Sistem

Sistem predstavlja skup elemenata i njihovih međusobnih veza. Ipak, skup elemenata sistema ne može biti neograničen skup, sistem se mora posmatrati u odnosu na njegovo *okruženje*; mora se definisati *granica sistema*. Zato se može reći je sistem u odnosu na okruženje jedan ograničen skup elemenata i njihovih međusobnih veza.

Granica sistema radvaja sistem od spoljnih sistema. Sistem je u neprekidnoj interakciji sa svojom okolinom. Interakcija sistema sa okruženjem se sastoji iz skupa ulaznih i izlaznih dejstava. Dejstvo okoline na sistem naziva se *ulaz*, a dejstvo sistema na okolinu *izlaz* sistema.

Analiza sistema

Analiza predstavlja postupak izučavanja neke pojave u cilju njenog boljeg razumevanja, odnosno nekog problema u cilju njegovog rešavanja. Postupak analize se, generalno, karakteriše time što se posmatrana pojava, odnosno problem *razlaže* na sastavne delove koji se pojedinačno razmatraju.

U postupku analize sistema se sistem najpre posmatra kao „*crna kutija*“. Crna kutija je sistem, ili deo sistema, sa poznatim ulazom, izlazom ali nepoznatom internom strukturom .Posmatrajući sistem kao jednu celinu, utvrđuje se koji ga to spoljni sistemi okružuju, odnosno sa kojim drugim sistemima je u interakciji. Zatim se definišu skupovi ulaznih (dejstvo okruženja na sistem) i izlaznih dejstava (dejstvo sistema na okruženje). U daljim koracima se, *postupno razlažući sistem*, uočavaju sistemski elementi i njihove međusobne veze. Postupak analize sistema se završava na onom nivou detalja na kom su svi dobijeni sistemski elementi i njihove veze sa strane posmatrača dovoljno jasni da se dalje ne moraju razlagati.

Modelovanje sistema

Modelovanje sistema predstavlja postupak kreiranja *modela sistema*. Postupak kreiranja modela sistema se definiše i opisuje *metodom modelovanja*.

Model sistema

Model predstavlja subjektivnu, apstraktnu (uprošćenu) sliku sistema, opisujući elemente tog sistema i njegove veze. Model sistema je uvek subjektivna slika sistema sagledana iz ugla gledanja posmatrača. sa jednog apsekta. Najprostiji primer modela je geografska karta sveta

***Modeli služe** za bolje razumevanje, analizu, izgradnju novog sistema ili poboljšanje postojećeg*. U toku analize sistema se uz pomoć modela zanemarivanjem nebitnih ističu samo one karakteristike sistema koje su od interesa za posmatrača i time olakšava razumevanje. pri izgradnji jednog sistema moguće je pomoću modela budućeg sistema ispitati njegove karakteristike, pa tako na vreme uočiti eventualne nedostatke.

Metoda modelovanja

Metoda modelovanja definiše jedan mogući način za modelovanje sistema. Da bi kreirali model, potrebno je da raspolažemo odgovarajućim alatom, odnosno potrebno je da imamo adekvatan skup koncepata kojim se možemo poslužiti da bi svoje znanje o sistemu pretočili u model. *Jezik modelovanja* definiše skup koncepata potrebnih za izgradnju modela, odnosno predstavlja alat za opis sistema modelom[1]. Jezikom modelovanja se definiše *sintaksa* i *semantika* koncepata koji se koriste i dodatno, *notacija*. Sintaksa definiše koncepte i pravila jezika i opisana je gramatikom, dok semantika definiše značenje koncepata. Notacija definiše za svaki koncept njegovu grafičku predstavu, odnosno simbol. Modela. Potrebno je uz jezik kao alat za opis modela dati i uputstvo za njegovu upotrebu. Potrebno uputstvo za upotrebu se definiše *postupkom modelovanja*, tako što se navodi *redosled koraka* za izgradnju modela, kao i *rezultat* koji se dobija primenom svakog od koraka.

Prema tome, jedna potpuna *metoda modelovanja* sastoji se iz: jezika modelovanja i postupka modelovanja, kao uputstva za korišćenje jezika u cilju uspešnog kreiranja modela. Jezik modelovanja se vrlo formalno može definisati, za razliku od postupka koji je obično neformalan. Modelovanje sistema se može zamisliti kao proces koji od posmatranog sistema (ulaz) generiše model sistema (izlaz), na način specificiran primenjenom metodom modelovanja (transformacija).

Strukturna sistemska analiza

(SSA) je jedna poptuna, samosvojna metoda za analizu i funkcionalnu specifikaciju informacionog sistema, odnosno softvera. Ona se na različite načine može povezati sa metodama drugih faza u neku specifičnu metodologiju celokupnog razvoja IS

Osnovni koncepti Strukturne sistemske analize

SSA posmatra informacioni sistem kao *funkciju (proces obrade)* koja, na bazi ulaznih podataka i stanja sistema predstavljenog preko *skladišta podataka*, generiše izlazne podatke. Ulazni podaci se dovode u proces obrade, a izlazni iz njega odvode preko tokova podataka. Isto tako, proces obrade koristi i menja podatke o stanju sistema razmenom tokova podataka sa skladištem. *Tok podataka* se tretira kao vod kroz koji stalno teku podaci ili kao pokretna traka koja stalno nosi podatke na najrazličitijim nosiocima - papirni dokumenti, niz poruka koje čovek unosi preko tastature terminala, „paket“ informacija dobijen preko neke telekomunikacione linije ili slično. Entiteti iz okruženja sa kojima IS komunicira nazivaju se *interfejsi*. Ako ceo IS posmatramo kao jedan proces, onda interfejsi predstavljaju izvore svih ulaznih i ponore svih izlaznih tokova podataka.

Dakle, funkcije, odnosno procesi obrade podataka, skladišta podataka, tokovi podataka i interfejsi predstavljaju osnovne koncepte metode SSA. Za njihov prikaz koriste se *Dijagrami tokova podataka* (DTP).
Procesi se predstavljaju elipsom, interfejsi pravougaonikom, tokovi podataka usmerenim linijama, a skladište podataka sa dve paralelne linije.

Dijagrami tokova podataka

Dijagram tokova podataka (DTP) predstavlja osnovni alat za opis funkcija sistema. Funkcije sistema se u metodi SSA, odnosno pri izgradnji DTP posmatraju kao *procesi obrade podataka*, tj. posmatraju se sa tačke gledišta *podataka*. Uočavajući podatke koji ulaze u sistem, obrađuju se i iz sistema izlaze, drugim rečima, koji teku sistemom, uočavamo i koji su procesi potrebni za obradu (transformaciju) tih podataka. Praćenjem tokova podataka je moguće ustanoviti kompletnu sliku o tome šta se to dešava unutar sistema, odnosno kako sistem funkcioniše. DeMarco opisuje taj postupak kao „intervjuisanje podataka“.

Proces

Proces predstavlja deo sistema (ili u posebnom slučaju ceo sistem, koji ima ulogu da transformiše ulazne podatke u izlazne. Grafički se sistem predstavlja elipsom. Proces predstavlja aktivnost, radnju, i važno je imenovati ga na adekvatan način. Proces se obično imenuje parom „predikat – objekat (predmet)“. U imenima najopštijih procesa se objekat radnje može izostaviti. Primeri: Nabavka, Učlanjivanje, Obrada porudžbine, itd.
predmet obrade procesa su uvek paketi podataka u nekoj formi, papirna ili elektronska dokumenta, pojedinačni podaci unešeni preko tastature računara i dr. Jako je važno imati u vidu da se procesi u organizaciji koju analiziramo u jednom trenutku vremena izvršavaju *paralelno*, a ne sekvencijalno.
Svaki proces poseduje pored imena i svoju brojnu oznaku. Brojna oznaka procesa služi samo za referenciranje procesa, a nikako ne predstavlja redosled izvršavanja procesa, zbog paralelizma u izvršavanju.

Tok podataka

Tok podataka se tretira kao vod kroz koji stalno teku podaci ili kao pokretna traka koja stalno prenosi pakete podataka iz jednog dela sistema u drugi, i na taj način ostvaruje vezu između komponenti sistema. svaki tok podataka mora imati svoje izvorište i ponorište. Tok predstavlja podatak u stanju kretanja. Skladište predstavlja podatke u stanju mirovanja. Grafički simbol za prikaz toka podataka je usmereni luk. Tok podataka se imenuje prema značenju podataka koji se tim tokom prenose. imena treba da budu prirodna, Primeri: Faktura, Narudžbenica. Tok podataka takođe govori o usmerenju kretanja podataka. Strelica označava da li podaci u jedan proces, skladište ili interfejs poniru ili iz njega izviru.

Skladište podataka

Skladište podataka predstavlja podatke u stanju mirovanja. Ime skladišta bi trebalo da predstavlja množinu imenice toka podataka koji u njega ulazi ili izlazi.Skladište procesima omogućava međusobnu vremensku nezavisnost, odnosno da se u slučaju različitih paketa podaka procesi mogu *paralelno* ili u slučaju istih paketa, sa *zakašnjenjem* izvršavati.
skladišta podataka na DTP služe da povežu dva procesa. Kad god izlazni tok jednog procesa predstavlja ulazni tok drugog procesa, potrebno je uvesti skladište podataka koje će premostiti vezu ta dva procesa.
Slučaj kada tok podataka iz jednog procesa ulazi u skladište se može tumačiti kao postupak upisivanja novih podataka, ažuriranja ili brisanja postojećih. Slučaj kada tok podataka iz skladišta izlazi i ulazi u proces opisuje se kao mogućnost procesa da čita podatke iz tog skladišta,

Interfejs

Interfejs predstavlja spoljni objekat sa kojim sistem komunicira. Spoljni objekat može npr. biti osoba ili grupa osoba, korisnika sistema. može biti odeljenje unutar organizacije ili van nje – ili čitava eksterna organizacija. Informacioni sistem iz okruženja ili njegov deo takođe može predstavljati interfejs, sa kojim naš sistem komunicira. Interfejs se na dijagramu vizuelno predstavlja kao pravougaonik.
Interfejse je lako identifikovati. Ako sistem posmatramo kao „crnu kutiju“, odnosno kao jedan proces koji na ulazu prima tokove podataka, vrši proces obrade i generiše izlazne tokove, lako možemo uočiti od kog spoljnog objekta prima podatke, a kome obrađene podatke dalje isporučuje.

Pravila i preporuke za kreiranje DTP

(1) Svaki *proces* mora da ima barem jedan ulazni i barem jedan izlazni tok podataka.
(2) Preporučuje se da se jedan *proces* povezuje sa drugim procesom samo posredno preko skladišta podataka. Direktnu vezu dva procesa samo preko toka podataka trebalo bi izbegavati, jer se na taj način veoma ograničava njihovo nezavisno izvršavanje. Trenutak generisanja izlaznog toka prvog procesa automatski uslovljava početak izvršavanja drugog procesa. Pored toga, nepostojanjem skladišta podataka između dva procesa sistem se lišava mogućnosti da zapamti sopstveno međustanje (nakon završetka jedne i početka druge obrade), a to je jedna od osnovnih osobina skladišta podataka.
(3) Samo *tokovi podataka* koji idu ka, odnosno od skladišta podataka ne moraju biti imenovani. Ako tok između procesa i skladišta nije imenovan, podrazumeva se da tok nosi celokupan sadržaj i strukturu podataka tog skladišta. Ukoliko to nije slučaj, treba ga imenovati.
(4) *Tokovi podataka* koji poniru u jedno skladište ili iz njega izviru, mogu da prenose samo one pakete podataka koji se u skladištu čuvaju.
   i. *Tok podataka* mora da ima izvor i ponor. Bilo koja druga komponenta DTP može da bude izvor ili ponor. Međutim, za jedan tok, bilo izvor, bilo ponor (bilo oba) mora da bude proces. Dva interfejsa, dva skladišta, ili interfejs i skladište ne mogu direktno biti povezani tokom podataka
   ii. *Skladišta* ne mogu međusobno direktno biti povezana tokom podataka, jer skladišta, kao pasivne komponente, ne vrše nikakvu obradu.
   iii. *Interfejsi* ne mogu međusobno direktno biti povezani tokom podataka, jer bi se na taj način opisivala komunikacija dva objekta van sistema koja nije od interesa za sistem koji posmatramo.
   iv. *Interfejs* i *skladište* ne mogu direktno biti povezani tokom podataka jer bi to značilo da spoljni objekat direktno, bez kontrole internog procesa poseduje pristup skladištu podataka.
(5) Svako *skladište* mora da ima barem jedan ulazni i barem jedan izlazni tok podataka..
(6) *Interfejsi* moraju biti povezani sa sistemom, odnosno procesima sistema barem sa jednim ulaznim ili izlaznim tokom podataka.
(7) Preporuka vezana za preglednost dijagrama kaže, da se u cilju izbegavanja nepotrebnog presecanja linija bilo *skladište* bilo *interfejs* na jednoj slici može višestruko ponoviti. U tom slučaju potrebno je samo pored imena koncepta dodati znak „*“.

Dekompozicija Dijagrama tokova podataka

Uloga DTP se sastoji u tome da prikaže sve sistemske procese koji na bazi ulaznih tokova podataka vrše obradu i generišu izlazne tokove. Izvori i ponori svih tokova podataka jednog sistema su na DTP predstavljeni ili spoljnim objektima ili skladištima podataka.

Veoma je važno da dijagrami tokova podataka budu pregledni, jasni i lako razumljivi. Način je da sadrži relativno mali broj procesa uključujući sve potrebne interfejse, tokove i skladišta podataka. Ali, ako imamo složen sistem koji se sastoji od nekoliko desetina, stotina ili hiljada procesa
Možemo najpre da ga podelimo na skup podsistema. Zatim svaki od podsistema dalje možemo podeliti na podpodsisteme. Ovaj princip postepenog razlaganja sistema možemo pratiti sve dok ne stignemo do nivoa prostog, primitivnog procesa koji se dalje ne može deliti.
Korišćenjem metode apstrakcije. tehnika izgradnje dijagrama tokova podataka se svodi na to da se na višim nivoima definišu globalniji procesi, a da se zatim svaki takav globalni proces, na sledećem nižem nivou, predstavi novim dijagramom toka podataka apstrakcije,
Dijagram tokova podataka na vrhu hijerarhije naziva se *dijagram konteksta*, a procesi na dijagramima najnižeg nivoa hijerarhije, koji se dalje ne mogu dekomponovati, nazivaju se *primitivni* procesi.

Dijagram konteksta

Najopštiji dijagram tokova podataka je dijagram konteksta. Dijagram konteksta posmatra sistem kao „crnu kutiju“, kao jedan proces, koji na bazi ulaza generiše izlazne podatke. Uloga dijagrama konteksta je da definiše kontekst analize, odnosno da odredi jasnu granicu sistema, da odgovori, šta su to sastavni delovi sistema, a šta su spoljni objekti sa kojima on interaguje. Zatim je potrebno identifikovati komunikaciju između posmatranog sistema i njegovog okruženja, odnosno uočenih spoljnih objekata. Dijagram konteksta mora sadržati samo jedan proces koji predstavlja sistem, skup interfejsa i tokove podataka, koji od interfejsa ulaze u sistem i obratno, koji kao rezultat procesa obrade izlaze iz sistema ka interfejsima.
Ime procesa obično predstavlja ime celokupnog informacionog sistema. *Interfejsi* su povezani sa sistemom preko ulaznih i izlaznih tokova podataka. Interfejsi ni u kom slučaju ne smeju međusobno direktno da komuniciraju. Ako ipak postoji povezanost dva interfejsa onda će oni međusobno posredno komunicirati preko sistemskog procesa. Kada su identifikovani svi interfejsi, potrebno je uočiti sve relevantne *tokove podataka*.

Na osnovu analize mogućih događaja iz okruženja na koje sistem reaguje i događaja koje sistem kao reakciju generiše, mogu se identifikovati neophodni tokovi podataka.. Dogadjaji koji deluju na sistem obično su spoljni događaji i najviše govore o mogućim tokovima podataka. Na primer, za jedan IS Opštine spoljni događaj može biti „Stranka podnosi zahtev za izdavanje potvrde od državljanstvu“, na koji IS sistem treba da reaguje procesom obrade zahteva. Tako bi jedan ulazni tok podataka onda bio: „Zahtev za izdavanje potvrde o državljanstvu“, a izlazni tok podataka: „Potvrda o državljanstvu“.
Izgradnja dijagrama konteksta je najvažniji zadatak u postupku kreiranja DTP. Vrlo je važno uočiti sve moguće interfejse na samom početku, kao i sve tokove podataka koji ulaze u sistem i iz njega nakon obrade izlaze.

Primitivni procesi

Primitivni procesi su oni procesi koji se dalje ne mogu dekomponovati. Predstavljaju se Dijagramima tokova podataka na dnu hijerarhije, kao rezultat konačne dekompozicije funkcija sistema. Nazivaju još i atomskim procesima. Za razliku od složenih procesa, primitivni procesi se izvršavaju *sekvencijalno*, redosled aktivnosti u okviru njih je definisan.

Identifikovanjem svih primitivnih procesa jednog sistema završava se proces dekompozicije, pa samim tim i proces analize funkcija sistema,
Sledeći korak u metodi Strukturne sistemske analize jeste dati *specifikaciju logike primitivnih procesa*. Specifikacija logike primitivnih procesa, tzv. Mini-speficikacija ima zadatak da za svaki primitivni proces detaljnije opiše sekvencijalni postupak njegovog izvršavanja,kako na osnovu ulaznih podataka transformacijom dobiti izlazne podatke.Ssistem, shvaćen kao funkcija koja na bazi ulaznih podataka generiše izlazne, može da se posmatra kao „bela kutija“, jer je poznata ne samo njegova interna struktura, već i način funkcionisanja.

Najzastupljenije tehnike za opis logike primitivnih procesa su dijagrami toka programa, strukturni jezici, pseudokod, tabele odlučivanja, stabla odlučivanja i dr.

Pravila i kriterijumi dekompozicije

na vrhu hijerarhije se nalazi Dijagram konteksta. Njegovom dekompozicijom se dobija tzv. *Prvi nivo dekompozicije*. Daljom dekompozicijom dijagrama prvog nivoa, dobijaju se dijagrami drugog nivoa čiji ukupan broj odgovara ukupnom broju procesa sa prvog nivoa. Postupak se sukcesivno nastavlja na sledećem nivou, sve dok se ne dođe do nivoa primitivnih procesa.

Sledeća pravila regulišu postupak dekompozicije:

- *Pravilo balansa tokova*- Najznačajnije pravilo koje se mora poštovati pri dekompoziciji procesa. Ulazni i izlazni tokovi na celokupnom DTP-u koji je dobijen dekompozicijom nekog procesa P moraju odgovarati ulaznim i izlaznim tokovima toga procesa P na dijagramu višeg nivoa. Drugim rečima, svi tokovi koji ulaze i izlaze iz procesa moraju se pojaviti i na DTP sledećeg nivoa, koji taj proces dekomponuje. Međutim, moguće je odstupiti od ovog pravila u slučaju *dekompozicije samih tokova podataka*. Ukoliko je na višem nivou hijerarhije uveden tok podataka koji se suštinski sastoji iz više pojedinačnih tokova, na sledećem nivou se on može dekomponovati i time prividno narušiti balans tokova. Dekompozicija tokova podataka se eksplicitno dokumentuje u Rečniku podataka, čime balans tokova u hijerarhiji dijagrama ostaje nenarušen.
- *Pravilo numerisanja procesa i dijagrama*: Procesi se u cilju lakše čitljivosti i preglednosti numerišu na sledeći način. Procesi na Prvom nivou dekompozicije se numerišu redom sa 1, 2, ...N. Svaki dijagram nižeg nivoa nosi oznaku procesa kojeg dekomponuje, a u njemu sadržani procesi na taj broj dodaju svoj jedinstveni broj. Dakle, dekompozicija procesa 1 predstavljena je dijagramom sa oznakom 1, i sastoji se od M podprocesa sa oznakama 1.1, 1.2, 1.3...1.M. Pravilo se nastavlja za svaki sledeći nivo hijerarhije daljim dodavanjem jedinstvenog broja procesa sa tog nivoa, 1.1.1, 1.1.2, ...
- *Skladišta podataka*. Skladišta podataka, pretstavljaju stanja sistema, odnosno fundamentalne, *unutrašnje* karakteristike kako celog IS, tako i svakog pojedinačnog procesa. Zbog toga se ona mogu pojaviti i na nižim nivoima dekompozicije, iako se nisu pojavljivala na prethodnim. Pravilo za uvođenje novih skladišta glasi: *Skladišta se uvode po prvi put na onom DTP-u na kome predstavljaju vezu između dva ili više procesa. Nakon što su se pojavila na jednom nivou uz jedan proces, skladišta se moraju pojavljivati i na svim dijagramima nižeg nivoa koji dekomponuju taj proces*.
- *Dodatak pravilima dekompozicije*. Nepisano pravilo pri dekompoziciji dijagrama glasi da se svaki proces može dekomponovati najviše u sedam podprocesa, Veći broj procesa od ovoga mogao bi značiti da je „preskočen“ jedan nivo apstrakcije.

Proces dekompozicije u jednom trenutku mora prestati, kada dođemo do primitivnih procesa.
Nešto precizniji kriterijum glasi da dekompoziju procesa treba prekinuti u trenutku kada svaki proces poseduje samo jedan ulazni i jedan izlazni tok.

Paralelni procesi se nazivaju *suštinski procesi* jedne organizacije. Uz to, paralelizam suštinskih procesa omogućen je postojanjem *suštinskih skladišta*. Primitivni procesi se izvršavaju sekvencijalno i da se opis njihove logike, umesto dekompozicijom, može opisati tehnikama za opis sekvencijalne logike (Pseudokod, Strukturni jezici i dr.). Upravo iz razlike u karakteru procesa koji se odvijaju u organizaciji, odnosno iz razlike između paralelnih i sekvencijalnih procesa Lazarević dopunjuje originalne autore i izvodi osnovni kriterijum dekompozicije u SSA:

Dekompoziciju treba vršiti dok se neka funkcija prirodno može dekomponovati na suštinske paralelne funkcije koje međusobno komuniciraju isključivo preko suštinskih skladišta. Drugim rečima, dekompoziciju treba okončati kada se dođe do procesa koji su prirodno sekvencijalni

Rečnik podataka

Dijagrami tokova podataka prikazuju i pakete podataka, koji tokovima podataka cirkulišu u sistemu, odnosno koji bivaju sačuvani u skladištima podataka. DTP u pogledu podataka daju samo uvid u to, koji podaci se u sistemu pojavljaju, bez detaljnijeg bavljenja njihovom strukturom, tj. *strukturom tokova podataka i skladišta podataka*. Baš kao i procesi, i tokovi podataka i skladišta mogu imati složenu strukturu, pa je potrebno dekomponovati ih na prostije elemente. Dekompozicija tokova podataka i skladišta se opisuje u *Rečniku podataka*. Rečnik podataka (RP) predstavlja alat za strukturirani opis podataka u sistemu, odnosno opis njihovog sadržaja i strukture. Opis je strukturiran zato što RP definiše poseban skup koncepata i pravila za opis podataka, odnosnu sintaksu za opis strukture podataka[2].

Primitivne komponente strukture

Osnovne, primitivne komponente strukture podataka predstavljaju elementarni podaci, tzv. *polja*. Polja su osnovna komponenta strukture podataka jer predstavljaju podatke koji se dalje ne mogu dekomponovati. Primeri polja mogu biti: *Ime, Prezime, BrLičneKarte, Ocena, Cena, BrIndeksa* itd. Svako polje ima svoju vrednost. Skup iz kojeg jedno polje može da uzima vrednosti naziva se *domen*. Na primer, polje *Ocena* uzima vrednosti iz skupa pozitivnih realnih brojeva u intervalu od 5 do 10. *Ograničenje nad domenom* sužava skup mogućih vrednosti koje polje može da uzme iz domena. Po svojoj prirodi domeni mogu biti *predefinisani* ili *semantički* domeni. Predefinisani domeni su standardni, sveprisutni domeni kao što su skup celih brojeva, skup realnih brojeva, skup karaktera ili skup logičkih vrednosti (tačno/netačno). Semantički domeni daju novo ime predefenisanim domenima i obično kroz ograničenje definišu samo podskup predefinisanog skupa elemenata. Na primer, za polje *Ocena* može se definisati semantički domen *FakultetskeOcene* koji sužava skup prirodnih brojeva na interval od 5 do 10.

Složene strukture podataka

Tokovi i skladišta podataka definisani na DTP obično imaju složenu, kompozitnu strukturu. Tipični paketi podataka sa složenom strukturom koji „teku“ kroz jedan IS su dokumenta kao što su: Katalog (filmova, proizvoda), Narudžbenica, Račun, Ispitna prijava, Uverenje, Cenovnik i sl.

Složene strukture podataka se sastoje iz više komponenata. Komponenta može biti elementarna, tj. polje i/ili neka druga definisana složena struktura.

Moguće su sledeće konstrukcije kojima se od komponenata (primitivnih ili složenih) gradi struktura. Za svaku kontrukciju se navodi i način njenog predstavljanja u Rečniku podataka:

0. *Agregacija komponenti*. Agregacija predstavlja složenu strukturu u vidu liste N komponenti, bilo elementarnih, bilo složenih. Za njeno predstavljanje se koristi sledeća notacija: <K1, K2,...Kn.>, gde su K1, K2, Kn sastavne komponente strukture. Na primer, tok podataka *IspitnaPrijava* se zapisuje na sledeći način:

   IspitnaPrijava: <BrIndeksa, ImeStudenta, NazivPredmeta, DatumPolaganja, Ocena, ImeNastavnika>

1. *Specijalizacija (unija) komponenti*. Specijalizacija predstavlja strukturu u kojoj se bira jedna (*ekskluzivna* specijalizacija) ili više (*neekskluzivna* specijalizacija) od navedenih komponenti (Opcija). Komponente ekskluzivne specijalizacije se navode unutar uglastih zagrada: [K1, K2,...Kn]. Primer specijalizacije sa ekskluzivnim izborom je skladište podataka *PoslovniPartneri*:

   PoslovniPartneri: <SifraPP, NazivPP, AdresaPP, [ImeKontaktOsobe, Pol]>

   Struktura PoslovniPartneri predstavlja agregaciju polja SifraPP, NazivPP, AdresaPP, i ekskluzivne specijalizacije polja ImeKontaktOsobe (u slučaju kada je poslovni partner pravno lice) i Pol (u slučaju kada je poslovni partner fizičko lice).

   Komponente neekskluzivne specijalizacije se navode untar kosih zagrada: /K1, K2, ...Kn/. Primer specijalizacije u kojoj je moguće izabrati više od jedne komponente je složeni tok podataka Uverenje:

   Uverenje: </ UverenjeOUpisu, UverenjeOPolIspit />

   Fakultet može na zahtev studenta da izda bilo uverenje o upisu, bilo uverenje o položenim ispitima, bilo oba uverenja.

2. *Skup (Iteracija)*. Skup je struktura u kojoj se jedna komponenta ponavlja više puta, tj. iterira (kroz različite vrednosti komponente). Komponenta sa ponavljanjem se unutar neke strukture predstavlja vitičastim zagradama: {K1}. Primer za strukturu skupa je upravo dokument Narudžbenica, koji u sebi pored elementarnih polja sadrži i složenu strukturu u vidu agregacije polja koja se višestruko ponavljaju:

   Narudžbenica: <BrojNarudžbenice, ŠifraKupca, NazivKupca, DatumNaručivanja, {<RedniBroj, ŠifraArtikla, NazivArtikla, NarKoličina>}>

   Ovakva struktura je vrlo česta i najlakše se prepoznaje kod dokumenata koji u sebi pored elementarnih polja sadrže i neki oblik nabrajanja u vidu tabele ili liste.