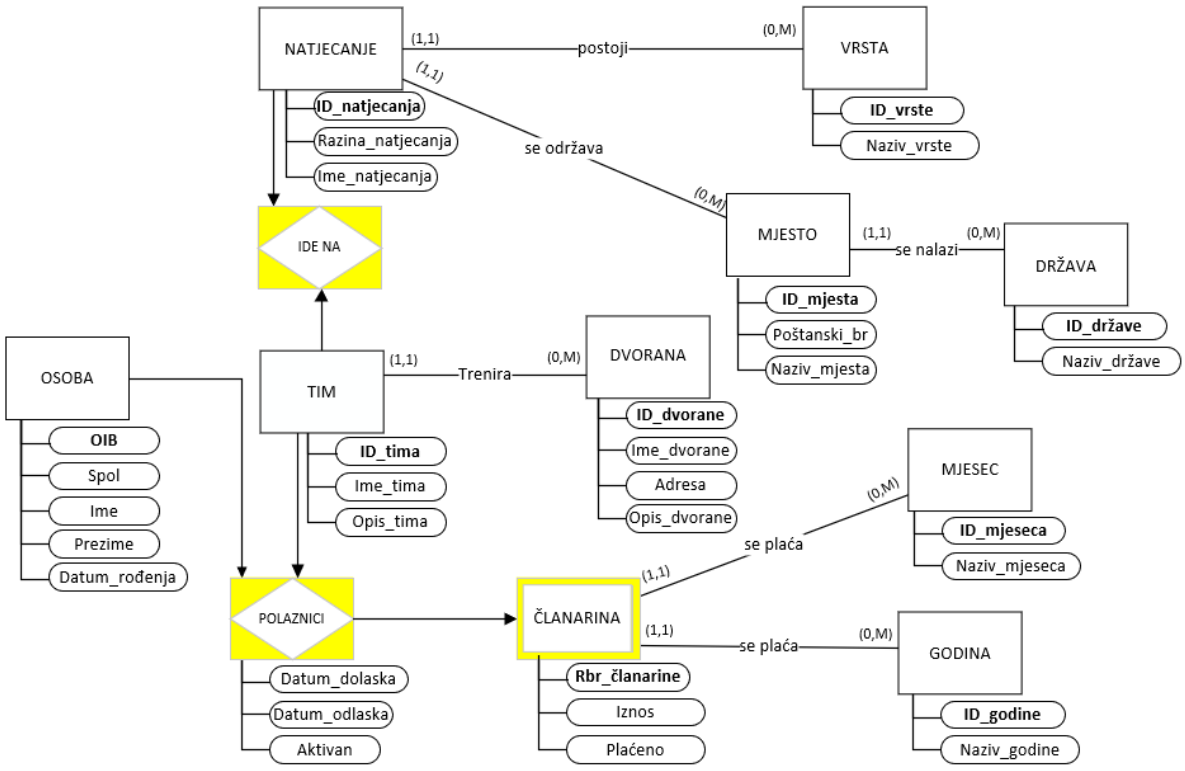
Agregacija je veza iz metode EV metodologije MIRIS koja se stvara kada se utvrdi da je brojnost tipa veze između entiteta sa obje strane (M,M). Agregacija može imati svoje atribute te je ona najsličnija nespecifičnoj vezi iz metode IDEF1X koja je veza između dva entiteta u kojoj svaka instanca prvog entiteta može biti povezana sa niti jednom, jednom ili više instanci drugog entiteta te se može stvoriti novi entitet koji sadrži svoje atribute i ovisi o dva entiteta s kojima je povezan.

Identifikacijsku vezu kod IDEF1X metode možemo usporediti sa specijalnom vezom između jakog i slabog tipa entiteta iz metode EV. U Identifikacijskoj metodi je svaka instanca entiteta dijete povezana sa točno jednom instancom entiteta roditelj i ona egzistencijalno ovisi o njemu. Primarni ključ entiteta roditelj biti će dio primarnog ključa entiteta dijete. Grafički se uvijek označava punom linijom između entiteta roditelj i entiteta dijete gdje se nalazi i točka. Slično kao kod EV modela slabi tip entiteta (entitet dijete) egzistencijalno je ovisan o tipu entiteta roditelj i preuzima njegov primarni ključ. Grafički ga možemo prikazati poput obične veze sa strelicom prema jakom tipu entiteta (entitetu roditelj).

Ključevi

Kandidate za ključ u IDEF1X metodi čine atributi čije su vrijednosti jedinstveno identificirane. Ukoliko postoji više kandidata za ključ, biramo jednoga koji najviše odgovara entitetu i imenujemo ga primarnim ključem, ista stvar se događa u metodi EV. Preostale kandidate za ključ u IDEF1X metodi nazivamo zamjenskim ključevima. Grafički se primarni ključu IDEF1X prikazuje atributom zapisanim na vrhu liste atributa u entitetu i crtom koja služi za odvajanje od ostalih atributa. Kod metode EV primarni ključ je podcrtan ili podebljan i zapisan prvi u nizu atributa nekog entiteta. Ukoliko se radi o vanjskim ključevima u IDEF1X metodi, oni se označavaju simbolom „FK“ pored naziva atributa. Ukoliko postoji specijalna veza ili veza kategorije između dva entiteta onda atributi koji tvore primarni ključ entiteta roditelj prelaze kao atributi entiteta dijete. U modelu podataka kod metode EV, vanjski ključevi se ne prikazuju.

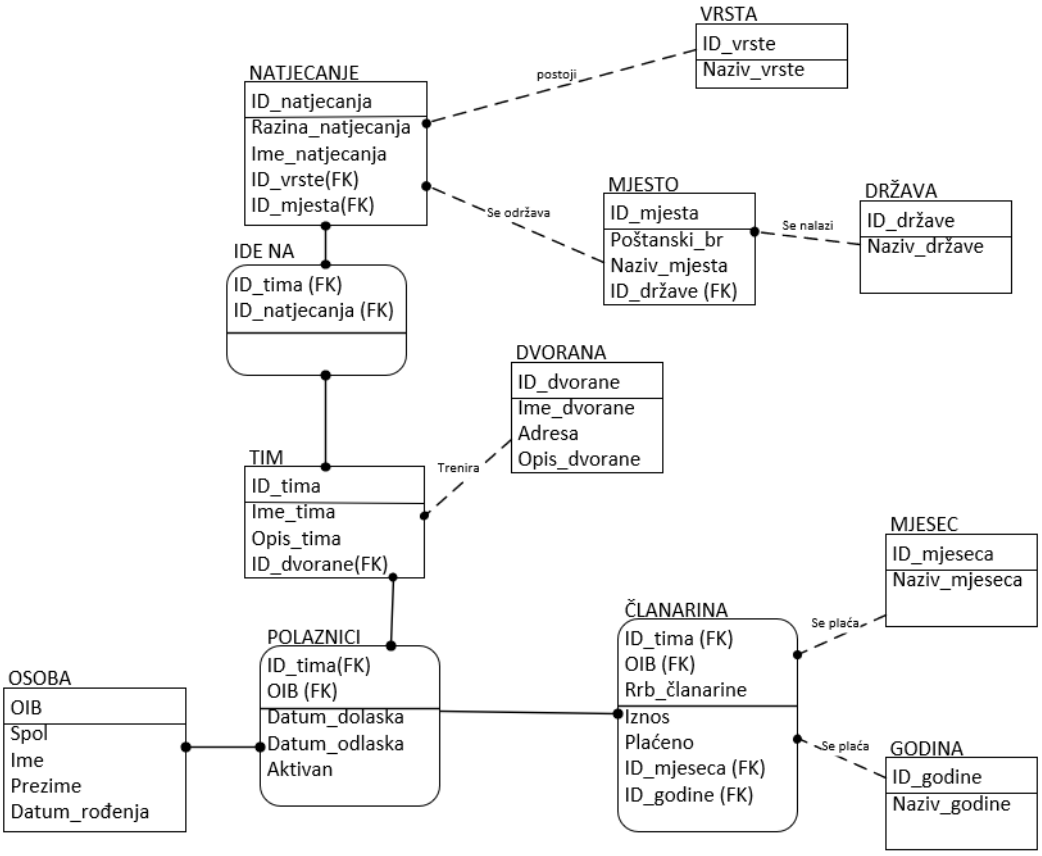
Vlastiti primjer EV modela



Slika 11 - DEV vlastiti primjer funkcioniranja tima

Slika 11 prikazuje model entiteti – veze koji opisuje jedan primjer funkcioniranja sportskog tima. Tip entiteta od kojeg se kreće je *Tim*, on je jaki tip entiteta. Uz tim jaki tip entiteta čine entiteti *Osoba*, *Natjecanje, Dvorana, Godina, Mjesec, Vrsta, Mjesto, Država*. Iz modela je vidljivo da entitetu *Tim* moramo pridružiti entitet *Osobe* jer se svaki tim sastoji od nekoliko osoba i zbog brojnosti veze (M,M) između ta dva entiteta dobiti ćemo agregaciju *Polaznici*. Slabi tip entiteta je *Članarina* i ona egzistencijalno ovisi o agregaciji *Polaznici* jer ukoliko nema polaznika nema ni plaćanja članarine. Kako bi postojala veza između tipa entiteta *Tim* i tipa entiteta *Natjecanje* (M,M) između njih se također uvodi agregacija pod nazivom *Ide na*. Kako bi znali za koji mjesec i za koju godinu se plaća članarina entiteti *Mjesec* i *Godina* su povezani sa slabim tipom entiteta *Članarina.* Entitet *Tim* povezujemo sa entitetom *Dvorana* kako bi se znalo gdje koji tim trenira. Entitet *Natjecanje* je povezano s entitetom *Vrsta* kako bi znali je li to natjecanje školsko, županijsko ili državno. Sa entitetom *Natjecanje* još moramo povezati entitet *Mjesto* kako bi znali u kojem se mjestu natjecanje održava. Posljednji entitet *Država* povezujemo sa entitetom *Mjesto* kako bi znali u kojoj se državi mjesto nalazi. Kod modela EV ne prikazuju se vanjski ključevi.

Vlastiti primjer IDEF1X modela



Slika 12 - IDEF1X vlastiti primjer funkcioniranja tima

Model IDEF1X na slici 12 se razlikuje od modela entiteti – veze u grafičkom prikazu koncepata. Na primjer: kod EV modela entitet *Tim* se označavao pravokutnikom proizvoljne veličine i njegov primarni ključ (atribut jedinstvene identifikacije) je podebljan, dok je ovdje entitet označen pravokutnikom kojemu je primarni ključ (atribut jedinstvene identifikacije) odvojen horizontalnom od ostalih atributa. Isto vrijedi za ostale jake tipove entiteta poput O*soba*, G*odina*, M*jesec*, *Dvorana*, *Natjecanje*, *Mjesto*, *Država, Vrsta*. Sljedeća razlika koju uočavamo je kod slabog tipa entiteta *Članarina*. Ona je u EV modelu označena dvostrukim pravokutnikom, dok je u IDEF1X modelu prikazana pravokutnikom sa zaobljenim rubovima koji horizontalnom linijom odvaja primarne ključeve entiteta *Članarina,* što vlastite, što one koje je naslijedio od entiteta roditelj *Polaznici*. Agregacija u EV modelu *Polaznici* slična je nespecifičnom tipu veze koja povezuje dva entiteta *Tim* i *Osoba* (M,M) vezom, te nam daje mogućnost stvaranja novog entiteta koji sadrži svoje atribute, njihove primarne ključeve i ovisi o oba entiteta s kojima je povezan. Ista stvar se događa kod nespecifične veze IDEF1X modela *Ide na.* Grafički se takva veza prikazuje pomoću linije koja na oba kraja ima točku. Vezu između entiteta *Natjecanje* i entiteta *Mjesto* nazivamo neidentifikacijskom jer se sve instance entiteta mogu odrediti bez povezanosti sa roditeljskim entitetom. Takva veza označava se isprekidanom linijom sa točkom na kraju.

Zaključak

Modeliranje podataka pojavljuje se kroz sve faze izgradnje informacijskog sustava. Kako bi sustav pravilno funkcionirao, sadržavao što manje grešaka i imao veću mogućnost za uspjeh njegov model podataka mora biti detaljno i točno razrađen. Kroz vrijeme se razvilo mnogo metodologija, u ovom radu izdvojene su MIRIS i IDEF metodologije. Njihove metode pokazale su se prikladnima za modeliranje, jer dizajneru daju mogućnost detaljne obrade podataka i sa svojim ograničenjima kontroliraju netočnost i neredundantnosti podataka u samom modelu. Metoda entiteti – veze metodologije MIRIS i IDEF1X metoda metodologije IDEF imaju slične koncepte poput entiteta, atributa, ključeva pa i nekih veza. Kako bi se napravio dobar model za neki sustav potrebno je dobro poznavati metodu s kojom se radi. Upravo zbog toga metodologija MIRIS i njena metoda entiteti – veze ima prednost kao jednostavnija i razumljivija metoda kod razvoja informacijskih sustava jer se od početka projekta dizajner upoznaje sa sustavom i modeliranje podataka tog sustava postaje jednostavnije. Radi toga niti ne čudi činjenica da je model entiteti – veze metodologije MIRIS jedan od najkorištenijih modela u praksi. Nešto zahtjevnija metoda IDEF1X zahtjeva jako dobro poznavanje svih koncepata i pravila kako bi se modelirao kvalitetan sustav. Usprkos tome ona svakodnevno pronalazi svoje mjesto u svijetu izgradnje informacijskih sustava i modeliranju podataka.

Literatura

Pavlić, Mile (2011.) *Oblikovanje baza podataka.* Rijeka: Odjel za informatiku Sveučilišta u Rijeci. Rijeka

Pavlić, Mile (2009.) *Informacijski sustavi.* Rijeka

Varga, Mladen (2020.) *Baze podataka: konceptualno, logičko i fizičko modeliranje podataka*, Udžbenik Sveučilišta u Zagrebu. Vlastita naknada, Zagreb

Halpin T., Morgan T. (2008.) *Information Modeling and Relation Databases*, Morgan Kaufmann; 2nd edition

IDEF – Object Oriented Design Method, Knowledge Based Systems, preuzeto 30. 8. 2021 s https://www.idef.com/

IDEF0 – Object Oriented Design Method, Knowledge Based Systems, preuzeto 30. 8. 2021 s https://www.idef.com/idefo-function_modeling_method/

IDEF1 – Object Oriented Design Method, Knowledge Based Systems, preuzeto 30. 8. 2021 s https://www.idef.com/idef1-information_modeling_method/

IDEF1X – Object Oriented Design Method, Knowledge Based Systems, preuzeto 30. 8. 2021 s https://www.idef.com/idef1x-data-modeling-method/

IDEF3 – Object Oriented Design Method, Knowledge Based Systems, preuzeto 30. 8. 2021 s https://www.idef.com/idef3-process-description-capture-method/

IDEF4 – Object Oriented Design Method, Knowledge Based Systems, preuzeto 30. 8. 2021 s https://www.idef.com/idef4-object-oriented-design-method/

IDEF4 – Object Oriented Design Method, Knowledge Based Systems, preuzeto 30. 8. 2021 s https://www.idef.com/idef5-ontology-description-capture-method/

IDEF – Integration Definition, preuzeto 28. 8. 2021. s https://en.wikipedia.org/wiki/IDEF

Federal information processing standards publication, INTEGRATION DEFINITION FOR INFORMATION MODELING (IDEF1X). Computer Systems Laboratory National Institute of Standards and Technology, 21. prosinca 1993.

Popis slika