SVEUČILIŠTE U ZAGREBU

FILOZOFSKI FAKULTET

ODSJEK ZA INFORMACIJSKE I KOMUNIKACIJSKE ZNANOSTI

Ak. god. 2016/2017.

Domagoj Vočanec

Izrada arhivskog informacijskog sustava Saveza društava Naša djeca Hrvatske

diplomski rad

Mentor: prof. dr. sc. Krešimir Pavlina

Zagreb, 2017.

Sadržaj

Uvod

Arhivi su vrlo bitna karika u upravljanju životnim ciklusom informacija. Prihvaćaju zapise, bilježe ih i pružaju informacije korisnicima u željenom obliku i na željeno mjesto, uz pomoć današnjih tehnoloških rješenja. Tehnologija je moderni pokretač potražnje. Zastarjeli sustavi, velike količine podataka, sustavi za upravljanje, hiperprodukcija i nekontrolirani pristup informacijama veliki su izazovi za moderne arhive, ali također i za organizacije koje stvaraju gradivo. Oni koji prepoznaju te izazove i poduzmu mjere orijentirane prema budućnosti, pretvoriti će nastalu situaciju u svoju korist i stvoriti temelj za budući uspjeh. Savez društava Naša djeca Hrvatske na početku je toga puta. Riječ je o neprofitnoj organizaciji koja je stvaratelj i imatelj privatnoga gradiva. Budući da u organizaciji radi svega nekoliko stalnih zaposlenika arhiv, odnosno pismohrana nije pravilno ustrojena i kao posljedica gradivo je desetljećima sustavno zanemarivano. Situacija se počela mijenjati kada je Hrvatski državni arhiv proglasio gradivo Saveza od državnoga značaja pa je nedugo zatim uspostavljen program s ciljem zaštite i očuvanja arhivskoga gradiva. U međuvremenu se počeo provoditi postupak sređivanja gradiva i može se reći da je arhivsko gradivo Saveza većim dijelom dovedeno u stanje sređenosti. Sljedeći korak koji je potrebno napraviti je početi uvoditi tehnološka rješenja, odnosno izraditi arhivski informacijski sustav. Riječ je o manjem sustavu koji se sastoji od baze podataka i korisničkog sučelja koje osim što prikazuje informacije o arhivskome gradivu, također pruža korištenje funkcionalnosti za upravljanje podacima. Cilj ovog diplomskog rada upravo je prikazati postupak izrade arhivskog informacijskog sustava Saveza. Sadržaj rada podijeljen je na tri osnovne cjeline. U prvom poglavlju biti će prikazane definicije i temeljna obilježja baze podataka te opisan logički model podataka. Drugo poglavlje posvećeno je programskim jezicima SQL i C#. Posljednje poglavlje prikazuje program zaštite i očuvanja arhivskoga gradiva Saveza društava Naša djeca Hrvatske, međunarodnu arhivističku normu ISAD(G) te praktični dio programiranja baze podataka i izrade korisničkog sučelja.

1. Baza podataka

1.1 Definiranje baze podataka

U osnovnom značenju baza podataka je organizirana zbirka podataka.[1] Složenije, bazu podataka se definira kao dijeljenu i integriranu računalnu strukturu koja pohranjuje skup podataka za krajnje korisnike, odnosno sirove činjenice koje su predmet zanimanja krajnjih korisnika te metapodatke, odnosno podatke o podacima pomoću kojih su podaci za krajnje korisnike integrirani i upravljani.[2] Bazu podataka možemo definirati i kao skup međusobno povezanih podataka, pri čemu su podaci poznate činjenice koje mogu biti zabilježene i koje imaju implicitno značenje.[3]

Navedene definicije su općenite i ne predočuju u potpunosti složenost termina baza podataka. Uobičajeno korištenje termina baza podataka je više ograničeno i sadrži sljedeća implicitna svojstva:[4]

- baza podataka predstavlja određeni aspekt stvarnog svijeta, ponekad zvan mikrokozmos (engl. *miniworld*)
- baza podataka je logički dosljedan skup podataka sa trajnim značenjem; slučajni odabir podataka ne može se smatrati bazom podataka
- baza podataka je projektirana i izgrađena s određenom svrhom; posjeduje ciljanu skupinu korisnika i pojedine unaprijed odabrane aplikacije koje su tim korisnicima potrebne

Drugim riječima, baza podataka preuzima podatke iz nekog drugog izvora, provodi određeni stupanj interakcije s događajima u stvarnom svijetu i ima publiku koja je aktivno zainteresirana za njen sadržaj. Promjenu informacija u bazi podataka mogu izazvati krajnji korisnici obavljanjem transakcija (npr. klijent kupuje kameru) ili nekim važnim događajem (npr. zaposlenik je dobio dijete). Da bi baza podataka bila točna i pouzdana u svakom trenutku, mora biti odraz mikrokozmosa koji predstavlja. Stoga, promjene u bazi podataka moraju biti vidljive u najkraćem mogućem roku.

"Centar za online baze podataka - online priručnik," 11, accessed September 21, 2016, http://onlinebaze.irb.hr/prirucnik.[1] "Centar za online baze podataka - online priručnik," accessed September 21, 2016, http://onlinebaze.irb.hr/prirucnik.

[2] Carlos Coronel, Steven Morris, and Peter Rob, *Database Systems: Design, Implementation and Management*, 9 edition (Australia; United States: Cengage Learning, 2009), 7.

[3] Ramez Elmasri and Shamkant B. Navathe, *Fundamentals of Database Systems*, 7 edition (Hoboken, NJ: Pearson, 2015), 4.

[4] Ibid., 5.

Većina baza podataka u današnje vrijeme je izgrađena i održavana u elektroničkom obliku, pomoću posebno razvijenih programskih aplikacija ili pomoću cijelog sustava za upravljanje bazama podataka (*Database Management System*, DBMS).

DBMS možemo definirati kao skup programa koji upravljaju strukturom baze podataka i kontroliraju pristup podacima koji su pohranjeni u bazi.[5] Posrednik je između podataka i korisnika te olakšava procese definiranja, izgradnje, manipulacije i dijeljenja baza podataka među različitim korisnicima i aplikacijama.[6] Definiranje baze podataka podrazumijeva specificiranje vrste podataka, strukture i ograničenja podataka koji se pohranjuju u bazu podataka. Definicija baze podataka ili opisne informacije su također pohranjene u obliku kataloga ili rječnika, koji su ispunjeni metapodacima. Izgradnja baze podataka je proces pohrane podataka na određeni medij za pohranu koji je kontroliran od strane DBMS-a. Manipulacija baze podataka uključuje funkcije poput slanja upita bazi podataka kako bi se dohvatili željeni podaci, ažuriranja baze kako bi se odrazile promjene unutar mikrokozmosa te generiranja izvješća iz podataka. Dijeljenje baze podataka omogućuje istovremeni pristup većem broju korisnika i programa.

Važne usluge koje DBMS pruža su i zaštita i održavanje baze podataka kroz duži vremenski period.[7] Zaštita podrazumijeva zaštitu sustava od neispravnosti hardvera i softvera, kao i sigurnosnu zaštitu od neovlaštenog ili zlonamjernog pristupa. Životni ciklus karakteristične velike baze podataka može trajati dugi niz godina, tako da DBMS mora biti u stanju održavati sustav baze podataka na način da omogući daljnji razvoj sustava unatoč promjenama zahtjeva tijekom vremena.

Ono što je još bitno za istaknuti jest postojanje termina koji objedinjuje pojmove baze podataka i sustava za upravljanje bazama podataka, a to je sustav baze podataka.[8] Termin se odnosi na organizaciju komponenti koje definiraju i reguliraju prikupljanje, pohranu, upravljanje i korištenje podataka unutar okruženja baze podataka. Sustav se sastoji od pet glavnih komponenti: hardver, softver, ljudi, postupci i podaci.[9] Sustav baze podataka možemo poistovjetiti sa elektroničkim sustavom za upravljanje zapisima. Drugim riječima, to je sustav čija cjelokupna svrha je pohrana informacija te omogućiti korisnicima dohvaćanje i ažuriranje

[5] Coronel, Morris, and Rob, *Database Systems*, 7.
[6] Elmasri and Navathe, *Fundamentals of Database Systems*, 6.
[7] Ibid.
[8] Ibid.
[9] Coronel, Morris, and Rob, *Database Systems*, 18.

tih informacija.[10] Bitno je napomenuti da se termini poistovjećuju pa je danas uobičajeno koristiti termin baza podataka kao univerzalni naziv koji se koristi za cjelokupni sustav baze podataka.

1.2 Temeljna obilježja baze podataka

Kada uspoređujemo sustav baza podataka sa tradicionalnim papirnatim pristupom upravljanja zapisima, razlike među sustavima jasno su vidljive. Prednosti korištenja baza podataka su:[11]

- *kompaktnost* – ne postoji potreba za potencijalno opsežnom tiskanom dokumentacijom
- *brzina* – računalo može dohvatiti i ažurirati podatke znatno brže nego čovjek.
- *manje napora* – većina monotonog dijela ručnog održavanja dokumenata je eliminirana; mehaničke zadatke uvijek bolje odradi računalo
- *ažurnost* – precizne, aktualne informacije dostupne su u svakom trenutku
- *zaštita* – podaci se mogu bolje zaštititi od nehotičnog gubitka ili protuzakonitog pristupa

Navedene prednosti se još više ističu u sustavu s više korisnika, gdje će baza podataka biti puno veća i kompleksnija nego kod sustava sa jednim korisnikom. Prednost koja se posebno ističe kod većih sustava je da sustav baze podataka omogućuje poduzeću centralizirani nadzor podataka.[12] Potpuno suprotna situacija je kod poduzeća koja ne koriste sustav baze podataka, gdje uobičajeno svaka aplikacija ima svoje vlastite datoteke pa su podaci raspršeni i teško ih je kontrolirati na sistematičan način.

Upravo usporedbom sa starijim pristupom programiranja prilagođenih aplikacija za pristup pohranjenim podacima, ističu se glavna obilježja korištenja baza podataka. Pri tradicionalnoj obradi podataka svaki korisnik definira i koristi vlastite datoteke unutar sustava. To dovodi do pojave redundantnosti podataka što rezultira uzaludnim trošenjem prostora za pohranu i suvišnim nastojanjima da se održi aktualnost podataka. S druge strane, u sustavu baze podataka jedan repozitorij održava podatke koji su prvo jedinstveno definirani, a zatim je omogućen pristup različitim korisnicima kroz upite, transakcije i aplikacije. Glavna obilježja koja se ističu i koja će biti posebno opisana u sljedećim odjeljcima su:[13]

- samo-opisujuća priroda sustava baze podataka

[10] C. J. Date, *An Introduction to Database Systems*, 8 edition (Boston: Pearson, 2003), 6.
[11] Ibid., 17.
[12] Ibid.
[13] Elmasri and Navathe, *Fundamentals of Database Systems*, 10.

- razdvajanje aplikacija i podataka, kao i apstrakcija podataka
- podrška za višestruke prikaze podataka
- dijeljenje podataka i obrada transakcija većeg broja korisnika

Temeljno obilježje pristupa izgradnji baze podataka je da sustav baze podataka, uz samu bazu podataka, sadrži i njen potpun opis strukture i ograničenja. Opis je pohranjen u DBMS katalogu, koji sadrži informacije poput strukture svake podatkovne datoteke, vrstu i format pohrane svakog podatka te razna ograničenja koja se primjenjuju na podatke. Informacije pohranjene u katalogu su zapravo metapodaci koji opisuju strukturu prvobitne baze podataka. Treba napomenuti da određene novije vrste sustava baze podataka, poznatiji kao NOSQL sustavi, ne zahtijevaju korištenje metapodataka, već se podaci pohranjuju kao samoopisujući te uključuju nazive i vrijednosti podataka u jednoj zajedničkoj strukturi.

Kad je riječ o tradicionalnoj obradi podataka, struktura podatkovnih datoteka je ugrađena u aplikacije. Ako dođe do bilo kakve promjene strukture datoteke, velika je vjerojatnost da će se morati mijenjati sve aplikacije koje imaju pristup datoteci. Suprotno tome, DBMS aplikacije ne zahtijevaju takve promjene. Struktura podatkovnih datoteka u DBMS katalogu pohranjena je odvojeno od aplikacija. To svojstvo nazivamo neovisnost između aplikacija i podataka (engl. *program-data independence*).[14]

Obilježje koje omogućuje neovisnost između aplikacija i podataka je apstrakcija podataka. DBMS korisnicima pruža konceptualnu reprezentaciju podataka koja ne sadrži mnoge pojedinosti o tome kako se podaci pohranjuju ili kako se pojedini postupci implementiraju. Neformalno, model podataka je vrsta apstrakcije koja pruža konceptualni prikaz podataka. Model podataka koristi logičke koncepte, poput objekata, njihovih svojstva i međusobne veze, koje bi korisnicima trebali biti razumljiviji od koncepata računalne pohrane podataka. Dakle, model podataka sakriva pojedinosti o pohrani i implementaciji za koje većina korisnika nema interesa.[15]

Baza podataka obično ima različite vrste korisnika, gdje svaki korisnik može zahtijevati drugačiju perspektivu ili pogled na podatke u bazi. Pogled može biti dio same baze podataka ili može sadržavati virtualne podatke koji su izvedeni iz baze podataka, ali nisu eksplicitno pohranjeni.[16] Neki korisnici ne moraju biti svjesni jesu li podaci kojima pristupaju pohranjeni

[14] Ibid., 12.
[15] Ibid.
[16] Ibid., 13.

ili izvedeni. Višekorisnički DBMS, čiji korisnici imaju niz različitih aplikacija za korištenje, mora osigurati prostor za definiranje više različitih pogleda.

Višekorisnički DBMS, kao što i samo ime govori, većem broju korisnika mora dopustiti istovremeni pristup bazi podataka. To je ključno obilježje koje osigurava da podaci budu integrirani i održavani u jedinstvenoj bazi podataka. DBMS mora sadržavati softver za nadzor istovremenog pristupa kako bi omogućio da više korisnika koji pokušavaju ažurirati iste podatke, učine to na kontrolirani način sa ispravnim krajnjim rezultatom.[17] Temeljna uloga softvera višekorisničkog DBMS-a je osigurati ispravan i učinkovit rad istovremenih transakcija. Pojam transakcije postao je središnja točka za mnoge baze podataka. Definira se kao izvedbeni program ili proces koji uključuje jedan ili više pristupa bazi podataka, poput čitanja ili ažuriranja zapisa baze podataka. Svaka transakcija izvršiti će logički ispravan pristup bazi podataka ako je izvršen u cijelosti, bez smetnji drugih transakcija. DBMS mora primijeniti određena svojstva transakcije. Svojstvo izolacije osigurava da se svaka transakcija izvrši izolirana od drugih transakcija, unatoč tome što se stotine transakcija mogu izvršavati istovremeno. Svojstvo atomarnosti osigurava da se svi postupci unutar baze podataka izvršavaju ili u potpunosti ili uopće ne.[18]

Uz opisana glavna obilježja i prednosti sustava baza podataka u odnosu na tradicionalne pristupe, postoje dodatne pogodnosti koje treba istaknuti: održavanje integriteta baze podataka, ograničavanje neovlaštenog pristupa, izrada sigurnosne kopije i mogućnost oporavka sustava, pružanje više različitih korisničkih sučelja te skraćeno vrijeme razvoja aplikacija.[19]

Problem integriteta je problem osiguravanja točnosti podataka. Nedosljednost između dva ulaza koja tvrde da predstavljaju istu činjenicu primjer je nedostatka integriteta. Takav problem se može pojaviti samo ako su pohranjeni podaci redundantni. Međutim, baza podataka može sadržavati neispravne informacije i kada nema redundantnosti podataka. Na primjer, zaposleniku može biti zapisano da je umjesto 40 radnih sati odradio 400. Centralizirani nadzor podataka može pomoći u izbjegavanju ovakvog problema, dopuštajući administratorima da definiraju i implementiraju ograničenja koja osiguravaju integritet kao alat provjere podataka za vrijeme njihovog ažuriranja.[20]

[17] Ibid.
[18] Ibid., 14.
[19] Ibid., 19–22.
[20] Date, *An Introduction to Database Systems*, 2003, 19.

Kada veći broj korisnika dijeli veliku bazu podataka, vrlo je vjerojatno da većina korisnika neće imati pristup svim informacijama. Osim toga, nekim korisnicima jedina mogućnost je dohvaćanje podataka, dok je drugima dopušteno dohvaćanje i ažuriranje podataka. Dakle, vrsta pristupa također mora biti pod nadzorom. Uobičajen postupak je korisnicima omogućiti izradu računa zaštićenih lozinkom pomoću kojih pristupaju bazi podataka. DBMS ima obavezu osigurati podsustav za sigurnost i autorizaciju, kojeg administrator koristi za izradu korisničkih računa i određivanje njihovih ograničenja. Određena ograničenja DBMS mora provoditi automatski.[21]

DBMS pomoću podsustava za izradu sigurnosne kopije i mogućnosti oporavka osigurava prostor za oporavak od hardverskih i softverskih kvarova. Na primjer, ako računalni sustav padne za vrijeme kompleksne transakcije ažuriranja podataka, podsustav je odgovoran za vraćanje baze podataka u stanje prije početka izvršavanja transakcije. Izrada sigurnosne kopije diska također je potrebna u slučaju nemogućnosti popravka zbog nepovratnog kvara.[22]

Budući da baze podataka koriste vrste korisnika s različitim razinama tehničkog znanja, DBMS bi trebao pružiti raznovrsna korisnička sučelja. To uključuje aplikacije za mobilne korisnike, jezike temeljene na upitima za povremene korisnike, sučelja za programere aplikacija te sučelja sa izbornikom i na prirodnom jeziku za samostalne korisnike. Zajednički naziv za takva sučelja je grafička korisnička sučelja (g*raphical user interface,* GUI). Mnogi posebni jezici i okruženja postoje kako bi razvijali GUI. Treba napomenuti da su mogućnosti pružanja web GUI sučelja za baze podataka također uobičajena.[23]

Izvrsna prodajna značajka korištenja baza podataka je kratko vrijeme za razvoj nove aplikacije, kao npr. dohvaćanje određenih podataka iz baze podataka za potrebe tiskanja novog izvješća. Projektiranje i implementiranje višekorisničke baze podataka od nule može trajati duže od pisanja jedne specijalizirane aplikacije. No, kada se baza podataka uspostavi i normalno funkcionira, potrebno je znatno manje vremena za stvaranje novih aplikacija unutar DBMS-a. Vrijeme razvoja korištenjem DBMS sustava procjenjuje se na jednu šestinu do jedne četvrtine naspram običnog sustava datoteka.[24]

[21] Elmasri and Navathe, *Fundamentals of Database Systems,* 19.
[22] Ibid., 20.
[23] Ibid., 20–21.
[24] Ibid., 23.

1.3 Logički model baze podataka

Prilikom izrade baze podataka naglasak se stavlja na strukturu baze podataka i način na koji će se ta struktura koristiti za pohranu i upravljanje podacima krajnjih korisnika. Modeliranje podataka, prvi korak izrade baze podataka, odnosi se na proces stvaranja specifičnog modela podataka unutar određene domene (engl. *problem domain*). U ovom slučaju, domena je jasno definirano područje u realnom okruženju, s dobro definiranim opsegom i granicama. Model podataka se definira kao relativno jednostavan prikaz, obično grafički, složenijih struktura podataka u realnom okruženju. Općenito, model je apstrakcija od složenijeg realnog objekta ili događaja, a njegova glavna funkcija je pomoći u razumijevanju složenosti realnog okruženja.[25] Pravilno oblikovan, završni model podataka u stvari je nacrt svih uputa za izgradnju baze podataka koji će zadovoljiti sve zahtjeve krajnjih korisnika.

Modele podataka moguće je kategorizirati prema vrstama koncepata koje koriste za opis strukture baze podataka. Modeli visoke razine ili konceptualni modeli podataka pružaju koncepte koji su bliski načinu na koji mnogi korisnici doživljavaju podatke, a modeli niske razine ili fizički modeli podataka pružaju koncepte koji opisuju pojedinosti o tome kako su podaci pohranjeni na medijima. Koncepti fizičkog modela podataka općenito su namijenjeni računalnim stručnjacima, a ne krajnjim korisnicima. Između tih dviju krajnosti nalazi se reprezentacijski ili implementacijski model podataka, koji pruža korisnicima lako razumljive koncepte. Reprezentacijski modeli podataka skrivaju mnoge detalje načina na koji se podaci pohranjuju na disk, ali ih je moguće izravno implementirati na računalni sustav. Riječ je o modelima podataka koji se najčešće koriste u tradicionalnim DBMS-ovima. Najpoznatiji i najrašireniji danas je relacijski model podataka, koji je zamijenio mrežne i hijerarhijske modele podataka. U novije vrijeme javljaju se i objektni modeli podataka kao implementacijski modeli podataka visoke razine koji su srodniji konceptualnim modelima podataka te samoopisujući modeli podataka na temelju kojih spremišta podataka kombiniraju opis podataka s vrijednostima podataka. Najistaknutiji primjeri samoopisujućih modela podataka su XML i NOSQL sustavi.[26]

Kod konceptualnih modela podataka osnovni elementi su entiteti, atributi i veze među entitetima. Entitet se može definirati kao bilo što (osoba, mjesto, stvar ili događaj) o čemu želimo prikupljati i pohranjivati informacije. Entiteti mogu biti fizički objekti, npr. student ili učionica, i mogu biti apstraktni pojmovi, poput glazbenih koncerata ili kolegija. Složenije,

[25] Coronel, Morris, and Rob, *Database Systems*, 30.
[26] Elmasri and Navathe, *Fundamentals of Database Systems*, 33.

entitet je objekt koji možemo jednoznačno odrediti i na taj način izdvojiti iz okoline (odnosno prepoznati u skupu) te ga promatrati (prikupljati informacije o njemu). Atributi su svojstva koja opisuju entitet. Na primjer, entitet student se može opisati atributima ime, prezime, adresa stanovanja i JMBAG. Veza je koncept koji predstavlja neku interakciju među entitetima u sustavu odnosno predstavlja znanje o njihovoj povezanosti. Ovisna je o entitetima sustava i ne može postojati sama za sebe. Modeli podataka koriste sljedeća tri tipa veza:[27]

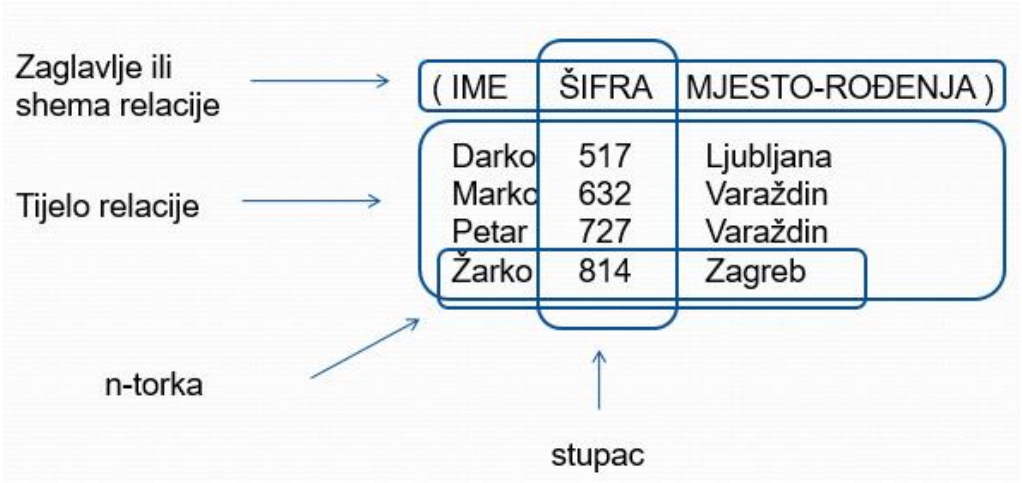
- **jedan-prema-jedan (1:1) –** jedan primjerak prvog tipa entiteta može biti u vezi s najviše jednim primjerkom drugog tipa entiteta i obrnuto; npr. veza *je pročelnik* između entiteta *nastavnik i zavod*
- **jedan-prema-mnogo (1:N)** – jedan primjerak prvog tipa entiteta može biti u vezi s više primjeraka drugog tipa entiteta, ali jedan primjerak drugog tipa može biti u vezi s najviše jednim primjerkom prvog tipa entiteta; npr. veza *predaje* između tipova entiteta *nastavnik* i *kolegij*
- **Mnogo-prema-mnogo (M:N)** - jedan primjerak prvog tipa entiteta može biti u vezi s više primjeraka drugog tipa entiteta, ali i jedan primjerak drugog tipa entiteta može biti u vezi s više primjeraka prvog tipa; npr. veza *upisao* između tipova entiteta *student* i *kolegij*

Složene aktivnosti izrade baze podataka zahtijevaju konceptualnu jednostavnost kako bi se dobili uspješni rezultati. Iako je relacijski model bio golem napredak u odnosu na mrežne i hijerarhijske modele, još uvijek nije bio učinkoviti alat za izradu baze podataka. Budući da je strukture lakše prikazivati grafički, a teže opisivanjem u tekstualnom obliku, dizajneri baza podataka radije koriste grafički alat pomoću kojeg se prikazuju entiteti i veze među njima. Prema tome, model entitet-veze (*Entity relationship (ER) model, ERM)* postao je široko prihvaćen standard za modeliranje podataka. ER modeli se prikazuju pomoću ER dijagrama, koji koriste grafički prikaz za modeliranje komponenti baze podataka. Peter Chen prvi put je predstavio ER model podataka 1976. godine, a model je vrlo brzo postao popularan jer nadopunjuje koncepte relacijskog modela podataka.[28] Udruživanjem relacijskog modela podataka i ERM-a razvijena je čvrsta i jasna struktura za izradu baze podataka.

[27] Coronel, Morris, and Rob, *Database Systems*, 31–32.
[28] Ibid., 38.

Relacijski model podataka predstavio je E. F. Codd 1970. godine u svom radu *Relacijski model podataka za velike dijeljene banke podataka*.[29] Relacijski model prikazuje bazu podataka kao skup relacija. Neformalno, svaka relacija se može gledati kao tablica vrijednosti, odnosno matrica ispresijecanih redaka i stupaca. Svaki redak u tablici predstavlja skup povezanih vrijednosti podataka. Redak predstavlja činjenicu koja u pravilu odgovara entitetu ili vezi iz realnog okruženja. Imena tablice i stupaca se koriste za interpretaciju značenja vrijednosti pojedinog retka. Na primjer, tablica je nazvana Student jer svaki redak prikazuje činjenice za pojedinačni entitet (osobu koja je student), a nazivi stupaca poput Ime, Prezime, JMBAG ili Kolegij, specificiraju kako interpretirati vrijednosti podataka u svakom retku.

Kada se koristi formalna terminologija za relacijski model podataka, tablica se naziva relacija, redak se naziva n-torka, a stupac se naziva atribut.[30] Atribut je element informacije kojim je jednoznačno određena vrsta svojstva. Atribut mora imati domenu i može imati samo jednu domenu. Domena predstavlja skup svih vrijednosti koje atribut može poprimiti. Konačan skup naziva atributa, zajedno sa skupom ograničenja koja su zadana na tom skupu atributa naziva se relacijska shema. Relacijska shema koristi se za opisivanje relacije. Relacija je konačan skup n-torki. Pojedina n-torka popis je *n* vrijednosti, a pojedina vrijednost mora biti element domene atributa ili nul vrijednost.[31] Nul vrijednost je koncept koji se koristi u slučaju kada vrijednost ne postoji ili je nepoznata. Primjer relacije prikazan je na slici 1.



Slika 1. Primjer relacije

[29] Ibid., 36.
[30] Elmasri and Navathe, *Fundamentals of Database Systems*, 151.
[31] Ibid., 152.

Relacijska baza podataka obično se sastoji od većeg broja relacija, čije n-torke su povezane na različite načine. Stanje cijele baze podataka odgovarati će stanjima svih njenih relacija u određenom trenutku u vremenu. Općenito, postoje mnoga ograničenja za stvarne vrijednosti koje utječu na stanje baze podataka. Ta ograničenja izvedena su iz pravila u mikrokozmosu koji baza podataka predstavlja.[32] U relacijskom modelu, ograničenja koja je potrebno posebno istaknuti su ključevi relacija. Ključevi su značajni jer se koriste kako bi se osiguralo da se svaki redak u relaciji može jedinstveno identificirati. Također, koriste se za uspostavljanje veza između relacija i osiguravaju integritet podataka.[33] Ključ relacije možemo definirati kao atribut ili skup atributa čije vrijednosti jednoznačno određuju neku n-torku. Vrijednost atributa ključa mora imati svojstva:[34]

1. *jednoznačnosti* – ne postoje dvije n-torke s jednakim vrijednostima svih atributa ključa u bilo kojem stanju relacije
2. *minimalnosti* –nemoguće je ukloniti bilo koji od atributa ključa i zadržati svojstvo jednoznačnosti; niti jedan pravi podskup od ključa nema svojstvo jednoznačnosti

Općenito, relacija može imati više ključeva. U tome slučaju svaki od ključeva se naziva kandidat za ključ. Uobičajeni je postupak da se jedan od kandidata odredi kao primarni ključ relacije. Riječ je o kandidatu za ključ čije vrijednosti se koriste za identifikaciju n-torki unutar relacije.[35] Kada relacija ima nekoliko mogućih kandidata za ključ, izbor primarnog ključa je proizvoljan, ali pritom je potrebno imati na umu da je bolje odabrati ključ koji će se sastojati od jednog ili manjeg broja atributa.

Slika 2. Odabir primarnog ključa

Pri izradi relacija često se uvodi polje surogat ključa, umjetno stvorenog atributa, koji se koristi kao primarni ključ (npr. atribut ID na slici 2). Kako polje surogat ključa ne predstavlja

[32] Ibid., 157.
[33] Coronel, Morris, and Rob, *Database Systems*, 62.
[34] Elmasri and Navathe, *Fundamentals of Database Systems*, 159.
[35] Ibid., 163.

određeno svojstvo entiteta iz stvarnog svijeta, moguće ga je u potpunosti kontrolirati i možemo biti sigurni da nikada neće doći do gubitka svojstva jednoznačnosti primarnoga ključa.

Uz navedene ključeve, koji se odnose na pojedinačnu relaciju, postoji i strani ključ koji povezuje dvije različite relacijske sheme. Riječ je o onome atributu čija je vrijednost primarni ključ u drugoj relaciji na koju pokazuje.[36] Strani ključ mora ispuniti dva osnovna pravila: atributi stranog ključa imaju istu domenu kao atributi primarnog ključa druge relacije i strani ključ može poprimiti vrijednost primarnog ključa druge relacije ili nul vrijednost.[37] Na slici 3 prikazane su dvije relacije, mjesto i osoba, koje su povezane putem primarnog ključa POŠT-BROJ i stranog ključa MJESTO. Posljednju n-torku relacije osoba nije moguće zapisati u bazu podataka jer je prema pravilima vrijednost stranog ključa neispravna.

Slika 3. Primjer stranog ključa

Kao što je ranije istaknuto, ključevi osiguravaju integritet podataka, odnosno njihovu istinitost ili ispravnost. Skup općih pravila integriteta sastoji se od dva pravila: entitetskog i referencijalnog integriteta.[38] Entitetski integritet ističe da primarni ključ mora biti unesen i ne smije biti nul vrijednost. Razlog tome je što se vrijednost primarnog ključa koristi za identifikaciju pojedinačnih n-torki u relaciji. Zapisivanje nul vrijednosti u tom slučaju bi značilo nemogućnost prepoznavanja određenog broja n-torki. S druge strane, referencijalni integritet vezan je uz strani ključ i koristi se za održavanje konzistentnosti između n-torki dviju relacija. Referencijalni integritet ističe da n-torka jedne relacije koja se referira na drugu relaciju mora se referirati na postojeću n-torku u toj relaciji. Drugim riječima, integritet se osigurava ako je ispunjeno pravilo da je strani ključ jedne relacije jednak primarnom ključu druge relacije ili je nul vrijednost.

[36] C. J. Date, *An Introduction to Database Systems*, 272.
[37] Elmasri and Navathe, *Fundamentals of Database Systems*, 163.
[38] Ibid.

2. Programski jezici

2.1 SQL

SQL (*Structured Query Language*) standardni je jezik za pristup i upravljanje relacijskim bazama podataka. Sastavljen je od naredbi koje omogućuju korisnicima izradu baza podataka i strukturiranje tablica, manipuliranje i administriranje podataka te postavljanje upita bazi podataka radi dohvaćanja korisnih informacija. SQL je razvio IBM 1974. godine, a standardizirali su ga zajedničkim snagama institucije ANSI (*American National Standards Institute*) i ISO (*International Standards Organization*).[39] Danas ga kao standardni jezik koriste svi komercijalni relacijski DBMS-ovi (npr. Oracle, MySQL, SQL Server, Microsoft Access).

Jezik je relativno lagano naučiti jer skup naredbi ima vokabular manji od 100 riječi. Njegova jednostavnost pojačana je činjenicom da veliki dio rada se samostalno odvija u pozadini. SQL jest deklarativan jezik jer njime pomoću upita opisujemo što želimo napraviti, a ne kako nešto želimo napraviti, već odluku o potrebnim procedurama za izvršavanje upita prepuštamo bazi podataka.[40] Za izdavanje neke od SQL naredbi, krajnji korisnici i programeri ne moraju poznavati fizički format za pohranu podataka ili kompleksne aktivnosti koje se odvijaju prilikom izvođenja SQL naredbe.

Iako je vrlo koristan i moćan alat za dohvaćanje i upravljanje podacima, SQL nije razvijen da djeluje kao samostalna aplikacija. Samostalno ne izrađuje izbornike, obrasce za posebna izvješća, pop-up prozore ili bilo kakve druge alate i obilježja koje krajnji korisnik obično očekuje. SQL je jezik prvenstveno usredotočen na definiranje podataka (*Data Definition Language*, DDL) i manipuliranje podacima (*Data Manipulation Language*, DML).[41] Uz te dvije osnovne kategorije dodatno može definirati poglede na bazu podataka, određivati sigurnost i autorizaciju, definirati ograničenja integriteta te odrađivati kontrolu transakcija.[42]

Glavna SQL naredba za definiranje podataka je CREATE, koja se može koristiti za izradu shema, tablica, vrsta i domena, ali i za izradu drugih konstrukcija poput pogleda i okidača.[43] Najčešće korištena naredba je CREATE TABLE, pomoću koje se dodjelom naziva definira nova relacija te se određuju njeni atributi i početna ograničenja. Prvo se definiraju atributi i svakom atributu se dodjeljuje naziv, određuje vrsta podataka za pojedinu domenu

[39] Ibid., 178.
[40] Ibid.
[41] Coronel, Morris, and Rob, *Database Systems*, 220.
[42] Elmasri and Navathe, *Fundamentals of Database Systems*, 178.
[43] Ibid., 179.

vrijednosti te moguća ograničenja atributa, poput nul vrijednosti. Vrste podataka mogu biti brojčane (int, float, decimal), tekstualne (char, varchar, text), datum i vrijeme (datetime) te logičke vrijednosti (bit). Primjer izrade tablice tblStudent korištenjem naredbe CREATE TABLE i kako izgleda kada se u polja upišu vrijednosti (slika 4):

```
CREATE TABLE tblStudent (
StudentID int  IDENTITY,
Ime nvarchar(100),
Prezime nvarchar(100) NOT NULL,
ECTS int,
PRIMARY KEY(StudentID))
```

StudentID	Ime	Prezime	ECTS
1	Pero	Perić	1
2	Ivana	Ivić	5
3	Marko	Marković	5
4	Petra	Petrić	4
5	Ante	Ivić	2

Slika 4. Tablica tblStudent

Ključevi, entitetski i referencijalni integritet mogu se odrediti naredbom CREATE TABLE nakon što su atributi definirani ili mogu biti naknadno dodani korištenjem naredbe ALTER TABLE. Općenito, ALTER TABLE naredbom se naknadno mogu mijenjati vrijednosti stvorene tablice pa je tako moguće mijenjati vrijednosti atributa, dodavati nove atribute ili brisati neželjene atribute. Za definiranje podataka tablice još se koristi i DROP TABLE naredba, čijim izvršavanjem se briše tablica i svi njeni podaci.

SQL ima jednu osnovnu naredbu za dohvat podataka iz baze podataka, a to je SELECT naredba. Upiti u SQL-u mogu biti vrlo kompleksni. Osnovni oblik korištenja SELECT naredbe, ponekad zvan mapiranje ili select-from-where blok, sastoji se od sljedećih komponenti:[44]

```
SELECT    [popis atributa]
FROM      [popis tablica]
WHERE     [uvjet]
```

[44] Ibid., 188.

Popis atributa predstavlja popis naziva atributa čije vrijednosti postavljeni upit dohvaća. Popis tablica je popis naziva relacija potrebnih da se upit obradi. Relacije je moguće međusobno povezivati pomoću izraza *join_type* (npr. INNER JOIN i LEFT OUTER JOIN). Da bi se dohvatile n-torke koji zadovoljavaju određene uvjete, moguće je postavljati uvjete u upitu. Za slaganje složenijih upita mogu se koristiti logički operatori (AND, OR, NOT), posebni operatori (npr. IN, BETWEEN, LIKE), funkcije niza (npr. LEN za duljinu znakovnog niza), funkcije vezane uz vrijeme i datum (npr. GETDATE koji vraća trenutni sistemski datum i vrijeme) i agregatne funkcije koje rade kalkulacije nad grupom vrijednosti i kao rezultat vraćaju jednu vrijednost (MAX, MIN, SUM, AVG, COUNT). Uz osnovne naredbe za dohvaćanje podataka, postoji još ORDER BY naredba pomoću koje se sortiraju izlazni rezultati zadani prema kolonama, od najmanje vrijednosti prema najvećoj (ASC) ili od najveće vrijednosti prema najmanjoj (DESC). Primjer malo složenijeg upita na ranije izrađenoj tablici tblStudent:

```
SELECT    *              (oznakom * dohvaćamo sva polja u tablici)
FROM      tblStudent
WHERE     ECTS>1
ORDER BY  Prezime
```

StudentID	Ime	Prezime	ECTS
2	Ivana	Ivić	5
5	Ante	Ivić	2
3	Marko	Marković	5
4	Petra	Petrić	4

Slika 5. Tablica tblStudent nakon izvršenja SELECT naredbe

U SQL-u se koriste tri naredbe za manipuliranje podacima: INSERT, UPDATE i DELETE. U svojem najosnovnijem obliku, INSERT naredba se koristi za dodavanje jedne n-torke (retka) u relaciju (tablicu). Potrebno je odrediti naziv relacije i popis vrijednosti za n-torku. Vrijednosti bi trebale biti navođene isti redom kako su atributi definirani prilikom izrade tablice CREATE TABLE naredbom. Drugi oblik INSERT naredbe dopušta korisniku da navede vrijednosti samo za atribute koje istakne uz naziv relacije. Takav način je koristan ako se relacija sastoji od mnogo atributa, a samo manjem broju tih atributa je potrebno dodijeliti vrijednost u novoj n-torci. Međutim, vrijednosti moraju biti dodijeljene svim atributima koji nisu nul vrijednost i koji nemaju zadanu vrijednost. Atributi koji mogu biti nul vrijednost i

imaju zadanu vrijednost se mogu izostaviti. Na primjer, pri dodavanju n-torke u tablicu tblStudent izostavlja se atribut StudentID jer je definiran kao Samonumeriranje pa se o unosu toga polja brine baza podataka i korisnik ga ne može mijenjati:

```
INSERT INTO     tblStudent (Ime, Prezime, ECTS)
VALUES          ('Tomislav', 'Andrić', 3)
```

StudentID	Ime	Prezime	ECTS
1	Pero	Perić	1
2	Ivana	Ivić	5
3	Marko	Marković	5
4	Petra	Petrić	4
5	Ante	Ivić	2
6	Tomislav	Andrić	3

Slika 6. Tablica tblStudent nakon izvršenja INSERT naredbe

Naredba UPDATE koristi se za promjenu vrijednosti atributa jedne ili više odabranih n-torki. Uključuje naredbu SET koja određuje atribute koji će biti mijenjani i njihove nove vrijednosti te naredbu WHERE koja izdvaja n-torke pojedine relacije. Veći broj n-torki je moguće modificirati sa jednom UPDATE naredbom, ali svaka naredba se eksplicitno odnosi na samo jedno relaciju. Za modifikaciju višestrukih relacija potrebno je izvršiti nekoliko UPDATE naredbi. Primjer promjene podataka u tablici tblStudent:

```
UPDATE      tblStudent
SET         Ime='Andrej', Prezime='Andrić'
WHERE       StudentID=5
```

StudentID	Ime	Prezime	ECTS
1	Pero	Perić	1
2	Ivana	Ivić	5
3	Marko	Marković	5
4	Petra	Petrić	4
5	Andrej	Andrić	2

Slika 7. Tablica tblStudent nakon izvršenja UPDATE naredbe

DELETE naredba briše n-torke iz relacije. Uključuje naredbu WHERE koja izdvaja n-torke namijenjene za brisanje. N-torke je moguće istovremeno brisati iz samo jedne tablice. Međutim, brisanje se može proširiti na n-torke u drugim relacijama ako su specificirane referencijalne akcije u ograničenjima referencijalnog integriteta jezika za definiranje podataka.[45] Ovisno o broju n-torki na koje se odnosi uvjet definiran WHERE naredbom, izvršavanjem jedne DELETE naredbe moguće je pobrisati nijednu, jednu ili više n-torki. Ako nema definiranog uvjeta sve će n-torke u relaciji biti pobrisane. Definirana tablica i dalje je dostupna u bazi podataka jer tablicu je moguće pobrisati samo uz pomoć naredbe DROP TABLE. Primjer brisanja podataka u tablici tblStudent:

```
DELETE FROM     tblStudent
WHERE           StudentID=3
```

StudentID	Ime	Prezime	ECTS
1	Pero	Perić	1
2	Ivana	Ivić	5
4	Petra	Petrić	4
5	Ante	Ivić	2

Slika 8. Tablica tblStudent nakon izvršenja DELETE naredbe

2.2 C#

C# (čita se *see sharp*) je objektno-orijentirani jezik koji omogućuje programerima izradu raznolikih sigurnih i robusnih aplikacija koje se pokreću na .NET Framework platformi.[46] Jezik su razvili Microsoftovi inženjeri, a može se koristiti za izradu Windows klijentskih aplikacija, XML web servisa, klijent-poslužitelj aplikacija, aplikacija baza podataka i dr. C# je jednostavan, što ga čini prikladnim za programere početnike, ali isto tako uključuje svu podršku za strukturirano, objektno-orijentirano programiranje koja se očekuje od modernog jezika izgrađenog na principa jezika C++ i Java. Drugim riječima, C# je potpuno opremljen i visoko

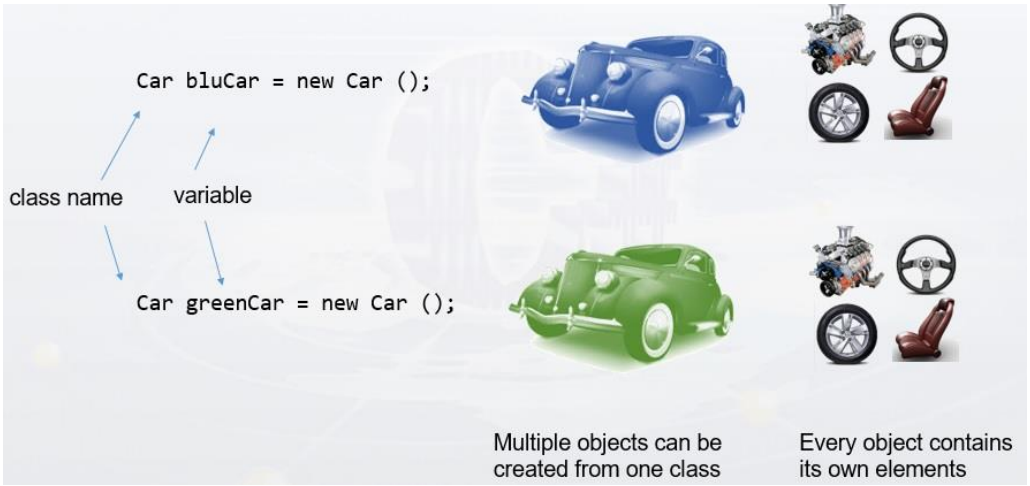
[45] Ibid., 200.

[46] "Introduction to the C# Language and the .NET Framework," accessed February 27, 2017, https://msdn.microsoft.com/en-us/library/z1zx9t92.aspx.

ekspresivan jezik prikladan za razvoj velikih aplikacija, ali u isto vrijeme dizajniran na način da ga je moguće lako naučiti.[47] Općenito, razlozi zbog kojih je C# široko korišten jezik su:[48]

- moderni programski jezik opće namjene
- objektno-orijentiran jezik
- komponentno-orijentiran jezik
- jednostavan za naučiti
- strukturirani jezik
- stvara učinkovite programe
- jezik koji je moguće kompajlirati na različitim računalnim platformama
- dio .NET Framework okruženja

Kao objektno-orijentirani jezik, C# podržava koncepte enkapsulacije, nasljeđivanja i polimorfizma.[49] Sve varijable i metode, uključujući i metodu *Main* koja je ulazna točka aplikacije, učahurene su u definiciji klase. Klasa je tip podataka koji se definira zajedničkim grupiranjem varijabli drugih tipova podataka. Funkcionalnost klase nemoguće je izravno koristiti, odnosno ne može se koristiti metode, svojstva, događaje i konstruktore izravno na klasi. Ti elementi se koriste na objektima koji se procesom instantizacije (engl. *Instantiation*) stvaraju iz klasa (slika 9). Stoga, definiranjem klase određuje se značenje naziva klase, odnosno od čega se objekt klase sastoji i koje se operacije na njemu mogu izvoditi.



Slika 9. Proces instantizacije

[47] "Chapter 1. C# and .NET Programming," accessed February 7, 2017, https://msdn.microsoft.com/en-us/library/orm-9780596521066-01-01.aspx.

[48] tutorialspoint.com, "C# Overview," *Www.tutorialspoint.com*, accessed February 27, 2017, https://www.tutorialspoint.com/csharp/csharp_overview.htm.

[49] "Introduction to the C# Language and the .NET Framework."

2.2.1 .NET Framework

C# programi se pokreću na .NET Framework platformi koja se sastoji od zajedničkog jezika izvođenja (engl. *Common Runtime Language*, CLR) i knjižnice klasa. CLR je Microsoftova komercijalna implementacija infrastrukture zajedničkog jezika (engl. *Common Language Infrastructure*, CLI), internacionalnog standarda koji je osnova za stvaranje izvršnih i razvojnih okruženja u kojima jezici i programske knjižnice zajedno rade besprijekorno.[50] Knjižnica klasa pruža fond testiranog, ponovno iskoristivog koda koji programeri mogu pozvati iz svojih vlastitih aplikacija. Usluge koje .NET Framework pruža aplikacijama koje se izvode uključuju sljedeće:[51]

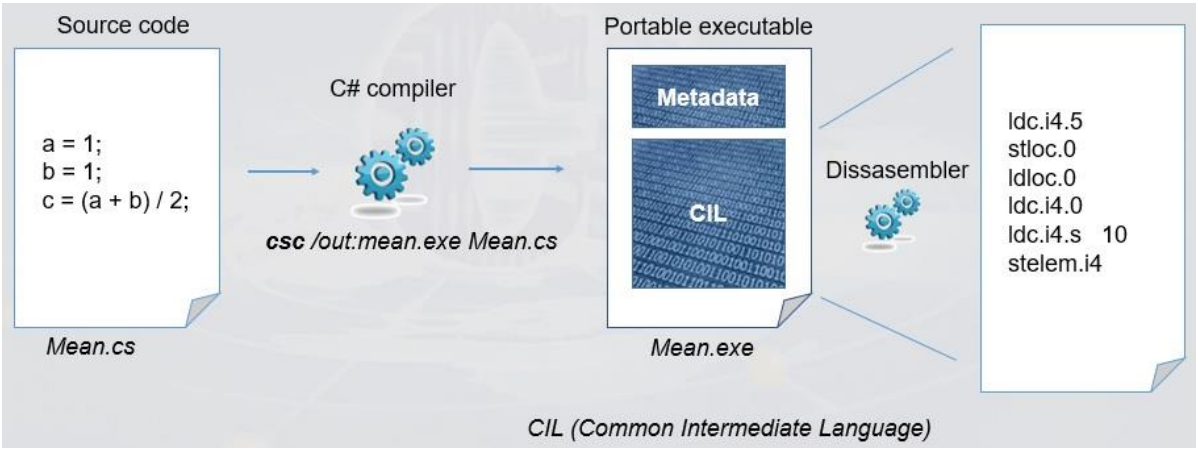
- *upravljanje memorijom* – umjesto programera, CLR je odgovoran za alokaciju i ispuštanje memorije te za rukovanje životnim ciklusom objekata
- *sustav zajedničkih tipova* – u tradicionalnim programskim jezicima osnovni tipovi su definirani kompajlerom, što otežava međujezičnu interoperabilnost; u .NET Framework okruženju osnovni tipovi definirani su pomoću .NET Framework sustava tipova te su zajednički za sve jezike koji su usmjereni na .NET Framework
- *opsežna knjižnica klasa* – umjesto pisanja ogromne količine koda za obradu uobičajenih programskih operacija niske razine, programeri mogu koristiti lako dostupnu knjižnicu tipova i njihovih članova
- *razvojni okviri i tehnologije* – .NET Framework sadrži programske knjižnice za pojedina područja razvoja aplikacija, kao što su ASP.NET za web aplikacije i ADO.NET za pristup podacima
- *jezična interoperabilnost* – jezik kompajlera koji su usmjereni na .NET Framework emitiraju međukod pod nazivom zajednički međujezik (*Common Intermediate Language*, CIL), koji se kompajlira za vrijeme izvođenja CLR-a; na taj način, rutine napisane u jednom jeziku dostupne su na drugim jezicima pa se programeri mogu posvetiti izradi aplikacija u željenom jeziku ili jezicima
- *kompatibilnost verzija* – uz rijetke iznimke, aplikacije koje su razvijene korištenjem određene verzije .NET Framework okruženja mogu bez izmjena raditi na novijoj verziji

[50] Ibid.

[51] "Getting Started with the .NET Framework," accessed February 7, 2017, https://msdn.microsoft.com/library/hh425099(v=vs.110).aspx.

- *usporedno izvršavanje* – .NET Framework pomaže u rješavanju sukoba verzija dopuštajući postojanje više verzija CLR-a na računalu; to znači da više verzija aplikacija može koegzistirati te da se aplikacija može izvoditi na .NET Framework verziji na kojoj je izgrađena

Pisanje aplikacije korištenjem .NET Framework okruženja znači pisanje koda pomoću .NET Framework programske knjižnice kodova. Da bi se C# kod mogao izvršiti, mora se pretvoriti u jezik koji ciljani operacijski sustav razumije, tzv. izvorni kod.[52] Ova pretvorba se naziva kompajliranje koda, čin koji izvodi kompajler. Unutar .NET Framework okruženja taj se proces odvija u dva stupnja. Kada se kompajlira kod koji koristi .NET Framework programsku knjižnicu, ne stvara se odmah izvorni kod specifičnog operacijskog sustava. Umjesto toga, prvo se kod kompajlira u CIL kod, odnosno kod zajedničkog međujezika (slika 10). Taj kod nije specifičan za bilo koji operacijski sustav niti za C# jezik. Da bi se aplikacija izvršila potreban je drugi kompajler, tzv. *točno na vrijeme* (*just-in-time*, JIT) kompajler, koji kompajlira CIL u izvorni kod specifičan ciljanom operacijskom sustavu i računalnoj arhitekturi.[53] Samo u ovoj fazi kompajliranja koda operacijski sustav može izvršiti aplikaciju. Naziv JIT kompajlera odražava činjenicu da CIL kod se kompajlira samo kada je to potrebno.



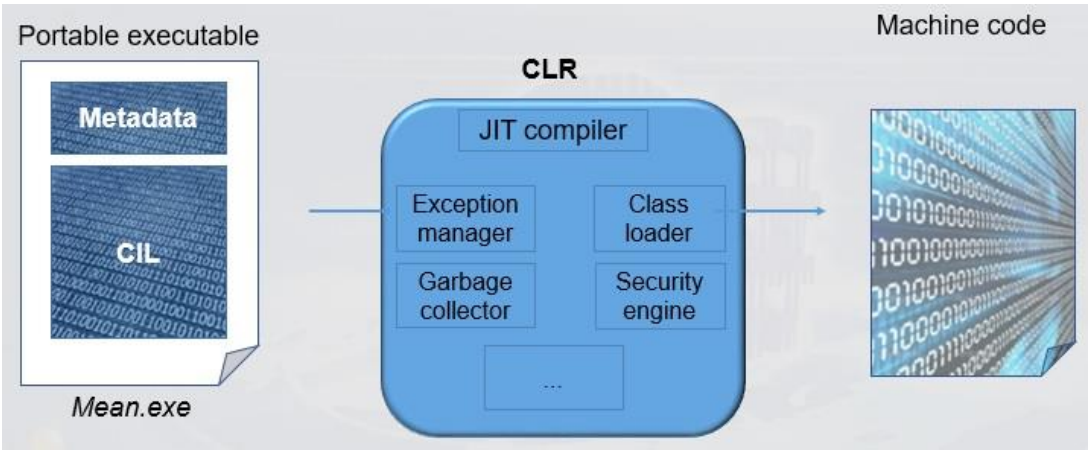
Slika 10. Kompajliranje u zajednički međujezik

Prilikom kompajliranja aplikacije, stvoreni CIL kod je pohranjen u programskom sklopu (engl. *assembly*).[54] Sklopovi uključuju obje izvršne datoteke aplikacije koje se mogu pokretati izravno na Windows sustavu, bez potrebe za korištenjem bilo kojeg programa (.exe verzija datoteke) i programskih knjižnica (.dll verzija datoteke) drugih aplikacija. Uz CIL kod, sklopovi

[52] "Chapter 1: Introducing C#," accessed February 7, 2017, https://msdn.microsoft.com/en-us/library/hh145616(VS.88).aspx.
[53] Ibid.
[54] Ibid.

sadrže metapodatke i dodatne resurse koje CIL koristi, poput slika i zvučnih datoteka (slika 10 i 11). Metapodaci omogućuju sklopovima potpunu samoopisivost. To znači da za korištenje sklopova nisu potrebne nikakve dodatne informacije o ciljanom operacijskom sustavu pa je moguće samo pokrenuti izvršnu datoteku aplikacije i, pod pretpostavkom da je instaliran .NET CLR, na taj način pokrenuti samu aplikaciju.

Općenito, uloga CLR-a ne završava nakon što je kod konačno kompajliran u izvorni kod (slika 11). Kodom napisanim u .NET Framework okruženju upravlja se kada je izvršen.[55] To znači da CLR brine o aplikacijama na način da upravlja memorijom, rukuje sigurnošću, omogućuje međujezično ispravljanje pogrešaka (engl. *cross-language debugging*) i sl. Suprotno tome, aplikacije koje se ne pokreću pod kontrolom CLR-a su neregulirane i mogu biti pisane na određenim jezicima kao što je C++. Na primjer, aplikacija napisana C++ jezikom može pristupiti funkcijama niže razine operacijskog sustava. U C# moguće je napisati samo kod koji se pokreće u reguliranom okruženju. Programer će iskoristiti upravljačka obilježja CLR-a i prepustiti .NET platformi da samostalno rješava bilo kakvu interakciju s operativnim sustavom.



Slika 11. Izvršavanje aplikacije pomoću CLR-a

Jedna od najvažnijih značajki reguliranog koda je koncept čišćenja memorije (tzv. *garbage collection*).[56] Riječ je od .NET metodi koja osigurava potpuno oslobađanje memorije koju aplikacija koristi kada sama aplikacija više nije u upotrebi. Prije pojave .NET platforme to je uglavnom bila odgovornost programera i nekoliko jednostavnih pogrešaka u kodu moglo je rezultirati nestajanjem velikih blokova memorije. Takav razvoj događaja je uglavnom značio progresivno usporavanje računala i naposljetku pad sustava. .NET čišćenje memorije radi na

[55] Ibid.
[56] Ibid.

način da periodički provjerava memoriju računala i iz nje uklanja sve što više nije potrebno. Ne postoji vremenski rok čišćenja memorije; može se dogoditi tisuću puta u sekundi, svakih nekoliko sekundi ili u nekom drugom intervalu, ali sigurno je da će biti očišćen onaj dio memorije koji se više ne koristi.

2.2.2 ASP.NET

ASP.NET (*Active Server Pages.NET*) je razvojna platforma koja pruža model za programiranje, sveobuhvatnu softversku infrastrukturu i razne usluge potrebne za razvoj web aplikacija. Dio je .NET Framework okruženja što znači da se za kodiranje koristi cjelokupna knjižnica klasa pa su ASP.NET aplikacije zapravo kompajlirani kodovi napisani pomoću proširivih i ponovno iskoristivih komponenata ili objekata prisutnih u .NET Framework okruženju. ASP.NET se koristi za izradu interaktivnih, temeljenih na podacima aplikacija na internetu. Funkcionira na vrhu HTTP protokola i koristi HTTP naredbe i pravila za uspostavljanje dvosmjerne preglednik-poslužitelj komunikacije i suradnje.[57]

ASP.NET nudi tri okvira za izradu web aplikacija: ASP.NET Web Forms, ASP.NET MVC i ASP.NET Web Pages.[58] Okviri su stabilni i razvijeni te je moguće sa bilo kojim od njih razviti kvalitetnu aplikaciju. Svaki od okvira cilja drugačiji stil razvoja aplikacija. Za koji od okvira će se pojedini programer odlučiti ovisi o njegovom programerskom znanju i vještinama, vrsti aplikacije koju razvija i općenito koliko mu je ugodno raditi u pojedinom okviru. U sljedećim odlomcima opširnije ćemo predstaviti okvire za izradu web aplikacija.

ASP.NET Web Forms (.aspx datoteke) cilja programere koji preferiraju deklarativno i temeljeno na kontroli programiranje, poput Microsoft Windows Forms (WinForms). On nudi tzv. *drag-and-drop*, temeljen na događajima model razvoja pa je zato popularan kod programera koji traže okruženje za brzi razvoj web aplikacija (*Rapid application development*, RAD). Konkretno, model web obrazaca pruža sljedeće značajke:[59]

- model temeljen na događajima prikazuje događaje koje je moguće programirati na način koji se programira klijentska aplikacija poput WinForms
- kontrole poslužitelja koje prikazuju obrazac u HTML obliku; moguće ga je prilagoditi podešavanjem svojstva i stilova

[57] "ASP.NET - Introduction," *Www.tutorialspoint.com*, accessed February 14, 2017, https://www.tutorialspoint.com/asp.net/asp.net_introduction.htm.

[58] "ASP.NET Overview," accessed March 1, 2017, https://msdn.microsoft.com/en-us/library/4w3ex9c2.aspx.

[59] Ibid.

- bogati asortiman kontrola za pristup i prikaz podataka (npr. *Button, Radio Button, TextBox, CheckBox, HyperLink, FileUpload*)
- automatsko očuvanje stanja (podataka) između HTTP zahtjeva

Riječ je o programerskom okviru koji radi na web poslužitelju kako bi dinamički proizveo i prikazao ASP.NET web stranice. Web stranice je moguće zatražiti preko bilo kojeg preglednika ili klijentskog uređaja, koje kao rezultat web poslužitelj prikazuje u HTML obliku. Pravilo je da se ista stranica može koristiti na više različitih preglednika. Stanice ASP.NET web obrazaca potpuno su objektno-orijentirane. Unutar stranice moguće je raditi sa HTML elementima koristeći svojstva, metode i događaje. Ovaj okvir uklanja detalje razdvajanja klijenta i poslužitelja koji su svojstveni za web aplikacije i predstavlja ujedinjeni model koji odgovara na klijentove događaje kodom koji se pokreće na poslužitelju.[60] Također, okvir automatski održava stanje stranice i kontrole stranice za vrijeme ciklusa njihove obrade.

ASP.NET MVC cilja programere koji su zainteresirani za obrasce i principe poput razvoja temeljenom na testiranju, razdvajanja problema, inverzije kontrole (*Inversion of Control*) i međuzavisnosti objekata (*Dependency Injection*).[61] Okvir potiče odvajanje sloja poslovne logike od prezentacijskog sloja web aplikacije. Dijeljenjem aplikacije na model (M), poglede (V) i kontrolore (C), ASP.NET MVC može smanjiti kompleksnost većih aplikacija. Na primjer, više timova dizajnera mogu raditi na razvoju web stranice, a programeri kodirati poslovnu logiku. Unutar ovog okvira radi se izravnije sa HTML-om i HTTP protokolom u odnosu na web obrasce. Okvir web obrazaca nastoji sakriti neka od tih ponašanja. Na primjer, web obrasci mogu automatski sačuvati stanje između HTTP zahtjeva, dok se u slučaju MVC-a to mora eksplicitno kodirati. MVC model omogućuje preuzimanje potpune kontrole nad onim što aplikacija radi i kako se ponaša u web okruženju.

U ASP.NET Web Pages (.cshtml i .vbhtml datoteke) okviru prvo se izrađuju HTML stranice, a kasnije se stranici dodaje kod koji se nalazi na poslužitelju i koji dinamički kontrolira kako će stranica biti prikazana. Ovaj okvir se smatra idealnom početnom točkom za upoznavanje sa ASP.NET platformom, posebno za ljude koji poznaju HTML, a nemaju široko znanje i vještine programiranja. Kao i u slučaju web obrazaca, ASP.NET Web Pages usredotočen je na brzi razvoj aplikacije. Okvir pruža komponente, tzv. pomagače (engl. *helpers*), koje je moguće dodati na stranicu i koji omogućuju korištenje nekoliko linija koda za

[60] Ibid.
[61] Ibid.

izvršavanje zadataka koji bi inače bili zamorni ili kompleksni.[62] Tako primjerice postoje pomagači za prikaz podataka iz baze podataka, prijave putem Facebook računa, dodavanje karata na stranicu i sl.

Sva tri ASP.NET okvira temelje se na .NET Framework okruženju i dijele osnovne funkcionalnosti .NET i ASP.NET platformi. Zanimljivo je da okviri nisu u potpunosti neovisni i odabir jednog ne isključuje korištenje drugog istovremeno. Na primjer, MVC pogledi često su .cshtml ili vbhtml datoteke, što znači da mogu iskoristiti neke od značajki okvira web stranica, poput pomagača. Budući da okviri mogu koegzistirati u istoj web aplikaciji, nije neuobičajeno vidjeti individualne komponente aplikacije pisane korištenjem različitih okvira. Na primjer, većina stranice je napisana u MVC-u, ali dio stranice vezan za pristup podacima može biti napisan pomoću web obrasca jer je to okvir specijaliziran za pristup podacima. U takvim slučajevima, programeri izabiru hibridna rješenja kako bi si olakšali rješavanje pojedinih zadataka aplikacije.

3. Zaštita i očuvanje arhivskog gradiva Saveza društava Naša djeca Hrvatske

3.1 Detaljan opis programa

Savez društava Naša djeca Hrvatske (u nastavku Savez DND) je dragovoljna, edukativna, humanitarna i neprofitna udruga građana, sa statusom pravne osobe. Djeluje za opće dobro, promiče, organizira te vodi akcije i aktivnosti namijenjene dobrobiti djece. U Savez DND se kao u krovnu udrugu dragovoljno udružuju osnovna Društva Naša djeca koja, također kao pravne osobe, neposredno djeluju u gradovima i općinama Hrvatske radi ostvarivanja zajedničkih ciljeva. Savez DND osnovan je 1950. godine, a osnovni cilj mu je pridonositi ostvarivanju prava i potreba djece, poticati i pomagati osnovna DND u provođenju odgojnih, zdravstvenih, socijalnih, kulturnih i rekreativnih aktivnosti i akcija s djecom i za djecu od rođenja do završetka osnovnog školovanja u njihovom slobodnom vremenu te pružati podršku roditeljima u razvoju i odgoju djece. Cjelokupna aktivnost zasniva se na odredbama Konvencije UN-a o pravima djeteta, kao i na željama, interesima i potrebama djece i njihovih roditelja.[63]

Prema Zakonu o arhivskom gradivu i arhivima, arhivsko gradivo su zapisi ili dokumenti koji su nastali djelovanjem pravnih ili fizičkih osoba u obavljanju njihove djelatnosti, a od trajnog su značenja za kulturu, povijest i druge znanosti, bez obzira namjesto i vrijeme njihova

[62] Ibid.
[63] "Zaštita i očuvanje arhivskog gradiva Saveza društava Naša djeca Hrvatske," n.d., 1.

nastanka, neovisno o obliku i tvarnom nosaču na kojem su sačuvani. U Zakonu se također ističe da je arhivsko gradivo od interesa za Republiku Hrvatsku i ima njenu osobnu zaštitu na koju se primjenjuju i propisi o zaštiti kulturnih dobara.[64] Imajući u vidu zakonske obveze kao i širu društvenu svrhu zaštite gradiva od trajnog značaja Savez DND, kao stvaratelj i ujedno imatelj arhivskog gradiva, želi osigurati adekvatno i propisno čuvanje svoje bogate i značajne građe. Savezu kao imatelju arhivskog gradiva izdano je Uvjerenje o upisu u Evidenciju stvaratelja arhivskog gradiva, Rješenje o kategorizaciji i Rješenje o upisu u Upisnik vlasnika i imatelja privatnog arhivskog gradiva u RH pa je stoga obavezan provoditi mjere zaštite gradiva određene odnosnim propisima.[65] Obveze stvaratelja i imatelja privatnog arhivskog gradiva su: osiguranje odgovarajućih uvjeta pohrane i zaštite, sređenost gradiva i popisanost te vođenje evidencija o ukupnom gradivu i dostava popisa nadležnom arhivu.[66]

Hrvatski državni arhiv procijenio je gradivo Saveza od državnog značaja. Savez DND, kao prva, i u ono vrijeme među jedinima društvenim organizacijama za djecu u tadašnjoj državi, okupljala je velik broj najuglednijih i najvećih stručnjaka onog vremena koji su djelovali na području rada s djecom i osmišljavanja sadržaja za djecu. Potrebno je posebno naglasiti kako su se pri Savezu DND osnivali različiti centri: Centar za porodični odgoj, Centar za odgoj djece predškolske dobi, Centar za vanškolski odgoj djece, Centar za rad s roditeljima i dr. Tako je primjerice Centar za vanškolski odgoj djece kasnije uklopljen u Zavod za školstvo pod nadzorom Ministarstva prosvjete 1977. (a Ministarstva su stvaratelji arhivskog gradiva označeni kao javni stvaratelj prve kategorije) te se dio građe neposredno vezane za djelatnosti tog Zavoda i svih ostalih centara prirodno čuva u arhivu Saveza DND.

Glavna zadaća Saveza odnosi se na zaštitu i promicanje interesa i prava građana. Čuvanjem svjedočanstva i temelja ovog arhivskog gradiva može se proučavati povijest i razvoj izvaninstitucionalnog odgoja i obrazovanja djece, fenomen slobodnog vremena djece uopće, umjetničkog stvaranja za djecu, izdavaštva za djecu, društvenim akcijama i aktivnostima za djecu, stručnih skupova i događanja usmjerenih djecu, prvim počecima rada na području dječjih prava i dječje participacije, kao i kontinuiranim poticanjem volonterstva postoji potreba za očuvanjem tih vrijednosti koje su se promicale i njegovale od početaka djelovanja Saveza do danas.[67] Arhivsko gradivo Saveza DND, osim društvene i kulturološke vrijednosti, također je

[64] "Zakon o arhivskom gradivu i arhivima - Zakon.hr," accessed March 29, 2017, https://www.zakon.hr/z/373/Zakon-o-arhivskom-gradivu-i-arhivima.
[65] " Zaštita i očuvanje arhivskog gradiva Saveza društava Naša djeca Hrvatske," 2.
[66] "Zakon o arhivskom gradivu i arhivima - Zakon.hr."
[67] "Zaštita i očuvanje arhivskog gradiva Saveza društava Naša djeca Hrvatske," 3.

neophodna i važna u kontekstu razvoja organizacija civilnog društva i aktivnog građanstva u lokalnim zajednicama. Treba naglasiti da se tijekom povijesti djelovanja Saveza DND u njega udruživalo više stotina Društava Naša djeca koja su, surađujući sa Savezom, ostavljala pisani trag o svojoj djelatnosti, ali i o povijesti svakodnevice i slobodnog vremena u svojim lokalnim zajednicama. Djelatnost ovih društava uglavnom se zasniva na volonterskom doprinosu svojih članova pa tako postoji vrlo velika vjerojatnost da se dio gradiva koji se čuvao u raznim manjim sredinama Hrvatske zagubio ili propao.

Gradivo Saveza DND dosad je bilo dostupno djelatnicima Stručne službe, članovima Društava Naša djeca i zainteresiranim stručnjacima-pojedincima, a želja Saveza je učiniti to gradivo dostupnim široj javnosti jer postoji svijest o društvenom značaju arhivske djelatnosti. Kako bi se smanjila vjerojatnost da se pisani tragovi prošlosti zauvijek izgube uslijed nepovoljnih uvjeta skladištenja, nedostatka osoblja i stručnog znanja u Savezu DND i osnovnim Društvima Naša djeca, potrebno je učiniti maksimalne napore da se bogati arhivski materijal, koji je Savez stvorio i preko mreže osnovnih DND-a sakupio, zaštititi, očuva i postane dostupan javnosti u skladu s modernim standardima arhiviranja. Stoga, opći cilj ovog programa je doprinos razvoju arhivske djelatnosti u skladu s najsuvremenijim trendovima i potrebama zajednice, a specifični ciljevi su sljedeći:[68]

- osiguravanje fizičkih uvjeta zaštite gradiva i uređenje prostora pismohrane Saveza DND kako bi se spriječilo propadanje ili oštećenje vrijednog arhivskog gradiva
- nastavak sređivanja i popisivanja gradiva Saveza prema dokumentacijskim cjelinama
- arhiviranje i ažuriranje arhivskog gradiva Saveza sukladno zakonskim propisima i pravilnicima o arhiviranju radi dostave Hrvatskom državnom arhivu
- osiguranje vidljivosti i iskoristivosti arhivskog gradiva Saveza DND zainteresiranoj javnosti i struci

Budući da je Savez DND neprofitna udruga koja raspolaže s ograničenim financijskim sredstvima, u ostvarivanju ciljeva programa zaštite i očuvanja arhivskog gradiva u velikoj mjeri se oslanja na volonterski rad. Većina volontera koja je radila na sređivanju arhivskog gradiva su studenti Informacijskih znanosti na Filozofskom fakultetu Sveučilišta u Zagrebu pa se tako razvila ideja izrade arhivskog informacijskog sustava koji će olakšati pristup i pronalaženje

[68] Ibid., 4.

gradiva i informacija sadržanih u gradivu. Gledano iz arhivističke perspektive, sustav bi obavljao ulogu obavijesnog pomagala u kojem su zabilježene informacije o karakteristikama i stanju sređenosti arhivskog gradiva s ciljem postizanja zadovoljavajuće razine intelektualne kontrole. Za izradu obavijesnog pomagala ključan je postupak opisa arhivskog gradiva i budući da želimo informacijski sustav koji je u skladu s arhivističkim standardima, u sljedećem potpoglavlju biti će prikazana opća međunarodna norma za opis arhivskoga gradiva, ISAD (G).

3.2 ISAD (G)

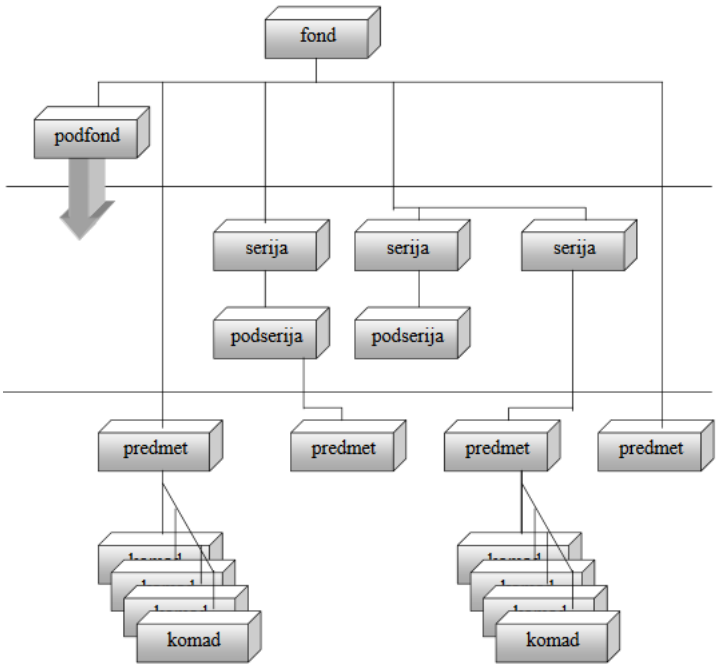
ISAD (G) normu izradilo je Povjerenstvo za norme opisa Međunarodnoga arhivskoga vijeća (ICA/CDS) 1994. godine, a drugo revidirano izdanje koje je još uvijek aktualni standard objavljeno je 2000. godine. Norma daje opće smjernice za izradu opisa arhivskoga gradiva i predviđena je za korištenje uz nacionalne norme ili kao temelj za razvoj nacionalnih normi. Cilj opisa arhivskoga gradiva je identificirati i pojasniti kontekst i sadržaj arhivskoga gradiva i na taj način olakšati njegovu dostupnost. To se ostvaruje izradom jasnih i prikladnih prikaza i njihovim organiziranjem sukladno unaprijed utvrđenim modelima. Postupci vezani uz opis mogu započeti kod stvaranja arhivskih zapisa i nastaviti se kroz njihov životni ciklus. Ti postupci omogućuju uspostavu intelektualnoga nadzora, potrebnoga kako bi se osiguralo da se opisani arhivski zapisi trajno prenose kao pouzdani, autentični i dostupni.[69]

Specifični elementi obavijesti o arhivskome gradivu zapisuju se u svakoj fazi njegovoga upravljanja (npr. stvaranja, vrednovanja, preuzimanja, čuvanja, sređivanja) kako bi gradivo bilo sigurno pohranjeno i nadzirano te dostupno svima koji imaju pravo na njegovo korištenje. Arhivski opis u najširem smislu riječi obuhvaća svaki element obavijesti, bez obzira u kojem razdoblju upravljanja je identificiran ili utvrđen. U svakom razdoblju obavijest o gradivu ostaje dinamična i može biti podložna izmjeni ako dođe do novih spoznaja o njegovom sadržaju ili kontekstu njegovoga nastanka. Računalni informacijski sustavi mogu osobito pomoći da se prema konkretnim potrebama integriraju ili selekcioniraju elementi obavijesti ili da ih osuvremenimo, odnosno poboljšamo. Premda su pravila ove norme prvenstveno odnose na opis arhivskoga gradiva nakon što je odabrano za trajno čuvanje, ona se mogu primijeniti i u ranijim fazama.[70]

[69] International Council of Archives, ed., *ISAD(G): General International Standard Archival Description ; Adopted by the Committee on Descriptive Standards, Stockholm, Sweden, 19-22 September 1999*, 2. ed (Ottawa: International Council of Archives, 2000), 7.

[70] Ibid.

Norme arhivističkoga opisa utemeljene su na općeusvojenim teorijskim načelima. Na primjer, načelo da arhivistički opis ide od općega ka posebnome praktična je posljedica načela poštivanja fonda.[71] Slika 12 prikazuje hijerarhijski model razina sređivanja i njihovih sastavnih dijelova. U njemu su prikazane razine opisa s različitim stupnjevima pojedinosti, koje odgovaraju svakoj razini sređivanja. Na primjer, fond može biti opisan kao cjelina u jednostavnom opisu ili može biti prikazan kao cjelina i u dijelovima na različitim razinama opisa. Fond predstavlja najvišu razinu opisa, a dijelovi su podređene razine čiji opis često ima svoje puno značenje samo ako se promatra u kontekstu opisa cjeline fonda. Tako može postojati opis na razini fonda, serije, predmeta i/ili komada. Moguće je predvidjeti i međurazine, kao što su podfond ili podserija. Svaka od tih razina može se dalje dijeliti zavisno od složenosti upravne strukture i/ili funkcija organizacije čijim je radom gradivo nastalo i prema samoj organizaciji zapisa.



Slika 12. Model razina sređivanja arhivskoga fonda

ISAD (G) sadrži opća pravila za arhivistički opis, koja se mogu primijeniti bez obzira na oblik ili nosač arhivskih zapisa. Ta opća pravila dio su postupka kojemu je cilj:[72]

- osigurati izradu dosljednih, uporabivih i razumljivih opisa
- olakšati pronalaženje i razmjenu obavijesti o arhivskome gradivu

[71] Ibid.
[72] Ibid.

- omogućiti razmjenu normativnih podataka
- omogućiti objedinjavanje opisa iz različitih arhiva u jedinstveni informacijski sustav

Pravila će ispuniti navedene ciljeve identificiranjem 26 elemenata opisa koji se mogu međusobno spajati kako bi tvorili opis nekog arhivističkog entiteta. Struktura i sadržaj obavijesti u svakome od tih elemenata treba biti oblikovana sukladno odgovarajućim nacionalnim pravilima. Budući da je riječ o općim pravilima, ona su zamišljena tako da se mogu primijeniti za svaki opis arhivskih zapisa, bez obzira na narav ili opseg opisane jedinice. Pravila su podijeljena na sedam područja opisa:[73]

1. **područje identifikacije** – sadrži bitne obavijesti za identifikaciju jedinice opisa
2. **područje konteksta** – sadrži obavijesti o porijeklu i sačuvanosti jedinice opisa
3. **područje sadržaja i ustroja** – sadrži obavijesti o predmetu i sređenosti jedinice opisa
4. **područje uvjeta dostupnosti i korištenja** – sadrži obavijesti o dostupnosti jedinice opisa
5. **područje dopunskih izvora** – sadrži obavijesti o gradivu koje je u važnom odnosu s jedinicom opisa
6. **područje napomena** – sadrži posebne obavijesti i obavijesti koje ne mogu biti smještene u bilo koje drugo područje
7. **područje kontrole opisa** – sadrži obavijesti o tome kako, kada i tko je izradio arhivistički opis

Pri opisu je moguće koristiti svih 26 elemenata obuhvaćenih općim pravilima, ali za svaki pojedini opis neophodno je samo nekoliko njih. Sljedećih šest elemenata smatra se nužnima za međunarodnu razmjenu obavijesti o opisu:[74]

- identifikacijska oznaka(e)/signatura(e)
- naslov
- stvaratelj
- vrijeme nastanka gradiva
- količina jedinice opisa
- razina opisa

[73] Ibid., 8.
[74] Ibid.

Elementi se nalaze u području identifikacije, uz iznimku stvaratelja koji je dio područja konteksta. U nastavku će za svaki od navedenih elemenata biti prikazani cilj, pravilo korištenja i primjeri iz stvarnog života.

Pomoću identifikacijske oznake/signature cilj je jedinstveno identificirati jedinicu opisa i omogućiti vezu s opisom koji je prikazuje. Kao pravilo je potrebno navesti sljedeće elemente nužne za jedinstvenu identifikaciju i mogućnost razmjene obavijesti na međunarodnoj razini: oznaku zemlje prema posljednjoj verziji ISO 3166 Oznake za prikaz imena zemlje, oznaku arhiva sukladno nacionalnoj normi za označavanje arhiva ili neku drugu jednoznačnu identifikacijsku oznaku te posebnu smještajnu oznaku/signaturu, kontrolni broj ili drugu jednoznačnu identifikacijsku oznaku.[75] Primjeri elementa identifikacijska oznaka/signatura:

HR HDA 1.1.0/10[2] (*fond*)
Hrvatska, Hrvatski državni arhiv

FR CHAN/363 AP 15 (*predmet*)
France, Centre historique des Archives nationales

Zapisivanjem naslova imenujemo jedinicu opisa. Osnovno pravilo je navesti izvorni naslov ili sažeti nadomjesni naslov sukladno pravilima višerazinskog opisa i nacionalnim propisima.[76] U nadomjesne naslove na višoj razini potrebno je uključiti naziv stvaratelja gradiva, dok na nižim razinama on može uključivati ime autora dokumenta i pojam koji označava vrstu gradiva sadržanog u jedinici opisa te neki izraz koji odražava funkcije, djelatnost, predmet, mjesto ili temu. Primjer naslova:

Hrvatsko kraljevsko vijeće (Consilium Regium Croaticum) (*fond*)
Povjerenstvo za poslove zdravstva (*serija*)
Hrvatska, Hrvatski državni arhiv

Cilj elementa vrijeme nastanka gradiva je utvrditi i navesti nadnevak nastanka gradiva u jedinici opisa. Potrebno je navesti najmanje jednu od sljedećih vrsta nadnevaka za jedinicu opisa: nadnevak primitka gradiva u tijeku poslovanja ili obavljanja vlastite djelatnosti te nadnevak kada su dokumenti nastali.[77] Ako je potrebno, moguće je navesti pojedinačni nadnevak ili raspon godina. Vrijeme nastanka gradiva treba uvijek biti uključeno, osim ako se

[75] Ibid., 13.
[76] Ibid., 14.
[77] Ibid., 15.

radi o jedinici (ili njezinom dijelu) opisa tekućih zapisa u pisarnicama. Primjeri bilježenja vremena nastanka gradiva:

1941-1945 (*fond*)
Hrvatska, Hrvatski državni arhiv

Finne anni '30- primi anni '40 (*predmet*)
Italy, Istituto Storico della Resistenza in Toscana
Napomena: Nadnevak preuzimanja predmeta

Pomoću razine opisa utvrđujemo razinu jedinice opisa u strukturi fonda. Kao što je ranije prikazano na slici 12, osnovne razine jedinice opisa su: fond, podfond, serija, podserija, predmet i komad. Cilj elementa količina i nosač jedinice opisa je utvrditi i opisati fizički ili logički opseg i nosač jedinice opisa. Osnovno pravilo je navesti količinu jedinice opisa arapskim brojevima i mjernim jedinicama ili navesti dužne ili kubične metre jedinice opisa. Podaci o količini označavaju se tehničkim jedinicama (knjiga, kutija, komad, svežanj, mapa, svitak, kaseta, omot, registrator) i dužnim metrima.[78] Primjer bilježenja količine i nosača jedinice opisa:

122 fotografije (*serija*)
Hrvatska, Hrvatski državni arhiv

103.5 cubic feet (98 boxes) (*fond*)
U.S., Minnesota Historical Society

Naziv stvaratelja je jedini od nužnih elemenata koji nije dio područja identifikacije, a njegov osnovni cilj je utvrditi stvaratelja ili stvaratelje jedinice opisa. Potrebno je navesti organizacije ili pojedinca koji su odgovorni za nastanak, prikupljanje i čuvanje gradiva u jedinici opisa. Naziv treba donijeti u normiranom obliku, kao što je propisano međunarodnim ili nacionalnim propisima sukladno načelima Međunarodne norme arhivističkog normiranog zapisa za pravne i fizičke osobe te obitelji ISAAR (CPF).[79] Primjer naziva stvaratelja:

Zagrebačka županija (*fond*)
Hrvatska, Hrvatski državni arhiv

Budući da je ISAD (G) predviđen za korištenje uz nacionalne norme ili kao temelj za stvaranje istih, zanimljivo je prikazati elemente opisa definirane u Pravilniku o zaštiti i čuvanju

[78] Ibid., 16.
[79] "ISAAR (CPF): International Standard Archival Authority Record for Corporate Bodies, Persons and Families, 2nd Edition | International Council on Archives," accessed April 4, 2017, http://www.ica.org/en/isaar-cpf-international-standard-archival-authority-record-corporate-bodies-persons-and-families-2nd.

arhivskog i registraturnog gradiva izvan arhiva. Prema Pravilniku, arhivsko gradivo se organizira u dokumentacijske cjeline ili zbirke koje se mogu sastojati od više manjih dokumentacijskih skupina, ako je to uputno zbog količine i raznovrsnosti gradiva koje sadrži ili radi lakšeg čuvanja i zaštite. Imatelj arhivskog gradiva dužan je nadležnom arhivu dostaviti popis svih dokumentacijskih cjelina, skupina i arhivskih jedinica koje posjeduje. Opis dokumentacijske skupine u popisu arhivskoga gradiva obuhvaća:[80]

- redni broj
- oznaka (klasifikacijska oznaka, arhivski ili drugi znak koji jednoznačno identificira cjelinu)
- naziv
- sadržaj (opis dokumentacije u cjelini, predmeta ili djelatnosti na koji se odnosi)
- medij i vrsta zapisa (podaci o vrsti medija i obliku zapisa dokumentacije u cjelini)
- količina
- tehničke jedinice (oznake ili raspon oznaka pripadajućih tehničkih jedinica – kutija, registratora i dr.)
- vrijeme nastanka
- napomene

Popis arhivskih jedinica (dokumenata, predmeta, dosjea, svezaka i dr.) u dokumentacijskoj skupini vodi se prema sljedećim podacima: redni broj, oznaka, naziv, vrsta, vrijeme nastanka, rok čuvanja, tehničke jedinice. ISAD (G) i Pravilnik o zaštiti i čuvanju arhivskog i registraturnog gradiva izvan arhiva dva su ključna izvora za izradu baze podataka sa elementima opisa koji će olakšati pretraživanje gradiva, ali i izradu popisa gradiva koji su imatelji dužni dostaviti nadležnom arhivu.

3.3 Programiranje baze podataka

Baza podataka će biti izrađena pomoću SQL Server Management Studija (SSMS). Riječ je o integriranom okruženju za upravljanje SQL infrastrukturom, od SQL poslužitelja do SQL baze podataka. SSMS pruža alate za konfiguriranje, nadzor, administraciju i nadogradnju podatkovnih komponenata, poput baza podataka i skladišta podataka, te izradu upita i skripti.[81] Potrebno je istaknuti da se prilikom pokretanja SSMS mora prijaviti na bazu podataka, odnosno

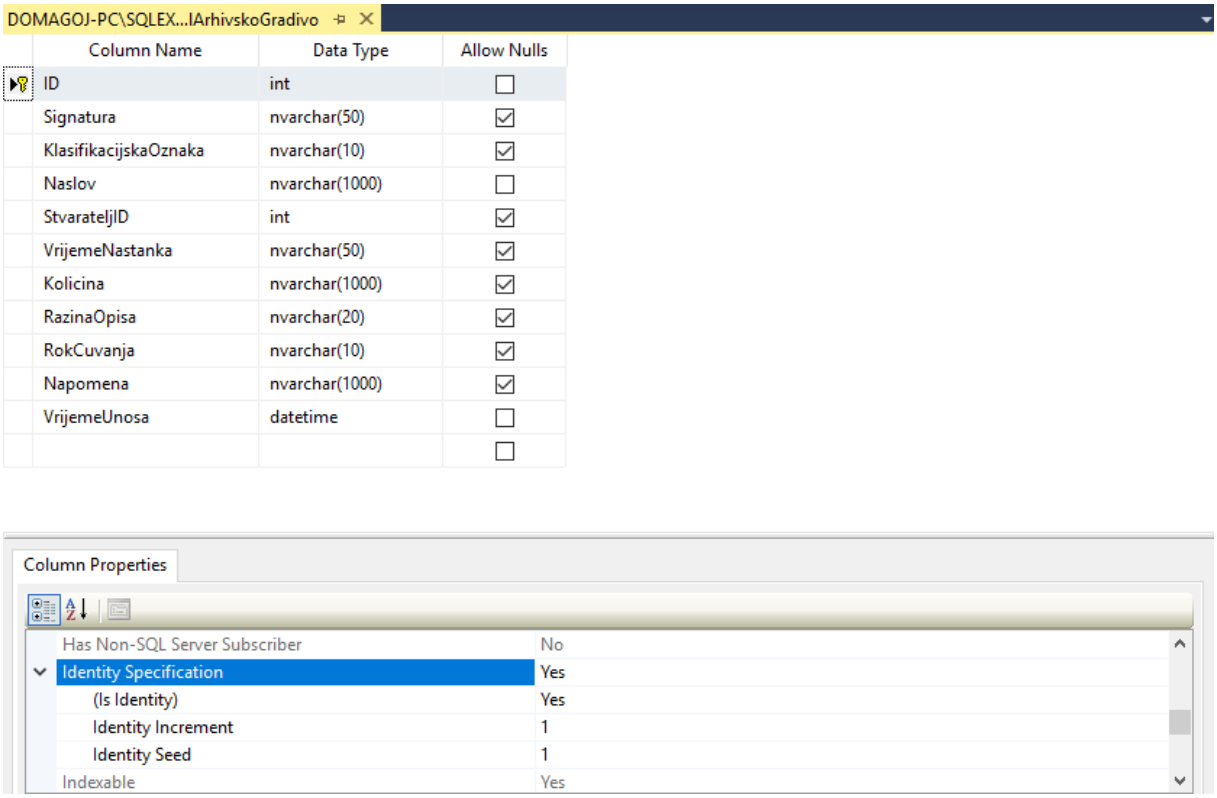
[80] "Pravilnik o zaštiti i čuvanju arhivskog i registraturnog gradiva izvan arhiva," accessed April 10, 2017, http://narodne-novine.nn.hr/clanci/sluzbeni/2004_05_63_1383.html.

[81] "SQL Server Management Studio (SSMS)," accessed April 11, 2017, https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms.

mora se povezati s aktivnim SQL poslužiteljem. U našem slučaju riječ je o poslužitelju SQL Server 2016 SP1 Express koji se koristi za izradu manjih, podatkovno orijentiranih mobilnih i web aplikacija veličine do 10 gigabajta.[82]

Glavni cilj prilikom programiranja baze podataka je da krajnji produkt bude baza podataka koja je detaljna i sadrži sve potrebne informacije o arhivskom gradivu koje se čuva, ali koja je također razumljiva i prilagođena zaposlenicima koji će ju kasnije koristiti. Stoga, baza podataka arhiva Saveza DND sastojati će se od tri tablice (relacije): tblArhivskoGradivo, tblStvaratelj i tblKlasifikacijskiPlan.

Tablica tblArhivskoGradivo je glavna tablica i sadrži atribute koji su definirani na temelju norme ISAD (G) i Pravilnika o zaštiti i čuvanju arhivskog i registraturnog gradiva izvan arhiva. Atributi su redom: ID, Signatura, KlasifikacijskaOznaka, Naslov, StvarateljID, VrijemeNastanka, Količina, RazinaOpisa, RokČuvanja, Napomena, VrijemeUnosa (Slika 13).



Slika 13. Tablica tblArhivskoGradivo

Svakome atributu unutar tablice moguće je odrediti određena svojstva u prozoru *Column Properties*. Tako je primjerice polju primarnog ključa moguće uključiti svojstvo *Is Identity*

[82] "Microsoft® SQL Server® 2016 SP1 Express," *Microsoft Download Center*, accessed April 11, 2017, https://www.microsoft.com/en-us/download/details.aspx?id=54284.

(slika 13) i time bazi podataka prepustiti automatsko numeriranje polja primarnog ključa. Ovo svojstvo se najčešće koristi za izradu surogat ključeva koji ne predstavljaju određeno svojstvo entiteta iz stvarnog svijeta pa ih je moguće u potpunosti kontrolirati i možemo biti sigurni da nikada neće doći do gubitka svojstva jednoznačnosti primarnoga ključa. Upravo iz tog razloga kao primarni ključ je postavljen atribut ID.

Za jedinstvenu identifikaciju zapisa predviđeni su atributi Signatura i KlasifikacijskaOznaka. Potonji atribut predstavlja strani ključ koji se povezuje s tablicom tblKlasifikacijskiPlan. Klasifikacijski plan je hijerarhijski prikaz podjele gradiva na dokumentacijske cjeline prema poslovnim aktivnostima Saveza DND. Tablica sadrži atribute Naziv i KlasifikacijskaOznaka, koji ima ulogu primarnog ključa. Iako sadrži samo dva atributa, tablica tblKlasifikacijskiPlan biti će posebno izdvojena u razvijenom korisničkom sučelju i u svakom trenutku dostupna kao pomoćni izvor prilikom izrade zapisa jedinice gradiva.

Atribut Signatura je detaljnija identifikacijska oznaka i trebala bi omogućiti pronalaženje gradiva na razini komada. Sastavljena je kombiniranjem pravila norme ISAD (G) i Uredbe o uredskom poslovanju. Ideja je da sadrži sva tri elementa koja propisuje ISAD (G) kako bi se omogućila razmjena obavijesti na međunarodnoj razini i koristi logiku slaganja identifikacijske oznake koja je opisana u Uredbi. Identifikacijska oznaka u Uredbi sastavljena je od četiri grupe brojčanih oznaka: klasifikacije prema sadržaju, klasifikacije prema vremenu, klasifikacije prema obliku i rednog broja predmeta.[83] Klasifikacija prema sadržaju određuje se prema sadržaju prvog dokumenta u predmetu. Dokumenti koji se primaju svrstavaju se prema upravnom području u glavne grupe, grupe, podgrupe i djelatnosti unutar podgrupe. Navedene vrijednosti odvajaju se kosom crtom (/) od klasifikacije prema vremenu. Klasifikacija prema vremenu određuje godinu otvaranja predmeta, a označava se s posljednja dva broja kalendarske godine u kojoj je određeni predmet otvoren. Iza ta dva broja stavlja se crtica (-) i slijedi brojčana oznaka klasifikacije prema obliku predmeta. Klasifikacija prema obliku brojčana je oznaka skupa predmeta koja po potrebi razrađuje klasifikaciju prema sadržaju na uže cjeline. Odvaja se ravnom crticom (-) od klasifikacije prema vremenu, a označava se s dva ili više brojeva, od 01 pa nadalje, do potrebnog broja. Redni broj predmeta označava redoslijed predmeta unutar klasifikacije prema sadržaju, vremenu, obliku i odvaja se kosom crtom (/) od brojčane oznake klasifikacije prema obliku, a označava se s dva ili više brojeva od 01 pa do potrebnog broja.[84]

[83] "Uredba o uredskom poslovanju," accessed April 12, 2017, http://narodne-novine.nn.hr/clanci/sluzbeni/2009_01_7_171.html.

[84] Ibid.

U našem slučaju, referirati ćemo se na postojeći klasifikacijski plan i vrijednosti atributa KlasifikacijskaOznaka koristiti kao klasifikaciju prema sadržaju. Ako uzmemo u obzir da je klasifikacijski plan razvijen barem do razine podserija, modificirati ćemo identifikacijsku oznaku tako da mjesto klasifikacije prema obliku zauzme redni broj predmeta, a posljednju brojčanu oznaku predstavljati će redni broj komada unutar predmeta. Također, klasifikaciju prema vremenu nije potrebno navoditi jer je za vrijeme sređivanja gradiva odlučeno da će se predmeti slagati prema vremenu njihova nastanka. Konačna vrijednost atributa Signatura mogla bi izgledati ovako:

HR SDND 02.03/01-001
Hrvatska, Savez DND, Sjednice i rad Nadzornog odbora, prvi predmet, prvi komad

Budući da je klasifikacijski plan sređivanja gradiva razrađen prema poslovnim aktivnostima Saveza DND, atribut Naslov će, uz osnovnu ulogu imenovanja jedinice opisa, odrađivati ulogu elementa opisa sadržaja. Tako je odlučeno jer su opisi sadržaja opširni, a cilj baze podataka je da informacije budu sažete i pregledne. Kod atributa VrijemeNastanka je zanimljivo da se kao vrsta podataka ne koristi vrijeme i datum, nego tekstualni podaci (nvarchar), a razlog tome je što za određene zapise je potrebno napisati raspon godina. Domena atributa RazinaOpisa su razine jedinice opisa definirane prema ISAD (G) (slika 12). Atribut Količina također je definiran prema ISAD (G) normi jer koristi samo jedan opisni element (količina i nosač jedinice opisa), dok Pravilnik o zaštiti i čuvanju arhivskog i registraturnog gradiva izvan arhiva propisuje više elemenata (medij i vrsta zapisa, količina te tehničke jedinice). Stoga, umjesto da baza podataka sadrži tri zasebna atributa, unutar jednog atributa će detaljnijim opisom biti zapisane sve potrebne informacije.

Atribut StvarateljID je strani ključ koji povezuje glavnu tablicu tblArhivskoGradivo sa tablicom tblStvaratelj u kojoj se nalaze informacije o stvarateljima gradiva. Tablica se sastoji od atributa definiranih ISAAR (CPF) normom: ID (primarni ključ), Naziv, VrstaEntiteta, OznakaPravneOsobe i VrijemeDjelovanja. ID stvaratelja možemo smatrati oznakom normiranog zapisa, a u polje atributa Naziv bilježi se normirani oblik naziva ili imena stvaratelja koji se opisuje, u skladu sa svim relevantnim nacionalnim i međunarodnim konvencijama i pravilima što ih koristi ustanova koja je izradila normirani zapis.[85] Elementu vrsta entiteta cilj je utvrditi je li entitet koji se opisuje pravna, fizička osoba ili obitelj. Ako je riječ o pravnoj

[85] "ISAAR (CPF): International Standard Archival Authority Record for Corporate Bodies, Persons and Families, 2nd Edition | International Council on Archives," 16.

osobi, cilj je navesti brojčane ili slovno-brojčane oznake kojima se pravna osoba može identificirati. Gdje god je moguće, poželjno je zabilježiti službeni broj ili drugi jedinstveni identifikator (npr. matični broj poduzeća, OIB) te uputiti na tijelo i sustav po kojem je oznaka utvrđena.[86] Vrijeme djelovanja pripada području opisa i mora se bilježiti kao zasebni element. Za pravne osobe potrebno je uključiti nadnevke osnutka, utemeljenja, donošenja zakonskih propisa i prestanka rada. Za fizičke osobe zabilježiti točno ili približno vrijeme rođenja i smrti odnosno, ukoliko su ti datumi nepoznati, najznačajnije nadnevke u njihovu životu.[87] Slično kao i kod opisa sadržaja jedinice gradiva po potrebi je moguće naknadno dodati element koji opisuje povijest stvaratelja.

Čuvanje dokumentacije u sređenom stanju podrazumijeva redovito provođenje postupaka odabiranja i izlučivanja gradiva. Pravilnikom o vrednovanju te postupku odabiranja i izlučivanja arhivskog gradiva utvrđena je, među ostalim, i obveza donošenja općeg popisa gradiva s rokovima čuvanja. Navedeni popis donosi Hrvatsko arhivsko vijeće na prijedlog Hrvatskog državnog arhiva.[88] Popis s rokovima čuvanja služi prvenstveno kao pomoć u odabiranju i izlučivanju gradiva. Možemo ga shvatiti i kao svojevrsni kalendar upravljanja životnim vijekom dokumenta od njegova nastanka do izlučivanja ili predaje nadležnom arhivu. Popis treba pomoći u identifikaciji dokumenata koji imaju dugotrajnu vrijednost, no ujedno i omogućiti pravovremeno izlučivanje dokumentacije koju više nije potrebno čuvati iz poslovnih, pravnih ili drugih razloga. Opći popis gradiva s rokovima čuvanja pomagalo je pri izradi vlastitog popisa gradiva s rokovima čuvanja. Namijenjen je prvenstveno stvarateljima javnog arhivskog i registraturnog gradiva i ostalim kategoriziranim stvarateljima, no koristiti ga mogu svi. Izrada vlastitoga popisa gradiva s rokovima čuvanja preduvjet je za provođenje postupka odabiranja i izlučivanja, a time i za učinkovito i racionalno upravljanje poslovnom i službenom dokumentacijom.[89] Popisi s rokovima čuvanja izrađuju se sukladno razredbenom nacrtu i sadržavaju razredbenu oznaku, naziv ili sadržaj, rok čuvanja, vrstu nosača na kojem se zapis čuva i naznaku postupka s gradivom po isteku roka čuvanja.[90] Atribut RokČuvanja u bazi podataka ujedinjuje element rok čuvanja i naznaku postupka s gradivom po isteku roka čuvanja. Rokovi čuvanja dokumenata računaju se u pravilu od kraja godine u kojoj je nastupio događaj

[86] Ibid., 19.
[87] Ibid., 19–20.
[88] "Pravilnik o vrednovanju te postupku odabiranja i izlučivanja arhivskoga gradiva," accessed April 14, 2017, http://narodne-novine.nn.hr/clanci/sluzbeni/2002_07_90_1476.html.
[89] "Opći popis gradiva s rokovima čuvanja" (Hrvatsko arhivsko vijeće, 2012), 2, http://arhinet.arhiv.hr/_Download/PDF/Opci_popis_gradiva_s_rokovima_cuvanja.pdf.
[90] " Pravilnik o vrednovanju te postupku odabiranja i izlučivanja arhivskoga gradiva."

kojim je rok počeo teći. Mogu se računati i od kraja drugog, kraćeg razdoblja (npr. mjeseca, tromjesečja) ako alati kojima se upravlja dokumentacijom omogućuju pouzdano upravljanje rokovima čuvanja i izlučivanjem kao učestalim rutinskim radnjama. U pravilu se radi o godini (ili drugom razdoblju) u kojoj je spis (predmet, dosje) zaključen, a kod dokumenata kao što su pravilnici, ugovori, odluke i sl. o godini u kojoj su prestali važiti ili su zamijenjeni drugim takvim dokumentom. Takav početak računanja roka čuvanja ovdje se u popisu označava slovom „Z“ iza kojeg se navodi broj godina koliko je spis nakon toga potrebno čuvati. Općenito, oznake koje će se koristiti kao vrijednosti atributa RokČuvanja su:[91]

- **N**= rok čuvanja računa se od isteka godine u kojoj je dokumentacija nastala
- **Z**= rok čuvanja računa se od isteka godine u kojoj je spis zaključen, odnosno u kojoj je dokument (ugovor, odluka, pravilnik i sl.) prestao važiti ili je zamijenjen drugim odgovarajućim dokumentom
- **D**= djelomično odabrati; po isteku roka čuvanja odabire se prema uputama nadležnog arhiva dio dokumentacije za trajno čuvanje. U pravilu se radi o slučajevima gdje se među istovrsnim predmetima i dokumentima mogu naći oni koji se odnose na značajnije događaje, odluke, stvari ili osobe te ih se uslijed toga odabire za trajno čuvanje.
- **I**= izlučiti; po isteku roka dokumentacija se može izlučiti u cjelini, uz pribavljeno odobrenje nadležnog državnog arhiva
- **T** = trajno čuvati; po isteku roka dokumentacija se u cjelini odabire za trajno čuvanje

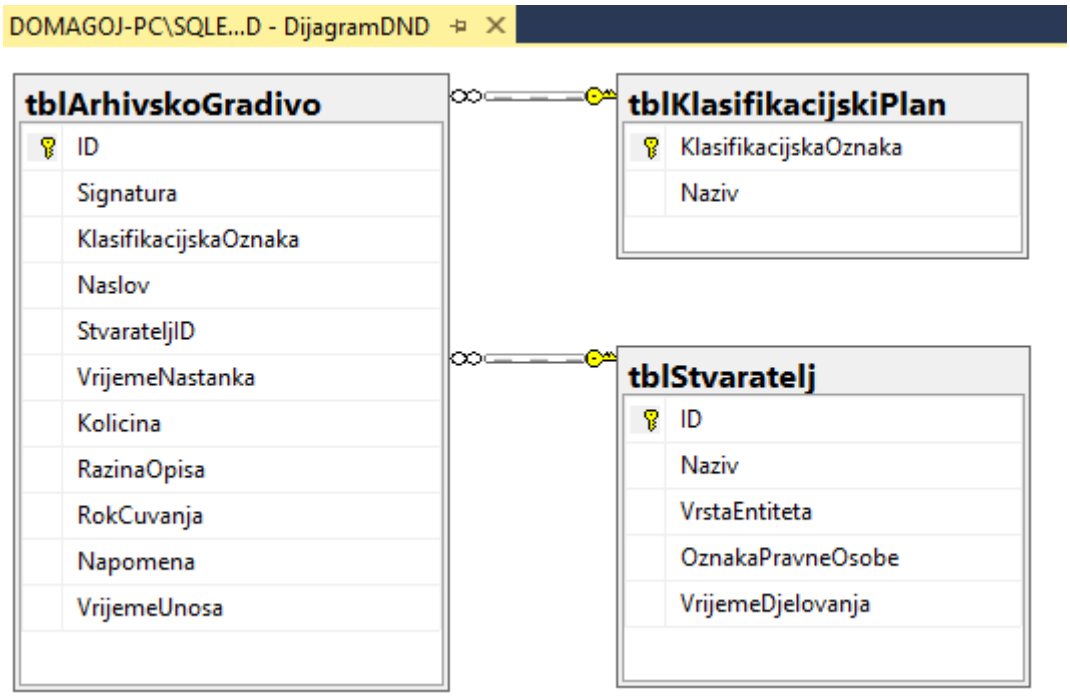
Posljednja dva atributa tablice tblArhivskoGradivo su Napomena i VrijemeUnosa. Atribut Napomena možemo zamisliti kao univerzalni atribut koji pruža obavijesti koje se ne mogu navesti ni u jednom drugom području opisa, odnosno sve dodatne informacije o zapisu koje nije moguće navesti u jednom od atributa baze podataka. Atribut VrijemeUnosa pripada području kontrole opisa i cilj mu je pokazati kada je opis nastao. Treba istaknuti da u tablici nema atributa koji navodi tko je izradio opis jer je predviđeno da taj posao odrađuje jedna osoba. Kako bi se osobi koja izrađuje opis olakšalo bilježenje nadnevka izrade zapisa, atribut VrijemeUnosa je programiran tako da automatski bilježi vrijeme korištenjem funkcije GETDATE(). Funkcija vraća trenutnu vremensku oznaku sustava baze podataka, a ta vrijednost je derivirana iz operacijskog sustava računala na kojem je instanca SQL poslužitelja

[91] “Opći popis gradiva s rokovima čuvanja,” 3.

pokrenuta.[92] Funkciju je moguće jednostavno zadati u prozoru *Column Properties* koristeći svojstvo *Default Value or Binding*, ali primjera radi atribut VrijemeUnosa dodan je u tablicu tblArhivskoGradivo pomoću sljedećeg SQL upita:

```
ALTER TABLE tblArhivskoGradivo ADD VrijemeUnosa datetime NOT NULL DEFAULT GETDATE()
```

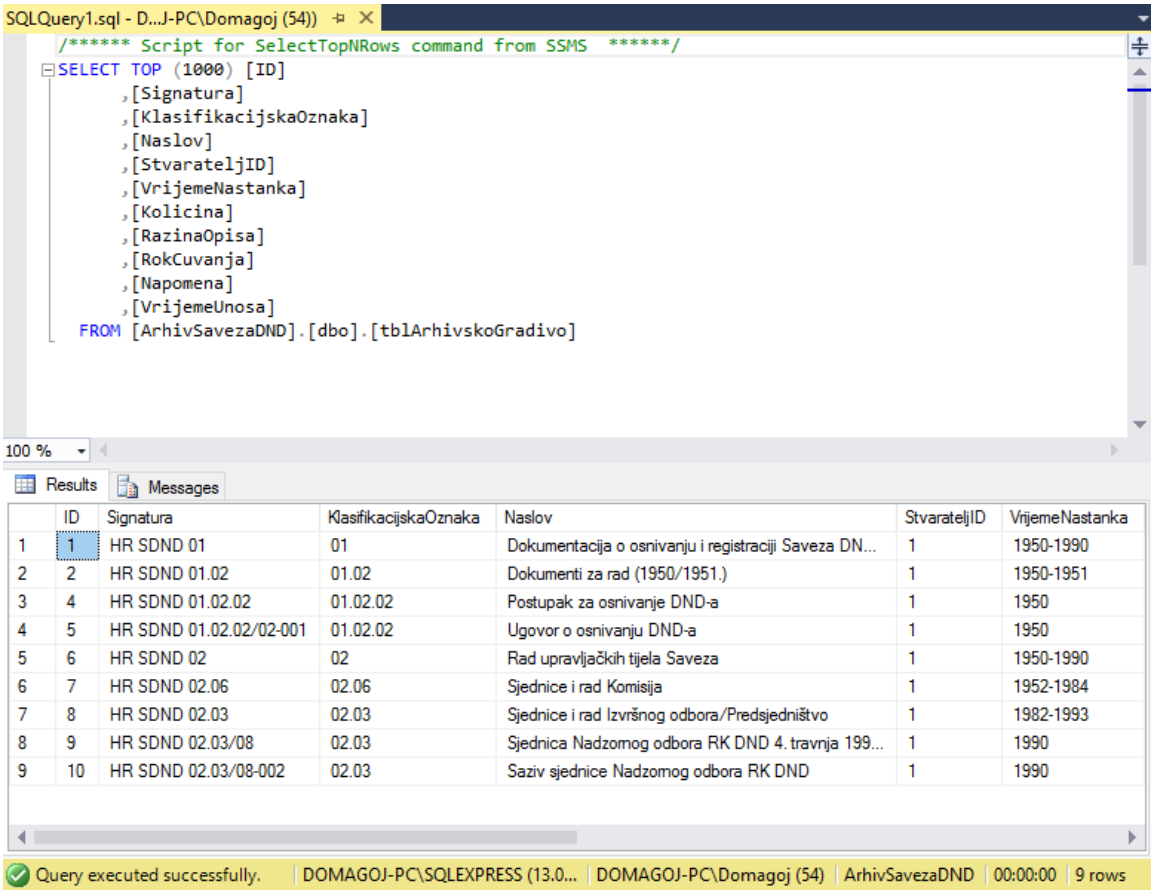
Kada su tablice izrađene i svi atributi definirani potrebno ih je međusobno povezati. Tablice je najlakše povezivati koristeći dijagram baze podataka. Na dijagramu baze podataka pokazane su tablice i njihovi atributi. Za povezivanje dviju tablicu potrebno je odabrati polje stranog ključa i odvući ga iznad tablice s kojom želimo povezati odabranu tablicu. U našem slučaju, strane ključeve KlasifikacijskaOznaka i StvarateljID tablice tblArhivskoGradivo potrebno je povezati sa primarnim ključevima KlasifikacijskaOznaka i ID tablica tblKlasifikacijskiPlan i tblStvaratelj. Grafički prikaz povezanosti tablica u dijagramu vidljiv je na slici 14.



Slika 14. Dijagram baze podataka

Povezivanjem tablica baza podataka postaje u potpunosti funkcionalna i spremna za unos podataka. Primjer ispisa unesenih podataka u SSMS-u prikazan je na slici 15.

[92] "GETDATE (Transact-SQL)," accessed April 16, 2017, https://docs.microsoft.com/en-us/sql/t-sql/functions/getdate-transact-sql.



	ID	Signatura	KlasifikacijskaOznaka	Naslov	StvarateljID	VrijemeNastanka
1	1	HR SDND 01	01	Dokumentacija o osnivanju i registraciji Saveza DN...	1	1950-1990
2	2	HR SDND 01.02	01.02	Dokumenti za rad (1950/1951.)	1	1950-1951
3	4	HR SDND 01.02.02	01.02.02	Postupak za osnivanje DND-a	1	1950
4	5	HR SDND 01.02.02/02-001	01.02.02	Ugovor o osnivanju DND-a	1	1950
5	6	HR SDND 02	02	Rad upravljačkih tijela Saveza	1	1950-1990
6	7	HR SDND 02.06	02.06	Sjednice i rad Komisija	1	1952-1984
7	8	HR SDND 02.03	02.03	Sjednice i rad Izvršnog odbora/Predsjedništvo	1	1982-1993
8	9	HR SDND 02.03/08	02.03	Sjednica Nadzornog odbora RK DND 4. travnja 199...	1	1990
9	10	HR SDND 02.03/08-002	02.03	Saziv sjednice Nadzornog odbora RK DND	1	1990

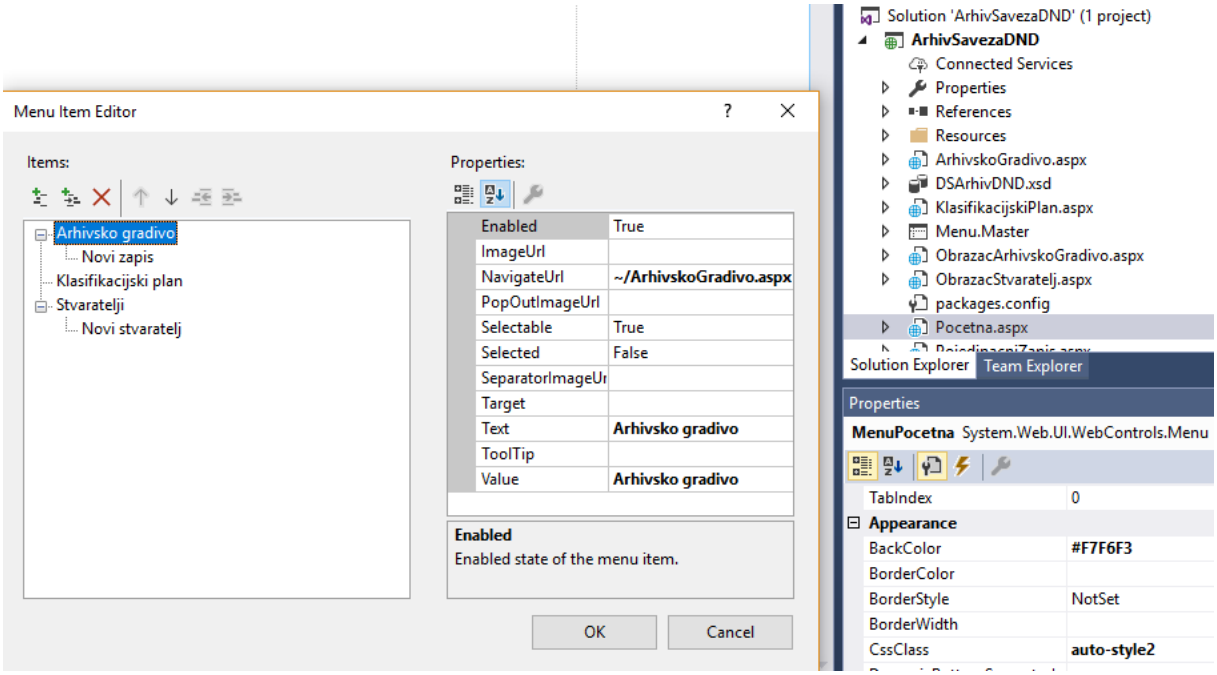
Slika 15. Ispis podataka u SSMS-u

3.4 Izrada korisničkog sučelja

Na slici 15 jasno je vidljivo da je prozor za ispis podataka premalen i da podaci nisu u potpunosti pregledni. Stoga, za potpunu vidljivost i općenito jednostavnije upravljanje podacima koristiti će se posebno izrađeno korisničko sučelje. Sučelje je razvijano u integriranom razvojnom okruženju, Microsoft Visual Studiju (Community verzija). Riječ je o softverskoj platformi koja na jednom mjestu programerima pruža sveobuhvatan skup alata za razvoj softvera.[93] U našem slučaju, sučelje će biti izrađeno korištenjem ASP.NET web obrazaca. Obrasci će većinom biti oblikovani u dizajnerskom prozoru (engl. *Designer*) pomoću kontrola iz alatnog okvira (engl. *Toolbox*) čiji kod se automatski generira. Uz korištenje kontrola, dodatne funkcionalnosti sučelja biti će napisane C# jezikom u prozoru predviđenom za kodiranje.

[93] "Best Integrated Development Environment (IDE) Software in 2017," *G2 Crowd*, accessed April 25, 2017, https://www.g2crowd.com/categories/integrated-development-environment-ide.

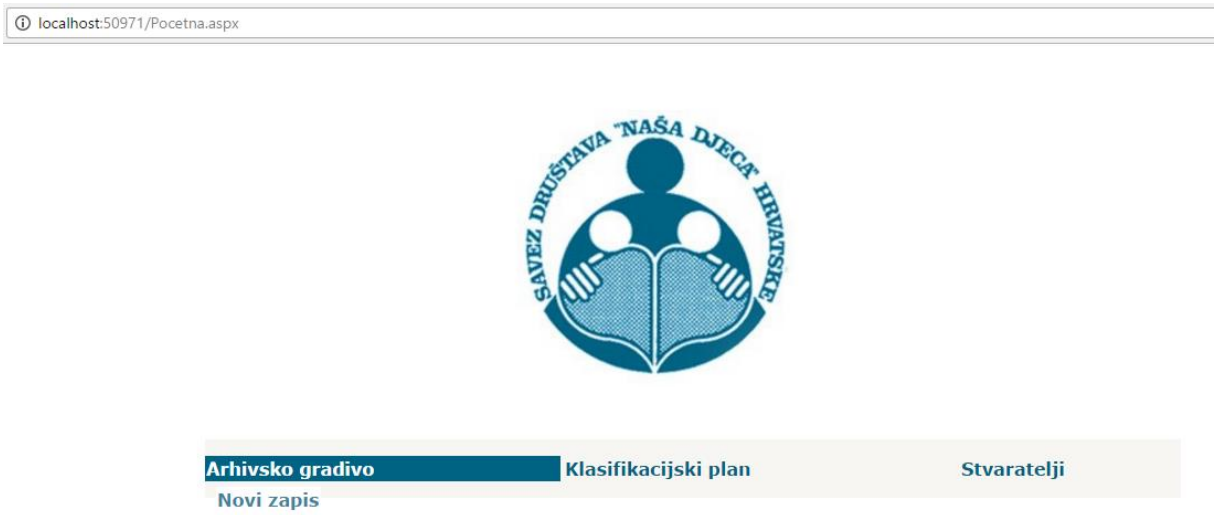
Korisničko sučelje sastoji se od početne stranice i stranica s popisom arhivskoga gradiva, klasifikacijskim planom, popisom stvaratelja te obrazaca za unos novih jedinica gradiva i novih stvaratelja. Početna stranica minimalistički je dizajnirana, sadrži samo logotip Saveza (kontrola *Image*) i izbornik (*Menu* iz skupine kontrola za navigaciju). Logotip je moguće dodati na početnu stranicu jednostavnim povezivanjem na URL adresu već postojeće slike, no zbog mogućnosti njenog brisanja ili nedostupnosti nakon određenog vremenskog perioda kao bolje rješenje nameće se spremanje slike kao resursa unutar same aplikacije gdje će uvijek biti aktivna i dostupna. Izbornik je isto tako moguće preuzeti sa određene stranice povezivanjem na važeću *sitemap* ili XML datoteku. Budući da sadrži samo najrelevantnije sastavnice, izbornik korisničkog sučelja biti će samostalno oblikovan. Prvi korak je u dizajnerskom prozoru definirati elemente izbornika kao što je prikazano na slici 16.



Slika 16. Definiranje sastavnica izbornika

Elemente je potrebno jasno imenovati i hijerarhijski rasporediti. Tako su definirani Arhivsko gradivo, Klasifikacijski plan i Stvaratelji kao elementi više razine, a Novi zapis i Novi stvaratelj kao elementi niže razine u izborniku. Elemente je potrebno povezati s ostalim stranicama korisničkog sučelja pomoću svojstva *NavigateUrl*, npr. adresa ~/ArhivskoGradivo.aspx upućuje na stranicu s popisom arhivskog gradiva. Izgled izbornika i ostali detalji definiraju se u prozoru sa svojstvima (engl. *Properties*) koji se nalazi u donjem desnom kutu Visual Studija (slika 16). Zanimljivo je istaknuti da prilikom odabira boje slova ili pozadine pomoću alata za prepoznavanje boje moguće je dobiti istu nijansu kakvu ima

logotip Saveza. Potpuno uređena početna stranica korisničkog sučelja prikazana je na slici 17. Bitno je napomenuti da se cijela aplikacija pokreće na poslužitelju SQL Server Express, a korisničkim sučeljem se kreće putem web preglednika. Primjerice, početna stranica se pokreće na adresi: http://localhost:50971/Pocetna.aspx.



Slika 17. Početna stranica

Logotip Saveza i izbornik osmišljeni su kao standardni dio ostalih stranica korisničkog sučelja. Kako bi se ubrzala izrada stranica i ostvarila njihova dosljednost navedeni elementi su oblikovani u glavnoj stranici web obrazaca (engl. *Web Forms Master Page*). Glavna stranica definira izgled i standardno ponašanje za sve stranice u aplikaciji. Kada korisnik zatraži određenu stranicu sa sadržajem, ona se spaja s glavnom stranicom i kao produkt se dobiva kombinacija predloška glavne stranice i sadržaja stranice sa sadržajem.[94] Stoga, napravljeni predložak logotipa i izbornika biti će jednako prikazan i imati će iste funkcionalnosti na svim ostalim stranicama korisničkog sučelja. Jedini izuzetak je početna stranica koja predstavlja obični web obrazac, dok ostale stranice su web obrazac s glavnom stranicom. Dodatkom izbornika na sve stranice moguće je u svakom trenutku pristupiti bilo kojoj od sastavnica sučelja. Za povratak na početnu stranicu iskorišten je logotip Saveza kojem je dodijeljena uloga gumba (kontrola *Image Button*). Klikom na logotip poziva se metoda koja upućuje na početnu stranicu:

[94] "ASP.NET Master Pages," accessed April 27, 2017, https://msdn.microsoft.com/en-us/library/wtxbf3hh.aspx.

```
protected void imgbtnLogo_Click(object sender, ImageClickEventArgs e)
            {
               Response.Redirect("Pocetna.aspx");
            }
```

Stranice Arhivsko gradivo, Klasifikacijski plan i Stvaratelji podatke prikazuju kontrolom prikaza rešetke (engl. *grid view*). Razlog zbog kojeg se koristi je što ima svojstvo povezivanja sa izvorom podataka i prikazivanja njegovih vrijednosti u tablici. Stoga, prikaz rešetke je kontrola koja izravno povezuje korisničko sučelje s bazom podataka arhiva Saveza DND i prikazuje podatke koji su pohranjeni u njoj. Za pravilno povezivanje potrebno je stvoriti novu vezu. Izvor podataka je SQL baza podataka, mora se upisati ime poslužitelja i ime baze podataka, koje Visual Studio automatski prepozna ako je ime poslužitelja ispravno napisano. Nakon stvaranja veze jedino preostaje oblikovanje SQL upita kojim se odabiru atributi tablice (ili više tablica) čiji će podaci biti prikazani. Dobiveni prikaz moguće je uređivati mijenjanjem njegovih svojstava, a cilj je biti u skladu sa svim stranicama korisničkog sučelja. Potrebno je uskladiti boju pozadine zaglavlja i redaka s logotipom Saveza. Moguće je mijenjati nazive stupaca što je važno jer prikaz rešetke preuzima nazive atributa iz baze podataka koji nemaju prirodan izgled, odnosno naziv je niz znakova bez razmaka gdje se nova riječ u nizu označava velikim slovom, a takav prikaz naziva u korisničkom sučelju nije vizualno prihvatljiv. Također, mogu se uređivati postojeći stupci pa tako ako se dogodi da određeni stupac u jednom trenutku nije potrebno prikazati moguće ga je privremeno učiniti nevidljivim. Vrlo korisno svojstvo je sortiranje zapisa prema određenom stupcu koje omogućuje korisniku sučelja da pretražuje gradivo prema različitim parametrima, npr. signaturi, naslovu ili vremenu nastanka gradiva. Završna verzija stranice s popisom arhivskoga gradiva prikazana je na slici 18.

Arhivsko gradivo Klasifikacijski plan Stvaratelji

ID	Signatura	Naslov	Stvaratelj	Vrijeme nastanka	Razina opisa	Količina	Rok čuvanja	Napomena	Vrijeme unosa			
1	HR SDND 01	Dokumentacija o osnivanju i registraciji Saveza DND Hrvatske	Savez Društava Naša djeca Hrvatske	1950-1990	serija	2 registratora; 0,5 dužnih metara	T		16.4.2017. 15:41:35	Obriši	Promijeni	Pojedinačni zapis
2	HR SDND 01.02	Dokumenti za rad (1950/1951.)	Savez Društava Naša djeca Hrvatske	1950-1951	podserija	1 registrator, 20 omota	T		16.4.2017. 15:45:35	Obriši	Promijeni	Pojedinačni zapis
4	HR SDND 01.02.02	Postupak za osnivanje DND-a	Savez Društava Naša djeca Hrvatske	1950		1 omot	T		16.4.2017. 15:48:00	Obriši	Promijeni	Pojedinačni zapis
5	HR SDND 01.02.02/02-001	Ugovor o osnivanju DND-a	Savez Društava Naša djeca Hrvatske	1950	komad	1 komad, ugovor	T		16.4.2017. 15:54:42	Obriši	Promijeni	Pojedinačni zapis

Slika 18. Stranica Arhivsko gradivo

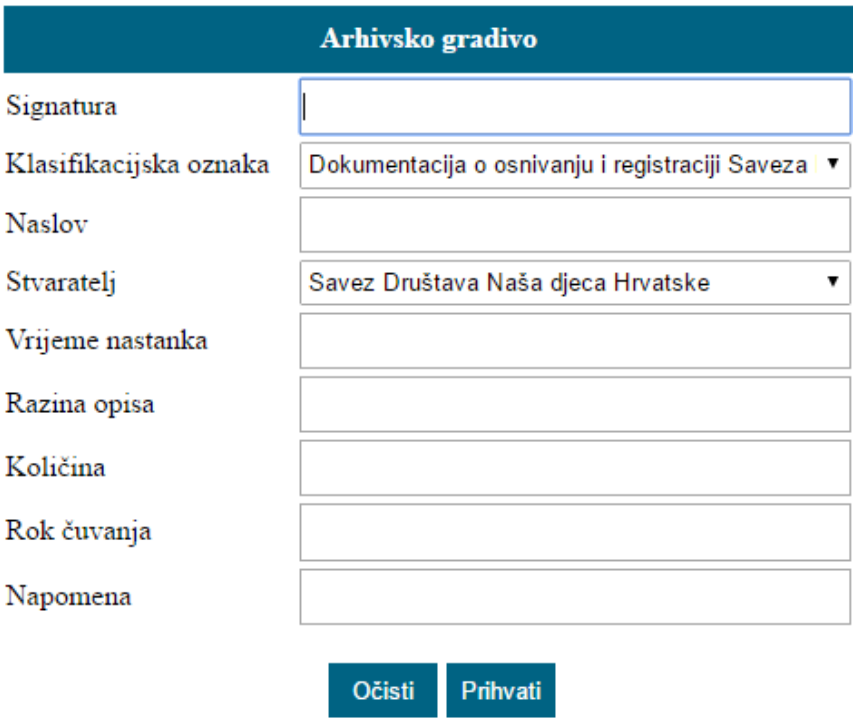
Dodatna pogodnost prikaza rešetke je mogućnost dodavanja novih stupaca s poveznicama koje izvršavaju SQL naredbe SELECT, DELETE, INSERT i UPDATE nad pojedinim retkom. Korisničkom sučelju Saveza dodane su poveznice Obriši, Promijeni i Pojedinačni zapis (slika 18). Poveznicu za brisanje zapisa najjednostavnije je dodati jer je već definirana, a jedino je potrebno obratiti pažnju na usklađenost poveznice s ograničenjima definiranim u bazi podataka jer u suprotnom naredba brisanja podataka neće se izvršiti i sustav će javiti grešku. Poveznice za unos i ažuriranje podataka su složenije jer naredbu ne izvršavaju automatski, već upućuju na posebno izrađene obrasce za unos jedinica arhivskoga gradiva i stvaratelja. Ono što je važno napomenuti jest da za povezivanje s obrascima se ne koristi svojstvo *NavigateUrl*, već *DataNavigateUrlField* svojstvo. Svojstvo se koristi za određivanje polja izvora podataka koje se spaja s URL adresom poveznice. Na taj način svaka poveznica u stupcu ima različitu URL adresu.[95] U paru s navedenim svojstvom može se koristiti *DataNavigateUrlFormatString* svojstvo koje određuje format kako su URL adrese poveznica prikazane.[96] Na primjer, u korisničkom sučelju spojiti ćemo jedinstveno polje ID (iz tablice tblArhivskoGradivo) s poveznicom Promijeni pa će tako prilikom ažuriranja zapisa čiji ID je jedan poveznica upućivati na obrazac s već ispunjenim podacima navedenog zapisa (ObrazacArhivskoGradivo.aspx?ArhivskoGradivoID=1). Cijeli postupak izvršenja naredbi unosa i ažuriranja biti će samostalno kodiran i detaljnije prikaza u narednim odlomcima.

Obrazac za unos i ažuriranje jedinica arhivskoga gradiva (dodijeljen mu je naziv ObrazacArhivskoGradivo) izrađen je pomoću tablice iz skupine HTML kontrola. Navedena vrsta tablice predstavlja idealan predložak za izradu jer njezin okvir (redovi i stupci) je vidljiv u dizajnerskom prozoru, a u pregledniku nije. Sastoji se od stupca s nazivima atributa gradiva i stupca s kontrolama koje omogućuju unos i odabir podataka. Kontrole koje se koriste su tekstualni okvir (engl. *TextBox*) i padajući popis (engl. *DropDownList*). Kao što sam naziv sugerira, prvo spomenuta kontrola omogućuje unos teksta i svaki atribut ima vlastiti okvir. Budući da u bazi podataka postoje samostalne tablice sa podacima o stvarateljima i klasifikacijskom planu, iskoristiti ćemo kontrolu padajući popis koja omogućuje odabir jednoga naziva stvaratelja ili klasifikacijske oznake iz popisa. Popis se povezuje s tablicom u bazi podataka na isti način kao prikaz rešetke. Uz navedene kontrole obrascu je još potrebno dodati gumbe koji će obrisati ili prihvatiti unesene podatke. Ono što se mora posebno istaknuti je

[95] "HyperLinkColumn.DataNavigateUrlField Property (System.Web.UI.WebControls)," accessed April 28, 2017, https://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.hyperlinkcolumn.datanavigateurlfield(v=vs.110).aspx.

[96] Ibid.

potreba za pravilnim imenovanjem kontrola kako bi ih se moglo jednoznačno identificirati prilikom kodiranja pa tako kontroli tekstualni okvir se dodaje prefiks txt, padajućem popisu ddl i gumbima btn. Dizajn obrasca prikazan je na slici 19.

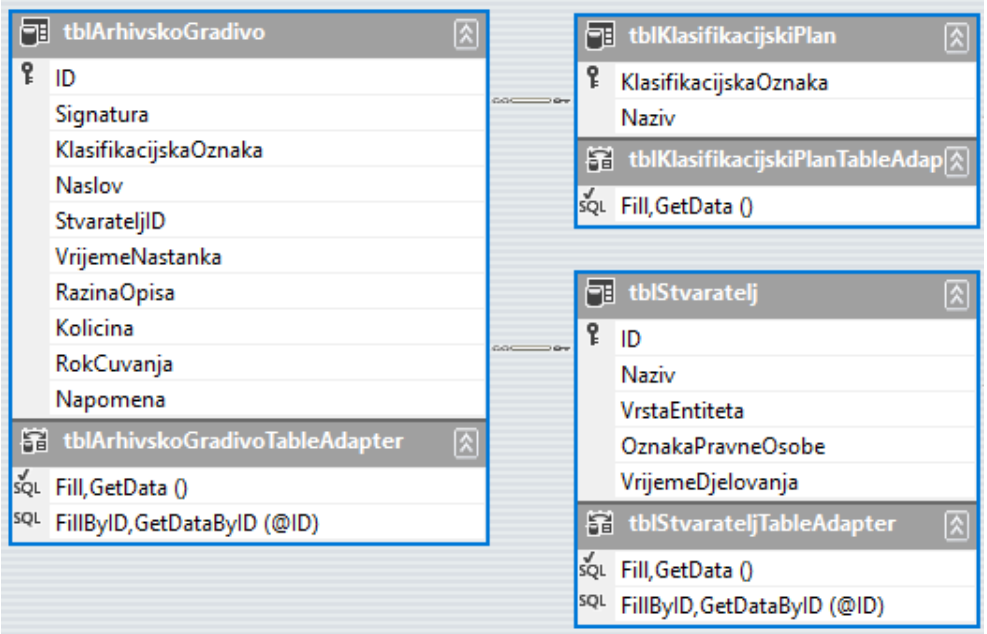


Slika 19. ObrazacArhivskoGradivo

Prije početka programiranja naredbi unosa i ažuriranja podataka potrebno je izraditi klasu skup podataka (engl. *DataSet*) naziva DSArhivDND. Riječ je o memorijskoj reprezentaciji podataka koja omogućuje dosljedan relacijski model programiranja neovisno o izvoru podataka. Ova klasa predstavlja cjelovit skup podataka koji uključuje tablice, ograničenja i odnose među tablicama. Budući da ne ovisi o izvoru podataka, skup podataka može uključivati lokalne podatke aplikacije i podatke iz više različitih izvora.[97] Interakcija s postojećim izvorima podataka kontrolira se pomoću adaptera tablica (engl. *TableAdapters*). Adapter tablica osigurava komunikaciju između aplikacije i baze podataka. Preciznije, adapter se povezuje s bazom podataka, izvršava upit ili pohranjenu proceduru te vraća novu tablicu (metoda *GetData()*) ili ispunjava podacima već postojeću tablicu unutar skupa podataka (metoda *Fill()*). Također, adapteri tablica se koriste za slanje ažuriranih podataka iz aplikacije natrag u bazu podataka.[98]

[97] "Populating a DataSet from a DataAdapter," accessed April 29, 2017, https://msdn.microsoft.com/en-us/library/bh8kx08z(v=vs.110).aspx.

[98] "TableAdapter Overview," accessed April 29, 2017, https://msdn.microsoft.com/en-us/library/bz9tthwx(v=vs.80).aspx.



Slika 20. Skup podataka DSSavezDND

Budući da se skup podataka DSSavezDND povezuje s bazom podataka arhiva Saveza, njegov prikaz u dizajnerskom prozoru praktički je identičan dijagramu baze podataka u SSMS-u (slika 20). Ključni postupak koji se mora napraviti u skupu podataka je izrada upita koji odabire podatke na temelju atributa ID i dodaje metodu *FillByID()*. Time se ostvaruje mogućnost vraćanja samo jednog retka iz tablice i prikaza njegovog sadržaja u obrascu kada korisnik želi ažurirati pojedini zapis. Primjer kako izgleda upit za adapter tablice tblArhivskoGradivo:

```
SELECT    ID, Signatura, KlasifikacijskaOznaka, Naslov, StvarateljID,
          VrijemeNastanka, RazinaOpisa, Kolicina, RokCuvanja, Napomena
FROM      tblArhivskoGradivo
WHERE     ID = @ID
```

U ovom trenutku sve je spremno za početak programiranja naredbi za unos i ažuriranje jedinica arhivskoga gradiva. Prvo je potrebno definirati redak na kojem se trenutno radi (arhivskoGradivo). Zatim se kreira metoda za dohvaćanje gradiva (dohvatiArhivskoGradivo). U metodi prvo se stvara novi objekt (tablicaArhivskoGradivo) iz klase tablica podataka koja je dio skupa podataka DSSavezDND. Ako je riječ od naredbi za unos podataka potrebno je postaviti uvjet: ako je vrijednost atributa ID iz URL adrese definirane svojstvom *DataNavigateUrlFormatString* jednaka nul vrijednosti, retku arhivskoGradivo dodjeljuje se vrijednost novog praznog retka unutar objekta tablicaArhivskoGradivo. Atributu ID se zadaje vrijednost -1 jer pozitivne vrijednosti mogu biti dodijeljene već postojećim zapisima. Kod

naredbe za ažuriranje zapisa stvara se i objekt klase adapter tablice skupa podataka DSSavezDND. Također, potrebno je definirati varijablu arhivskoGradivoID kojoj se dodjeljuje vrijednost atributa ID preuzeta iz URL adrese. Inače, do navedene vrijednosti dolazi se korištenjem svojstva *Request.Params ()*, a budući da su zatraženi parametri niz znakova potrebno ih je pretvoriti u brojčane podatke metodom *Parse()*. Definirana varijabla i objekt klase tablice koriste se kao parametri metode *FillByID()* kojom ispunjavamo objekt adapterArhivskoGradivo. Na kraju retku na kojem radimo dodjeljuje se vrijednost prvog retka objekta tablicaArhivskoGradivo. Kod metode dohvatiArhivskoGradivo je sljedeći:

```
DSArhivDND.tblArhivskoGradivoRow arhivskoGradivo;

private void dohvatiArhivskoGradivo()
{
	DSArhivDND.tblArhivskoGradivoDataTable tablicaArhivskoGradivo = new
	DSArhivDND.tblArhivskoGradivoDataTable();

	if (Request.Params["ArhivskoGradivoID"] == null)
	{
		arhivskoGradivo = tablicaArhivskoGradivo.NewtblArhivskoGradivoRow();
		arhivskoGradivo.ID = -1;
	}

	else
	{
		DSArhivDNDTableAdapters.tblArhivskoGradivoTableAdapter
		adapterArhivskoGradivo = new
		DSArhivDNDTableAdapters.tblArhivskoGradivoTableAdapter();

		int arhivskoGradivoID = int.Parse(Request.Params["ArhivskoGradivoID"]);

		adapterArhivskoGradivo.FillByID(tablicaArhivskoGradivo,
		arhivskoGradivoID);

		arhivskoGradivo =
		(DSArhivDND.tblArhivskoGradivoRow)tablicaArhivskoGradivo.Rows[0];
	}
```

Iduća metoda koju je potrebno kreirati koristi se za prikaz podataka (PrikaziPodatke) i jedina je metoda koja se odnosi samo na naredbu za ažuriranje zapisa. U metodi se kontrolama obrasca i njihovom svojstvu tekst (engl. *Text*), odnosno odabrana vrijednost (engl. *Selected Value*) za padajući popis, dodjeljuju vrijednosti atributa retka arhivskoGradivo. Atribut StvarateljID je potrebno pretvoriti u niz znakova metodom *ToString()*. Linije koda za prikaz podataka su sljedeće:

```
private void prikaziPodatke()
{
	if (arhivskoGradivo.ID == -1) return;
	txtSignatura.Text = arhivskoGradivo.Signatura;
	ddlKlasifikacijskaOznaka.SelectedValue = arhivskoGradivo.KlasifikacijskaOznaka;
	txtNaslov.Text = arhivskoGradivo.Naslov;
```

```
        txtVrijemeNastanka.Text = arhivskoGradivo.VrijemeNastanka;
        txtRazinaOpisa.Text = arhivskoGradivo.RazinaOpisa;
        txtKolicina.Text = arhivskoGradivo.Kolicina;
        ddlStvarateljID.SelectedValue = arhivskoGradivo.StvarateljID.ToString();
        txtRokCuvanja.Text = arhivskoGradivo.RokCuvanja;
        txtNapomena.Text = arhivskoGradivo.Napomena;
}
```

Treća metoda definira se s ciljem da pokupi podatke s obrasca (pokupiPodatkeSForme). Koristi iste vrijednosti kao i prethodna metoda, samo su uloge zamijenjene. Tako se atributima retka arhivskoGradivo dodjeljuju vrijednosti kontrola obrasca. U ovom slučaju atribut StvarateljID treba pretvoriti u brojčanu vrsta podataka korištenjem metode *Parse()*. Kod za ovu metodu je sljedeći:

```
private void pokupiPodatkeSForme()
{
        arhivskoGradivo.Signatura = txtSignatura.Text;
        arhivskoGradivo.KlasifikacijskaOznaka = ddlKlasifikacijskaOznaka.SelectedValue;
        arhivskoGradivo.Naslov = txtNaslov.Text;
        arhivskoGradivo.VrijemeNastanka = txtVrijemeNastanka.Text;
        arhivskoGradivo.RazinaOpisa = txtRazinaOpisa.Text;
        arhivskoGradivo.Kolicina = txtKolicina.Text;
        arhivskoGradivo.StvarateljID = int.Parse(ddlStvarateljID.SelectedValue);
        arhivskoGradivo.RokCuvanja = txtRokCuvanja.Text;
        arhivskoGradivo.Napomena = txtNapomena.Text;
}
```

Za izvršavanje promjena potreban nam je ranije dizajnirani gumb Prihvati i njegov događaj *Click*. Visual Studio automatski generira naziv metode gumba (btnPrihvati_Click). U metodi se prvo pozivaju metode dohvatiArhivskoGradivo i pokupiPodatkeSForme te se stvara novi objekt iz klase adapter tablice. U slučaju naredbe za unos podataka postavljamo uvjet: ako arhivskoGradivo.ID je jednako -1 u adapter tablicu će biti unesene vrijednosti definirane u metodi pokupiPodatkeSForme. Nakon što se naredba izvrši podaci u obrascu će biti izbrisani što omogućuje unos više novih zapisa bez odlaska na neku drugu stranicu korisničkog sučelja. Potrebno je posebno istaknuti da se obrascu za unos novih jedinica arhivskoga gradiva pristupa putem poveznice Novi zapis koja se nalazi u glavnom izborniku. Kod naredbe za ažuriranje zapisa pozivamo istoimenu metodu koja u adapter tablice pohranjuje promjene napravljene na dohvaćenom retku arhivskoGradivo. U ovom slučaju, na kraju se izvršava preusmjeravanje na stranicu s popisom arhivskoga gradiva. Kod metode btnPrihvati_Click je sljedeći:

```
protected void btnPrihvati_Click(object sender, EventArgs e)
{
        dohvatiArhivskoGradivo();

        pokupiPodatkeSForme();
```

```
        DSArhivDNDTableAdapters.tblArhivskoGradivoTableAdapter adapterArhivskoGradivo =
        new DSArhivDNDTableAdapters.tblArhivskoGradivoTableAdapter();

        if (arhivskoGradivo.ID == -1)
        {
                adapterArhivskoGradivo.Insert(

                        arhivskoGradivo.Signatura,
                        arhivskoGradivo.KlasifikacijskaOznaka,
                        arhivskoGradivo.Naslov,
                        arhivskoGradivo.StvarateljID,
                        arhivskoGradivo.VrijemeNastanka,
                        arhivskoGradivo.RazinaOpisa,
                        arhivskoGradivo.Kolicina,
                        arhivskoGradivo.RokCuvanja,
                        arhivskoGradivo.Napomena);

                        txtSignatura.Text = "";
                        ddlKlasifikacijskaOznaka.DataBind();
                        txtNaslov.Text = "";
                        ddlStvarateljID.DataBind();
                        txtVrijemeNastanka.Text = "";
                        txtRazinaOpisa.Text = "";
                        txtKolicina.Text = "";
                        txtRokCuvanja.Text = "";
                        txtNapomena.Text = "";
        }

        else
        {
                adapterArhivskoGradivo.Update(arhivskoGradivo);

                Response.Redirect("ArhivskoGradivo.aspx");
        }
}
```

Za slučaj da podaci u obrascu nisu zadovoljavajući definira se metoda gumba koja briše sve podatke s obrasca (btnOcisti_Click). U metodi je potrebno svojstvu tekst kontrola obrasca dodijeliti prazan niz znakova, a za kontrolu padajući popis koristimo metodu *DataBind()*. Kod metode je sljedeći:

```
protected void btnOcisti_Click(object sender, EventArgs e)
{
        txtSignatura.Text = "";
        ddlKlasifikacijskaOznaka.DataBind();
        txtNaslov.Text = "";
        txtVrijemeNastanka.Text = "";
        txtRazinaOpisa.Text = "";
        txtKolicina.Text = "";
        ddlStvarateljID.DataBind();
        txtRokCuvanja.Text = "";
        txtNapomena.Text = "";
}
```

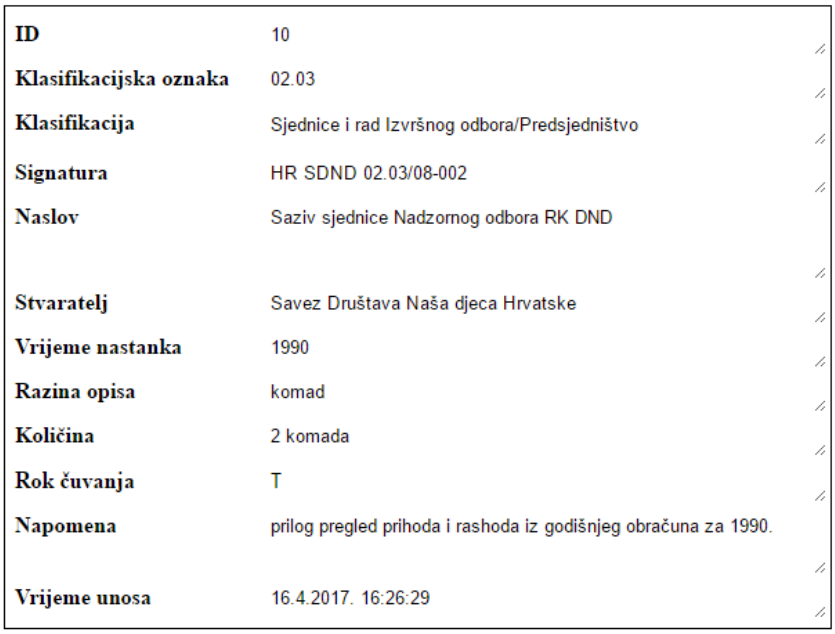
Osim obrasca za unos i ažuriranje zapisa predviđena je izrada obrasca koji prikazuje detaljnije informacije pojedinačnog zapisa (obrazac PojedinačniZapis). Riječ je o obrascu s

vlastitom stranicom u korisničkom sučelju na kojem nije moguće vršiti promijene, a pristupa mu se preko poveznice na stranici s popisom arhivskoga gradiva. Ideja je da se korisniku omogući prikaz podataka samo jednog zapisa i ako je za poslovanje potrebno omogući ispis tog sadržaja (npr. uredi Saveza su na četvrtkom katu zgrade, a gradivo se nalazi u podrumu bez pristupa računalu). Izgledom i količinom informacija može se usporediti s kataložnim zapisom. Ono po čemu se razlikuje od ostalih obrazaca je što sadrži informacije iz svih tablica baze podataka. Stoga je iz praktičnih razloga za ovaj obrazac izrađen dodatan adapter tablice u skupu podataka DSSavezDND. Potrebno je izraditi složeni upit koji osim atributa iz tablice tblArhivskoGradivo dohvaća i atribute Naziv iz tablica tblKlasifikacijskiPlan i tblStvaratelj:

```
SELECT    tblArhivskoGradivo.ID,tblArhivskoGradivo.Signatura,
          tblArhivskoGradivo.KlasifikacijskaOznaka,
          tblKlasifikacijskiPlan.Naziv AS Klasifikacija,
          tblArhivskoGradivo.Naslov, tblArhivskoGradivo.StvarateljID,
          tblStvaratelj.Naziv AS Stvaratelj,
          tblArhivskoGradivo.VrijemeNastanka,
          tblArhivskoGradivo.RazinaOpisa, tblArhivskoGradivo.Kolicina,
          tblArhivskoGradivo.RokCuvanja, tblArhivskoGradivo.Napomena,
          tblArhivskoGradivo.VrijemeUnosa

FROM      tblArhivskoGradivo INNER JOIN
          tblKlasifikacijskiPlan ON tblArhivskoGradivo.KlasifikacijskaOznaka =
          tblKlasifikacijskiPlan.KlasifikacijskaOznaka INNER JOIN tblStvaratelj
          ON tblArhivskoGradivo.StvarateljID = tblStvaratelj.ID

WHERE     tblArhivskoGradivo.ID = @ID
```



Slika 21. Pojedinačni zapis

Za dizajn obrasca korišteni su HTML tablica i ovom slučaju samo kontrole tekstualnog okvira. Bitno je istaknuti da se granice kontrola ne vide i da je tekst svojstvo namijenjeno samo za čitanje (engl. *read-only*). Od koda se koriste samo metode za dohvaćanje i prikaz gradiva. Primjer pojedinačnog zapisa prikazan je na slici 21.

Stranica s popisom stvaratelja dizajnirana je prema istim načelima kao i stranica s arhivskim gradivom. Jedini izuzetak je nedostatak poveznice na obrazac s pojedinačnim zapisima stvaratelja koji nije potreban jer prikaz rešetke dohvaća mali broj atributa tablice. Konačna verzija stranice prikazana je na slici 22.

Arhivsko gradivo Klasifikacijski plan Stvaratelji

ID	Naziv	Vrijeme djelovanja	Vrsta entiteta	Oznaka pravne osobe		
1	Savez Društava Naša djeca Hrvatske	1950-	pravna osoba	OIB: 82683059627	Obriši	Promijeni
2	DND Kutjevo	1980-	pravna osoba	OIB: 32764436188	Obriši	Promijeni
3	DND Našice	1952-	pravna osoba	OIB: 00206475967	Obriši	Promijeni
5	DND Konjščina	1950-	pravna osoba		Obriši	Promijeni

Slika 22. Stranica Stvaratelji

Jednaka situacija je i s obrascem za unos i ažuriranje podataka o stvarateljima. Funkcionalnosti se kodiraju definiranjem istih metoda koje se koriste kod obrasca za arhivsko gradivo. Jedina iznimka je što se u obrascu stvaratelja ne koristi kontrola padajućeg popisa, već samo tekstualni okvir. Također, potrebno je istaknuti da se novi zapisi unose putem poveznice Novi stvaratelji u glavnom izborniku. Obrazac za unos i ažuriranje podataka o stvarateljima s dohvaćenim podacima prikazan je na slici 23.

Stvaratelj

Naziv	Savez Društava Naša djeca Hrvatske
Vrsta entiteta	pravna osoba
Oznaka pravne osobe	OIB: 82683059627
Vrijeme djelovanja	1950-

Očisti Prihvati

Slika 23. Obrazac Stvaratelj

Stranica koja prikazuje klasifikacijski plan također je uređena u skladu s ostatkom korisničkog sučelja. Međutim, na njoj se u sučelju ne mogu izvršavati nikakve operacije. Stranica je zamišljena da služi samo kao pomoćni izvor na koji se korisnik može referirati. Pretpostavka je da jednom definirani plan je konačan, ali u slučaju da se pojavi potreba za izmjenom ili ažuriranjem pojedinih stavki, intervencija će biti odrađena u samoj bazi podataka, odnosno u SSMS sučelju. Konačna verzija stranice prikazana je na slici 24.

Arhivsko gradivo | Klasifikacijski plan | Stvaratelji

Klasifikacijska oznaka	Naziv
01	Dokumentacija o osnivanju i registraciji Saveza DND Hrvatske
01.01	Inicijative za osnivanje, proglasi i druge odluke
01.01.01	Prva zemaljska konferencija
01.01.02	Sekretarijat Savjeta
01.02	Dokumenti za rad (1950/1951.)
01.02.01	Uputstvo za rad DND-a
01.02.02	Postupak za osnivanje DND-a
01.02.03	Pravila Društva Naša djeca
01.02.04	Pravila Glavnog odbora društava Naša djeca
01.03	Registracija udruge (od 1990.)

Slika 24. Stranica Klasifikacijski plan

Zaključak

Savez društava Naša djeca Hrvatske tipičan je primjer organizacije koja zanemaruje arhivsko gradivo nauštrb drugih poslovnih aktivnosti. Razlozi su opravdani jer je Savez neprofitna udruga koja raspolaže sa ograničenim financijskim i ljudskim resursima. Stoga nije u potpunosti neočekivano što Savez nema ustrojenu pismohranu i što gradivo nije sustavno arhivirano. Bez obzira na nerazvijenost strukture za upravljanje gradivom, Savez je kroz više desetljeća stvorio i sakupio bogato arhivsko gradivo nastalo tijekom provođenja brojnih originalnih vrijednih aktivnosti s djecom i za djecu, od kojih su neke uvrštene u nacionalne planove aktivnosti za djecu u Hrvatskoj. Dokaz vrijednosti gradiva je potvrda Hrvatskog državnog arhiva koji je gradivo proglasio o državnog značaja. Pozitivna je stvar što se počela razvijati svijest o potrebi zaštite i očuvanja arhivskoga gradiva. Budući da su resursi Saveza ograničeni, u uspostavljanju arhivske djelatnosti najveći doprinos daju volonteri, a riječ je većinom o studentima budućim arhivistima koji su željni primijeniti stečena znanja. Tako je postupak sređivanja gradiva većim dijelom proveden. Sljedeći korak naprijed je implementacija izrađenog arhivskog informacijskog sustava koji će pravilno opisivati gradivo i učiniti ga vidljivijim. Sustav možda nije previše kompleksan, ali je razvijen na način da trenutačno omogućuje pravilnu izradu zapisa o jedinicama gradiva i stvarateljima te njihovu dostupnost na jednom mjestu, s mogućnošću da u budućnosti bude nadograđen s novim funkcionalnostima i aktivnostima. Na kraju se može reći da sustav sigurno neće riješiti sve postojeće nedostatke, ali svakako predstavlja korak naprijed prema ispunjavanju općeg cilja, a to je razvoj arhivske djelatnosti u skladu s najsuvremenijim trendovima i potrebama zajednice.

Literatura

"ASP.NET - Introduction." *Www.tutorialspoint.com*. Accessed February 14, 2017. https://www.tutorialspoint.com/asp.net/asp.net_introduction.htm.

"ASP.NET Master Pages." Accessed April 27, 2017. https://msdn.microsoft.com/en-us/library/wtxbf3hh.aspx.

"ASP.NET Overview." Accessed March 1, 2017. https://msdn.microsoft.com/en-us/library/4w3ex9c2.aspx.

"Best Integrated Development Environment (IDE) Software in 2017." *G2 Crowd*. Accessed April 25, 2017. https://www.g2crowd.com/categories/integrated-development-environment-ide.

"Centar za online baze podataka - Online priručnik." Accessed September 21, 2016. http://onlinebaze.irb.hr/prirucnik.

"Chapter 1. C# and .NET Programming." Accessed February 7, 2017. https://msdn.microsoft.com/en-us/library/orm-9780596521066-01-01.aspx.

"Chapter 1: Introducing C#." Accessed February 7, 2017. https://msdn.microsoft.com/en-us/library/hh145616(VS.88).aspx.

Coronel, Carlos, Steven Morris, and Peter Rob. *Database Systems: Design, Implementation and Management*. 9 edition. Australia; United States: Cengage Learning, 2009.

Date, C. J. *An Introduction to Database Systems*. 8 edition. Boston: Pearson, 2003.

Elmasri, Ramez, and Shamkant B. Navathe. *Fundamentals of Database Systems*. 7 edition. Hoboken, NJ: Pearson, 2015.

"GETDATE (Transact-SQL)." Accessed April 16, 2017. https://docs.microsoft.com/en-us/sql/t-sql/functions/getdate-transact-sql.

"Getting Started with the .NET Framework." Accessed February 7, 2017. https://msdn.microsoft.com/library/hh425099(v=vs.110).aspx.

"HyperLinkColumn.DataNavigateUrlField Property (System.Web.UI.WebControls)." Accessed April 28, 2017. https://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.hyperlinkcolumn.datanavigateurlfield(v=vs.110).aspx.

International Council of Archives, ed. *ISAD(G): General International Standard Archival Description ; Adopted by the Committee on Descriptive Standards, Stockholm, Sweden, 19-22 September 1999*. 2. ed. Ottawa: International Council of Archives, 2000.

"Introduction to the C# Language and the .NET Framework." Accessed February 27, 2017. https://msdn.microsoft.com/en-us/library/z1zx9t92.aspx.

"ISAAR (CPF): International Standard Archival Authority Record for Corporate Bodies, Persons and Families, 2nd Edition | International Council on Archives." Accessed April 4, 2017. http://www.ica.org/en/isaar-cpf-international-standard-archival-authority-record-corporate-bodies-persons-and-families-2nd.

"Microsoft® SQL Server® 2016 SP1 Express." *Microsoft Download Center*. Accessed April 11, 2017. https://www.microsoft.com/en-us/download/details.aspx?id=54284.

"Opći popis gradiva s rokovima čuvanja." Hrvatsko arhivsko vijeće, 2012. http://arhinet.arhiv.hr/_Download/PDF/Opci_popis_gradiva_s_rokovima_cuvanja.pdf.

"Populating a DataSet from a DataAdapter." Accessed April 29, 2017. https://msdn.microsoft.com/en-us/library/bh8kx08z(v=vs.110).aspx.

"Pravilnik o vrednovanju te postupku odabiranja i izlučivanja arhivskoga gradiva." Accessed April 14, 2017. http://narodne-novine.nn.hr/clanci/sluzbeni/2002_07_90_1476.html.

"Pravilnik o zaštiti i čuvanju arhivskog i registraturnog gradiva izvan arhiva." Accessed April 10, 2017. http://narodne-novine.nn.hr/clanci/sluzbeni/2004_05_63_1383.html.

"SQL Server Management Studio (SSMS)." Accessed April 11, 2017. https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms.
"TableAdapter Overview." Accessed April 29, 2017. https://msdn.microsoft.com/en-us/library/bz9tthwx(v=vs.80).aspx.
tutorialspoint.com. "C# Overview." *Www.tutorialspoint.com*. Accessed February 27, 2017. https://www.tutorialspoint.com/csharp/csharp_overview.htm.
"Uredba o uredskom poslovanju." Accessed April 12, 2017. http://narodne-novine.nn.hr/clanci/sluzbeni/2009_01_7_171.html.
"Zakon o arhivskom gradivu i arhivima - Zakon.hr." Accessed March 29, 2017. https://www.zakon.hr/z/373/Zakon-o-arhivskom-gradivu-i-arhivima.
"Zaštita i očuvanje arhivskog gradiva Saveza društava Naša djeca Hrvatske," n.d.

Izrada arhivskog informacijskog sustava Saveza društava Naša djeca Hrvatske

Sažetak: Savezu društava Naša djeca Hrvatske kao stvaratelju i imatelju arhivskoga gradiva cilj je adekvatno i propisno čuvati svoje od državnog značaja gradivo te osigurati uvjete da postane javno dostupno. Termin baza podataka danas se koristi kao univerzalni naziv za cjelokupni sustav baze podataka čija svrha je pohrana informacija i omogućiti korisnicima dohvaćanje i ažuriranje tih informacija. Baze podataka postale su osnovni alat za pretraživanje koji omogućuje interaktivan i nelinearan pristup informacijama, bez vremenskih i prostornih ograničenja. Izrađeni informacijski sustav se sastoji od baze podataka i korisničkog sučelja koje korisniku daje detaljan prikaz klasifikacijskog plana, popisa arhivskoga gradiva i popisa stvaratelja. Atributi sustava koji opisuju sadržaj arhivskoga gradiva definirani su na temelju aktualnih međunarodnih arhivističkih normi i nacionalnih propisa.

Ključne riječi: ASP.NET web obrazac, .NET Framework, relacijski model podataka, Savez DND, SQL

Designing the archival information system of The Union of Societies Our children Croatia

Summary: The Union of Societies Our children Croatia as a creator and holder of archive material aims to adequately and properly preserve material of national significance and to ensure the conditions to become publicly available. The term database nowadays is used as a universal term for database system as whole with it's main purpose of information storage and giving user possibility of retrieving and altering these information. Databases have become essential tool for information search with their immense possibilites regarding interactive and nonlinear information access without time or space barriers. The developed information system consists of a database and a user interface which gives the user a detailed display of the classification plan, list of archival material and list of creators. System attributes describing the archive material are defined on the basis of current international archival norms and national regulations.

Keywords: ASP.NET Web Form, .NET Framework, relational data model, Savez DND, SQL

Životopis

Domagoj Vočanec rođen je 11. prosinca 1989. godine u Zagrebu. Nakon završetka matematičko-informatičkog smjera u XV. gimnaziji, 2011. godine upisuje preddiplomski studij informacijskih i komunikacijskih znanosti na Filozofskom fakultetu Sveučilišta u Zagrebu, a 2014. upisuje diplomski studij arhivistike i bibliotekarstva. Za vrijeme studija radi kao demonstrator u Zatvorenom spremištu knjižnice Filozofskog fakulteta i volontira na sređivanju i opisu građe u knjižnici Umjetničkog paviljona te arhivu Saveza Društava ″Naša djeca″. Dobitnik je Nagrade Filozofskog fakulteta Sveučilišta u Zagrebu za izvrsnost na preddiplomskom studiju i Nagrade Zaklade Dr. Ljerka Markić Čučuković za najboljeg studenta bibliotekarstva u ak. god. 2014/15. U slobodno vrijeme bavi se restauracijom starih bicikala.