Model inteligentnog tutorskog sustava zasnovan na obradi kontroliranog jezika nad ontologijom

Žitko, Branko

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Branko Žitko

MODEL INTELIGENTNOG TUTORSKOG SUSTAVA ZASNOVAN NA OBRADI KONTROLIRANOG JEZIKA NAD ONTOLOGIJOM

DOKTORSKA DISERTACIJA

Zagreb, 2010.

Doktorska disertacija je izrađena u Zavodu za elektroniku, mikroelektroniku, računalne i inteligentne sustave Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

Mentor: prof. dr. sc. Slavomir Stankov (Prirodoslovno-matematički fakultet Sveučilišta u Splitu)

Doktorska disertacija ima 222 stranica

Disertacija br.:

Povjerenstvo za ocjenu doktorske disertacije:

1. Akademik dr.sc. Leo Budin, profesor emeritus (u miru)
   Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva
2. Dr.sc. Slavomir Stankov, redoviti profesor
   Sveučilište u Splitu Prirodoslovno-matematički fakultet
3. Dr.sc. Vlado Glavinić, redoviti profesor
   Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva

Povjerenstvo za obranu doktorske disertacije:

1. Akademik dr.sc. Leo Budin, profesor emeritus (u miru)
   Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva
2. Dr.sc. Slavomir Stankov, redoviti profesor
   Sveučilište u Splitu Prirodoslovno-matematički fakultet
3. Dr.sc. Vlado Glavinić, redoviti profesor
   Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva
4. Dr.sc. Marko Tadić. izvanredni profesor
   Sveučilište u Zagrebu Filozofski fakultet
5. Dr.sc. Bojana Dalbelo Bašić, redovita profesorica
   Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva

Datum obrane disertacije: 03. ožujka 2010. godine

ZAHVALA

Svome mentoru i prijatelju prof. dr.sc. Slavomiru Stankovu neizmjerno se zahvaljujem na velikoj podršci, razumijevanju, vrijednim savjetima i nesebičnoj pomoći tijekom svih faza nastanka ovog rada.

Zahvaljujem na uloţenom trudu i vremenu članovima povjerenstva za ocjenu rada: akademiku prof. dr.sc. Leu Budinu i prof. dr.sc. Vladi Glaviniću.

Duţnost mi je izraziti iskrenu i duboku zahvalnost prof. dr.sc. Marku Tadiću na stručnoj i praktičnoj pomoći te ustupanju morfološkog leksikona bez kojeg bi realizacija rada bila nemoguća.

Veliko hvala prof. dr.sc. Bojani Dalbelo Bašić na korisnim savjetima i dr.sc. Nives Mikelić Preradović na ustupanju valencijskog leksikona.

Zahvaljujem na pomoći Ţeljku Agiću, dipl.ing. koji je pomogao u ostvarivanju pristupa morfološkom leksikonu.

Iskreno zahvaljujem mojim kolegicama i kolegama s Prirodoslovno-matematičkog fakulteta i Filozofskog fakulteta Sveučilišta u Splitu, ponajprije mr.sc. Ani Grubišić, prof. dr.sc. Marku Rosiću i mr.sc. Suzani Tomaš koji su uvijek davali prijateljsku i stručnu pomoć kada bih se našao u dilemi ili neznanju.

Nadalje zahvaljujem kolegicama Marini Trumbić, Antoniji Šuvar, Ani Gudelj i Marini Kodru čija su istraţivanja i radovi pomogli pri razvoju ideje ovog rada.

Na kraju najljepše hvala mojoj obitelji, posebice supruzi Tini, sinu Karlu, roditeljima i sestri na potpori i strpljenju.

Sadrţaj

1 Uvod

Računalo je konstruirano kako bi obraĎivalo podatke na osnovu programski opisanih algoritama. Rezultati računalne obrade podataka su informacije koje se razlikuju od pukih podataka po tome što imaju značenje. UvoĎenjem značenja podaci postaju informacije, a grupiranjem specijaliziranih informacija se podaci izdiţu do razine znanja. Na sličan način, ako bi algoritme koji obraĎuju znanje izdigli do razine zaključivanja, onda za takav računalni sustav moţemo reći kako implementira jedan od vidova umjetne inteligencije jer oponaša ljudsko biće koje posjeduje znanje te je u stanju provoditi zaključivanje nad tim znanjem.
Inteligentni tutorski sustavi (ITS) su računalni sustavi zasnovani na tehnikama umjetne inteligencije kako bi simulirali ljudske tutore koji znaju što poučavaju, koga poučavaju i kako poučavaju. Za ITS-ove se kaţe kako spadaju u kategoriju *sustava temeljenih na znanju* jer prikazuju znanje o područnom znanju kojeg poučavaju, znanje o učeniku koga poučavaju i znanje o tutorskim metodama koje koriste za poučavanje učenika [SELF1974]. Ovakav pristup je uvjetovao dekompoziciju ITS-a na modul stručnjaka, modul učenika i modul učitelja. Učinkovitost ITS-a u poučavanju učenika ovisi o učinkovitosti njegovih brojnih izvedbi temeljenih na jedinstvenom modela u kojem se oblikuje znanje i ponašanje stručnjaka, učitelja i učenika. MeĎutim, vaţnost komunikacije s učenikom je uvjetovala dodavanje komponente korisničkog sučelja u tradicionalnu arhitekturu ITS-a. Zbog navedenog se inteligentni tutorski sustavi smatraju i *sustavima za komunikaciju znanjem* [WENG1987]. Cilj komunikacije znanjem je uspostaviti korisničko sučelje ITS-a koje implementira model diskursa kako bi se razriješile nejasnoće učenika. Pri tome tutorski oblik učenja i poučavanja se manje bavi didaktičkim objašnjenima sa strane tutora, već je više orijentiran na interakciju s učenikom. Pokazano je kako učenikovo samostalno konstruiranje znanja na osnovu interakcije s tutorom ima vaţan utjecaj na učenje i poučavanje [CHIS2001].
Ljudski tutori koriste razne tehnike komuniciranja kao što su govor tijela, geste, oklijevanje, naglašavanje i prije svega dijalog. Pošto većina ITS-ova izbjegava korištenje prirodnog jezika, istraţivači su predloţili korištenje dijaloga na prirodnom jeziku kako bi se ITS "približio" učeniku. Ovakva inicijativa je takoĎer utjecala i na istraţivanje pedagoških agenata koji koriste neverbalne načine komuniciranja.
Očekivano je kako će ITS-ovi koji podrţavaju neke od tehnika obrade prirodnog jezika biti učinkovitiji. Osim toga, pokazalo se kako inteligentni tutorski sustavi podrţanim s dijalogom pokazuju veći učinak od učenja temeljenog na čitanju nastavnog sadrţaja [VANL2007]. Nadalje, učenici koji su bili poučavani izradi računalnog programa uz pomoć ITS-a na prirodnom jeziku pokazuju bolje sposobnosti planiranja i dekompozicije programskog rješenja [LANE2005].
Tehnike i autorski alati za obradu prirodnog jezika omogućavaju dobro razumijevanje učenikovih izjava što je bitno za produktivno i realistično reagiranje ITS-a.
Obrada prirodnog jezika je općenito fokusirana na razumijevanje ili generiranje prirodnog jezika i to na nekoliko razina. Razina *sintakse* se odnosi na strukturu riječi i rečenica, dok se na *semantičkoj* razini proučava značenje grupe riječi. Na *pragmatičkoj* razini se razmatra namjena grupa riječi, a razina *dijaloga* je odgovorna za realizaciju razmjene grupa riječi meĎu sudionicima komunikacije na prirodnom jeziku. Tijekom generiranja jezika, tutori generiraju fraze, rečenice ili dijalog. Oni mogu primiti zahtjev za izvršavanjem nekih komunikacijskih činova (pragmatika) ili oblikovati strukture koje ukazuju na značenje učenikovog iskaza (semantika) na osnovu kojeg se generiraju sintaktičke strukture (sintaksa). Sintaktička struktura se na kraju moţe realizirati u obliku teksta ili zvuka. Pet faza obrade prirodnog

jezika [JURM2000] pruţaju prikladnu metaforu koraka obrade temeljenoj na znanju [WOOL2008]:

- fonološka faza provodi izvlačenje riječi iz govora,
- sintaktička faza interpretira oblik učenikovih rečenica,
- semantička faza interpretira rečenice učenika,
- pragmatička faza interpretira namjeru učenika,
- faza diskursa interpretira višerečenične strukture i preokrete u komunikaciji.

MeĎutim, ove faze često ne odgovaraju izravnoj implementaciji obrade prirodnog jezika. Zapravo, mnoge faze simultano ili interaktivno funkcioniraju i posjeduju dvojne aspekte ovisno o tome radi li se o generiranju ili razumijevanju prirodnog jezika. U oba slučaja koriste se različiti interni prikazi podataka, a sustavi zasnovani na obradi prirodnog jezika tipično sjedinjuju preslikavanja prikaza jedne razine u prikaze druge razine.
Izgradnja računalnih sustava koji interakciju s korisnikom provode preko sučelja na prirodnom jeziku nije jednostavna kao što je recimo implementacija sučelja s pedagoškim agentima. Sloţenost obrade prirodnog jezika posljedica je sloţenosti jezika kojeg obraĎuje. Morfološka sloţenost hrvatskog jezika uvjetuje sloţeniju obradu jezika na razini riječi, a time i na ostalim sintaktičkim razinama. Riječi hrvatskog jezika ne uvjetuju samo sintaktičku sloţenost, već i semantičku jer posjeduje veliki broj istopisnica. Na primjer riječ "računala" moţe označavati imenicu srednjeg roda, kao i glagol u prošlom vremenu. Nadalje, hrvatski standardni jezik posjeduje sloţeni tvorbeni sustav za glagole što oteţava odreĎivanje oblika riječi koje se prilaţu glagolu.
U inteligentnim tutorskim sustavima zasnovanim na obradi prirodnog jezika cilj je ostvariti dijalog miješane inicijative otvorenog tipa. Sustavi koji vode dijalog miješane inicijative će razumjeti iskaze učenika kao i generirati prirodan jezik, ali većina ipak implementira dijalog jednostrane inicijative u kojem tutor postavlja pitanja, a učenik odgovara. Pokazano je kako ITS-ovi koji posjeduju samo generiranje prirodnog jezika pospješuju učenje i poučavanje [DIEU2005a]. Arhitektura inteligentnih tutorskih sustava zasnovanih na obradi prirodnog jezika je uglavnom temeljena na proširivanju postojećih komponenata funkcijama za obradu jezika, ako je to bilo moguće. MeĎutim oblikovane su i nove verzije sustava kako bi se podrţala i implementirala obrada prirodnog jezika.

Istraţivanje u okviru doktorske disertacije orijentirano je na novi model inteligentnog tutorskog sustava zasnovanog na obradi kontroliranog jezika nad ontologijom. Nastavak je to dugogodišnjeg istraţivanja, razvoja i primjene modela Tutor-Expert System (TEx-Sys) za oblikovanje inteligentnih tutorskih sustava za po volji odabrana područna znanja [STAN1997] [STAN2008]. Upotreba kontroliranog jezika i ontologije u ovom pristupu inspirirana je nizom provedenih eksperimenata s učenicima različite kronološke dobi koje se provodilo sa različitim verzijama TEx-Sys modela [STAN2004] [GRUB2006]. Tijekom posljednjih petnaest godina razvijene su tri verzije sustava TEx-Sys koje se razlikuju po arhitekturi, implementiranim funkcionalnostima i podrţanim sudionicima. *Prva* verzija se zasebno postavljala na računalo zajedno s bazama znanja, što je oteţavalo administraciju sustava. Arhitektura implementirane prve verzije TEx-Sys-a je monolitna opisana u skladu s formalizmom Unified Modeling Language (UML) dijagrama [ŢITK2001]. U prvoj verziji, osim učenika, sudjeluje učitelj koji je ujedno imao ulogu stručnjaka radi oblikovanja područnog znanja. Razlog tome je što je nastavni sadrţaj bio ekvivalentan područnom znanju. *Druga* verzija zvana Distributed Tutor-Expert System (DTEx-Sys) [ROSI2000] je nastala po paradigmi učenja i poučavanja na daljinu te je implementirana tehnologijom dinamičkih Web stranica. Arhitektura sustava DTEx-Sys je troredna te u sloju podataka koristi baze znanja

koje je učitelj razvio u autorskom okruţenju prve verzije sustava TEx-Sys. Sustavi TEx-Sys i DTEx-Sys su izjednačavali područno znanje s nastavnim sadrţajem, a jedino TEx-Sys je do tada posjedovao autorski alat za oblikovanje nastavnog sadrţaja. Teţnja za normiranjem nastavnog sadrţaja i izdvajanja od područnog znanja je rezultirala *trećom* verzijom sustava zvanom eXtended Tutor-Expert System (xTEx-Sys) [STAN2003] koja je temeljena na Web uslugama [ROSI2004], a nastavni sadrţaj je oblikovan po Shareable Content Object Reference Model (SCORM) normi [ADVA2004]. MeĎutim, objekti nastavnog sadrţaja u xTEx-Sys sustavu nisu meĎusobno uvjetovani, odnosno učenik je mogao u bilo kojem trenutku odabrati bilo koji objekt nastavnog sadrţaja. xTEx-Sys posjeduje autorsko okruţenje za oblikovanje područnog znanja od strane stručnjaka, kao i za oblikovanje nastavnog sadrţaja od strane učitelja.

U svim verzijama sustava TEx-Sys prikaz znanja je temeljen na semantičkoj mreţi s okvirima. Učenje se realizira navigacijom kroz mreţu čvorova semantičke mreţe, a testiranje se provodi dinamičkim kvizom. Jedino u prvoj verziji sustava je implementiran pristup testiranju znanja po konstruktivističkom načelu u kojem učenik nastoji nadopuniti problemsku mreţu područnog znanja do početne originalne mreţe područnog znanja koju je oblikovao učitelj. Ovakav način testiranja znanja pokazao se pogodnim za dijagnosticiranje nedostajućeg i pogrešnog poimanja učenika. Kviz u TEx-Sys modelu dinamički generira pitanja i odgovore temeljem strukture područnog znanja uz prisutnu prilagodbu teţine pitanja za učenika. Iskazi pitanja i odgovora su oblikovani popunjavanjem predloţaka, a dobiveni iskazi su često bili gramatički neispravni. Nepravilna gramatička struktura dinamički generiranih pitanja je oteţavala učeniku razumijevanje onoga što ga se pita. Pošto se generiranje vrši nad elementima područnog znanja, onda je gramatička nepravilnost, prije svega, posljedica nepostojanja dogovora za unošenje naziva elemenata područnog znanja prilikom njegovog oblikovanja. Osim toga, semantička mreţa s okvirima korištena u TEx-Sys modelu nije u stanju podrţati stvaranje sloţenijih izraza koji bi uključili nabrajanje i negaciju, što je bilo potrebno radi prirodnijeg oblika krajnjeg izraza. Istraţivanjem se pokazalo kako opisna logika posjeduje navedene osobine te zadrţava svojstvo mreţnog prikaza znanja.

Jezik za opisivanje ontologije na Web-u Web Ontology Language (OWL) [BECH2004] uključuje formalizam opisne logike koji je strojno obradiv i čovjeku razumljiv. Ovom zamjenom tehnike za prikaz znanja postiglo se strukturalno bogatije izraţavanje [STAN2005] [ŢITK2005], ali generirana pitanja su i dalje ostala gramatički nepravilna. Osim toga, učenici su i dalje mogli jednostavnom metodom eliminacije odabrati točan odgovor. Kako bi se poboljšala kvaliteta odabranih odgovora na generirana pitanja, uključen je i mehanizam zaključivanja [ŢITK2009].

Temeljem svega iznesenog, posve je jasan nastavak i pravac daljnjeg istraţivanja (proveden u okviru ove disertacije) orijentiran na računalnu obradu hrvatskog standardnog jezika. Problematika morfologije riječi hrvatskog jezika je kočila daljnje istraţivanje sve dok se nije otkrila otvorenost pristupa leksičkom posluţitelju Hrvatskog morfološkog leksikona [TADI2003]. Hrvatski morfološki leksikon je omogućio generiranje i prepoznavanje oblika riječi, a u sljedećem koraku istraţivanja je bilo potrebno definirati pravila za grupiranje riječi u sloţenije sintaktičke strukture, kao što su fraze i rečenice. Kako bi se pravilno odredili oblici riječi koje se prilaţu uz glagole upotrijebljen je valencijski leksikon hrvatskih glagola [MIKE2008]. Valencijskim leksikonom glagola se osigurala pravilna gramatička struktura fraza koje posjeduju glagole. Pošto je novi model inteligentnog tutorskog sustava zasnovan na ontološkom prikazu znanja, onda se sintaksa rečenice ograničila na sintaksu izraza u opisnoj logici. Ograničenje je napravljeno i na nazivima elemenata ontologije, što je naposljetku rezultiralo restrikcijom obrade hrvatskog standardnog jezika na obradu kontroliranog jezika.

Kontrolirani jezik je gramatički ograničen, a njegova obrada se vrši nad manjim skupom riječi hrvatskog jezika. Generiranje kontroliranog jezika u novom modelu inteligentnog tutorskog

sustava omogućava učeniku učenje na osnovu prikaza rečenica. Ideja kviza za testiranje znanja učenika je napuštena jer je bilo moguće realizirati dijalog jednosmjerne inicijative u kojem računalni tutor postavlja pitanja, a učenik odgovara i dobiva povratnu informaciju na kontroliranom jeziku. Ipak, odgovori na pitanja su sintaktički ograničeni na fraze ili nizove fraza kontroliranog jezika, a ne na rečenice ili nizove rečenica. Ako bi model ITS-a bio u stanju obraditi rečenice učenikovog unosa, onda bi taj proces podrazumijevao razumijevanje kontroliranog jezika. MeĎutim, ograničenje na fraze je ograničilo i razumijevanje kontroliranog jezika do razine prepoznavanja. Obrada kontroliranog jezika u novom modelu inteligentnog tutorskog sustava se u jednom smjeru realizira generiranjem, a u drugom smjeru prepoznavanjem kontroliranog jezika.
Za razliku od sustava xTEx-Sys učeniku se ipak ţeljelo ograničiti izbor objekata nastavnog sadrţaja. Ideja je realizirana slijedom objekata nastavnog sadrţaja namijenjenih učenju i objekata nastavnog sadrţaja namijenjenih testiranju. Učenik je prvo morao učiti, a zatim je testirao naučeno znanje putem dijaloga. Ako rezultat testa nije bio dovoljan za prelazak na sljedeći objekt nastavnog sadrţaja, onda je učenje i testiranje ponovljeno.
Novi model inteligentnog tutorskog sustava se razlikuje od xTEx-Sys-a i po tome što ne sadrţi autorsko okruţenje za oblikovanje nastavnog sadrţaja. Ovu funkcionalnost obavlja algoritam za planiranje i generiranje sadrţaja i slijeda objekata nastavnog sadrţaja, a na osnovi prethodno oblikovanog područnog znanja. Ipak stručnjak mora koristiti autorski alat kako bi oblikovao područno znanje.
Vaţno je za napomenuti kako je učenik prije pristupa nekoj od verziji sustava TEx-Sys morao odslušati kratki tečaj u kojem je predstavljen koncept učenja pomoću semantičke mreţe s okvirima. Ovakav ili bilo kakav uvodni tečaj nije potreban kako bi se učenik uveo u rad na prototipnoj verziji novog modelu ITS-a jer se komunikacija odvija na kontroliranom jeziku.

U *drugom poglavlju* se iznosi zamisao modela inteligentnog tutorskog sustava temeljenog na ontološkom prikazu znanja i koji komunikaciju temelji na kontroliranom jeziku. U skladu s tradicionalnom arhitekturom se opisuje koncept novog modela ITS-a koji uključuje alat za opisivanje područnog znanja i alat za obradu kontroliranog jezika. Nadalje se iznosi struktura novog modela temeljena na tradicionalnoj modularnoj arhitekturi ITS-a. Osim toga se predlaţu funkcionalnosti svakog od modula kao i pristupi modeliranju podataka nad kojima se realiziraju funkcionalnosti. Prilikom opisivanja svakog od modula navode se i tradicionalni pristupi u njihovom modeliranju. Posebna paţnja je usmjerena na komunikacijski modul koji je odgovoran za komunikaciju na kontroliranom jeziku. Uz osvrt na pristupe u obradi prirodnog jezika ujedno se iznosi koncept kontroliranog jezika, kao i zamisao njegove obrade.

Treće poglavlje pruţa analizu struktura i funkcionalnosti za četiri odabrana inteligentna tutorska sustava koji koriste obradu prirodnog jezika. Za svaki je naveden i analiziran primjer dijaloga s učenikom te su istaknute funkcionalnosti pojedinih modula kao i koncept njihovih ulaznih i izlaznih podataka. Na kraju je napravljena usporedba sa stajališta njihovih općih karakteristika i s obzirom na obradu i korištenje prirodnog jezika. Osim toga su naglašene zajedničke karakteristike usporeĎenih inteligentnih tutorskih sustava i vlastitog pristupa realiziranog u okviru modela CoLaB Tutor-a (Controlled Language Based Tutor).

Formalni model CoLaB Tutor-a je prikazan u *četvrtom poglavlju*, koje započinje opisom CoLaB Tutor-a preko njegovih sudionika i funkcionalnosti. Izdvojeno je grupiranje funkcionalnosti čime se uspostavljaju tri osnovne faze CoLaB Tutor-a. Nadalje je prikazana struktura CoLaB Tutor-a u kojoj se ističu osnovne komponente i skupovi podataka. Slijedi formalna definicija modela osnovnih skupova podataka temeljena na teoriji skupova i teoriji grafova. Kao poseban skup podataka se formalno opisuje kontrolirani jezik i to na razinama

riječi, fraza i rečenica. Realizacija funkcionalnosti je provedena po fazama: *prva* faza opisuje metode koje se mogu primijeniti prilikom ontološkog opisivanja znanja; *druga* prikazuje realizaciju funkcionalnosti postavljanja u kojoj se od ontološkog opisa znanja dobiva područno znanje, nastavni sadrţaj i inicijalni model učenika, a ujedno se formira i rječnik kontroliranog jezika; *treća* faza je odgovorna za tijek učenja i testiranja znanja učenika, pa se iznose modeli strojeva s konačnim brojem stanja radi realizacije tijeka učenja i testiranja. Stroj za učenje omogućava slijed rečenica kontroliranog jezika koje se prikazuju učeniku pa se opisuje proces generiranja rečenica na osnovu područnog znanja. Za realizaciju tutorskog dijaloga je odgovoran stroj za testiranje te se opisuje proces generiranja pitanja i povratnih informacija učeniku, kao i proces prepoznavanja učenikovog odgovora. Kako bi se odredila točnost učenikovog odgovora, stroj za testiranje suraĎuje s mehanizmom zaključivanja nad područnim znanjem. Na kraju ovog poglavlja se opisuju principi rada mehanizama zaključivanja koji su zasnovani na pravilima i na mreţnom prikazu znanja.

U *petom poglavlju* je iznesena arhitektura prototipa CoLaB Tutor-a te provedena analiza eksperimenta na implementiranom prototipu. Opis arhitekture započinje prikazom komponenti koje se nadalje razraĎuju do klasa i objekata. Prvo se prikazuje arhitektura inteligentnog tutorskog sustava CoLaB Tutor-a kao i klase modela skupova podataka i modela strojeva s konačnim brojem stanja. Zatim se opisuje struktura podsustava za obradu kontroliranog jezika, nakon čega slijedi klasni prikaz arhitekture prototipa aplikacije kojom se realizira Web okruţenje za učenje i testiranje. Nadalje se opisuju instrumenti, tijek i okruţenje za rad eksperimentalnih istraţivanja provedenih na dvije skupine ispitanika. Ujedno se interpretiraju rezultati svake eksperimentalne skupine kao i informacije prikupljene anketnim upitnikom.

U *šestom poglavlju* se daje osvrt na doktorsku disertaciju po poglavljima i iznose se pojedini znanstveni doprinosi.

Spisak korištene literature se nalazi u *sedmom poglavlju*.

Prilozi rada iz *osmog poglavlja* sadrţe razvijenu ontologiju namijenjenu poučavanju i testiranju učenika, primjer zahtjeva i odgovora HML-a, primjer zahtjeva za generiranje rečenica kontroliranog jezika, dijagrame klasa skupova podataka, dijagrame objekata strojeva s konačnim brojem stanja prototipne verzije sustava i anketne upitnike provedenog eksperimentalnog istraţivanja.

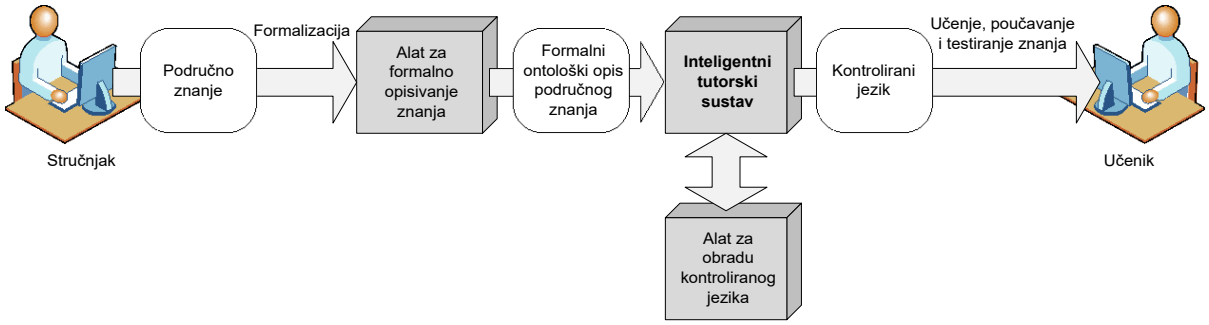
2 Zamisao novog modela inteligentnog tutorskog sustava

Standardna arhitektura inteligentnog tutorskog sustava uključuje komponente koje imaju znanje o područnom znanju, znanju učenika i znanje o voĎenju procesa učenja, poučavanja i testiranja [SELF1990]. Zasnovanost komponenti inteligentnog tutorskog sustava na navedenim vrstama znanja je proizašla iz njegovog tradicionalnog modela koji uključuje modul stručnjaka, model učenika i modul učitelja, te korisničko sučelje prema učeniku [BURN1988]. U tradicionalnom modelu modul stručnjaka sadrţi područno znanje, modul učenika dijagnosticira znanje učenika i stvara model učenika, a modul učitelja identificira znanje koje učeniku nedostaje i provodi strategije za učenikovo usvajanje znanja. Korisničkim sučeljem se ostvaruje komunikacijski kanal izmeĎu modula učitelja, odnosno računalnog tutora i učenika.
Očito je kako se nemonolitna struktura inteligentnih tutorskih sustava realizira komponentama koje su temeljene na spoznajama iz brojnih područja umjetne inteligencije. MeĎutim spoznaje iz područja inteligentnih tutorskih sustava takoĎer utječu na rezultate istraţivanja iz raznih područja umjetne inteligencije, pri čemu se ističu tri značajne skupine problema [SLEB1986]:

- Implementacija prijateljskog korisničkog sučelja i konverzacijskih sustava,
- Modeliranje učenika induktivnim tehnikama i
- Specijalne tehnike deduktivnog zaključivanja u modulu učenika.

Tehnike umjetne inteligencije, koje se bave navedenim problemima, zastupljene su i u modulima novog modela inteligentnog tutorskog sustava. MeĎutim, komunikacija izmeĎu čovjeka i računala kod inteligentnog tutorskog sustava je u fokusu ovog rada jer se obradom kontroliranog jezika rješavaju komunikacijske poteškoće utvrĎene u modelu TEx-Sys. Komunikacija izmeĎu učenika i računalnog tutora u novom modelu ostvaruje se jezikom koji je gramatički i leksički podskup hrvatskog standardnog jezika i pri tome zadrţava svojstvo strojne obradivosti. Zbog toga se nad formalizmom područnog znanja definira kontrolirani jezik koji se učeniku prezentira kao hrvatski standardni jezik. S tim u vezi od učenika se ne zahtjeva poznavanje formalizma za opisivanje područnog znanja što znatno ubrzava prihvaćanje ovakvih sustava.

Novi model inteligentnog tutorskog sustava zasniva se na formalnom ontološkom opisu područnog znanja i na kontroliranom jeziku kao komunikacijskom mediju u procesu učenja, poučavanja i testiranja znanja. Sintaksa kontroliranog jezika temeljena je na formalizmu ontološkog opisa. Stručnjak oblikuje područno znanje i uz pomoć alata za formalno opisivanje stvara ontološki opis područnog znanja (slika 2.1).

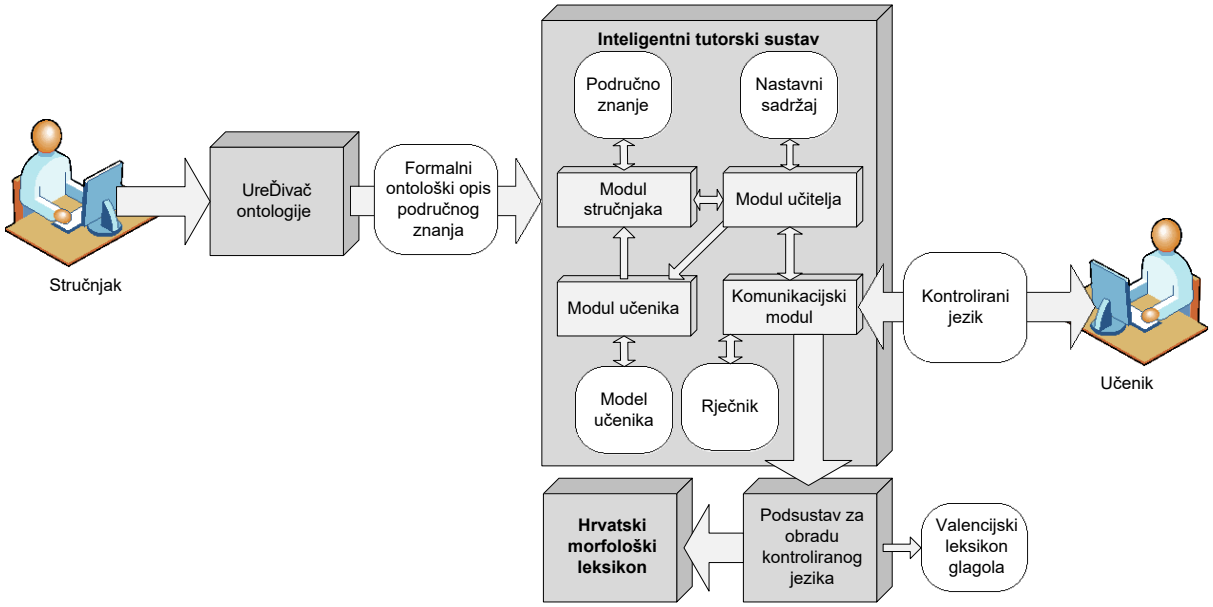


Slika 2.1. Novi model inteligentnog tutorskog sustava

Područno znanje je formalno ontološki opisano, stoga stručnjak najčešće koristi neki alat za formalizaciju područnog znanja. Nakon postavljanja područnog znanja, inteligentni tutorski sustav je spreman za voĎenje procesa učenja, poučavanja i testiranja znanja na kontroliranom jeziku. Vaţno je napomenuti kako se obrada kontroliranog jezika vrši pomoću zasebnog alata. Kad je u pitanju pogled prema učeniku tada izraţajnost komunikacije ovisi o izraţajnosti formalizma prikaza područnog znanja. Područno znanje nosi informacije potrebne za oblikovanje strukture i plana izlaganja nastavnog sadrţaja, dijagnosticiranje modela učenika i za komunikaciju s učenikom.

Struktura novog modela inteligentnog tutorskog sustava se temelji na standardnoj arhitekturi u koju se osim modula stručnjaka, modula učenika i modula učitelja uvodi i komunikacijski modul. Komunikacijski modul opravdava istraţivanja osnovne zamisli tradicionalne arhitekture inteligentnih tutorskih sustava, što je rezultiralo proširenjem korisničkog sučelja [WENG1987], a modul stručnjaka postaje dijelom novog modula zvanog modul područnog znanja [WOOL1992].

Temeljna zamisao novog modela dalje se dekomponira prikazom strukture inteligentnog tutorskog sustava pri čemu se naglašavaju meĎuzavisnosti modula (slika 2.2). UreĎivač ontologije je alat koji sluţi stručnjaku za formalno opisivanje znanja. Alat za obradu kontroliranog jezika uključuje podsustav za obradu kontroliranog jezika, morfološki leksikon hrvatskog standardnog jezika i valencijski leksikon glagola.



Slika 2.2. Struktura alata za formalno opisivanje znanja, inteligentnog tutorskog sustava i alata za obradu kontroliranog jezika

Modul stručnjaka inteligentnog tutorskog sustava je oblikovan po modelu prozirne kutije [ANDE1988] gdje je područno znanje deklarativne prirode, a pristup zaključivanju je analitički. Formalni ontološki opis područnog znanja je ulazni skup podataka modela stručnjaka. Na osnovu ovog opisa modul stručnjaka formira područno znanje. U modulu stručnjaka se analizira ontološki opis područnog znanja radi odreĎivanja meĎusobne povezanosti koncepata. Povezanost koncepata relacijama unutar područnog znanja je bitna informacija koja modulu učitelja omogućava oblikovanje modela nastavnog sadrţaja. Model nastavnog sadrţaja je opisan nizom objekata nastavnog sadrţaja pri čemu svaki objekt nastavnog sadrţaja sadrţi niz koncepata područnog znanja. Prilikom oblikovanja strukture i

objekata nastavnog sadrţaja modul učitelja proučava zavisnost koncepata te na toj osnovi niţe koncepte unutar objekata nastavnog sadrţaja. Na posljetku, modul učitelja niţe same objekte nastavnog sadrţaja kako bi se definirala struktura nastavnog sadrţaja. Osim oblikovanja modela nastavnog sadrţaja modul učitelja je odgovoran za voĎenje procesa učenja, poučavanja i testiranja znanja, pri čemu se testiranje realizira tutorskim dijalogom. Za vrijeme tutorskog dijaloga, modul učitelja suraĎuje s modulom stručnjaka i komunikacijskim modulom, radi prezentiranja pitanja, analiziranja učenikovih odgovora i pruţanje pomoćnih objašnjenja. TakoĎer, uz pomoć modela učenika, modul učitelja planira daljnje akcije koje utječu na voĎenje procesa učenja, poučavanja i testiranja znanja.

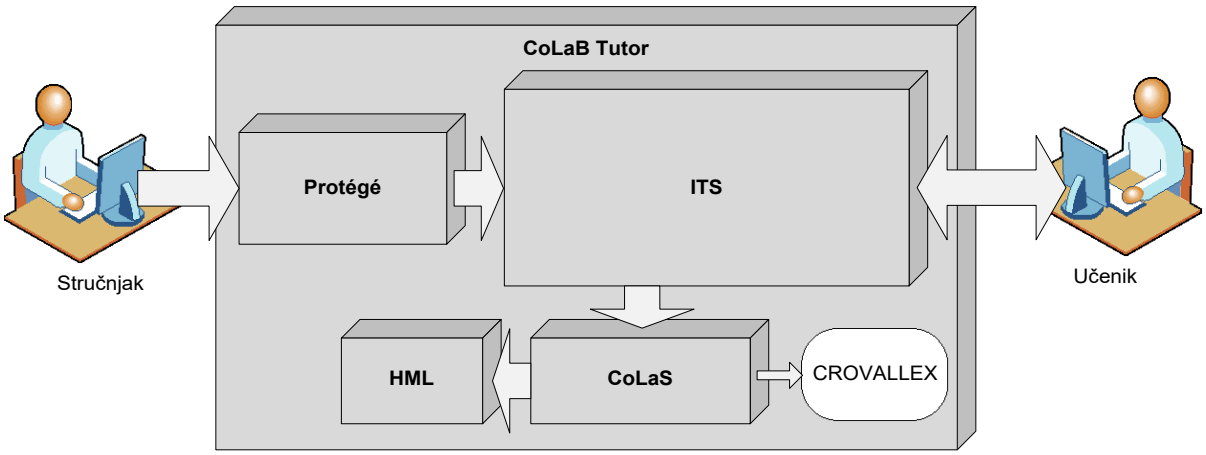
Modul stručnjaka metodom prekrivanja usporeĎuje znanje učenika sa znanjem stručnjaka, odnosno područnim znanjem. Praćenje učinka učenika se opisuje nizom konačnih stanja koje učenik realizira tijekom tutorskog dijaloga. Pošto informacije u modelu učenika utječu na tijek procesa učenja, poučavanja i testiranja znanja, bitno je da one budu čitljive od strane modula učitelja.
Interakcija inteligentnog tutorskog sustava s učenikom se realizira preko komunikacijskog modula. U novom modelu inteligentnog tutorskog sustava komunikacijski modul posjeduje korisničko sučelje tekstualnog tipa. Tekst koji se prezentira učeniku je generiran na osnovu formalizma područnog znanja uz pomoć kontroliranog jezika. Za vrijeme testiranja znanja učenik unosi tekst kao odgovor na postavljena pitanja, koji se prepoznaje unutar istog kontroliranog jezika.
Generiranje i prepoznavanje kontroliranog jezika podrţano je s leksičkom bazom hrvatskog morfološkog leksikona kako bi se ustvrdio potreban oblik riječi kontroliranog jezika. U kontroliranom jeziku koriste se one riječi koje oblikuju imena koncepata i relacija aktualnog područnog znanja. Zbog toga komunikacijski modul sadrţi rječnik koji se mijenja ako se inteligentni tutorski sustav postavi nad nekim drugim područnim znanjem.

Izloţena je cjelokupna zamisao modela inteligentnog tutorskog sustava zasnovanog na obradi kontroliranog jezika nad ontologijom namijenjena učenju, poučavanju i testiranju znanja učenika kojeg smo nazvali Controlled Language based Tutor-a (CoLaB Tutor).

Okruţenje CoLaB Tutora je oslonjeno na:

- formalni opis znanja CoLaB Tutor-a uz pomoć ureĎivača ontologije Protégé [KNUB2004a] u kojemu se ontologija opisuje OWL jezikom,
- alat za obradu kontroliranog jezika CoLaB Tutor-a sastoji se od podsustava Controlled Language Service (CoLaS) namijenjenog generiranju i prepoznavanju kontroliranog jezika,
- uslugu odreĎivanja morfoloških oblika riječi hrvatskog standardnog jezika implementiranih u sustavu Hrvatski morfološki leksikon (HML) [TADI2003] i
- Hrvatski valencijski leksikon glagola (CROVALLEX) [MIKE2008] kojeg koristi podsustav CoLaS za odreĎivanje valencije glagola.

Podsustavi i skupovi podataka koji čine okruţenje CoLaB Tutor-a su prikazani na slici 2.3.



Slika 2.3. Okruţenje CoLaB Tutor-a

U daljnjim podpoglavljima ovog poglavlja navodi se teoretska podloga za implementaciju i postavljanje pojedinih modula kao i njihovih skupova podataka unutar novog modela inteligentnog tutorskog sustava. S tim u vezi, zamisao novog modela se dalje dekomponira do razine modula i obuhvaća: modul stručnjaka (područno znanje), modul učitelja (nastavni sadrţaj), modul učenika (modeliranje učenika) i komunikacijski modul (kontrolirani jezik). Na ovaj način su postavljeni temelji inteligentnog tutorskog sustava CoLaB Tutor.

2.1 Modul stručnjaka i područno znanje

Modul stručnjaka je povezan s konceptualnim modelom koji opisuje znanje ili ponašanja unutar ekspertize odreĎenog područnog znanja. On je okosnica svakog inteligentnog tutorskog sustava jer kvaliteta učenja, poučavanja i testiranja znanja direktno ovisi o kvaliteti područnog znanja. Najčešća su sljedeća tri pristupa modeliranju i prikazivanja područnog znanja u modulu stručnjaka [ANDE1988].

Prvi pristup koristi model crne kutije kojim se interno funkcioniranje modela sakriva od vanjskog promatrača. Ulazno-izlazne informacije modela crne kutije jednostavno nisu pogodne za učenje i poučavanje jer nisu u stanju izloţiti proces zaključivanja učeniku.

U *drugom pristupu* upotrebljava se model prozirne kutije. Prilikom izgradnje ovakvog modela koristi se neka metoda inţenjerstva znanja kako bi se ljudsko znanje prenijelo i prikazalo uz pomoć računala.

Za razliku od modela prozirne kutije kojim se simulira ljudsko znanje, u *trećem pristupu* se simulira i način korištenja ljudskog znanja. Ovaj pristup se referencira kao kognitivno modeliranje jer se njim opisuje proces rješavanja problema i ljudskog razmišljanja.

Na oblikovanje modula stručnjaka utječe i vrsta znanja koja se prikazuje. Proceduralno i deklarativno znanje su osnovne vrste znanja kojim se opisuje odreĎeno područno znanje. Prilikom formalizacije proceduralnog znanja koriste se produkcijska pravila, dok za formalizaciju deklarativnog znanja pogoduje formalni jezik simbola ili neka druga shema prikazivanja znanja. Tradicionalno se prikazivanje znanja zasniva na izvjesnom analitičkom načinu korištenjem formalnog logičkog jezika, meĎutim ljudsko zaključivanje posjeduje heurističke osobine, pa se načini prikazivanja znanja općenito mogu podijeliti na analitički i heuristički [EVAN2003] [HONK2007].

Analitički način prikazivanja znanja omogućava logičko zaključivanje. Formalni logički jezici, produkcijska pravila i semantičke mreţe su neke od osnovnih tehnika analitičkog načina prikazivanja znanja.

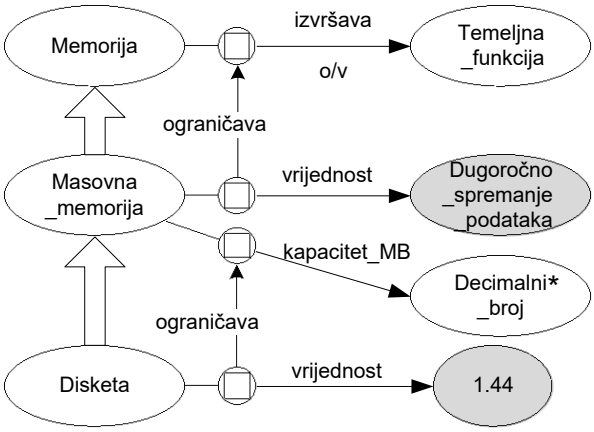
Heuristički način se razlikuje od analitičkog po tome što ne koristi strogo definirana pravila u funkciji prikazivanja znanja. Ovaj način prikazivanja znanja se temelji na neodreĎenosti podataka. Za razliku od analitičkog načina, algoritmi kod heurističkog načina prikazivanja znanja ne nude precizna rješenja, već proizvode zadovoljavajuća rješenja ili rješavaju manje probleme koji su povezani s većim problemom. NeodreĎenost heurističkog načina prikazivanja znanja se ističe kod modela kao što su neizrazita logika, neuronske mreţe, stabla odluke, Bayesova mreţa i ostalih. Bitne osobine heurističkog načina prikazivanja znanja su brza i paralelna obrada velikih količina podataka, te pragmatičnost.
Iz navedenih karakteristika prikazivanja znanja i modela sustava zasnovanih na znanju postavljaju se sljedeća pitanja [WAYE1991] koja se mogu vezati za modeliranje područnog znanja u modulu stručnjaka:

1. Koja je priroda znanja i kako se ono prikazuje?
2. Hoće li prikaz znanja biti proceduralan ili deklarativan?
3. Koliko je izraţajan model prikaza znanja?

Odgovorom na *prvo pitanje* se odreĎuje pristup modelu prikazivanja znanja. Većina ekspertnih sustava, nad kojima se zasnivaju moduli stručnjaka koriste kombinaciju crnog i prozirnog modela kutije, čime se djelomično otkriva područno znanje. Kognitivno modeliranje je najsloţeniji pristup kojim se ostvaruje vjerno simuliranje ljudskog znanja i razmišljanja.
Drugo pitanje odreĎuje što se sa znanjem ţeli učenika naučiti. Proceduralnim znanjem se opisuje kako se problemi rješavaju ili kako nešto radi. Većinski se prikazivanje proceduralnog znanja zasniva na pravilima. MeĎu prve inteligentne tutorske sustave temeljeni na pravilima spadaju LISP Tutor [REIS1985], Geometry Tutor [ANDE1986] i BUGGY (tutor namijenjen učenju aritmetičkih operacija) [BROW1980]. U mnogim slučajevima se od učenika traţi poznavanje osnovnih činjenica područnog znanja koje se opisuje deklarativnim prikazom znanja. SCHOLAR [CARB1970] koristi deklarativno znanje za opisivanje geografije Juţne Amerike.
Treće pitanje, odnosno pitanje izraţajnosti prikaza znanja ovisi o mogućnostima opisivanja vanjskog svijeta. Što je mogućnost opisivanja detalja vanjskog svijeta veća, to je jezik prikazivanja znanja izraţajniji. MeĎutim, kod izraţajnijih jezika je teţe vršiti automatizirano zaključivanje, jer su oni često nekonzistentni i nepotpuni, za razliku od manje izraţajnih jezika kao što je propozicijska logika. Upravo izraţajnost prikaza znanja znatno utječe na komunikaciju izmeĎu inteligentnog tutorskog sustava i učenika.

Novi model inteligentnog tutorskog sustava koristi opisnu logiku u formi ontologije kao tehniku prikaza znanja. Nedostaci semantičke mreţe kao što su mogućnost implementacije negacije i disjunkcije prirodno su podrţani u opisnoj logici. Druge značajne osobine su korištenje kvantifikatora prilikom ograničavanja svojstava koncepata, te primjena primitivnih tipova podataka kod specifikacije koncepata.
U računarstvu ontologijom se opisuje konceptualni model nekog područja. Teoretski se ontologija definira kao formalna i eksplicitna specifikacija dijeljene konceptualizacije [GRUB1993]. Konceptualizacija je pojednostavljeno i apstraktno viĎenje vanjskog svijeta, a osnovni strukturni elementi konceptualnog modela su koncepti i relacije koji čine formalni prikaz znanja [GENE1987]. Jedna od formalnih osnova ontologije je logika. Formalnim logičkim prikazom konceptualnog modela se simboličkim proračunom iz eksplicitne specifikacije znanja dobiva implicitno znanje. Osim ovih osobina ontologije, prenosivost prikaza znanja je karakteristika koja omogućava istraţivačima dijeljenje i ponovnu upotrebu ontologije bez gubljenja svojstva strojnog proračuna.

Izraţajnost ontologije ovisi o izraţajnosti formalnog logičkog jezika na kojem se temelji. Opisna logika je temeljni formalni logički jezik ontologije. Izraţajnost joj je veća od izraţajnosti semantičke mreţe jer posjeduje većinu elemenata predikatne logike prvog reda. Moţe se reći kako opisna logika ujedinjuje prednosti grafičke notacije semantičke mreţe s formalnom logikom koja je podskup predikatne logike prvog reda. Opisni jezici su nasljednici jezika Knowledge Language One (KL-ONE) [BRAC1977] koji ima izrazitu grafičku notaciju [BRAC1979] (slika 2.4) pošto nasljeĎuje svojstva semantičke mreţe s okvirima.



Slika 2.4. Grafička notacija KL-ONE jezika

Na slici 2.4 koncepti su označeni elipsama, dok strelice označavaju različite veze. Bijelim elipsama su označeni generički koncepti, za razliku od sivih elipsi koje označavaju individualne koncepte. Generički koncept Decimalni_broj je označen s "*" kako bi se istaknuo ugraĎeni primitivni tip podatka. Strelice s dvostrukom linijom prikazuju klasifikaciju koncepata, dok strelice s krugom u sredini predstavljaju uloge. Na primjer Masovna_memorija ima dvije uloge pri čemu jedna ograničava ulogu izvršava koncepta Memorija. Neke uloge su označene s o/v što znači da ograničavaju vrijednosti te uloge kod opisa koncepta.
U opisnoj logici se koristi termin uloga koji je ekvivalent relaciji u ontologiji. Terminološka usklaĎenost ontologije i deskriptivne logike je prikazana u tablici 2.1.

Tablica 2.1. Usporedba termina ontologije i opisne logike

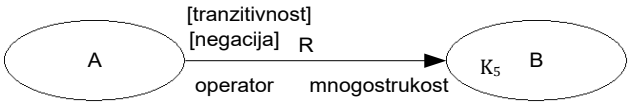
Ontologija	**Opisna logika**
Generički koncept	Klasa
Individualni koncept	Individua
Relacija	Uloga

Logička notacija opisne logike posjeduje jednostavnije simboličke oznake od predikatne logike prvog reda. Sintaksa opisne logike sastoji se od unarnih predikata za odreĎivanja naziva koncepata, binarnih predikata za odreĎivanje naziva uloga i rekurzivnih definicija konstruktora novih naziva koncepata preko postojećih naziva koncepata i uloga. Naziv koncepata i naziv uloge su simboli koji sluţe za identifikaciju koncepata i uloge.
Postoje brojne familije opisne logike koje sluţe za odreĎivanje njene izraţajnosti preko dozvoljenih konstruktora i proširenja svojstava uloga [BAAD2003]. Atributni jezik $\mathcal{AL}$ smatra se temeljnim jezikom opisne logike kojim se dozvoljava disjunkcija koncepata i negacija atomskih koncepata te univerzalna i ograničena egzistencijalna kvantifikacija uloge. MeĎutim, jezik $\mathcal{ALC}$ upotpunjuje jezik $\mathcal{AL}$ konjunkcijom i negacijom svih koncepata i potpunom egzistencijalnom kvantifikacijom uloge [SCHA1991] te se on koristi za usporeĎivanje s drugim opisnim logikama.

U novom modelu inteligentnog tutorskog sustava područno znanje je prikazano $\mathcal{SHOIN}^{(\mathcal{D})}$ familijom opisne logike. Ova familija proširuje $\mathcal{ALC}$ jezik s nabrajanjem koncepata, kardinalnim ograničavanjem uloge, te uvoĎenjem funkcionalnih, simetričnih, refleksivnih, tranzitivnih i inverznih uloga. TakoĎer omogućava stvaranje hijerarhije uloga i dozvoljava definiranje svojstava nad primitivnim tipovima podataka. U (2.1) su linearnom notacijom opisne logike prikazani koncepti iz primjera sa slike 2.4.

$$\begin{aligned} Memorija &\sqsubseteq \exists izvrsava.Temeljna\_funkcija \\ Masovna\_memorija &\\ &\sqsubseteq Memorija \sqcap izvrsava.\{Spremanje\_podataka\} \\ &\sqcap \exists kapacitet\_MB.Decimalni\_broj \\ Disketa &\sqsubseteq Masovna\_memorija \sqcap kapacitet\_MB.\{1.44\} \end{aligned} \tag{2.1}$$

Proširenjem konstruktora koncepata i uloga, te proširivanjem osobina uloga raste izraţajnost jezika opisne logike. Sintaksa $\mathcal{SHOIN}^{(\mathcal{D})}$ jezika sasvim je pogodna za opisivanje ontologije korištenjem sloţenih izjava dosta sličnim izjavama prirodnog jezika.
Razlog njegovog izbora kao jezika za opisivanje područnog znanja u novom modelu inteligentnog tutorskog sustava jest što je on ujedno i temelj OWL DL (Web Ontology Language Description Logic) standardnog jezika za opisivanje ontologija na Web-u. Osim podrţavanja $\mathcal{SHOIN}^{(\mathcal{D})}$ jezika opisne logike, motivacija prihvaćanja OWL DL jezika za opisivanje područnog znanja u novom modelu inteligentnog tutorskog sustava je upravo njegova standardizacija. Namjena ove standardizacije je prije svega povezivanje koncepata na Web-u, a orijentiranost prema Web-u je dodatna pozitivna karakteristika.
Modul stručnjaka provodi sintaktičnu analizu ontološki opisanog znanja kako bi stvorio područno znanje. Model područnog znanja je opisan meĎusobno povezanim konceptima preko relacija. Na slici 2.5 koncept A je povezan relacijom R s konceptom B.



Slika 2.5. Koncepti i relacije područnog znanja

Relacija R moţe biti negirana i moţe imati svojstvo tranzitivnosti. Negacija relacije R označava da koncept A nije u relaciji R s konceptom B. Tranzitivnost relacije R je bitna kod zaključivanja, jer ako je recimo koncept B povezan relacijom R s konceptom C, onda se zaključuje da je koncept A povezan s konceptom C preko tranzitivne relacije R. Operator relacije moţe biti ili konjunkcija ili disjunkcija. Neka je operator primjera sa slike 2.5 konjunkcija, te neka postoji veza koncepta A s nekim konceptom D preko relacije R takoĎer označena konjunkcijom, tada zaključujemo da je koncept A u relaciji R s konjunkcijom koncepata B i D. Mnogostrukost relacije R govori o broju primjeraka koncepta B na koje je povezan koncept A. Negacija, tranzitivnost, operatori i mnogostrukost su osobine relacije R koje, osim što utječu na proces zaključivanja, sluţe i kod oblikovanja izjava kontroliranog jezika kao što je opisano u poglavlju 2.4.3.1.

2.2 Modul učitelja i model nastavnog sadrţaja

Modul učitelja, odnosno računalni tutor vodi proces učenja, poučavanja i testiranja znanja. Na osnovu tri tutorske karakteristike inteligentnog tutorskog sustava se definiraju osnovne funkcionalnosti voĎenja modula učitelja [HALF1988]:

- Upravljanje slijedom i sadrţajem objekata nastavnog sadrţaja,
- Sposobnost odgovaranja na upite učenika vezanih za nastavni sadrţaj i
- Strategije pomaganja učeniku.

Inteligentni tutorski sustav mora, u nekom obliku, podrţavati sve tri funkcionalnosti. U modelu učitelja se osnovne funkcionalnosti temelje na nastavnom sadrţaju i njegovoj realizaciji. Pod nastavnim sadrţajem se smatra izbor i slijed objekata nastavnog sadrţaja, dok se dostava objekta nastavnog sadrţaja smatra realizacijom nastavnog sadrţaja. Problemi koji se postavljaju pred nastavni sadrţaj su oblikovanje inicijalne strukture objekta nastavnog sadrţaja te izbor i organizacija slijeda te reprezentacije. Principi izbora i slijeda objekta nastavnog sadrţaja prije svega ovise o vrsti znanja koje inteligentni tutorski sustav poučava. Kod deklarativnog znanja identificirana su temeljna dva principa takozvanog "mreţnog učenja" (engl. web learning) [NORM1973]:

- Princip davanja prioriteta konceptima koji su u bliskoj vezi s postojećim znanjem
- Princip generalizacije gdje se kreće od općih koncepata prema specifičnim.

Na osnovu ovih principa se postavlja statička osnova nastavnog sadrţaja, koja nije u stanju dinamički oblikovati nastavni sadrţaj tijekom procesa učenja, poučavanja i testiranja znanja. Dinamička metodologija u programu Meno-tutor posjeduje dva mehanizma za usmjeravanje tutorskog dijaloga [WOOL1985]:

- Mehanizam planiranja koji je zasnovan na mreţnom učenju i kojim se nastoji odrţati fokus u dijalogu i
- Mehanizam odgovaranja na učenikovu trenutno situaciju s predlaganjem izmjena u početnoj putanji slijeda tutorskog dijaloga.

Za razliku od deklarativnog znanja gdje se usvajaju činjenice, kod proceduralnog znanja se uče i poučavaju proceduralne vještine kroz primjere i vjeţbe te se postavlja pitanje slijeda primjera i vjeţbi. Istraţivanje vezano za izbor i slijed vjeţbi je predloţilo tri standarda.
Prvi standard navodi kako svaka vjeţba mora biti rješiva i svaki primjer mora biti razumljiv učeniku koji je prešao prethodne objekte nastavnog sadrţaja. U ovom smislu se predlaţe hijerarhijska struktura ciljeva instrukcija [GAGN1979] kojima se ujedno postavljaju temelji računalnog instrukcijskog dizajna.
U *drugom standardu* se naglašava strukturalna transparentnost, odnosno slijed vjeţbi i primjera mora odraţavati strukturu procedure koja se poučava. Za posljedicu učenik mora biti u stanju ostvariti proceduru koju je naučio. Ovi zahtjevi strukturalne transparentnosti su ostvareni modelom učenja gdje svaki objekt nastavnog sadrţaja posjeduje predloţak kojim se definira slijed primjera i vjeţbi po induktivnom principu [SMIT1982]. Induktivni princip je zasluţan za izvlačenje zaključaka iz objekta nastavnog sadrţaja. U teoriji koraka (engl. step theory) se od učenika traţi stvaranje zaključaka uz praćenje učenikovog rješavanja problema [VANL1983a].
Individualizacija je *treći standard* kojim se poučavanje prilagoĎava učeniku. Za razliku od statičkih osobina prethodna dva standarda, individualizacijom se dinamički oblikuje nastavni sadrţaj tijekom poučavanja učenika. BIP-II posjeduje modul učitelja koji u suradnji s modelom učenika i proceduralnim modulom stručnjaka prilagoĎava strukturu nastavnog sadrţaja pojedinom učeniku [WESC1977].
Osim izbora i slijeda objekata nastavnog sadrţaja, modul učenika vrši inicijalnu prezentaciju objekata nastavnog sadrţaja korištenjem metoda realizacije nastavnog sadrţaja. Proceduralni prikaz znanja inteligentnog tutorskog sustava uvjetuje upotrebu voĎenih primjera i vjeţbi kao metodu za realizaciju nastavnog sadrţaja. Kod inteligentnih tutorskih sustava temeljenih na

deklarativnom prikazu znanja, osnovna metoda realizacije nastavnog sadrţaja je dijalog. Istraţeni su opći ciljevi dijaloga [COLL1975]:

- Poučavanje činjenica i koncepata,
- Poučavanje pravila i relacija,
- Poučavanje vještina za izvoĎenje pravila.

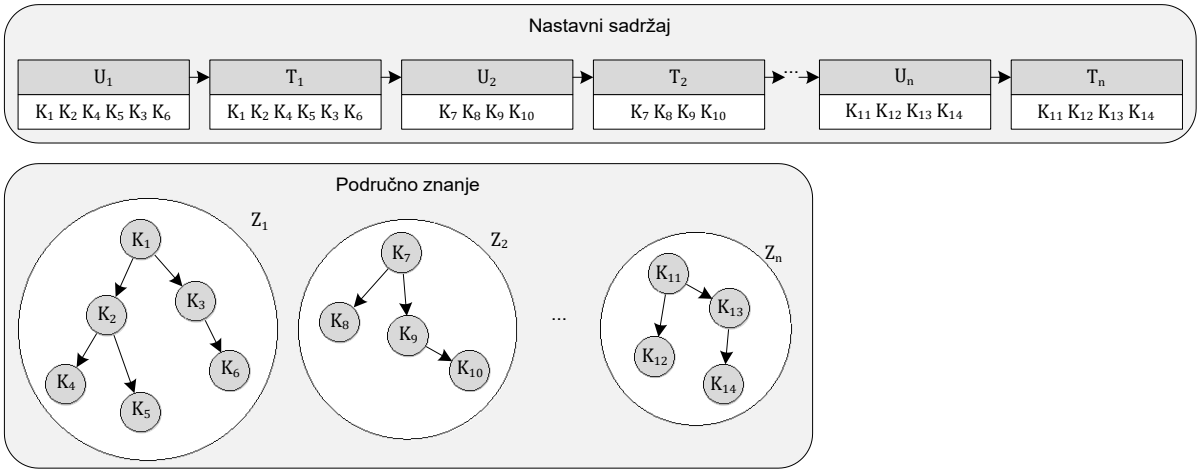
Ovi ciljevi se takoĎer podudaraju s onim predloţenim u sustavu IDS (Instructional Design System) [GAGN1979]. Interaktivnost dijaloga u sustavu WHY ovisi o uvjetovanju radnji modula učitelja na osnovu učenikovih odgovora [COLL1982], ali ovaj dijalog je uokviren čvrstim planom koji se ne moţe preformulirati tijekom interakcije. Meno-Tutor nudi dinamičku fleksibilnost i kod strukture nastavnog sadrţaja i kod njegove realizacije.
Odgovaranje na pitanje je esencijalna funkcija ljudskih tutora, a teškoće obrade prirodnog jezika uvjetovale su njegovu kasnu implementaciju u inteligentnim tutorskim sustavima. Sustav SHOLAR koristi strategiju preklapanja predloţaka pitanja, dok je u sustavu SOPHIE [BROW1982] zastupljen epistemološki i lingvistički pristup problemu. Odgovaranje na učenikovo pitanje u sustavu SOPHIE I i SOPHIE II zahtijevalo je pokretanje matematičkog modela simuliranog ureĎaja. Pošto pokretanje matematičkog modela nije rezultiralo nikakvim zaključivanjem onda SOPHIE I nije bio u stanju objasniti postupak dolaska do rješenja. Tek u SOPHIE III je implementiran dodatni modul kojim se daje uvid u zaključivanje.
Osim odgovaranja na pitanja, računalni tutor moţe promijeniti aktivnost učenikovog učenja intervencijom kako bi pomogao učeniku. IzvoĎenjem pravila pokretanja intervencije i oblikovanjem sadrţaja intervencije se automatizira ova osobina modula učitelja. Postoje dva opća pristupa intervenciji. Modelom praćenja učenika (engl. model tracing) [ANDE1986] se nastoji učenika dovesti na put kojeg bi prelazio idealan učenik u kognitivnom smislu. Pri tome modul učitelja intervenira kad god učenik napusti idealnu putanju. MeĎutim do intervencije moţe doći i kada učenik eventualno izvrši bolju aktivnost učenja od pretpostavljene idealne. Zbog toga problemska intervencija (engl. issue-based intervention) funkcionira i kod manje savršenih modela stručnjaka unutar inteligentnog tutorskog sustava. Za razliku od intervencije temeljene na modelu praćenja koja se pokreće svaki put kad modul učitelja ne moţe pozitivno identificirati učenikovu akciju, problemski pristup predlaţe intervenciju samo onda kada modul učitelja djelomično shvati akciju učenika. Na primjer, inteligentni tutorski sustav WEST [BURT1982a] daje savjete učeniku za vrijeme igranja računalne aritmetičke igre. Igra je opisana brojnim problemima i strategijama koji mogu biti od koristi učeniku kod odreĎenih poteza, a modul učitelja identifikacijom pogreške povremeno podsjeća učenika na pojedine strategije. Kombinacija oba pristupa moţe biti prisutna u inteligentnom tutorskom sustavu. Jedan od primjera je implementacija kataloga pogrešaka (engl. bug catalog). Katalog pogrešaka je skup pravila kojima se identificiraju pogreške, a oslanja se na praćenju modela učenika. Drugi primjer je zastupljen u COACH tutoru koji prati učenje jednostavnog programskog jezika FLOW [GENT1977]. Intervencija u COACH tutoru je zasnovana na modelu koji kombinira strukturu nastavnog sadrţaja sa strukturom programskog jezika FLOW [GENT1979]. Hijerarhijska struktura ovog modela kreće od cjelovitih vjeţba do pojedinih pritisaka na tipke tijekom unošenja naredbi jezika. Intervencija modula učitelja se moţe svesti na vjeţbu i na najmanje segmente vjeţbe kao što je unošenje naredbe. Sadrţaj intervencije je najčešće jednostavna izjava kojom se učenik upozorava na pogreške i nudi se akcija za ispravljanje pogreške. Sofisticiranije metode, osim objašnjavanja razloga nastanka učenikove pogreške, nude i apstraktna rješenja čijom primjenom učenik moţe ispraviti pogrešku.
Navedeni tradicionalni inteligentni tutorski sustavi i dan danas zastupaju temelje oblikovanja modula učitelja. Preseljenjem inteligentnih tutorskih sustava na Web stvara se teţnja standardizaciji skupova podataka inteligentnih tutorskih sustava. Jedan od standarda je

Shareable Content Object Reference Model (SCORM) [ADVA2004] kojim se omogućuje opisivanje i dijeljenje nastavnog sadrţaja meĎu sustavima namijenjenih učenju na Web-u. SCORM je prvenstveno namijenjen za dijeljenje nastavnog sadrţaja kod sustava za poslovanje učenjem (Learning Management Systems - LMS) iako teţi domenu primjene proširiti na inteligentne tutorske sustave. SCORM-ov model agregacije sadrţaja (Content Aggregation Model) opisuje statičku strukturu nastavnog sadrţaja kao stablastu strukturu čiji su listovi objekti nastavnog sadrţaja, a ostali elementi su agregacije koje sluţe za grupiranje objekata nastavnog sadrţaja. Dinamička struktura SCORM-ovog modela nastavnog sadrţaja opisuje slijed i ciljeve objekata nastavnog sadrţaja (Sequencing and Navigation) što se moţe dovesti u vezu s dinamičkom prilagodbom nastavnog sadrţaja kod inteligentnih tutorskih sustava. MeĎutim sami objekti nastavnog sadrţaja SCORM-ovog modela sastoje se od resursa koji su namijenjeni prezentaciji na Web pregledniku i u osnovi su statičke prirode bez inteligentnih svojstava.

Objekti nastavnog sadrţaja novog modela inteligentnog tutorskog sustava su meĎusobno strukturalno uvjetovani i čine slijed koji se opisuje izrazom

$$U_1\ T_1\ U_2\ T_2\ \dots U_n\ T_n \tag{2.2}$$

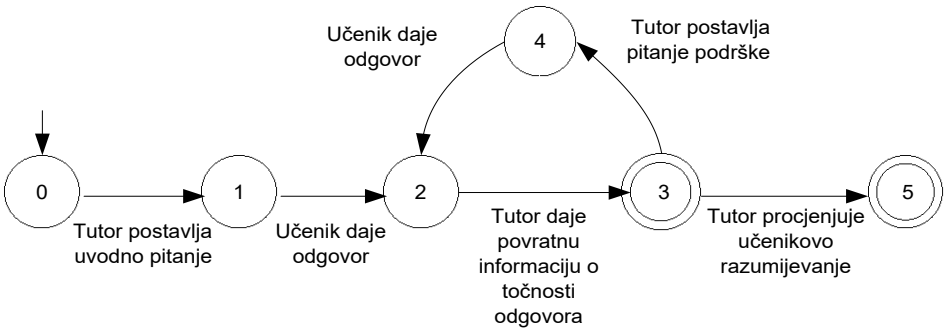
gdje je U oznaka za objekt nastavnog sadrţaja namijenjen učenju, a T oznaka za objekt nastavnog sadrţaja namijenjenog testiranju znanja. Izraz (2.2) opisuje slijed u kojemu učenik prvo uči objekt U_i a nakon njega naučeno znanje testira nad objektom T_i. Ako je učenik zadovoljio test T_i onda nastavlja s učenjem objekta U_{i+1}. Ako učenik nije zadovoljio test T_i tada se vraća na učenje znanja sadrţanog u objektu U_i. Ovim se početni slijed objekata nastavnog sadrţaja dinamički mijenja ovisno o tome je li učenik zadovoljio test.
Sadrţaj objekata učenja se opisuje konceptima područnog znanja, pri tome se slijed koncepata usklaĎuje po pravilima mreţnog učenja [NORM1973] kao što se vidi na slici 2.6.



Slika 2.6. Odnos nastavnog sadrţaja i područnog znanja novog modela inteligentnog tutorskog sustava

Na primjer, objekt nastavnog sadrţaja U_1 referencira skupinu koncepata Z_1 iz područnog znanja po slijedu $K_1\ K_2\ K_4\ K_5\ K_3\ K_6$ pri čemu redoslijed ovisi o poloţaju koncepta u odnosu na druge koncepte. Slijed koncepata objekta U_1 započinje najopćenitijim konceptom K_1, nakon njega dolazi podreĎeni koncept K_2, pa njegovi podreĎeni koncepti K_4 i K_5. K_4 i K_5 nemaju podreĎenih koncepata pa se slijed nastavlja konceptima K_3 i K_6.
I objekt nastavnog sadrţaja namijenjen testiranju T_i referencira iste koncepte kao i U_i, ali kod testiranja postoji mogućnost dinamičke promjene slijeda radi prilagodbe modula učitelja

učenikovim odgovorima na pitanja. Testiranje u novom modelu inteligentnog tutorskog sustava se realizira putem tutorskog dijaloga. Ovaj dijalog je opisan slijedom tutorskih okvira, a svaki tutorski okvir se sastoji od pet općih koraka [GRAE1995]. Modul učitelja novog modela inteligentnog tutorskog sustava provjeru znanja, vezanog za pojedini koncept, realizira putem tutorskog okvira opisanog strojem s konačnim brojem stanja (slika 2.7).



Slika 2.7. Okvir tutorskog dijaloga novog modela inteligentnog tutorskog sustava

Uvodnim pitanjem tutorskog okvira provjerava se učenikovo poznavanje pojedinog koncepta. Nakon što učenik odgovori na uvodno pitanje, modul učitelja provjerava točnost odgovora i daje povratnu informaciju. Ako je učenik pozitivno odgovorio na uvodno pitanje, aktualni tutorski okvir završava i modul učitelja realizira sljedeći tutorski okvir koji je vezan za sljedeći koncept iz slijeda koncepata objekta nastavnog sadrţaja namijenjenog testiranju. U suprotnom, netočnim odgovorom se uvjetuje postavljanje pitanja podrške. Pitanjima podrške modul učitelja nastoji dovesti u vezu netočan učenikov odgovor s traţenim odgovorom. Pri tome modul učitelja nakon svakog učenikovog odgovora na pitanje podrške daje povratnu informaciju. Postavljanje pitanja podrške je u uskoj vezi s modelom praćenja učenika jer svaki učenikov odgovor utječe na dinamičku prilagodbu modula učitelja koja se očituje generiranjem novog pitanja podrške na osnovu dosadašnjeg stanja učenikovog znanja. Ovom prilagodbom se eventualno mijenja početno zacrtani slijed koncepata u objektu nastavnog sadrţaja namijenjenog testiranju. Ukoliko se pokazalo potrebnim postaviti jedno ili više pitanja podrške, onda modul učitelja završava aktualni tutorski okvir objašnjenjem kojim učeniku ukazuje na povezanost učenikovih odgovora s traţenim odgovorima uvodnog pitanja i pitanja podrške. Modul učitelja tijekom objašnjavanja takoĎer koristi model učenika kako bi objasnio slijed procesa zaključivanja kojim se dovodi u vezu odgovor učenika s traţenim odgovorom.

2.3 Modul učenika i modeliranje učenika

Inteligentni tutorski sustav mora biti u stanju pratiti i dijagnosticirati znanje učenika kako bi mu se moglo prilagoditi na individualnom nivou. Problematika modeliranja učenika u inteligentnom tutorskom sustavu odreĎena je oblikovanjem modela učenika i oblikovanjem modula učenika. Model učenika je struktura podataka koja odraţava stanje učenikovog znanja. Modul učenika ili dijagnostički modul je proces koji oblikuje model učenika na osnovu ulaznih informacija prikupljenih u komunikaciji izmeĎu računalnog tutora i učenika. Postoje brojne primjene modela učenika u inteligentnim tutorskim sustavima, a sljedeće četiri se najčešće pojavljuju u praksi [VANL1988]:

- Napredovanje – je primjena modela učenika kojim se odreĎuje prijelaz na odreĎeni objekt nastavnog sadrţaja. Ovaj pristup nije samo pogodan za linearni slijed objekata nastavnog sadrţaja, već i za istovremeno kombiniranje objekata nastavnog sadrţaja.

Na primjer, kod WUSOR tutora [GOLD1982] učenik istovremeno koristi nekoliko vještina kako bi savladao igru hvatanja zvijeri u pećini.

- Zahtijevanje savjeta – je karakteristika nekih inteligentnih tutorskih sustava kod kojih modul učitelja neće ponuditi savjet sve dok je učenik ne zatraţi. Kako bi inteligentni tutorski sustav bio u stanju ponuditi savjet, potrebno je poznavanje stanja znanja učenika u tom trenutku.
- Generiranje problema – je karakteristika inteligentnih tutorskih sustava koji praćenjem modela učenika predviĎaju teţinu problema koji će se dinamički oblikovati.
- PrilagoĎavanje objašnjenja – je primjena kod koje se objašnjenja učeniku temelje na onome što je učenik već naučio.

Ova klasifikacija problema modeliranja učenika se temelji na izlaznim i ulaznim podacima modula učenika. Druga klasifikacija problem modeliranja učenika opisuje trodimenzionalno, gdje prva dimenzija predstavlja ulazne podatke, a ostale dvije strukturalna svojstva modula učenika [VANL1988].

Obuhvat ulaznih podataka (engl. bandwith) je *prva dimenzija* kojom se opisuje količina i kvaliteta ulaznih podataka modula učenika. Postoje tri nivoa obuhvata ulaznih podataka koje su meĎusobno uključujuće. Pretpostavljanje mentalnih stanja je najširi obuhvat ulaznih podataka kojim se nastoji obuhvatiti i pretpostaviti kognitivna stanja učenika tijekom rješavanja problema. MeĎustanja su dio mentalnih stanja kojima se obuhvaćaju uočljivi koraci, dok je konačno stanje posljednje meĎustanje kojim se očituje trenutak kada je problem riješen. LISP tutor [REIS1985] je primjer sustava koji, pomoću izbornicima voĎenim korisničkim sučeljem, prati mentalna stanja učenika. Drugi primjer je PROUST [JOHN1984] sustav namijenjen učenju programskog jezika PASCAL. Model učenika sustava PROUST prati samo konačna stanja jer učenik rješava zadatak pisanjem izvornog koda, a tek kompiliranjem se ustanovljava stanje učenikovog znanja.

Ciljana vrsta znanja je *druga dimenzija* koja utječe na sloţenost dijagnosticiranja modela učenika u smislu pronalaţenja rješenja problema. Dijagnosticiranje nad proceduralnim znanjem je jednostavnije od deklarativnog znanja jer odluku temelji na trenutnom problemu i lokalnom znanju. S obzirom na sloţenost, proceduralno znanje koje ima hijerarhijski organizirane podciljeve je sloţenije od proceduralnog znanja bez podciljeva. Naime proceduralno znanje bez podciljeva na osnovu trenutnog stanja problema i stanja učenika moţe jednostavno predvidjeti sljedeću operaciju. Odlučivanje kod proceduralnog znanja s podciljevima modul učitelja, osim na problemu i stanju učenika, temelji i na podciljevima. Dijagnosticiranje deklarativnog znanja je ipak najsloţenije jer predviĎanje najčešće uključuje skup činjenica, a svaka činjenica moţe voditi prema rješenju problema.

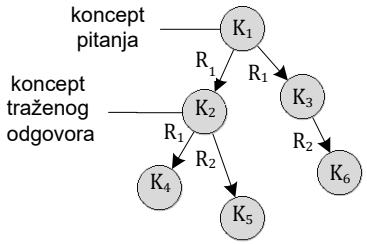
Razlike izmeĎu učenika i stručnjaka odreĎuju *treću dimenziju* jer većina inteligentnih tutorskih sustava koristi isti prikaz znanja kod modula stručnjaka i modula učenika. U ovom slučaju se model učenika prikazuje kao područno znanje zajedno sa skupom razlika. Skupine razlika znanja se dijele na:

- nedostajuće poimanje (engl. missing conception), odnosno znanje koje stručnjak ima, a učenik nema i na
- pogrešno poimanje (engl. misconception) odnosno znanje koje učenik ima a stručnjak nema.

Većina modula učenika dijagnosticiranje temelji na modelu prekrivanja (engl. overlay model) [CARR1977] gdje se model učenika zasniva samo na nedostajućem poimanju. Biblioteka pogrešaka (engl. bug library) je model učenika koji osim nedostajućeg poimanja sadrţi i pogrešno poimanje. Dijagnosticiranje nedostajućeg i pogrešnog poimanja se u ovom modelu realizira korištenjem pogrešaka unutar biblioteke. Tehnika stvaranja biblioteke pogrešaka

moţe biti temeljena na proučavanju literature, na detaljnoj analizi ponašanja učenika [BURT1982b] [VANL1982] i na predviĎanju pogrešaka koja uključuje ponovnu analizu i proširenje biblioteke pogrešaka [BROW1980] [VANL1983b]. Biblioteka dijelova pogrešaka (engl. bug part library) je još jedna tehnika kod koje se pogreška dinamički stvara od njenih dijelova kao što pokazuje ACM sustav [LANG1984].
Različite dijagnostičke tehnike se primjenjuju kod svih nivoa obuhvata ulaznih podataka. S obzirom na porijeklo, dijagnostičke tehnike se mogu podijeliti na kognitivne tehnike i na tehnike umjetne inteligencije [WOOL2008]. Neke od češćih kognitivnih tehnika su praćenje modela [ANDE1986] i model zasnovan na ograničenjima (engl. constraint-based model). Praćenje modela učenika se temelji na mogućnosti modeliranja cjelokupnog procesa učenja, za razliku od modela zasnovanog na ograničenjima koji prati samo pogreške. Tehnike umjetne inteligencije se mogu klasificirati na analitičke i heurističke tehnike. Kod analitičkih tehnika se formalnim pristupom dijagnosticira model učenika, čime se ograničava razumijevanje učenika koji učenju skoro uvijek pristupa neformalno. MeĎutim, neodreĎenost kod heurističkih tehnika je pogodna u slučaju kada je potrebno iz skupa dijagnostičkih predikcija izdvojiti one koje će se vjerojatno dogoditi.

Novi model inteligentnog tutorskog sustava analitičkom tehnikom dijagnosticira model učenika koji je ostvaren modelom prekrivanja. Prekrivanjem se opisuje odnos znanja stručnjaka i znanja učenika tako što se koncepti područnog znanja označavaju kao nedostajući. Koncepti se takoĎer označavaju teţinskom vrijednošću koja ukazuje na vaţnost koncepta. Pošto se učenje nastavnog sadrţaja temelji na mreţnom učenju [NORM1973], onda opći koncepti imaju veću vaţnost od specifičnih, a time i teţinsku vrijednost.
Model učenika se oblikuje i dijagnosticira za vrijeme tutorskog dijaloga. Na početku tutorskog dijaloga svi koncepti koji čine objekt nastavnog sadrţaja namijenjen testiranju su označeni kao nedostajući. Pitanje u tutorskom dijalogu se formira nad odreĎenim konceptom. Na primjer, neka se pitanje formira nad konceptom K_1 kao što je prikazano na slici 2.8.



koncept učenikovog odgovora	oznaka koncepta pitanja
K_2	nije nedostajući
K_3	nedostajući
K_4	djelomično nedostajući
K_5	djelomično nedostajući
K_6	nedostajući

Slika 2.8. Model prekrivanja kod tutorskog dijaloga

Neka je K_2 koncept traţenog odgovora na pitanje koji je direktno povezan s konceptom pitanja. Točnim odgovorom na postavljano pitanje se s koncepta skida oznaka da je nedostajući. Odgovori učenika mogu biti djelomično točni, pa se koncept označava kao djelomično nedostajući. U tom slučaju analizom povezanosti koncepta učenikovog odgovora i koncepta traţenog odgovora se generira slijed pitanja i objašnjenja koji će navest učenika na točan odgovor. Na primjer, ako učenik odgovori konceptom K_4, tada će K_4 biti koncept sljedećeg pitanja, a K_2 koncept traţenog odgovora tog pitanja.
Napredovanje kroz nastavni sadrţaj ovisi o ocjeni tutorskog dijaloga. Formiranje ocjene se temelji na statusu koncepata i na njihovoj teţinskoj vrijednosti. Što je manje nedostajućih koncepata to je ocjena viša te je veća mogućnost prelaska na sljedeći objekt nastavnog sadrţaja. Ako ocjena tutorskog dijaloga nije zadovoljavajuća, tada učenik ponavlja učenje nad istim objektom nastavnog sadrţaja.

2.4 Komunikacijski modul

Zadatak komunikacijskog modula je ostvarivanje konverzacije između učenika i modula učitelja. U osnovi, komunikacija se sastoji od primanja ulaznih podataka od učenika i prikazivanja izlaznih podataka učeniku. Idealna komunikacija između učenika i računalnog tutora nastoji biti što sličnija komunikaciji između učenika i ljudskog tutora. Po navedenom bi komunikacijski modul trebao biti u stanju obrađivati prirodni jezik i biti u stanju reagirati na socijalno ponašanje učenika. Kada učenik komunicira s računalom, on često interpretira socijalnu povezanost s računalom koja uključuje recipročnu komunikaciju [REEV1998]. Tehnologije, kao što su dijalog na prirodnom jeziku i pedagoški agenti, su u stanju ostvariti ovakve vidove komunikacije.
Neke od tehnika komunikacije u inteligentnim tutorskim sustavima su [WOOL2008] grafička komunikacija, socijalna inteligencija, komponentna sučelja i obrada prirodnog jezika.
Pedagoški agenti, sintetski ljudi i prividna stvarnost su neke od grafičkih tehnologija komunikacije. Animirani pedagoški agenti su grafički likovi koji motiviraju učenika na interakciju postavljanjem pitanja, pruţanjem podrške i davanjem povratnih informacija [SLAT2000]. Oni su autonomni sustavi koji koriste inteligenciju za primanje ulaznih podataka iz okoline, obrađuju ulazne podatke i zatim utječu na okolinu. Za razliku od animiranih pedagoških agenata, sintetski ljudi se grafički prikazuju kao realistični ljudski karakteri. Sintetski ljudi se najčešće koriste kod učenja vještina uz timski rad. Pedagoški agenti su uključeni i u grafičkim svjetovima prividne stvarnosti [RICK1997] [RICK1998]. Interakcija u prividnoj stvarnosti se ostvaruje pokretima i poloţajima ruke, glave, pa i cijelog tijela čovjeka koji rezultiraju promjenama u prividnom svijetu.
Socijalna inteligencija je strategija komunikacije temeljena na uspostavljanju emocionalne i socijalne veze s učenikom. Animirani pedagoški agenti su u stanju izraziti emocionalno stanje i potaknuti socijalni kontakt s učenikom. S druge strane, jedan od pristupa razumijevanju emocija učenika je analiza mjerljivih ponašajnih podataka, kao što su trajanje rješavanja problema, pogreške i zahtjevi za pomoć. Ostali pristupi se temelje na nestandardnim ulaznim podacima kao što su vizualno prepoznavanje emocija, metabolički indikatori i zastoji u govoru.
Jedinstvena vrsta sučelja koja zadovoljava specijalne komunikacijske potrebe su komponentna sučelja. Ova sučelja obrađuju specijalizirane ulazne podatke, kao što su formule, jednadţbe i vektori.
Grafička komunikacija, socijalna inteligencija i komponentna sučelja su korisne komunikacijske tehnike koje potpomaţu usvajanje znanja. One se često kombiniraju kako bi se ostvarile posebne potrebe kod procesa učenja, poučavanja i testiranja znanja učenika. Na primjer, pedagoški agenti realiziraju neverbalnu komunikaciju, za koju se pokazalo kako znatno utječe na dijalog prirodnim jezikom, što je posljedica utjecaja neverbalne komunikacije u dijalogu kod ljudi [DEUT1962]. Međutim, komunikacija prirodnim jezikom kod inteligentnih tutorskih sustava je definitivno osnovni oblik komunikacije koji, zbog svoje intuitivnosti, ne zahtjeva prethodno uvjeţbavanje.
Osnovne metode i tehnike obrade prirodnog jezika kod inteligentnih tutorskih sustava su navedene u sljedećem odjeljku, a zatim slijedi odjeljak u kojem se opisuje struktura i obrada kontroliranog jezika kao podskupa prirodnog jezika kojim se ostvaruje komunikacija između učenika i novog modela inteligentnog tutorskog sustava.

2.4.1 Obrada prirodnog jezika i dijalog u komunikacijskom modulu

Dvije su osnovne tehnike obrade prirodnog jezika kojima se implementira komunikacijski modul temeljen na prirodnom jeziku u inteligentnim tutorskim sustavima. Generiranje prirodnog jezika je proces koji nelingvističke ulazne podatke transformira u prirodan jezik, dok je razumijevanje prirodnog jezika obrnut proces. Kod ovih tehnika postoje tri razine obrade [JURM2000]:

1. Sintaksa – proučava rečenice i ustroj rečenice,
2. Semantika – se bavi proučavanjem značenja jezika, odnosno proučavanjem odnosa jezičnih izraza i stvarnosti. Najčešći pristup je slijedeći: smisao rečenice potpuno je određen smislom njenih elemenata,
3. Pragmatika – proučava konkretnu upotrebu jezika između subjekata u komuniciranju. Kod prirodnih jezika sintaksa i semantika nisu dovoljne za potpuno razumijevanje jezika. Kod umjetnih, posebice programskih jezika, pragmatika ne postoji.

Navedene razine obrađuju tekstualni oblik prirodnog jezika: sintaksa određuje ustroj teksta, semantika značenje, dok pragmatika razmatra ulogu teksta u konverzaciji. S obzirom na izvor, prirodni jezik može biti pisan ili govoren. Zbog toga se, u obradi prirodnog jezika, uključuju morfološka i fonološka razina. Ove dvije razine se bave riječima i glasovima. Morfologija pruža informacije o različitim oblicima riječi koje se mogu dobiti ili pretraživanjem leksikona ili generiranjem oblika riječi. Morfološki leksikon je izvor svih oblika riječi nekog prirodnog jezika koji služi i za pretraživanje i za generiranje. Fonološka razina obrađuje izgovor određenog glasa ili grupe glasova. Kod govorenog prirodnog jezika se, u fonološkoj razini, koriste tehnike automatskog prepoznavanja govora i sinteza govora iz teksta.
Riječ i glas su gramatičke jedinice koje dobivaju značenje tek kada se nađu u nekoj široj strukturnoj gramatičkoj jedinici. Osnovna strukturna jedinica u gramatičkoj organizaciji jezika kojom se izriče obavijest je rečenica. Sintaksa, semantika i pragmatika utječu na valjanost rečenica koje se ili generiraju ili razumijevaju. Pri tome sintaksa određuje način strukturiranja riječi u veće gramatičke jedinice, semantika daje značenje grupama riječi, a pragmatika namjeru grupe riječi. Dijalog nastavlja razinu pragmatike obradom razmjene grupe riječi među subjektima.
Tutorski dijalog intenzivno primjenjuje tehnike obrade prirodnog jezika. Komunikacija učenika s inteligentnim tutorskim sustavom zahtjeva generiranje i razumijevanje prirodnog jezika, bilo da se radi o pisanom ili govorenom prirodnom jeziku. Postoje četiri kategorije inteligentnih tutorskih sustava s obzirom na prilagodljivost i mogućnost dijaloga [WOOL2008]:

- Dijalog miješane inicijative (engl. mixed-initiative dialogue) u kojemu tutor ili učenik započinje i usmjerava dijalog. U ovom dijalogu učenik slobodno diskutira o nevezanim temama i inicira zahtjeve koji ne moraju zavisiti o područnom znanju,
- Dijalog jednostrane inicijative (engl. single-initiative dialogue) u kojemu tutor ima inicijativu, ali uvažava izjave učenika. Međutim ovaj dijalog ima slabu mogućnost konverzacije zbog ograničene domene, a učenikov odziv je sveden na kratke odgovore,
- Usmjereni dijalog (engl. directed dialogue) je potpuno vođen od strane tutora koji od učenika traži eksplicitne odgovore. Tutor razumije kratke odgovore i u stanju je generirati objašnjenja.
- "Profinjeni" dijalog (engl. finessed dialogue) realizira dijalog tekstualnim metodama, izbornicima, formalnom logikom i slično. Pri tome se prirodni jezik ne koristi direktno, nego se oblikuje navedenim metodama.

Metode kojima je podrţana obrada prirodnog jezika kod dijaloga mogu biti analitičke te heurističke ili hibridne. *Analitičke metode* se temelje na znanju, jer semantika izjave na prirodnom jeziku ima čvrsto uporište u semantici prikaza područnog znanja. Ovim načinom se postiţe dublji nivo znanja od heurističkih metoda [ROSÉ2000]. Analitički pristup je, od samih početaka inteligentnih tutorskih sustava, bio prihvaćen i većinski zastupljen, meĎutim heurističke metode inkrementalno postaju dominantne. *Heuristički*, odnosno statistički pristup je prvenstveno namijenjen razumijevanju prirodnog jezika. Ovaj pristup koristi veliku bazu teksta na prirodnom jeziku, odnosno korpus, koji se raznim statističkim alatima analizira. Pokazuje se kako statističke tehnike temeljene na korpusu premašuju performanse sustava temeljenih na znanju [CHAR1996]. Razlog popularnosti statističkih metoda je njihova pristupačnost, brzina i preciznost. Statističke metode mogu analizirati sintaksu rečenica i pronaći fraze, otkriti ponavljanje riječi u rečenicama i razjasniti dvosmislenost riječi u rečenicama. Statističke metode posjeduju svojstvo učenja jer, svakim novim ulaznim tekstom na prirodnom jeziku, statistički alati mogu profiniti svoje procedure. Ipak, analitičkom metodom se, u većini inteligentnih tutorskih sustava, uspostavlja čvrsta veza sa strogo formaliziranim područnim znanjem, čime se u potpunosti izbjegava dvosmislenost izjava na prirodnom jeziku, što rezultira jasnim i preciznim izjavama.

2.4.2 Struktura kontroliranog jezika

Novi model inteligentnog tutorskog sustava koristi analitičke metode za generiranje i prepoznavanje teksta. Analitička metoda se, u novom modelu, zasniva na preslikavanju formalnog prikaza znanja u tekstualne izjave kontroliranog jezika. Kontrolirani jezik, za razliku od prirodnog jezika, posjeduju ograničenu terminologiju, sintaksu i semantiku. Pragmatičnost se oslikava kod upotrebe kontroliranog jezika tijekom dijaloga kojim se realizira testiranje znanja učenika. Kontroliranim jezikom se, izmeĎu ostalog, pojačava jasnoća, upotrebljivost, prenosivost, pretraţivanje i saţimanje teksta. To se postiţe terminološkom konzistentnošću, pojednostavljenom strukturom rečenica i standardizacijom formata dokumenta. Terminologija kontroliranog jezika je specifična za neko područje primjene. Na primjer, "struja" u domeni elektronike označava tijek električnog naboja, dok u oceanografiji predstavlja gibanje morske vode.
Riječi koje definiraju terminologiju kontroliranog jezika novog modela inteligentnog tutorskog sustava su zapisane u ontološkom opisu područnog znanja. Svaki koncept i svaka relacija područnog znanja ima naziv koji se sastoji od jedne ili više riječi. Naziv koncepta i relacije je opisan frazom, a fraza je niz riječi koji predstavlja sintaktičku jedinicu rečenice.

2.4.2.1 Riječ

Riječ je najmanja samostalna jezična jedinica koja ima značenje. Kod kontroliranog jezika riječ je sintaktička jedinica koja se ne moţe dijeliti. Kontrolirani jezik novog modela inteligentnog tutorskog sustava je podskup hrvatskog standardnog jezika, stoga se morfološki leksikon sastoji od riječi hrvatskog jezika. Riječi hrvatskog standardnog jezika se dijele na promjenljive i na nepromjenljive, s obzirom na njihovo pojavljivanje u različitim oblicima. Morfološki oblik promjenljivih riječi ovisi o nekim gramatičkim osobinama riječi. Imenice su promjenljive vrste riječi koje imaju rod, broj i padeţ, a morfološki oblik imenice ovisi samo o broju i padeţu kao što se vidi u primjeru iz tablice 2.2.

Tablica 2.2. Svi morfološki oblici imenice "memorija"

oblik	rod	broj	padeţ
memorija	ţenski	jednina	nominativ
memorije	ţenski	jednina	genitiv
memoriji	ţenski	jednina	dativ
memorije	ţenski	jednina	akuzativ
memorijo	ţenski	jednina	vokativ
memoriji	ţenski	jednina	lokativ
memorijom	ţenski	jednina	instrumental
memorije	ţenski	mnoţina	nominativ
memorija	ţenski	mnoţina	genitiv
memorijama	ţenski	mnoţina	dativ
memorije	ţenski	mnoţina	akuzativ
memorije	ţenski	mnoţina	vokativ
memorijama	ţenski	mnoţina	lokativ
memorijama	ţenski	mnoţina	instrumental

Za razliku od imenica, pridjevi su takoĎer promjenljive vrste riječi čiji morfološki oblik, osim broja i padeţa, ovisi o rodu. Na morfološki oblik pridjeva utječe i stupanj komparacije. Primjer iz tablice 2.3 prikazuje neke oblike pridjeva "masovan".

Tablica 2.3. Neki morfološki oblici pridjeva "masovan"

oblik	stupanj	rod	broj	padeţ
masovan	pozitiv	muški	jednina	nominativ
masovnog	pozitiv	muški	jednina	genitiv
masovni	pozitiv	muški	mnoţina	nominativ
masovnih	pozitiv	muški	mnoţina	genitiv
masovna	pozitiv	ţenski	jednina	nominativ
masovne	pozitiv	ţenski	jednina	genitiv
masovne	pozitiv	ţenski	mnoţina	nominativ
masovnih	pozitiv	ţenski	mnoţina	genitiv
masovniji	komparativ	muški	jednina	nominativ
masovnijeg	komparativ	muški	jednina	genitiv
masovniji	komparativ	muški	mnoţina	nominativ
masovnijih	komparativ	muški	mnoţina	genitiv
masovnija	komparativ	ţenski	jednina	nominativ
masovnije	komparativ	ţenski	jednina	genitiv
masovnije	komparativ	ţenski	mnoţina	nominativ
masovnijih	komparativ	ţenski	mnoţina	genitiv

Od nepromjenljivih vrsta riječi jedino načinski prilozi nastali od pridjeva mogu imati različite morfološke oblike. Na primjer, "brzo", "brţe" i "najbrţe" su stupnjevi komparacije priloga "brzo". Tablica 2.4 prikazuje sve gramatičke osobine promjenljivih i nepromjenljivih vrsta riječi koje se koriste u kontroliranom jeziku novog modela inteligentnog tutorskog sustava.

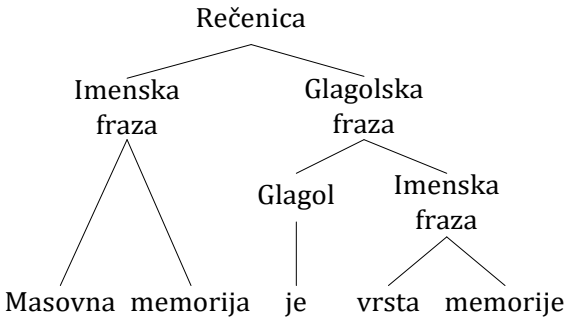
Tablica 2.4. Gramatičke osobine riječi koje utječu na morfološki oblik

vrsta riječi		rod	broj	padeţ	lice	stupanj	način
promjenljive	imenica		x	x			
	glagol	x	x		x		x
	pridjev	x	x	x		x	
	zamjenica	x	x	x	x		
	broj	x	x	x			
nepromjenljive	prilog					x	
	prijedlog						
	veznik						
	čestica						
	uzvik						

Pojedine vrste riječi, osim gramatičkih osobina, posjeduju i leksičke osobine koje ih dodatno kategoriziraju. Na primjer, s obzirom na značenje imenice se mogu podijeliti na opće i na vlastite, a s obzirom na čovjekov dodir dijele se na stvarne i nestvarne. Leksičke osobine ne utječu na morfološki oblik riječi, meĎutim leksička kategorizacija dodatno pojašnjava riječ.

2.4.2.2 Fraza

Fraza je niz riječi koji funkcionira kao jedinica u sintaksi rečenice. Većina fraza ima centralnu riječ koja odreĎuje vrstu fraze. Na primjer "masovna memorija" je imenska fraza jer joj je centralna riječ imenica "memorija", a pridjev "masovna" se slaţe u rodu i broju s centralnom imenicom. Teţnja formalnoj definiciji sintakse engleskog jezika opisuje gramatiku koja je u stanju izvesti rečenice kombiniranjem dvije vrste fraza: imenične i glagolske fraze [CHOM1956]. Predloţena generativna gramatika formalno opisuje rečenicu iz primjera na slici 2.9 generiranim stablom.

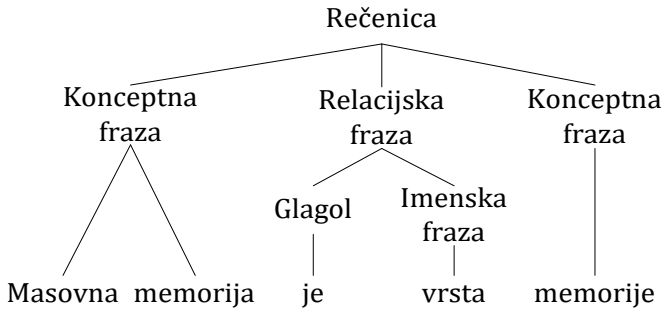


Slika 2.9. Generativna gramatika

U generativnoj gramatici fraze se mogu hijerarhijski umetati, kao što je imenska fraza postavljena iza glagola u glagolskoj frazi iz primjera sa slike 2.9.
Kontrolirani jezik novog modela inteligentnog tutorskog sustava koristi fraze koje su u skladu s nazivima koncepata i relacija područnog znanja. S obzirom na usku povezanost s područnim znanjem, definirane su dvije vrste fraza kontroliranog jezika:

- Konceptna fraza – predstavlja naziv koncepta područnog znanja. Produkcijsko pravilo konceptne fraze je slično produkcijskom pravilu imenske fraze.
- Relacijska fraza – imenuje relaciju područnog znanja i moţe uključivati druge konceptne fraze.

Najjednostavnija rečenica kontroliranog jezika se sastoji od dvije konceptne fraze i jedne relacijske fraze, što odgovara strukturi područnog znanja u kojemu relacija uvijek povezuje dva koncepta. Primjer stabla rečenice sa slike 2.10 opisuje jednostavnu rečenicu kontroliranog jezika koja je nastala iz područnog znanja u kojemu postoje koncepti s nazivima "masovna memorija" i "memorija" povezani relacijom s nazivom "je vrsta".



Slika 2.10. Gramatika rečenice kontroliranog jezika

Kontrolirani jezik ograničava sintaksu rečenice jer se temelji na formalizmu područnog znanja. Ograničenost kontroliranog jezika se očituje i u njegovim frazama koje sliče frazama generativne gramatike prirodnog jezika. Rečenica kontroliranog jezika je niz fraza, a riječi unutar fraze moraju biti pravilnog morfološkog oblika kako bi rečenica bila sintaktički ispravna. Zbog toga se definira oblik fraze kao posljedica morfološkog oblika riječi koje je čine. Gramatičke osobine fraze najčešće odgovaraju gramatičkoj osobini centralne riječi te fraze. Na primjer, imenska fraza "masovna memorija" ima rod, broj i padeţ koji odgovara rodu, broju i padeţu njene centralne riječi. Oblici ostalih riječi fraze se usklaĐuju upravo s centralnom riječi, pa je rod, broj i padeţ pridjeva "masovan" jednak gramatičkim osobinama imenice "memorija" kao što se vidi u tablici 2.5.

Tablica 2.5. Svi oblici imenske fraze "masovna memorija"

oblik	**rod**	**broj**	**padeţ**
masovna memorija	ţenski	jednina	nominativ
masovne memorije	ţenski	jednina	genitiv
masovnoj memoriji	ţenski	jednina	dativ
masovne memorije	ţenski	jednina	akuzativ
masovna memorijo	ţenski	jednina	vokativ
masovnoj memoriji	ţenski	jednina	lokativ
masovnom memorijom	ţenski	jednina	instrumental
masovne memorije	ţenski	mnoţina	nominativ
masovnih memorija	ţenski	mnoţina	genitiv
masovnim memorijama	ţenski	mnoţina	dativ
masovne memorije	ţenski	mnoţina	akuzativ
masovna memorije	ţenski	mnoţina	vokativ
masovnim memorijama	ţenski	mnoţina	lokativ
masovnim memorijama	ţenski	mnoţina	instrumental

MeĐutim, kod imenskih fraza, rod centralne riječi je nepromjenjiv i ne utječe na oblik fraze, ali je bitan podatak koji se koristi kod odreĐivanja oblika ostalih fraza u rečenici. Niz riječi relacijske fraze uvijek posjeduje glagol. Gramatičke osobine centralnog glagola su ponekad samo dio gramatičkih osobina fraze. Neke gramatičke osobine relacijske fraze imaju predefinirane vrijednosti kao što, na primjer, fraza "je vrsta" je uvijek u genitivu. Tablica 2.6 prikazuje sve oblike relacijske fraze "je vrsta".

Tablica 2.6. Svi oblici relacijske fraze "je vrsta"

oblik	broj	padeţ	negativan
je vrsta	jednina	genitiv	ne
nije vrsta	jednina	genitiv	da
su vrste	mnoţina	genitiv	ne
nisu vrste	mnoţina	genitiv	da

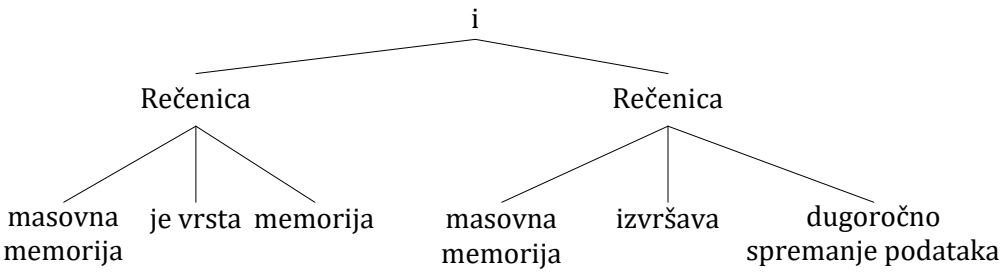
Relacijska fraza moţe imati negativan oblik koji ovisi o centralnom glagolu. Padeţ relacijske fraze utječe na padeţ konceptne fraze koja se nalazi poslije relacijske fraze u rečeničnom nizu. Zbog toga padeţ konceptne fraze "memorija" iz primjera sa slike 2.10 odgovara padeţu relacijske fraze.

2.4.2.3 Rečenica

Rečenica kontroliranog jezika je sintaktička jedinica koja nastaje nizanjem fraza. Neke fraze kontroliranog jezika ovise o nazivima koncepata i relacija područnog znanja, dok su druge fraze predefinirane i sluţe za povezivanje koncepata i rečenica veznicima ili zamjenjuju konceptne fraze zamjenicama kod upitnih rečenica. U primjeru (2.3) je dana izjavna rečenica u kojoj je korišten veznik konjunkcije.

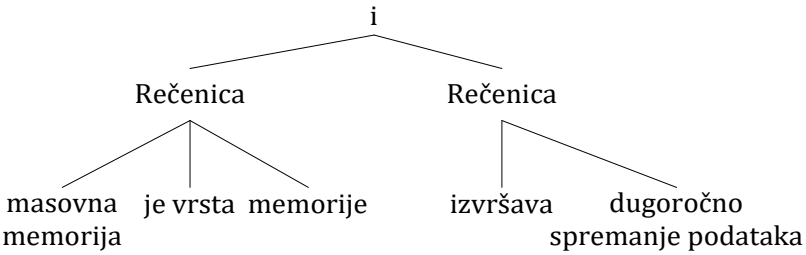
Masovna memorija je vrsta memorije i izvršava dugoročno spremanje podataka. (2.3)

Ova rečenica je nastala na osnovu područnog znanja gdje je koncept masovna memorija povezan relacijom "je vrsta" s konceptom "memorija" i povezan je relacijom "izvršava" s "dugoročno spremanje podataka". Stablo kontroliranog jezika koja opisuje izjavnu rečenicu iz primjera (2.3) je dana na slici 2.11.



Slika 2.11. Primjer stabla rečenice kontroliranog jezika

U ovom primjeru se koristi predefinirana fraza konjunkcije koja povezuje dvije jednostavne rečenice. Svaka jednostavna rečenica posjeduje tri grane koje, čitajući s lijeva prema desno, odgovaraju subjektu, predikatu i objektu rečenice. Primjer (2.3) posjeduje dvije konjugirane jednostavne rečenice gdje obje rečenice imaju istu frazu u subjektnoj grani. Radi izbjegavanja ponavljanja fraza, u drugoj rečenici će se izbaciti subjektna fraza, i zavisnim odreĎivanjem oblika fraza se dobiva stablo sa slike 2.12.

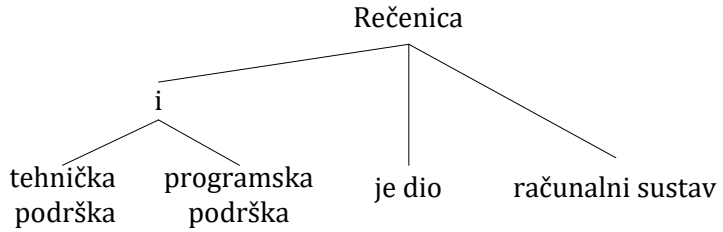


Slika 2.12. Brisanje ponavljanja konceptnih fraza u rečenici

U gramatici rečenica kontroliranog jezika koriste se sljedeća pravila:

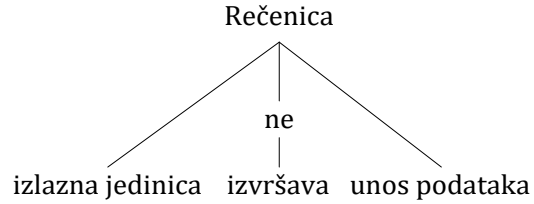
- povezivanje jednostavne rečenice veznicima konjunkcije i disjunkcije,
- povezivanje konceptne fraze veznicima konjunkcije i disjunkcije,
- negiranje relacijske fraze,
- odreĎivanje broja konceptne fraze.

Primjer sa slike 2.12 pokazuje konjunkciju dviju jednostavnih rečenica. Konceptne fraze u subjektnoj ili objektnoj grani rečenice takoĎer mogu biti vezane konjunkcijom ili disjunkcijom kao što je prikazano na slici 2.13.



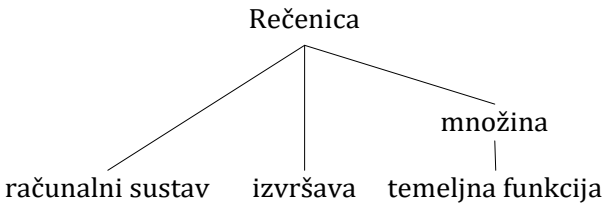
Slika 2.13. Konjunkcija konceptnih fraza u stablu rečenice

Veznik "i" se upotrebljava kod konjunkcije, a veznik "ili" kod disjunkcije konceptnih fraza. Ako veznik povezuje n fraza, gdje je n > 2, tada se n-1 prvih fraza odvajaju zarezom, a izmeĎu n-1 i n-te fraze stoji riječ veznika. Konjunkcija konceptnih fraza se moţe pojaviti u subjektnoj ili objektnoj grani rečenice, dok se negacija moţe pojaviti samo u predikatnoj grani rečenice ispred relacijske fraze kao što je prikazano na slici 2.14.



Slika 2.14. Negiranje relacijske fraze u stablu rečenice

Pod odreĎivanjem broja konceptne fraze se smatra postavljanje oblika konceptne fraze u jedninu ili mnoţinu kao što je prikazano na slici 2.15.



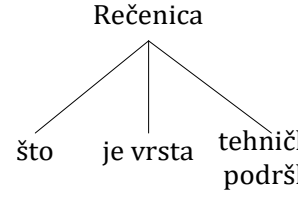
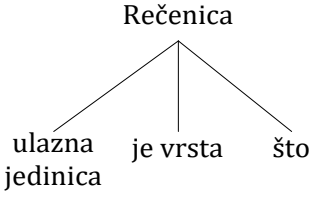
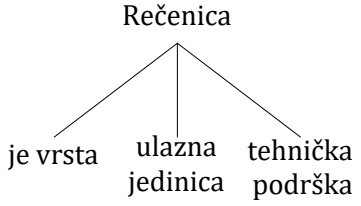
Slika 2.15. Broj konceptne fraze u stablu rečenice

Osim izjavnih rečenica, kontrolirani jezik sadrţi i upitne rečenice. Upitna rečenica kontroliranog jezika nastaje iz jednostavne rečenice koja ima subjektnu, predikatnu i objektnu granu. Na osnovu zamjene ili premještanja grane jednostavne rečenice, definiraju se tri vrste upitnih rečenica:

- upitna rečenica sa zamijenjenim subjektom,
- upitna rečenica sa zamijenjenim objektom i
- upitna rečenica s premještenim predikatom.

Upitna rečenica sa zamijenjenim subjektom ili objektom nastaje tako što se subjektna grana ili objektna grana zamjeni neodreĎenom zamjenicom "tko" ili "što". Ako se zamjeni subjektna grana tada je padeţ neodreĎene zamjenice nominativ. Kod zamjene objektne grane, padeţ neodreĎene zamjenice ovisi o padeţu relacijske fraze. Upitna rečenica s premještenim predikatom, po pravilu, nastaje tako što se predikatna grana postavi ispred subjektne grane jednostavne rečenice. Primjeri upitnih rečenica nastalih zamjenom subjektne, objektne i predikatne grane jednostavne rečenice "Ulazna jedinica je vrsta tehničke podrške" su dani u tablici 2.7.

Tablica 2.7. Upitne rečenice kontroliranog jezika

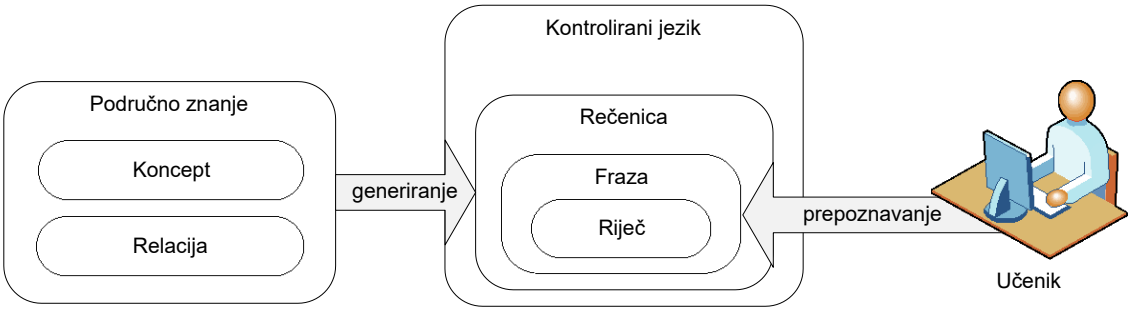
upitna rečenica sa zamijenjenim subjektom	**upitna rečenica sa zamijenjenim objektom**	**upitna rečenica s premještenim predikatom**
		
Što je vrsta tehničke podrške?	Ulazna jedinica je vrsta čega?	Je li ulazna jedinica vrsta tehničke podrške?

Primjer upitne rečenice s premještenim predikatom iz tablice 2.7 uključuje dodatnu obradu relacijske fraze. Centralna riječ relacijske fraze ostaje u predikatnoj grani dok se ostale riječi relacijske fraze prebacuju izmeĎu dviju konceptnih fraza. Ako je centralna riječ vezni glagol, onda se iza nje stavlja upitna čestica "li".

Razina izraţajnosti područnog znanja odreĎuje granice gramatike kontroliranog jezika, a nazivi koncepata i relacija ograničavaju rječnik kontroliranog jezika. Rječnik kontroliranog jezika sadrţi riječi koje se nalaze u frazama područnog znanja i riječi predefiniranih fraza kao što su fraze s neodreĎenim zamjenicama koje sluţe za formiranje upitnih rečenica i fraze čestica "da" i "ne" koje predstavljaju odgovore učenika. Kod obrade kontroliranog jezika rječnikom se ograničava dozvoljena terminologija, a sama obrada je ograničena gramatikom kojom je definirana struktura kontroliranog jezika.

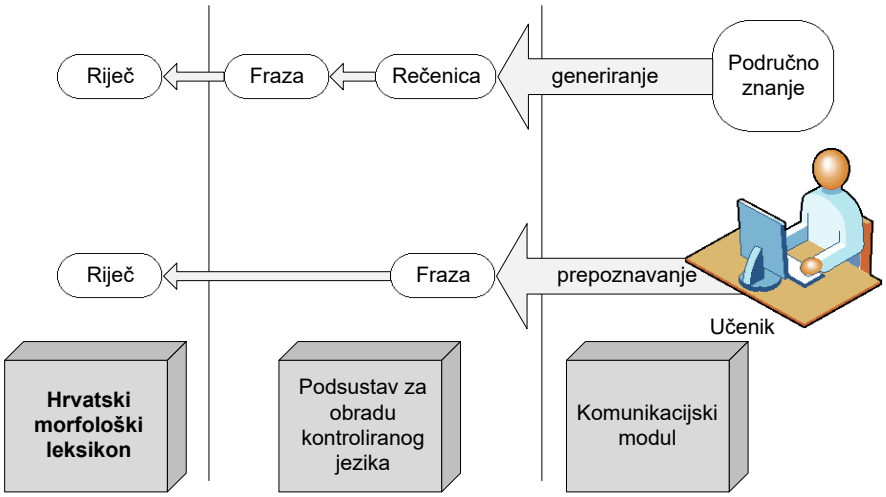
2.4.3 Obrada kontroliranog jezika u komunikacijskom modulu

Obrada kontroliranog jezika je funkcija komunikacijskog modula novog modela inteligentnog tutorskog sustava kojom se područno znanje namijenjeno učenju transformira u rečenice kontroliranog jezika, a prilikom testiranja tutorskim dijalogom se generiraju upiti, objašnjenja i procjene učenikovih odgovora. TakoĎer obrada kontroliranog jezika sluţi za prepoznavanje učenikovih odgovora.
Pod obradom kontroliranog jezika spadaju generiranje kontroliranog jezika i prepoznavanje kontroliranog jezika. Generiranje kontroliranog jezika je proces koji formalni prikaz područnog znanja transformira u rečenice kontroliranog jezika, dok se prepoznavanjem kontroliranog jezika prepoznaju učenikovi odgovori. U obradi kontroliranog jezika uključene su sve sintaktičke jedinice kontroliranog jezika, od riječi do rečenica, dok je prepoznavanje kontroliranog jezika definirano na nivou riječi i fraza kao što je prikazano na slici 2.16.



Slika 2.16. Nivoi obrade kontroliranog jezika

Komunikacijski modul novog modela inteligentnog tutorskog sustava koristi nezavisnu komponentu za obradu kontroliranog jezika. Ova komponenta vrši obradu kontroliranog jezika na nivou rečenica i fraza, dok za obradu riječi upošljava morfološki leksikon hrvatskog standardnog jezika. Slika 2.17 prikazuje uloge hrvatskog morfološkog leksikona, podsustava za obradu kontroliranog jezika i komunikacijskog modula tijekom obrade kontroliranog jezika.



Slika 2.17. Uloga komponenti kod obrade kontroliranog jezika

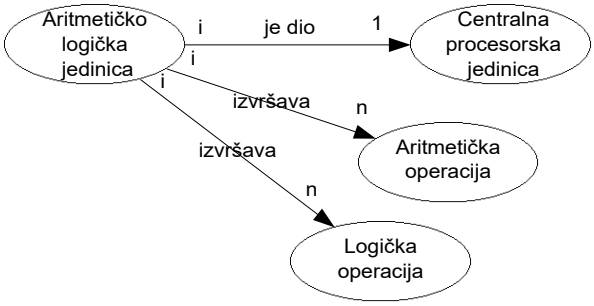
Podsustav za obradu kontroliranog jezika obraĎuje kontrolirani jezik koristeći sintaksu rečenica i fraza, dok je morfološki rječnik zaduţen za morfologiju riječi. Proces generiranja i prepoznavanja kontroliranog jezika se dijeli u nekoliko faza. Pojedina faza se bavi obradom

kontroliranog jezika na pojedinim nivoima njegovih sintaktičkih jedinica. Pri tome se ne prelaze granice sintakse kontroliranog jezika, a rječnik ne sadrţi terminologiju koja nije vezana za područno znanje.

2.4.3.1 Generiranje kontroliranog jezika

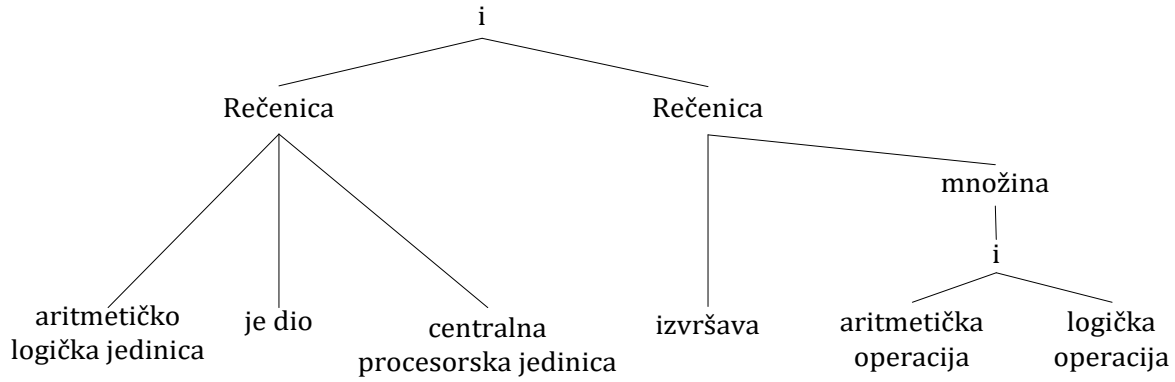
Područno znanje je ulazni skup podataka za proces generiranja kontroliranog jezika. Izlazni skup podataka su rečenice kontroliranog jezika. Faze generiranja kontroliranog jezika novog modela inteligentnog tutorskog sustava su slične fazama koje se odvijaju u modulima sustava za generiranje prirodnog jezika [REIT2000]. Na primjeru područnog znanja sa slike 2.18 će se pokazati proces generiranja kontroliranog jezika koji prolazi kroz sljedeće faze:

1. Analiza strukture područnog znanja i generiranje stabla rečenica.
2. OdreĎivanje oblika fraza u rečenicama.
3. Generiranje teksta iz stabla rečenica.



Slika 2.18. Primjer područnog znanja

U prvoj fazi se analizira struktura područnog znanja radi stvaranja rečenica kontroliranog jezika. Prvi korak stvaranja rečenice generira skup jednostavnih rečenica gdje svaka jednostavna rečenica predstavlja dva koncepta povezana relacijom. Pri tome se razmatra mnogostrukost relacije kako bi se odredio broj konceptne fraze u objektnoj grani. U drugom koraku se jednostavne rečenice povezuju veznicima koji su odreĎeni logičkim operatorima relacija. Optimizacija stabla rečenica je treći korak koji uključuje izbacivanje ponavljajućih fraza. Stablo rečenica sa slike 2.19 prikazuje rezultat prve faze kod generiranja prirodnog jezika.



Slika 2.19. Stablo rečenica u prvoj fazi generiranja kontroliranog jezika

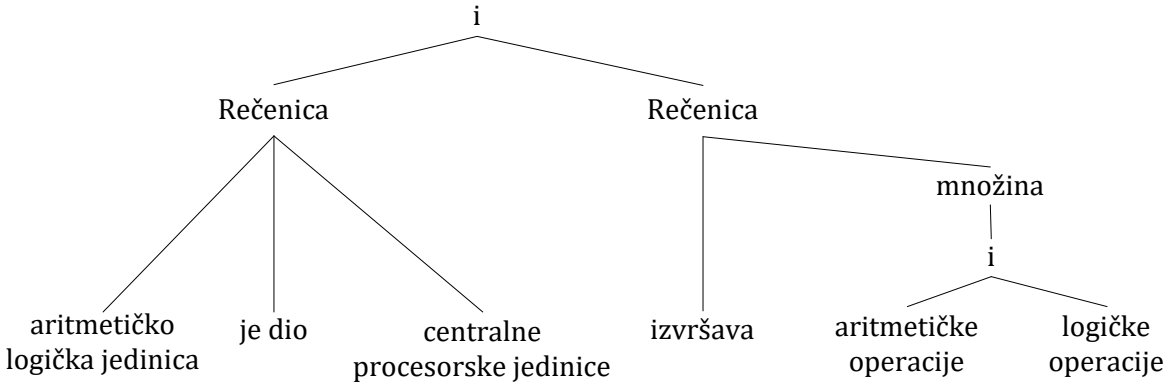
U drugoj fazi se prikupljaju fraze, koje predstavljaju nazive koncepata i relacija područnog znanja, radi odreĎivanja njihovog oblika. Prvi korak ove faze koristi rječnik svih oblika riječi

iz konceptnih i relacijskih fraza područnog znanja kako bi se dobio osnovni oblik fraze. U drugom koraku se analizom strukture stabla rečenica određuje traţeni oblik fraza. Osnovni i traţeni oblici fraza su dani u tablici 2.8.

Tablica 2.8. Osnovni i traţeni oblici fraza

Osnovni oblik fraze		**Traţeni oblik fraze**	
fraza	**gramatička osobina**	**fraza**	**gramatička osobina**
aritmetičko logička jedinica	nominativ jednine	aritmetičko logička jedinica	nominativ jednine
centralna procesorska jedinica	nominativ jednine	centralne procesorske jedinice	genitiv jednine
aritmetička operacija	nominativ jednine	aritmetičke operacije	akuzativ mnoţine
logička operacija	nominativ jednine	logičke operacije	akuzativ mnoţine
je dio	genitiv jednine	je dio	genitiv jednine
izvršava	akuzativ jednine	izvršava	akuzativ mnoţine

Traţeni oblici fraza zamjenjuju osnovne oblike fraza u stablu rečenica kao što je prikazano na slici 2.20.



Slika 2.20. Stablo rečenica u drugoj fazi generiranja kontroliranog jezika

U trećoj fazi se generira krajnji tekst rečenice koji se prikazuje učeniku. Oblici konceptnih i relacijskih fraza su odreĎeni u prethodnoj fazi i oni se direktno prebacuju u tekst rečenice. Tekst za povezane jednostavne rečenice i za povezane konceptne fraze ovisi o broju povezanih jednostavnih rečenica, odnosno konceptnih fraza. Ako su povezane dvije jednostavne rečenice (ili konceptne fraze), onda se izmeĎu teksta tih dviju jednostavnih rečenica (ili konceptnih fraza) ubacuje tekst veznika "i" odnosno "ili". U slučaju da su povezane više od dvije jednostavne rečenice (konceptne fraze), onda se samo izmeĎu zadnja dva teksta jednostavnih rečenica (konceptnih fraza) ubacuje tekst veznika, a preostali tekstovi jednostavnih rečenica (konceptnih fraza) se odvajaju zarezom. Krajnji generirani tekst rečenice za područno znanje iz primjera sa slike 2.20 glasi

> Aritmetičko logička jedinica je dio centralne procesorske jedinice i izvršava aritmetičke operacije i logičke operacije. (2.4)

Primjer područnog znanja sa slike 2.18 predstavlja eksplicitno znanje. Iste faze generiranja kontroliranog jezika se primjenjuju i kod implicitnog znanja nastalog procesom zaključivanja. U tutorskom dijalogu generiranje sluţi i za generiranje upitnih rečenica, izjava o točnosti učenikovog odgovora i objašnjenja kojima se opisuje slijed prepoznavanja.

2.4.3.2 Prepoznavanje kontroliranog jezika

Učenikov odgovor na pitanje u tutorskom dijalogu je niz fraza, fraza ili riječ koja se mora razumjeti kako bi se ustvrdila točnost odgovora. Prepoznavanje kontroliranog jezika se vrši na nivou fraze i riječi. Proces prepoznavanja kontroliranog jezika uspoređuje učenikov tekst s pravilima za oblikovanje fraza kontroliranog jezika. Ako je rezultat usporedbe pozitivan, onda se smatra da je računalo razumjelo učenikov tekst.
Faze prepoznavanja kontroliranog jezika na nivou fraze su:

1. Rastavljanje fraze na riječi,
2. Određivanje oblika svake riječi u frazi,
3. Određivanje pravila koje povezuje riječi u frazi.

U prvoj fazi se tekst rastavlja na skupine znakova koje su međusobno odvojene separatorom. Separator između skupine znakova je znak razmaka " " čiji je ASCII kod 32. Na primjer fraza "centralna procesorska jedinica" se rastavlja na skupine znakova "centralna", "procesorska" i "jedinica".
Ove skupine znakova predstavljaju riječi koje ulaze u drugu fazu prepoznavanja kontroliranog jezika. Uz pomoć hrvatskog morfološkog leksikona se određuju oblici svih izoliranih skupina znakova koje predstavljaju riječi fraze. Tablica 2.9 prikazuje vrste riječi i njihove gramatičke osobine za frazu "centralna procesorska jedinica"

Tablica 2.9. primjer svih oblika riječi fraze "centralna procesorska jedinica"

centralna			**procesorska**			**jedinica**		
pridjev			**pridjev**			**imenica**		
rod	**broj**	**padež**	**rod**	**broj**	**padež**	**rod**	**broj**	**padež**
ženski	jednina	nominativ	ženski	jednina	nominativ	ženski	jednina	nominativ
ženski	jednina	vokativ	ženski	jednina	vokativ	ženski	množina	genitiv
muški	jednina	akuzativ	srednji	množina	akuzativ			
muški	jednina	genitiv	srednji	množina	nominativ			
srednji	množina	akuzativ	srednji	množina	vokativ			
srednji	množina	nominativ						
srednji	množina	vokativ						
srednji	jednina	genitiv						

U trećoj fazi se svaka kombinacija oblika riječi provlači kroz sva pravila za oblikovanje fraza. Ova pravila opisuju redoslijed riječi i njihovu povezanost. Prepoznato pravilo za primjer fraze "centralna procesorska jedinica" kaže da se ispred imenice mogu i ne moraju nalaziti pridjevi koji se s imenicom slažu po rodu, broju i padežu. Ovom pravilu odgovara oblici riječi iz tablice 2.9 koje su ženskog roda u nominativu jednine. Određivanjem pravila je fraza prepoznata te je proces prepoznavanja kontroliranog jezika završen.

Generiranje i prepoznavanje kontroliranog jezika su osnovna dva procesa obrade kontroliranog jezika. Generiranje kontroliranog jezika iz područnog znanja se koristi i kod učenja i kod testiranja znanja, dok prepoznavanje ima funkciju određivanja učenikovih odgovora na pitanja u tutorskom dijalogu. U sljedećem poglavlju se analizira primjena obrade prirodnog jezika u aktualnim inteligentnim tutorskim sustavima zasnovanim na obradi prirodnog jezika. Većina analiziranih sustava obradu prirodnog jezika koristi za ostvarivanje dijaloga. Pošto se u realizaciji dijaloga uključeni podaci iz nekih ili svih modula inteligentnog tutorskog sustava, onda se navodi i kratak opis strukture analiziranih sustava.

3 Stanje istraţenosti inteligentnih tutorskih sustava zasnovanih na obradi prirodnog jezika

U prethodnom poglavlju je iznesena zamisao novog modela inteligentnog tutorskog sustava koji komunikaciju s učenikom provodi na ograničenom prirodnom jeziku, odnosno kontroliranom jeziku. Na razvoj modela CoLaB Tutor-a utječu rezultati istraţivanja provedeni nad inteligentnim tutorskim sustavima zasnovani na obradi prirodnog jezika. MeĎu aktualnim ITS-ovima zasnovanim na prirodnom jeziku, posebno su analizirani sustavi AutoTutor, CIRCSIM-Tutor, Why2-Atlas i DIAG-NLP. CIRSCIM Tutor je odabran za analizu jer je prvi ITS nove generacije koji koristi ustaljene tehnike obrade prirodnog jezika. Na odabir Why2-Atlas-a je utjecala njegova sposobnost razumijevanja prirodnog jezika prvenstveno temeljena na znanju. DIAG-NLP je odabran jer pokazuje kako samo generiranje prirodnog jezika povećava učinkovitost sustava. Na kraju, odabir AutoTutor-a je očigledan jer je najpoznatiji ITS s podrškom za dijalog miješane inicijative. Navedeni sustavi obraĎuju engleski jezik, koji je sintaktički i morfološki jednostavniji od hrvatskog standardnog jezika. MeĎutim, načela obrade prirodnog jezika u ovom sustavima korištena su kao potpora pri razvoju modela CoLaB Tutor-a. Opisat će se struktura i tehnika obrade prirodnog jezika u sustavima AutoTutor, CIRCSIM-Tutor, Why2-Atlas i DIAG-NLP. Na kraju poglavlja su usporeĎene opće karakteristike navedenih sustava, kao što su područje poučavanja, primijenjene tutorske metode, modeliranje učenika i načini komunikacije s učenikom. Osim toga iznesene su karakteristike obrade prirodnog jezika aktualnih sustava i njihov odnos s modelom CoLaB Tutor-a.

3.1 AutoTutor

Razvoj sustava AutoTutor je započeo 1997. godine na University of Memphis. U razvoju su sudjelovali istraţivači iz različitih znanstvenih područja. Osim stručnjaka iz računalne znanosti, razvoju AutoTutor-a su doprinijeli psiholozi, kognitivni psiholozi, lingvisti i pedagozi ujedinjeni u Tutoring Research Group pod vodstvom Arthura C. Graessera. Kako bi se ugradili pojedini pedagoški ciljevi i obradila različita područja, oblikovane su različite verzije AutoTutor-a. U do sada obraĎena područja spadaju: računalna pismenost, fizika, biologija, taktičko planiranje i kritičko razmišljanje. U verziji AutoTutor-a koja poučava osnove računalne pismenost obraĎene su teme računalne tehničke podrške, operacijskih sustava i korištenja Interneta.
AutoTutor je sloţeni sustav koji simulira čovjeka ili idealnog tutora [GRAE1999] tako što vodi dijalog s učenikom na prirodnom jeziku [GRAE2004] [GRAE2001a] [GRAE2001b]. U dijalogu AutoTutor prezentira nizove pitanja ili problema na koje učenik odgovara ili ih rješava u tekstualnom obliku s dvije ili više rečenica. AutoTutor posjeduje animiranog agenta koji moţe verbalno i neverbalno komunicirati preko izraza lica i pokreta. Osim toga, nova verzija AutoTutor-a posjeduje i interaktivne simulacije. Primjer dijaloga izmeĎu AutoTutor-a i učenika je dan u tablici 3.1.

Tablica 3.1. Primjer AutoTutor-ovog dijaloga prevedenog na hrvatski jezik

Tutor	Da, točno! Moţeš li što dodati?
Učenik	Sila mora biti jednaka i suprotna reakciji koja djeluje na oba tijela.
Tutor	Da! Skoro smo gotovi, ali postoji još jedan dio koji nedostaje. Siguran sam da znaš. Za vrijeme interakcije, svako tijelo vrši jednaku, ali suprotnu silu na drugo _____ ?
Učenik	tijelo
Tutor	Da, tako je! Na drugo tijelo. Dakle, kad se dvije kuglice, jedna plastična, a druga ţeljezna, sudare, izvrši li teţa ili nijedna kuglica veći udar sile na drugu?
Učenik	nijedna
Tutor	Točno! Nijedna, sila je jednaka po veličini. Zastanimo na trenutak za općenito pitanje iz fizike. Postavi mi neko i ja ću se potruditi da ti odgovorim.
Učenik	Ne.
Tutor	U redu. Da ponovimo o čemu smo upravo pričali, molim te daj mi svoj potpuni odgovor na postavljeno pitanje koje vidiš na vrhu ekrana.
Učenik	Iako automobil zabiljeţi veću promjenu u kretanju, oba vozila zabiljeţe jednaki udar sile, ali različitog smjera. Kamion ima veću masu i zato pruţa veći otpor nego automobil.
Tutor	Tako je! Dakle, prema Newton-ovom trećem zakonu kretanja, ako jedno tijelo djeluje silom na drugo tijelo onda drugo tijelo mora djelovati jednakom, ali suprotnom silom na prvo tijelo. Zato automobil i kamion zabiljeţe jednaki udar sile. Prema Newton-ovom drugom zakonu kretanja, automobil će zabiljeţiti veću promjenu u svom kretanju jer mu je masa znatno manja. Ovo pitanje je gotovo i idemo na sljedeće.

Dijalog iz tablice 3.1 oslikava konverzaciju AutoTutor-a i sposobnijeg učenika koji zna ponešto o temi. Kod većine slučajeva učenici odgovaraju na pitanja s jednom ili dvije rečenice. U tom slučaju AutoTutor potiče učenika na dijalog miješane inicijative kako bi spoznao koliko učenik razumije temu te mu pomaţe u oblikovanju odgovora. Dijaloški potezi AutoTutor-a kojima se usmjerava interakcija su:

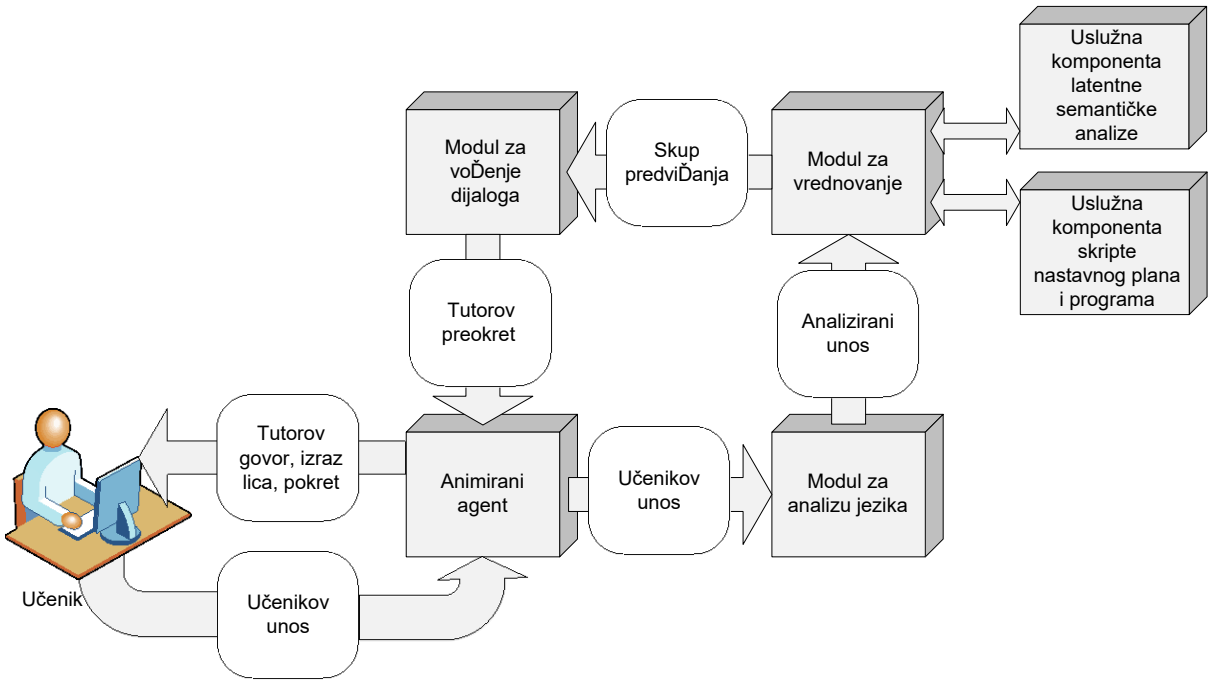
- *Povratna informacija* je dijaloški potez kojim se učeniku izlaţe kvaliteta njegovog odgovora. Postoje tri vrste povratne informacije: pozitivna, negativna i neutralna. Povratna informacija ne mora biti strogo verbalna, već se često kombinira s animiranim agentom koji izrazom lica podupire učenika.
- *Osiguravanjem potpunog* odgovora AutoTutor nastoji navesti učenika na dopunjavanje svog prvotnog odgovora. U poticaje osiguravanja potpunog odgovora spadaju poticajni izrazi, savjeti, pitanja nadopune, tvrdnje i saţetci potpunog odgovora pri samom kraju razgovora.
- *Ispravljanje pogrešaka* i zabluda se u ranijim verzijama AutoTutor-a svelo na puko ispravljanja pogrešnih informacija. Novije verzije nastoje učeniku postavljati pomno odabrana pitanja koja će ga navesti na shvaćanje vlastitih zabluda.
- *Odgovaranjem na učenikova pitanja* AutoTutor daje odziv na učenikovo pitanje. Najčešće učenikovo pitanje je definicijskog tipa i tada AutoTutor iz rječnika iznosi definiciju traţenog pojma.

Kako bi AutoTutor simulirao idealnog tutora, dijalog se strukturira po razinama. Svaka razina je odgovorna za voĎenje dijaloga u pravom smjeru jer se učenikov unos stalno usporeĎuje s očekivanjima i zabludama te se pomaţe učeniku, a ponekad se inicijativa prenosi na učenika. Strukturne razine dijaloga AutoTutor-a su:

- *Dijalog usklaĎen s očekivanjima i zabludama* se temelji na predviĎenim dobrim odgovorima i zabludama. VoĎenje dijaloga je usmjereno potpunom odgovaranju na pitanja te se paţljivo odabranim dijaloškim potezima nastoji u potpunosti odgovoriti na pitanje.

- *Dijaloški okvir od pet koraka* je učestao kod ljudskih tutora, stoga je implementiran i u AutoTutor-u.
- *Upravljanje konverzacijskim preokretom* se sastoji od tri informacijska dijela. Kratka povratna informacija o kvaliteti učenikovog posljednjeg unosa je najčešće prvi dio. U drugom dijelu AutoTutor poboljšava pokrivenost očekivanih dobrih odgovora. Kod trećeg dijela se traţi od učenika preuzimanje inicijative i to pitanjem ili gestom.

Aktualna verzija AutoTutor-3D [GRAE2005] posjeduje interaktivne trodimenzionalne simulacije implementirane u zasebnom modulu. Temeljna struktura verzije AutoTutor-3D sustava je prikazana na slici 3.1 koja ujedno prikazuje i tijek obrade podataka.



Slika 3.1. Struktura i tijek obrade podataka sustava AutoTutor-3D

Komponente AutoTutor-3D sustava se dijele na module i usluţne komponente. Moduli primaju podatke i nakon obrade ih prosljeĎuju ostalim modulima. Usluţna komponenta prima podatke od modula i rezultat obrade vraća istom modulu. Obrada prirodnog jezika započinje učenikovim unosom koji se prenosi od animiranog agenta do modula za analizu jezika. Ovaj modul analizira sintaksu učenikovog unosa te se u sljedećem koraku rezultat obrade prosljeĎuje modulu za vrednovanje. Vrednovanjem se aţurira model učenika i nakon toga se predviĎaju budući dijaloški pokreti učenika. Modul za vrednovanje koristi latentnu semantičku analizu i skriptu nastavnog plana i programa za generiranje skupa predviĎanja. Dobiveni skup predviĎanja se u narednom koraku prenosi modulu za voĎenje dijaloga koji, s obzirom na prethodne dijaloške preokrete, generira novi preokret. U sljedećem koraku se tutorov preokret prenosi animiranom agentu koji ga interpretira kao govor, izraz lica ili pokret rukom.

Obrada prirodnog jezika se obavlja u više modula AutoTutor-a. Usluţna komponenta latentne semantičke analize je jedna od najčešće korištenih komponenti kod obrade prirodnog jezika u AutoTutor-u. Tehnikom latentne semantičke analize (LSA) se moţe procijeniti kvaliteta učenikovog odgovora i pratiti obuhvaćenost idealnog odgovora. Osim toga LSA sluţi za odreĎivanje odgovora na učenikovo pitanje i odreĎivanje parametara reakcije animiranog agenta [GRAE2000]. MeĎutim, LSA ne moţe uzeti u obzir redoslijed riječi, sintaksu, logičke

izraze, negacije, retoričke veze meĎu rečenicama i druge analitičke komponente razumijevanja. Zbog toga u AutoTutor-u postoje i drugi moduli koji se bave obradom prirodnog jezika.

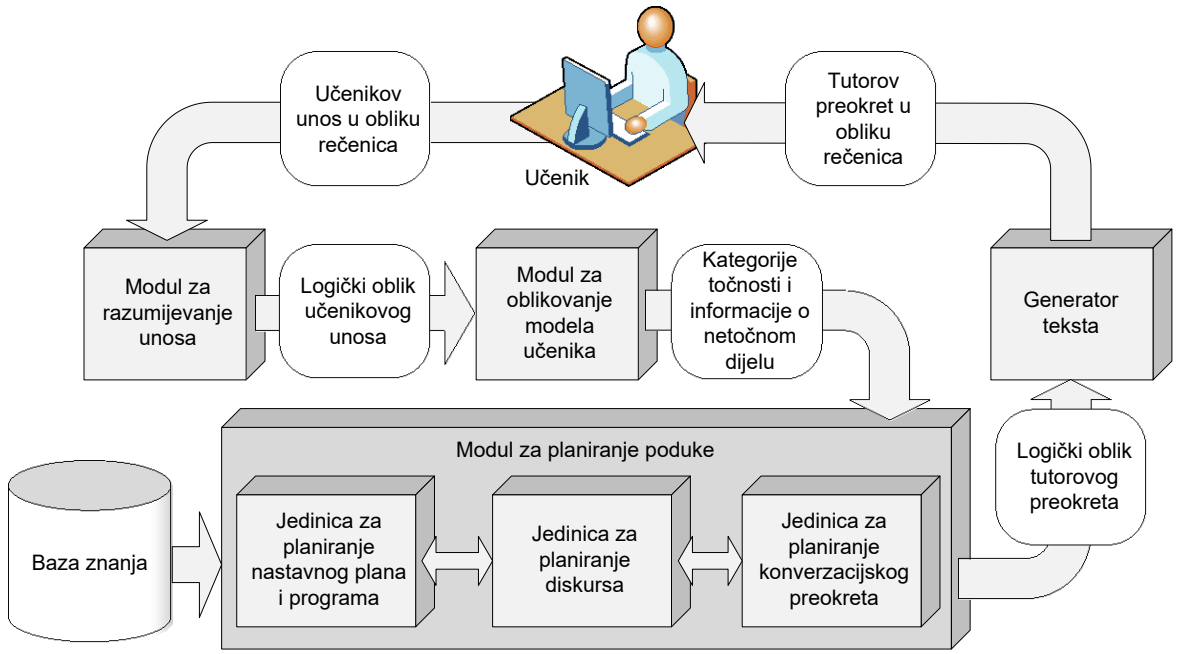
3.2 CIRCSIM-Tutor

Začetak CIRCSIM-Tutor-a [EVAN2005] se dogodio 1987. godine na Rush Medical College pod inicijativom profesora fiziologije Allena A. Rovicka i Joela A. Michaela koji su godinu ranije napisali BASIC program CIRSCIM namijenjen podršci učenju o baroreceptorskom refleksu [ROVI1986]. Od tada se u razvoj sustava uključio veliki broj stručnjaka različitih sveučilišta što je rezultiralo s tri verzije inteligentnog tutorskog sustava CIRSCIM-Tutor.
CIRCSIM-Tutor pomaţe studentima prve godine medicine u rješavanju problema vezanih uz baroreceptoski refleks i sustav regulacije krvnog pritiska u ljudskom tijelu [NAHK1989]. Prvo se učeniku prezentira opis neke fiziološke smetnje te se od učenika traţi predviĎanje promjene koje su vezane uz sedam najvaţnijih kardiovaskularnih parametara. Učenik promjenu procjenjuje kvalitativno i CIRCSIM-Tutor analizira njegovo predviĎanje, pronalazi pogreške i odabire metodu za ispravak svake pogreške. Tutor se zatim upušta u dijalog s učenikom nastojeći ispraviti detektirane pogreške i zablude. Tablica 3.2 prikazuje primjer dijaloga izmeĎu CIRCSIM-Tutor-a i učenika.

Tablica 3.2. Primjer CIRCSIM-Tutor-ovog dijaloga prevedenog na hrvatski jezik

Tutor	Možeš li mi reći koje su determinante srednjeg arterijskog pritiska?
Učenik	CO
Tutor	U redu. Drugu determinantu kontrolira živčani sustav. Dakle, koja je to determinanta?
Učenik	TPR
Tutor	U redu. MAP je određen s CO i TPR. Koja je determinanta dominantna u ovom slučaju?
Učenik	CO
Tutor	Točno. Je li veza od CO prema MAP-u direktna ili inverzna?
Učenik	Direktna.
Tutor	Dobro. Onda koja je točna vrijednost MAP-a?
Učenik	-
Tutor	Točno. Molim te promijeni vrijednost MAP-a u tablici predviđanja.

Područno znanje o baroreceptorskom refleksu je prikazano uzročnom konceptualnom mapom. Veza u konceptualnoj mapi predstavlja uzročnu povezanost dva parametra sustava regulacije krvnog pritiska. Uzročna povezanost se očituje kod promjene vrijednosti jednog parametra čime se mijenja vrijednost drugog parametra.
Poduka uz pomoć CIRCSIM-Tutor-a se sastoji od rješavanja problema gdje učenik popunjava tablicu predviĎanja za odabrani problem. Kod svakog netočno popunjenog predviĎanja započinje tutorski dijalog s ciljem ispravljanja pogrešnog predviĎanja i uklanjanja učenikovih zabluda. Kroz tutorski dijalog CIRCSIM-Tutor nastoji navesti učenika na shvaćanje vlastite pogreške i zablude. U tom procesu CIRCSIM-Tutor koristi različite vrste natuknica pohranjenih u sustavu.
Svakom novom verzijom se mijenjala struktura CIRSCIM-Tutor-a. Slika 3.2 opisuje strukturu treće verzije CIRSCIM-Tutor-a [RAMZ1994].



Slika 3.2. Struktura i tijek obrade podataka sustava CIRCSIM-Tutor v.3

Komponente sustava CIRCSIM-Tutor obraĎuju učenikov unos i vraćaju tutorov dijaloški preokret. Prvo se učenikov unos analizira u modulu za razumijevanje unosa. Analiza se sastoji od ispravljanje pogrešaka u pisanju i od prebacivanja u logički oblik. Prilikom ispravljanja pogrešaka, unesene riječi se usporeĎuju s riječima iz leksikona. Nakon toga se uz pomoć kaskadnih strojeva s konačnim stanjima [ROCH1997] i leksičke funkcionalne gramatike [BRES1986] riječi povezuju u logičku formu. Dobiveni logički oblik učenikovog unosa se u sljedećem koraku koristi za ureĎivanje modela učenika. Modul za razumijevanje unosa i modul za oblikovanje modela učenika odlučuju u koju od osam kategorija će smjestiti učenikov unos. Kategorije točnosti učenikovog unosa mogu biti: točan, djelomično točan, pogrešan, odgovor sa zabludom, skoro promašen odgovor, promašen odgovor, "ne znam" i kombinacija nekih od navedenih kategorija. U sljedećem koraku se kategorije točnosti učenikovog unosa prosljeĎuju modulu za planiranje poduke. Jedinice modula za planiranje poduke sudjeluju u planiranju poduke. Jedinica za planiranje nastavnog plana i programa nudi brojne i raznolike probleme koji obuhvaćaju najvaţnije stavke nastavnog plana i programa te osigurava razinu problema kako učeniku ne bi bilo dosadno i kako ne bi rješavao preteške probleme. Ova jedinica u suradnji s jedinicom za planiranje diskursa preoblikuje problem iz nastavnog plana i programa, odabire odgovarajuću metodu za poučavanje i postavlja logički oblik preokreta. Konačno, jedinica za planiranje konverzacijskog preokreta uobličava logički preokret kako bi rečenice bile prirodnije. U sljedećem koraku se logički oblikovan tutorov preokret prenosi do generatora teksta koji svaku logički oblikovanu rečenicu preokreta pretvara u rečenicu prirodnog jezika.

Modul za planiranje poduke suraĎuje s bazom znanja koja skupom pravila i tablica oblikuje razne vrste znanja. Baza znanja CIRCSIM-Tutor-a ne sadrţi samo područno znanje već i nastavni plan i program, povijest dijaloga, povijest poduke i leksikon.

U trećoj verziji CIRSCIM-Tutor-a se uveo novi pristup planiranju i voĎenju dijaloga implementiran u modulu za planiranje poduke. Temelj ovog modula je reaktivni hijerarhijski planer Atlas Planning Engine (APE) [FREE1996] [FREE1999] [FREE2000] [MILL2001] koji

- dopušta reaktivno planiranje jer je nemoguće obuhvatiti sve moguće učenikove odgovore,

- koristi višestruke tutorske protokole čime se simulira česta promjena tutorske metode ljudskog tutora s obzirom na učenikove odgovore,
- planira više konverzacijskih preokreta i
- uvodi leksičke raznolikosti koje osiguravaju razgovor s manje ponavljanja.

Razlaganje plana u APE je rekurzivno i pri tome se reaktivno pristupa planiranju. Ako bi se čekalo na izvršenje svakog plana u hijerarhiji, tekst prirodnog jezika bi se često ponavljao i djelovao bi neprirodno. Umjesto planiranja cijelog teksta, APE reaktivno mijenja planove ovisno o učenikovom unosu i planira točno onoliko koliko je potrebno za generiranje sljedećeg konverzacijskog preokreta. Pri tome APE koristi tri mehanizma za promjenu plana:

- procjenjivanjem uvjeta se utvrĎuje je li plan ispunjen te se ostatak plana preskače,
- povratak na točku odlučivanja i promjena odluke je mehanizam koji se koristi kada postoji više načina zadovoljavanja cilja čime se olakšava sloţeno reaktivno planiranje,
- mehanizam uklanjanja ciljeva briše ciljeve iz podsjetnika i zamjenjuje ih novim. Ovim mehanizmom se olakšava snalaţenje u slučaju neočekivanih odgovora.

Obrada prirodnog jezika u sustavu CIRCSIM-Tutor v.3 temelji se na APE koji sluţi za planiranje i upravljanje dijalogom. Reaktivno, hijerarhijsko i pravovremeno planiranje ima brojne prednosti. Jedna od glavnih je što sustav u bilo kojem trenutku moţe promijeniti plan koji se odnosi na daljnji tijek poduke i tako se prilagoditi razini učenikovog znanja.

3.3 Why2-Atlas

Cilj ATLAS projekta je omogućavanje učeniku učenje temeljeno na konverzaciji s inteligentnim tutorskim sustavom zasnovanom na prirodnom jeziku. Kako bi inteligentni tutorski sustav bio u stanju komunicirati na prirodnom jeziku, dodani su mu APE i Core component for Assessing the Meaning of Explanatory Language (CARMEL) [ROSÉ2000a] [ROSÉ2000b]. Zatim je postojeći ITS za Newton-ovu fiziku Andes [GERT1998] proširen APE i CARMEL komponentama i dobiven je sustav Atlas-Andes [FREE1999] [ROSÉ2001]. Atlas-Andes vodi učenika po pravcu zaključivanja kako bi učenik usvojio osnovno konceptualno znanje iz fizike. Pokazano je kako Andes sustav proširen s dijalogom značajno unapreĎuje učenje u odnosu na standardnu verziju [ROSÉ2001].
WHY2 projekt nastavlja koracima Atlas projekta u opisivanju fizičkih sustava na prirodnom jeziku. Kao nasljednik WHY sustava [STEV1977] koji pomaţe učeniku u artikuliranju objašnjenja rješenja problema, WHY2 otkriva, katalogizira i otklanja učenikova pogrešna poimanja. Nadalje, uključivanjem ATLAS sustava se dobio Why2-Atlas [KURT2002] [JORD2006] koji otklanjanje učenikovog pogrešnog poimanja realizira dijalogom miješane inicijative koji je prikazan u tablici 3.3.

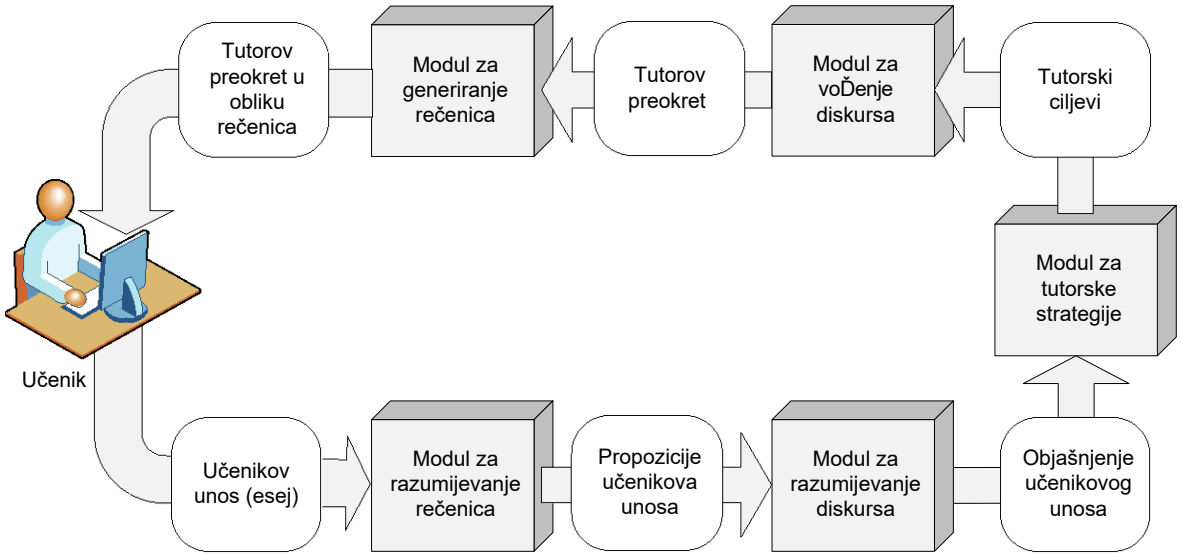
Tablica 3.3. Primjer Why2-Atlas-ovog dijaloga prevedenog na hrvatski jezik

Tutor	Znaš li zakon koji povezuje silu i akceleraciju? Ako znaš, molim te definiraj mi ga.
Učenik	f = ma
Tutor	Dobro. Primjenjujući ovo pravilo, koja je vrijednost horizontalne komponente akceleracije na jaje? Molim te objasni svoj zaključak.
Učenik	Nula jer nema horizontalne sile koja djeluje na jaje.
Tutor	Vrlo dobro. Kakva je horizontalna prosječna brzina jaja za vrijeme leta u odnosu na njegovu početnu horizontalnu brzinu?
Učenik	Jednaka

U dijalogu iz tablice 3.3 se miješa obrada matematičkih formula i teksta na prirodnom jeziku. Dva su temeljna cilja koje Why2-Atlas nastoji dostići [MAKA2005]:

- korištenje prirodnog jezika za opisivanje koncepata područnog znanja i
- razumijevanje odnosa meĎu konceptima i njihovo objašnjenje zakonima i formulama.

Why2-Atlas koristi duboku analizu sintakse i metode dokazivanja teorema kako bi preoblikovao učenikov esej u dokaz. Dokazom se formalizira ono što je učenik rekao zajedno s vjerovanjem što je ţelio reći. Ovime se sustavu omogućava otkrivanje pogrešnog poimanja, kao i nedostajućih dijelova objašnjenja. Ako sustav naiĎe nedostatke u eseju, onda se realizira dijalog kojime se nastoji otkloniti nedostajuća ili pogrešna poimanja učenika. Potrebno je nekoliko iteracija u kojima učenik, ispravljajući esej, pokušava dati prihvatljivo objašnjenje zadanog problema. Slika 3.3 prikazuje strukturu Why2-Atlas sustava i korake obrade učenikova eseja.



Slika 3.3. Struktura i tijek obrade podataka sustava Why2-Atlas

Nakon što student unese esej, modul za razumijevanje rečenice provede sintaktičku analizu rečenice i producira propozicije temeljene na predikatnoj logici prvog reda. U sljedećem koraku modul za razumijevanje diskursa asimilira propozicije formirajući objašnjenja. Objašnjenja se oblikuju tako što se svaka propozicija učenikova unosa proširi zaključcima koji vode do propozicije. Nadalje, modul za tutorske strategije analizira potpunost i ispravnost objašnjenja kako bi ustvrdio postoje li pogrešna poimanja. Zatim se kreiraju tutorski ciljevi kojima se nastoji pomoći učeniku u otklanjanju pogrešnog poimanja. U sljedećem koraku se uz pomoć modula za voĎenje diskursa oblikuje struktura dijaloga zvana Knowledge Constructed Dialogue (KCD) [ROSÉ2001]. KCD opisuje interaktivno usmjereno zaključivanje kojim se učenik navodi učenju jednog ili više koncepata po strategiji postavljanja upita. Modul za voĎenje diskursa izdvaja sljedeći tutorov preokret iz strukture dijaloga koji se preoblikuje u tekst na prirodnom jeziku uz pomoć modula za generiranje rečenica.

Svaki od modula Why2-Atlas sustava obraĎuje prirodan jezik na odreĎenom nivou. Za prepoznavanje prirodnog jezika se koristi modul za razumijevanje rečenica. Ovaj modul ujedinjuje leksički procesor, sintaktički analizator, komponentu za popravljanje rečenice i statistički analizator u CARMEL sustav. Leksički procesor koristi leksikon kako bi otkrio osnovne oblike riječi i ujedno ispravlja pravopisne pogreške. Zatim se sintaktičkim analizatorom otkriva struktura pojedinih rečenica na osnovu kombinacije funkcionalne

gramatike [HALL1985] i leksičke funkcionalne gramatike. Ako sintaktički analizator ne uspije opisati strukturu rečenice, tada se upošljava komponenta za popravljanje rečenice. Genetički algoritam komponente za popravljanje nastoji iz fragmenata rezultata sintaktičke analize doprijeti do strukture rečenice. U slučaju neuspjelog popravljanja rečenice izvršava se statistička analiza metodom naivne Bayes-ove mreţe ili latentnom semantičkom analizom. Na nivou semantike rečenice, modul za razumijevanje diskursa pronalazi dokaze za logičke forme učenikovog unosa na zadani problem. Dokaz je predstavljen skupom stabala čiji su listovi činjenice problema, a korijeni učenikove propozicije [MAKA2004]. Planiranje se izvršava u modulu za tutorske strategije koji analizira dokaze i pronalazi nedostatke. Za svaki nedostatak se formira cilj kojeg modul za voĎenje diskursa oblikuje u KCD. KCD struktura je opisana strojem s konačnim brojem stanja u kojemu stanja predstavljaju pitanja namijenjena učeniku. Modul za generiranje rečenica je zapravo RealPro sustav [LAVO1997] koji se koristi i u verzijama DIAG-NLP sustava.

3.4 DIAG-NLP

Projekt DIAG-NLP je trajao od 2000. do 2005. godine na University of Illinois pod vodstvom Barbare Di Eugenio. Ciljevi projekta su:

- procijeniti je li jednostavno generiranje prirodnog jezika djelotvorno u poboljšanju komunikacije s inteligentnim tutorskim sustavom,
- procjena što je učenik naučio za vrijeme interakcije na prirodnom jeziku s inteligentnim tutorskim sustavom,
- iskoristiti rezultate navedenih procjena i ostale prikupljene podatke za razvoj sofisticiranijih sustava.

Općenito se ţeljelo shvatiti što se moţe postići dodavanjem generatora prirodnog jezika već postojećem sustavu i koliko sustav, koji koristi jednostavne tehnike generiranja prirodnog jezika, moţe biti učinkovit. Umjesto gradnje potpuno samostalnog sustava odlučeno je poboljšati jezičnu povratnu informaciju već postojeće ljuske inteligentnog tutorskog sustava DIAG [TOWN1997]. DIAG sustav učeniku prikazuje vizualnu reprezentaciju relativno sloţenog sustava. Sloţeni sustav je modeliran skupom komponenata i skupom indikatora. DIAG simulira kvar jedne ili više komponenti, a učenik na osnovu indikatora otkriva kvar i zamjenjuje neispravne komponente. Kako bi se pomoglo učeniku u usvajanju vještine otkrivanja kvarova, DIAG moţe sugerirati rješenja na osnovu učenikovog istraţivanja kvara. Primjer sugestije za sustav izgaranja nafte je dan u tablici 3.4.

Tablica 3.4. Primjer povratne informacije DIAG sustava (prevedeno na hrvatski jezik)

DIAG
Vizualni indikator izgaranja pokazuje stanje paljenja, što je abnormalno.
Mlaznica nafte uvijek daje ovu abnormalnost kada zakaţe.
Ventil za opskrbu naftom uvijek daje ovu abnormalnost kada zakaţe.
Naftna pumpa uvijek daje ovu abnormalnost kada zakaţe.
Filter nafte uvijek daje ovu abnormalnost kada zakaţe.
Upravljački modul sustava ponekad daje ovu abnormalnost kada zakaţe.
Sklop upaljača nikada ne producira ovu abnormalnost kada zakaţe.
Motor plamenika uvijek daje ovu abnormalnost kada zakaţe.

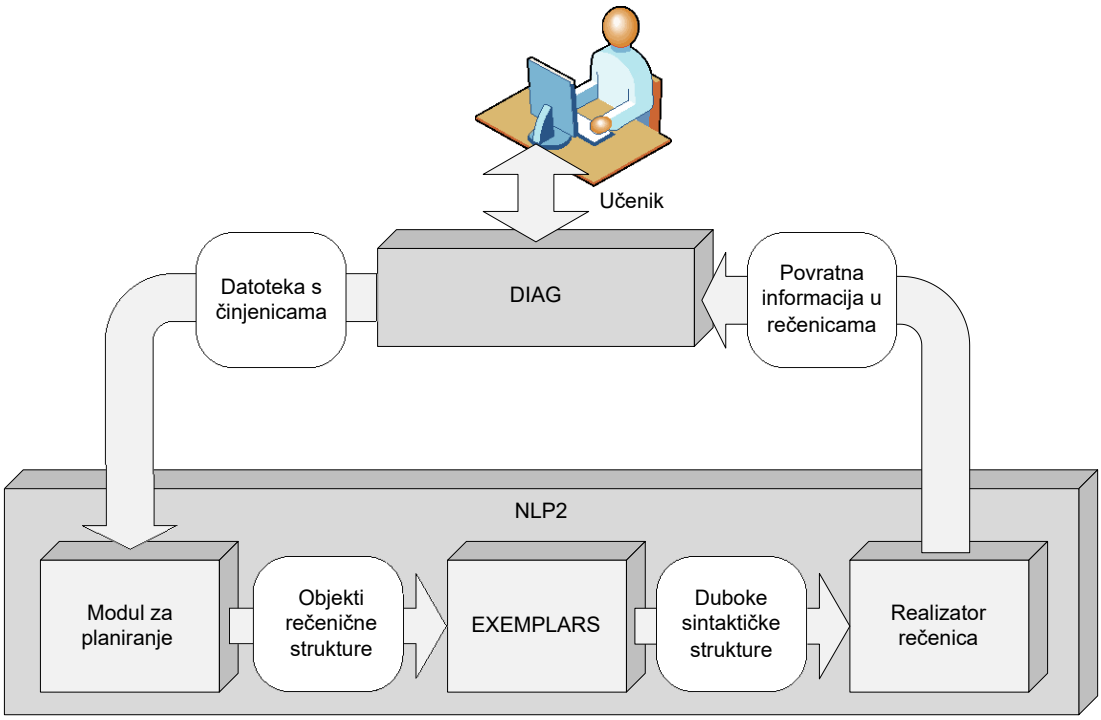
Razvijene su dvije verzije generatora povratne informacije koji su implementirani u dvije verzije sustava, DIAG-NLP1 [DIEU2002] i DIAG-NLP2. Obje verzije sustava su fokusirane na planiranje rečenica i pogotovo na njihovom grupiranju. Pokazalo se kako funkcionalno

grupiranje rečenica poboljšava učenje, dok sintaktičko grupiranje ne poboljšava učenje [DIEU2005a] [DIEU2005b]. Primjer različitih grupiranja rečenica povratne informacije kod sustava DIAG-NLP1 i DIAG-NLP2 je prikazano u tablici 3.5.

Tablica 3.5. Primjer grupiranja rečenica u sustavu DIAG-NLP1 i DIAG-NLP2 (prevedeno na hrvatski jezik)

DIAG-NLP1
Vizualni indikator izgaranja pokazuje stanje paljenja. To je abnormalno. Normalno je stanje izgaranja. U sustavu za taljenje, ovo se ponekada dogodi ako je upravljački modul sustava zakazao. U sustavu za izgaranje nafte ovo se nikada ne dogodi ako je sklop upaljača zakazao. Za razliku od toga, ovo se uvijek dogodi ako je motor plamenika, filter nafte, naftna pumpa, ventil za opskrbu naftom ili mlaznica nafte zakazala.
DIAG-NLP2
Izgaranje je abnormalno. U sustavu za izgaranje nafte, provjeri jedinice između naftne pumpe i motora plamenika.

DIAG-NLP2 značajno skraćuje originalnu DIAG povratnu informaciju i pri tome zadrţava smisao. Pokazuje se kako generator prirodnog jezika DIAG-NLP2 sustava realizira povratnu informaciju koja ujedno motivira učenika [DIEU2008]. Moduli koji sudjeluju u obradi povratne informacije DIAG-NLP2 sustava su prikazani na slici 3.4.



Slika 3.4. Struktura i tijek obrade podataka sustava DIAG-NLP2

Korisničko sučelje DIAG-NLP2 sustava osigurava DIAG sustav. Preko korisničkog sučelja učeniku se prezentira niz problema koje treba riješiti. Za vrijeme rješavanja problema učenik

moţe zatraţiti savjet od sustava tako što pošalje upit DIAG-NLP2 sustavu. Tada DIAG generira datoteku s činjenicama. U toj tekstualnoj datoteci sakupljene su informacije potrebne za generiranje povratne informacije sustava. Modul za planiranje odlučuje koje povratne informacije uključiti i producira jedan ili više objekata rečenične strukture. Svaki objekt rečenične strukture se sastoji od sintaktičkih elemenata, kao što su subjekt, objekt, glagol, i ostali. Objekti rečenične strukture se preko programskog okvira EXEMPLARS [WHIT1998] oblikuju u duboke sintaktičke strukture. Za površinsku realizaciju rečenica se koristi RealPro sustav koji nakon preuzimanja sintaktičke strukture rečenica od EXEMPLARS generira tekst.
EXEMPLARS je programski okvir koji koristi skup korisnički definiranih pravila zvanih egzemplari. Egzemplarima se opisuje tijek procesa generiranja sadrţaja rečenica. Izlazne rečenice su opisane strukturom stabla čijim su čvorovima pridruţena proširenja ili usavršavanja prethodno generiranih rečenica, a listovi predstavljaju generiranu sintaktičku strukturu rečenice. Dobivena sintaktička struktura predstavlja ulaz u RealPro programsko sučelje. RealPro se temelji na teoriji transformacije teksta u značenje (Meaning-Text Theory) [MELČ1988] i obrnuto. Po ovoj teoriji struktura rečenice je opisana stablom čiji su čvorovi leksički označeni oblicima riječi, a veze predstavljaju sintaktičke relacije, kao što su subjekt, objekt i tako dalje. Na osnovu strukture rečenice RealPro oblikuje tekst na prirodnom jeziku.

3.5 Usporedba inteligentnih tutorskih sustava zasnovanih na obradi prirodnog jezika

Kao predstavnici inteligentnih tutorskih sustava koji se koriste tehnikama obrade prirodnog jezika, opisana su četiri sustava: AutoTutor, CIRCSIM-Tutor, Why2-Atlas i DIAG-NLP. Prikazana je njihova struktura i tehnike korištene za obradu jezika. Za sustave koji imaju više od jedne verzije, istraţene su posljednje verzije sustava. AutoTutor-3D je verzija sustava AutoTutor koja posjeduje interaktivne simulacije, a ujedno se razvija nova verzija koja moţe pratiti emotivna stanja učenika. Kod sustava CIRCSIM-Tutor za usporedbu su razmatrane karakteristike treće verzije sustava, a kod sustava DIAG-NLP karakteristike njegove druge verzije.
Obrada i korištenje prirodnog jezika u navedenim sustavima ujedno ovisi o standardnim karakteristikama sustava. Tablica 3.6 prikazuje usporedbu navedenih inteligentnih tutorskih sustava obzirom na problemsko područje poučavanja, tutorske strategije, modeliranje učenika i načine komunikacije s učenikom.

Tablica 3.6. Usporedba inteligentnih tutorskih sustava AutoTutor-3D, CIRCSIM-Tutor v.3, Why2-Atlas i DIAG-NLP2 s obzirom na njihove opće karakteristike

AutoTutor-3D	CIRCSIM-Tutor v.3	Why2-Atlas	DIAG-NLP 2
Područje poučavanja			
Generički sustav. Razvijena područja računarstva, fizike, biologije.	baroreceptorski refleks i sustav regulacije krvnog pritiska u ljudskom tijelu	kvalitativni problemi iz Newton-ove fizike	pronalaţenje kvara u sloţenim sustavima
Načini komunikacije			
• animirani pedagoški agent • dijalog na prirodnom jeziku • interaktivne simulacije	• izbornici za odabir problema • tablica predviĎanja • dijalog na prirodnom jeziku	• dijalog na prirodnom jeziku	• interaktivne grafičke simulacije • izbornici za upit • povratna informacija na prirodnom jeziku
Tutorska sesija			
1. sustav učeniku prezentira problem ili pitanje 2. dijalog miješane inicijative s učenikom kroz koji se odgovor učenika oblikuje 3. na kraju dijaloga od učenika se traţi potpuno objašnjenje početnog problema	1. učenik odabire problem 2. za njega popunjava tablicu predviĎanja 3. svako netočno predviĎanje generira tutorski dijalog	1. sustav učeniku prezentira problem 2. učenik daje objašnjenje problema kroz esej 3. ukoliko esej sadrţi nedostatke ili pogrešna poimanja, pokreće se dijalog 4. učenik prepravlja esej 5. ako esej opet nije zadovoljavajući, ciklus se ponavlja.	1. sustav prezentira problem i grafičku simulaciju kvara u sustavu 2. učenik traţi kvar i zamjenjuje pokvarenu komponentu 3. putem izbornika učenik moţe zatraţiti pomoć 4. sustav na prirodnom jeziku odgovara na zatraţenu pomoć
Prilagodba učeniku			
Ovisi o: • prethodnoj uspješnosti u rješavanju problema • pokrivenosti očekivanih dobrih odgovora i zabluda za trenutni problem • učenikovim kognitivnim stanjima • učenikovim emotivnim stanjima (nova verzija). tijek dijaloga.	Ovisi o: • procjeni učenikove sposobnosti rješavanja problema. • povijesti učenikovih odgovora • pogreškama tijekom rješavanja problema • povijesti obraĎenih tema • povijesti korištenih tutorskih metoda. • zapisu o učinkovitosti pojedine metode	Ovisi o povijesti poduke. PrilagoĎava se reaktivnim sposobnosti svakog učenikovog konverzacijskog preokreta. PrilagoĎavaju se: • tutorski plan • metode poučavanja • teme poučavanja	Ovisi o učenikovom postupku rješavanja planiranje na temelju eseja i
Razine teţine problema			
Lagani, srednji i teški problemi. Razina teţine se prilagoĎava s obzirom na učenikovu uspješnost u rješavanju prethodnih problema.	Četiri teţine problema. Razina teţine se prilagoĎava s obzirom na učenikovu uspješnost u rješavanju prethodnih problema.	Problemi nisu podijeljeni prema teţini. Postoje različite razine teţine tutorovih pitanja u dijalogu.	Svaki sljedeći problem je teţi od prethodnog.
Baze znanja			
• skripta nastavnog plana i programa • zbirka elektronskih tekstova • povijest dijaloga	• područno znanje • nastavni plan i program • povijest dijaloga • povijest poduke • leksikon	• unaprijed pripremljeni dijalozi • lanci zaključivanja	relacijska baza podataka

Što se tiče obrade prirodnog jezika, od navedena četiri sustava najjednostavniji je DIAG-NLP2. To je ujedno i jedini od njih koji se više ne razvija jer su ciljevi projekta DIAG ispunjeni. Jednostavnost sustava DIAG-NLP2 se u prvom redu odnosi na način interakcije izmeĎu učenika i sustava koja se ne odvija u obliku pravog dijaloga na prirodnom jeziku jer učenik radi s grafičkim interaktivnim modelom. Ako mu zatreba pomoć, putem izbornika šalje zahtjev na kojeg sustav generira odgovor na prirodnom jeziku. Dakle, u sustavu DIAG-NLP2 proces obrade prirodnog jezika svodi se na generiranje prirodnog jezika. Kako je jedan od ciljeva projekta DIAG bio saznati koliko sustav koji koristi jednostavne i jeftine tehnike

generiranja prirodnog jezika moţe biti učinkovit, a generiranje se temelji na jednoj od najjednostavnijih tehnika, a to je popunjavanje predloţaka.

Nešto sloţeniji je sustav CIRCSIM-Tutor. On spada u grupu inteligentnih tutorskih sustava koji vode učenika korak po korak kroz proces rješavanja problema. Učenik rješava zadani problem popunjavanjem tablice predviĎanja. Pogrešno popunjavanje tablice uzrokuje generiranje dijaloga na prirodnom jeziku kojim sustav navodi učenika na uviĎanje pogrešaka i njihovo ispravljanje. U takvom dijalogu učenik najčešće daje kratke odgovore koji ne zahtijevaju sloţene tehnike za razumijevanje prirodnog jezika. Za odreĎivanje ciljeva dijaloga koristi se reaktivno planiranje koje karakterizira mogućnost promjene plana u svakom trenutku dijaloga s ciljem što bolje prilagodbe učeniku i njegovim odgovorima.

Dijalozi korišteni u CIRCSIM-Tutoru posluţili su kao inspiracija za oblikovanje dijaloga konstruiranja znanja u sustavu Why2-Atlas. U oba sustava, CIRCSIM-Tutor-u i Why2-Atlas-u, za generiranje prirodnog jezika koristi se isti reaktivni planer, a dijalozi, u kojima su odgovori učenika najčešće kratki, generiraju se isključivo kako bi se ispravile učenikove pogreške i zablude. MeĎutim, u sustavu Why2-Atlas učenik umjesto popunjavanja tablice predviĎanja mora dati objašnjenje fizikalnog problema u obliku eseja na prirodnom jeziku pa su potrebne sloţenije metode razumijevanja prirodnog jezika. Osnovna ideja za analiziranje učenikovih eseja, pomoću popisa neizostavnih točaka i zabluda, preuzeta je od sustava AutoTutor. Razlika je u tome što su za razumijevanje prirodnog jezika u Why2-Atlas-u korištene tehnike oslonjene na kombinaciju statističkih metoda i metoda temeljenih na znanju. U sustavu AutoTutor obrada prirodnog jezika se temelji na latentnoj semantičkoj analizi, posebnoj statističkoj metodi za klasifikaciju teksta. Od navedenih sustava AutoTutor je jedini razvijen za više različitih područja poučavanja, a jedno od njih je i područje fizike. Tutoring Research Group koja je radila na projektu AutoTutor suraĎivala je s Natural Language Tutoring Group koji su radili na projektu Atlas kako bi usporedili različite pristupe obradi prirodnog jezika u inteligentnim tutorskim sustavima koji poučavaju slično područje. Zbog toga se verzija AutoTutor-a namijenjena za poučavanje fizike naziva Why2-AutoTutor.

Osim poboljšanog pristupa obradi prirodnog jezika, AutoTutor koristi i različite tutorske metode koje povlače brojne oblike interakcije s učenikom. Sustav učeniku prezentira problem i kroz dijalog miješane inicijative potiče učenika na oblikovanje objašnjenja tog problema. Na kraju dijaloga sustav od učenika traţi potpuni odgovor na početno pitanje, za razliku od Why2-Atlas-a koji prvo traţi potpuni odgovor na pitanje u obliku eseja, a tek onda započinje dijalog dopune i ispravka tog odgovora. AutoTutor od učenika tijekom cijelog dijaloga traţi opširne odgovore i čak ga potiče na postavljanje pitanja. Uz to, od navedenih sustava, AutoTutor je jedini koji za interakciju s učenikom koristi animirane agente koji se, osim govora, koriste i neverbalnim znakovima komunikacije kao što su izrazi lica, ton glasa i pokreti glave i ruku.

Usporedba navedena četiri sustava, prikazana u tablici 3.7, fokusirana je na pitanja obrade i korištenja prirodnog jezika. Ova se pitanja u prvom redu odnose na metode obrade prirodnog jezika, odnosno pristupe i tehnologije korištene za generiranje i razumijevanje prirodnog jezika. Potom je dana usporedba dijaloga u tim sustavima s obzirom na pristup korišten za upravljanje dijalogom, kategoriju dijaloga i njegove opće karakteristike. Posebna pozornost je posvećena opisu tutorovog konverzacijskog preokreta s naglaskom na vrstama povratne informacije o kvaliteti učenikovog odgovora, oblicima pomoći koje pojedini sustav nudi učeniku i načinima ispravka zabluda i pogrešaka detektiranih u učenikovom odgovoru. Na kraju je usporeĎena sposobnost pojedinog sustava da odgovori na pitanja koja mu učenik postavi, ako je sustav uopće posjeduje.

Tablica 3.7. Usporedba inteligentnih tutorskih sustava AutoTutor-3D, CIRCSIM-Tutor v.3, Why2-Atlas i DIAG-NLP2 s obzirom na obradu i korištenje prirodnog jezika

AutoTutor-3D	CIRCSIM-Tutor v.3	Why2-Atlas	DIAG-NLP2
Metode obrade prirodnog jezika			
statističke metode	metode temeljene na znanju	hibridne metode	metode temeljene na znanju
Razumijevanje prirodnog jezika			
• klasifikacija dijelova govora • jezični analizatori • sintaktički analizator • detektor negacije leksikon • latentna semantička analiza	• ispravljanje pravopisnih pogrešaka u pisanju • leksikon gramatičkih oblicima riječi kaskadnih konačnih automata	• hibridni semantički analizator CARMEL • identifikator jednadţbi • sintaktičko-semantički • analizator u obliku	
Generiranje prirodnog jezika			
• planiranje tutorovog preokreta iz skripte nastavnog plana i programa • produkcijska pravila za • ulančavanje teksta savjeta tutora	• reaktivno planiranje teksta • generiranje temeljeno na predlošcima • realizacija temeljena na leksičkoj funkcionalnoj • realizator rečenica gramatici	• reaktivno planiranje teksta • generiranje temeljeno na predlošcima • realizator rečenica tutorov	• generiranje temeljeno na predlošcima • funkcionalno grupiranje rečenica Real Pro preokret RealPro
Upravljanje dijalogom			
Mreţa napredovanja dijaloga u obliku automata konačnih stanja proširena produkcijskim pravilima osjetljivima na učenikove sposobnosti i povijest dijaloga	Reaktivni planer APE	Reaktivni planer APE i mreţa konačnih stanja	
Kategorija dijaloga			
dijalog miješane inicijative	dijalog jednostrane inicijative	Učenik ima inicijativu u fazi pisanja eseja, ali dijalog karakterizira jednostrana inicijativa (samo sustav ima inicijativu)	
Karakteristike dijaloga			
Dijalog je na prirodnom jeziku. Svrha mu je oblikovanje objašnjenja zadanog problema pa potiče opširne učenikove odgovore. UsklaĎen je s očekivanim dobrim odgovorima i zabludama, a karakterizira ga okvir dijaloga od 5 koraka.	Dijalog je na prirodnom jeziku, a sluţi isključivo za ispravljanje pogrešaka i zabluda (generira se nakon detekcije pogreške ili zablude). Učenik najčešće daje kratke odgovore.	Dijalog konstruiranja znanja (poseban oblik dijaloga na prirodnom jeziku) se generira ukoliko se učenikov esej ne procijeni kao točan i potpun. Svaki dijalog ima pravac zaključivanja koju prati. Kroz dijalog se potiče učenika na samostalno zaključivanje i konstruiranje odreĎenog znanja. Učenikovi odgovori mogu biti kratki odgovori (najčešće) ili objašnjenja.	
Karakteristike tutorovog konverzacijskog preokreta			
Obuhvaća kratku povratnu informaciju o kvaliteti učenikovog odgovora, dio za poboljšavanje obuhvaćenosti očekivanog dobrog odgovora i signal učeniku za početak govora. Moţe sadrţavati oznake diskursa.	Potiče učenika na rješavanje problema i samostalnu konstrukciju objašnjenja. Moţe sadrţavati potvrde o točnosti učenikovog odgovora, oznake diskursa i ublaţivače.	Navodi učenika na uviĎanje i ispravak zabluda te "izvlači" potpunije objašnjenje. Svako tutorov preokret završava pitanjem. Preokret moţe sadrţavati potvrde o točnosti učenikovog odgovora i oznake diskursa.	
Povratna informacija o kvaliteti učenikovog odgovora			

Verbalna: pozitivna, negativna i neutralna Neverbalna: izrazi lica i tjelesni pokreti animiranog agenta	pozitivna i negativna (verbalna)	pozitivna i negativna (verbalna)	pozitivna i negativna (verbalna)
Oblici pomoći			
• poticajni izrazi • potpitanja • rečenice nadopune • tvrdnje • saţetci	• rečenice nadopune • natuknice • saţetci	• slični, ali pojednostavljeni problemi • saţeci • natuknice	Odgovor na jedno od dva postojeća tipa upita za pomoć koje sustav generira na učenikov zahtjev.
Način ispravka pogrešaka i zabluda			
• generirani ispravak • pomoćne interaktivne simulacije • kratki dijalozi	Generiranje dijaloga kojim se učenika nastoji navesti na shvaćanje i ispravljanje vlastitih pogrešaka i zabluda.	Generira se dijalog konstruiranja znanja kojim se traţi potpunije objašnjenje od učenika i navodi ga se na prepoznavanje i ispravljanje pogrešaka i zabluda	Kroz odgovor na učenikov upit sustav ukazuje na pogreške i predlaţe sljedeći korak.
Odgovaranje na učenikova pitanja			
Sustav potiče učenika na postavljanje pitanja (jedno za svaki problem). Ukoliko pitanja nisu u skripti nastavnog plana i programa, sustav pokušava odgovoriti na pitanje uz pomoć posebnog sustava za odgovaranje na pitanja.	Sustav ne potiče učenika na postavljanje pitanja. Moţe odgovoriti na jednostavna (definicijska) učenikova pitanja, ali nije u mogućnosti odgovarati na otvorena pitanja.	Učenik ne postavlja pitanja.	Učenik ne postavlja pitanja već odabire jedan od ponuĎenih upita za pomoć.

Ako se usporede opće karakteristike navedenih sustava s modelom CoLaB Tutor-a, onda CoLaB Tutor moţe sluţiti za poučavanje bilo kojeg područja ili segmenta područja koji se moţe formalno opisati opisnom logikom. Tutorska sesija CoLaB Tutor-a izmjenjuje učenje i testiranje znanja učenika. Prilikom učenja učenik usvaja znanje o pojedinom konceptu i zatim bira sljedeći povezani koncept. Testiranje se odvija dijalogom jednostrane inicijative temeljenom na pet koraka. U ovom dijaloškom okviru, prilagoĎavanje učeniku ovisi o uspješnosti prethodnih učenikovih odgovora. Pri tome se prate nedostajuća i pogrešna poimanja učenika.

CoLaB Tutor obradu kontroliranog jezika temelji na znanju. Generiranje i prepoznavanje teksta se oslanja na konceptima i relacijama kojima je opisano područno znanje. Za upravljanje dijalogom je zaduţen stroj s konačnim brojem stanja i potisnim stogom. Ovaj stroj je u stanju reaktivno generirati pomoćna pitanja i davati pozitivnu i negativnu povratnu informaciju. MeĎutim, stroj nije u stanju obraditi učenikova postavlja pitanja.

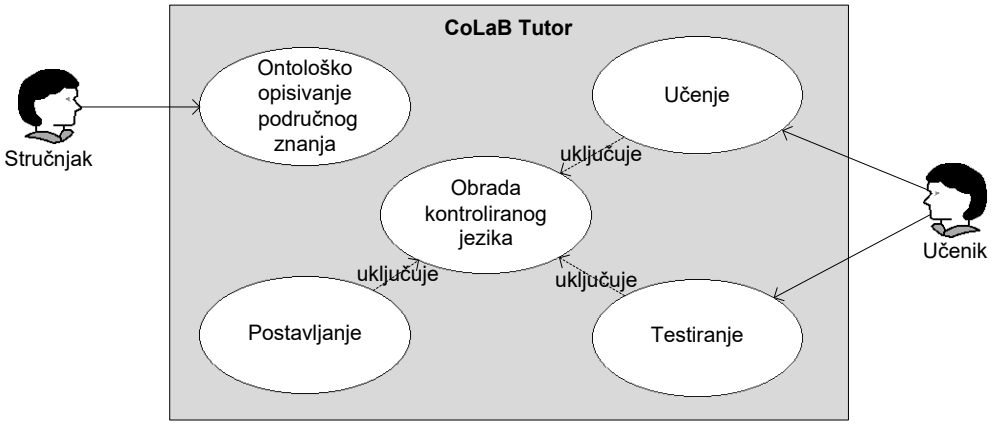
Zbog gramatike hrvatskog jezika, u CoLaB Tutor-u se nisu mogle iskoristiti komponente postojećih sustava jer su temeljene na morfologiji i sintaksi engleskog jezika. U svezi s navedenim, modelirane su posebne komponente isključivo orijentirane na sloţenost obrade hrvatskog standardnog jezika. Ipak, model CoLaB Tutor-a, koji se opisuje u sljedećem poglavlju, uključuje odabrane karakteristike analiziranih sustava.

4 Model sustava CoLaB Tutor

CoLaB Tutor ujedinjuje alat za formaliziranje znanja s ITS-om i alatom za obradu kontroliranog jezika kako bi učenik mogao učiti i testirati svoje znanje komunicirajući sa sustavom na kontroliranom jeziku. Karakteristika CoLaB Tutor-a je neovisnost o sadrţaju područnog znanja, to jest, bilo koje prethodno opisano područno znanje od strane stručnjaka predstavlja znanje namijenjeno učenju i testiranju. Ključna komponenta CoLaB Tutor-a je ITS koji vodi proces učenja, poučavanja i testiranja znanja, a za ostvarivanje komunikacije na kontroliranom jeziku koristi alat za obradu kontroliranog jezik koji je u stanju generirati i prepoznati izjave na kontroliranom jeziku.
Poglavlje započinje razmatranjem sudionika, funkcionalnosti i strukture CoLaB Tutor-a. Struktura CoLaB Tutor-a je opisana komponentama i skupovima podataka koji predstavljaju ulazne ili izlazne podatke komponenti. Modeli osnovnih skupova podataka CoLaB Tutor-a, kao što su područno znanje, nastavni sadrţaj i model učenika su formalno opisani u 4.3 poglavlju. Kao poseban skup podataka se formalno definira kontrolirani jezik u 4.4 poglavlju. Skupovima podataka je opisana statička struktura CoLaB Tutor-a, a dinamika se opisuje realizacijom funkcionalnosti koje su grupirane po fazama. Dinamički model po fazama je slijedno opisan u podpoglavljima 4.5, 4.6 i 4.7.

4.1 Sudionici i funkcionalnosti

Uvaţavajući zamisao novog modela inteligentnog tutorskog sustava opisanog u poglavlju 2, sudionici CoLaB Tutor-a su stručnjak i učenik. Stručnjak je zaduţen za ontološko opisivanje područnog znanja, dok učenik uči i testira znanje komunicirajući sa sustavom na kontroliranom jeziku, kao što je prikazano na slici 4.1.

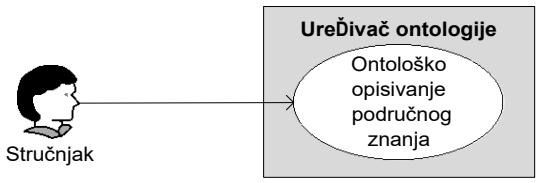


Slika 4.1. Sudionici i funkcionalnosti CoLaB Tutor-a

Za razliku od većine ITS-ova koji imaju područno znanje duboko ukorijenjeno u implementaciji modula stručnjaka, CoLaB Tutor nije zasnovan na jedinstvenom područnom znanju nad kojim se učenik uči i testira. Prethodno ontološki opisano područno znanje je potrebno postaviti kao izvor znanja CoLaB Tutor-u, nakon čega učenik moţe pristupiti učenju i testiranju svojeg znanja. Ontološki opisivanje područnog znanja i postavljanje područnog znanja u CoLaB Tutor su funkcionalnosti koje prethode učenju i testiranju učenika. Zbog toga

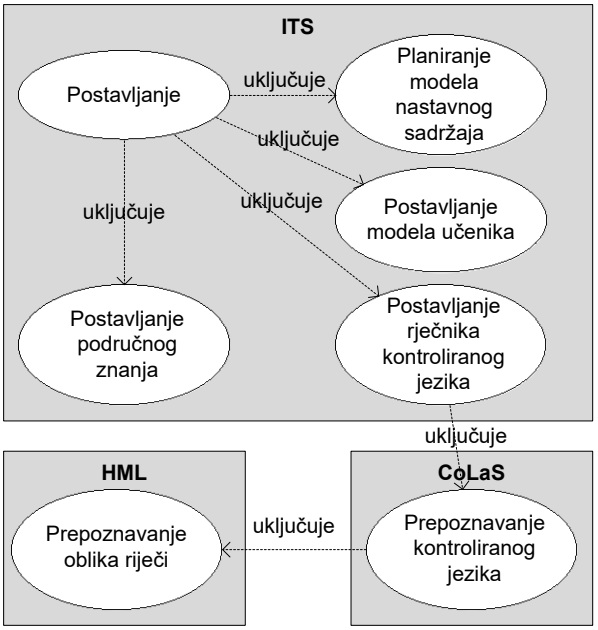
se realizacija funkcionalnosti CoLaB Tutor-a slijedno dijeli u 3 faze gdje su pojedini podsustavi CoLaB Tutor-a zaduţeni za realizaciju funkcionalnosti pojedinih faza:

1. Faza oblikovanja područnog znanja,
2. Faza postavljanja CoLaB Tutor-a,
3. Faza učenja i testiranja znanja učenika uz pomoć CoLaB Tutor-a.



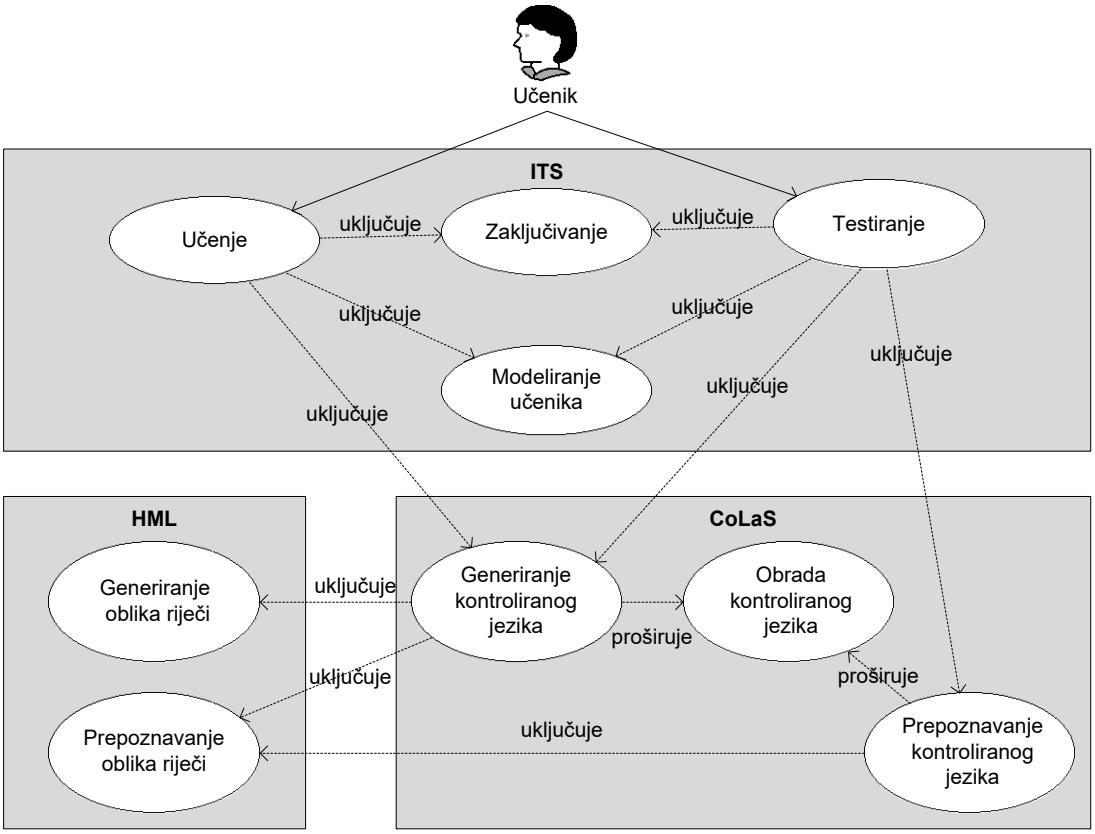
Slika 4.2. Faza oblikovanja područnog znanja

Prva faza nije direktno vezana s ITS-om CoLaB Tutor-a. U ovoj fazi stručnjak koristi nezavisni alat za opisivanje područnog znanja u ontološkom formatu (slika 4.2). UreĎivač ontologije Protégé je primjerak alata koji sluţi stručnjaku prilikom oblikovanja područnog znanja. Funkcionalnost ontološkog opisivanja područnog znanja sastoji se od opisivanja koncepata i relacija u OWL DL formatu.



Slika 4.3. Funkcionalnosti faze postavljanja CoLaB Tutor-a

Druga faza, koja slijedi nakon oblikovanja područnog znanja, je postavljanje CoLaB Tutor-a (slika 4.3). U ovoj fazi ITS, na osnovu ontološkog opisa područnog znanja, stvara područno znanje i planira strukturu nastavnog sadrţaja. Ontološki opisano područno znanje je eksterna reprezentacija znanja koja se za vrijeme postavljanja CoLaB Tutor-a analizira i transformira u područno znanje, odnosno u internu reprezentaciju znanja. Nad područnim znanjem se provodi analiza povezanosti koncepata kako bi se isplanirao nastavni sadrţaj. Objekti nastavnog sadrţaja dijele područno znanje na manje skupine koncepata čija meĎusobna povezanost utječe na postavljanje modela učenika. Druga faza uključuje postavljanje rječnika kontroliranog jezika koji se sastoji od riječi unutar naziva koncepata i relacija, odnosno riječi koje čine konceptne i relacijske fraze. Postavljanje rječnika je funkcionalnost u kojoj su uključene funkcionalnosti prepoznavanja kontroliranog jezika unutar CoLaS podsustava kao i prepoznavanja oblika riječi uz pomoć HML podsustava.



Slika 4.4. Funkcionalnosti faze učenja i testiranja znanja učenika

Treća faza započinje nakon planiranja i generiranja nastavnog sadrţaja jer su time stvoreni svi preduvjeti za učenje i testiranje znanja učenika (slika 4.4). Učenje i testiranje su dvije osnovne funkcionalnosti CoLaB Tutor-a koje su povezane s učenikom. Zaključivanje nad područnim znanjem je funkcionalnost koja je uključena za vrijeme učenja i testiranja. Funkcionalnost zaključivanja je realizirana postavljanjem upita nad eksplicitnim područnim znanjem rezultirajući novim implicitnim znanjem. Realizacija funkcionalnosti modeliranja učenikovog znanja rezultira promjenama u modelu učenika i utječe na daljnji tijek učenja i testiranja. Kod učenja i testiranja dolazi do izraţaja komunikacija kontroliranim jezikom izmeĎu učenika i ITS-a. CoLaB Tutor za obradu kontroliranog jezika koristi usluge CoLaS podsustava. Generiranje kontroliranog jezika je vrsta obrade kontroliranog jezika koja je uključena i kod učenja i kod testiranja znanja. Za vrijeme učenja, znanje se prikazuje u obliku rečenica kontroliranog jezika, a kod testiranja generiraju se upitne rečenice. Osim toga, tijekom testiranja učenikovi odgovori se, na komunikacijskom nivou, obraĎuju realizacijom funkcionalnosti prepoznavanja kontroliranog jezika. Generiranje i prepoznavanje kontroliranog jezika su funkcionalnosti CoLaS podsustava čija se realizacija temelji na uslugama Hrvatskog morfološkog leksikona HML [TADI2003] i na podacima Hrvatskog valencijskog leksikona glagola (CROVALLEX) [MIKE2008]. Obrada kontroliranog jezika CoLaS podsustava se provodi na razinama sintaktičkih jedinica rečenice i fraze, dok obradu morfoloških oblika riječi prepušta HML-u. Lematizacijski posluţitelj HML-a [TADI2006] pruţa usluge generiranja i prepoznavanja svih oblika riječi hrvatskog standardnog jezika koje su potrebne za obradu kontroliranog jezika. Za odreĎivanje gramatičke poveznice glagola i njegovih dopuna u frazama kontroliranog jezika CoLaS koristi CROVALLEX zapisan u XML formatu.

Jedini sudionici CoLaB Tutor-a su stručnjak, koji u prvoj fazi oblikuje područno znanje, i učenik koji sudjeluje u fazi učenja i testiranja znanja. U fazi postavljanja inteligentnog

tutorskog sustava nema sudjelovanja sudionika pošto se postavljanje obavlja strojno pomoću komponenti ITS-a.

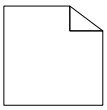
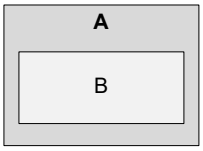
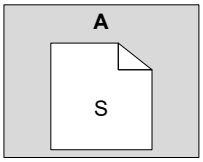
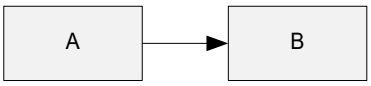
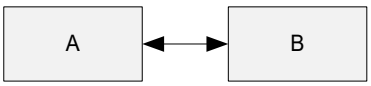
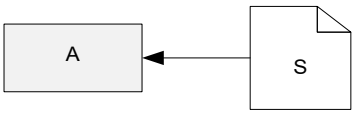
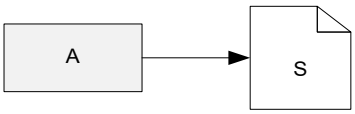
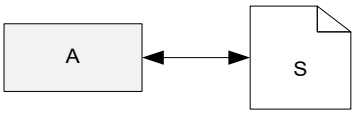
4.2 Struktura CoLaB Tutor-a

Realizacija funkcionalnosti CoLaB Tutor-a je usko povezana s njegovom strukturom. Osnovni elementi koji čine strukturu CoLaB Tutor-a su komponente i skupovi podataka. Komponenta je dio sustava koja, općenito gledano, vrši obradu podataka, a podaci se nalaze u skupovima podataka ili u parametrima pozvanih funkcija komponente. Funkcije su sastavni dio komponente, a slijed poziva funkcija komponenti opisuje realizaciju funkcionalnosti CoLaB Tutor-a. Ako jedna komponenta poziva barem jednu funkciju druge komponente, onda kaţemo kako te dvije komponente suraĎuju. Svaka funkcija moţe i ne mora imati ulazne i izlazne parametre. Ulazni parametri utječu na tijek izvršavanja funkcije, dok su izlazni parametri rezultati obrade koju funkcija provodi. Ulazni i izlazni parametri funkcije moţe biti i skup podataka CoLaB Tutor-a. Postoje dvije podvrste komponenti u CoLaB Tutor-u:

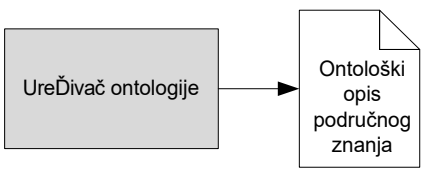
1. Podsustav je komponenta na najvišoj razini. Najčešće se sastoji od drugih komponenti, a nju direktno sadrţi sam sustav.
2. Modul je komponenta unutar podsustava koja moţe sadrţavati druge komponente.

Struktura CoLaB Tutor-a je, izmeĎu ostalog, opisana hijerarhijskim odnosom komponenti i veza meĎu njima. Ako su dvije komponente povezane, to označava kako jedna komponenta poziva funkciju druge komponente. Struktura CoLaB Tutor-a opisuje i vezu izmeĎu komponente i skupa podataka. Ova veza označava skup podataka kao ulazni ili izlazni parametar neke funkcije odreĎene komponente. Tablica 4.1 prikazuje grafičku notaciju koja se koristi za opisivanje strukture CoLaB Tutor-a.

Tablica 4.1. Grafička notacija komponenti CoLaB Tutor-a

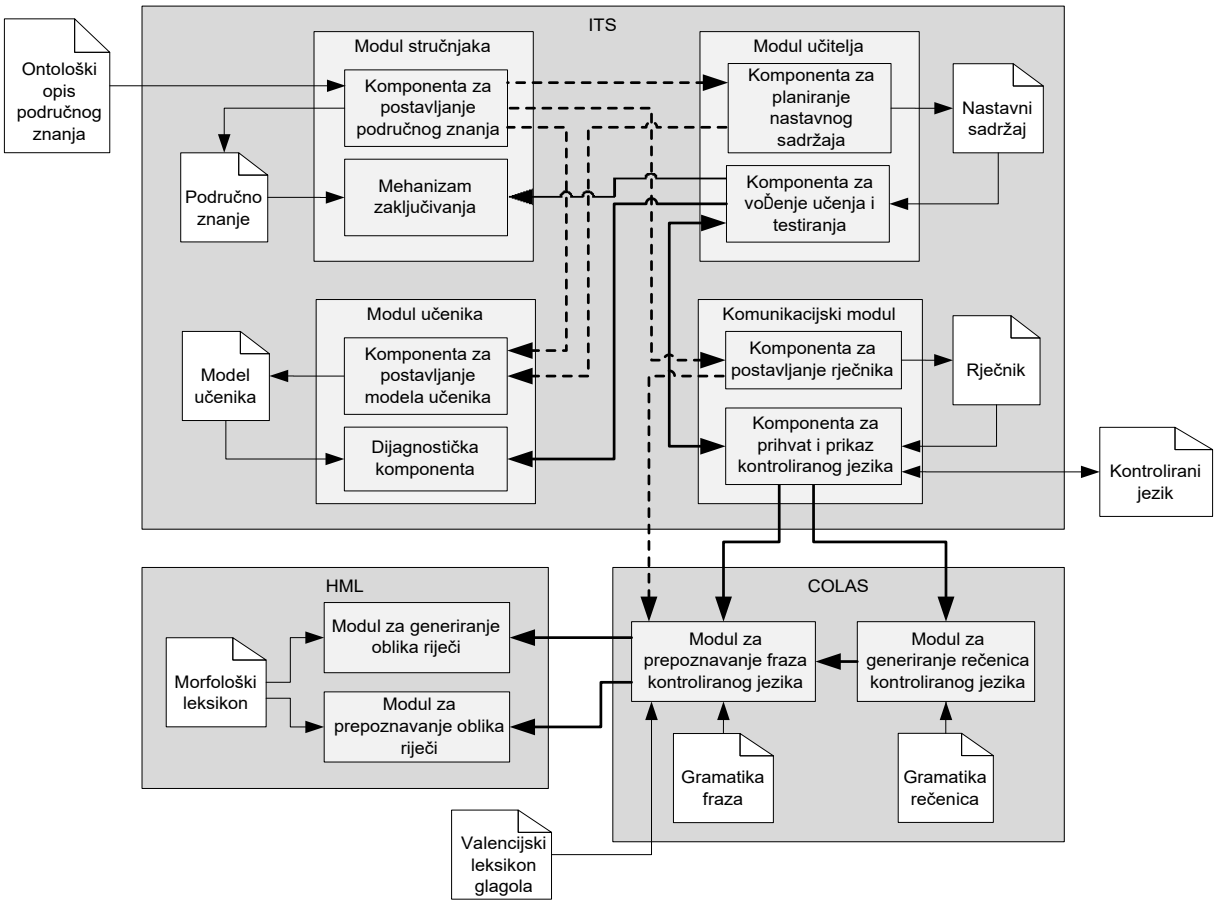
Grafička notacija	Značenje
	Oznaka za komponentu, modul ili podsustav
	Oznaka za skup podataka
	Komponenta A sadrţi komponentu B
	Komponenta A sadrţi skup podataka S
	Komponenta A poziva funkciju ili funkcije komponente B
	Komponenta A poziva funkciju ili funkcije komponente B i komponenta B poziva funkciju ili funkcije komponente A
	Skup podataka S je ulazni parametar funkcije ili funkcija komponente A
	Skup podataka S je izlazni parametar funkcije ili funkcija komponente A
	Skup podataka S je ulazni ili izlazni parametar funkcije ili funkcija komponente A

Grafička notacija CoLaB Tutor-a je izvedenica UML grafičke notacije za elemente dijagrama komponenti i dijagrama suradnje. MeĎusobno sadrţavanje komponenti i skupova podataka je notacija preuzeta iz UML dijagrama komponenti, dok povezanost izmeĎu dvije komponente ili komponente i skupa podataka je oznaka preuzeta iz UML dijagrama suradnje.
Grafičke notacije komponenti i skupova podataka sluţe za opisivanje strukture CoLaB Tutor-a. Kao što je u podpoglavlju 4.1 navedeno, realizacija funkcionalnosti CoLaB Tutor-a je podijeljena u tri faze. U prvoj fazi ureĎivač ontologije je podsustav kojim se definira alat za formalno opisivanje znanja. Kao rezultat prve faze dobiva se ontološki opis područnog znanja kao što je prikazano na slici 4.5.



Slika 4.5. Struktura alata za formalno opisivanje znanja

Realizacija funkcionalnosti druge i treće faze provodi se preko komponenti ITS podsustava i komponenti alata za obradu kontroliranog jezika. Podsustavi CoLaS i HML definiraju alat za obradu kontroliranog jezika. Struktura CoLaB Tutor-a za vrijeme druge i treće faze prikazana je na slici 4.6. Isprekidanom linijom je označena suradnja komponenti tijekom realizacije druge faze, a punom linijom za vrijeme treće faze.



Slika 4.6. Struktura inteligentnog tutorskog sustava i alata za obradu kontroliranog jezika

U drugoj fazi, odnosno fazi postavljanja CoLaB Tutor-a, ontološki opis područnog znanja obraĎuje komponenta za postavljanja područnog znanja. Kao rezultat obrade dobiva se područno znanje. Komponenta za postavljanje područnog znanja suraĎuje s komponentom za postavljanje rječnika kako bi se oblikovao rječnik kontroliranog jezika. Pri tome komponenta za postavljanje rječnika suraĎuje s modulom za prepoznavanje fraza kontroliranog jezika koji koristi gramatiku fraza, hrvatski valencijski leksikon glagola i suradnju s modulom za prepoznavanje oblika riječi kako bi odredio pravilne oblike riječi koje čine ureĎenu frazu kontroliranog jezika. Nadalje, komponenta za planiranje nastavnog sadrţaja oblikuje nastavni sadrţaj. Konačno, komponenta za postavljanje modela učenika oblikuje inicijalni model učenika čime je zaključena faza postavljanja.
Učenje i testiranje su osnovne funkcionalnosti treće faze u kojoj komponenta za voĎenje učenja i testiranja modula učitelja suraĎuje s komponentama modula stručnjaka, modula učenika i komunikacijskog modula. U realizaciji komunikacije kontroliranim jezikom, komunikacijski modul suraĎuje s modulima podsustava CoLaS i HML. Tablica 4.2 prikazuje odnos komponenti i funkcionalnosti CoLaB Tutor-a kod realizacije svih faza.

Tablica 4.2. Komponente i funkcionalnosti CoLaB Tutor-a

Komponenta			Funkcionalnost
podsustav	**modul**	**komponenta**	
UreĎivač ontologije			Ontološko opisivanje područnog znanja
ITS	Modul stručnjaka	Komponenta za postavljanje područnog znanja	Postavljanje područnog znanja
		Mehanizam zaključivanja	Zaključivanje
	Modul učitelja	Komponenta za planiranje nastavnog sadrţaja	Planiranje nastavnog sadrţaja
		Komponenta za voĎenje učenja i testiranja	Učenje
			Testiranje
	Modul učenika	Komponenta za postavljanje modela učenika	Postavljanje modela učenika
		Dijagnostička komponenta	Modeliranje učenika
	Komunikacijski modul	Komponenta za postavljanje rječnika	Postavljanje rječnika kontroliranog jezika
		Komponenta za prihvat i prikaz kontroliranog jezika	
CoLaS	Modul za generiranje rečenica		Generiranje kontroliranog jezika
	Modul za prepoznavanje fraza		Prepoznavanje kontroliranog jezika
HML	Modul za generiranje oblika riječi		Generiranje oblika riječi
	Modul za prepoznavanje oblika riječi		Prepoznavanje oblika riječi

Funkcionalnost ontološkog opisivanja područnog znanja je jedina funkcionalnost faze oblikovanja područnog znanja u kojoj sudjeluje samo stručnjak. U ovoj fazi se koristi alat za ontološko opisivanje područnog znanja kako bi se oblikovalo znanje koje će tijekom realizacije funkcionalnosti faze postavljanja sluţiti kao temelj područnom znanju, nastavnom sadrţaju, modelu učenika i rječniku kontroliranog jezika. Pri tome se koriste standardni moduli ITS-a kao i kod realizacije treće faze, odnosno faze učenja i testiranja. Kako bi se bolje razumjela realizacija navedenih funkcionalnosti, koja je opisana pozivima funkcija komponenti CoLaB Tutor-a, potrebno je formalno definirati modele skupova podataka.

4.3 Skupovi podataka CoLaB Tutor-a

Slika 4.6, osim što prikazuje komponente, prikazuje i temeljne skupove podataka strukture inteligentnog tutorskog sustava i alata za obradu kontroliranog jezika CoLaB Tutor-a, odnosno CoLaS podsustava. Faza postavljanja CoLaB Tutor-a zahtjeva prethodno ontološki opisano područno znanje koje će komponente inteligentnog tutorskog sustava preslikati u područno znanje i na osnovu područnog znanja oblikovati nastavni sadrţaj. Dobiveni nastavni sadrţaj i područno znanje sadrţe informacije potrebne za postavljanje modela učenika, a tijekom faze učenja i testiranja se vrši modeliranje učenika. Osim toga, ontološki opis područnog znanja uključuje nazive koncepata i relacija koji će, u fazi postavljanja, tvoriti rječnik kontroliranog jezika. Ontološki opis područnog znanja je temeljen na opisnoj logici, a

za potrebe CoLaB Tutor-a se vrši transformacija u područno znanje čiji je model temeljen na teoriji skupova i teoriji grafova. Na osnovi teorija upotrijebljenih kod modeliranja područnog znanja postavljen je model nastavnog sadrţaja i model učenika, a model kontroliranog jezika uključuje i formalnu teoriju jezika.

4.3.1 Ontološki opis područnog znanja

U fazi ontološkog opisivanja područnog znanja stručnjak oblikuje područno znanje korištenjem OWL DL jezika koji je podskup OWL jezika za opisivanje ontologija na Web-u. OWL jezik je zamišljen kao proširenje RDF, to jest, sintaksa OWL-a je sintaksa RDF-a, dok je semantika OWL-a proširenje semantike RDF-a [HAYE2004]. OWL posjeduje dosta zajedničkih osobina s opisnom logikom, meĎutim ima i mnoge razlike. Prije svega, sintaksa OWL-a je jednaka RDF sintaksi, odnosno informacije opisane OWL jezikom se zapisuju u RDF/XML obliku [BECK2004] [BECH2004] koji se zatim sintaktički analizira pomoću RDF grafa sastavljenog od ureĎenih trojki [KLYN2004]. Zbog jednostavnosti sintakse RDF grafa mnogi konstruktori opisne logike u OWL jeziku se preslikavaju u nekoliko ureĎenih trojki. RDF graf moţe biti i cikličan, što pruţa mogućnost stvaranja cikličkih sintaktičkih struktura u OWL-u koje nisu dozvoljene u opisnoj logici.
MeĎutim, OWL DL je jezik blizak opisnoj logici i to $\mathcal{SHOIN}^{(\mathcal{D})}$ familiji opisne logike. Zabranom korištenja opisnih konstruktora sa sintaktičkim ciklusima i disjunkcijom koncepata i relacija uvjetuje se jednaka izraţajnost OWL DL jezika i opisne logike. Zbog navedenih sintaktičkih restrikcija razvila se apstraktna sintaksa OWL DL jezika [PATE2004] koja se ne razlikuje mnogo od sintakse opisne logike. OWL DL apstraktna sintaksa sadrţi klase i područja podataka, koji su analogni konceptima i primitivnim tipovima podataka opisne logike. Nadalje, aksiomi i činjenice se jednako grupiraju kao i u opisnoj logici.
U semantici opisne logike, pa tako i u OWL DL ontologiji, koristi se interpretacija zasnovana na teoriji skupova kojom se generički koncept opisne logike, odnosno klasa OWL DL ontologije, interpretira kao skup individualnih koncepata, odnosno individua. Uloga se interpretira skupom ureĎenih parova individualnih koncepata. Interpretacija je definirana nad nekom domenom, koja moţe biti ili konačni ili beskonačni skup. Neka je $\Delta^{\mathcal{I}}$ domena interpretacije $\mathcal{I}$, tada ova interpretacija svakom konceptu A pridruţuje skup $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, a svakoj atomskoj individualnoj ulozi R pridruţuje binarnu relaciju $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. Atomski koncepti i atomske individualne uloge su osnovni elementi koji se ne mogu opisati preko drugih koncepata i uloga korištenjem konstruktora. TakoĎer opisna logika posjeduje univerzalni koncept $\top$ za kojeg vrijedi $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$, i prazni koncept $\bot$ koji je interpretiran s $\bot^{\mathcal{I}} = \emptyset$. Radi definiranje semantike jezika opisne logike funkcija interpretacije se proširuje opisivanjem koncepata induktivnim definicijama. Tablica 4.3 prikazuje odnos konstruktora i aksioma OWL DL ontologije sa sintaksom i semantikom opisne logike.

Tablica 4.3. Sintaksa i semantika $\mathcal{SHOIN}^{(\mathcal{D})}$ familije opisne logike [HORR2003a] [HORR2003b]

Konstruktor	Sintaksa	Semantika
atomska klasa	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
univerzalna klasa	$\top$	$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
prazna klasa	$\bot$	$\bot^{\mathcal{I}} = \emptyset$
tipovi podataka	D	$D^{\mathrm{D}} \subseteq \Delta_{\mathrm{D}}^{\mathcal{I}}$
individualna uloga	R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
podatkovna uloga	U	$U^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta_{\mathrm{D}}^{\mathcal{I}}$
individue	o	$o^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
vrijednosti podataka	v	$v^{\mathcal{I}} \in v^{\mathrm{D}}$
inverzna uloga	R^{-}	$(R^{-})^{\mathcal{I}} = (R^{\mathcal{I}})^{-}$
konjunkcija	$C_1 \sqcap \ldots \sqcap C_n$	$(C_1 \sqcap \ldots \sqcap C_n)^{\mathcal{I}} = C_1^{\mathcal{I}} \cap \ldots \cap C_n^{\mathcal{I}}$
disjunkcija	$C_1 \sqcup \ldots \sqcup C_n$	$(C_1 \sqcup \ldots \sqcup C_n)^{\mathcal{I}} = C_1^{\mathcal{I}} \cup \ldots \cup C_n^{\mathcal{I}}$
negacija	$\neg C$	$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$
nabrajanje individua	$\{o_1, \ldots, o_n\}$	$\{o_1, \ldots, o_n\}^{\mathcal{I}} = \{o_1{}^{\mathcal{I}}, \ldots, o_n{}^{\mathcal{I}}\}$
nabrajanje vrijednosti podataka	$\{v_1, \ldots, v_n\}$	$\{v_1, \ldots, v_n\}^{\mathcal{I}} = \{v_1{}^{\mathcal{I}}, \ldots, v_n{}^{\mathcal{I}}\}$
egzistencijalno ograničenje objektne uloge	$\exists R.C$	$(\exists R.C)^{\mathcal{I}} = \{x \mid \exists y.(x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
univerzalno ograničenje individualne uloge	$\forall R.C$	$(\forall R.C)^{\mathcal{I}} = \{x \mid \forall y.(x,y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
ograničenje vrijednosti individualne uloge	$R{:}\,o$	$(R{:}\,o)^{\mathcal{I}} = \{x \mid (x, o^{\mathcal{I}}) \in R^{\mathcal{I}}\}$
minimalno ograničenje individualne uloge	$\leq nR$	$(\leq nR)^{\mathcal{I}} = \{x \mid \lvert\{y.(x,y) \in R^{\mathcal{I}}\}\rvert \leq n\}$
maksimalno ograničenje individualne uloge	$\geq nR$	$(\geq nR)^{\mathcal{I}} = \{x \mid \lvert\{y.(x,y) \in R^{\mathcal{I}}\}\rvert \geq n\}$
egzistencijalno ograničenje podatkovne uloge	$\exists U.D$	$(\exists U.D)^{\mathcal{I}} = \{x \mid \exists y.(x,y) \in U^{\mathcal{I}} \wedge y \in D^{\mathrm{D}}$
univerzalno ograničenje podatkovne uloge	$\forall U.D$	$(\forall U.D)^{\mathcal{I}} = \{x \mid \forall y.(x,y) \in U^{\mathcal{I}} \rightarrow y \in D^{\mathrm{D}}\}$
ograničenje vrijednosti podatkovne uloge	$U{:}\,v$	$(\forall U{:}\,v)^{\mathcal{I}} = \{x \mid (x, v^{\mathcal{I}}) \in U^{\mathcal{I}}\}$
minimalno ograničenje podatkovne uloge	$\leq nU$	$(\leq nU)^{\mathcal{I}} = \{x \mid \lvert\{y.(x,y) \in U^{\mathcal{I}}\}\rvert \leq n\}$
maksimalno ograničenje podatkovne uloge	$\geq nU$	$(\geq nU)^{\mathcal{I}} = \{x \mid \lvert\{y.(x,y) \in U^{\mathcal{I}}\}\rvert \geq n\}$
Aksiom	**Sintaksa**	**Semantika**
sadrţavanje koncepata	$C_1 \sqsubseteq C_2$	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
sadrţavanje objektnih uloga	$R_1 \sqsubseteq R_2$	$R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$
tranzitivnost objektne uloge	R^{+}	$(R^{+})^{\mathcal{I}} = (R^{\mathcal{I}})^{+}$
sadrţavanje podatkovnih uloga	$U_1 \sqsubseteq U_2$	$U_1^{\mathcal{I}} \subseteq U_2^{\mathcal{I}}$
sadrţavanje individue	$a{:}\,C$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
jednakost individua	$a = b$	$a^{\mathcal{I}} = b^{\mathcal{I}}$
različitost individua	$a \neq b$	$a^{\mathcal{I}} \neq b^{\mathcal{I}}$
postojanje klase	$\exists C$	$\lvert C^{\mathcal{I}}\rvert \geq 1$

Karakteristika OWL jezika je što koristi Uniform Resource Identifier (URI) reference za identifikaciju klasa, individua, individualnih uloga i podatkovnih uloga, RDF oznake za prikazivanje vrijednosti podataka i ugraĎene XML tipove podataka [BIRO2004]. URI reference i vrijednosti podataka se tvore od nizova znakova, gdje je svaki znak kodiran po

Unicode standardu [BECK1988] koristeći Unicode Transformation Format-8 (UTF-8) [UNIC2003] kodnu stranicu.
URI referenca služi za određivanje lokaciju na Web-u, za identifikaciju sadržaja na Web-u ili za oboje. U ontološkom opisu područnog znanja pomoću OWL DL jezika, URI referenca identifikatora se razlikuje od drugih po nizu znakova koji slijede nakon # oznake fragmenta. Niz znakova ispred # predstavlja URI referencu XML prostor naziva (engl. XML Namespace) [BRAY2006] koji će za sve identificirane elemente ontologije biti jednak. Na primjer, URI referenca

```
http://www.pmfst.hr/racsus.owl#Memorija
```
(4.1)

sastoji se od XML prostora naziva `http://www.pmfst.hr/racsus.owl#` i od identifikatora fragmenta `#Memorija`. Skraćenica XML prostora naziva se najčešće definira u korijenskom elementu XML dokumenta preko `xmlns` atributa kako bi se kasnije mogao koristiti skraćeni oblik URI referenciranja. Na primjer, korijenski element

```
<rdf:RDF xmlns="http://www.pmfst.hr/racsus.owl#"
  xml:base="http://www.pmfst.hr/racsus.owl"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:owl="http://www.w3.org/2002/07/owl#">
```
(4.2)

definira skraćenice za standardne XML prostore naziva OWL ontologije, kao što je `owl` skraćenica za `http://www.w3.org/2002/07/owl#`. Primjer (4.2) definira i prazni XML prostor naziva `xmlns="http://www.pmfst.hr/racsus.owl#"`. Ako neki skraćeni oblik URI reference ne sadrži skraćenicu XML prostora naziva, onda je njegov XML prostor naziva prazan. RDF/XML zapis ontologije, nakon definiranog osnovnog XML prostora naziva, koristi fragment URI reference za referenciranje elemenata ontologije. U donjem primjeru prikazan je segment ontologije gdje se identificirana klasa `#Memorija` opisuje koristeći referencu klase `#Tehnicka_podrska`, individualne uloge `#sluzi_za` i individue `#Pohrana_podataka`.

```
<owl:Class rdf:about="#Memorija">
  <rdfs:label>memorija</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <rdf:Description rdf:about="#Tehnicka_podrska"/>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#sluzi_za"/>
          <owl:hasValue rdf:resource="#Pohrana_podataka"/>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>
```
(4.3)

Apstraktna sintaksa za aksiom i konstruktore klase `#Memorija` iz segmenta (4.3) ontološki opisanog područnog znanja glasi

```
Class(#Memorija intersectionOf(#Tehnicka_podrska
restriction(#sluzi_za hasValue(#Pohrana_podataka))          (4.4)
```

Ako se klasa `#Memorija` označi konceptom C_1, klasa `#Tehnicka_podrska` konceptom C_2, objektna uloga `#sluzi_za` s R, a individua `#Pohrana_podataka` s o, onda aksiom i konstruktori u sintaksi opisne logike za izraz (4.4) pisan u apstraktnoj sintaksi glasi

$$C_1 \sqsubseteq C_2 \sqcap R{:}\,o \tag{4.5}$$

Za proizvoljnu interpretaciju $\mathcal{I}$ semantika izraza (4.4) glasi

$$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}} \cap \{x | (x, o^{\mathcal{I}}) \in R^{\mathcal{I}}\} \tag{4.6}$$

Kao što je u ovom odjeljku navedeno, OWL ontologija se razlikuje od opisne logike po tome što koristi RDF/XML zapis i što se zasniva na RDF-u. RDF i OWL dopuštaju korištenje *notacijskih svojstava* (engl. annotation property) kojima se pojedinim elementima ontologije i samoj ontologiji mogu dodavati vrijednosti koje ne utječu na semantiku prikazanog znanja. Prilikom ontološkog opisivanja područnog znanja koristi se ugrađeno notacijsko svojstvo `rdfs:label` za dodavanje čovjeku čitljivih verzija naziva resursa [BRIC2004], i to klasa, individua, individualnih uloga i podatkovnih uloga. Na primjer, u segmentu ontološki opisanog područnog znanja (4.3) definira se koncept identificiran s `#Memorija`, a njegov naziv je naveden uz pomoć `rdfs:label` svojstva. `rdfs:label` svojstvo dopušta korištenje XML atributa `xml:lang` [BRAY2008] čija vrijednost ukazuje na jezičnu lokalizaciju naziva elementa područnog znanja [PHIL2006], na primjer

```
<owl:Class rdf:about="#Memorija">
  <rdfs:label xml:lang="hr">memorija</rdfs:label>
  <rdfs:label xml:lang="en">memory</rdfs:label>
  ...
<owl:class>
```
(4.7)

URI referenci `#Memorija` pridruţuje različite nazive za hrvatski i engleski jezik. Ovime se omogućuje višejezično imenovanje koncepata i relacija na prirodnom jeziku. Sam naziv koncepta i relacije se koristi kao fraza rečenice kontroliranog jezika što je opisano u odjeljku 4.4.2.

4.3.2 Model područnog znanja

U ovom odjeljku se formalno definira područno znanje. Prvo se uvodi definicija elemenata područnog znanja koja naglašava simboličko označavanje elemenata. Kako bi se napravila poveznica s identifikatorima elemenata iz ontološkog opisa područnog znanja, definira se funkcija znakovnog označivanja koja svakom simbolu elementa područnog znanja pridruţuje niz znakova identifikatora iz ontologije. Ujedno se definira graf područnog znanja preko skupa uređenih trojki čiji su članovi elementi područnog znanja. Na ovakava način, graf područnog znanja oblikuje mreţu elemenata područnog znanja. Pošto je područno znanje temelj generiranja rečenica kontroliranog jezika, definiraju se jezične oznake uređenih trojki grafa područnog znanja.

Područno znanje sastoji se od koncepata koji su meĎusobno povezani relacijama. Koncepti područnog znanja dijele se na klase, individue i vrijednosti, dok se relacije dijele na individualne uloge i podatkovne uloge.
Koriste se sljedeće oznake za elemente područnog znanja:

- $k_1, k_2, \ldots$ za klase,
- $i_1, i_2 \ldots$ za individue,
- $v_1, v_2, \ldots$ za vrijednosti,
- $r_1, r_2, \ldots$ za individualne uloge,
- $p_1, p_2, \ldots$ za podatkovne uloge.

TakoĎer se uvodi oznaka $\emptyset_E$ za prazni element područnog znanja. U definiciji **4.1** koriste se navedene oznake kako bi se definirali skup elemenata područnog znanja i njegovi podskupovi.

Definicija 4.1: Neka su $E_K, E_I, E_V, E_{IU}, E_{PU}$ skupovi i $\emptyset_E$ prazni element područnog znanja. Neprazni skup E za kojeg vrijedi:

1) $E = E_K \cup E_I \cup E_V \cup E_{IU} \cup E_{PU} \cup \{\emptyset_E\}$ 2)
$E_K \cap E_I \cap E_V \cap E_{IU} \cap E_{PU} \cap \{\emptyset_E\} = \emptyset$ 3)
$|E_V| \geq 2, |E_{IU}| \geq 2$

zovemo *skupom elemenata područnog znanja.* Podskupove skupa elemenata područnog znanja E zovemo

- *Skup klasa* $E_K = \{k_1, k_2, \ldots k_{n_1}\}$ gdje je $n_1 \geq 0$,
- *Skup individua* $E_I = \{i_1, i_2, \ldots i_{n_2}\}$ gdje je $n_2 \geq 0$,
- *Skup vrijednosti* $E_V = v_1, v_2, \ldots, v_{n_3} \cup \top, \bot$ gdje je $n_3 \geq 0$,
- *Skup individualnih uloga* $E_{IU} = \{r_1, r_2, \ldots r_{n_4}\} \cup \{gen^+, ind^+\}$ gdje je $n_4 \geq 0$,
- *Skup podatkovnih uloga* $E_{PU} = p_1, p_2, \ldots p_{n_5}$ gdje je $n_5 \geq 0$,

Ovi podskupovi skupa elemenata područnog znanja E ne čine particiju od E jer skupovi E_K, E_I i E_{PU} mogu biti prazni. Skupovi E_V i E_{IU} nikad nisu prazni jer sadrţe predefinirane elemente čije se oznake razlikuju od ostalih elemenata. Individualne uloge mogu biti tranzitivne ili imati inverzne uloge, pa se uvode dva podskupa individualnih uloga:

- $E_{IU}^+ = \{r_1^+, r_2^+, \ldots r_j^+\} \cup \{gen^+, ind^+\}$ – *skup tranzitivnih individualnih uloga*,
- $E_{IU}^- = \{r_1, r_2, \ldots r_k, r_1^-, r_2^-, \ldots r_k^-\}$ – *skup inverznih individualnih uloga* koji sadrţi uloge r_i i njihove inverze r_i^-.

Tranzitivna individualna uloga moţe posjedovati i inverznu ulogu, odnosno općenito vrijedi $E_{IU}^+ \cap E_{IU}^- \neq \emptyset$, a tranzitivna i inverzna individualna uloga se označava s $r_i^\pm$. TakoĎer se definiraju dvije specijalne tranzitivne individualne uloge koje sluţe za oblikovanje taksonomije u područnom znanju:

- $gen^+ \in E_{IU}^+$ – *generalizacija* je tranzitivna individualna uloga koja ukazuje na pripadnost klase nekoj drugoj klasi.
- $ind^+ \in E_{IU}^+$ – *individualizacija* je tranzitivna individualna uloga koja ukazuje na pripadnost individue nekoj klasi.

Skup vrijednosti E_V sadrţi dvije predefinirane vrijednosti $\top$ i $\bot$ koje sluţe za odreĎivanje istinitosti podatkovne uloge.
Ako individualna uloga r ima inverznu individualnu ulogu r^- onda se piše $(r^-)^- = r$.

Definicija 4.2: Neka je E skup elemenata područnog znanja i E_{IU}^{-} skup inverznih individualnih uloga. *Uparivanje inverznih individualnih uloga* je funkcija $inv: E_{IU}^{-} \rightarrow \{\{r, r^{-}\} | r, r^{-} \in E_{IU}^{-} \wedge r \neq r^{-}\}$.

Ako su r i r^{-} međusobno inverzne uloge, onda vrijedi $inv(r) = inv(r^{-}) = \{r, r^{-}\}$. Inverz uloge r je jedini član skupa $inv(r) \backslash \{r\}$.

Definicija 4.3: *Model elemenata područnog znanja* je uređeni par $M_E = (E, inv)$ gdje je E skup elemenata područnog znanja i inv uparivanje inverznih individualnih uloga.

Prije definicije formalnog modela područnog znanja, potrebno je uvesti znakove i nizove znakova koji će služiti za označavanje elemenata područnog znanja. Znakovi koji se koriste za označavanje elemenata područnog znanja su kodirani po Unicode standardu [BECK1988].
Z je *skup znakova* čiji je kodni broj po Unicode standardu element skupa {U+0021, …, U+FFFC}. Kao podskup skupa Z definira se $Z_{AN} = \{a, b, \ldots, z, A, B, \ldots, Z, 0, \ldots, 9, \_\}$ *skup alfanumeričkih znakova* koji uključuje znakove iz segmenata:

- {U+0061, …, U+007A} (mala slova engleske abecede),
- {U+0041, …, U+005A} (velika slova engleske abecede),
- {U+0030, …, U+0039} (znamenke dekadskog brojevnog sustava) i
- {U+005F} (povlaka).

Niz duljine n nad skupom Ω je uređena n-torka $W = (w_1, w_2, \ldots, w_n)$ gdje su $w_i \in \Omega$. Iz praktičnih razloga, niz W će se označavati s

$$W = w_1 w_2 \ldots w_n \tag{4.8}$$

Ako je Ω skup znakova onda W zovemo *nizom znakova*. *Prazni niz znakova* ima duljinu 0 i označava se s $\emptyset_{NZ}$.
U formalnom modelu područnog znanja definirana su dva skupa nizova znakova:

- $NZ_{ID} = \{\#z_1 z_2 \ldots z_n | z_i \in Z_{AN}, n > 0\}$ – *skup identifikatora* je skup svih nizova znakova nad skupom alfanumeričkih znakova čiji je prvi znak #.
- $NZ_{VP} = \{z_1 z_2 \ldots z_n | z_i \in Z, z_1 \neq \#, n > 0\}$ – *skup vrijednosti podataka* je skup svih nizova znakova nad skupom znakova koji ne započinju znakom #.

Skup identifikatora zahtjeva da prvi znak niza znakova bude # kako bi se niz znakova iz NZ_{ID} razlikovala od niza znakova iz NZ_{VP} . Na primjer, niz znakova #Aritmeticko_logicka_jedinica je primjer identifikatora, dok je niz znakova centralna procesorska jedinica primjer vrijednosti. centralna procesorska jedinica ne može biti identifikator jer prvi znak nije # i sadrži znak razmaka koji nije iz skupa Z_{AN}. Svakom elementu područnog znanja se pridružuje jedinstveni niz znakova. Slijedi definicija funkcije koja vrši navedeno pridruživanje.

Definicija 4.4: Neka je E skup elemenata područnog znanja, $E_V \subseteq E$ skup vrijednosti, NZ_{ID} skup identifikatora, NZ_{VP} skup vrijednosti podataka i $\emptyset_{NZ}$ prazni niz znakova. Bilo koja injekcija $zo: E \rightarrow NZ_{ID} \cup NZ_{VP} \cup \{\emptyset_{NZ}\}$ za koju vrijedi:

1) $zo(E_V) \subseteq NZ_{VP}, zo(E \backslash E_{VP}) \subseteq NZ_{ID}, zo\ \emptyset_E = \emptyset_{NZ}$
2) $zo(\top) = \top, zo(\bot) = \bot$
3) $zo(gen^{+}) = \#\#gen, zo(ind^{+}) = \#\#ind$

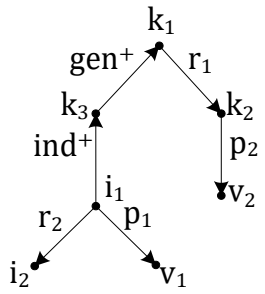
naziva se *znakovno označavanje elemenata područnog znanja*.

Funkcija znakovnog označavanja elemenata područnog znanja preslikava vrijednosti područnog znanja u *vrijednosti podataka*, a klase, individue, individualne uloge i podatkovne uloge područnog znanja preslikava u *identifikatore*. Nadalje, vrijednosti $\top$ i $\bot$ preslikava u ekvivalentne Unicode znakove, a generalizaciju i individualizaciju preslikava u identifikatore $\#\#gen$ i $\#\#ind$.
Područno znanje povezuje koncepte, individue i vrijednosti pomoći individualnih uloga i podatkovnih uloga. Povezivanje se opisuje ureĎenim trojkama grafa područnog znanja kako slijedi u definiciji **4.5**.

Definicija 4.5: Neka je $M_E = (E, inv)$ model elemenata područnog znanja nad skupom elemenata područnog znanja E. *Graf područnog znanja* GPZ_E nad modelom područnog znanja M_E je unija skupova:

1) $\{(x, y, z) \mid x, z \in E_K, y \in E_{IU} \backslash \{ind^+\}\}$
2) $x, y, z \quad x \in E_I, y \in E_{IU} \backslash\ gen^+\ , z \in E_I\}$
3) $x, y, z \quad x \in E_K \cup E_I, y \in E_{IU} \backslash \{gen^+, ind^+\}, z = E_I \cup \{\emptyset_E\}\}$
4) $x, y, z \quad x \in E_K \cup E_I, y \in E_{PU}, z \in E_V \cup \{\emptyset_E\}\}$

Element grafa područnog znanja $(x, y, z) \in GPZ_E$ zove se *trojka grafa područnog znanja* i vrijedi $x \in E_K \cup E_I$, $y \in E_{IU} \cup E_{PU}$, $z \in E_K \cup E_I \cup E_V \cup \{\emptyset_E\}$. Element x zovemo *subjektni koncept*, y *predikatnom relacijom*, a z *objektni koncept* trojke grafa područnog znanja (x, y, z). Skup 1) grafa područnog znanja iz definicije **4.5** kaţe da individualizacijom ne mogu biti povezane dvije klase. Drugi skup pod oznakom 2) govori o tome da generalizacija ne moţe povezivati individuu s drugom individuom. Skup 3) klasu ili individuu povezuje s individuom ili praznim elementom preko individualne uloge. Podatkovna uloga se u skupu 4) koristi za povezivanje klase ili individue s vrijednosti ili praznim elementom.
Slika 4.7 prikazuje grafičku notaciju grafa područnog znanja $\{(k_1, r_1, k_2), (k_3, gen^+, k_1), (i_1, ind^+, k_3), (i_1, p_1, v_1), (i_1, r_2, i_2), (k_2, p_2, v_2)\}$ gdje su $k_1, k_2 \in E_K$, $i_1, i_2 \in E_I$, $v_1, v_2 \in E_V$, $r_1, r_2 \in E_{IU}$ i $p_1, p_2 \in E_{PU}$.



Slika 4.7. Primjer grafa područnog znanja

U grafičkoj notaciji, svaka (x, y, z) trojka grafa područnog znanja se prikazuje kao dvije točke x i z povezani strelicom y.
Putanja grafa područnog znanja GPZ_E je svaki niz trojki $(x_1, y_1, z_1)(x_2, y_2, z_2) \ldots (x_n, y_n, z_n)$ za kojeg vrijedi $x_{i+1} = z_i$ za $i \in \{1,2, \ldots, n-1\}$.
Na slici 4.7 primjer putanje je $(i_1, ind^+, k_3)(k_3, gen^+, k_1)$.
Pošto se nad područnim znanjem vrši generiranje kontroliranog jezika, kao što je opisano u pododjeljku 4.7.2.2, elementi područnog znanja su dodatno označeni oznakama koje imaju svoju ulogu kod obrade kontroliranog jezika.

Definicija 4.6: Neka je $\mathbb{N}$ skup prirodnih brojeva. *Skup jezičnih oznaka* $JO = \{\wedge,\vee\} \times \{\forall,\exists\} \times \{\top,\bot\} \times (\ \forall,\exists\ \cup (\{=,\leq,\geq\} \times \mathbb{N}))$ je skup uređenih četvorki (l,s,n,o) gdje je:

1) $l \in \{\wedge,\vee\}$ jezična oznaka veznika,
2) $s \in\ \forall,\exists$ jezična oznaka broja,
3) $n \in \{\neg,\lrcorner\}$ jezična oznaka negacije,
4) $o \in \{\forall,\exists\} \cup (\{=,\leq,\geq\} \times \mathbb{N})$ proširena jezična oznaka broja.

Članovi uređene četvorke iz skupa jezičnih oznaka su simboli namijenjeni jezičnom označavanju trojki grafa područnog znanja. Simboli $\wedge$ i $\vee$ zovu se *konjunkcija* i *disjunkcija*. Ako je jezična oznaka negacije simbol $\neg$, onda je trojka grafa područnog znanja *negirana*, inače za simbol $\lrcorner$ *nije negirana*. Jezična oznaka broja $\forall$ je oznaka *univerzalne kvantifikacije*, dok je $\exists$ oznaka *egzistencijalne kvantifikacije*. Proširena jezična oznaka moţe moţe biti predstavljena uređenim parom simbola iz skupa $\{=,\leq,\geq\} \times \mathbb{N}$, gdje je $\mathbb{N}$ skup prirodnih brojeva. Uređeni parovi simbola $(=,n)$, $(\leq,n)$ i $(\geq,n)$ označavaju *kardinalnost n*, *maksimalnu kardinalnost n* i *minimalnu kardinalnost n*.
Slijedi definicija relacije koja grafu područnog znanja pridruţuje jezične oznake.

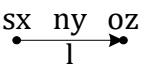
Definicija 4.7: Neka je JO skup jezičnih oznaka. *Jezično označavanje* nad grafom područnog znanja GPZ je binarna relacija $jo \subseteq GPZ_E \times JO$ za koju vrijedi

1) $\forall(x,y,z) \in GPZ_E\ \exists(l,s,n,o) \in JO$ tako da je $((x,y,z),(l,s,n,o)) \in jo$
2) $((x,y,\emptyset_E),(l,s,n,o)) \in jo$ ako i samo ako je o $\in \{=,\leq,\geq\} \times \mathbb{N}$

Pravilom 1) se zahtjeva da sve trojke grafa područnog znanja imaju jezičnu oznaku, dok obrnuto ne vrijedi. Pravilo 2) ograničava jezičnu oznaku mnogostrukosti objekta za trojke grafa područnog znanja koje imaju prazni element.
Jezična oznaka trojke grafa područnog znanja $((x,y,z),(l,s,n,o))$ skraćeno se piše

$$l[sx,ny,oz] \tag{4.9}$$

Slika 4.8 prikazuje grafičku notaciju izraza iz (4.9).



Slika 4.8. Jezična oznaka trojke grafa područnog znanja

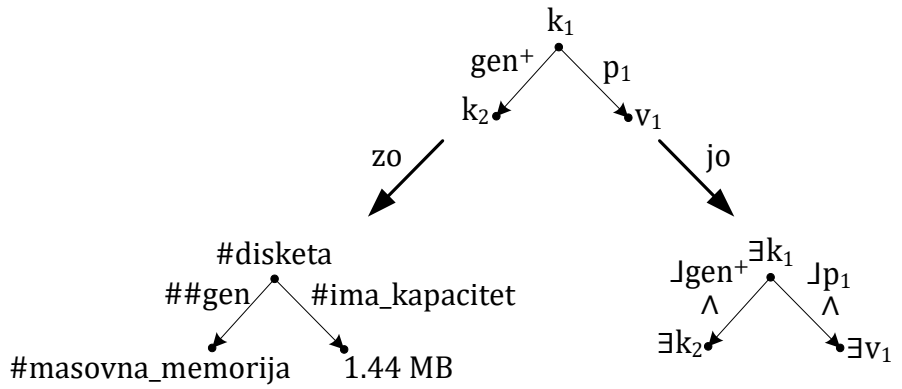
Na primjer, $\wedge[\exists x,\lrcorner y,\exists z]$ je označena konjunkcijom, ne negacijom i egzistencijalnom kvantifikacijom subjekta i objekta. Iz praktičnih razloga skraćeno se piše:

- $l[x,ny,oz]$ ako je trojka označena egzistencijalnom kvantifikacijom subjekta,
- $l[sx,ny,z]$ ako je trojka označena egzistencijalnom kvantifikacijom objekta,
- $[sx,ny,oz]$ ako je trojka označena konjunkcijom,
- $l[sx,y,nz]$ ako trojka nije označena negacijom.

Na primjer, označena trojka grafa područnog znanja $[x,y,z]$ ima jezičnu oznaku konjunkcije, ne negacije i egzistencijalne kvantifikacije subjekta i objekta, dok je $[\forall x,\neg y,\leq 3]$ jezično označena konjunkcijom, univerzalnom kvantifikacijom subjekta, negacijom predikata i maksimalnom kardinalnošću 3 objekta.
Uvedena funkcija označavanja nad skupom elemenata područnog znanja i relacija jezičnog označavanja nad grafom područnog znanja, koji je definiran nad istim skupom elemenata područnog znanja, sluţe za formalnu definiciju područnog znanja.

Definicija 4.8: Neka je $M_E = (E, inv)$ model elemenata područnog znanja, NZ_{ID} skup identifikatora i NZ_{VP} skup vrijednosti podataka. *Područno znanje* PZ nad modelom elemenata M_E je uređena četvorka (M_E, GPZ_E, zo, jo) gdje je GPZ_E graf područnog znanja nad M_E, $zo: E \rightarrow NZ_{ID} \cup NZ_{VP} \cup \{\emptyset_{NZ}\}$ znakovno označavanje elemenata područnog znanja i $jo \subseteq PZ_G \times JO$ jezično označavanje.

Ovom definicijom se grafu područnog znanja pridruţuju oznake koje identificiraju koncepte, individue, individualne uloge i podatkovne uloge, te navode vrijednosti podataka.
Slika 4.9 prikazuje primjer grafa područnog znanja na kojemu je primijenjeno znakovno i jezično označavanje elemenata područnog znanja.



Slika 4.9. Znakovno i jezično označavanje grafa područnog znanja

Skup elemenata područnog znanja E za primjer područnog znanja sa slike 4.9 sastoji se od skupa klasa $E_K = \{k_1, k_2\}$, skupa vrijednosti $E_V = \{v_1\}$, skupa individualnih uloga $E_{IU} = \{gen^+, ind^+\}$, skupa podatkovnih uloga $E_{PU} = \{p_1\}$ i praznog skupa individua $E_I = \emptyset$. Graf područnog znanja nad modelom elemenata područnog znanja $M_E = (E, inv)$ ima sljedeće trojke $GPZ_E = \{\ k_1, gen^+, k_2\ , (k_1, p_1, v_1)\}$. Funkcija znakovnog označavanja za primjer područnog znanja sa slike 4.9 preslikava elemente područnog znanja u nizove znakova i to $zo(k_1) =$ #disketa, $zo(k_2) =$ #masovna_memorija, $zo(p_1) =$ #ima_kapacitet i $zo(v_1) =$ 1.44 MB . Na kraju, relacija jezičnog označavanja jo posjeduje sljedeće elemente $\{\wedge [\exists k_1, \lrcorner gen^+, \exists k_2], \wedge [\exists k_1, \lrcorner p_1, \exists v_1]\}$ što se skraćeno moţe pisati $jo = \{[k_1, gen^+, k_2], [k_1, p_1, v_1]\}$.

Ovakav model područnog znanja je temelj ostalih modela skupova podataka. U modelu nastavnog sadrţaja se definiraju objekti nastavnog sadrţaja opisani nizom elemenata područnog znanja.

4.3.3 Model nastavnog sadrţaja

Osnovni graĎevni element nastavnog sadrţaja je koncept područnog znanja. Koncepti se unutar nastavnog sadrţaja grupiraju i ujedno definiraju podskup grafa područnog znanja.

Definicija 4.9: Neka je PZ područno znanje nad skupom elemenata područnog znanja E koji sadrţi skup klasa E_K i skup individua E_I. *Objekt nastavnog sadržaja* nad područnim znanjem PZ je niz klasa i individua $\sigma = x_1 x_2 \ldots x_n$ gdje su $x_i \in E_K \cup E_I$ i $x_i \neq x_j$ za $1 \leq i \neq j \leq n$.

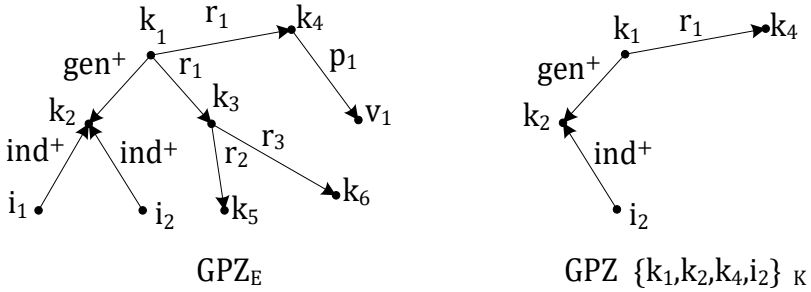
Klase i individue koje čine niz u objektu nastavnog sadrţaja su meĎusobno različite, odnosno dvije klase ili individue se ne ponavljaju. Postoje dvije vrste nastavnog sadrţaja. Sa $\sigma = x_1 x_2 \dots x_n$ se označava *nastavni sadržaj namijenjen učenju*, dok je $\sigma' = x_1 x_2 \dots x_n$ oznaka za *objekt nastavnog sadržaja namijenjen testiranju*.

Definicija 4.10: *Nastavni sadržaj NS* nad područnim znanjem *PZ* je niz objekata nastavnog sadrţaja $\sigma_1 \sigma_1' \sigma_2 \sigma_2' \dots \sigma_m \sigma_m'$ gdje je $\sigma_i = \sigma_i' = x_1^i x_2^i \dots x_{n_i}^i$.

Po ovoj definiciji nakon svakog objekta nastavnog sadrţaja namijenjenog učenju σ_i slijedi objekt nastavnog sadrţaja namijenjen testiranju σ_i' čiji je niz klasa i individua jednak nizu klasa i individua od σ_i.
Objekt nastavnog sadrţaja sadrţi klase i individue koje čine restrikciju grafa područnog znanja. Restrikcija grafa područnog znanja se definira nad nekim skupom koncepata (klasa, individua, vrijednosti) ili nad skupom relacija (individualnih uloga, podatkovnih uloga).

Definicija 4.11: Neka je GPZ_E graf područnog znanja nad modelom elemenata područnog znanja $M_E = (E, inv)$ i neka je $\{x_1, x_2, \dots, x_n\} \subseteq E_K \cup E_I \cup E_V$ proizvoljan skup koncepata. *Konceptna restrikcija grafa područnog znanja* nad skupom koncepata $\{x_1, x_2, \dots, x_n\}$ je skup $GPZ\ \{x_1, x_2, \dots, x_n\}_K = \{(x, y, z) \in GPZ_E \mid x, z \in \{x_1, x_2, \dots, x_n\}\}$.

Konceptnom restrikcijom se ograničava graf područnog znanja na odreĎeni podskup elemenata područnog znanja. Slika 4.10 prikazuje grafičku notaciju grafa područnog znanja i njegove konceptne restrikcije na skup $\{k_1, k_2, k_4, i_2\}$.

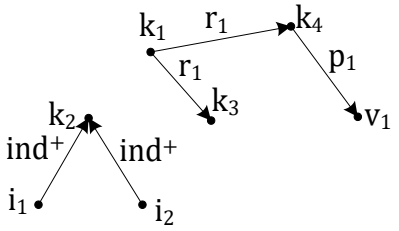


Slika 4.10. Primjer konceptne restrikcije grafa područnog znanja

Slično kao konceptna restrikcija, definira se i relacijska restrikcija grafa područnog znanja.

Definicija 4.12: Neka je GPZ_E graf područnog znanja nad modelom elemenata područnog znanja M_E i $\{x_1, x_2, \dots, x_n\} \subseteq E_{IU} \cup E_{PU}$. *Relacijska restrikcija grafa područnog znanja* nad skupom relacija $\{x_1, x_2, \dots, x_n\}$ je skup $GPZ\ \{x_1, x_2, \dots, x_n\}_R = \{(x, y, z) \in PZ_G \mid y \in \{x_1, x_2, \dots, x_n\}\}$.

Relacijskom restrikcijom se odreĎuje drugi član trojke područnog grafa. Slika 4.11 prikazuje relacijsku restrikciju grafa područnog znanja iz primjera sa slike 4.10 na skup $\{r_1, ind^+, p_1\}$.



$\mathrm{GPZ}\ \{r_1, \mathrm{ind}^+, p_1\}_R$

Slika 4.11. Primjer relacijske restrikcije grafa područnog znanja

Neka je GPZ_E graf područnog znanja i $k \in E_K \cup E_I \cup E_V$ koncept područnog znanja. Definiraju se sljedeće okoline koncepta *k*

- *donja okolina* od k je $\varepsilon^{\downarrow}(k) = \{k_i \in E_K \cup E_I \cup E_V | (k, y, k_i) \in GPZ_E\}$
- *gornja okolina* od k je $\varepsilon^{\uparrow}(k) = \{k_i \in E_K \cup E_I | (k_i, y, k) \in GPZ_E\}$
- okolina od k je $\varepsilon(k) = \varepsilon^{\downarrow}(k) \cup \varepsilon^{\uparrow}(k)$

Okolina koncepta je skup koncepata koji su direktno vezani s danim konceptom. Tablica 4.4 prikazuje okoline koncepata iz primjera grafa područnog znanja sa slike 4.10.

Tablica 4.4. Okoline koncepata

$\boldsymbol{k}$	$\boldsymbol{\varepsilon^{\downarrow}(k)}$	$\boldsymbol{\varepsilon^{\uparrow}(k)}$	$\boldsymbol{\varepsilon(k)}$
k_1	$\{k_2, k_3, k_4\}$	$\emptyset$	$\{k_2, k_3, k_4\}$
k_2	$\emptyset$	$\{k_1, i_1, i_2\}$	$\{k_1, i_1, i_2\}$
k_3	$\{k_5, k_6\}$	$\{k_1\}$	$\{k_1, k_5, k_6\}$
k_4	$\{v_1\}$	$\{k_1\}$	$\{k_1, v_1\}$
k_5	$\emptyset$	$\{k_3\}$	$\{k_3\}$
k_6	$\emptyset$	$\{k_3\}$	$\{k_3\}$
i_1	$\{k_2\}$	$\emptyset$	$\{k_2\}$
i_2	$\{k_2\}$	$\emptyset$	$\{k_2\}$
v_1	$\emptyset$	$\{k_4\}$	$\{k_4\}$

Okolina koncepta se koristi kod definicije restrikcije grafa područnog znanja nad objektom nastavnog sadrţaja.

Definicija 4.13: *Restrikcija grafa područnog znanja nad objektom nastavnog sadržaja* $\sigma = x_1 x_2 \ldots x_n$ je $GPZ\ \sigma = GPZ\ \{x_1, x_2, \ldots, x_n\} \cup \bigcup_{i=1}^{n} \varepsilon(x_i)_K$.

Po definiciji **4.13** restrikcija nad objektom nastavnog sadrţaja proširuje konceptnu restrikciju nad skupom članova niza objekta nastavnog sadrţaja s okolinom svakog od člana. Neka je $\sigma = k_1 k_2 k_3$ objekt nastavnog sadrţaja gdje su k_1, k_2 i k_3 koncepti iz primjera grafa područnog znanja sa slike 4.10. Tada je $GPZ\ \sigma = GPZ\ \{k_1, k_2, k_3\} \cup \varepsilon(k_1) \cup \varepsilon(k_2) \cup \varepsilon(k_3)_K = GPZ\ \{k_1, k_2, k_3\} \cup \{k_2, k_3, k_4\} \cup \{k_1, i_1, i_2\} \cup \{k_5, k_6\}_K =$
$GPZ\ \{k_1, k_2, k_3, k_4, k_5, k_6, i_1, i_2\}_K$.

Model nastavnog sadrţaja opisan u ovom odjeljku se sastoji od objekata nastavnog sadrţaja koji sadrţe koncepte područnog znanja. Nad konceptima i objektima se, u sljedećem odjeljku, definira funkcija pridruţivanja teţinske vrijednosti kako bi se odredila vaţnost koncepta, odnosno objekta u modelu učenika. Nadalje, uvodi se stanje učenja i testiranja kojim će se pratiti napredak učenika tijekom učenja i testiranja znanja.

4.3.4 Model učenika

Model učenika je izlazni skup podataka modeliranja učenika. Proces modeliranja učenika prati stanje učenja i testiranja učenika te vrednuje znanje učenika. Model učenika se definira nad konceptima područnog znanja i objektima nastavnog sadrţaja, a prije definicije samog modela, uvode se funkcije koje sluţe za vrednovanje koncepata i objekata nastavnog sadrţaja.

Definicija 4.14: Neka je $\sigma = x_1 x_2 \dots x_n$ objekt nastavnog sadrţaja nad područnim znanjem PZ i $PZ_G\ \sigma_K$ konceptna restrikcija grafa područnog znanja na objekt nastavnog sadrţaja σ. Funkcija $t_\sigma: \{x_1, x_2, \dots, x_n\} \to \mathbb{R}^+$ zove se *pridruţivanje teţinske vrijednosti objekta nastavnog sadrţaja* σ.

Funkcija teţinske vrijednosti t_σ svakom konceptu koji pripada objektu nastavnog sadrţaja σ pridruţuje pozitivni realan broj koji se zove *teţinska vrijednost koncepta*. OdreĎivanje teţinske vrijednosti se provodi nad restrikcijom grafa područnog znanja na objekt nastavnog sadrţaja gdje teţinska vrijednost koncepta ovisi o poloţaju i povezanosti koncepta s ostalim konceptima unutar restrikcije grafa. Postupak odreĎivanja funkcije teţinske vrijednosti objekta nastavnog sadrţaja je opisan u odjeljku 4.6.3.

Definicija 4.15: Neka je $NS = \sigma_1 \sigma_1' \sigma_2 \sigma_2' \dots \sigma_m \sigma_m'$ nastavni sadrţaj, $\sigma_i = x_1^i x_2^i \dots x_{n_i}^i$ i $t_{\sigma_i} = t_{\sigma_i'}$ pridruţivanja teţinskih vrijednosti objekta nastavnog sadrţaja. Funkcija $nt_{NS}: \{\ \sigma_i, x_j^i\ | NS = \sigma_1 \sigma_1' \sigma_2 \sigma_2' \dots \sigma_m \sigma_m' \wedge \sigma_i = x_1^i x_2^i \dots x_{h_i}^i\} \to\ 0,1]$ definirana s $nt_{NS}(\sigma_i, x_j^i) = nt_{NS}(\sigma_i', x_j^i) = \frac{t_{\sigma_i}(x_j^i)}{max\ (\{t_{\sigma_i}(x_k^i) | k \in \{1,\dots,n_i\}\})}$ zove se *normiranje teţinske vrijednosti nastavnog sadrţaja* NS.

Domena normiranja teţinske vrijednosti je podskup kartezijevog produkta skupa svih objekata nastavnog sadrţaja i svih koncepata koji čine te objekte nastavnog sadrţaja. Ako je koncept x_i dio objekta nastavnog sadrţaja σ, onda ureĎeni par (σ, x_i) je element domene normiranja teţinske vrijednosti, a realan broj $nt_{NS}(\sigma, x_i)$ zove se *normirana teţinska vrijednost*. U nastavnom sadrţaju, koji je predstavljen nizom objekata nastavnog sadrţaja $\sigma_1 \sigma_1' \sigma_2 \sigma_2' \dots \sigma_m \sigma_m'$ očito vrijedi $t_{\sigma_i} = t_{\sigma_i'}$, jer objekt nastavnog sadrţaja namijenjen učenju σ_i je jednak objektu nastavnog sadrţaja namijenjen testiranju σ_i'. Kao posljedica vrijedi $nt_{NS}(\sigma_i, x_j^i) = nt_{NS}(\sigma_i', x_j^i)$. Tablica 4.5 prikazuje odnos teţinske vrijednosti i normirane teţinske vrijednosti za primjer nastavnog sadrţaja $\sigma_1 \sigma_1' \sigma_2 \sigma_2'$ gdje su $\sigma_1 = \sigma_1' = x_1^1 x_2^1 x_3^1$ i $\sigma_2 = \sigma_2' = x_1^2 x_2^2$.

Tablica 4.5. Primjer pridruţivanja teţinske vrijednosti konceptima i objektima nastavnog sadrţaja

σ	x	$t_\sigma(x)$	$nt_{NS}(\sigma, x)$
σ_1	x_1^1	0.1	$\frac{0.1}{0.5} = 0.2$
	x_2^1	0.5	$\frac{0.5}{0.5} = 1$
	x_3^1	0.2	$\frac{0.2}{0.5} = 0.4$
σ_1'	x_1^1	0.1	0.2
	x_2^1	0.5	1
	x_3^1	0.2	0.4
σ_2	x_2^1	0.8	$\frac{0.8}{0.8} = 1$
	x_2^2	0.2	$\frac{0.2}{0.8} = 0.25$
σ_2'	x_1^2	0.8	1
	x_2^2	0.2	0.25

Definicija **4.16** opisuje *stanje koncepta* unutar objekta nastavnog sadrţaja namijenjenog učenju i unutar objekta nastavnog sadrţaja namijenjenog testiranju.

Definicija 4.16: Neka je $\sigma = x_1 x_2 \dots x_n$ objekt nastavnog sadrţaja nad područnim znanjem PZ, $\mathbb{N}$ skup prirodnih brojeva i $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. Funkcija $su_\sigma : \{x_1, x_2, \dots, x_n\} \to \mathbb{N}_0$ zove se *pridruživanje stanja učenja objekta nastavnog sadržaja* σ, a funkcija $st_\sigma : \{x_1, x_2, \dots, x_n\} \to 2^{\mathbb{N} \times [0,1]}$ zove se *pridruživanje stanja testiranja objekta nastavnog sadržaja* σ.

Broj $su_\sigma(x_i)$ zove se *stanje učenja koncepta* x_i i on govori koliko puta je učenik u nastavnom sadrţaju namijenjenog učenju σ *pristupio* konceptu x_i. *Stanje testiranja koncepta* x_i je skup ureĎenih dvojki (p, q) gdje je p *identifikator pristupa*, a q *uspješnost pristupa* koncepta. Ako je stanje testiranja nekog koncepta skup {(1,0.4),(2,0.7)}, onda je učenik pristupio konceptu 2 puta. Kod prvog pristupa je uspješnost pristupa 0.4, a kod drugog testiranja uspješnost pristupa iznosi 0.7. Ako je stanje učenja 0, odnosno ako je stanje testiranja prazan skup, onda učenik nikada nije pristupio učenju, odnosno testiranju nad odreĎenim konceptom.

Definicija 4.17: Neka je $NS = \sigma_1 \sigma_1' \sigma_2 \sigma_2' \dots \sigma_m \sigma_m'$ nastavni sadrţaj. Funkcija $su_{NS} = \bigcup_{i=1}^{m} su_{\sigma_i} : \{\sigma_1, \sigma_2, \dots, \sigma_m\} \to [0,1]$ definirana sa $su_{NS}(\sigma_i) = \frac{|\{x_j^i | j \in \{1,2,\dots,n_i\} \wedge su_{\sigma_i}(x_j^i) > 0\}|}{n_i}$, gdje je $\sigma_i = x_1^i x_2^i \dots x_{n_i}^i$, zove se *pridruživanje stanja učenja nastavnog sadržaja* NS. Funkcija $st_{NS} = \bigcup_{i=1}^{m} st_{\sigma_i'} : \{\sigma_1', \sigma_2', \dots, \sigma_m'\} \to [0,1]$ definirana s $st_{NS}(\sigma_i') = \frac{|\{x_j^i | j \in \{1,2,\dots,n_i\} \wedge |st_{\sigma_i}(x_j^i)| > 0\}|}{n_i}$, gdje je $\sigma_i' = x_1^i x_2^i \dots x_{n_i}^i$, zove se *pridruživanje stanja testiranja nastavnog sadržaja* NS.

Po definiciji **4.17** stanje učenja nastavnog sadrţaja za objekt nastavnog sadrţaja je omjer broja pozitivnih stanja koncepata i ukupnog broja koncepata. Tablica 4.6 prikazuje stanja učenja i testiranja za primjer nastavnog sadrţaja iz tablice 4.5.

Tablica 4.6. Primjer pridruţivanja stanja učenja i testiranja konceptima i objektima nastavnog sadrţaja

σ	x	stanje koncepta	stanje objekta nastavnog sadrţaja
σ_1	x_1^1	1	$su_{NS}(\sigma_1) = \frac{2}{3} = 0.66$
	x_2^1	2	
	x_3^1	0	
σ^1	x_1^1	{(1,0.3),(2,0.3)}	$st_{NS}(\sigma^1) = \frac{1}{3} = 0.33$
	x_2^1	∅	
	x_3^1	∅	
σ_2	x_1^2	1	$su_{NS}(\sigma_2) = \frac{2}{2} = 1$
	x_2^2	1	
$\sigma_2^{'}$	x_1^2	{(1,0.5)}	$st_{NS}(\sigma_2^{'}) = \frac{2}{2} = 1$
	x_2^2	{(1,0.4),(2,0.5)}	

Svakom konceptu i svakom objektu nastavnog sadrţaja namijenjenog testiranju, s obzirom na stanje testiranja, se pridruţuje broj koji opisuje ocjenu koncepta, odnosno objekta nastavnog sadrţaja namijenjenog testiranju.

Definicija 4.18: Neka je $\sigma^{'} = x_1 x_2 \dots x_n$ objekt nastavnog sadrţaja namijenjen testiranju nad područnim znanjem PZ, $t_{\sigma'}$ pridruţivanje teţinske vrijednosti i $st_{\sigma'}$ pridruţivanje stanja testiranja objekta nastavnog sadrţaja $\sigma^{'}$. Funkcija $oc_{\sigma'}: \{x_1, x_2, \dots, x_n\} \to [0,1]$ definirana s $oc_{\sigma'}(x_i) = \begin{matrix} nt_{NS}(\sigma^{'}, x_i) \frac{\sum_{(p,q)\in st_{\sigma'}(x_i)} q}{|st_{\sigma'}(x_i)|} za\ st_{\sigma'}(x_i) \neq \emptyset \\ 0\ za\ st_{\sigma'}(x_i) = \emptyset \end{matrix}$ zove se *ocjenjivanje objekta nastavnog sadržaja* $\sigma^{'}$.

Ocjenjivanje objekta nastavnog sadrţaja namijenjenog testiranju $\sigma^{'}$ svakom konceptu x_i pridruţuje *ocjenu koncepta* $oc_{\sigma'}(x_i)$. Ocjena koncepta će biti nula ako je stanje testiranja $st_{\sigma'}(x_i)$ prazan skup ili ako je suma uspješnosti pristupa $\sum_{(p,q)\in st_{\sigma'}(x_i)} q = 0$.

Definicija 4.19: Neka je $NS = \sigma_1 \sigma_1^{'} \sigma_2 \sigma_2^{'} \dots \sigma_m \sigma_m^{'}$ nastavni sadrţaj. *Ocjenjivanje nastavnog sadržaja NS* je funkcija $oc_{NS} = \sum_{i=1}^{m} oc_{\sigma_i'}: \{\sigma_1^{'}, \sigma_2^{'}, \dots, \sigma_m^{'}\} \to [0,1]$ definirana s $oc_{NS}(\sigma_i^{'}) = \frac{\sum_{j=1}^{n_i} oc_{\sigma_i'}(x_j^i)}{n_i}$

Primjer ocjena i ocjenjivanja koncepata i objekata nastavnog sadrţaja za primjer nastavnog sadrţaja iz tablice 4.5 je prikazan u tablici 4.7.

Tablica 4.7. Primjer ocjenjivanja koncepata i objekata nastavnog sadrţaja

σ	x	$nt_{NS}(\sigma, x)$	stanje koncepta	$oc_{\sigma}(x)$	$oc_{NS}(\sigma)$
σ_1	x_1^1	0.2	1		
	x_2^1	1	2		
	x_3^1	0.4	0		
$\sigma_1^{'}$	x_1^1	0.2	{(1,0.3),(2,0.3)}	$0.2\frac{0.3+0.3}{2}=0.06$	$\frac{0.06+0+0}{3}=0.02$
	x_2^1	1	∅	0	
	x_3^1	0.4	∅	0	
σ_2	x_1^2	1	1		
	x_2^2	0.25	1		
$\sigma_2^{'}$	x_1^2	1	{(1,0.5)}	$\frac{0.5}{1}=0.5$	$\frac{0.5+0.1125}{2}=0.30625$
	x_2^2	0.25	{(1,0.4),(2,0.5)}	$0.25\frac{0.4+0.5}{2}=0.1125$	

Na osnovi definirane funkcije za pridruţivanje teţinske vrijednosti, normirane teţinske vrijednosti, funkcije za pridruţivanja stanja učenja i testiranja i funkcije ocjenjivanja, definira se model učenika.

Definicija 4.20: *Model učenika nad nastavnim sadržajem* $NS = \sigma_1\sigma_1^{'}\sigma_2\sigma_2^{'}\ldots\sigma_m\sigma_m^{'}$ je uređena petorka $MU_{NS} = (NS, nt_{NS}, su_{NS}, st_{NS}, oc_{NS})$ gdje je nt_{NS} normiranje teţinske vrijednosti, su_{NS} pridruţivanje stanja učenja, st_{NS} pridruţivanje stanja testiranja i oc_{NS} ocjenjivanje nastavnog sadrţaja.

Na kraju, *ocjena modela učenika* je aritmetička sredina ocjena objekata nastavnog sadrţaja koja, za primjer iz tablice 4.7, iznosi

$$\frac{0.02+0.30625}{2}=0.163125 \tag{4.10}$$

Model učenika je definiran na osnovi nastavnog sadrţaja, a nastavni sadrţaj je oblikovan nad područnim znanjem. Navedeni skupovi podataka su u međusobnoj zavisnosti s ontološkim opisom područnog znanja. U sljedećem podpoglavlju se opisuje skup podataka koji sluţi za komunikaciju s učenikom na kontroliranom jeziku.

4.4 Kontrolirani jezik

Izjave kontroliranog jezik CoLaB Tutor-a su opisane rečenicama koje se sastoje od fraza, a fraze od riječi. Sintaksa rečenice se temelji na područnom znanju, a fraze su nazivi koncepata i relacija područnog znanja. Fraza u područnom znanju nije specificirana već je opisana nizom nespecificiranih riječi. Kako bi se odredila karakteristike riječi, a time i fraze, koristi se morfološki leksikon hrvatskih riječi. Dodatno se glagolske fraze specificiraju uz pomoć valencijskog leksikona hrvatskih glagola. Model kontroliranog jezika se na nivou riječi oslanja na specifikaciju morfoloških oblika riječi. Pri tome se uvode parametrizirani podskupovi morfološkog leksikona i definiraju se relacije uređaja kako bi se mogle usporediti pojedini oblici riječi, što predstavlja temelj za prepoznavanje kontroliranog jezika. Parametrizacija i uspoređivanje se uvodi i na nivou fraza, a gramatika rečenica kontroliranog jezika je opisana kontekstno neovisnom gramatikom.

4.4.1 Morfološki leksikon

Hrvatski standardni jezik moţe biti morfološki specificiran MULTEXT-East normom, kojom se svakom obliku riječi pridruţuje *morfosintatički opis*. Koristeći MULTEXT-East normu, Hrvatski morfološki leksikon (HML) je strukturiran kao skup ureĎenih trojki gdje svaka trojka ima oblik riječi, osnovni oblik riječi, odnosno lemu, i morfosintaktički opisa oblika riječi. Na primjer,

(računala, računalo, Ncnpa) (4.11)

je element HML-a, računala je oblik riječi, računalo je lema, a Ncpna je morfosintaktički opis oblika riječi računala. Drugi primjer bio bi ureĎena trojka glagola čita

(čita, čitati, Vmip3s) (4.12)

Kod hrvatskog morfološkog leksikona oblik riječi i lema su nizovi slova hrvatske abecede, dok je morfosintaktički opis riječi niz alfanumeričkih znakova koji se mogu ograničiti regularnim izrazom. Regularni izrazi za morfosintaktičke opise gramatičkih kategorija hrvatskog morfološkog leksikona su prikazani u tablici 4.8.

Tablica 4.8. Regularni izrazi morfosintaktičkih opisa riječi hrvatskog morfološkog leksikona

Gramatička kategorija	**Regularni izraz**
Imenica	N(c|p)(m|f|n)(s|p)(n|g|a|d|v|l|i)--(n|y)---
Glagol	V(m|a|o|c)(i|m|c|n|p)(p|i|f|s|l|a)(1|2|3)(s|p)(m|f|n)(a|p)(n|y)-------
Pridjev	A(f|s)(p|c|s)(m|f|n)(s|p)(n|g|a|d|v|l|i)(n|y)-(n|y)----
Zamjenica	P(p|d|i|s|q|r|x)(1|2|3)(m|f|n)(s|p)(n|g|a|d|v|l|i)(s|p)(m|f|n)(n|y)(p|s)(n|a)-(n|y)-----
Prilog	Rg(p|c|s)----
Prijedlog	Sp(s|c)(g|d|a|l|i)-
Veznik	C(c|s)(s|c)-----
Broj	M(c|o|m|s)(m|f|n)(s|p)(n|g|d|a|v|l|i)(d|r|l)---(n|y)---
Uzvik	I-(s|c)
Skraćenica	Y(n|r)(m|f|n)(s|p)(n|g|d|a|l|i)-
Čestica	Q(z|q|o|r)--

Notacija morfosintaktičkog opisa riječi ostvaruje se nizom znakova gdje je svaki znak vrijednost nekog atributa. Karakteristika morfosintatičkog opisa je što:

- atribute odreĎuje pozicijom,
- vrijednosti predstavlja jednim znakom,
- a znak povlake "–" označava vrijednost koja nije primjenjiva za odreĎeni atribut.

U tablici 4.9 su prikazani svi atributi s mogućim vrijednostima pojedinih gramatičkih kategorija hrvatskog standardnog jezika.

Tablica 4.9. Atributi i vrijednosti atributa morfosintaktičkog opisa

Gramatička kategorija	0	1	2	3	4	5	6	7	8
Imenica	N	vrsta	rod	broj	padeţ				
		cp	mfn	sp	ngdavli				
Glagol	V	vrsta	oblik	vrijeme	lice	broj	rod		negacija
		maoc	imcnp	pifsla	123	sp	mfn		ny
Pridjev	A	vrsta	stupanj	rod	broj	padeţ			
		fs	pcs	mfn	sp	ngdavli			
Zamjenica	P	vrsta	lice	rod	broj	padeţ			
		pdisqrx	123	mfn	sp	ngdavli			
Prilog	R	vrsta	stupanj						
		g	pcs						
Prijedlog	S	vrsta	formacija	padeţ					
		p	sc	gdali					
Veznik	C	vrsta	formacija						
		cs	sc						
Broj	M	vrsta	rod	broj	padeţ	oblik			
		coms	mfn	sp	ngdavli	drl			
Uzvik	I		formacija						
			sc						
Skraćenica	Y	vrsta	rod	broj	padeţ				
		nr	mfn	sp	ngdali				
Čestica	Q	vrsta							
		zqor							

Kod imenica, na primjer, drugi atribut odreĎuje rod, a dopuštene vrijednosti su m (muški rod), f (ţenski rod) i n (srednji rod).
Regularnim izrazima iz tablice 4.8 se opisuju sve vrijednosti atributa svih gramatičkih kategorija riječi korištenih u hrvatskom morfološkom leksikonu. Ako se znak povlake nalazi na kraju niza, onda se on iz praktičnih razloga ne piše. Potpuni zapis morfosintaktičkog opisa imenice iz primjera (4.11) glasio bi Ncnpa---. Jedino znak na poziciji 0 odreĎuje gramatičku kategoriju riječi, dok ostali znakovi odreĎuju vrijednosti atributa te gramatičke kategorije. Na primjer, Ncnpa iz primjera (4.11) je morfosintaktički opis opće (c-common) imenice (N-Noun) srednjeg roda (n-neuter) u akuzativu (a-accusative) mnoţine (p-plural), dok Vmip3s iz primjera (4.12) označava indikativ (i-indicative) glavnog (m-main) glagola (V-verb) u trećem licu (3) prezenta (p-present) jednine (s-singular).

Za potrebe opisivanja kontroliranog jezika Hrvatski morfološki leksikon HML se formalno definira kao podskup skupa $HSJ \times HSJ \times JMO$, gdje je HSJ hrvatski standardni jezik, a JMO regularni jezik morfosintaktičkog opisa opisan regularnim izrazima iz tablice 4.8. Elementi skupa HML su ureĎene trojke, koje se zovu *morfološke trojke*.
Na skupu JMO se definira relacija parcijalnog ureĎaja "manje ili jednako" ($\leq$) na sljedeći način. Neka su $X = x_0 \dots x_n$ i $Y = y_0 \dots y_m$ dva morfosintaktička opisa. Ako je $n \leq m$ onda kaţemo da je $X \leq Y$ ako vrijedi

$$x_i = y_i \vee y^i = \text{-}, i \in \{1, \dots, n\} \tag{4.13}$$

U slučaju kad je $n > m$, onda je $X \leq Y$ ako vrijedi (4.13) i ako

$$x_i = \text{-}, i \in \{m+1, \ldots, n\} \tag{4.14}$$

odnosno ostali znakovi morfosintaktičkog opisa su vrijednosti koje nisu primjenjive za odreĎeni atribut. Na osnovu relacije ureĎaja "≤" definiraju se relacija "veće ili jednako" (≥) i relacija jednakosti (=)

$$\begin{gathered} X \geq Y \Leftrightarrow Y \leq X \\ X = Y \Leftrightarrow X \leq Y \wedge Y \leq X \end{gathered} \tag{4.15}$$

Mnogi oblici riječi imaju slične morfosintaktičke opise i razlikuju se po jednom ili više atributa. UvoĎenjem parametara za atribute morfosintaktičkog opisa se stvara osnova za definiranje nizova oblika riječi koji imaju zajedničke poveznice.

4.4.1.1 Parametrizacija morfološkog leksikona

Svi oblici leme imaju neke zajedničke atribute morfosintaktičkog opisa. Lema računalo ima 14 različitih ureĎenih oblika riječi

(računalo, računalo, Ncnsn), (računala, računalo, Ncnsg),
(računalu, računalo, Ncnsd), (računalo, računalo, Ncnsa),
(računalo, računalo, Ncnsv), (računalu, računalo, Ncnsl),
(računalom, računalo, Ncnsi), (računala, računalo, Ncnpn), (4.16)
(računala, računalo, Ncnpg), (računalima, računalo, Ncnpd),
(računala, računalo, Ncnpa), (računala, računalo, Ncnpv),
(računalima, računalo, Ncnpl), (računalima, računalo, Ncnpi)

čiji se morfosintaktički opisi razlikuju po vrijednostima trećeg i četvrtog atributa, a vrijednosti prvog i drugog atributa su nepromijenjene. Treći atribut moţe imati vrijednost "s" ili "p", što označava jedninu ili mnoţinu, dok četvrti atribut označava jedan od sedam padeţa imenice.
Nepromjenjivi atributi morfosintaktičkog opisa su atributi koji imaju istu vrijednost za sve morfosintaktičke opise svih ureĎenih oblika leme. *Promjenjivi atribut morfosintatičkog opisa* nema istu vrijednost za sve morfosintaktičke opise svih ureĎenih oblika leme. Iz primjera (4.16) očito je kako prvi i drugi atribut ne utječu, a treći i četvrti atribut utječu na morfološki oblik imenice. Nulti atribut je kod svih riječi iste vrste nepromjenjiv i on odreĎuje gramatičku kategoriju riječi.
Za sve gramatičke kategorije riječi morfološkog leksikona definira se *parametrizirani morfosintaktički opis* kao niz promjenjivih atributa, nepromjenjivih atributa i/ili vrijednosti atributa odnosno znakova. Promjenjivi i nepromjenjivi atributi morfosintaktičkog opisa se jednim imenom zovu *parametri morfosintaktičkog opisa*.
Na osnovu ove definicije, svaki morfosintaktički opis je ujedno i parametrizirani morfosintaktički opis, jer se sastojati samo od vrijednosti atributa, odnosno znakova. Kod parametriziranog morfosintaktičkog opisa, znak se uvijek nalazi na nultoj poziciji jer označava gramatičku kategoriju riječi.
Iz praktičnih razloga, uvodi se notacija parametriziranog morfosintaktičkog opisa koja koristi

- veliki znak za označavanje gramatičke kategorije riječi
- male znakove za označavanje vrijednosti atributa

- nakošene male znakove za označavanje promjenjivih atributa morfosintaktičkog opisa
- nadcrtane kose male znakova za označavanje nepromjenjivih atributa morfosintaktičkog opisa

Na primjer Nvmbp, je primjerak parametriziranog morfosintaktičkog opisa riječi koja na nultoj poziciji ima znak N kojim se označava gramatička kategorija imenice, v je nepromjenjivi atribut, m je vrijednost atributa, a b i p su promjenjivi atributi morfosintaktičkog opisa. Za razliku od oznaka za vrijednosti atributa koje su ograničene regularnim izrazom iz tablice 4.8, oznaka za promjenjivi ili nepromjenjivi atribut moţe biti bilo koji nakošeni znak ili nadcrtani kosi znak. Na primjer, parametrizirani morfosintaktički opis Nn_1mn_2n_3 je ekvivalentan parametriziranom morfosintaktičkom opisu Nvmbp. Tablica 4.10 prikazuje najčešće korištene oznake parametara morfosintaktičkog opisa.

Tablica 4.10. Parametrizacija morfosintaktičkog opisa

Gramatička kategorija	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Imenica	N	*v*	*r*	*b*	*p*				
Glagol	V	*v*	*o*	*t*	*l*	*b*	*r*		*n*
Pridjev	A	*v*	*s*	*r*	*b*	*p*			
Zamjenica	P	*v*	*l*	*r*	*b*	*p*			
Prilog	R	*v*	*s*						
Prijedlog	S	*v*	*f*	*p*					
Veznik	C	*v*	*f*						
Broj	M	*v*	*r*	*b*	*p*	*o*			
Uzvik	I		*f*						
Skraćenica	Y	*v*	*r*	*b*	*p*				
Čestica	Q	*v*							

Parametriziranim morfosintaktičkim opisima se moţe napraviti restrikcija morfološkog leksikona. N$n_1n_2n_3n_4$ je varijabla koja obuhvaća sve imenice, bez obzira na vrstu, rod, broj i padeţ, dok varijabla Nn_1mn_2n_3 obuhvaća sve imenice muškog roda.

Slično kao i kod morfosintaktičkog opisa (4.13) (4.14), i kod parametriziranih morfosintaktičkih opisa se definira relacija parcijalnog uređaja "$\leq$" na sljedeći način. Neka su $X = x_0 \dots x_n$ i $Y = y_0 \dots y_m$ dva parametrizirana morfosintaktička opisa. Ako je $n \leq m$ onda kaţemo da je $X \leq Y$ ako vrijedi

$$x_i = y^i \vee y_i = \text{-} \vee y_i \text{ je parametar}, i \in \{1, \dots, n\} \tag{4.17}$$

U slučaju da je $n > m$, onda je $X \leq Y$ ako vrijedi (4.17) i ako

$$x_i = \text{-} \vee x_i \text{ je parametar}, i \in \{m+1, \dots, n\} \tag{4.18}$$

Relacije "$\geq$" i "$=$" se kao i kod morfosintaktičkih opisa (4.15) definiraju na sljedeći način

$$\begin{gathered} X \geq Y \Leftrightarrow Y \leq X \\ X = Y \Leftrightarrow X \leq Y \wedge Y \leq X \end{gathered} \tag{4.19}$$

Parametri i relacije uređaja se ne uvode samo kod morfosintaktičkog opisa, već i kod morfoloških trojki.

Parametrizirani hrvatski morfološki leksikon $PHML$ je podskup skupa $HSJ \cup \{\varepsilon\} \times HSJ \cup \{\varepsilon\} \times PJMO \cup \{\varepsilon\}$, gdje je HSJ hrvatski standardni jezik, $PJMO$ parametrizirani jezik morfosintaktičkih opisa, a ε oznaka za prazni niz znakova. Elementi jezika $PJMO$ su parametrizirani morfosintaktički opisi, odnosno nizovi parametara i znakova morfosintaktičkog opisa, dok se elementi parametriziranog morfološkog leksikona $PHML$ zovu *parametrizirane morfološke trojke*.

Za razliku od morfološke trojke, parametrizirana morfološka trojka moţe sadrţavati prazni niz znakova ε i parametrizirane morfosintaktičke opise. Parametriziranim morfosintaktičkim opisom se vrši restrikcija HML-a na skup morfoloških trojki koji zadovoljavaju odreĎene uvjete. Na primjer, (ε, računalo, ε) predstavlja one morfološke trojke koji imaju lemu računalo, dok (ε, ε, N$vrbp$) predstavlja sve imenice. Očito je kako (ε, ε, ε) predstavlja sve morfološke trojke *HML*-a.

Ako su (X, Y, Z) i (X', Y', Z') parametrizirane morfološke trojke. Relacija parcijalnog ureĎaja "≤" na skupu *PHML* se definira na sljedeći način:

$$(X, Y, Z) \leq (X', Y', Z') \Leftrightarrow (X = X') \vee (X' = \varepsilon), (Y = Y') \vee (Y' = \varepsilon), \\ (Z \leq Z') \vee (Z = \varepsilon), \tag{4.20}$$

Za dvije parametrizirane morfološke trojke (X, Y, Z) i (X', Y', Z'), relacije "≥" i "=" se definiraju na sljedeći način

$$X \geq Y \Leftrightarrow Y \leq X \\ X = Y \Leftrightarrow X \leq Y \wedge Y \leq X \tag{4.21}$$

Kako bi se iz parametriziranog morfosintaktičkog opisa i njemu manje ili jednakog morfosintaktičkog opisa izdvojili parametri i njihove vrijednosti, definira se pridruţivanje vrijednosti parametara. *Funkcija pridruživanja vrijednosti parametara*

$$vp: PJMO \times JMO \rightarrow P \times VP \tag{4.22}$$

gdje su *P* skup parametara i *VP* skup vrijednosti parametara, pridruţuje paru parametriziranog morfosintaktičkog opisa i morfosintaktičkog opisa ureĎene parove gdje je prvi član parametar, a drugi član vrijednost tog parametra. Neka su $X = x_1 \ldots x_n$ parametrizirani morfosintaktički opis i $Y = y_1 \ldots y_m$ morfosintaktički opis, onda je

$$vp(X, Y) = \{(x_i, y_i) | x_i - \text{parametar}, y_i - \text{vrijedost parametra}\} \tag{4.23}$$

Skup $vp(X, Y) \subseteq P \times VP$ zovemo *skupom vrijednosti parametara* od *X* i *Y*. Skup vrijednosti parametara $vp(X, Y)$ će biti prazan ako i samo ako je $X \not\geq Y$ ili ako *X* ne sadrţi niti jedan parametar.

Na primjer, $vp(\text{N}crbp, \text{Ncmsa}) = \{(r, \text{m}), (b, \text{s}), (p, \text{a})\}$ što se skraćeno piše

r	b	p
m	s	a

(4.24)

Na skupu vrijednosti parametara se definira relacija parcijalnog ureĎaja "≤" na sljedeći način

$$X \leq Y \Leftrightarrow \{(p, v) | (p, v) \in X \wedge v \neq \text{-}\} \subseteq Y \tag{4.25}$$

gdje su $X, Y \subseteq P \times VP$. Na primjer

$$\begin{matrix} r & b & p \\ \text{m} & \text{-} & \text{a} \end{matrix} \leq \begin{matrix} r & b & p \\ \text{m} & \text{s} & \text{a} \end{matrix} \Leftrightarrow \{(r,m),(p,a)\} \subseteq \{(r,m),(b,s),(p,a)\} \quad (4.26)$$

Očito, svaka morfološka trojka je ujedno i parametrizirana morfološka trojka (koja nema promjenjivih parametara). Parametrizacije i relacije parcijalnog uređaja definirane na nivou riječi će sluţiti za opisivanje svojstava fraza kontroliranog jezika.

4.4.2 Fraza i gramatika fraza

Formalni jezik je definiran formalnom gramatikom koja se sastoji od konačnog skupa nezavršnih znakova, konačnog skupa završnih znakova, konačnog skupa produkcija i početnog nezavršnog znaka. Pri tome su nezavršni znakovi formalnog jezika podskup abecede jezika. U gramatici fraza, završni znakovi su parametrizirane morfološke trojke, a nezavršni znakovi zovu se *parametrizirane fraze*.
Primjer fraze

(ulazna, ulazan, Afpfsn) (jedinica, jedinica, Ncfsn) (4.27)

se sastoji od pridjeva ulazan i imenice jedinica. Ove dvije riječi se slaţu po rodu, broju i padeţu, a regularni izraz koji bi ih opisivao glasi

(ε, ε, Avs$\boldsymbol{rbp}$) (ε, ε, N$v\boldsymbol{rbp}$) (4.28)

Zajednički parametri morfosintaktičkih opisa ovih riječi su označeni podebljanim nakošenim slovima. Parametrizirane morfološke trojke iz primjera (4.28) su završni znakovi gramatike fraza kontroliranog jezika. Nezavršni znak odnosno parametrizirana fraza P1$\boldsymbol{rbp}$ za primjer (4.28) se definira i skraćeno piše

P1$\boldsymbol{rbp}$ = Avs$\boldsymbol{rbp}$ N$v\boldsymbol{rbp}$ (4.29)

Parametrizirana fraza uključuje parametre koji utječu na parametre parametriziranih morfosintaktičkih opisa. Frazu iz primjera (4.27) prihvaća produkcija (4.29), ali i fraza

(tehničkim, tehnički, Afpfpi) (podrškama, podrška, Ncfpi) (4.30)

je prihvaćena istom produkcijom jer se pridjev nalazi ispred imenice i s njom se slaţe u rodu broju i padeţu. Neki regularni izrazi uključuju parametrizirane morfološke trojke s postavljenim lemama, na primjer

(ε, biti, Vcip3$\boldsymbol{b}$--$\boldsymbol{n}$) (ε, vrsta, N$v\boldsymbol{rb}$n) (4.31)

prihvaća jedino sljedeće fraze

(je, biti, Vcip3s) (vrsta, vrsta, Ncfsn),
(su, biti, Vcip3p) (vrste, vrsta, Ncfpn),
(nije, biti, Vcip3s--y) (vrsta, vrsta, Ncfsn),
(nisu, biti, Vcip3p--y) (vrste, vrsta, Ncfpn) (4.32)

, a skraćeni zapis nezavršnog znaka pravila (4.31) je

$$\text{P2}\boldsymbol{rbn} = [\text{biti}, Vcip3\boldsymbol{b}\text{--}\boldsymbol{n}]\ [\text{vrsta}, \text{N}v\boldsymbol{rb}\text{n}] \tag{4.33}$$

Postoje dvije vrste fraza kontroliranog jezika i to *konceptne fraze* CP i *relacijske fraze* RP. Fraze (4.27) i (4.30) su primjerci konceptnih fraza, dok su (4.32) primjerci relacijskih fraza. Konceptne fraze se razlikuju o relacijskih fraza po tome što ne sadrţe glagol. Njima se najčešće nešto imenuje, pa obično sadrţe imenice, pridjeve, zamjenice i brojeve.
Parametrizirana konceptna fraza uključuje parametre roda, broja i padeţa. Relacijska fraza, osim roda, broja i padeţa, uključuje atribut negacije kojim se negira glagol u relacijskoj frazi. Padeţ parametrizirane relacijske fraze često ne ovisi o padeţu nijedne morfološke trojke koja čini relacijsku frazu. On se u tom slučaju odreĎuje s obzirom na valencijska svojstva glagola fraze, kao što je opisano u pododjeljku 4.4.2.2.

4.4.2.1 Konceptna fraza

Koncept u područnom znanju često označava neki objekt iz vanjskog svijeta koji se imenuje ili se nabraja. Zbog toga konceptna fraza kontroliranog jezika uvijek posjeduje imenicu i/ili broj. Najjednostavnija konceptna fraza je imenica. Uz imenicu se često vezuju pridjevi koji se s njom slaţu u rodu, broju i padeţu, pa se konceptna fraza proširuje pridjevima, a na kraju i brojevima. Parametrizirana fraza

$$\text{NP}\boldsymbol{rbp} = \text{M}v\boldsymbol{rbp}o?\ ((\text{A}vs\boldsymbol{rbp} \mid \text{A}vs\text{nsn})?\ \text{A}vs\boldsymbol{rbp})?\ \text{N}v\boldsymbol{rbp}?\ \text{N}v\boldsymbol{rbp} \tag{4.34}$$

zove se *parametrizirana imenska fraza* i ona prihvaća fraze iz tablice 4.11.

Tablica 4.11. Primjer parametriziranih imenskih fraza

Oblici riječi fraze	**Fraza**
jedinica	NP***fsn*** = (jedinica, jedinica, Nc***fsn***)
ulazne jedinice	NP***fpn*** = (ulazne, ulazan, Afs***fpn***) (jedinice, jedinica, Nc***fpn***)
centralna procesorska jedinica	NP***fsn*** = (centralna, centralan, Afs***fsn***) (procesorska, procesorski, Afs***fsn***) (jedinica, jedinica, Nc***fsn***)
aritmetičko logičkim jedinicama	NP***fpd*** = (aritmetičko, aritmetički, Afsnsn) (logičkim, logički, Afs***fpd***) (jedinicama, jedinica, Nc***fpd***)
jedan računalni sustav	NP***msn*** = (jedan, jedan, Mc***msn***l) (računalni, računalan, Afs***msn***) (sustav, sustav, Nc***msn***)

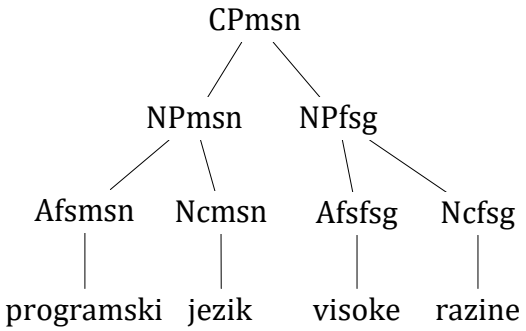
U konceptne fraze spadaju i neodreĎene zamjenice, i to one čija lema je "tko", "što" i "kolik", što se definira produkcijom

$$\text{PP}\boldsymbol{p} = \text{tko}, \text{Px3m-}\boldsymbol{p}\text{--n-n-y} \mid \text{što}, \text{Px3n-}\boldsymbol{p}\text{--n-n-y} \mid [\text{kolik}, \text{Pi3ns}\boldsymbol{p}\text{--n-a}] \tag{4.35}$$

Ova produkcija se koristi za formiranje upitnih rečenica kao što je prikazano u pododjeljku 4.7.3.3. Na posljetku, parametrizirana konceptna fraza se definira produkcijom

$$\text{CP}\boldsymbol{rbp} = (\text{NP}\boldsymbol{rbp}\ \text{NP}r_1b_1\text{g}?\ (Spsp_2\ \text{NP}r_2b_2p_2)?) \mid \text{PP}\boldsymbol{p} \tag{4.36}$$

Konceptna fraza se sastoji od niza imenskih fraza (4.34) ili od fraze zamjenice (4.35). Primjer generiranog stabla konceptne fraze na slici 4.12 sastoji se od dvije imenske fraze, gdje je svaka imenska fraza sastavljena od pridjeva i imenice.



Slika 4.12. Generirano stablo parametrizirane fraze

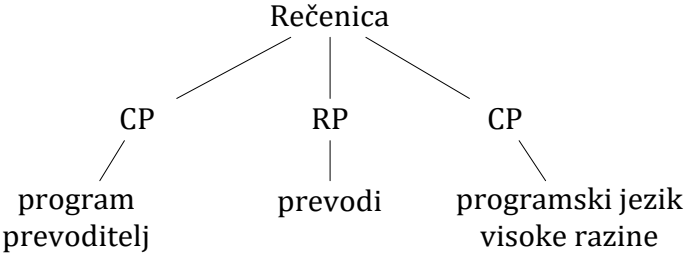
4.4.2.2 Relacijska fraza i valencija glagola

Svaka relacijska fraza ima barem jedan glagol, koji se zove *osnovni (centralni) glagol*. Osnovni glagol u relacijskoj frazi moţe biti vezni ili glavni i moţe biti u negativnom obliku. U relacijskoj frazi se moţe nalaziti imenska fraza i prijedlog. Tablica 4.12 prikazuje oblike relacijskih fraza koje su u jednini, a pojedini oblici, odnosno kategorije, su numerirani rednim brojevima.

Tablica 4.12. Primjer svih oblika relacijske fraze u jednini

	kategorizacija relacijske fraze			**re ac jska fra a**			
				pozitivni oblik l i		**z negativni oblik**	
	osnovni glagol	**imenska fraza**	**prijedlog**	**jednina**	**mnoţina**	**jednina**	**mnoţina**
1	vezni	ne	ne	je	su	nije	nisu
2	vezni	ne	da	je na	se na	nije na	nisu na
3	vezni	da	ne	je vrsta	su vrste	nije vrsta	nisu vrste
4	vezni	da	da	je dio od	su dijelovi od	nije dio od	nisu dijelovi od
5	glavni	ne	ne	prevodi	prevode	ne prevodi	ne prevode
6	glavni	ne	da	prevodi za	prevode za	ne prevodi za	ne prevode za
7	glavni	da	ne	prevodi program	prevode program	ne prevodi program	ne prevode program
8	glavni	da	da	prevodi program u	prevode program u	ne prevodi program u	ne prevode program u

Parametrizacija relacijske fraze uključuje rod, broj, padeţ i atribut negativnosti glagola. Kako bi se bolje razumjela parametrizacija fraze potrebno je navesti kako se oblikuje jednostavna rečenica kontroliranog jezika. Strukturu jednostavne rečenice kontroliranog jezika čine dvije konceptne fraze koje su povezane relacijskom frazom kao što je prikazano na slici 4.13.



Slika 4.13. Jednostavna rečenica kontroliranog jezika

Na parametre relacijske fraze utječu parametri konceptnih fraza u rečenici. U ovom pododjeljku je bitno odrediti parametar padeţa relacijske fraze, jer on utječe na padeţ nadolazeće konceptne fraze u rečenici. Primjer sa slike 4.13 sadrţi relacijsku frazu u akuzativu i zbog toga nadolazeća konceptna fraza mora biti u akuzativu. OdreĎivanje padeţa relacijske fraze nije trivijalan zadatak jer njega utječe *valencija glagola*.
Hrvatski valencijski leksikon glagola (CROVALLEX) [MIKE2008] sadrţi valencijske sheme 1739 najučestalijih glagola hrvatskog standardnog jezika. Identifikacija glagola u valencijskom leksikonu se vrši preko leme, a svaka lema sadrţi jedan ili više *valencijskih okvira* koji su poredani po učestalosti pojavljivanja. Nadalje, valencijski okvir se sastoji od niza *valencijskih funktora* kojima se izraţavaju vrste relacija glagola s njegovim dopunama. Svaki valencijski funktor ima *morfemski oblik* kojim se odreĎuje oblik nadopuna glagola. Skup morfemskih oblika je ograničen i kategoriziran, a za odreĎivanju valencije glagola relacijske fraze, koristit će se sljedeći eksplicitno deklarirani morfemski oblici:

- morfemski oblik padeţa gdje brojevi od 1 do 7 označavaju padeţe nadopuna od nominativa do instrumentala, a broj 0 označava nepostojanje nadopune
- morfemski oblik prijedloga s padeţom ima sintaksu prijedlog+broj, na primjer nadopuna "za+2" kaţe da nakon glagola slijedi prijedlog za i nadopuna u genitivu.
- morfemski oblik nedeklinirane dopune s padeţom je oblika indeclinabilia+broj i najčešće se javlja kod nabrajanja dopune, npr. Kapacitet diskete iznosi 1.44 Megabajta.

Osim morfemskih oblika, valencijski funktor posjeduje tipove nadopuna i to obvezne (obl) i tipične (typ) odnosno neobvezne.
Glagol prevesti sadrţi sljedeći valencijski okvir

$$\mathrm{AGT}^{\mathrm{obl}}_{0\ \mathrm{or}\ 1}\ \mathrm{PAT}^{\mathrm{obl}}_{4}\ \mathrm{ORIG}^{\mathrm{typ}}_{\mathrm{s}+2}\ \mathrm{RESL}^{\mathrm{typ}}_{\mathrm{na}+4} \tag{4.37}$$

Funktor $\mathrm{AGT}^{\mathrm{obl}}_{0\ \mathrm{or}\ 1}$ odnosi se na vršitelja radnje. On je obvezna dopuna glagolu, meĎutim morfemski oblik padeţa 0 ili 1 znači da ne mora imati dopunu, a ako je ima, ona je u nominativu. Trpitelj radnje $\mathrm{PAT}^{\mathrm{obl}}_{4}$ je obvezno u akuzativu. Izvor radnje $\mathrm{ORIG}^{\mathrm{typ}}_{\mathrm{s}+2}$ nije obvezan, a morfemska forma prijedloga s padeţom diktira genitiv dopune koja slijedi iza prijedloga "s". Rezultat dogaĎaja $\mathrm{RESL}^{\mathrm{typ}}_{\mathrm{na}+4}$ nije obvezan, dolazi uz prijedlog "na" i u akuzativu je. Neka je dana relacijska fraza "prevodi knjigu na" i konceptne fraze "knjiţevnik" i "engleski jezik". Provjerom valencije osnovnog glagola "prevesti" relacijske fraze primjenom valencijskog okvira (4.37) dobiva se rečenica iz tablice 4.13 u kojoj je konceptna fraza "engleski jezik" u akuzativu.

Tablica 4.13. Primjer primjene valencijskog okvira na relacijsku frazu i prikaz utjecaja na konceptne fraze rečenice

Vrsta fraze	CP	RP			CP
Rečenica	knjiţevnik	prevodi	knjigu	na	engleski jezik
valencijski okvir	$\mathrm{AGT}^{\mathrm{obl}}_{0\ \mathrm{or}\ 1}$		$\mathrm{PAT}^{\mathrm{obl}}_{4}$	$\mathrm{RESL}^{\mathrm{typ}}_{\mathrm{na}+4}$	

U relacijskoj frazi ponekad je potrebno koristiti valencijski leksikon radi odreĎivanja padeţa relacijske fraze. Ukoliko se parametar padeţa odreĎuje uz pomoć valencijskog leksikona, on će se označiti s uskličnikom "!". Na kategorizaciju relacijskih fraza ne utječe vrijeme glagola, iako su u kontroliranom jeziku definirane parametrizirane relacijske fraze za prošlo i buduće vrijeme. S obzirom na kategorizaciju relacijske fraze, svaka relacijska fraza u sadašnjem vremenu je oblika

(<indikativ prezenta veznog glagola>|(<čestica "ne">?
< indikativ prezenta glavnog glagola>)) <imenska fraza>? <prijedlog>? (4.38)

odnosno iza indikativa prezenta veznog glagola se mogu pojaviti imenska fraza i prijedlog. Tablica 4.14 prikazuje sve reprezentativne primjere relacijskih fraza i njihove parametrizirane relacijske fraze za sadašnje vrijeme.

4.14. Parametrizirane relacijske fraze za sadašnje vrijeme

1	je	RP-***bnn*** = Vcip3***b***--***n***
2	je na	RP-***bp*****!** ***n*** = Vcip3***b***--***n*** Sps***p***
3	je vrsta	RP-***bgn*** = Vcip3***b***--***n*** NP***r******b***n
4	je dio od	RP-***bp*****!** ***n*** = Vcip3***b***--***n*** NP***r******b***n Sps***p***
5	prevodi	RP-***bp*****!** ***n*** = [ne, Qz]?***n*** Vmip3***b***--***n***
6	prevodi za	RP-***bp*****!** ***n*** = [ne, Qz]?***n*** Vmip3***b***--***n*** Sps***p***
7	prevodi program	RP-b_1p_2**!** ***n*** = [ne, Qz]?***n*** Vmip3b_1--***n*** NP***r***b_2p_2
8	prevodi program u	RP-bp_2**!** ***n*** = [ne, Qz]?***n*** Vmip3***b***--***n*** NP***r******bp***$_1$ Spsp_2

Parametrizirane relacijske fraze iz tablice 4.14 sadrţe česticu "ne" koja se moţe i ne mora pojaviti. Čestica "ne" će se pojaviti ako je negativni oblik glavnog glagola u morfološkom leksikonu jednak pozitivnom obliku. Za glavni glagol "prevesti" ne postoji negativni oblik, pa se uz njega stavlja čestica "ne". Glagol "imati" ima negativni oblik "nemati" pa nije potrebno ubacivati česticu "ne". Stoga je regularni izraz parametrizirane morfološke trojke čestice "ne" dodatno označen s atributom ***n***.
Relacijska fraza u prošlom vremenu uvijek započinje veznim glagolom i moţe se opisati općim izrazom

<indikativ prezenta veznog glagola> (<particip prošli veznog glagola "biti"> |
< particip prošli glagola >) <imenska fraza>? <prijedlog>? (4.39)

gdje iza veznog glagola slijedi particip prošli veznog ili glavnog glagola, a zatim se moţe pojaviti imenska fraza i prijedlog. Negativni oblik relacijske fraze nastaje postavljanjem negativnog oblika na prvi vezni glagol koji je u indikativu prezenta. Na primjer, za relacijsku frazu "je bio vrsta" negativni oblik bi bio "nije bio vrsta". Tablica 4.15 prikazuje parametrizaciju relacijskih fraza u prošlom vremenu.

4.15. Parametrizirane relacijske fraze za prošlo vrijeme

1	je bi(o|la|lo)	RPr***b***n***n*** = Vcip3***b***--***n*** [biti, Vcps-***br***a]
2	je bi(o|la|lo) na	RPr***bp*****!** ***n*** = Vcip3***b***--***n*** [biti, Vcps-***br***a] Sps***p***
3	je bi(o|la|lo) vrsta	RPr***b***g***n*** = Vcip3***b***--***n*** [biti, Vcps-***br***a] NP***r******b***n
4	je bi(o|la|lo) dio od	RPr***bp*****!** ***n*** = Vcip3***b***--***n*** [biti, Vcps-***br***a] NP***r******b***n Sps***p***
5	je prevodi(o|la|lo)	RPr***bp*****!** ***n*** = Vcip3***b***--***n*** Vmps-***br***a
6	je prevodi(o|la|lo) za	RPr***bp*****!** ***n*** = Vcip3***b***--***n*** Vmps-***br***a Sps***p***
7	je prevodi(o|la|lo) program	RPrb_1p_2**!** ***n*** =Vcip3b_1--***n*** Vmps-b_1***r***a NP***r***b_2p_2
8	je prevodi(o|la|lo) program u	RPr***bp***$_2$**!** ***n*** = Vcip3***b***--***n*** Vmps-***br***a NP***r******bp***$_1$ Spsp_2

Jedino kod relacijske fraze u prošlom vremenu se pojavljuje parametar roda, koji utječe na rod veznog ili glavnog glagola u participu prošlom.
Za buduće vrijeme, relacijska fraza se moţe opisati općim izrazom

<indikativ prezenta pomoćnog glagola "htjeti"> (<infinitiv veznog glagola "biti"> | < infinitiv glavnog glagola >) <imenska fraza>? <prijedlog>? (4.40)

dok tablica 4.16 prikazuje parametrizirane relacijske fraze za buduće vrijeme.

4.16. Parametrizirane relacijske fraze za buduće vrijeme

1	će $b_i$ti	RP-***b***n***n*** = [htjeti, Vaip3***b***--***n***] [biti, Vcn]
2	će $b_i$ti na	RP-***bp*** ! ***n*** = [htjeti, Vaip3***b***--***n***] [biti, Vcn] Sps***p***
3	će $b_i$ti $_V$rs_ta	RP-***bgn*** = [htjeti, Vaip3***b***--***n***] [biti, Vcn] NP***rb***n
4	će $b_i$ti dio od	RP-***bp*** ! ***n*** = [htjeti, Vaip3***b***--***n***] [biti, Vcn] NP***rb***n Sps***p***
5	će $p_r e_V od_i$ti	RP-***bp*** ! ***n*** = [htjeti, Vaip3***b***--***n***] Vmn
6	će $p_r e_V od_i$ti $_Z$a	RP-***bp*** ! ***n*** = [htjeti, Vaip3***b***--***n***] Vmn Sps***p***
7	će $p_r e_V od_i$ti $p_r o_g$ram	RP-$\boldsymbol{b_1 p_2}$! ***n*** = [htjeti, Vaip3***b***--***n***] Vmn NP$\boldsymbol{rb_2 p_2}$
8	će $p_r e_V od_i$ti $p_r o_g$ram u	RP-$\boldsymbol{bp_2}$! ***n*** = [htjeti, Vaip3***b***--***n***] Vmn NP$\boldsymbol{rbp_1}$ Sps$\boldsymbol{p_2}$

Kod parametriziranih relacijskih fraza za buduće vrijeme, negativni oblik se dobiva postavljanjem negativnog oblika indikativa prezenta pomoćnog glagola "htjeti".
Posebnu skupinu čine relacijske fraze koje sadrţe riječ "se". U valencijskom rječniku hrvatskih glagola, kod sloţenih glagola riječ "se" se pojavljuje kao čestica i kao zamjenica [ORAI2009]. Relacijska fraza koja sadrţi riječ "se" tretira je kao česticu čija morfološka trojka glasi (se, se, Qo). Obzirom na vrijeme relacijske fraze sa "se" se definiraju izrazima iz tablice 4.17.

Tablica 4.17. Izrazi relacijskih fraza sa "se" po vremenu

vrijeme	izraz
sadašnje	<čestica "se"> <čestica "ne">? <indikativ prezenta glavnog glagola> <prijedlog>?
prošlo	<čestica "se"> <indikativ prezenta veznog glagola>? <particip prošli glavnog glagola> <prijedlog>?
buduće	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "se"> <infinitiv glavnog glagola> <prijedlog>?

Kod relacijskih fraza sa "se" osnovni glagol je uvijek glavni glagol. Iza glavnog glagola nema imenske fraze, ali se moţe koristiti prijedlog. Parametrizacija relacijske fraze sa "se" je prikazana u tablici 4.18.

Tablica 4.18. Parametrizirane relacijske fraze sa "se"

sadašnje vrijeme		
9	se $p_r e_V$odi	RP-***bp*** ! ***n*** = [se, Qo] [ne, Qz]?***n*** Vmip3***b***--***n***
10	se $p_r e_V$odi $_Z$a	RP-***bp*** ! ***n*** = [se, Qo] [ne, Qz]?***n*** Vmip3***b***--***n*** Sps**p**
prošlo vrijeme		
9	se $p_r e_V od_{i(o\|la\|lo)}$	RPr***bp*** ! ***n*** = [se, Qo]? [je, Vcip3***b***--***n***]?***n*** Vmps-***br***a
10	se $p_r e_V od_{i(o\|la\|lo)}$ $_Z$a	RPr***bp*** ! ***n*** = [se, Qo]? [je, Vcip3***b***--***n***]?***n*** Vmps-***br***a Sps**p**
buduće vrijeme		
9	će se pre_Voditi	RP-***bp*** ! ***n*** = [htjeti, Vaip3***b***--***n***] [se, Qo] Vmn
10	će se pre_Voditi $_Z$a	RP-***bp*** ! ***n*** = [htjeti, Vaip3***b***--***n***] [se, Qo] Vmn Sps**p**

Parametrizirane relacijske fraze 9 i 10 slične su parametriziranim relacijskim frazama 5 i 6. Razlikuju se po tome što, u sadašnjem i prošlom vremenu, fraza uvijek započinje česticom "se", a u budućem vremenu, čestica "se" stoji ispred glavnog glagola.

Sve parametrizirane relacijske fraze iz ovog poglavlja i parametrizirane konceptne fraze iz pododjeljka 4.4.2.1 čine produkcije gramatike fraza kontroliranog jezika. Nezavršni znakovi su CP, RP, NP, PP , a početni nezavršni znak SP je opisan produkcijom

$$\text{SP}\boldsymbol{rbpn} = \text{CP}\boldsymbol{rbp} \mid \text{RP}\boldsymbol{rbpn} \tag{4.41}$$

Provjera pripadnosti fraze jeziku fraza započinje nezavršnim znakom SP, a zatim se provjerava je li fraza konceptna ili relacijska. Kada se doţe do produkcije koju fraza zadovoljava, onda se kreira parametrizirana relacijska fraza čije vrijednosti parametara su vrijednosti morfosintaktičkih opisa oblika riječi koje čine frazu. Jedino je kod relacijskih fraza ponekad potrebno koristiti valencijski leksikon kako bi se odredio padeţ parametrizirane fraze.

4.4.3 Rečenica i gramatika rečenica

Kontekstno neovisna gramatika koja opisuje sintaksu rečenica kontroliranog jezika sastoji se od sljedećih završnih znakova:

- logički operatori AND, OR, NOT,
- veznik LINK,
- rečenice SENTENCE,
- brojevi SINGLE, MULTIPLE,
- mnogostrukosti MIN, MAX,
- konceptne fraze CP,
- relacijske fraze RP i
- vrijednosti podataka VP.

Nezavršni znakovi gramatike rečenica su Complex_Sentence, Sentence_And_List, Sentence_Or_List, Sentence, Subject_Branch, Predicate_Branch, Object_Branch, Subject_And_List, Subject_Or_List, Object_And_List, Object_Or_List, Not_Item, Single_Item, Multiple_Item, Min_Item, Max_Item.
Početni nezavršni znak gramatike rečenica je sloţena rečenica Complex_Sentence opisana nizom rečenica Sentence koje mogu biti povezane logičkim operatorima AND i OR. Logički operatori sluţe za nizanje rečenica, pri čemu se stvara sloţena rečenica.

```
Complex_Sentence = Sentence | Sentence_And_List | Sentence_Or_List
Sentence_And_List = AND[(Sentence | Sentence_Or_List)+]
Sentence_Or_List = OR[(Sentence | Sentence_And_List)+]
```
(4.42)

Rečenica kontroliranog jezika moţe imati najviše tri znaka (grane), koje se referenciraju kao subjektna grana, predikatna grana i objektna grana. TakoĎer gramatika uvaţava rečenice koje imaju samo predikatnu i objektnu granu.

```
Sentence = SENTENCE[Subject_Branch? Predicate_Branch Object_Branch]
```
(4.43)

U predikatnoj grani rečenice nalazi se relacijska fraza ispred koje se moţe nalaziti operator negiranja NOT. Operator negiranja postavlja relacijsku frazu u negativni oblik.

```
Predicate_Branch = RP | Not_Item
Not_Item = NOT[RP]
```
(4.44)

Subjektna grana rečenice moţe biti konceptna fraza, niz znakova povezanih logičkim operatorima "i" i "ili", te konceptne fraze u jednini i mnoţini. Ako je konceptna fraza u mnoţini, onda se ispred nje nalazi završni znak MULTIPLE. Podrazumijeva se ako se ispred konceptne fraze ne nalazi znak broja SINGLE ili MULTIPLE, onda je ona u jednini.

```
Subject_Branch = CP | Subject_Or_List | Subject_And_List | Single_Item |
Multiple_Item
Subject_Or_List = OR[(CP | Subject_And_List | Single_Item | Multiple_Item)+]
Subject_And_List = AND[(CP | Subject_Or_List | Single_Item | Multiple_Item)+]
Single_Item = SINGLE[CP]
Multiple_Item = MULTIPLE[CP]
```
(4.45)

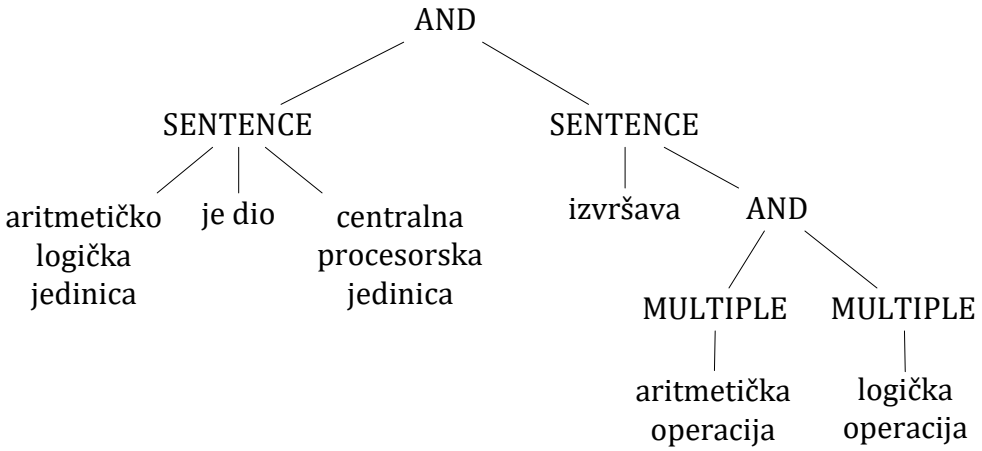
Objektna grana rečenice je slična subjektnoj grani, a razlikuje se po tome što se u njoj moţe nalaziti vrijednost podatka VP, koja moţe biti dodatno ograničena mnogostrukostima MIN i MAX.

```
Object_Branch = CP | VP | Object_Or_List | Object_And_List | Single_Item |
Multiple_Item | Min_Item | Max_Item
Object_Or_List = OR[(CP | Object_And_List | Single_Item | Multiple_Item |
Min_Item | Max_Item)+]
Object_And_List = AND[(CP | Object_Or_List | Single_Item | Multiple_Item |
Min_Item | Max_Item)+]
Min_Item = MIN[VP]
Max_Item = MAX[VP]
```
(4.46)

Navedene produkcije, uz skup završnih i nezavršnih znakova, čine gramatiku rečenica kontroliranog jezika. Potrebno je napomenuti kako se završni znak veznika LINK upotrebljava kod dodatne obrade rečenice kontroliranog jezika. Veznik LINK je opisan produkcijom

$$\text{LINK}[T_1 \ldots T_n] \tag{4.47}$$

gdje je T_i bilo koji znak gramatike rečenica.
Slika 4.14 prikazuje primjer sloţene rečenice kontroliranog jezika koja se sastoji od dvije rečenice povezane s AND logičkim operatorom. Lijeva rečenica ima sve tri grane, dok desna nema subjektnu granu. Objektna grana desne rečenice sastoji se od dva koncepta u mnoţini koji su povezani AND operatorom.



Slika 4.14. Primjer stabla rečenica

Slijed produkcija koji opisuje sloţenu rečenicu sa slike 4.14 dan je u tablici 4.19.

Tablica 4.19. Primjer slijeda produkcija sloţene rečenice

1	Complex_sentence
2	Sentence_And_List
3	AND[Sentence Sentence]
4	AND[SENTENCE[Subject_Branch Predicate_Branch Object_Branch] SENTENCE[Predicate_Branch Object_Branch]]
5	AND[SENTENCE[CP_1 RP_1 CP_2] SENTENCE[RP_2 Object_And_List]]
6	AND[SENTENCE[CP_1 RP_1 CP_2] SENTENCE[RP_2 AND[Multiple_Item Multiple_Item]]]
7	AND[SENTENCE[CP_1 RP_1 CP_2] SENTENCE[RP_2 AND[MULTIPLE[CP_3] MULTIPLE[CP_4]]]]

Rečenice, fraze i riječi kontroliranog jezika opisane u ovom odjeljku, zajedno s modelima skupova podataka opisanih u podpoglavlju 4.3, predstavljaju statičku sliku sustava. Navedene skupove podataka obraĎuju komponente CoLaB Tutor-a tijekom realizacija funkcionalnosti. Funkcionalnost ontološkog opisivanja područnog znanja je početna i jedina funkcionalnost faze oblikovanja područnog znanja. Ostale faze i njihove funkcionalnosti se izvršavaju nakon oblikovanja područnog znanja jer se u fazi oblikovanja područnog znanja kreira ontološki opis područnog znanja na temelju kojeg se postavljaju područno znanje, nastavni sadrţaj, model učenika i rječnik kontroliranog jezika.

4.5 Faza oblikovanja područnog znanja

Oblikovanje ontološki opisanog područnog znanja nema strogo definiranu metodologiju. Razlog tome je što metode i metodologije ontološkog opisivanja znanja spadaju u relativno mlado područje ontološkog inţenjerstva (engl. ontology engineering). Metodologije oblikovanja ontologije su samo dio ontološkog inţenjerstva, ali su za potrebe faze oblikovanja područnog znanja sasvim dovoljne.
U CoLaB Tutor-u, stručnjak bi trebao biti upoznat s osnovnim principima oblikovanja ontologije [USCH1995] [BECK2002]. TakoĎer bi trebao biti upoznat s funkcionalnostima ureĎivača ontologije, kao što je Protégé [NOYF2000] ili Swoop [KALY2005], koje vode ontološkom opisivanju znanja [NOYM2001] [KNUB2004b]. Na osnovu kriterija za analizu različitih metodologija [FERN1999] u fazi oblikovanja područnog znanja CoLaB Tutor-a se ističu:

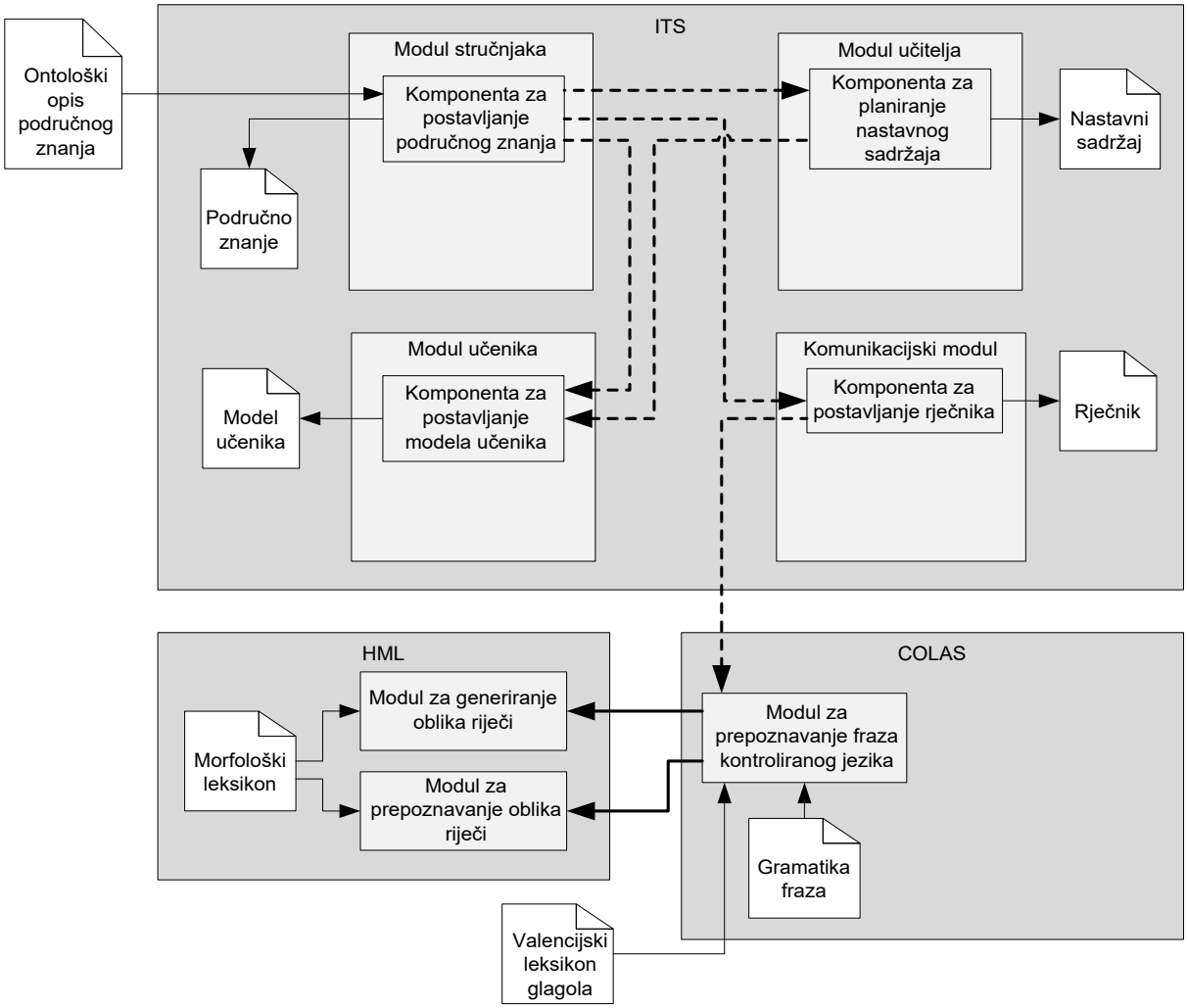
- preporuke za formalizaciju znanja,
- strategije identificiranja elemenata ontologije i
- strategije imenovanja elemenata ontologije.

Preporuke za formalizaciju znanja se odnose na tehnike prikaza znanja koje bi mogle pomoći prilikom oblikovanja područnog znanja. Poznate metodologije nemaju preporuke za formalizaciju znanja kao što su metodologija korištenja prilikom oblikovanja "Enterprise Ontology" [USCH1995], projekt Esprit KACTUS (Knowledge About Complex Technical systems for multiple USe) [SCHR1995] [BERN1996] i METHONTOLOGY [GÓME1996] [FERN1997] [GÓME1998]. Logički pristup formalizacije znanja je preporuka TOVE projekta (TOronto Virtual Enterprise) [GRÜN1994], dok je semantička mreţa osnovni pristup SENSUS ontologije [KNIG1994]. Prilikom oblikovanja područnog znanja CoLaB Tutor-a kombiniraju se semantička mreţa, okviri i logički pristup formalizaciji znanja. Semantička mreţa je pogodna za iznošenje temeljnih odnosa meĎu konceptima, odnosno odnosa meĎu klasama i pripadnosti individua klasama, čime se postavljaju temelji ontološkog opisa područnog znanja. Za uspostavljanje ne hijerarhijskih odnosa meĎu konceptima preporučuje se logički pristup. Logičkim pristupom se najčešće ograničavanjem relacija dodatno opisuju pojedini koncepti, a nerijetko se opisivanje koncepta vrši izravnim povezivanjem s drugim konceptima korištenjem osnovnih logičkih operatora. Prikaz znanja temeljen na okvirima pomaţe prilikom postavljanja vrijednosti svojstava individua.
Moguće *strategije identificiranja elemenata ontologije* kreću se od konkretnog prema apstraktnom, od apstraktnom prema konkretnom ili od relevantnog prema apstraktnom i konkretnom. U CoLaB Tutor-u izbor smjera identificiranja elemenata ontologije ovisi o prirodi područnog znanja koje se formalizirano opisuje. Za ontološko oblikovanje ontologije prototipne verzije područnog znanja "Računalo kao sustav" iz Priloga A prvo se pristupilo identificiranju temeljnih klasa i pripadnih individua, a zatim se pristupilo opisivanju relevantnih koncepata. Moţe se reći kako će se identificiranje elemenata ontologije kod područja s klasificiranim znanjem, poput medicine i biologije, kretati od apstraktnog prema konkretnom. Struktura rečenica kontroliranog jezika CoLaB Tutor-a ovisi o strukturi koncepata i relacija ontologije, stoga će stručnjak odabranim pristupom identificiranju elemenata ontologije utjecati ne samo na strukturu nastavnog sadrţaja, strukturu modela učenika, već i na oblike generiranih rečenica kontroliranog jeziku.
Strategije imenovanja elemenata ontologije imaju vaţnu ulogu u oblikovanju područnog znanja CoLaB Tutor-a pošto gramatička ispravnost rečenica kontroliranog jezika ovisi i o imenovanim konceptima i relacijama. Današnje ontologije posjeduju različite strategije imenovanja koje općenito nisu standardizirane [FLIE2007], a ponekad nisu ni dosljedno primijenjene kod nazivanja različitih elemenata iste ontologije. Većina sustava OWL ontologije prikazuju grafički, meĎutim postoje dva pristupa verbalizaciji OWL ontologije, Attemto Controlled English (ACE) [FUCH2005] i Swoop [HEWL2005]. ACE pristup koristi kontrolirane uzorke za verbalizaciju OWL ontologije čime se omogućava jednostavna interpretacija od strane čovjeka, te je moguće prevesti kontrolirani engleski jezik u ontološki oblik. Swoop s druge strane implementira algoritme koristeći standardne tehnike obrade prirodnog jezika radi prevoĎenja OWL ontologije u uzorke prirodnog jezika.
Pošto se ontologija područnog znanja CoLaB Tutor-a opisuje OWL jezikom, imena koncepata i relacija se zapisuju u XML formatu koji ima ograničenje očitovano u nazivu XML elementa koji ne moţe sadrţavati prazne znakove. Često se prilikom imenovanja elemenata ontologije pojedine riječi imena spajaju graničnim znakom ili se uopće ne odvajaju, već se kombinacijom velikih i malih znakova odreĎuju pojedine riječi u imenu elementa ontologije. Na primjer, naziv koncepta "Računalni_sustav" sadrţi graničnik "_", dok "RačunalniSustav" nema graničnika. Ipak dosljednom primjenom jednog ili drugog načina imenovanja, mogu se

razdvojiti pojedine riječi imena. Ipak se stručnjaku predlaţe imenovanje elemenata ontologije uz pomoć svojstva `rdfs:label` OWL ontologije gdje se ime koncepta ili relacije napiše u prirodnom obliku, odnosno "Računalni sustav". Imena koncepata i imena relacija nemaju jednaka pravila za imenovanje. Ime koncepta najčešće uključuje imenicu, dok relacija posjeduje glagol. Pravila kojima se opisuje poredak i vrsta riječi u imenima koncepata opisana su u pododjeljku 4.4.2.1. U pododjeljku 4.4.2.2 su opisana pravila za imenovanje relacije.
Stručnjaku se preporučuje korištenje navedenih strategija što rezultira generiranjem gramatički ispravnih rečenica kontroliranog jezika nad ontološki opisanim područnim znanjem.

4.6 Faza postavljanja

Postavljanje inteligentnog tutorskog sustava CoLaB Tutor-a je funkcionalnost čija realizacija za ulaz koristi ontološki opis područnog znanja, a za izlaz daje područno znanje, nastavni sadrţaj, model učenika i rječnik kontroliranog jezika. Postavljanjem područnog znanja se na osnovu OWL zapisa ontološki opisanog područnog znanja dobiva područno znanje po modelu iz odjeljka 4.3.2. Nadalje, područno znanje predstavlja ulaz u proces planiranja nastavnog sadrţaja. Na osnovu područnog znanja i nastavnog sadrţaja se postavlja model učenika. Postavljanje rječnika kontroliranog jezika takoĎer koristi područno znanje kako bi identificirao i parametrizirao sve konceptne i relacijske fraze koje se javljaju u područnom znanju. U realizaciji postavljanja ITS-a sudjeluju komponente prikazane na slici 4.15.



4.15. Komponente i skupovi podataka CoLaB Tutor-a koji sudjeluju u fazi postavljanja

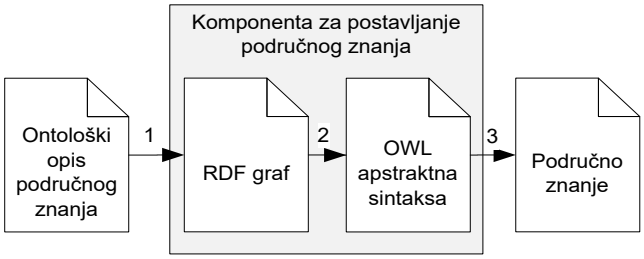
Prvo se opisuje rad komponente za postavljanje područnog znanja koja analizira sintaksu OWL zapisa ontološki opisanog područnog znanja i generira područno znanje.

4.6.1 Postavljanje područnog znanja

Ontološki opis područnog znanja iz odjeljka 4.3.1 je ulaz u proces postavljanja područnog znanja, a za izlaz se dobiva područno znanje temeljeno na modelu opisanog u odjeljku 4.3.2. Postavljanje se vrši u 3 koraka:

1. U prvom koraku se vrši sintaktička analiza ontološkog opisa područnog znanja i kao rezultat se dobiva RDF graf.
2. U drugom koraku se RDF graf sintetizira u OWL apstraktnu sintaksu
3. U trećem koraku se OWL apstraktna sintaksa transformira u područno znanje

Komponenta za postavljanje područnog znanja provodi navedene korake (slika 4.16).



Slika 4.16. Koraci obrade skupova podataka za funkcionalnost postavljanja područnog znanja

U narednim pododjeljcima se opisuju koraci obrade kod postavljanja područnog znanja.

4.6.1.1 Sintaktička analiza OWL ontologije

RDF/XML zapis ontologije se u ovom koraku pretvara u RDF graf, odnosno skup RDF trojki. Sintaktička analiza OWL ontologije se temelji na RDF/XML gramatici [BECK2004] čije produkcije opisuju generiranje RDF trojki. RDF/XML zapis klase identificirane s #Aritmeticko_logicka_jedinica je dan u tablici 4.20.

Tablica 4.20. RDF/XML zapis klase #Aritmeticko_logicka_jedinica

```
<owl:Class rdf:about="#Aritmeticko_logicka_jedinica">
  <rdfs:label>aritmetičko logička jedinica</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="#je_dio"/>
          <owl:someValuesFrom
           rdf:resource="#Centralna_procesorska_jedinica"/>
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#izvrsava"/>
          <owl:allValuesFrom>
            <owl:Class>
              <owl:intersectionOf rdf:parseType="Collection">
                <rdf:Description
                 rdf:about="#Aritmeticka_operacija"/>
                <rdf:Description
                 rdf:about="#Logicka_operacija"/>
              </owl:unionOf>
            </owl:Class>
          </owl:allValuesFrom>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>
```

RDF/XML gramatika je definirana na abecedi znakova koji se zovu događaji i odgovaraju standardu XML informacijskog skupa [COWA2004]. Za primjer RDF/XML zapisa iz tablice 4.20, odgovarajući zapis XML informacijskog skupa je dan u tablici 4.21. Pojedine linije RDF/XML zapisa i zapisa XML informacijskog skupa su numerirane kako bi se istaklo preslikavanje određenog elementa jednog formata zapisa u drugi format.

Tablica 4.21. Zapis XML informacijskog skupa za klasu #Aritmeticko_logicka_jedinica 1

```
start-element(uri(owl:Class)
     attribute(uri(rdf:about),#Aritmeticko_logicka_jedinica)
2    start-element(uri(rdfs:label)
       text-event(aritmetičko logička jedinica)
     )end-element
3    start-element(uri(rdfs:subClassOf)
4      start-element(uri(owl:Class)
5        start-element(uri(owl:intersectionOf)
           attribute(uri(rdf:parseType),Collection)
6          start-element(uri(owl:Restriction)
7            start-element(uri(owl:onProperty)
               attribute(uri(rdf:resource),#je_dio)
             )end-element
8            start-element(uri(owl:someValuesFrom)
               attribute(uri(rdf:resource),
                         #Centralna_procesorska_jedinica)
             )end-element
           )end-element
9          start-element(uri(owl:Restriction)
10           start-element(uri(owl:onProperty)
               attribute(uri(rdf:resource),#izvrsava)
             )end-element
11           start-element(uri(owl:allValuesFrom)
12             start-element(uri(owl:Class)
13               start-element(uri(owl:intersectionOf)
                   attribute(uri(rdf:parseType),Collection)
14                 start-element(uri(rdf:Description)
                     attribute(uri(rdf:about),
                               #Aritmeticka_operacija)
                   )end-element
15                 start-element(uri(rdf:Description)
                     attribute(uri(rdf:about),
                               #Logicka_operacija)
                   )end-element
                 )end-element
               )end-element
             )end-element
           )end-element
       )end-element
     )end-element
   )end-element
)end-element
```

Na osnovu produkcija [BECK2004] koje opisuju transformaciju zapisa XML informacijskog skupa u RDF graf, tablica 4.22 prikazuje rezultat transformacije zapisa iz tablice 4.21. Radi jednostavnijeg praćenja transformacije, u tablici 4.22 su numerirane odgovarajuće trojke RDF grafa.

Tablica 4.22. RDF graf klase #Aritmeticko_logicka_jedinica

1	<#Aritmeticko_logicka_jedinica> <rdf:type> <owl:Class>.
2	<#Aritmeticko_logicka_jedinica> <rdfs:label> "aritmetičko logička jedinica".
3	<#Aritmeticko_logicka_jedinica> <rdfs:subClassOf> _:001.
4	_:001 <rdf:type> <owl:Class>.
5	_:001 <owl:intersectionOf> _:002. _:002 <rdf:first> _:004. _:002 <rdf:rest> _:003. _:003 <rdf:first> _:005. _:003 <rdf:rest> <rdf:nil>.
6	_:004 <rdf:type> <owl:Restriction>.
7	_:004 <owl:onProperty> <#je_dio>.
8	_:004 <owl:someValuesFrom> <#Centralna_procesorska_jedinica>.
9	_:005 <rdf:type> <owl:Restriction>.
10	_:005 <owl:onProperty> <#izvrsava>.
11	_:005 <owl:allValuesFrom> _:006.
12	_:006 <rdf:type> <owl:Class>.
13	_:006 <owl:intersectionOf > _:007.
14	_:007 <rdf:first> <#Aritmeticka_operacija>. _:007 <rdf:rest> _:008.
15	_:008 <rdf:first> <#Logicka_operacija>. _:008 <rdf:rest> <rdf:nil>.

Dobiveni RDF graf na strukturalno jednostavan način opisuje područno znanje preko skupa RDF trojki. MeĎutim, trojke RDF grafa se grupiraju kako bi se sintetizirala čitljivija sintaksa.

4.6.1.2 Sinteza OWL apstraktne sintakse

Dobiveni RDF graf se u drugom koraku sintetizira u OWL apstraktnu sintaksu. Za svaki element OWL apstraktne sintakse je definiran skup RDF trojki koje mu odgovaraju. Tablica 4.23 prikazuje pravila transformacije skupa RDF trojki u izraze OWL apstraktne sintakse. U posljednjem stupcu tablice su numerirane trojke RDF grafa iz tablice 4.22 na koje će se primijeniti odgovarajuća transformacija.

Tablica 4.23. Pravila transformacije OWL apstraktne sintakse u RDF graf

OWL apstraktna sintaksa	RDF trojke	
Class(classID [Deprecated] partial annotation1 … annotationm description1 … descriptionn)	classID rdf:type owl:Class . [classID rdf:type owl:DeprecatedClass .] classID T(annotation1) … classID T(annotationm) classID rdfs:subClassOf T(description1) . … classID rdfs:subClassOf T(descriptionn) .	1,3
annotation(annotationPropertyID dataLiteral)	annotationPropertyID T(dataLiteral) . annotationPropertyID rdf:type owl:AnnotationProperty . annotationPropertyID rdf:type rdf:Property . [opt]	2
intersectionOf(description1 … descriptionn)	_:x rdf:type owl:Class . _:x rdf:type rdfs:Class . [opt] _:x owl:intersectionOf T(SEQ description1…descriptionn) .	4,5|12,13
restriction(ID someValuesFrom(required))	_:x rdf:type owl:Restriction . _:x rdf:type owl:Class . [opt] _:x rdf:type rdfs:Class . [opt] _:x owl:onProperty T(ID) . _:x owl:someValuesFrom T(required) .	6,7,8
restriction(ID allValuesFrom(range))	_:x rdf:type owl:Restriction . _:x rdf:type owl:Class . [opt] _:x rdf:type rdfs:Class . [opt] _:x owl:onProperty T(ID) . _:x owl:allValuesFrom T(range) .	9,10,11
SEQ item1…itemn	_:l1 rdf:type rdf:List . [opt] _:l1 rdf:first T(item1) . _:l1 rdf:rest _:l2 . … _:ln rdf:type rdf:List . [opt] _:ln rdf:first T(itemn) . _:ln rdf:rest rdf:nil .	14,15

Na osnovu pravila sintakse iz tablice 4.23 se dobiva OWL apstraktna sintaksa klase #Aritmeticko_logicka_jedinica koja glasi

```
Class(#Aritmeticko_logicka_jedinica
  intersectionOf(
    restriction(#je_dio someValuesFrom(#Centralna_procesorska_jedinica))
    restriction(#izvrsava allValuesFrom(                                    (4.48)
      intersectionOf(#Aritmeticka_operacija #Logicka_operacija)))
annotation(rdfs:label "Aritmetičko logička jedinica"))
```

4.6.1.3 Transformacija OWL apstraktne sintakse u područno znanje

Neka je $PZ = (M_E, GPZ_E, zo, jo)$ područno znanje na skupu elemenata područnog znanja E. Transformacijska tablica 4.23 daje pravila koja transformiraju OWL apstraktnu sintaksu u trojke grafa područnog znanja GPZ_E i njihove jezične oznake jo. Neke transformacije eksplicitno odreĎuju jezično označene trojke grafa područnog znanja dok većina transformacija mijenja pojedine jezične oznake i elemente prenesene trojke grafa područnog znanja. Prenesene jezične oznake koriste simbole ***l***,***s***,***n***,***o*** preuzete iz definicije **4.6**, dok su preneseni elementi trojke grafa označeni s ***x*** i ***y***. Identifikatori individua, klasa, individualnih i

podatkovnih uloga OWL apstraktne sintakse sluţe za definiranje znakovnog označavanja zo elemenata područnog znanja. Skup elemenata područnog znanja nije eksplicitno naveden u transformacijskoj tablici, već se koristi znakovna oznaka elementa područnog znanja.

Tablica 4.24. Pravila transformacije OWL apstraktne sintakse u graf područnog znanja

Broj pravila	OWL apstraktna sintaksa - S	Transformacija - T(S)
1	Individual(iID type($type_1$)… type($type_n$) value(pID_1 v_1) … value(pID_k v_k))	$[iID, ind^+, T(type_1)]$ … $[iID, ind^+, T(type_n)]$ $[iID, pID_1, v_1]$ … $[iID, pID_1, v_1]$
2	Class(cID description$_1$ … description$_n$)	$[cID, gen^+, T(description_1)]$ … $[cID, gen^+, T(description_n)]$
3	unionOf(description$_1$ … description$_n$)	$\vee [\boldsymbol{sx}, \boldsymbol{ny}, \boldsymbol{o}T(description_1)]$ … $\vee [\boldsymbol{sx}, \boldsymbol{ny}, \boldsymbol{o}T(description_n)]$
4	intersectionOf(description$_1$ … description$_n$)	$\wedge [\boldsymbol{sx}, \boldsymbol{ny}, \boldsymbol{o}T(description_1)]$ … $\wedge [\boldsymbol{sx}, \boldsymbol{ny}, \boldsymbol{o}T(description_n)]$
5	complementOf(description)	$\boldsymbol{l}[\boldsymbol{sx}, \neg \boldsymbol{y}, \boldsymbol{o}T(description)]$
6	oneOf(iID$_1$ … iID$_n$)	$\boldsymbol{l}[\boldsymbol{sx}, \boldsymbol{ny}, \boldsymbol{o}iID_1]$ … $\boldsymbol{l}[\boldsymbol{sx}, \boldsymbol{ny}, \boldsymbol{o}iID_n]$
7	oneOf(v_1 … v_n)	$\boldsymbol{l}[\boldsymbol{sx}, \boldsymbol{ny}, = v_1]$ … $\boldsymbol{l}[\boldsymbol{sx}, \boldsymbol{ny}, = v_n]$
8	restriction(ID allValuesFrom(range))	$\boldsymbol{l}[\boldsymbol{sx}, \boldsymbol{n}ID, \forall T(range)]$
9	restriction(ID someValuesFrom(required))	$\boldsymbol{l}[\boldsymbol{sx}, \boldsymbol{n}ID, \exists T(required)]$
10	restriction(ID value(value))	$\boldsymbol{l}[\boldsymbol{sx}, \boldsymbol{n}ID, value]$
11	restriction(ID minCardinality(min))	$\boldsymbol{l}[\boldsymbol{sx}, \boldsymbol{n}ID, \geq min]$
12	restriction(ID maxCardinality(max))	$\boldsymbol{l}[\boldsymbol{sx}, \boldsymbol{n}ID, \leq max]$
13	restriction(ID cardinality(card))	$\boldsymbol{l}[\boldsymbol{sx}, \boldsymbol{n}ID, = card]$

Na primjeru izraza (4.48) će se prikazati transformacija u graf područnog znanja. Svaki identifikator čini znakovnu oznaku elementa područnog znanja, pa se za primjer (4.48) dobiva sljedeće znakovno označavanje

$zo(c_1)$=#Aritmeticko_logicka_jednica
$zo(c_2)$=#Centralna_procesorska_jedinica
$zo(c_3)$=#Aritmeticka_operacija
$zo(c_4)$=#Logicka_operacija
$zo(r_1)$=#je_dio
$zo(r_2)$=#izvrsava (4.49)

Nadalje, izraz u OWL apstraktnoj sintaksi se moţe skraćeno zapisati kao

Class(c_1 intersectionOf(restriction(r_1 someValuesFrom(c_2)) restriction(r_2 allValuesFrom(intersectionOf(c_3 c_4))))) (4.50)

Primjenom pravila 2 dobiva se trojka

$$[c_1, gen^+, \text{T(intersectionOf(restriction}(r_1 \text{ someValuesFrom}(c_2)) \text{ restriction}(r_2 \text{ allValuesFrom(intersectionOf}(c_3\ c_4))))]] \quad (4.51)$$

Pravilo 4 trojku pretvara u

$$[c_1, gen^+, \text{T(restriction}(r_1 \text{ someValuesFrom}(c_2)]$$
$$[c_1, gen^+, \text{T(restriction}(r_2 \text{ allValuesFrom}(c_3\ c_4)] \quad (4.52)$$

Primjenom pravila 8 i 9 dobiva se

$$[c_1, r_1, \exists c_2]$$
$$[c_1, r_2, \forall \text{intersectionOf}(c_3\ c_4)] \quad (4.53)$$

Još preostaje primijeniti pravilo 4 na drugu trojku iz (4.53) i time se dobivaju sljedeće trojke grafa područnog znanja

$$[c_1, r_1, \exists c_2]$$
$$[c_1, r_2, \forall c_3]$$
$$[c_1, r_2, \forall c_4] \quad (4.54)$$

U transformacijskoj tablici nisu uključeni svi elementi OWL apstraktne sintakse pošto oni ne utječu na stvaranje trojki grafa područnog znanja. MeĎutim konstruktor individualne i podatkovne uloge utječe na definiranje modela elemenata područnog znanja M_E. Za model elemenata područnog znanja bitno je odrediti koja su individualna svojstva tranzitivna te koja svojstva imaju inverzno individualno svojstvo. Po OWL apstraktnoj sintaksi individualne uloge odreĎuje se je li uloga tranzitivna i ima li inverznu ulogu. U primjeru (4.48) individualna uloga #je_dio ima inverznu ulogu #se_sastoji_od što je opisano OWL apstraktnom sintaksom

$$\text{ObjectProperty(\#je\_dio inverseOf(\#se\_sastoji\_od))} \quad (4.55)$$

, stoga modela elemenata područnog znanja M_E uparuje individualne uloge r_1 i r_1^- gdje je znakovna oznaka

$$zo(r_1^-) = \#\text{se\_sastoji\_od} \quad (4.56)$$

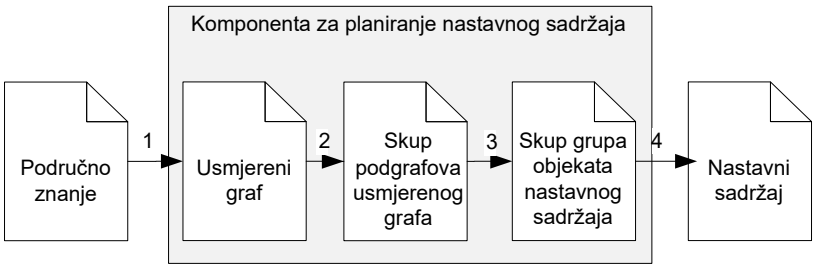
Dobiveno područno znanje se koristi u procesu planiranja nastavnog sadrţaja.

4.6.2 Planiranje nastavnog sadrţaja

Funkcionalnost planiranja nastavnog sadrţaja je proces u fazi postavljanja CoLaB Tutor-a kojim se analizira graf područnog znanja i oblikuju objekti nastavnog sadrţaja koji čine nastavni sadrţaj. Planiranje nastavnog sadrţaja provodi se u 4 koraka:

1. Transformacija područnog znanja u usmjereni graf
2. Disjunkcija usmjerenog grafa
3. Generiranje grupa objekata nastavnog sadrţaja
4. Generiranje nastavnog sadrţaja

U prvom koraku se transformacijom grafa područnog znanja dobiva usmjereni graf. Analizom usmjerenog grafa izdvajaju se međusobno disjunktni podgrafovi koji, u trećem koraku, služe za generiranje objekata nastavnog sadržaja. Jedan podgraf može generirati više objekata nastavnog sadržaja koji se grupiraju radi daljnje analize u četvrtom koraku. Kao rezultat dobiva se nastavni sadržaj, a komponenta za planiranje nastavnog sadržaja provodi sve navedene korake, kao što je prikazano na slici 4.17.



Slika 4.17. Koraci obrade skupova podataka za vrijeme planiranja nastavnog sadržaja

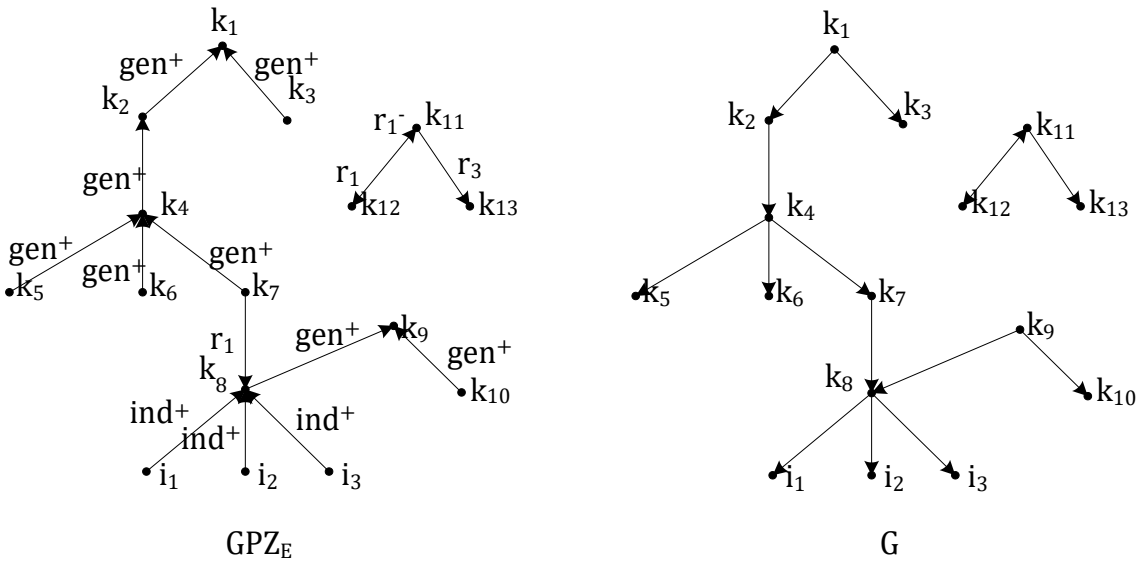
U daljnjim pododjeljcima ovog odjeljka su prikazani koraci u planiranju nastavnog sadržaja.

4.6.2.1 Transformacija područnog znanja u usmjereni graf

Područno znanje $PZ = (M_E, GPZ_E, zo, jo)$ predstavlja ulazni skup podataka funkcionalnosti planiranja nastavnog sadržaja. Graf područnog znanja se preslikava u usmjereni graf $G = (V, A)$ čiji su vrhovi označeni elementima skupa $V = E_K \cup E_I \cup E_V$, a lukovi se određuju ovisno o trojkama grafa područnog znanja po sljedećim transformacijskim pravilima:

1) ako su $x, z \in E_K$ i $y \in E_{IU} \backslash \{gen^+, ind^+\}$ onda je $(x, z) \in A$
2) ako su $x, z \in E_K$ i $y = gen^+$ onda je $(z, x) \in A$
3) ako su $x \in E_I$, $z \in E_K$ i $y = ind^+$ onda je $(z, x) \in A$
4) ako su $x \in E_K \cup E_I$, $z \in E_I$ i $y \in E_{IU} \backslash \{gen^+, ind^+\}$ onda je $(x, z) \in A$
5) ako su $x \in E_K \cup E_I$, $z \in E_V$ i $y \in E_{PU}$ onda je $(x, z) \in A$

Po ovim transformacijskim pravilima, jedino (x, y, z) trojke grafa područnog znanja kojima je predikat y generalizacija gen^+ ili individualizacija ind^+ se transformiraju u usmjerene lukove $(z, x) \in A$. Graf područnog znanja se transformira u jednosmjerni označeni graf. Na primjer, slika 4.18 prikazuje rezultat transformacije grafa područnog znanja u usmjereni graf.



Slika 4.18. Transformacija grafa područnog znanja GPZ_E u usmjereni graf G

Dobiveni usmjereni graf se u sljedećem koraku rastavlja na podgrafove koji su meĎusobno disjunktni. Primjer znakovnog označavanja grafa područnog znanja sa slike 4.18 je dan u tablici 4.25.

Tablica 4.25. Znakovno označavanje elemenata područnog znanja

element	znakovna oznaka elementa
k_1	#komponenta_racunalnog_sustava
k_2	#programska_podrska
k_3	#tehnicka_podrska
k_4	#sistemska_programska_podrska
k_5	#operacijski_sustav
k_6	#usluzni_program
k_7	#program_prevoditelj
k_8	#programski_jezik_visoke_razine
k_9	#programski_jezik
k_{10}	#programski_jezik_niske_razine
k_{11}	#brojevni_sustav
k_{12}	#baza
k_{13}	#znamenka
i_1	#C
i_2	#Pascal
i_3	#Basic
gen^+	##gen
ind^+	##ind
r_1	#prevodi
r_2	#se_sastoji_od
r_2^-	#je_dio
r_3	#ima

Znakovno označavanje elemenata ne utječe na postavljanje nastavnog sadrţaja.

4.6.2.2 Disjunkcija usmjerenog grafa

Usmjereni graf *G* iz primjera 4.18 očito ima podgrafove $G_1 = (V_1, A_1)$ i $G_2 = (V_2, A_2)$ gdje su

V_1={k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10}, i_1, i_2, i_3}
A_1={(k_1, k_2), (k_1, k_3), (k_2, k_4), (k_4, k_5), (k_4, k_6), (k_4, k_7), (k_7, k_8), (k_8, i_1), (k_8, i_2),
(k_8, i_3), (k_9, k_8), (k_9, k_{10}) } (4.57)
V_2={k_{11}, k_{12}, k_{13}}
A_2={(k_{11}, k_{12}), (k_{12}, k_{11}), (k_{11}, k_{13})}

meĎusobno disjunktni odnosno vrijedi $G = G_1 \cup G_2$ i $G_1 \cap G_2 = \emptyset$.
Algoritam za odreĎivanje disjunktnih podgrafova označenog usmjerenog grafa G obilazi sve usmjerene lukove grafa i na osnovu njih stvara podgrafove.

Algoritam 4.1. Određivanje disjunktnih podgrafova

```
1   SPG = ∅
2   for each (x,z) ∈ A
3         found = false
4         for each G_i ∈ SPG
5               if x ∈ V_i ∨ z ∈ V_i then
6                     V_i = V_i ∪ x ∪{z}
7                     A_i = A_i ∪{ x,z }
8                     found = true
9               end if
10        end for
11        if not found then
12              PG = ({x,z},{(x,z)})
13              SPG = SPG ∪{PG}
14        end if
15  end for
```

U liniji 1 se postavlja prazni skup SPG koji će sadrţavati sve podgrafove usmjerenog grafa $G = (V, A)$. Zatim se za svaki usmjereni luk $(x, z) \in A$ i za svaki podgraf $G_i \in SPG$ provjerava da li skup vrhova podgrafa sadrţi x ili z. Ako sadrţi onda se podgraf G_i nadopunjuje vrhovima x, z i usmjerenim lukom (x, z). Ako nije pronaĎen niti jedan takav podgraf, onda se kreira novi graf *PG* i u skup podgrafova *SPG* se dodaje graf *PG*.
Dobiveni skup svih podgrafova usmjerenog grafa *G* predstavlja ulaz u treći korak planiranja nastavnog sadrţaja.

4.6.2.3 Generiranje grupe objekata nastavnog sadrţaja

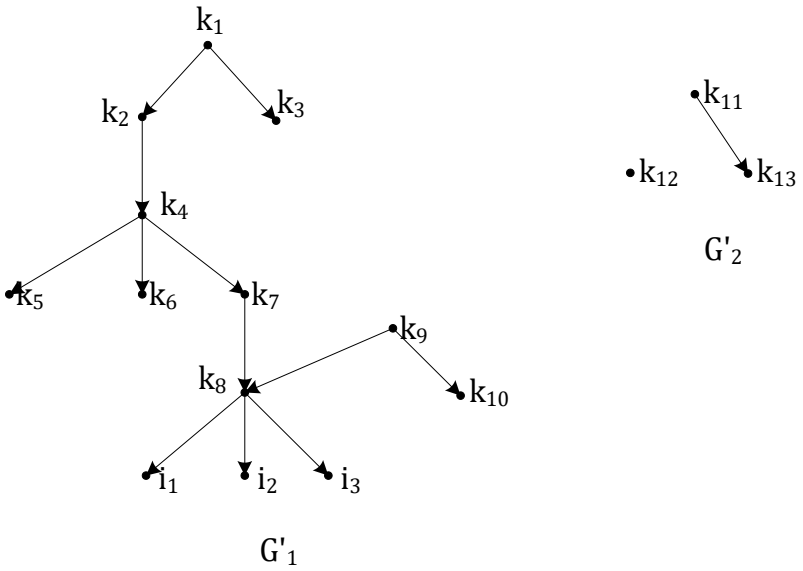
Model nastavnog sadrţaja je definiran kao niz objekata nastavnog sadrţaja, a svaki objekt nastavnog sadrţaja je niz klasa i individua područnog znanja. Za odreĎivanje objekta nastavnog sadrţaja koristi se podgraf usmjerenog grafa *G*. Svaki podgraf generira najmanje jedan objekt nastavnog sadrţaja. Generiranje objekta nastavnog sadrţaja iz podgrafa *G* sastoji se od sljedećih radnji

1. odreĎivanja podgrafa bez ciklusa i dvosmjernih veza,
2. odreĎivanja skupa *početnih vrhova* podgrafa,
3. pretraţivanja grafa po dubini i grupiranje generiranih objekata nastavnog sadrţaja.

Za određivanje skupa početnih vrhova podgrafa potrebno je izbaciti sve cikluse i dvosmjerne lukove iz grafa čime se dobiva podgraf $G_i^{'} = (V_i, A_i^{'})$

$$A_i^{'} = \{(x, y) \in A_i | x \neq y \wedge (y, x) \notin A_i\} \quad (4.58)$$

Skup vrhova podgrafa bez ciklusa i dvosmjernih lukova je jednak skupu vrhova originalnog grafa, dok se iz skupa lukova izbacuju ciklusi i dvosmjerne veze. Na slici 4.19 su dani podgrafovi bez ciklusa i dvosmjernih veza za podgrafove $G_1^{'}$ i $G_2^{'}$ iz primjera sa slike 4.18.



Slika 4.19. Podgrafovi bez ciklusa i dvosmjernih veza

Pošto podgraf G_1 nije imao ciklusa ni dvosmjernih veza, očito vrijedi $G_1^{'} = G_1$. Međutim, podgraf G_2 ima dvosmjerne veze koje su u $G_2^{'}$ uklonjene iz skupa lukova.
Nakon što su uklonjeni ciklusi i dvosmjerne veze iz podgrafova, tada se za svaki podgraf određuje *skup početnih vrhova*. Vrh y podgrafa bez ciklusa i dvosmjernih veza $G_i^{'} = (V_i, A_i^{'})$ zovemo početnim ako $\forall x \in V_i \backslash \{y\}$ vrijedi $(x, y) \notin A_i^{'}$, odnosno ne postoji vrh x tako da je usmjereni luk (x, y) element od $A_i^{'}$. Skup svih početnih vrhova grafa G_i označava se s $p(V_i)$. Za podgrafove G_1 i G_2 iz primjera (4.57) skupovi početnih vrhova su $p(V_1) = \{k_1, k_9\}$ i $p(V_2) = \{k_{11}, k_{12}\}$.
Pretraţivanjem podgrafa po dubini se određuje niz vrhova koji će činiti objekt nastavnog sadrţaja, s time da pretraţivanje uvijek kreće od početnog vrha. Svakim pretraţivanjem se generira jedan objekt nastavnog sadrţaja. Tablica 4.26 prikazuje objekte nastavnog sadrţaja nastali pretraţivanjem po dubini krenuvši od početnog vrha podgrafa.

Tablica 4.26. Primjer generiranih objekata nastavnog sadrţaja

podgraf	početni vrh	objekt nastavnog sadrţaja
G_1	k_1	$\sigma_{G_1,k_1} = k_1 k_2 k_4 k_5 k_6 k_7 k_8 i_1 i_2 i_3 k_3$
	k_9	$\sigma_{G_1,k_9} = k_9 k_8 i_1 i_2 i_3 k_{10}$
G_2	k_{11}	$\sigma_{G_2,k_{11}} = k_{11} k_{12} k_{13}$
	k_{12}	$\sigma_{G_2,k_{12}} = k_{12} k_{13} k_{11}$

Objekti nastavnog sadrţaja su indeksirani podgrafom od kojeg su nastali i početnim vrhom koji je ujedno prva klasa ili individua u nizu. Algoritam pretraţivanja ne određuje jednoznačno objekt nastavnog sadrţaja, jer ako je vrh v_1 povezan lukovima s vrhovima v_2 i

v_3 tada o nićemu ne ovisi hoće li se prvo izabrati vrh v_2 ili v_3. Po ovome objekt nastavnog sadrţaja iz tablice 4.26 mogao je biti i oblika $\sigma_{G_1,k_1} = k_1k_3k_2k_4k_7k_8i_1i_2i_3k_5k_6$.
Pretraţivanjima po dubini svih podgrafova i svih početnih vrhova dobiva se *skup grupa objekata nastavnog sadržaja* $GONS$. Elementi skupa svih objekata nastavnog sadrţaja su skupovi objekata nastavnog sadrţaja koji su nastali od istog grafa i zovu se *grupe objekata nastavnog sadržaja*. Za primjer iz tablice 4.26 skup svih objekata nastavnog sadrţaja je

$$GONS = \{\{\sigma_{G_1,k_1}, \sigma_{G_1,k_9}\}, \{\sigma_{G_2,k_{11}}, \sigma_{G_2,k_{12}}\}\} \tag{4.59}$$

Skup svih objekata nastavnog sadrţaja grupira objekte nastavnog sadrţaja koji su nastali od istog podgrafa. Takve grupe će se u sljedećem koraku analizirati i kao rezultat će se generirati nastavni sadrţaj.

4.6.2.4 Generiranje nastavnog sadrţaja

UsporeĎivanjem objekata nastavnog sadrţaja iz grupe objekata nastavnog sadrţaja odreĎuje se pripadnost i poloţaj objekta u nastavnom sadrţaju. Neka je $\{\sigma_{G_i,k_l} | l \in \{1, \dots, n\}\}$ grupa od n objekata nastavnog sadrţaja. Objekt nastavnog sadrţaja σ_{G_i,k_l} neće pripadati nastavnom sadrţaju ako postoji σ_{G_i,k_m} gdje je $l \neq m$ tako da je $\sigma_{G_i,k_l} \subseteq \sigma_{G_i,k_m}$. Na kraju se grupa smanjuje na one objekte koji će pripadati nastavnom sadrţaju. Skup grupa objekata nastavnog sadrţaja iz (4.59) se smanjuje na skup

$$GONS = \{\{\sigma_{G_1,k_1}, \sigma_{G_1,k_9}\}, \{\sigma_{G_2,k_{11}}\}\} \tag{4.60}$$

jer je $\sigma_{G_2,k_{11}} = \sigma_{G_2,k_{12}}$.
Preostaje za svaku grupu odrediti redoslijed objekata u nastavnom sadrţaju. Redoslijed ovisi o duljini niza koncepata objekta nastavnog sadrţaja, a pošto u prvoj grupi vrijedi $|\sigma_{G_1,k_1}| > |\sigma_{G_1,k_9}|$, a druga grupa ima samo jedan element, onda se navedeni primjer nastavnog sadrţaja opisuje nizom

$$\sigma_{G_1,k_1}\sigma'_{G_1,k_1}\sigma_{G_1,k_9}\sigma'_{G_1,k_9}\sigma_{G_2,k_{11}}\sigma'_{G_2,k_{11}} = \sigma_1\sigma'_1\sigma_2\sigma'_2\sigma_3\sigma'_3 \tag{4.61}$$

Dobiveni nastavni sadrţaj i postavljeno područno znanje iz odjeljka 4.6.1 sluţe za postavljanje modela učenika.

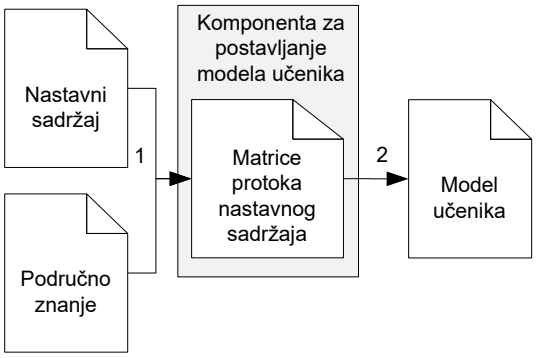
4.6.3 Postavljanje modela učenika

Realizacijom funkcionalnosti postavljanja modela učenika odreĎuju se normirane teţinske vrijednosti svakog koncepta u svim objektima nastavnog sadrţaja.
Postavljanje modela učenika se odvija po sljedećim koracima:

1. pridruţivanje matrice protoka objektu nastavnog sadrţaja
2. odreĎivanje normirane teţinske vrijednosti objekata nastavnog sadrţaja

U prvom koraku se svakom objektu nastavnog sadrţaja pridruţuje kvadratna matrica čiji su redovi i stupci indeksirani konceptima tog objekta nastavnog sadrţaja. Vrijednosti kvadratne matrice su realni pozitivni brojevi koji će, u drugom koraku, sluţiti za izračun teţinske vrijednosti svakog koncepta u objektu nastavnog sadrţaja. Trećim korakom se teţinske

vrijednosti normiraju čime je za svaki koncept objekta nastavnog sadrţaja odreĎena normirana teţinska vrijednost.



Slika 4.20. Koraci obrade skupova podataka za vrijeme postavljanja modela učenika

Postavljanje modela učenika provodi komponenta za postavljanje učenika (slika 4.20), a koraci postavljanja su opisani u narednim pododjeljcima.

4.6.3.1 Matrica protoka objekta nastavnog sadrţaja

Svaki objekt nastavnog sadrţaja σ_{G_i,x_j} je nastao od podgrafa $G_i = (V_i, A_i)$ usmjerenog grafa G koji se dobio iz grafa područnog znanja GPZ_E kao što je opisano u odjeljku 4.6.2. U ovom koraku svakom objektu σ_{G_i,x_j} se pridruţuje kvadratna matrica MP koja se zove *matrica protoka*. Neka su x_k i x_l koncepti objekta nastavnog sadrţaja, koji su ujedno i vrhovi grafa V_i. Realan broj $MP(x_k, x_l)$ će imati vrijednost veću od nule ako postoji putanja od x_l do x_k ili ako je $x_l = x_k$. U slučaju postojanja putanje od x_l do x_k broj $MP(x_k, x_l)$ se odreĎuje na osnovu broja djece vrha x_k koji nisu posjećeni tijekom pretraţivanja po širini. Pseudokod algoritma za odreĎivanje matrice protoka objektu nastavnog sadrţaja σ_{G_i,v_j} je prikazan u algoritmu 4.2.

Algoritam 4.2. Punjenje matrice protoka

```
1   MP = new array(|V_i|, |V_i|)
2   for each x ∈ V_i
3         queue = new queue()
4         visited = ∅
5         queue.enque(x)
6         while not queue.empty
7               z=queue.deque()
8               visited = visited ∪ {w}
9               children = {y|(z,y) ∈ A_i} / visited
10              if children = ∅ then
11                    MP(z,x) = 1
12              else
13                    MP(z,x) = 1 + 1/|children|
14                    for each y ∈ children
15                          queue.enque(y)
16                    end for
17              end if
18        end while
19  end for
```

U liniji 1 se inicijalizira kvadratna matrica protoka reda $|V_i|$, gdje je V_i skup vrhova podgrafa G_i, a vrhovi su ujedno koncepti objekta nastavnog sadrţaja σ_{G_i,x_j}. Zatim se za svaki vrh inicijalizira prazan red `queue` i prazan skup posječenih čvorova *visited*. Red `queue` sluţi za rekurzivno pretraţivanje grafa G_i po širini, a *visited* pamti vrhove koji su posječeni tijekom pretraţivanja. Zatim se u red stavlja vrh z od kojeg započinje pretraţivanje po širini. Uvjetna petlja od linije 6 do linije 18 sluţi za provoĎenje pretraţivanja i za postavljanje vrijednosti matrice protoka. Prvo se u liniji 7 uzima element s reda i doda se u skup *visited*. Zatim se odreĎuje skup djece *children* koji sadrţi sve one vrhove grafa koji su djeca od z i nisu posjećeni. Ako je skup djece prazan onda je $MP(z,x) = 1$, inače se $MP(z,x)$ povećava za recipročnom vrijednosti broja djece. Linije 14, 15 i 16 stavljaju u red elemente skupa *children* od kojih će se nastavit pretraţivanje.
Tablica 4.27 prikazuje matricu protoka objekta nastavnog sadrţaja σ_{G_1,k_1} iz tablice 4.26 nastalog po podgrafu G_1' sa slike 4.19.

Tablica 4.27. Primjer matrice protoka

	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	i_1	i_2	i_3
k_1	1.50	0	0	0	0	0	0	0	0	0	0
k_2	2	2	0	0	0	0	0	0	0	0	0
k_3	1	0	1	0	0	0	0	0	0	0	0
k_4	1.33	1.33	0	1.33	0	0	0	0	0	0	0
k_5	1	1	0	1	1	0	0	0	0	0	0
k_6	1	1	0	1	0	1	0	0	0	0	0
k_7	1	1	0	1	0	0	2	0	0	0	0
k_8	1.33	1.33	0	1.33	0	0	1.33	1.33	0	0	0
i_1	1	1	0	1	0	0	1	1	1	0	0
i_2	1	1	0	1	0	0	1	1	0	1	0
i_3	1	1	0	1	0	0	1	1	0	0	1

4.6.3.2 Normiranje teţinske vrijednosti objekta nastavnog sadrţaja

Neka je MP_σ matrica protoka objekta nastavnog sadrţaja $\sigma = x_1 x_2 \ldots x_n$. Teţinska vrijednost objekta nastavnog sadrţaja t_σ definirana u **4.15** se odreĎuje na osnovu matrice protoka po formuli

$$t_\sigma\ x_i\ = \sum_{j=1}^{n} MP(x_j, x_i) \tag{4.62}$$

, odnosno teţinska vrijednost koncepta x_i jednaka je sumi vrijednosti stupca x_i matrice protoka. Za matricu protoka iz tablice 4.27 teţinske vrijednosti su dane u tablici 4.28.

Tablica 4.28. Primjer teţinskih vrijednosti

$t_\sigma\ k_1$	$t_\sigma\ k_2$	$t_\sigma\ k_3$	$t_\sigma\ k_4$	$t_\sigma\ k_5$	$t_\sigma\ k_6$	$t_\sigma\ k_7$	$t_\sigma\ k_8$	$t_\sigma\ i_1$	$t_\sigma\ i_2$	$t_\sigma\ i_3$
13.17	10.67	1	8.67	1	1	6.33	4.33	1	1	1

Normirana teţinska vrijednost objekta nastavnog sadrţaja $nt_\sigma(x_i)$ za koncept x_i je u definiciji **4.15** opisana kao omjer teţinske vrijednosti objekta nastavnog sadrţaja $t_\sigma(x_i)$ za koncept x_i i maksimalne teţinske vrijednosti objekta nastavnog sadrţaja. Maksimalna teţinska vrijednost za primjer iz tablice 4.28 iznosi 13.17, a normirana teţinska vrijednost je dana u tablici 4.29.

Tablica 4.29. Primjer normiranih teţinskih vrijednosti

$nt_\sigma\ k_1$	$nt_\sigma\ k_2$	$nt_\sigma\ k_3$	$nt_\sigma\ k_4$	$nt_\sigma\ k_5$	$nt_\sigma\ k_6$	$nt_\sigma\ k_7$	$nt_\sigma\ k_8$	$nt_\sigma\ i_1$	$nt_\sigma\ i_2$	$nt_\sigma\ i_3$
1	0.81	0.08	0.66	0.08	0.08	0.48	0.33	0.08	0.08	0.08

Normirana teţinska vrijednost doprinosi ocjeni modela učenika, a ujedno govori o vaţnosti koncepta u objektu nastavnog sadrţaja. Po pravilu, koncepti koji su općenitiji imat će veću teţinsku vrijednost i nalaze se na početku usmjerenog grafa odreĎenog konceptnom restrikcijom grafa područnog znanja za taj objekt nastavnog sadrţaja $PZ_G\ \sigma\ _K$.

Nakon normiranja teţinske vrijednosti svakom konceptu u svakom objektu nastavnog sadrţaja se postavlja inicijalno stanje koncepta. Ako se radi o konceptu koji pripada objektu nastavnog sadrţaja namijenjenog učenju, njegovo inicijalno stanje će biti jednako 0. Inicijalno stanje koncepta u objektu nastavnog sadrţaja namijenjenog učenju je prazan skup ∅. Stanja koncepta se mijenjaju tijekom faze učenja i testiranja kao što je navedeno u odjeljku 4.7.3.

S odreĎenim normiranim teţinskim vrijednostima i postavljanim inicijalnim vrijednostima stanja koncepata svih objekata nastavnog sadrţaja je završeno postavljanje modela učenika. Konačno, posljednja funkcionalnost faze postavljanja je postavljanje rječnika kontroliranog jezika.

4.6.4 Postavljanje rječnika kontroliranog jezika

Rječnik kontroliranog jezika sadrţi sve fraze i sve riječi u frazama kojima se imenuju koncepti i relacije ontološkog opisa područnog znanja. OWL apstraktna sintaksa konstruktora klase, individue, objektne uloge i podatkovne uloge sadrţi identifikator i oblik fraze koji će se procesom prepoznavanja fraze parametrizirati i preko identifikatora postaviti u rječnik kontroliranog jezika. Osim parametriziranih fraza, rječnik kontroliranog jezika sadrţi i morfološke trojke svih riječi koje se pojavljuju u frazama koje su identificirane preko leme.

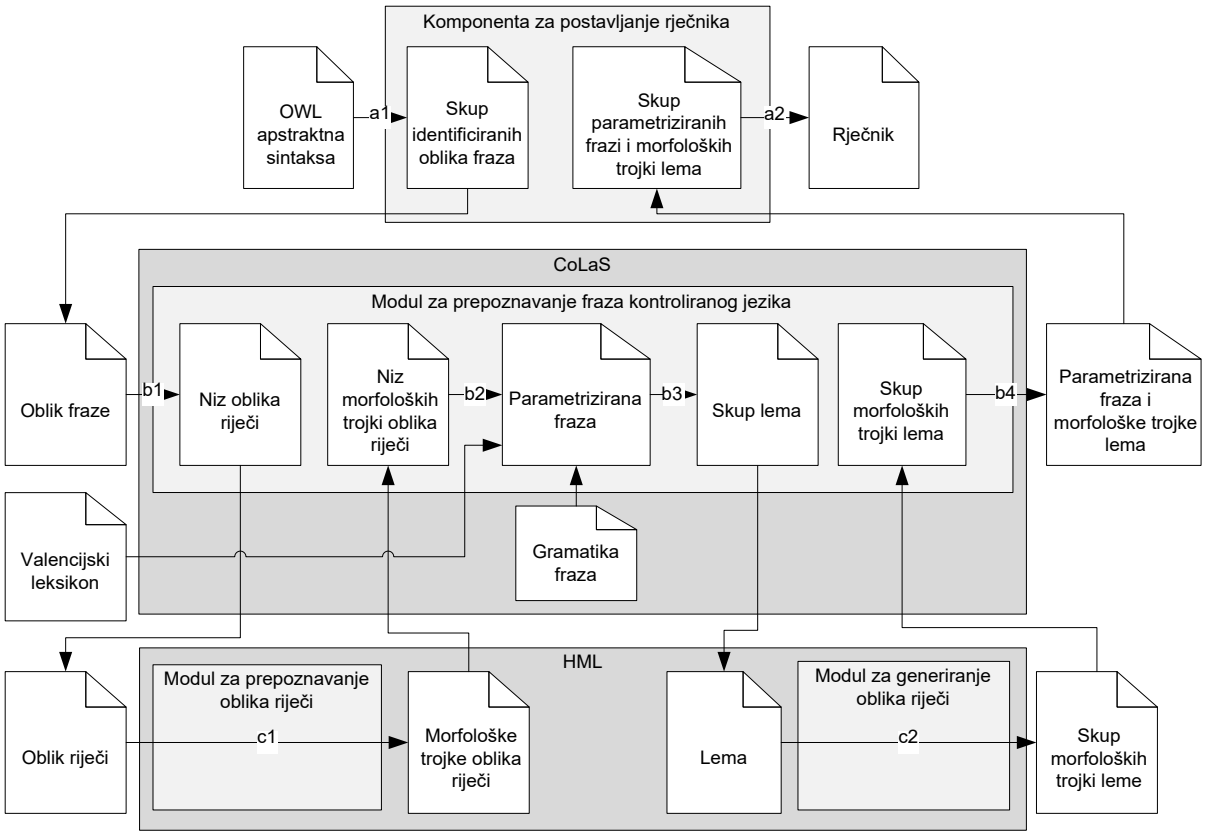
Postavljanje rječnika kontroliranog jezika je proces u kojem sudjeluju komponenta za postavljanje rječnika kontroliranog jezika komunikacijskog modula ITS-a, modula za prepoznavanje fraza CoLaS podsustava i HML-a.

Komponenta za postavljanje rječnika kontroliranog jezika provodi postavljanje po sljedećim koracima:

a1) identifikacija oblika fraza
- b1) rastavljanje oblika fraze na oblike riječi
  - c1) prepoznavanje oblika riječi
- b2) parametrizacija fraze
- b3) izdvajanje lema iz parametrizirane fraze
  - c2) generiranje oblika riječi
- b4) grupiranje parametrizirane fraze i morfoloških trojki lema

a2) postavljanje rječnika fraza i rječnika morfoloških trojki.

Koraci a1) i a2) se realiziraju pomoću komponente za postavljanje rječnika. U realizaciji koraka a1) sudjeluje modul za prepoznavanje fraza kontroliranog jezika koji obavlja korake b1), b2), b3) i b4). HML sudjeluje u realizaciji koraka b1) i b3) tako što modul za prepoznavanje oblika riječi realizira korak c1), a modul za generiranje oblika riječi realizira korak c2). Svi koraci, komponente i skupovi podataka koji sudjeluju u postavljanju rječnika kontroliranog jezika su prikazani na slici 4.21.



Slika 4.21. Koraci obrade skupova podataka za vrijeme postavljanja rječnika kontroliranog jezika

Nakon što komponenta za postavljanje rječnika identificira sve oblike fraze iz OWL apstraktne sintakse, oblici fraze se šalju u modul za prepoznavanje fraza kontroliranog jezika koji iz oblika fraze izdvaja oblike riječi. U HML-u se svaki izdvojeni oblik riječi prepoznaje što rezultira nizom morfoloških trojki svih oblika riječi. Iz ovog niza se izdvajaju kombinacije morfoloških trojki i uz pomoć gramatike fraza se ustanovljava pripadaju li jeziku fraza. U slučaju da pripadaju, onda se niz morfoloških trojki parametrizira kako bi se dobila parametrizirana fraza. Za relacijske fraze je u ovom koraku potrebno uključiti valencijski leksikon CROVALLEX kako bi se odredio parametar padeţa. Uspješnom parametrizacijom fraze se dobiva skup lema za koje će modul za generiranje oblika riječi HML-a dati skup morfoloških trojki lema. Na kraju, parametriziranoj frazi se pridruţuje identifikator i dodaje se u rječnik. U rječnik kontroliranog jezika se takoĎer postavljaju sve morfološke trojke lema čime je završeno postavljanje rječnika, a i sama faza postavljanja CoLaB Tutor-a.

4.6.4.1 Identifikacija oblika fraza

OWL apstraktna sintaksa konstruktora klasa, individua, individualnih uloga i podatkovnih uloga definira njihove identifikatore i nazive. Svaki od navedenih konstruktora obavezno sadrţi identifikator, a moţe sadrţavati i naziv. Konstruktor klase iz primjera (4.48) sadrţi

identifikator #Aritmeticko_logicka_jedinica i naziv "aritmetičko logička jedinica". Naziv u izrazu se određuje preko annotation elementa čiji je atribut rdfs:literal, a vrijednost naziv koncepta ili relacije, što ovisi o konstruktoru. Sintaktičkom analizom OWL apstraktne sintakse izdvajaju se svi identifikatori i nazivi koji predstavljaju oblike fraza. Kao rezultat dobije se skup uređenih parova identifikatora i pridruţenih im oblika fraze.

4.6.4.2 Rastavljanje oblika fraze na oblike riječi

Oblik fraze je predstavljen nizom znakova koji su kodirani po Unicode standardu. U ovom nizu znakova se grupiraju znakovi koji čine oblike riječi i znakovi separatora. Oblik riječi sadrţi znakove koji pripadaju Unicode kategorijama znakova [DAVI2008] danih u tablici 4.30.

Tablica 4.30. Unicode kategorije znakova oblika riječi

Kategorija	Opis	Primjer
Ll	mala slova	a b c d e f g h i j
Lu	velika slova	A B C D E F G H I J
Lt	naslovna slova	Dž Lj Nj Dz Ά Ή Ώ
Lo	ostala slova	ƻ ǀ ‖ ǂ ! ʔ א ב ג ד ה
Nd	brojevi	0 1 2 3 4 5 6 7 8 9 ৭ ১ ০
Pc	interpunkcijei veznici	_ ‿ | ⁀ ﹍ ﹎ ﹏

Klasa znakova regularnog izraza na skupu Unicode znakova u koju spadaju znakovi oblika riječi jest "\w" i ona odgovara Unicode kategorijama znakova iz tablice 4.30. Svi ostali Unicode znakovi pripadaju klasi "\W" i odgovaraju separatorima oblika fraze. Po navedenom, regularni izraz koji opisuje oblik fraze moţe se napisati

$$(\w+)(\W+)* \tag{4.63}$$

Ovaj regularan izraz rastavlja oblik fraze na grupe znakova oblika riječi i znakova separatora gdje iza svakog oblika riječi moţe biti separator. Ako nema separatora, onda oblik fraze ima samo jedan oblik riječi, odnosno oblik fraze je jednak jednom obliku riječi. Regularni izraz koji će izdvojiti samo oblike riječi je

$$(\w+)+ \tag{4.64}$$

Primjenom ovog regularnog izraza na oblik fraze "aritmetičko logička jedinica" dobiva se niz oblika riječi "aritmetičko", "logička" i "jedinica". Svaki oblik riječi iz niza oblika riječi se u sljedećem koraku prepoznaje, odnosno pronalazi se skup morfoloških trojki čiji oblici riječi su jednaki danom obliku riječi.

4.6.4.3 Prepoznavanje oblika riječi

Oblik riječi je ulazni podatak procesa prepoznavanja riječi. Za oblike riječi "aritmetičko", "logička" i "jedinica" se, nakon prepoznavanja oblika riječi, dobiva niz morfoloških trojki svih oblika riječi prikazanih u tablici 4.31.

Tablica 4.31. Niz skupova morfoloških trojki

1	2	3
(aritmetičko, aritmetički, Afpnsay)	(logička, logički, Afpfsny)	(jedinica, jedinica, Ncfsn)
(aritmetičko, aritmetički, Afpnsny)	(logička, logički, Afpfsvy)	(jedinica, jedinica, Ncfpg)
(aritmetičko, aritmetički, Afpnsvy)	(logička, logički, Afpnpay)	
	(logička, logički, Afpnpny)	

Element niza morfoloških trojki svih oblika riječi je skup svih morfoloških trojki za odreĎeni oblik riječi. Prvi element niza iz tablice 4.31 su 3 morfološke trojke za oblik riječi "aritmetičko", drugi element niza ima 4 morfološke trojke, a treći element niza ima 2 morfološke trojke. Postoji 12 mogućih fraza za ovaj niz skupova morfoloških trojki. U sljedećem koraku se za svaku kombinaciju morfoloških trojki provjerava definira li ona frazu kontroliranog jezika.

4.6.4.4 Parametrizacija fraze

Na osnovi gramatike fraza se za odreĎeni niz morfoloških trojki potvrĎuje njegova pripadnost jeziku fraza. Gramatika fraza prihvaća niz morfoloških trojki ako postoji produkcija gramatike fraza koja prihvaća taj niz. Sve produkcije gramatike fraza su zapravo regularni izrazi i na osnovu njih se mogu kreirati ekvivalentni konačni automati. Standardna definicija nedeterminističkog konačnog automata s ε prijelazima (ε-NKA) se za potrebe gramatike fraza proširuje parametrizacijom.
Parametrizirani nedeterministički konačni automat s ε prijelazima (ε-PNKA) gramatike fraza je ureĎena sedmorka $(Q, PHML, \delta, s, F, vp, SVP)$ gdje je

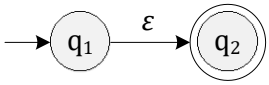
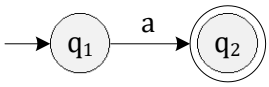
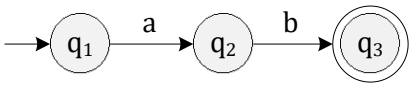
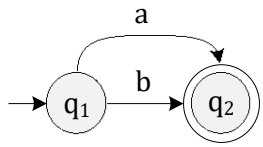
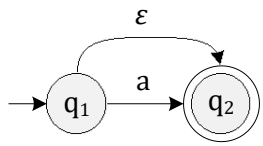
- Q – konačan skup stanja
- $PHML$ – parametrizirani hrvatski morfološki leksikon
- δ – funkcija prijelaza $\delta: Q \times PHML \rightarrow 2^{Q\times(P\times VP)}$
- s – početno stanje
- $F \subseteq Q$ – skup prihvatljivih stanja
- vp – proširena funkcija pridruţivanja vrijednosti parametara $vp: PHML \times HML \rightarrow P \times VP$
- $SVP \subseteq P \times VP$ – skup vrijednosti parametara

Ulazni znakovi od ε-PNKA su parametrizirane morfosintaktičke trojke. Prazni znak ε je zapravo parametrizirana morfološka trojka $(\varepsilon, \varepsilon, \varepsilon)$. Proširenje funkcije pridruţivanja vp se definira preko funkcije pridruţivanja vrijednosti parametara vp (4.22). Ako je $X = (x_1, x_2, x_3) \in PHML$ parametrizirana morfološka trojka i $Y = (y_1, y_2, y_3) \in HML$ morfološka trojka, onda je

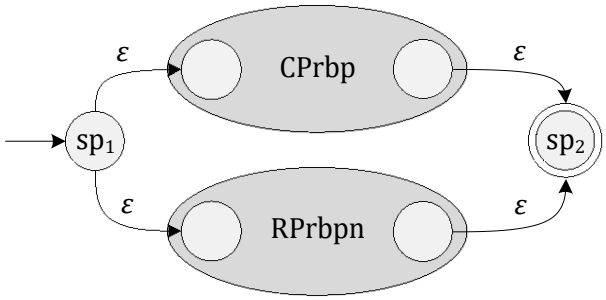
$$vp(X, Y) = vp(x_3, y_3) \tag{4.65}$$

gdje je x_3 parametrizirani morfosintaktički opis, a y^3 morfosintaktički opis.
ε-PNKA se razlikuju od ε-NKA po tome što koriste pridruţivanje vrijednosti parametara i skup vrijednosti parametara kako bi odredili da li prijelaz prihvaća ulazni znak, odnosno morfološku trojku. MeĎutim, stanja i prijelazi od ε-PNKA se definiraju slično kao i kod ε-NKA. Tablica 4.32 prikazuje transformaciju temeljnih regularnih izraza u nedeterminističke konačne automate s ε prijelazima.

Tablica 4.32. Transformacija regularnog izraza u nedeterministički konačni automat s ε prijelazima

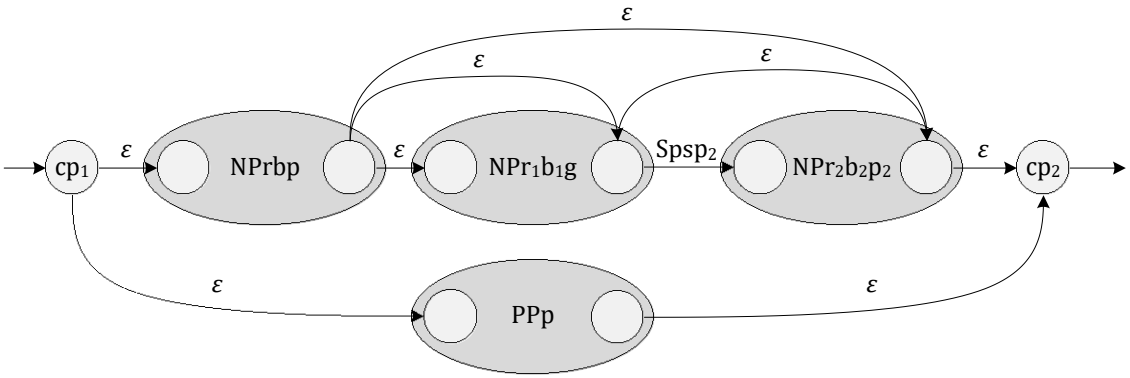
regularni izraz	nedeterministički konačni automat s ε prijelazima
ε	
a	
ab	
a | b	
a?	

Na osnovu transformacijske tablice 4.32 početni nezavršni znak gramatike fraza SP iz (4.41) se opisuje konačnim automatom sa slike 4.22.



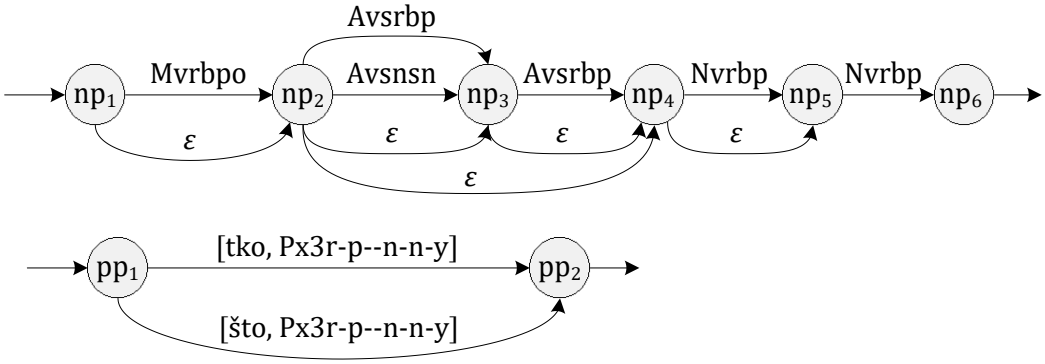
Slika 4.22. ε-PNKA početnog nezavršnog znaka SP gramatike fraza

Pošto se početni nezavršni znak SP opisuje preko nezavršnog znaka konceptne fraze CP i nezavršnog znaka relacijske fraze RP, onda se definiraju konačni automati za CP i RP. Po produkciji nezavršnog znaka CP iz (4.36) je definiran konačni automat sa slike 4.23.



Slika 4.23. ε-PNKA nezavršnog znaka konceptne fraze CP

Nadalje, nezavršni znak CP je definiran preko nezavšnog znaka imenske fraze NP iz (4.34) i nezavršnog znaka zamjenične fraze PP iz (4.35), kao što je prikazano na slici 4.24.



Slika 4.24. ε-PNKA nezavršnog znaka imenske fraze NP i nezavšnog znaka zamjenične fraze PP

Tek u ε-PNKA sa slike 4.24 svi prijelazi su završni znakovi, odnosno parametrizirane morfološke trojke.
Za prijelaz iz stanja q kaţemo da prihvaća morfološku trojku $X = (x_1, x_2, x_3)$ ako je definirana funkcija prijelaza $\delta(q, Y)$ gdje je $Y = (y_1, y_2, y_3)$ parametrizirana morfološka trojka za koju vrijedi $X \leq Y$ i $vp(Y, X) \leq VP$ gdje je VP skup vrijednosti parametara. Nakon što prijelaz prihvati morfološku trojku X, skup vrijednosti parametara postane jednak $VP \cup vp(Y, X)$. Na primjer, neka je np_2 trenutno stanje od ε-PNKA sa slike 4.24, neka je

$$VP = \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} & \boldsymbol{n} \\ - & - & - & -\end{matrix} \tag{4.66}$$

skup vrijednosti podataka i neka je morfološka trojka (aritmetičko, aritmetički, Afpnsay) ulazni znak kojeg ε-PNKA čita. Pošto iz stanja np_2 postoji prijelaz Avs$\boldsymbol{rbp}$ za kojeg vrijedi (aritmetičko, aritmetički, Afpnsay) $\leq$ Avs$\boldsymbol{rbp}$, onda je

$$\delta\ np_1,\ \text{aritmetičko, aritmetički, Afpnsay} \quad = \quad np_3, \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} \\ \text{n} & \text{s} & \text{a}\end{matrix} \tag{4.67}$$

gdje se potencijalni novi skup vrijednosti parametara odredio pomoću funkcije vp na sljedeći način

$$vp(\text{Avs}\boldsymbol{rbp}, (\text{aritmetičko, aritmetički, Afpnsay})) = \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} \\ \text{n} & \text{s} & \text{a}\end{matrix} \tag{4.68}$$

Očito za skup vrijednosti parametara prije prijelaza vrijedi

$$SVP = \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} & \boldsymbol{n} \\ - & - & - & -\end{matrix} \leq \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} \\ \text{n} & \text{s} & \text{a}\end{matrix} \tag{4.69}$$

koji se nakon prijelaza mijenja u

$$SVP = \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} & \boldsymbol{n} \\ - & - & - & -\end{matrix} \cup \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} \\ \text{n} & \text{s} & \text{a}\end{matrix} = \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} & \boldsymbol{n} \\ \text{n} & \text{s} & \text{a} & -\end{matrix} \tag{4.70}$$

Proširenje funkcije prijelaza δ zahtijeva definiranje funkcije ε-okruţenja. Funkcija ε-okruţenja $o_\varepsilon : Q \to 2^Q$ je definirana s

$$o_\varepsilon(q) = \{r | \exists \{q = p_1, \dots, r = p_n\} \wedge \delta(p_i, \varepsilon) = (p_{i+1}, SVP) \wedge i \in \{1, \dots, n-1\}\} \cup \{q\} \tag{4.71}$$

, odnosno ε-okruţenje stanja q je skup svih stanja do kojih se moţe doći iz stanja q isključivo preko ε prijelaza uključujući samo stanje q. Stanju np_2 sa slike 4.24 funkcija ε-okruţenja pridruţuje sljedeći skup

$$o_\varepsilon(np_2) = \{np_2, np_3, np_4, np_5\} \tag{4.72}$$

Funkcija $\delta: Q \times PHML \rightarrow 2^{Q \times (P \times VP)}$ je proširenje funkcije prijelaza definirana s

$$\delta(q, X) = \bigcup_{r \in o_\varepsilon(q)} \delta(r, X) \tag{4.73}$$

gdje je *X* bilo koja morfološka trojka. Za stanje np_2 od ε-PNKA sa slike 4.24 i morfološku trojku Afpnsny proširenje funkcije prijelaza iznosi

$$\begin{aligned} &\delta(np_2, \text{Afpnsny}) = \\ &= \delta(np_2, \text{Afpnsny}) \cup \delta(np_3, \text{Afpnsny}) \cup \delta(np_4, \text{Afpnsny}) \cup \delta(np_5, \text{Afpnsny}) \\ &= \{(np_3, \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} \\ - & - & -\end{matrix}), (np_3, \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} \\ \text{n} & \text{s} & \text{n}\end{matrix})\} \cup \{(np_4, \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} \\ \text{n} & \text{s} & \text{n}\end{matrix})\} \cup \emptyset \cup \emptyset \\ &= \{(np_3, \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} \\ - & - & -\end{matrix}), (np_3, \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} \\ \text{n} & \text{s} & \text{n}\end{matrix}), (np_4, \begin{matrix}\boldsymbol{r} & \boldsymbol{b} & \boldsymbol{p} \\ \text{n} & \text{s} & \text{n}\end{matrix})\} \end{aligned} \tag{4.74}$$

Cilj prepoznavanja je, iz niza skupova morfoloških trojki oblika riječi, odrediti onu kombinaciju koju konačni automat prihvaća. Niz skupova morfoloških trojki iz tablice 4.31 ima 12 različitih kombinacija. Prva kombinacija je niz morfoloških trojki

(aritmetičko, aritmetički, Afpnsay) (logička, logički, Afpfsny)
(jedinica, jedinica, Ncfsn) (4.75)

Tablica 4.33 opisuje prijelaze ε-PNKA sa slike 4.24 za ulazni niz morfoloških trojki (4.75). U stupcu prijelaza se nalaze argumenti proširene funkcije prijelaza, odnosno trenutno stanje, morfološka trojka koja se čita (ulazni znak) i parametrizirana morfološka trojka kojom je definiran prijelaz (znak za usporedbu). Stupac rezultata prijelaza sadrţi nova stanja i skupove vrijednosti podataka. Dobiveni skup vrijednosti podataka *SP* iz rezultata prijelaza se uspoređuje s *SVP* i ako vrijedi $SP \leq SVP$, onda se elementi skupa *SP* nadodaju skupu *SVP*. Skup *SVP* iz prijelaza u prijelaz mijenja svoje elemente, što je prikazano stupcima prije prijelaza i nakon prijelaza.

Tablica 4.33. Prvi primjer prijelaza od ε-PNKA

	argumenti prijelaza			**rezultat prijelaza**	***SVP***	
	stanje	**ulazni znak**	**znak za usporedbu**		**prije prijelaza**	**poslije prijelaza**
1	np_1	(aritmetičko, aritmetički, Afpnsay)	Avs**rbp**	np_3, **r** n **b** s **p** a	**r b p n** - - - -	**r b p n** n s a -
			Avs**rbp**	(np_4, **r** n **b** s **p** a)	**r b p n** - - - -	**r b p n** n s a -
1.1	np_3	(logička, logički, Afpfsny)	Avs**rbp**	(np_4, **r** f **b** s **p** n)	**r b p n** n s a -	
1.2	np_4	(logička, logički, Afpfsny)	Nv**rbp**	∅	**r b p n** n s a -	

Počevši od stanja np_1 ulazna morfološka trojka prijelazi u stanja np_3 i np_4. U redovima 1.1 i 1.2 se uzima sljedeća morfološka trojka i vrši se prijelaz iz stanja np_3 i np_4. MeĎutim u redu 1.1 se vidi kako rezultirajući skup vrijednosti parametara nije manji ili jednak skupu *SVP* prije prijelaza, pa se za ovo stanje automat zaustavlja. Automat se zaustavlja i za argumente prijelaza iz reda 1.2 jer takav prijelaz nije definiran. Stoga se zaključuje kako automat sa slike 4.24 ne prihvaća niz morfoloških trojki (4.75).
Sljedeći niz morfoloških trojki čiji su prijelazi dani u tablici 4.34 je

(aritmetičko, aritmetički, Afpnsay) (logička, logički, Afpfsny)
(jedinica, jedinica, Ncfsn) (4.76)

Tablica 4.34. Drugi primjer prijelaza od ε-PNKA

	argumenti prijelaza			**rezultat prijelaza**	***SVP***	
	stanje	**ulazni znak**	**znak za usporedbu**		**prije prijelaza**	**poslije prijelaza**
1	np_1	(aritmetičko, aritmetički, Afpnsny)	Avsnsn	np_3, ∅	**r b p n** - - - -	**r b p n** - - - -
			Av^s**rbp**	(np_3, **r** n **b** s **p** n)	**r b p n** - - - -	**r b p n** n s n -
			Av^s**rbp**	(np_4, **r** n **b** s **p** n)	**r b p n** - - - -	**r b p n** n s n -
1.1	np_3	(logička, logički, Afpfsny)	Av^s**rbp**	(np_4, **r** f **b** s **p** n)	**r b p n** - - - -	**r b p n** f s n -
1.2	np_3	(logička, logički, Afpfsny)	Avs**rbp**	(np_4, **r** f **b** s **p** n)	**r b p n** n s n -	
1.3	np_4	(logička, logički, Afpfsny)	Nv**rbp**	∅	**r b p n** n s n -	
1.1.1	np_4	(jedinica, jedinica, Ncfsn)	Nv**rbp**	(np_5, **r** f **b** s **p** n)	**r b p n** f s n -	**r b p n** f s n -
			Nv**rbp**	(np_6, **r** f **b** s **p** n)	**r b p n** f s n -	**r b p n** f s n -

Pošto automat nema više ulaznih znakova, a ε-okolina od np_6 sadrţi stanje prihvaćanja sp_2, onda niz morfoloških trojki (4.76) odreĎuje parametriziranu frazu i to konceptnu. Parametrizacija fraze se odreĎuje praćenjem slijeda ulaznih znakova i pripadnim im znakova za usporedbu kao što je za gornji primjer dano u tablici 4.35.

Tablica 4.35. Primjer rezultata prekrivanja ulaznog znaka i znaka za usporedbu

ulazni znak	znak za usporedbu	izlazni znak
(aritmetičko, aritmetički, Afpnsny)	Avsnsn	[aritmetički, Afpnsny]
(logička, logički, Afpfsny)	Avs***rbp***	[logički, Afp***rbp***]
(jedinica, jedinica, Ncfsn)	Nv***rbp***	[jedinica, Nc***rbp***]

Za potrebe odreĎivanja izlaznog znaka uvodi se operator prekrivanja i njegovo proširenje. Neka je $X = x_0 \dots x_n \in JMO$, $Y = y_0 \dots y_m \in PJMO$ i $X \leq Y$. Operator prekrivanja $+: JMO \times PJMO \rightarrow PJMO$ je definiran formulom

$$x_0 \dots x_n + y^0 \dots y^m = z_0 \dots z_p \tag{4.77}$$

gdje je $p = min(n, m)$ i $z_i = y_i$ ako je y_i zajednički parametar ili ako je $y_i =$ - , inače $z_i = x_i$. Ako je $X \not\leq Y$ onda $x_0 \dots x_n + y_0 \dots y_m = \varepsilon$. Po ovoj formuli očito je Afpfsny + Avs***rbp*** = Afp***rbp***.

Proširenje operatora prekrivanja na domeni $HML \times PHML$ se za morfološku trojku (X_1, X_2, X_3) i parametriziranu morfološku trojku (Y_1, Y_2, Y_3) definira formulom

$$(X_1, X_2, X_3) + (Y_1, Y_2, Y_3) = (\varepsilon, X_2, X_3 + Y_3) = [X_2, X_3 + Y_3] \tag{4.78}$$

ako za morfosintaktički opis X_3 i za parametrizirani morfosintaktički opis Y_3 vrijedi $X \leq Y$, inače $(X_1, X_2, X_3) + (Y_1, Y_2, Y_3) = \varepsilon$. Za primjer, izlazni znakovi iz tablice 4.35 su nastali prekrivanjem ulaznog znaka i znaka za usporedbu. Nizanjem izlaznih znakova se dobiva parametrizirana fraza

CP***rbp*** = [aritmetički, Afpnsny] [logički, Afp***rbp***] [jedinica, Nc***rbp***] (4.79)

Pošto je kod imenica rod nepromjenjivi atribut, a "jedinica" imenica ţenskog roda, onda se u parametriziranoj frazi postavljaju vrijednosti nepromjenjivih atributa i dobiva se parametrizirana fraza

CPf***bp*** = [aritmetički, Afpnsny] [logički, Afpf***bp***] [jedinica, Ncf***bp***] (4.80)

4.6.4.5 Parametrizacija relacijske fraze

Za parametrizaciju relacijske fraze se isto kao i kod konceptne fraze koristi ε-PNKA. MeĎutim nakon izvršene parametrizacije ponekad se zahtjeva dodatna obrada parametrizirane fraze radi odreĎivanja vrijednosti parametra padeţa i vrijednosti parametra negativnog oblika fraze. Za odreĎivanje parametra padeţa se koristi valencijski leksikon CROVALLEX opisan u pododjeljku 4.4.2.2. Postoji 8 kategorija relacijskih fraza (tablica 4.12), a ako se zanemari vrsta osnovnog glagola, onda se dobivaju 4 kategorije ovisno o tome ima li relacijska fraza imensku frazu i prijedlog. Kategorija relacijske fraze utječe na odabir valencijskog okvira osnovnog glagola.

Relacijsku frazu prve kategorije čini samo osnovni glagol. Za ovu relacijsku frazu se pretragom valencijskog leksikona traţi valencijski okvir koji sadrţi valencijski funktor AGT i neki drugi funktor koji ima morfemski oblik padeţa. Na primjer, u tablici 4.36 je za oblik fraze "ima" pronaĎen valencijski okvir čiji funktor PAT_4^{obl} odreĎuje vrijednost parametar padeţa u parametriziranoj frazi (4=akuzativ).

Tablica 4.36. Primjena valencijskog okvira na prvu kategoriju relacijske fraze

oblik fraze	ima
produkcija	RP-$\boldsymbol{bp}!\,\boldsymbol{n}$ = [ne, Qz]?$\boldsymbol{n}$ Vmip3$\boldsymbol{b}$--$\boldsymbol{n}$
valencijski okvir	$\mathrm{AGT}^{\mathrm{obl}}_{0\text{ or }1}$ $\mathrm{PAT}^{\mathrm{obl}}_{4}$
parametrizirana fraza	RP-$\boldsymbol{ba}\boldsymbol{n}$ = [imati, Vmip3$\boldsymbol{b}$]

Drugu kategoriju čine relacijske fraze s osnovnim glagolom i prijedlogom. U ovom slučaju se traţi valencijski okvir s funktorom AGT i funktorom koji ima morfemski oblik prijedloga s padeţom. Naravno, prijedlog naveden u morfemskom obliku funktora mora biti jednak prijedlogu relacijske fraze. Tablica 4.37 demonstrira primjenu valencijskog okvira glagola "imati" na parametriziranoj relacijskoj frazi s osnovnim glagolom i prijedlogom.

Tablica 4.37. Primjena valencijskog okvira na drugu kategoriju relacijske fraze

oblik fraze	ima u
produkcija	RP-$\boldsymbol{bp}!\,\boldsymbol{n}$ = [ne, Qz]?$\boldsymbol{n}$ Vmip3$\boldsymbol{b}$--$\boldsymbol{n}$ Sps$\boldsymbol{p}$
valencijski okvir	$\mathrm{AGT}^{\mathrm{obl}}_{0\text{ or }1}$ $\mathrm{LOC}^{\mathrm{typ}}_{\mathrm{u}+6}$
parametrizirana fraza	RP-$\boldsymbol{b}\mathrm{l}\boldsymbol{n}$ = [imati, Vmip3$\boldsymbol{b}$] [u, Spsl]

U treću kategoriju relacijskih fraza spadaju relacijske fraze s osnovnim glagolom i imenskom frazom. Za ovu relacijsku frazu je bitno je li ona imenuje podatkovnu ulogu čija je kodomena broj. U tom slućaju se traţi valencijski okvir koji osim funktora AGT sadrţi funktor čiji morfemski oblik ima nedeklariranu dopunu s padeţom kao što je prikazano u tablici 4.38.

Tablica 4.38. Primjena valencijskog okvira s nedeklariranom dopunom na treću kategoriju relacijske fraze

oblik fraze	vraća podatak
produkcija	RP-$\boldsymbol{b}_1\boldsymbol{p}_2!\,\boldsymbol{n}$ = [ne, Qz]?$\boldsymbol{n}$ Vmip3$\boldsymbol{b}_1$--$\boldsymbol{n}$ NP$\boldsymbol{r}\boldsymbol{b}_2\boldsymbol{p}_2$
valencijski okvir	$\mathrm{AGT}^{\mathrm{obl}}_{0\text{ or }1}$ $\mathrm{EXT}^{\mathrm{obl}}_{\mathrm{indeclinabilia}\ \ 2}$ $\mathrm{DIR3}^{\mathrm{typ}}_{\mathrm{na}+4}$
parametrizirana fraza	RP-$\boldsymbol{bg}\boldsymbol{n}$ = [vratiti, Vmip3$\boldsymbol{b}$][podatak, NPm$\boldsymbol{b}$g]

Nedeklarirana dopuna je broj koji će se umetnuti ispred imenske fraze kada se budu generirale rečenice kontroliranog jezika (pododjeljak 4.7.2.2). MeĎutim, ako se ne radi o broju, onda se traţi valencijski okvir kao i kod prve kategorije što je prikazano tablicom 4.39.

Tablica 4.39. Primjena valencijskog okvira na treću kategoriju relacijske fraze

oblik fraze	ima komponentu
produkcija	RP-$\boldsymbol{b}_1\boldsymbol{p}_2!\,\boldsymbol{n}$ = [ne, Qz]?$\boldsymbol{n}$ Vmip3$\boldsymbol{b}_1$--$\boldsymbol{n}$ NP$\boldsymbol{r}\boldsymbol{b}_2\boldsymbol{p}_2$
valencijski okvir	$\mathrm{AGT}^{\mathrm{obl}}_{0\text{ or }1}$ $\mathrm{PAT}^{\mathrm{obl}}_{4}$
parametrizirana fraza	RP-$\boldsymbol{ba}\boldsymbol{n}$ = [imati, Vmip3$\boldsymbol{b}$][komponenta, NPm$\boldsymbol{b}$a]

Za četvrtu kategoriju se traţi valencijski okvir s funktorom AGT, funktorom morfemskog oblika padeţa i funktorom morfemskog oblika prijedloga s padeţom. Funktor morfemskog oblika padeţa odreĎuje parametar padeţa imenske fraze, a funktor morfemskog oblika prijedloga s padeţom odreĎuje parametar padeţa prijedloga, a ujedno i cjelokupne parametrizirane fraze. Tablica 4.40 prikazuje primjenu valencijskog okvira na relacijsku frazu četvrte kategorije.

Tablica 4.40. Primjena valencijskog okvira na četvrtu kategoriju relacijske fraze

oblik fraze	ima komponentu na
produkcija	RP-$\boldsymbol{bp}_2$! $\boldsymbol{n}$ = [ne, Qz]?$\boldsymbol{n}$ Vmip3$\boldsymbol{b}$--$\boldsymbol{n}$ NP$\boldsymbol{r}\boldsymbol{bp}_1$ Sps$\boldsymbol{p}_2$
valencijski okvir	$\mathrm{AGT}^{\mathrm{obl}}_{0\text{ or }1}$ $\mathrm{PAT}^{\mathrm{obl}}_{4}$ $\mathrm{TWHEN}^{\mathrm{typ}}_{\mathrm{adv-sljedećeg\ tjed}}$ $\mathrm{LOC}^{\mathrm{typ}}_{\mathrm{na+6}}$
parametrizirana fraza	RP-$\boldsymbol{b}$l$\boldsymbol{n}$ = [imati, Vmip3$\boldsymbol{b}$][komponenta, NPm$\boldsymbol{b}$a][na, Spsl]

Podjela relacijskih fraza s obzirom na sadašnje, buduće i prošlo vrijeme, kao i relacijske fraze s česticom "se" ne utječu na odabir valencijskog okvira osnovnog glagola. Odabir valencijskog okvira ovisi jedino o kategoriji relacijske fraze, odnosno o tome ima li relacijska fraza imensku frazu i prijedlog.
Osim upotrebe valencijskog okvira za određivanje parametra padeža, neke relacijske fraze zahtijevaju dodatnu obradu kako bi se ustanovila vrijednost parametra negacije. U tu skupinu spadaju relacijske fraze sadašnjeg vremena koje sadrže glavni glagol. Neki glavni glagoli imaju negativni oblik, kao što je glagol "imati" i parametar negativnosti parametrizirane relacijske fraze u sadašnjem vremenu koja sadrži glagol "imati" se određuje iz morfosintaktičkog opisa glagola, kao što je prikazano u tablici 4.41.

Tablica 4.41. Parametrizirana relacijska fraza s negativnim oblikom glavnog glagola

oblik fraze	nema
produkcija	RP-$\boldsymbol{bp}$! $\boldsymbol{n}$ = [ne, Qz]?$\boldsymbol{n}$ Vmip3$\boldsymbol{b}$—$\boldsymbol{n}$
parametrizirana fraza	RP-$\boldsymbol{b}$a$\boldsymbol{n}$ = [imati, Vmip3$\boldsymbol{b}$--$\boldsymbol{n}$]

Međutim, većina glavnih glagola nema negativni oblik i kod njih je negativni oblik određen pojavljivanjem čestice "ne" ispred glavnog glagola. U tablici 4.42 je dan primjer parametrizirane relacijske fraze koja je nastala iz oblika fraze koji sadrži česticu "ne".

Tablica 4.42. Parametrizirana relacijska fraza bez negativnog oblika glavnog glagola

oblik fraze	ne prevodi
produkcija	RP-$\boldsymbol{bp}$! $\boldsymbol{n}$ = [ne, Qz]?$\boldsymbol{n}$ Vmip3$\boldsymbol{b}$--$\boldsymbol{n}$
parametrizirana fraza	RP-$\boldsymbol{b}$a$\boldsymbol{n}$ = [ne, Qz][prevoditi, Vmip3$\boldsymbol{b}$]

Upotrebom valencijskog leksikona za određivanje parametra padeža i dodatnom obradom kojom se određuje parametar negativnosti je završena parametrizacija relacijske fraze. U sljedećem koraku se izdvajaju leme iz parametriziranih fraza kako bi se u rječnik kontroliranog jezika mogli postaviti svi oblici riječi za danu lemu.

4.6.4.6 Izdvajanje lema iz parametrizirane fraze

Parametrizirana fraza je predstavljena kao niz parametriziranih morfoloških trojki čiji morfosintaktički opisi mogu imati zajedničke parametre. Svaka parametrizirana morfološka trojka parametrizirane fraze sadrži lemu. Parametriziranoj frazi iz primjera (4.80) se izdvajaju leme "aritmetički", "logički" i "jedinica" za koje se u sljedećem koraku generiraju morfološke trojke svih oblika riječi leme.

4.6.4.7 Generiranje oblika riječi

Generiranje oblika riječi je funkcionalnost HML-a koja za ulaz ima lemu, a za izlaz skup svih morfoloških trojki ulazne leme. Tablica 4.43 sadrži sve morfološke trojke leme "jedinica".

Tablica 4.43. Generirani oblici riječi leme "jedinica"

(jedinica, jedinica, Ncfsn)	(jedinice, jedinica, Ncfpn)
(jedinice, jedinica, Ncfsg)	(jedinice, jedinica, Ncfpg)
(jedinici, jedinica, Ncfsd)	(jedinicama, jedinica, Ncfpd)
(jedinicu, jedinica, Ncfsa)	(jedinice, jedinica, Ncfpa)
(jedinico, jedinica, Ncfsv)	(jedinice, jedinica, Ncfpv)
(jedinici, jedinica, Ncfsl)	(jedinicama, jedinica, Ncfpl)
(jedinicom, jedinica, Ncfsi)	(jedinicama, jedinica, Ncfpi)

Parametrizirana fraza i generirani oblici riječi lema parametrizirane fraze se u sljedećem koraku postavljaju u rječnik kontroliranog jezika.

4.6.4.8 Postavljanje rječnika fraza i rječnika morfoloških trojki

Rječnik kontroliranog jezika je zajednički naziv za rječnik fraza i rječnik morfoloških trojki. Sve parametrizirane fraze koje su se pojavile u ontološkom opisu područnog znanja čine rječnik fraza, dok rječnik morfoloških trojki sadrţi sve morfološke trojke riječi koje se pojavljuju u parametriziranim frazama.
U prvom koraku postavljanja rječnika kontroliranog jezika se iz OWL apstraktne sintakse izdvojio identifikator i oblik fraze koji predstavlja naziv elementa OWL ontologije. Oblik fraze se u sljedećim koracima parametrizirao i iz njega su se izdvojile leme za koje su ustanovljeni svi oblici riječi. Parametrizirana fraza i identifikator čine jedan element rječnika fraza. Za identifikator #Aritmeticko_logicka_jedinica i pripadni oblik fraze "aritmetičko logička jedinica" se nakon parametrizacije dobila parametrizirana fraza (4.80) koja zajedno s identifikatorom čini jedan element rječnika fraza.
U rječnik fraza ulaze i predefinirane parametrizirane fraze koje nisu eksplicitno identificirane u OWL apstraktnoj sintaksi. One pripadaju modelu područnog znanja i gramatici rečenica kontroliranog jezika. Generalizacija gen^+ i individualizacija ind^+ su individualne uloge područnog znanja čije su znakovne oznake postavljene u definiciji **4.4**. Za generalizaciju se predefinira oblik fraze "je vrsta" dok individualizacija ima oblik fraze "je". Nadalje, MIN i MAX su završni znakovi gramatike rečenica kojima se pridruţuju identifikatori ##min i ##max, a tim identifikatorima oblici fraza "najmanje" i "najviše". Za završne znakove AND i OR su predefinirani identifikatori ##and i ##or. Tablica 4.44 prikazuje neke elemente rječnika fraza.

Tablica 4.44. Primjer rječnika fraza

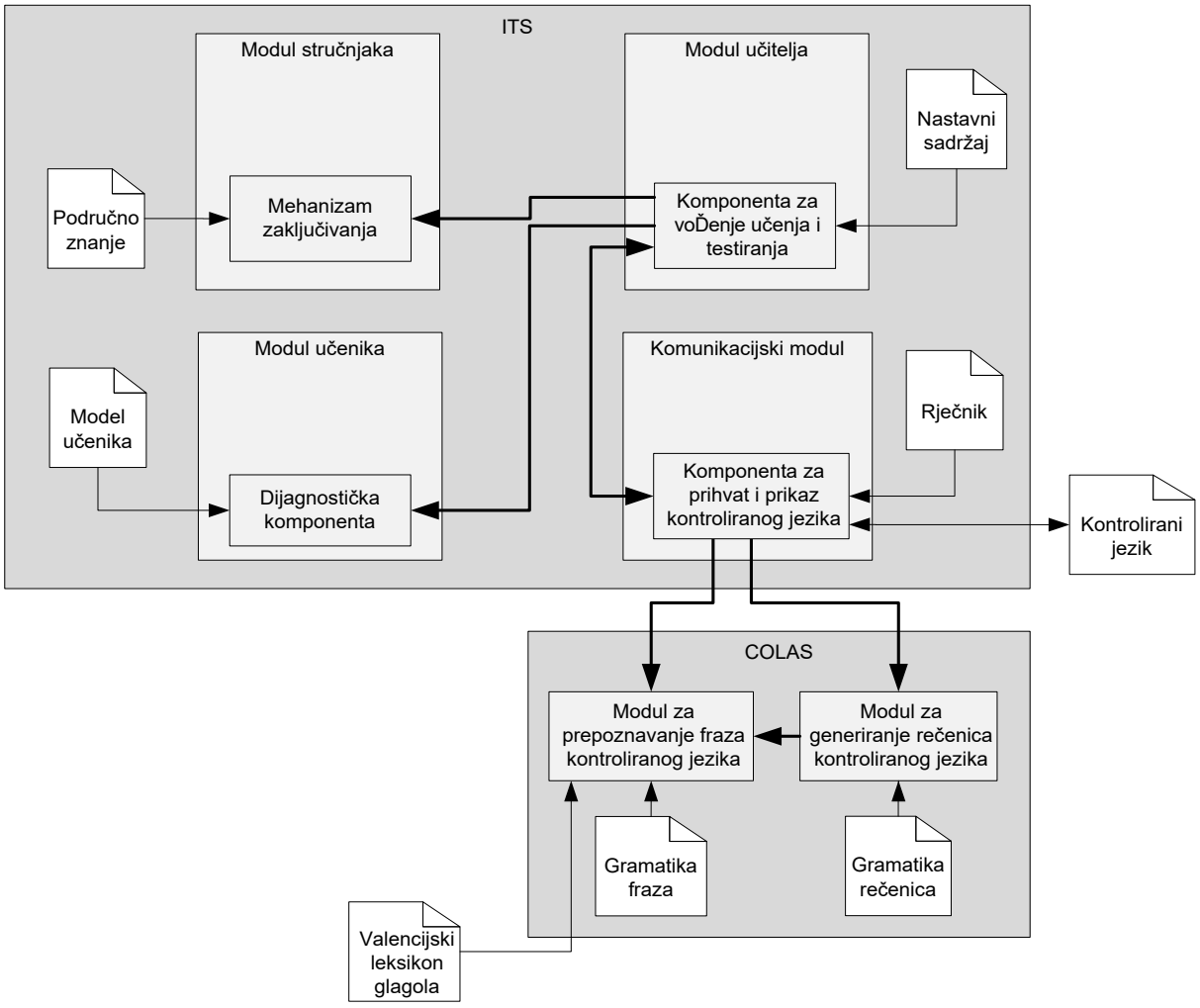
identifikator	parametrizirana fraza
##gen	RP-$\boldsymbol{bgn}$ = [biti, Vcip3$\boldsymbol{b}$--$\boldsymbol{n}$][vrsta, Ncf$\boldsymbol{b}$n]
##ind	RP-$\boldsymbol{bnn}$ = [biti, Vcip3$\boldsymbol{b}$--$\boldsymbol{n}$]
##min	[najmanje, Rns]
##max	[najviše, Rns]
##and	[i, Ccs]
##or	[ili, Ccs]
#Aritmeticko_logicka_jedinica	CPf$\boldsymbol{bp}$ = [aritmetički, Afpnsny] [logički, Afpf$\boldsymbol{bp}$] [jedinica, Ncf$\boldsymbol{bp}$]

Rječnik morfoloških trojki kontroliranog jezika je zapravo podskup morfološkog leksikona HML. Razlika je u tome što sadrţi samo one morfološke trojke riječi koje se pojavljuju u

oblicima fraza ontološkog opisa područnog znanja i riječi unaprijed definiranih parametriziranih fraza.
Postavljanjem rječnika kontroliranog jezika je završena faza postavljanja CoLaB Tutor-a. Slijedi faza učenja i testiranja u kojoj učenik usvaja nastavni sadrţaj, pri čemu se komunikacija provodi kontroliranim jezikom.

4.7 Faza učenja i testiranja

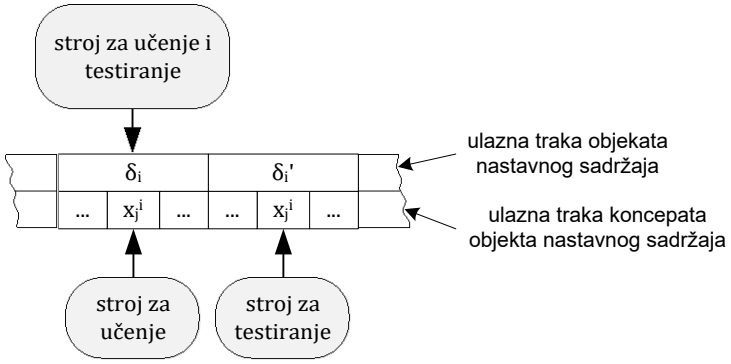
U fazi postavljanja se, na osnovu ontološkog opisa područnog znanja, postavilo područno znanje, nastavni sadrţaj i model učenika. Sljedeća faza, odnosno faza učenja i testiranja, provodi funkcionalnost učenja i testiranja gdje učenik, kao sudionik, kroz interakciju s komunikacijskim modulom CoLaB Tutor-a uči i testira svoje znanje. Na slici 4.25 su prikazane komponente i skupovi podataka koji sudjeluju u učenju i testiranju.



Slika 4.25. Komponente i skupovi podataka CoLaB Tutor-a koji sudjeluju u fazi učenja i testiranja

Središnja komponenta omogućava voĎenje procesa učenja i testiranja znanja učenika na način da upravlja slijedom objekata nastavnog sadrţaja i pripadnih koncepata koji se prezentiraju učeniku. Za prezentaciju nastavnog sadrţaja, komponenta za voĎenje učenja i testiranja uz pomoć mehanizma zaključivanja odabire dijelove područnog znanja koje će komponenta za prihvat i prikaz kontroliranog jezika prikazati učeniku. Područno znanje se prikazuje u obliku

rečenica kontroliranog jezika, stoga komponenta za prihvat i prikaz kontroliranog jezika suraĎuje s CoLaS podsustavom. Ista komponenta u suradnji s CoLas podsustavom prihvaća učenikove odgovore, koje komponenta za voĎenje učenja i testiranja uz pomoć mehanizma zaključivanja analizira i preko dijagnostičke komponente osvjeţava model učenika. Svaka interakcija učenika s komunikacijskim modulom utječe na model učenika, a time i na slijed elemenata nastavnog sadrţaja.
Strojevi s konačnim brojem stanja su upravljačke jedinke komponente za voĎenje učenja i testiranja zaduţeni za upravljanjem slijedom objekata nastavnog sadrţaja i pripadnim im konceptima. Stanja strojeva su proširena *akcijama* kojima se pozivaju funkcije ostalih komponenata radi izvoĎenja zadataka kao što su zaključivanje nad područnim znanjem, generiranje i prepoznavanje kontroliranog jezika i modeliranje učenika. Na najvišem nivou je definiran *stroj za učenje i testiranje* (SUT) koji upravlja slijedom objekata nastavnog sadrţaja. *Stroj za učenje* (SU) i *stroj za testiranje* (ST) su zaduţeni za upravljanje slijedom koncepata unutar pojedinog objekta nastavnog sadrţaja. Slika 4.26 prikazuje konceptni model strojeva koji se koriste u realizaciji faze učenja i testiranja.



Slika 4.26. Koncepti model strojeva namijenjenih fazi učenja i testiranja

SUT čita objekte nastavnog sadrţaja σ_i i $\sigma_i^{'}$, SU čita koncepte x_j^i unutar objekta nastavnog sadrţaja namijenjenog učenju σ_i, a ST čita koncepte x_j^i objekta nastavnog sadrţaja namijenjenog testiranju $\sigma_i^{'}$.
Znakovi s ulaznih traka, osim informacija o elementima nastavnog sadrţaja, sadrţe informacije preuzete iz modela učenika koje su povezane sa svakim elementom nastavnog sadrţaja. Tako je, osim objekta nastavnog sadrţaja, uključeno i stanje učenja, odnosno stanje testiranja tog objekta nastavnog sadrţaja i pripadnih koncepata. Kako bi se pojednostavile ulazne trake, umjesto stanja učenja se definira funkcija posječenosti objekta nastavnog sadrţaja i pripadnog mu koncepta.
Neka je $PZ = (M_E, GPZ_E, zo, jo)$ područno znanje nad modelom elemenata M_E, NS nastavni sadrţaj nad PZ odreĎen nizom $\sigma_1\sigma_1^{'}\sigma_2\sigma_2^{'}\ldots\sigma_m\sigma_m^{'}$ i $MU_{NS} = (NS, nt_{NS}, su_{NS}, st_{NS}, oc_{NS})$ model učenika nad NS. U NS, σ_i su objekti nastavnog sadrţaja namijenjeni učenju, a $\sigma_i^{'}$ objekti nastavnog sadrţaja namijenjeni testiranju. Po definiciji **4.9** vrijedi $\sigma_i = \sigma_i^{'} = x_1^i x_2^i \ldots x_{n_i}^i$ gdje su x_j^i koncepti područnog znanja. Pridruţivanje stanja učenja st_{NS} je funkcija modela učenika MU_{NS} koja svakom σ_i pridruţuje broj iz segmenta [0,1]. Za svaki $\sigma_i^{'}$ je takoĎer definirano stanje testiranja pomoću funkcije st_{NS}.
Kako bi modul učitelja mogao odrediti je li odreĎeni objekt nastavnog sadrţan "posječen" uvode se konstante *prag učenja Pu* i *prag testiranja Pt*, gdje su $Pt, Pu \in [0,1]$ i na osnovu njih se definira funkcija *pridruživanja posjećenosti nastavnog sadržaja* po_{NS}.

$$po_{NS}(\sigma) = \begin{matrix} 0, \text{ako je } \sigma \text{ namijenjen učenju i } su_{NS}(\sigma) < Pu \\ 0, \text{ako je } \sigma \text{ namijenjen testiranju i } st_{NS}(\sigma) < Pt \\ 1, \text{ako je } \sigma \text{ namijenjen učenju i } su_{NS}(\sigma) \geq Pu \\ 1, \text{ako je } \sigma \text{ namijenjen testiranju i } st_{NS}(\sigma) \geq Pt \end{matrix} \quad (4.81)$$

Ako je stanje od σ jednako 0, onda σ nije posječen, a ako je jednako 1, onda je σ posječen. Na sličan način se za koncept x_j^i objekta nastavnog sadrţaja σ_i definira *posječenost objekta nastavnog sadržaja* funkcijom

$$po_{\sigma_i}(x_j^i) = \begin{matrix} 0, \text{ako je } \sigma \text{ namijenjen učenju i } su_{\sigma_i}(x_j^i) = 0 \\ 0, \text{ako je } \sigma \text{ namijenjen testiranju i } st_{\sigma_i}(x_j^i) > 0 \\ 1, \text{ako je } \sigma \text{ namijenjen učenju i } su_{\sigma_i}(x_j^i) = 0 \\ 1, \text{ako je } \sigma \text{ namijenjen testiranju i } st_{\sigma_i}(x_j^i) > 0 \end{matrix} \quad (4.82)$$

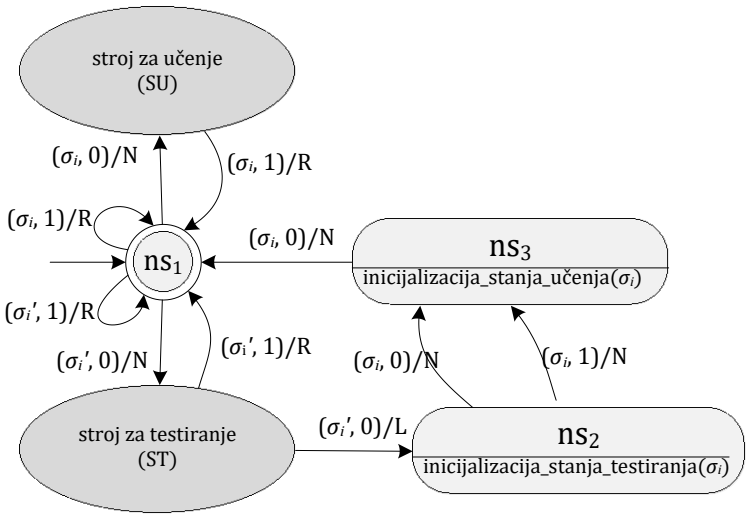
Kao što je u odjeljku 4.6.3 navedeno, nakon postavljanja modela učenika svi koncepti će imati stanje učenja i stanje testiranja jednako 0, što znači da još nisu posječeni. Posječenost objekta nastavnog sadrţaja i posječenost koncepta je bitna informacija koju strojevi implementirani u komponenti za voĎenje modela učitelja koriste za odreĎivanje prijelaza u stanja tijekom faze učenja i testiranja.

4.7.1 Stroj za učenje i testiranje

SUT na osnovu prijeĎenosti objekta nastavnog sadrţaja odreĎuje sljedeće stanje i smjer pomicanja glave za čitanje. Za definiciju SUT-a koriste se

- konačan skup stanja $Q = \{ns_1, ns_2, ns_3, \text{SU}, \text{ST}\}$ gdje je SU sloţeno stanje opisano strojem za učenje, a ST sloţeno stanje opisano strojem za testiranje,
- skup ulaznih znakovi $\Sigma = \{(\sigma_i, 0)\} \cup \{(\sigma_i, 1)\} \cup \{(\sigma_i', 0)\} \cup \{(\sigma_i', 1)\}$ sadrţi ureĎene parove objekata nastavnog sadrţaja i stanja njihove posječenosti,
- početno stanje ns_1,
- skup prihvatljivih stanja $F = \{ns_1\}$,
- skup pomaka glave za čitanje $G = \{\text{L}, \text{R}, \text{N}\}$ gdje L pomiče glavu lijevo, R desno, a N ostavlja glavu u istom poloţaju,
- funkcija prijelaza $\delta: Q \times \Sigma \to Q \times G$ koja za ulazne znakove odreĎuje sljedeće stanje i smjer pomicanja glave za čitanje,
- akcija init_su pridruţena stanju ns_3 i init_st pridruţena stanju ns_2.

Dijagram stanja na slici 4.27 grafički prikazuje prijelaze i stanja SUT-a.



Slika 4.27. Dijagram stanja SUT-a

SUT nakon što pročita ulazni znak, prijelazom određuje sljedeće stanje i smjer pomicanja glave za čitanje. Prelaskom u novo stanje mogu se pokrenuti akcije koje vrše promjenu modela učenika, što utječe na prijeđenost objekta nastavnog sadrţaja. Prijelaz $\delta(ns_1, (\sigma_i, 0)) = (\mathrm{SU}, \mathrm{N})$ opisuje prijelaz SUT-a iz stanja ns_1 u sloţeno stanje SU kada glava za čitanje pročita ulazni znak $(\sigma_i, 0)$. Pri tome glava za čitanje ostaje na istom mjestu. Alternativni prikaz SUT-a se moţe prikazati tablicom prijelaza 4.45.

Tablica 4.45. Tablica prijelaza SUT-a

trenutno stanje	ulazni znak	pomicanje glave	sljedeće stanje	niz akcija s argumentima
ns_1	$(\sigma_i, 0)$	N	SU	
	$(\sigma_i, 1)$	R	ns_1	
	$(\sigma_i', 0)$	N	ST	
	$(\sigma_i', 1)$	R	ns_1	
ns_2	$(\sigma_i, 0)$	N	ns_3	inicijalizacija_stanja_učenja(σ_i)
	$(\sigma_i, 1)$	N	ns_3	
ns_3	$(\sigma_i, 0)$	N	ns_1	
SU	$(\sigma_i, 1)$	R	ns_1	
ST	$(\sigma_i, 0)$	L	ns_2	inicijalizacija_stanja_testiranja(σ_i')
	$(\sigma_i, 1)$	R	ns_1	

Akcija inicijalizacija_stanja_učenja objektu nastavnog sadrţaja namijenjen učenju σ_i postavlja početne vrijednosti stanja učenja pripadnim konceptima, što rezultira postavljanjem posječenosti od σ_i na 0. Pokretanjem ove akcije se mijenja model učenika, a time i znak na ulaznoj traci koji odgovara objektu nastavnog sadrţaja σ_i. Zapravo, od učenika se zahtjeva da ponovno posjeti σ_i. Argument akcije inicijalizacija_stanja_testiranja je objekt nastavnog sadrţaja namijenjenog testiranju σ_i' kojemu se postavljaju početne vrijednosti stanja testiranja pripadnih koncepata.

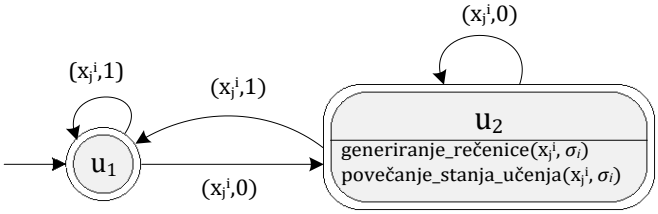
Prijelazom u stanje SU se aktivira stroj za učenje SU koji čita koncepte trenutnog objekta nastavnog sadrţaja. Slično se događa nakon prijelaza u stanje ST čime se aktivira stroj za testiranje ST.

4.7.2 Stroj za učenje

Kada SUT čitanjem objekta nastavnog sadrţaja namijenjenog učenju σ_i prijeĎe u stanje SU, tada stroj SU započne čitanje koncepata x_j^i objekta σ_i. VoĎenje učenja pomoću SU se sastoji od preskakanja posječenih koncepata, a za neposječene koncepte se generiraju rečenice kontroliranog jezika.
Definicija SU sadrţi sljedeće elemente:

- konačni skup stanja $Q = \{u_1, u_2\}$,
- skup ulaznih znakova $\Sigma = \{(x_j^i, 0)\} \cup \{(x_j^i, 1)\}$,
- funkcija prijelaza $\delta: Q \times \Sigma \to Q$,
- početno stanje u_1,
- skup prihvatljivih stanja $F = \{u_1, u_2\}$,
- akcija generiranje_rečenice() i povečanje_stanja_učenja() pridruţenih stanju u_2.

Dijagram stanja 4.28 i tablica prijelaza 4.46 ekvivalentno opisuju prijelaze SU-a.



Slika 4.28. Dijagram stanja SU-a

Tablica 4.46. Tablica prijelaza SU-a

trenutno stanje	ulazni znak	s jedeće stanje	niz akcija s argumentima
u_1	$(x_j^i, 0)$	u_2	generiranje_rečenice (x_j^i, σ_i) povećanje_stanja_učenja(x_j^i, σ^i)
	$(x_j^i, 1)$	u_1	
u_2	$(x_j^i, 0)$	u_2	generiranje_rečenice(x_j^i, σ_i) povećanje_stanja_učenja(x_j^i, σ^i)
	$(x_j^i, 1)$	u_1	

Prijelazom u stanje u_2 se pokreće akcija generiranje_rečenice čiji su argumenti koncept x_j^i objekta nastavnog sadrţaja σ_i za kojeg će se generirati rečenice kontroliranog jezika kao što je opisano u pododjeljku 4.7.2.2.
Druga akcija je povećanje_stanja_učenja koja stanje učenja koncepta x_j^i povećava za 1, čime koncept u objektu nastavnog sadrţaja σ_i postaje posječen.
Generiranje rečenica je proces čiji su ulazni podaci jezično označene trojke područnog znanja. Kod generiranja rečenica tijekom učenja, jezično označene trojke su elementi okoline koncepta područnog znanja.

4.7.2.1 Određivanje ulaznih podataka generatora rečenica

Mehanizam zaključivanja je zasluţan za odreĎivanje jezično označenih trojki područnog znanja koje pripadaju okolini promatranog koncepta. Za dani koncept k mehanizam

zaključivanja vraća sve trojke grafa područnog znanja kojima je subjekt ili objekt koncept *k*. Na osnovu okoline koncepta $\varepsilon(k)$, uvedene u odjeljku 4.3.3, se određuje konceptna restrikcija grafa područnog znanja

$$GPZ\ \varepsilon(k)\ _K = \{(x,y,z)|x = k \vee z = k\} \tag{4.83}$$

Elementima konceptne restrikcije se pridruţuju jezične oznake po definiciji **4.7**, i dobiva se skup

$$jo(GPZ\ \varepsilon(k)\ _K) = \{l[sx,ny,oz]|x = k \vee z = k\} \tag{4.84}$$

Okolina koncepta c_1 iz primjera (4.54), čija je znakovna oznaka $zo(c_1)$=#Aritmeticko_logicka_jednica, ima okolinu

$$\varepsilon(c_1) = \{c_1, c_2, c_3, c_4\} \tag{4.85}$$

, a jezično označena restrikcija grafa područnog znanja na ovu okolinu je

$$jo(GPZ\ \varepsilon(c_1)\ _K) = \{[c_1, r_1, \exists c_2], [c_1, r_2, \forall c_3], [c_1, r_2, \forall c_4]\} \tag{4.86}$$

Ovaj skup jezično označenih trojki predstavlja ulaz u proces generiranja rečenica. Prilikom generiranja, jezično označene trojke se organiziraju temeljem zajedničkih osobina kako bi se odredio redoslijed pretvaranja jezično označenih trojki u apstraktne rečenice kontroliranog jezika.

4.7.2.2 Generiranje rečenica

Kada učenik tijekom učenja dođe do koncepta *k* čiji je status po modelu učenika manji od zadane granične vrijednosti, tada se za koncept *k* generira niz rečenica. Generiranje rečenica je proces u kojemu sudjeluju modul za generiranje rečenica kontroliranog jezika CoLaS podsustava i komponenta za prihvat i prikaz kontroliranog jezika.
Generiranje rečenica se sastoji od sljedećih koraka:

1. svrstavanje jezično označenih trojki područnog znanja u skupine ,
2. generiranje apstraktne rečenice,
3. generiranje rečenice kontroliranog jezika,
4. generiranje teksta rečenice kontroliranog jezika.

Modul za generiranje rečenica svrstava skup jezično označenih trojki u skupine na osnovu zajedničkih osobina, te oblikuje niz jezično označenih trojki koje se transformiraju u apstraktno stablo rečenica. Uključivanjem fraza i morfoloških trojki u apstraktnu rečenicu dobivaju se rečenice kontroliranog jezika. Rečenice kontroliranog jezika je ponekad potrebno dodatno obraditi kako bi se prilikom generiranja teksta dobila gramatičko ispravna rečenica.

4.7.2.2.1 Svrstavanje u skupine

Dobivene jezično označene trojke iz konceptne restrikcije grafa područnog znanja svrstavaju se u skupove na osnovu zajedničkih osobina. Zajedničke osobine jezično označenih trojki koje se uzimaju u razmatranje su

- jednakost jezične oznake veznika,
- jednakost subjekta i jezične oznake broja subjekta,
- jednakost predikata i jezične oznake negativnosti,
- jednakost objekta i jezične oznake broja objekta.

Skupovi jezično označenih trojki koje dijele jednu ili više zajedničkih osobina nazivaju se *skupine jezično označenih trojki*. Neka je *S* skup jezično označenih trojki. U osnovnu skupinu jezično označenih trojki skupa *S* spadaju one trojke koje dijele jednu zajedničku osobinu. Osnovne skupine se označavaju i definiraju na sljedeći način:

- $S\ a,,\ = \{l[sx, ny, oz] || l = a\}$ - skupina po vezniku *a*,
- $S\ ,ab,,\ = \{l[sx, ny, oz] || sx = ab\}$ - skupina po subjektu *ab*,
- $S\ ,,ab,\ = \{l[sx, ny, oz] || ny = ab\}$ - skupina po predikatu *ab*,
- $S\ ,,,ab\ = \{l[sx, ny, oz] || oz = ab\}$ - skupina po objektu *ab*.

Skupine mogu dijeliti dvije ili tri osobine, na primjer $S\ a, bc,,\ = \{l[sx, ny, oz] || l = a \wedge sx = bc\}$ je skupina po vezniku *a* i subjektu *bc*.
Algoritam svrstavanja u skupine skupa jezično označenih trojki u koracima opisuje dobivanje niza jezično označenih trojki na osnovi ulaznog skupa jezično označenih trojki.
Prvi korak algoritma svrstavanja za ulazni skup jezično označenih trojki *S* pronalazi sve skupine po subjektu i po objektu. Neka je *FSS* familija skupina po subjektu, a *FSO* familija skupina po objektu.
U *drugom koraku* algoritma svrstavanja uspoređuju se elementi familije *FSS* i *FSO* te se iz njih izbacuju određene skupine. Za skupine $S\ ,sx,,\ \in FSS$ i $S\ ,,,oz\ \in FSO$ se na osnovu sljedećih pravila određuje koja od njih će biti izbačena iz *FSS* ili *FSO*:

1) ako je $S\ ,sx,,\ \subset S\ ,,,oz$ tada je $FSS = FSS \backslash \{S\ ,sx,,\ \}$
2) ako je $S\ ,,,oz\ \subset S\ ,sx,,$ tada je $FSO = FSO \backslash \{S\ ,,,oz\ \}$
3) ako je $S\ ,sx,,\ = S\ ,,,oz$ tada je $FSO = FSO \backslash \{S\ ,,,oz\ \}$

Neka je *S* jednak skupu jezično označenih trojki iz primjera (4.86). Familije skupina za dani primjer su

$$\begin{aligned} FSS &= S\ ,\exists c_1,,\ = \{\{\ c_1, r_1, \exists c_2\ ,\ c_1, r_2, \forall c_3\ ,\ c_1, r_2, \forall c_4\ \}\} \\ FSO &= S\ ,,,\exists c_2\ ,S\ ,,,\forall c_3\ ,S\ ,,,\forall c_3 \\ &= \{\{\ c_1, r_1, \exists c_2\ \}, \{\ c_1, r_2, \forall c_3\ \}, \{\ c_1, r_2, \forall c_4\ \}\} \end{aligned} \tag{4.87}$$

, a nakon usporedbe elemenata one su

$$\begin{aligned} FSS &= S\ ,\exists c_1,,\ = \{\{\ c_1, r_1, \exists c_2\ ,\ c_1, r_2, \forall c_3\ ,\ c_1, r_2, \forall c_4\ \}\} \\ FSO &= \emptyset \end{aligned} \tag{4.88}$$

Treći korak algoritma za svaku skupinu iz *FSS* i *FSO* generira po dva niza skupina. Za skupinu $S\ ,sx,,$ iz familije *FSS* prvi niz će sadrţavati one skupine iz *FSS* čiji predikat nije individualizacija ind^+ i označavat će se s *NSS*(*sx*). *NSSI*(*sx*) je oznaka niza skupina iz *FSS* koji sadrţe individualizaciju kod predikata. Na sličan način se za skupinu $S\ ,,,oz\ \in FSO$ uvode nizovi *NSO*(*oz*) i *NSOI*(*oz*).
Redoslijed skupina u nizu ovisi o predikatu i jezične oznake negacije predikata. Vrsta uloge predstavljene predikatom jezično označene trojke određuje hoće li skupina biti bliţa početku ili kraju niza. Tablica 4.47 prikazuje prioritet predikata za niz bez individualizacije.

Tablica 4.47. Prioritet predikata za niz skupina bez individualizacije

prioritet	vrsta predikata (uloge)
1	generalizacija gen^+
2	tranzitivne individualne uloge
3	individualne (ne tranzitivne) uloge
4	podatkovne uloge

Na redoslijed skupina utječu jezična oznaka negacije i veznika. Niz skupina se tvori tako što se za svaki prekidat *y* redoslijedom u niz postavljaju skupine oblika $S\ \wedge, sx, \lrcorner y,\ , S\ \vee, sx, \lrcorner y,\ , S\ \wedge, sx, \urcorner y,$ i $S\ \vee, sx, \urcorner y,$. Ove skupine se razlikuju po primjeni jezičnih oznaka veznika i negacije. OdreĎena skupina moţe biti prazan skup pa se ne postavlja u niz. Uvaţavajući prioritet predikata i redoslijed skupina s obzirom na jezične oznake veznika i negacije, za skupinu $S\ , sx, ,\ \in FSS$ se generira opći niz skupina oblika

$$\begin{aligned} &NSS\ sx\ = S\ \wedge, sx, \lrcorner gen^+,\ S\ \vee, sx, \lrcorner gen^+,\ S\ \wedge, sx, \urcorner gen^+,\ S\ \vee, sx, \urcorner gen^+,\\ &S\ \wedge, sx, \lrcorner r_i^+,\ S\ \vee, sx, \lrcorner r_i^+,\ S\ \wedge, sx, \urcorner r_i^+,\ S\ \vee, sx, \urcorner r_i^+,\\ &S\ \wedge, sx, \lrcorner q_j,\ S\ \vee, sx, \lrcorner q_j,\ S\ \wedge, sx, \urcorner q_j,\ S\ \vee, sx, \urcorner q_j,\\ &S\ \wedge, sx, \lrcorner p_k,\ S\ \vee, sx, \lrcorner p_k,\ S\ \wedge, sx, \urcorner p_k,\ S\ \vee, sx, \urcorner p_k, \end{aligned} \tag{4.89}$$

gdje su r_i^+ tranzitivne individualne uloge, q_j individualne (ne tranzitivne) uloge i p_k podatkovne uloge. Niz *NSSI*(*sx*) sadrţi samo individualizaciju kao predikat, stoga će generirani niz biti oblika

$$NSSI\ sx\ = S\ \wedge, sx, \lrcorner ind^+,\ S\ \vee, sx, \lrcorner ind^+,\ S\ \wedge, sx, \urcorner ind^+,\ S\ \vee, sx, \urcorner ind^+, \tag{4.90}$$

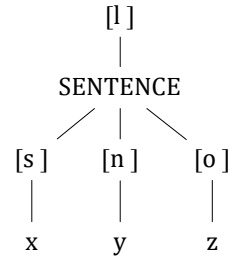
Na istovjetan način se za $S\ , , , oz\ \in FSO$ definiraju generirani oblici nizova *NSO*(*oz*) i *NSOI*(*oz*). U primjeru (4.87) jedino familija skupina *FSS* ima elemente, a za svaki element će se generirati po dva niza. *FSS* iz primjera (4.88) sadrţi samo jednu skupinu, a jezične oznake iz skupine nemaju individualizaciju kao predikat, stoga jedini generirani niz je

$$NSS\ c_1\ = S\ \wedge, c_1, r_1, ,\ S\ \wedge, c_1, r_2, \tag{4.91}$$

Gornji niz skupina je zapisan u skraćenom obliku gdje se skraćeni zapis preuzeo od zapisa jezično označenih trojki opisanog u odjeljku 4.3.2.

4.7.2.2.2 Generiranje apstraktne rečenice

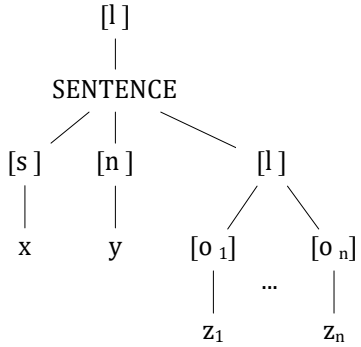
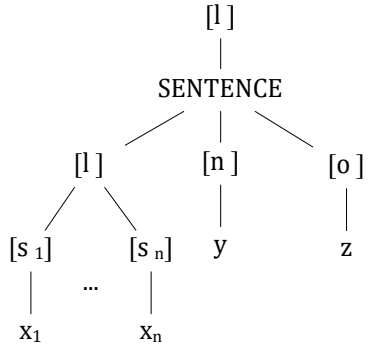
Svrstavanjem u skupine se dobivaju nizovi skupina jezično označenih trojki koje će u ovom koraku sluţiti za generiranje apstraktne rečenice. Svaki niz skupina generira jednu apstraktnu rečenicu. Ako se niz sastoji od jedne skupine $S\ l, sx,$ ny, koja ima samo jednu jezično označenu trojku $l[sx, ny, oz]$ ona se transformira u apstraktnu rečenicu sa slike 4.29.



Slika 4.29. Apstraktna rečenica jezično označene trojke *l*[*sx,ny,oz*]

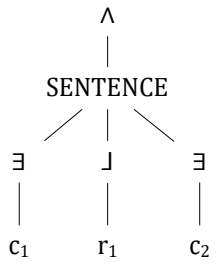
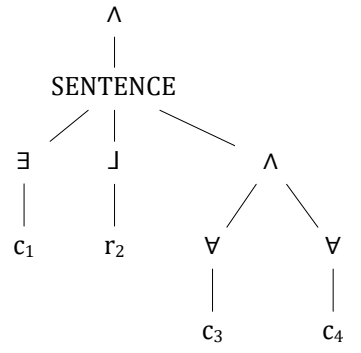
Tablica 4.48 prikazuje apstraktne rečenice za skupine po subjektu i po objektu koje imaju više od jedne jezično označene trojke.

Tablica 4.48.Apstraktne rečenice skupina po subjektu i po objektu koje imaju više od jednog elementa

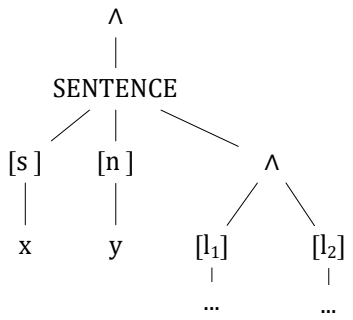
$S\ l, sx, \text{ny},$ $= \{l\ sx, ny, o_1 z_1\ , \dots, l\ sx, ny, o_n z_n\ \}$	$S\ l, , \text{ny}, zo$ $= \{l\ s_1 x_1, ny, oz\ , \dots, l\ s_n x_n, ny, oz\ \}$
	

Za skupine iz primjera (4.91) se generiraju apstraktne rečenice iz tablice 4.49

Tablica 4.49. Primjer generiranih apstraktnih rečenica na osnovu skupina

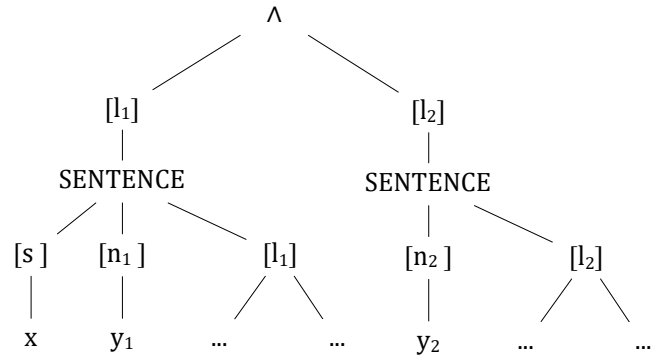
$S\ \wedge, c_1, r_1, ,\ \ = \{\ c_1, r_1, \exists c_2\ \}$	$S\ \wedge, c_1, r_2,\ \ = \{\ c_1, r_2, \forall c_3\ ,\ c_1, r_2, \forall c_4\ \}$
	

Ako se dvije skupine $S\ l_1, sx, \text{ny},\ S\ l_2, sx, \text{ny},$ u nizu razlikuju samo po jezičnoj oznaci veznika ($l_1 \neq l_2$), onda one generiraju jednu apstraktnu rečenicu, kao što je prikazano na slici 4.30.



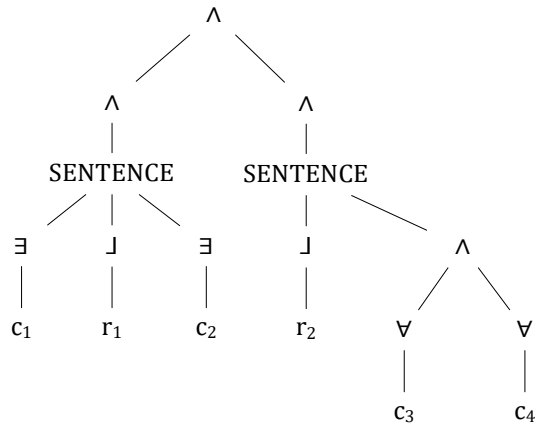
Slika 4.30. Apstraktna rečenica niza skupina *S*⟨l1,sx,ny,⟩ *S*⟨l2,sx,ny,⟩

Dvije skupine $S\ l_1, sx, n_1 y_1,\ S\ l_2, sx, n_2 y_2,$ u nizu koje se razlikuju po predikatu, bez obzira na jezične oznake veznika, generirat će apstraktnu rečenicu sa slike 4.31.



Slika 4.31. Apstraktna rečenica niza skupina *S*⟨l1,sx,n1y1,⟩ *S*⟨l2,sx,n2y2,⟩

Svaki niz sastoji se od skupina u kojemu se dvije susjedne skupine razlikuju po vezniku ili po predikatu, kao što je prikazano kod općih nizova iz (4.89) i (4.90). Jedan opći niz po pravilima sa slike 4.30 i 4.31 generira jednu apstraktnu rečenicu. Za niz skupina iz primjera (4.91) se po istim pravilima generira apstraktno stablo prikazano na slici 4.32.

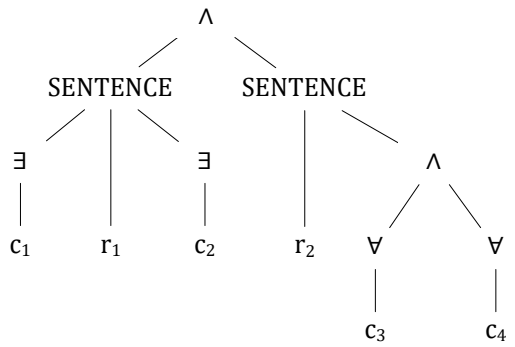


Slika 4.32. Apstraktna rečenica niza skupina *S*⟨∧,c1,r1,⟩ *S*⟨∧,c1,r2,⟩

Apstraktnu rečenicu je ponekad potrebno optimizirati prije nego li se preĎe na sljedeći korak generiranja rečenice kontroliranog jezika. Pod optimizacijom se smatra izbacivanje suvišnih jezičnih oznaka veznika iz apstraktne rečenice, izbacivanje jezične oznake negacije ⌋ i jezične oznake jednakosti =. Veznik se izbacuje ako

- sadrţi samo jednu podgranu
- ako je veznik nadgrane jednak vezniku koji se izbacuje

Jezična oznaka negacije ⊥ i jezična oznaka jednakosti = se uvijek izbacuju. Primjenom pravila optimizacije, za apstraktne rečenice sa slike 4.32 se dobiva apstraktna rečenica sa slike 4.33.



Slika 4.33. Optimizirana apstraktna rečenica

Dobivena optimizirana apstraktna rečenica se u sljedećem koraku transformira u rečenicu kontroliranog jezika.

4.7.2.2.3 Generiranje rečenice kontroliranog jezika

Proces generiranja rečenice kontroliranog jezika iz apstraktne rečenice se odvija u nekoliko koraka:

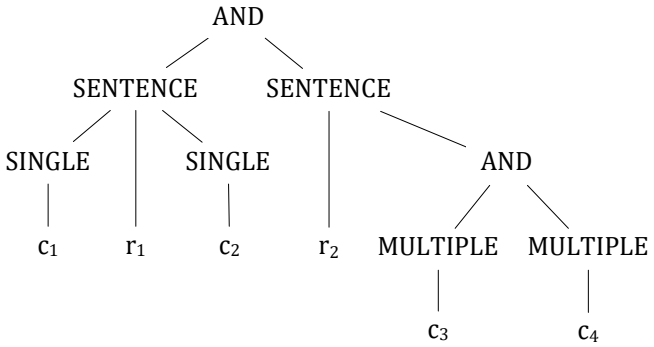
1. zamjena jezičnih oznaka znakovima gramatike rečenica,
2. zamjena elemenata područnog znanja parametriziranim frazama,
3. odreĎivanje vrijednosti parametara parametriziranih fraza,
4. dodatna obrada rečenice kontroliranog jezika.

Dobivena apstraktna rečenica zamjenjuje jezične oznake znakovima gramatike rečenica opisane u odjeljku 4.4.3, a elemente područnog znanja parametriziranim frazama. Tablica 4.50 prikazuje zamjenu jezičnih oznaka znakovima gramatike rečenica

Tablica 4.50. Zamjena jezičnih oznaka znakovima gramatike rečenica

jezična oznaka	znak gramatike rečenica
∧	AND
∨	OR
¬	NOT
∀	MULTIPLE
∃	SINGLE
≤	MAX
≥	MIN

Zamjenom jezičnih oznaka apstraktnog stabla rečenice sa slike 4.33 dobiva se rečenica na slici 4.34.



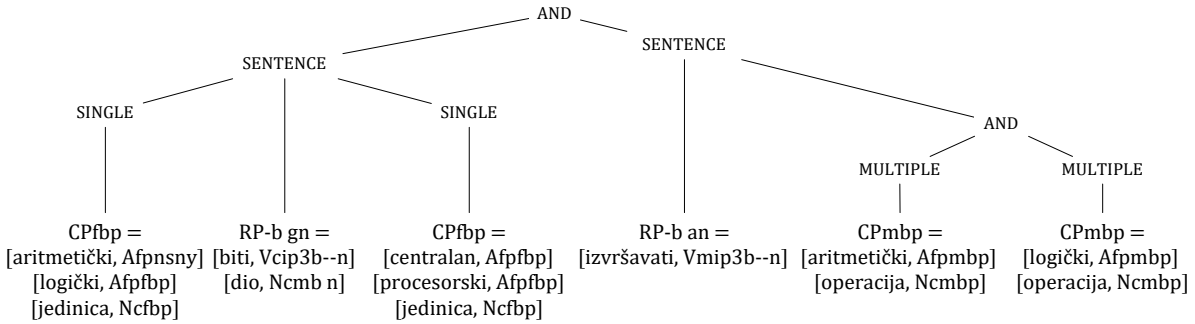
Slika 4.34. Primjer apstraktne rečenice sa zamijenjenim jezičnim oznakama

Zatim slijedi primjena znakovnog označavanja koncepata na osnovu primjera (4.49) i primjera rječnika kontroliranog jezika iz tablice 4.51.

Tablica 4.51. Primjer rječnika kontroliranog jezika

identifikator	parametrizirana fraza
##gen	RP-***bgn*** = [biti, Vcip3***b***--***n***][vrsta, Ncf***b***n]
##~~f~~nd	RP-***bnn*** = [biti, Vcip3***b***--***n***]
##min	[najmanje, Rns]
##max	[najviše, Rns]
##and	[i, Ccs]
##or	[ili, Ccs]
#Aritmeticko_logicka_jedinica	CPf***bp*** = [aritmetički, Afpnsny] [logički, Afpf***bp***] [jedinica, Ncf***bp***]
#je_dio	RP-***b***g***n*** = [biti, Vcip3***b***--***n***] [dio, Ncm***b***n]
#Centralna_procesorska_jedinica	CPf***bp*** = [centralan, Afpf***bp***] [procesorski, Afpf***bp***] [jedinica, Ncf***bp***]
#izvršava	RP-***ba n*** = [biti, Vmip3***b***--***n***]
#Aritmeticka_operacija	CPf***bp*** = [aritmetički, Afpf***bp***] [operacija, Ncm***bp***]
#Logicka_operacija	CPf***bp*** = [logički, Afpf***bp***] [operacija, Ncm***bp***]

Pošto koncept c_1 ima znakovnu oznaku #Aritmeticko_logicka_jedinica, a ovoj znakovnoj oznaci je u rječniku kontroliranog jezika pridruţena parametrizirana fraza iz tablice 4.51, onda se daljnjom zamjenom ostalih elemenata područnog znanja dobiva stablo sa slike 4.35.



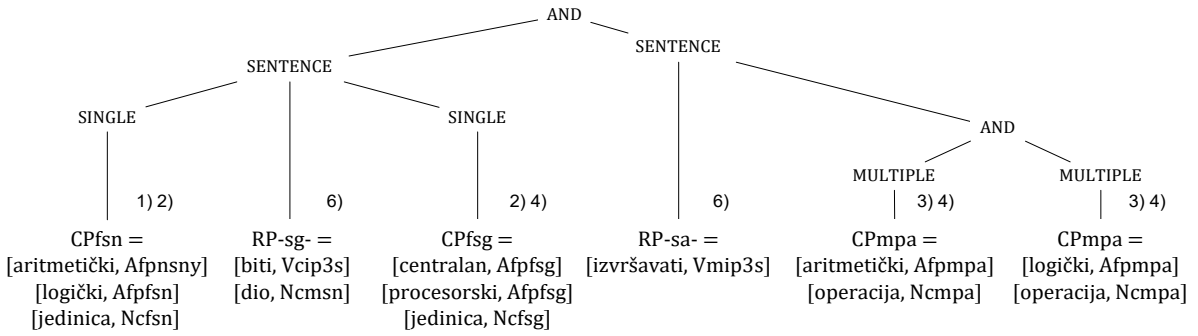
Slika 4.35. Primjer apstraktne rečenice sa zamijenjenim elementima područnog znanja

Parametrizirane fraze nalaze se uvijek u listovima apstraktne rečenice i za njih je potrebno odrediti vrijednosti parametara. Dobivena apstraktna rečenica moţe se sastojati od više SENTENCE elemenata. Gledano s lijeva prema desno, prvi SENTENCE element će uvijek imati subjektnu granu. Ako SENTENCE element nema subjektnu granu, onda se smatra da je

njegova subjektna grana jednaka subjektnoj grani prvog njemu lijevog SENTENCE elementa koji ima subjektnu granu.
Vrijednosti parametara se određuju na osnovu sljedećih pravila:

1) Ako SENTENCE ima subjektnu granu, onda će padeţ svake parametrizirane fraze u subjektnoj grani biti nominativ.
2) Ako se parametrizirana fraza nalazi u SINGLE grani, onda je ona u jednini.
3) Ako se parametrizirana fraza nalazi u MULTIPLE grani, onda je ona u mnoţini.
4) Padeţ parametriziranih fraza u objektnoj grani SENTENCE elementa jednak je padeţu parametrizirane fraze u predikatnoj grani istog SENTENCE elementa.
5) Ako SENTENCE element ima subjektnu granu, onda je rod parametrizirane fraze u predikatnoj grani jednak rodu posljednje parametrizirane fraze iz subjektne grane. U slučaju kada SENTENCE element nema subjektnu granu, tada se razmatra prvi njemu lijevi SENTENCE element koji ima subjektnu granu.
6) Ako SENTENCE element ima subjektnu granu, onda je broj parametrizirane fraze u predikatnoj grani jednina kada subjektna grana ima samo jednu parametriziranu frazu, inače je mnoţina. U slučaju kada SENTENCE element nema subjektnu granu, tada se razmatra prvi njemu lijevi SENTENCE element koji ima subjektnu granu.
7) Ako se parametrizirana fraza nalazi u NOT grani, onda se mijenja parametar negativnosti parametrizirane fraze u njemu suprotan.

Slika 4.36 prikazuje rezultat primjene pravila za odreĎivanje vrijednosti parametara na rečenicu sa slike 4.35.



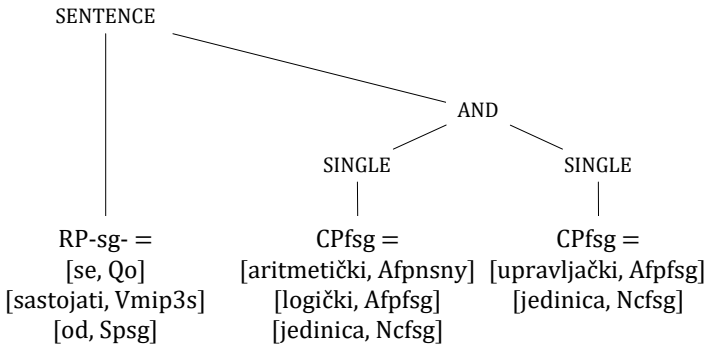
Slika 4.36. Primjer rečenice kontroliranog jezika s postavljenim parametrima parametriziranih fraza

Primjenom pravila 1) i 2) prva parametrizirana fraza iz primjera sa slike 4.35 se postavlja u nominativ jednine. Pošto prvi SENTENCE element ima samo jednu parametriziranu frazu, onda druga parametrizirana fraza primjenom pravila 6) se postavlja u jedninu. Primjenom istog pravila se i četvrta parametrizirana fraza u drugom SENTENCE elementu postavlja u jedninu. Padeţ treće parametrizirane fraze se odreĎuje primjenom pravila 4) s obzirom na padeţ druge parametrizirane relacijske fraze. Primjenom pravila 4) se odreĎuju padeţi pete i šeste parametrizirane fraze s obzirom na padeţ četvrte parametrizirane relacijske fraze. Pravilom 3) se iste fraze postavljaju u mnoţinu.

4.7.2.2.4 Dodatna obrada rečenice kontroliranog jezika

U dodatnu obradu rečenice kontroliranog jezika ubrajaju se premještanje postojećih riječi unutar relacijske fraze i ubacivanje riječi izmeĎu postojećih riječi relacijske fraze. Ubacivanje riječi unutar relacijske fraze SENTENCE elementa ovisi o strukturi elementa i o kategorizaciji i vremenu relacijske fraze. Ako SENTENCE element nema subjektnu granu i

ako je relacijska fraza sadašnjeg vremena bez čestice "se" i ako nema imensku frazu, onda je potrebno premjestiti parametrizirane morfološke trojke relacijske fraze. Kod rečenice kontroliranog jezika sa slike 4.37 će se dogoditi premještanje parametriziranih morfoloških trojki.



Slika 4.37. Primjer rečenice bez subjektne grane s relacijskom frazom sa "se"

Relacijska fraza SENTENCE elementa bez subjektne grane spada u relacijsku frazu sadašnjeg vremena s česticom "se" i kod nje će doći do premještanja. Za ovu vrstu relacijske fraze čestica "se" premješta se iza indikativa prezenta glavnog glagola i dobiva se relacijska fraza

RP-sg-=[sastojati, Vmip3s] [se, Qo] [od, Spsg] (4.92)

U tablici 4.17 iz pododjeljka 4.4.2.2 je naveden opći izraz relacijske fraze sa "se" u sadašnjem vremenu, a izraz

<čestica "ne">? <indikativ prezenta glavnog glagola> <čestica "se"> <prijedlog>? (4.93)

opisuje relacijsku frazu nakon premještanja. Po ovom izrazu i za ovu vrstu relacijske fraze čestica "se" se prebacuje iza indikativa prezenta glavnog glagola. Prebacivanje morfoloških trojki kod relacijske fraze sa "se" ne ovisi samo o vremenu, nego i o tome je li fraza u negativnom obliku, kao što je prikazano tablicom 4.52.

Tablica 4.52. Prebacivanje morfoloških trojki kod relacijske fraze sa "se"

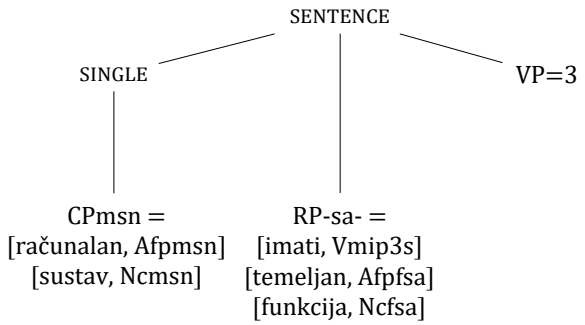
	pozitivni oblik	**negativni oblik**
sadašnje	<indikativ prezenta glavnog glagola> <čestica "se"> <prijedlog>?	<čestica "ne">? <indikativ prezenta glavnog glagola> <čestica "se"> <prijedlog>?
prošlo	<particip prošli glavnog glagola> <čestica "se"> <prijedlog>?	<indikativ prezenta veznog glagola>? <čestica "se"> <particip prošli glavnog glagola> <prijedlog>?
buduće	<infinitiv glavnog glagola> <indikativ prezenta pomoćnog glagola "htjeti"> <čestica "se"><prijedlog>?	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "se"> <infinitiv glavnog glagola> <prijedlog>?

Ostale relacijske fraze (bez čestice "se") ne zavise o negativnom obliku, već samo o vremenu. Tablica 4.53 opisuje premještanje morfoloških trojki relacijske fraze koja se pojavi u SENTENCE elementu bez subjektne grane.

Tablica 4.53. Prebacivanje morfoloških trojki kod relacijske fraze bez "se"

sadašnje	<imenska fraza> <indikativ prezenta veznog glagola> <prijedlog>?
prošlo	(<particip prošli veznog glagola "biti"> | < particip prošli glavnog glagola >) <indikativ prezenta veznog glagola> <imenska fraza>? <prijedlog>?
buduće	(<infinitiv veznog glagola "biti"> | < infinitiv glavnog glagola >) <indikativ prezenta pomoćnog glagola "htjeti"> <imenska fraza>? <prijedlog>?

Za sadašnje vrijeme, jedino kod relacijskih fraza 3 i 4 iz tablice 4.12 se provodi prebacivanje. Svih osam kategorija relacijskih fraza u prošlom i budućem vremenu uvijek prebacuju morfološke trojke.
Dodatna obrada rečenice kontroliranog jezika je potrebna i kod SENTENCE elemenata koji u objektnoj grani sadrţe neke od sljedećih elemenata: MIN, MAX elemente i vrijednosti podataka. Ako ovakav SENTENCE element sadrţi relacijsku frazu sedme kategorije, onda će doći do dodatne obrade relacijske fraze. Na slici 4.38 je prikazana rečenica koja u objektnoj grani ima vrijednost podataka, a predikatna grana sadrţi relacijsku frazu sedme kategorije.

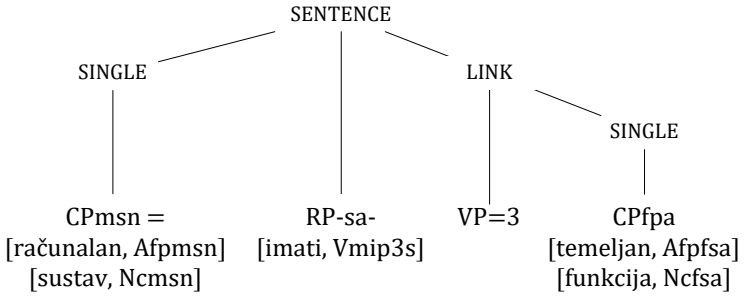


Slika 4.38. Primjer rečenice s relacijskom frazom sedme kategorije i s vrijednostima podataka

Relacijska fraza kategorije 7 se rastavlja na relacijsku frazu kategorije 5 i konceptnu frazu. Za relacijsku frazu sa slike 4.38 se dobivaju fraze

$$\begin{matrix} \text{RP-sa-=[imati, Vmip3s]} \\ \text{CPfsa=[temeljan, Afpfsa] [funkcija, Ncfsa]} \end{matrix} \tag{4.94}$$

Nadalje, SENTENCE element u objektnoj grani postavlja LINK element kojim se povezuje konceptna fraza dobivena rastavljanjem početne relacijske fraze s vrijednosti podataka, kao što je prikazano na slici 4.39.



Slika 4.39. Rezultat obrade relacijske fraze sedme kategorije

U objektnoj grani je potrebno odrediti broj i padeţ konceptne fraze, što ovisi o elementima objektne grane. Ako je prvi list objekte grane brojčana vrijednost podatka, onda se u obzir

uzimaju dvije posljednje znamenke (iza decimalne točke). Na osnovu tablice 4.54 se određuje padeţ i broj konceptne fraze u novoj grani.

Tablica 4.54. Promjena padeţa i broja konceptne fraze na osnovu znamenki brojčane vrijednosti podatka

posljednje dvije znamenke brojčane vrijednosti podatka	padeţ	broj
0	genitiv	mnoţina
1	akuzativ	jednina
2 - 4	akuzativ	jednina
5 - 20	genitiv	mnoţina
21, 31, …, 91	akuzativ	jednina
22 - 24, 32 - 34, …, 92 - 94	akuzativ	mnoţina
25 - 30, 35 - 40, …, 95 - 99	genitiv	mnoţina

Ako vrijednost podataka nije broj, onda ne dolazi do promjene padeţa i broja konceptne fraze. MeĎutim, takva rečenica kontroliranog jezika moţe biti gramatički nepravilna jer je vrijednost podataka niz znakova koji nije u nikakvom morfološki zavisnom odnosu s elementima rečenice.

4.7.2.2.5 Generiranje teksta rečenice kontroliranog jezika

Nakon dodatne obrade relacijske fraze, svaka rečenica kontroliranog jezika se transformira u tekst. *Tekst* je niz Unicode znakova i na njemu je definiran operator povezivanja "+" koji dva niza znakova pretvara u jedan niz znakova nadovezivanjem znakova drugog niza na znakove prvog niza. Radi označavanja početka i kraja niza znakova, tekst se omeĎuje navodnicima.
Funkcija text pretvara rečenicu kontroliranog jezika u niz znakova, odnosno tekst. Domena funkcije text su izrazi nastali po produkcijama gramatike rečenica, opisane u odjeljku 4.4.3, koji uključuju samo završne znakove. Tablica 4.55 definira funkciju text.

Tablica 4.55. Funkcija text

izraz	text(izraz)	
AND[T_1 … T_n]	text(T_1) + " i " + text(T_2)	n=2
	text(T_1) + ", " + text(T_2) + ", " + … + " i " + text(T_n)	n > 2
OR[T_1 … T_n]	text(T_1) + " ili " + text(T_2)	n=2
	text(T_1) + ", " + text(T_2) + ", " + … + " ili " + text(T_n)	n > 2
NOT[T]	text(T)	
LINK[T_1 … T_n]	text(T_1) + " " + … + " " + text(T_n)	
SENTENCE[T_1 …T_n]	text(T_1) + " " + … + " " text(T_n)	$2 \leq n \leq 3$
SINGLE[T]	text(T)	
MULTIPLE[T]	text(T)	
MIN[T]	"najmanje " + text(T)	
MAX[T]	"najviše " + text(T)	
VP=""	"…"	
CP=[y_1, z_1] … [y_n, z_n]	x_1 + " " + … + " " + x_n	[y_i, z_i] parametrizirana morfološka trojka, (x_i, y_i, z_i) morfološka trojka
RP=[y_1, z_1] … [y_n, z_n]	x_1 + " " + … + " " + x_n	[y_i, z_i] parametrizirana morfološka trojka, (x_i, y_i, z_i) morfološka trojka

U prvom stupcu tablice 4.55 su navedeni izrazi nastali po produkcijama gramatike rečenica, a drugi stupac prikazuje njihovu transformaciju u tekst korištenjem funkcije text. Izrazi koriste varijablu T kako bi se označio ugnijeţĎeni izraz promatranog izraza. Funkcija text preslikava konceptnu frazu CP i relacijsku frazu RP opisanu nizom parametriziranih trojki [y_i, z_i], gdje je y_i lema, a z_i morfosintaktički opis, u oblik riječi x_i, koji je ujedno i tekst. Pri tome se morfološka trojka (x_i, y_i, z_i) pronalazi u rječniku kontroliranog jezika na osnovu leme y_i i morfosintaktičkog opisa z_i. Ako rječnik kontroliranog jezika uključuje morfološke trojke iz tablice 4.51, tada vrijedi

text([jedinica, Ncfsg])= "jedinice" (4.95)

Ukoliko se rečenica kontroliranog jezika sa slike 4.36 opiše produkcijama gramatike rečenica, dobiva se izraz

```
AND[
 SENTENCE[
  SINGLE[ (CPfsn=[aritmetički, Afpnsny] [logički, Afpfsn] [jedinica, Ncfsn]) ]
  (RP-sg-=[biti, Vcip3s] [dio, Ncmsn])
  SINGLE[ (CPfsg=[centralan, Afpfsg] [procesorski, Afpfsg] [jedinica, Ncfsg]) ] ]
 SENTENCE[
  (RP-sa-=[izvršavati, Vmip3s])
  AND[
   MULTIPLE[ (CPmpa=[aritmetički, Afpmfa] [operacija, Ncmfa]) ]
   MULTIPLE[ (CPmpa=[aritmetički, Afpmfa] [operacija, Ncmfa]) ] ] ]
```
(4.96)

Primjenom funkcije text na listove rečenice kontroliranog jezika dobiva se sljedeći niz znakova

```
text(CPfsn=[aritmetički, Afpnsny] [logički, Afpfsn] [jedinica, Ncfsn]) + " " +
text(RP-sg-=[biti, Vcip3s] [dio, Ncmsn]) + " " +
text(CPfsg=[centralan, Afpfsg] [procesorski, Afpfsg] [jedinica, Ncfsg]) + " i " +
text(RP-sa-=[izvršavati, Vmip3s]) + " " +
text(CPmpa=[aritmetički, Afpmfa] [operacija, Ncmfa]) + " i " +
text(CPmpa=[aritmetički, Afpmfa] [operacija, Ncmfa])
```
(4.97)

Pronalaţenjem oblika riječi iz rječnika kontroliranog jezika se dobiva sljedeći niz znakova.

```
"aritmetičko" + " " + "logička" + " " + "jedinica" + " " +
"je" + " " + "dio" + " " +
"centralne" + " " + "procesorske" + " " + "jedinice" + " i " +
"izvršava" + " "
"aritmetičke" + " " + "operacije" + " i " +
"logičke" + " " + "operacije"
```
(4.98)

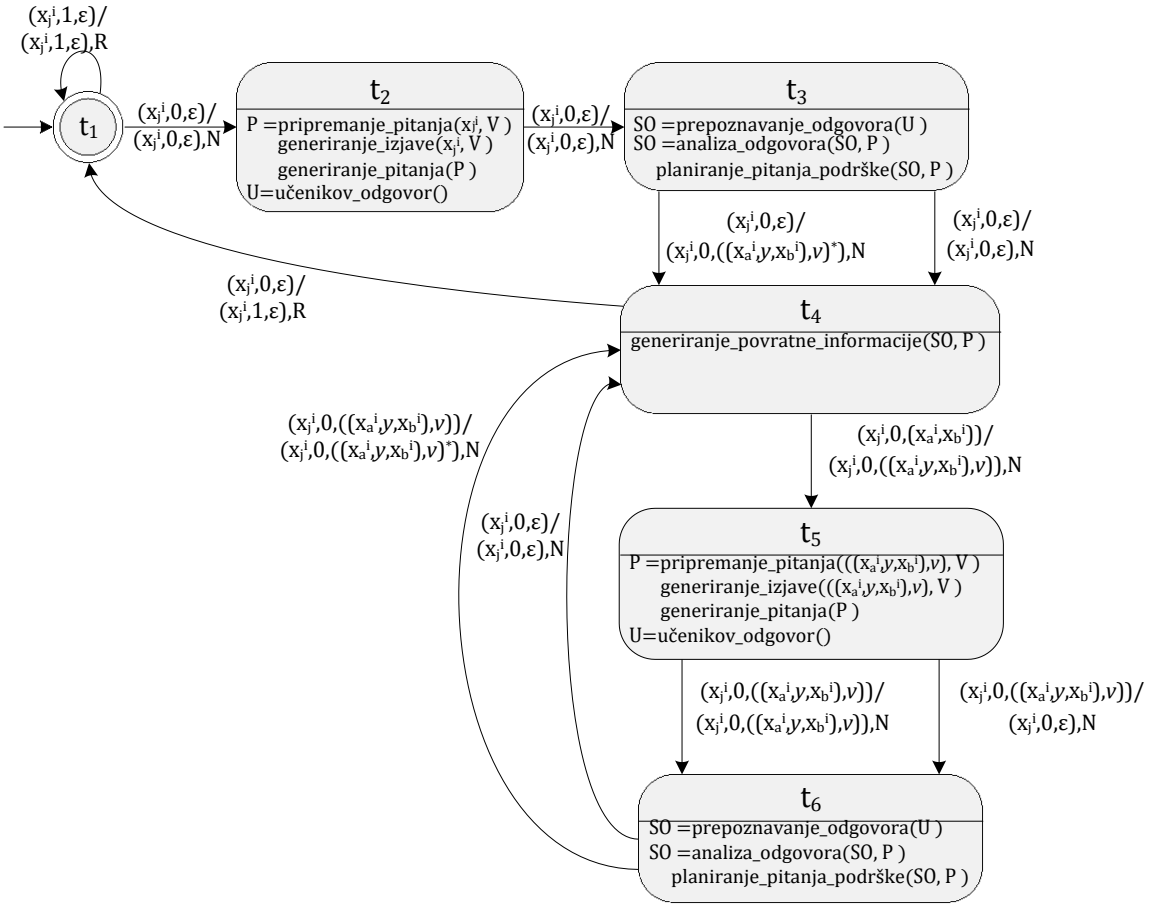
Primjenom operatora povezivanja dobiva se konačni niz znakova u kojemu se prvi znak zamjenjuje velikim slovom, a iza zadnjeg znaka se postavlja znak "." ako se radi o izjavnoj rečenici, odnosno znak "?" za upitnu rečenicu. Ciljni tekst generirane rečenice je

"Aritmetičko logička jedinica je dio centralne procesorske jedinice i izvršava aritmetičke operacije i logičke operacije." (4.99)

Generiranje rečenice kontroliranog jezika je završeno primjenom funkcije text. Rečenice se generiraju tijekom realizacije učenja, ali i tijekom testiranja znanja učenika.

4.7.3 Stroj za testiranje i tutorski dijalog

Testiranje u CoLaB Tutor-u se ostvaruje tutorskim dijalogom gdje tutor postavlja pitanja, prepoznaje i analizira učenikov odgovor, daje povratnu informaciju i planira pitanja podrške. Za voĎenje tutorskog dijaloga je odgovoran stroj za testiranje ST kojemu se znakovi ulazne trake sastoje od koncepata x_j^i objekta nastavnog sadrţaja namijenjenog testiranju objekta nastavnog sadrţaja $\sigma_i^{'}$, stanja posječenosti koncepta i niza elemenata $((x_{\mathrm{a}}^{\mathrm{i}}, y, x_{\mathrm{b}}^{\mathrm{i}}), v)$ kojima se opisuju pitanja podrške. Pitanje podrške je opisano trojkom $(x_{\mathrm{a}}^{\mathrm{i}}, y, x_{\mathrm{b}}^{\mathrm{i}})$ grafa područnog znanja i vrstom pitanja *v* koje će biti pojašnjeno u pododjeljku 4.7.3.1. Opisi pitanja podrške se postavljaju na stog ST-a. Inicijalno je stog prazan jer se u početnom stanju ST-a nikada ne postavlja pitanje podrške. Slika 4.40 opisuje stanja i prijelaze ST-a.



Slika 4.40. Dijagram stanja ST-a

ST objekta nastavnog sadrţaja namijenjen testiranju je definiran sljedećim elementima:

- konačni skup stanja $Q = \{t_1, t_2, t_3, t_4, t_5, t_6\}$
- skup ulaznih i izlaznih znakova $\Sigma = \{(x_{\mathrm{j}}^{\mathrm{i}}, 0)\} \cup \{(x_{\mathrm{j}}^{\mathrm{i}}, 1)\}$
- skup znakova potisnog stoga $\Gamma = \{((x_{\mathrm{a}}^{\mathrm{i}}, y, x_{\mathrm{b}}^{\mathrm{i}}), v)\}$ gdje je $(x_{\mathrm{a}}^{\mathrm{i}}, y, x_{\mathrm{b}}^{\mathrm{i}})$ trojka grafa područnog znanja, a $v \in \{s, p, o\}$ označava vrstu pitanja,

- skup pomaka glave za čitanje $G = \{\mathrm{L}, \mathrm{R}, \mathrm{N}\}$ gdje L pomiče glavu lijevo, R desno, a N ostavlja glavu u istom poloţaju,
- funkcija prijelaza $\delta: Q \times \Sigma \times \Gamma \rightarrow Q \times \Sigma \times \Gamma^* \times G$
- početno stanje t_1
- skup prihvatljivih stanja $F = \{t_1\}$
- skup posječenih trojki pitanja *V*.
- postavljeno pitanje *P*
- učenikov odgovor *U*
- stanje odgovora *SO*

ST sadrţi, osim potisnog stoga, i skup posječenih trojki grafa područnog znanja *V* u kojemu se nalaze one trojke koje su sluţile za generiranje pitanja. TakoĎer ST čuva u varijabli *P* postavljeno pitanje, *U* sluţi za spremanje teksta učenikovog odgovora, a *SO* je stanje odgovora nakon provedene analize točnosti.
U početnom stanju t_1 se odlučuje koji je sljedeći koncept za kojega će se generirati glavno pitanje. Koncept $x_{\mathrm{j}}^{\mathrm{i}}$ će biti sljedeći ako mu je stanje posječenosti 0, a ako mu je stanje 1 onda će se traţiti prvi sljedeći neposječeni koncept. Prelaskom u stanje t_2 se za koncept $x_{\mathrm{j}}^{\mathrm{i}}$ generira glavno pitanje i čeka se na učenikov odgovor. Kada učenik odgovori, prelazi se u stanje t_3 što rezultira analizom odnosa tutorskog točnog odgovora i učenikovog odgovora. Ako učenik nije u potpunosti točno odgovorio na postavljeno pitanje, tada se na potisni stog stavljaju ureĎeni parovi trojke i vrste pitanja $((x_{\mathrm{a}}^{\mathrm{i}}, y, x_{\mathrm{b}}^{\mathrm{i}}), v)$ kojima se opisuju pitanja podrške i prelazi se u stanje t_4. Prijelaz u stanje t_4 se ostvaruje i ako je učenik potpuno točno odgovorio na pitanje, ali tada ne dolazi do postavljanja opisa pitanja podrške na stog. U ovom slučaju se glava pomiče za jedno mjesto udesno i na ulaznu traku se piše $(x_{\mathrm{j}}^{\mathrm{i}}, 1, \varepsilon)$ što znači da je stanje posječenosti koncepta $x_{\mathrm{j}}^{\mathrm{i}}$ jednako 1, a ε označava kako se u stog ne postavljaju elementi. Simbol R nareĎuje glavi za čitanje da se pomakne jedno mjesto udesno, odnosno slijedi čitanje sljedećeg koncepta. Povratna informacija o uspješnosti učenikovog odgovora se generira u stanju t_4 i tada se na osnovu sadrţaja potisnog stoga prelazi u početno stanje t_1 ili u stanje t_5. U stanje t_1 se prelazi ako je stog prazan, a ako nije tada se uzima vrh stoga koji će sluţiti za generiranje pitanja podrške i prelazi se u stanje t_5. Stanja t_4 i t_5 su slična stanjima t_2 i t_3, odnosno sluţe za generiranje pitanja i analizu odgovora, s razlikom što se generirano pitanje temeljilo na sadrţaju stoga, a ne na konceptu s ulazne trake. Nakon provedene analize se, slično kao i kod stanja t_3, moţe nadopuniti potisni stog što će rezultirati novim ciklusima pomoćnih pitanja. MeĎutim, ako je učenik točno odgovorio na sva pomoćna pitanja, stog će se isprazniti i poslije generiranja povratne informacije u stanju t_4, prelazi se u početno stanje t_1 koje je ujedno i završno stanje.
Tablica 4.56 opisuje prijelaze iz jednog stanja u drugo stanje ovisno o pročitanom ulaznom znaku i vrhu stoga. MeĎutim sljedeće stanje se ne moţe unaprijed odrediti dok se ne izvrše akcije koje mogu utjecati na prijelaze ST-a.

Tablica 4.56. Tablica prijelaza ST-a

stanje	ulazni znak	vrh stoga	sljedeće stanje	izlazni znak	riječ stoga	pomak glave	niz akcija s argumentima
t_1	$(x_j^i, 1)$	ε	t_1	$(x_j^i, 1)$	ε	R	
	$(x_j^i, 0)$	ε	t_2	$(x_j^i, 0)$	ε	N	
t_2	$(x_j^i, 0)$	ε	t_3	$(x_j^i, 0)$	ε	N	P =pripremanje_pitanja(x_j^i, V) generiranje_izjave(x_j^i, V) generiranje_pitanja(P) U=učenikov_odgovor
t_3	$(x_j^i, 0)$	ε	t_4	$(x_j^i, 0)$	$((x_a^i, y, x_b^i), v)^*$	N	SO =prepoznavanje_odgovora(U) SO =analiza_odgovora(SO, P) planiranje_pitanja_podrške SO, P)
	$(x_j^i, 0)$	ε	t_4	$(x_j^i, 0)$	ε	N	
t_4	$(x_j^i, 0)$	(x_a^i, y, x_b^i)	t_5	$(x_j^i, 1)$	$((x_a^i, y, x_b^i), v)$	R	generiranje_povratne_informacije(SO, P)
t_5	$(x_j^i, 0)$	(x_a^i, y, x_b^i)	t_6	$(x_j^i, 0)$	$((x_a^i, y, x_b^i), v)^*$	N	P =pripremanje_pitanja(((x$_a^i$,y,x$_b^i$),v), V) generiranje_izjave(((x$_a^i$,y,x$_b^i$),v), V) generiranje_pitanja(P) U=učenikov_odgovor
	$(x_j^i, 0)$	(x_a^i, y, x_b^i)	t_6	$(x_j^i, 0)$	ε	N	
t_6	$(x_j^i, 0)$	ε	t_4	$(x_j^i, 0)$	ε	N	SO =prepoznavanje_odgovora(U) SO =analiza_odgovora(SO, P) planiranje_pitanja_podrške SO, P)
	$(x_j^i, 0)$	(x_a^i, y, x_b^i)	t_4	$(x_j^i, 0)$	$((x_a^i, y, x_b^i), v)$	N	

Sva stanja osim t_1 izvršavaju akcije koje su definirane ulaznim i izlaznim argumentima. U stanju t_2 se prvo provodi priprema pitanja na osnovu ulaznog koncepta x_j^i i skupa posječenih koncepata *V*. Kao rezultat se dobije pitanje *P* nakon čega slijedi generiranje izjave koja također ovisi o konceptu x_j^i i skupu *V*. Generiranje pitanja je sljedeća akcija stanja t_2 nakon koje se čeka na učenikov odgovor koji se sprema u varijablu *U*. U sljedećem stanju t_3 se prepoznaje učenikov odgovor *U* i rezultat prepoznavanja se sprema u stanje odgovora *SO*. Sljedeća akcija stanja t_3 je analiza odgovora koja, usporedbom dosadašnjeg stanja odgovora i pitanja *P*, nadopunjava stanje odgovora. Eventualno se nakon planirana pitanja podrške generira niz za opisivanje pitanja podrške ((x^a_i, *y*, x^b_i), *v*)* koji će, se prijelazom u sljedeće stanje, postaviti na potisni stog. Stanje t_4 sluţi za generiranje povratne informacije na osnovi stanja odgovora i pitanja, čime se učenik obavještava o točnosti svog odgovora. Akcije stanja t_5 su slične akcijama stanja t_2, s razlikom što se u stanju t_5 akcije pripremanja pitanja i generiranja izjave temelje na pitanju podrške s vrha stoga. U tablici 4.57 se daje primjer dijaloga u kojemu su tutorske izjave i učenikov odgovor popraćeni slijedom stanja i akcija ST-a.

Tablica 4.57. Primjer dijaloga

dijalog		akcija	stanje
		pripremanje_pitanja generiranje_izjave	t_2
Tutor:	Tko ili što su vrste memorije?	generiranje_pitanja	
Učenik:	radna memorija, disketa, RAM i monitor	učenikov_odgovor	
		prepoznavanje_odgovora analiza_ odgovora planiranje_pitanja_podrške	t_3
Tutor:	Radna memorija, disketa i RAM su vrste memorije, ali disketa i RAM nisu traţeni odgovor. Monitor nije vrsta memorije.	generiranje_povratne_informacije	t_4
Tutor:	RAM je vrsta radne memorije.	pripremanje_pitanja generiranje_izjave	t_5
Tutor:	Tko ili što još je vrsta radne memorije?	generiranje_pitanja	

U stanju t_2 iz primjera dijaloga se prvo pripremilo pitanje, za kojega nije bilo potrebno generirati uvodnu izjavu, te je generirano pitanje. Nakon što je učenik unio odgovor, u stanju t_3 se prepoznao tekst učenikovog odgovora te se analiziralo stanje odgovora. Pošto učenikov odgovor nije u potpunosti točan, onda se izvršilo planiranje pitanja podrške. Kao rezultat planiranja se, prijelazom u stanje t_4, na stog postavio niz kojim se opisuju naredna pitanja podrške. U stanju t_4 se generirala povratna informacija kojom se učenika obavijestilo o valjanosti njegovog odgovora. Prelaskom u stanje t_5 se za opis pitanja podrške sa stoga ponovio ciklus pripremanja pitanja, generiranja izjave i generiranja pitanja, kao i kod stanja t_2. Akcije ST-a su opisane u narednim pododjeljcima ovog odjeljka.

4.7.3.1 Pripremanje pitanja

U pododjeljku 2.4.2.3 je opisana podjela upitnih rečenica i s obzirom na ovu podjelu pitanja se dijele na subjektna, predikatna i objektna pitanja. Ulaz procesa generiranja pitanja je skup jezično označenih trojki grafa područnog znanja i vrsta pitanja koje se želi generirati. Općenito, pitanje P se može opisati kao uređeni par $P = (JT, v)$ gdje je JT skup jezično označenih trojki, a $v \in \{s, p, o\}$ gdje s označava subjektno pitanje, p predikatno pitanje i o objektno pitanje. Sve jezično označene trojke $l[sx, ny, oz]$ skupa JT uvijek imaju istu predikatnu relaciju y, oznaku negacije predikata n i oznaku veznika l. Ako JT ima više od jednog elementa, onda je ili subjektni koncept x ili objektni koncept z uvijek isti kod svih trojki. S obzirom na ovo svojstvo, pitanja se mogu podijeliti na *pitanja s fiksnim subjektom* i na *pitanja s fiksnim objektom*.

Pripremanje pitanja prethodi generiranju pitanja i odvija se u stanjima t_2 i t_5 stroja za testiranje ST. Kada je ST u stanju t_2, argumenti pripreme pitanja su koncept s ulazne trake c i skup posječenih trojki grafa područnog znanja V. Za koncept c se uzimaju trojke grafa područnog znanja koje se nalaze u konceptnoj restrikciji okoline koncepta c, odnosno

$$T = GPZ\ \varepsilon(c) \cup \{c\}_K \tag{4.100}$$

Iz skupa T se slučajnim odabirom uzima predikatna relacija r, i na osnovu odabrane relacije se uzima podskup od T koji nije u V i koji se, preko funkcije jezičnog označavanja, preslikava u skup JT pitanja P, odnosno

$$JT = jo(\{(x, y, z) \in T \mid y = r\} \setminus V) \tag{4.101}$$

Vrsta pitanja se u stanju t_2 također odabire slučajnim odabirom iz skupa $\{s, o\}$, odnosno generirano pitanje će biti subjektno ili objektno.

Kad je ST u stanju t_5, tada argumenti pripremanja pitanja su, osim skupa V, i vrh potisnog stoga Γ. Uređeni par $((x, r, z), v)$ s vrha stoga već unaprijed definira vrstu pitanja v i relaciju r. Ako je $v = p$, odnosno ako se radi o predikatnom pitanju, onda je $JT = jo(x, r, z)$. Međutim, ako se radi o subjektnom pitanju, tada se skup jezičnih trojki pitanja određuju po formuli

$$JT = jo(GPZ\ \varepsilon^{\downarrow}(x) \cup \{x\}_K \cap GPZ\ \{r\}_R \setminus V) \tag{4.102}$$

odnosno uzimaju se jezično označene trojke iz presjeka konceptne restrikcije donje okoline subjektnog koncepta x i relacijske restrikcije od r. Kod objektnog pitanja se umjesto subjektnog koncepta uzima objektni koncept z i njegova gornja okolina.

Pripremom pitanja se određuje pitanje $P = (JT, v)$ koje će se postaviti učeniku. Ponekad se prije generiranja pitanja generira izjava kojom se ponavlja što je učenik već naučio.

4.7.3.2 Generiranje izjave

Izjava je skup rečenica kontroliranog jezika i generiranje izjave se u suštini ne razlikuje od generiranja rečenice opisanog u pododjeljku 4.7.2.2. Jedina razlika je u tome što ulazni skup podataka nije okolina nekog koncepta već prethodno određeni skup trojki grafa područnog znanja. Ovaj skup trojki se određuje presjekom restrikcija na okolini koncepta i skupa posječenih trojki grafa područnog znanja *V* koji predstavlja ulaz u proces generiranja rečenice. U stanju t_2 se trojke izjave dobivaju slično kao i trojke pitanja *JT* iz (4.101) tako što se umjesto razlike sa skupom *V* uzima presjek skupova. Presjek skupova se primjenjuje umjesto razlike i u stanju t_5. Ako je skup trojki izjave prazan, onda se prelazi na generiranje pitanja i ne dolazi do generiranja izjave.

4.7.3.3 Generiranje pitanja

Pitanja u dijalogu su zapravo upitne rečenice i postupak generiranja pitanja je sličan postupku generiranja rečenica. Kao što je u pripremi pitanja opisano, skup *JT* se odabire ili na osnovu koncepta koji je sljedeći na ulaznoj traci stroja za testiranje ili se preuzima s potisnog stoga. Ako se pitanje generira na osnovu koncepta, onda se radi o glavnom pitanju za kojeg se skup *JT* dobiva presjekom okoline koncepta i skupa posječenih trojki iz kojeg se slučajnim odabirom izdvajaju one trojke koje imaju istu predikatnu relaciju. Potisni stog stroja za testiranje sadrţi pitanja podrške kojima se kod planiranja pitanja podrške, opisanog u pododjeljku 4.7.3.8, odredio par (*JT*, *v*). Pitanja podrške mogu biti i pitanja nadopune, a njihovo generiranje je također opisano u pododjeljku 4.7.3.8.
Neka je

$$JT = \{\ c_1, gen^+, c_3\ ,\ c_2, gen^+, c_3\ \} \equiv \{\wedge\ \exists c_1, \lrcorner gen^+, \exists c_3\ , \wedge\ \exists c_2, \lrcorner gen^+, \exists c_3\ \} \tag{4.103}$$

primjer skupa jezično označenih trojki čiji koncepti i relacije imaju sljedeće znakovne oznake

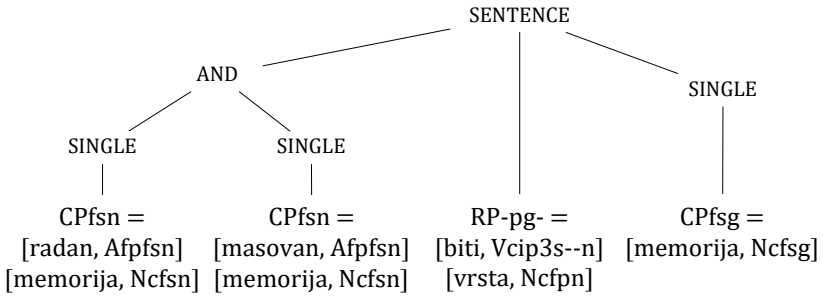
$$\begin{aligned} &zo(c_1) = \#\text{radna\_memorija}, zo(c_2) = \#\text{masovna\_memorija}, \\ &zo(c_3) = \#\text{memorija}, zo(gen^+) = \#\#\text{gen} \end{aligned} \tag{4.104}$$

, a ovim znakovnim oznakama, odnosno identifikatorima su pridruţene parametrizirane fraze iz rječnika kontroliranog jezika kao što je prikazano u tablici 4.58.

Tablica 4.58. Primjer rječnika kontroliranog jezika

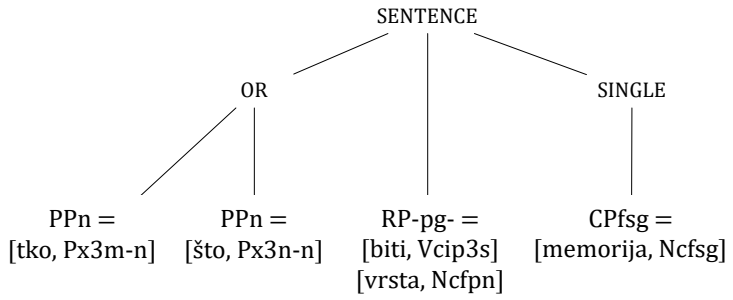
identifikator	parametrizirana fraza
#radna_memorija	CPf***rb***=[radan, Afpf***rb***][memorija, Ncf***rb***]
#masovna_memorija	CPf***rb***=[masovan, Afpf***rb***][memorija, Ncf***rb***]
#memorija	CPf***rb***=[radan, Afpf***rb***][memorija, Ncf***rb***]
##gen	RP-***b***g-=[biti, Vcip3***b***--n][vrsta, Ncf***b***n]

Skup *JT* iz primjera (4.103) ima isti objektni koncept c_3, stoga će on činiti pitanje s fiksnim objektom. Za ovaj primjer jezično označenih trojki se generira rečenica sa slike 4.41.



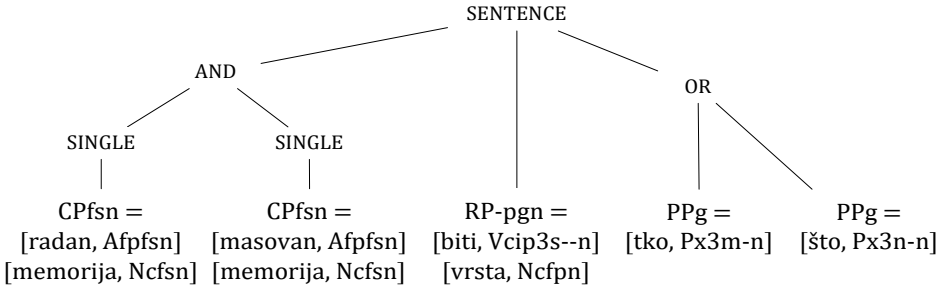
Slika 4.41. Primjer rečenice

Subjektna i objektna pitanja nastaju zamjenom subjektne ili objektne grane rečenice zamjeničnim parametriziranim frazama "tko" i "što" povezanih veznikom OR kao što je prikazano na slikama 4.42 i 4.43.



Tko ili što su vrste memorije?

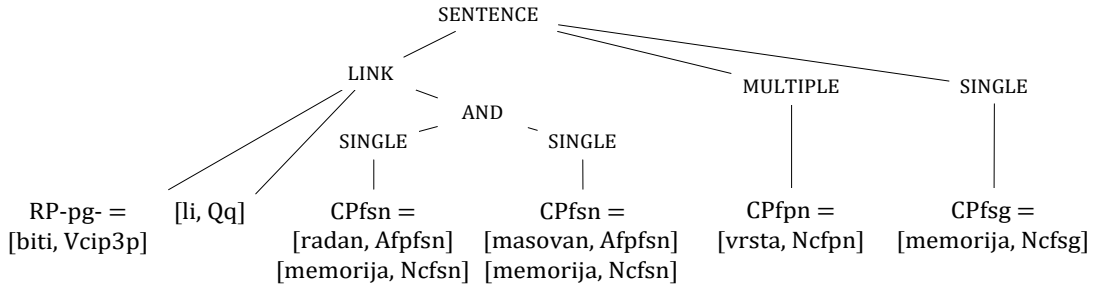
Slika 4.42. Primjer subjektnog pitanja (*JT*, *s*)



Radna memorija i masovna memorija su vrste koga ili čega?

Slika 4.43. Primjer objektnog pitanja (*JT*, *o*)

Kod predikatnih pitanja potrebno je provesti dodatnu obrada SENTENCE elementa kojom se mijenja subjektna i predikatna grana. Ovisno o kategoriji i vremenu relacijske fraze u predikatnoj grani, fraza će se rastaviti na odgovarajuće nizove riječi ili nove fraze koje će se rasporediti u subjektnoj i predikatnoj grani. Subjektna grana se gradi pomoću LINK elementa čije podgrane čine niz riječi i fraze rastavljene relacijske fraze, čestica "li" i subjektna grana prije obrade pitanja, kao što je prikazano na primjeru sa slike 4.44.



Jesu li radna memorija i masovna memorija vrste memorije?

Slika 4.44. Primjer predikatnog pitanja (*JT*, *p*)

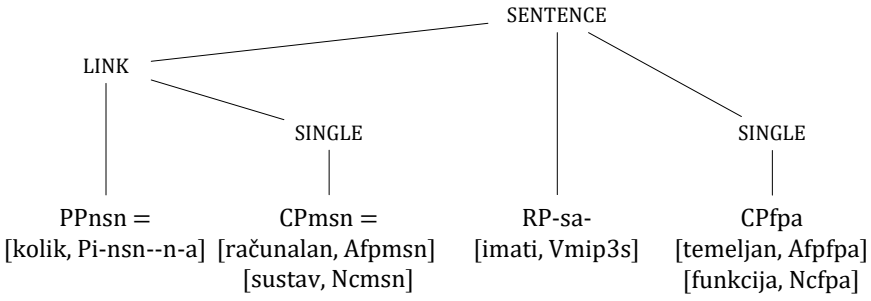
Tablica 4.59 opisuje način konstruiranja predikatnog pitanja ovisno o vremenu i kategoriji relacijske fraze. Prvi i drugi stupac tablice govore o vremenu i kategoriji relacijske fraze. U trećem stupcu se nalaze riječi relacijske fraze koje slijedom čine grane LINK elementa u novoj subjektnoj grani. Posljednja grana LINK elementa nije navedena jer ona uvijek sadrţi subjektnu granu prije početka obrade SENTENCE elementa. Četvrti stupac sadrţi one nizove riječi ili fraze rastavljene relacijske fraze koje će ostati u predikatnoj grani.

Tablica 4.59. Obrada predikatnog pitanja s obzirom na relacijsku frazu

		grane LINK elementa	**predikatna grana**
sadašnje	1	<indikativ prezenta veznog glagola> <čestica "li">	
	2	<indikativ prezenta veznog glagola> <čestica "li">	<prijedlog>
	3	<indikativ prezenta veznog glagola> <čestica "li">	<imenska fraza>
	4	<indikativ prezenta veznog glagola> <čestica "li">	<imenska fraza> <prijedlog>
	5	<indikativ prezenta glavnog glagola> <čestica "li">	
	6	<indikativ prezenta glavnog glagola> <čestica "li">	<prijedlog>
	7	<indikativ prezenta glavnog glagola> <čestica "li">	<imenska fraza>
	8	<indikativ prezenta glavnog glagola> <čestica "li">	<imenska fraza> <prijedlog>
	9	<indikativ prezenta glavnog glagola> <čestica "li"> <čestica "se">	
	10	<indikativ prezenta glavnog glagola> <čestica "li"> <čestica "se">	<prijedlog>
prošlo	1	<indikativ prezenta veznog glagola> <čestica "li">	<particip prošli veznog glagola "biti">
	2	<indikativ prezenta veznog glagola> <čestica "li">	<particip prošli veznog glagola "biti"> <prijedlog>
	3	<indikativ prezenta veznog glagola> <čestica "li">	<particip prošli veznog glagola "biti"> <imenska fraza>
	4	<indikativ prezenta veznog glagola> <čestica "li">	<particip prošli veznog glagola "biti"> <imenska fraza> <prijedlog>
	5	<indikativ prezenta veznog glagola> <čestica "li">	<particip prošli glavnog glagola>
	6	<indikativ prezenta veznog glagola> <čestica "li">	<particip prošli glavnog glagola> <prijedlog>

	7	<indikativ prezenta veznog glagola> <čestica "li">	<particip prošli glavnog glagola> <imenska fraza>
	8	<indikativ prezenta veznog glagola> <čestica "li">	<particip prošli glavnog glagola> <imenska fraza> <prijedlog>
	9	<particip prošli glavnog glagola> <čestica "li"> <čestica "se">	
	10	<particip prošli glavnog glagola> <čestica "li"> <čestica "se">	<prijedlog>
buduće	1	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "li">	<infinitiv veznog glagola "biti">
	2	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "li">	<infinitiv veznog glagola "biti"> <prijedlog>
	3	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "li">	<infinitiv veznog glagola "biti"> <imenska fraza>
	4	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "li">	<infinitiv veznog glagola "biti"> <imenska fraza> <prijedlog>
	5	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "li">	<infinitiv glavnog glagola>
	6	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "li">	<infinitiv glavnog glagola> <prijedlog>
	7	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "li">	<infinitiv glavnog glagola> <imenska fraza>
	8	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "li">	<infinitiv glavnog glagola> <imenska fraza> <prijedlog>
	9	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "li"> <čestica "se">	
	10	<indikativ prezenta pomoćnog glagola "htjeti"> <čestica "li"> <čestica "se">	<prijedlog>

Iznimke su objektna pitanja nastala iz rečenica koje u objektnoj grani imaju brojčane vrijednosti podataka, kao rečenica iz primjera sa slike 4.39. Subjektna i predikatna pitanja se na osnovu ovih rečenica dobivaju isto kao i za sve ostale rečenice, meĐutim kod objektnih pitanja je potrebno napraviti dodatnu obradu rečenice. Slika 4.45 prikazuje objektno pitanje za rečenicu iz primjera sa slike 4.39.



Slika 4.45. Primjer objektnog pitanja za rečenicu s brojčanim vrijednostima podataka

Kod rečenice s brojčanim vrijednostima podataka prvo se sve grane LINK elementa, osim posljednje, prebacuju iz objektne grane u subjektnu granu. U subjektnoj grani se prebačene grane vezuju LINK elementom s prethodnom subjektnom granom, a umjesto brojčane vrijednosti podataka u subjektnom LINK elementu se postavlja zamjenična fraza "kolik".

4.7.3.4 Traţeni tutorski odgovor

Na osnovu skupa jezično označenih trojki grafa područnog znanja *JT*, koji je sluţio za generiranje pitanja, se odreĎuje *traženi tutorski odgovor*. Za traţenje tutorskog odgovora uvode se slijedeće funkcije koje preslikavaju skup jezično označenih trojki grafa područnog $jo(GPZ)$ u skup elemenata područnog znanja E.
Neka je $l[sx, ny, oz] \in jo(GPZ)$, definira se

- funkcija *izdvajanja subjektnog koncepta* $ik_s: jo(GPZ) \rightarrow E$ s $ik_s(l[sx, ny, oz]) = x$
- funkcija *izdvajanja objektnog koncepta* $ik_o: jo(GPZ) \rightarrow E$ s $ik_o(l[sx, ny, oz]) = z$
- funkcija *izdvajanja koncepata* $ik: jo(GPZ) \rightarrow 2^E$ s $ik(l[sx, ny, oz]) = \{x, z\}$

Traţeni tutorski odgovor i prepoznati učenikov odgovor, opisan u pododjeljku 4.7.3.5, su skupovi fraza odgovora. *Skup fraza odgovora* se definira kao skup ureĎenih trojki (if, pf, of) koje se zovu *fraze odgovora* i gdje je

- if - identifikator fraze,
- pf - parametrizirana fraza i
- of - oblik fraze.

Ako fraza odgovora nema identifikator onda pišemo $if = \varepsilon$, a ako nema parametriziranu frazu onda pišemo $pf = \varepsilon$. Na osnovu navedenog se *prepoznata fraza odgovora* definira kao fraza odgovora kod koje je $if \neq \varepsilon$. *Neprepoznata fraza odgovora* će imati $if = \varepsilon$.
Za dvije fraze odgovora (if_1, pf_1, of_1) i (if_1, pf_1, of_1) kaţemo da su jednake ako vrijedi $if_1 \neq \varepsilon$, $if_2 \neq \varepsilon$ i $if_1 = if_2$, odnosno ako su obje fraze odgovora prepoznate i ako imaju jednake identifikatore.
UreĎena trojka prepoznate fraze odgovora ima identifikator i oblik fraze, a ne mora imati parametriziranu frazu, odnosno moţe biti oblika (if, pf, of) ili (if, ε, of). Prepoznata fraza neće imati parametriziranu frazu u slučaju kada se oblik fraze ne moţe parametrizirati, odnosno ako neki od oblika riječi oblika fraze se ne nalazi u HML-u. Na primjer, oblik riječi iz oblika fraze "RAM" nije u HML-u, meĎutim koncept s ovom frazom je identificiran znakovnom oznakom #RAM. UreĎena trojka neprepoznate fraze odgovora ne sadrţi identifikator, moţe sadrţavati parametriziranu frazu i sadrţi oblik fraze, odnosno moţe biti oblika $(\varepsilon, \varepsilon, of)$ ili (ε, pf, of).
Traţeni tutorski odgovor se uvijek sastoji od prepoznatih fraza odgovora. Postupak odreĎivanja tutorskog odgovora ovisi o vrsti pitanja. Kod subjektnih i objektnih pitanja se uzimaju koncepti na subjektnom mjestu jezično označenih trojki pitanja, odnosno na objektnom mjestu pri čemu se na skupu *JT* pitanja primjenjuje funkcija izdvajanja subjektnog koncepta ili izdvajanja objektnog koncepta. Za primjer jezično označenih trojki (4.103) koji ima subjektne koncepte $\{c_1, c_2\}$ i objektni koncept $\{c_3\}$ se za subjektno pitanje iz primjera sa slike 4.42 dobiva traţeni tutorski odgovor

{(#radna_memorija, CPf***rb***=[radan, Afpf***rb***][memorija, Ncf***rb***], radna memorija),
(#masovna_memorija, CPf***rb***=[masovan, Afpf***rb***][memorija, Ncf***rb***], masovna memorija)} (4.105)

, dok objektno pitanje iz primjera sa slike 4.43 će imati traţeni tutorski odgovor

{(#memorija, CPf***rb***=[memorija, Ncf***rb***], memorija)} (4.106)

Ako je koncept vrijednost podataka, onda će prepoznata fraza odgovora imati identifikator fraze i oblik fraze jednak znakovnoj oznaci koncepta i neće imati parametriziranu frazu.

Za odreĎivanje tutorskog odgovora predikatnog pitanja prvo se provjerava pripadnost jezično označenih trojki pitanja područnom znanju. Jezično označene trojke predikatnog pitanja su uvijek oblika $l_i[s_i x_i, ny, o_i z_i]$, odnosno imaju isti predikat i oznaku negacije predikata. Mehanizam zaključivanja sluţi za ustanovljavanje pripadnosti jezično označenih trojki $l_i[s_i x_i, ny, o_i z_i]$ područnom znanju kao što je opisano u odjeljku 4.7.4. Ako sve trojke pripadaju područnom znanju onda se tutorski odgovor sastoji od fraza odgovora čiji je identifikator ##da i fraze odgovora nastale od predikata *y* i oznake negacije *n*. U obrnutom slučaju će prva fraza odgovora nastati od identifikatora ##ne, a druga od predikata *y* i suprotne oznake negacije od *n*. Jezično označene trojke iz primjera (4.103) tvore predikatno pitanje sa slike 4.44. Pošto ove jezično označene trojke pripadaju područnom znanju, onda je tutorski odgovor na predikatno pitanje

{(##da, *ε*, *ε*), (##gen, RP-***b***g-=[biti, Vcip3***b***--n][vrsta, Ncf***b***n], je vrsta)} (4.107)

Ako barem jedna trojka ne pripada područnom znanju, onda bi odgovor na predikatno pitanje bio

{(##ne, *ε*, *ε*), (##gen, RP-***b***gy=[biti, Vcip3***b***--y][vrsta, Ncf***b***n], nije vrsta)} (4.108)

4.7.3.5 Prepoznavanje odgovora

Prepoznavanjem odgovora se povezuje niz znakova učenikova odgovora s identifikatorima parametriziranih fraza iz rječnika kontroliranog jezika. Moţe se dogoditi da se za niz znakova ne moţe pronaći identifikator parametrizirane fraze, meĎutim on se i dalje uzima u razmatranje. Prepoznavanje odgovora je proces koji za ulaz ima niz znakova učenikov odgovora, a za izlaz se dobiva skup fraza odgovora koji se zove *prepoznati učenikov odgovor* Prepoznavanje odgovora se izvodi u pet koraka:

1. razgraničavanje oblika fraze – rastavlja učenikovog odgovor na oblike fraze,
2. rastavljanje oblika fraze – odreĎuje oblike riječi iz oblika fraze,
3. prepoznavanje oblika riječi – pronalazi morfološke trojke svakog oblika riječi,
4. parametrizacija fraze – odreĎuje pripadnost niza morfoloških trojki kontroliranom jeziku,
5. pronalaţenje parametrizirane fraze – odreĎivanje identifikatora parametrizirane fraze na osnovu rječnika kontroliranog jezika.

U trećem koraku se moţe dogoditi da se za pojedini oblik riječi ne moţe pronaći morfološka trojka koja mu odgovara. Razlog neprepoznavanja oblika riječi moţe biti pravopisna pogreška u obliku riječi ili se oblik riječi ne nalazi u rječniku kontroliranog jezika. Za oblik riječi s pravopisnom pogreškom se nastoji pronaći najbliţi oblik riječi iz rječnika kontroliranog jezika, a ako se ne pronaĎe onda se koraci 4 i 5 ne provode. Postoji mogućnost da se četvrti korak uspješno provede, a dobivena parametrizirana fraza se ne moţe pronaći u rječniku kontroliranog jezika. Tada samo oblik fraze parametrizirane fraze predstavlja izlazni element procesa prepoznavanja odgovora, odnosno on je neprepoznata fraza.
Učenikov odgovor na pitanje je niz znakova koji predstavlja oblike fraza povezanih graničnicima. Na primjer,

"unos podataka, obrada podatka i prikazivanje pdoataka" (4.109)

su oblici fraza povezanih nizovima znakovima ",", " i " koji se zovu graničnici. Skup graničnika koji se uzimaju u razmatranje je {",", ", ", " i ", " ili "}.
Prvi korak prepoznavanja odgovora je razgraničavanje oblika fraza na temelju skupa graničnika. Razgraničavanjem niza (4.109) se dobiva skup oblika frazi

{"unos podataka", "obrada podatka", "prikazivanje pdoataka"} (4.110)

U drugom koraku se pojedini oblik fraze rastavlja na oblike riječi kao što je opisano u pododjeljku 4.6.4.2.
Treći korak je sličan prepoznavanju oblika riječi iz poglavlja 4.6.4.3, s razlikom što se ne koriste morfološke trojke iz HML-a, već morfološke trojke iz rječnika kontroliranog jezika. Time se ograničava skup riječi koje se mogu koristiti tijekom prepoznavanja odgovora.
Nerijetko učenikov odgovor moţe sadrţavati pravopisne pogreške, kao što je u primjeru (4.109) učenik pogrešno unio niz znakova "pdoataka" U tom slučaju se nastoji pronaći oblik riječi iz rječnika kontroliranog jezika koji je najbliţi pogrešno unesenoj riječi. Pri tome se koristi Levensthein-ova metrika. Ova metrika odreĎuje udaljenost dvaju nizova znakova na osnovi ukupnog broja akcija dodavanja, zamjena i brisanja znakova potrebnih kako bi se jedan niz znakova pretvorio u drugi niz znakova. Udaljenost izmeĎu "pdoataka" i "podataka" je 2 jer je potrebno izvršiti dvije zamjene znakova. Slika 4.46 prikazuje matricu udaljenosti nizova znakova "pdoataka" i "podataka" koja je nastala izvoĎenjem Levensthein-ovog algoritma. Udaljenost je broj koji se nalazi u posljednjem redu i posljednjem stupcu Levensthein-ove matrice.

		p	**o**	**d**	**a**	**t**	**a**	**k**	**a**
	0	1	2	3	4	5	6	7	8
p	1	0	1	2	3	4	5	6	7
d	2	1	1	1	2	3	4	5	6
o	3	2	1	2	2	3	4	5	6
a	4	3	2	2	2	3	3	4	5
t	5	4	3	3	3	2	3	4	5
a	6	5	4	4	3	3	2	3	4
k	7	6	5	5	4	4	3	2	3
a	8	7	6	6	5	5	4	3	**2**

Slika 4.46. Primjer Levensthein-ove matrice udaljenosti

Ako su se u trećem koraku, za svaki oblik riječi iz oblika fraze, dobile odgovarajuće morfološke trojke, onda se iz niza morfoloških trojki uzimaju leme i za svaku lemu se na osnovu pretraţivanja rječnika kontroliranog jezika odreĎuju morfološke trojke svih oblika riječi. U suprotnom, oblik fraze postaje neprepoznata fraza i proces prepoznavanja odgovora se za taj oblik fraze završava.
U četvrtom koraku se vrši parametrizacija fraze koja je opisana u pododjeljcima 4.6.4.4 i 4.6.4.5, odnosno za svaku kombinaciju iz niza skupova morfoloških trojki se odreĎuje čini li niz frazu kontroliranog jezika. Čim se pronaĎu morfološke trojke koje pripadaju kontroliranom jeziku, prelazi se na peti korak prepoznavanja odgovora. U suprotnom, oblik fraze postaje element prepoznatog učenikovog odgovora. Na osnovu morfoloških trojki čiji su osnovni oblici "prikazati" i "podatak" se prepoznala parametrizirana fraza

CPn***bp*** = [prikazivanje, Ncn***bp***] [podatak, Ncmpg] (4.111)

koja se u petom koraku usporeĎuje sa svim identificiranim parametriziranim frazama iz rječnika kontroliranog jezika.

Dvije parametrizirane fraze A i B su jednake ako imaju jednak broj parametriziranih morfoloških trojki i ako svaka parametrizirana morfološka trojka fraze A na poziciji *j* ima jednaku lemu kao i morfološka trojka fraze B na poziciji *j*.
Ako se ustanovi kako je neka identificirana parametrizirana fraza jednaka danoj parametriziranoj frazi, onda je proces prepoznavanja odgovora, za taj oblik fraze, završen i pronaĎena identificirana parametrizirana fraza postaje prepoznata fraza prepoznatog učenikovog odgovora. Za primjer (4.111) identifikator prepoznate fraze je #prikazivanje_podataka. U slučaju ne postojanja prepoznate fraze, sam oblik fraze postaje element prepoznatog učenikovog odgovora. Za primjer učenikovog odgovora (4.109) prepoznat učenikov odgovor je prikazan u tablici 4.60.

Tablica 4.60. Primjer prepoznatog učenikovog odgovora

identifikator (if)	parametrizirana fraza (pf)	oblik fraze (of)
#unos_podataka	CPmsn = [unos, Ncmsn] [podatak, Ncmpg]	unos podataka
#obrada_podataka	CPfsn = [obrada, Ncfsn] [podatak, Ncmpg]	obrada podatka
#prikazivanje_podataka	CPnsn = [prikazivanje, Ncnsn] [podatak, Ncmpg]	prikazivanje pdoataka

Tablica 4.60 prikazuje primjer prepoznatog učenikovog odgovor koji se sastoji od ureĎenih trojki gdje je prvi član identifikator odgovora, drugi član parametrizirana fraza, a treći član oblik fraze koji se odredio u prvom koraku. Postoji mogućnost neprepoznavanja odreĎenih oblika fraze učenikova odgovora. U tom slučaju, neprepoznata fraza nema identifikator, ali ona i dalje sudjeluje u usporedbi učenikovog odgovora i tutorskog odgovora.

4.7.3.6 Analiza i stanje odgovora

Na osnovu pitanja, traţenog tutorskog odgovora (T) i prepoznatog učenikov odgovora (U) se odreĎuje stanje odgovora. Stanje odgovora je skup ureĎenih parova koji se sastoji od stanja elementa i fraze odgovora iz prepoznatog učenikovog odgovora ili iz traţenog tutorskog odgovora. *Stanje fraze odgovora* moţe biti točno (t), djelomično točno (dt), pogrešno (p), nepoznato (np) i nedostajuće (nd).
Proces analize stanja odgovora ovisi o vrsti pitanja. Kod subjektnih i objektnih pitanja se vrši particija prepoznatog učenikovog odgovora U, s obzirom na stanja fraza odgovora, na skupove:

- U_t – skup točnih fraza odgovora,
- U_{dt} – skup djelomično točnih fraza odgovora,
- U_p – skup pogrešnih fraza odgovora i
- U_{np} – skup nepoznatih fraza odgovora.

Traţeni tutorski odgovor T subjektnog ili objektnog pitanja se dijeli samo na skup nedostajućih fraza odgovora T_{nd}.
Na primjeru subjektnog pitanja sa slike 4.42 nastalog na osnovu jezično označenih trojki (4.103) s traţenim tutorski odgovorom (4.105) će se pokazati proces analize stanja odgovora, uz pretpostavku da je tablicom 4.61 prikazan prepoznat učenikov odgovor U.

Tablica 4.61. Primjer prepoznatog učenikovog odgovora

identifikator (if)	parametrizirana fraza (pf)	oblik fraze (of)
#radna_memorija	CPf***rb*** = [radan, Afpf***rb***] [memorija, Ncf***rb***]	radna memorija
#disketa	CPf***rb*** = [disketa, Ncf***rb***]	disketa
#RAM	ε	RAM
#monitor	CPm***rb*** = [monitor, Ncm***rb***]	monitor
ε	ε	dioda

Skupovi U_t, U_{np} i T_{nd} se određuju po formulama

$$\begin{aligned} U_t &= T \cap U \\ U_{np} &= \{\text{if}, \text{pf}, \text{of} \in U \mid \text{if} = \varepsilon\} \\ T_{nd} &= T \setminus U \end{aligned} \tag{4.112}$$

Skup točnih fraza odgovora sadrţi zajedničke fraze odgovora skupa T i skupa U, odnosno one fraze odgovora koje imaju jednake identifikatore. Neprepoznate fraze skupa U, odnosno fraze odgovora bez identifikatora, čine skup nepoznatih fraza U_{np}. Skup nedostajućih fraza odgovora T_{nd} sadrţi one prepoznate fraze tutorskog odgovora T koje se ne nalaze u prepoznatom učenikovom odgovoru U.

Za primjer prepoznatog učenikovog odgovora iz tablice 4.61 skup U_t sadrţi frazu odgovora s identifikatorom #radna_memorija, skup U_{np} sadrţi frazu odgovora čiji je oblik fraze "dioda", a skup T_{nd} sadrţi frazu odgovora čiji je identifikator #masovna_memorija. Preostale fraze odgovora skupa U će biti djelomično točne ili pogrešne.

Neka je $P = U \setminus (U_t \cup U_{np})$ skup preostalih fraza odgovora skupa U. Sve fraze odgovora skupa P imaju identifikatore fraza. Ovi identifikatori su znakovne oznake koncepata područnog znanja, a koncepti definiraju *skup preostalih koncepata učenikovog odgovora* Ko. Jezično označene trojke koje su sluţile za generiranje pitanja definiraju *skup koncepata pitanja* Kp, koji je kod subjektnog pitanja jednak skupu svih objektnih koncepata trojki, a kod objektnog pitanja je jednak skupu svih subjektnih koncepata trojki. U subjektnom ili objektnom pitanju se koristi relacija *r* koja ima jezičnu oznaku negacije *n*. Na osnovu relacije *r*, oznake *n*, skupa P, Ko i Kp se algoritmom 4.3 određuju skupovi U_{dt} i U_p.

Algoritam 4.3. Određivanje skupa djelomično točnih i pogrešnih fraza odgovora subjektnog pitanja

```
1   for each x ∈ Ko
2         found = false
3         fo = (zo(x), pf, of) ∈ P
4         for each z ∈ Kp
5               if pripadnost(x n r z) then
6                     U_dt = U_dt ∪ {fo}
7                     found = true
8               end if
9         end for
10        if not found then
11              U_p = U_p ∪ {fo}
12        end if
13  end for
```

Za svaki koncept $x \in$ Ko i za svaki koncept $z \in$ Kp algoritam 4.3 provjerava je li jezično označena trojka [*x*, *nr*, *z*] pripada područnom znanju. Pripadnost jezično označene trojke se provjerava uz pomoć mehanizma zaključivanja temeljenog na pravilima postavljanjem upita pripadnost(*x*, *n*, *r*, *z*) što je opisano u pododjeljku 4.7.4.1. Ako jezično označena trojka

pripada, onda se skup djelomično točnih fraza odgovora U_{dt} proširuje frazom odgovora fo koncepta *x* koja se odredila u liniji 3. Fraza odgovora koncepta *x* je ona fraza odgovora iz skupa preostalih fraza odgovora čija znakovna oznaka *zo*(*x*) je jednaka identifikatoru neke fraze preostalih odgovora. Ako se za sve koncepte pitanja $z \in$ Kp ne pronaĎe niti jedna jezično označena trojka koja pripada područnom znanju, onda fraza odgovora fo koncepta *x* postaje element skupa pogrešnih fraza odgovora U_p.
Algoritam 4.3 se koristi i kod objektnih pitanja s razlikom što se u liniji 5 provjerava pripadnost jezično označene trojke [*z*, *nr*, *x*].
Neka su identifikatori fraza #disketa, #RAM i #monitor iz tablice 4.61 znakovne oznake koncepata c_4, c_5 i c_6, odnosno Ko=$\{c_4, c_5, c_6\}$. Skup koncepata pitanja Kp primjera subjektnog pitanja sa slike 4.42 sadrţi samo koncept c_3, a predikat i jezična oznaka negacije pitanja je $nr = \lrcorner gen^+$. Uz pomoć mehanizma zaključivanja temeljenog na pravilima se odreĎuje pripadnost jezično označenih trojki $[c_4, \lrcorner gen^+, c_3]$, $[c_5, \lrcorner gen^+, c_3]$ i $[c_6, \lrcorner gen^+, c_3]$ područnom znanju, odnosno mehanizmu zaključivanja se postavljaju upiti pripadnost(c_4, ⊥, gen^+, c_3), pripadnost(c_5, ⊥, gen^+, c_3) i pripadnost(c_6, ⊥, gen^+, c_3). Za ovaj primjer, skup U_{dt} se sastoji od fraza odgovora čiji su indentifikatori #disketa i #RAM, dok skup U_p sadrţi frazu odgovora s identifikatorom #monitor. Primjer stanja odgovora iz tablice 4.62 prikazuje vezu fraze odgovora sa stanjima fraza odgovora.

Tablica 4.62. Primjer stanja odgovora

identifikator (if)	parametrizirana fraza (pf)	oblik fraze (of)	stanje
#radna_memorija	CPf***rb*** = [radan, Afpf***rb***] [memorija, Ncf***rb***]	radna memorija	t
#disketa	CPf***rb*** = [disketa, Ncf***rb***]	disketa	dt
#RAM	ε	RAM	dt
#monitor	CPm***rb*** = [monitor, Ncm***rb***]	monitor	p
ε	ε	dioda	np
#masovna_memorija	CPf***rb*** = [masovan, Afpf***rb***] [memorija, Ncf***rb***]	masovna memorija	nd

Za predikatno pitanje, s obzirom na stanja fraza odgovora, skup U_{dt} je uvijek prazan. Skupovi U_t, U_{np} i T_{nd} se odreĎuju po formulama (4.112), a skup pogrešnih fraza odgovora U_p se odreĎuje razlikom skupova

$$U_P = U \setminus T \tag{4.113}$$

Nakon analize stanja odgovora se odreĎuje vrijednost stanja koncepta u modelu učenika. Za subjektno i objektno pitanje se stanje koncepta dobiva po formuli

$$\frac{|U_t| + \frac{1}{2}|U_{dt}| - |U_p|}{|U| + |T_{nd}|} \tag{4.114}$$

, dok se stanje koncepta predikatnog pitanja odreĎuje po formuli

$$\begin{matrix} 1, |Ut| > 0 \ \wedge \ |Up| = 0 \\ 0, |Up| > 0 \end{matrix} \tag{4.115}$$

4.7.3.7 Generiranje povratne informacije

Povratnom informacijom se učenik obavještava o točnosti svog odgovora. Strukturalno gledano, povratna informacija je niz rečenica kontroliranog jezika. Koje će se rečenice generirati ovisi o pitanju i o stanju odgovora. Rečenice povratne informaciju se generiraju na osnovu predloţaka rečenica. *Predložak rečenice* je SENTENCE element kojemu je subjektna ili objektna grana zamijenjena parametriziranim frazama ili oblicima fraza prepoznatog učenikovog odgovora. Neki predlošci rečenice su *generativni*, odnosno grane SENTENCE elementa se generiraju na osnovu jezično označenih trojki pitanja. Priprema generativnog predloška rečenice se ne razlikuje od generiranja rečenice opisanog u pododjeljku 4.7.2.2. Na primjer, za jezično označene trojke (4.103) se generira rečenica sa slike 4.41. Postoje četiri vrste generativnih predloţaka rečenice koje ovise o vrsti pitanja i o stanju fraza odgovora:

- predloţak rečenice subjektnog pitanja s točnim i djelomično točnim frazama ($PR^{S}_{\mathrm{T,DT}}$),
- predloţak rečenice objektnog pitanja s točnim i djelomično točnim frazama ($PR^{O}_{\mathrm{T,DT}}$),
- predloţak rečenice subjektnog pitanja s pogrešnim frazama (PR^{S}_{P}),
- predloţak rečenice objektnog pitanja s pogrešnim frazama (PR^{O}_{P}).

Sintaksa rečenice je

```
SENTENCE[
  SUBJECT_BRANCH
  PREDICATE_BRANCH
  OBJECT_BRANCH]
```
(4.116)

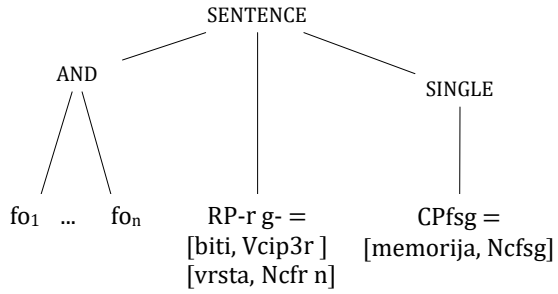
S obzirom na ovu sintaksu vrste generativnih predloţaka rečenice i njihova sintaksa su prikazani u tablici 4.63.

Tablica 4.63. Generativni predlošci rečenice

vrsta predloška rečenice	parametri predloška rečenice	sintaksa predloška rečenice
$PR^{S}_{\mathrm{T,DT}}(\mathrm{fo}_1, \ldots, \mathrm{fo}_n)$	$\mathrm{fo}_1, \ldots, \mathrm{fo}_n \in \mathrm{U_t} \cup \mathrm{U_{dt}}$	SENTENCE[AND[fo_1 … fo_n] PREDICATE_BRANCH OBJECT_BRANCH]
$PR^{O}_{\mathrm{T,DT}}(\mathrm{fo}_1, \ldots, \mathrm{fo}_n)$	$\mathrm{fo}_1, \ldots, \mathrm{fo}_n \in \mathrm{U_t} \cup \mathrm{U_{dt}}$	SENTENCE[SUBJECT_BRANCH PREDICATE_BRANCH AND[fo_1 … fo_n]]
$PR^{S}_{\mathrm{P}}(\mathrm{fo}_1, \ldots, \mathrm{fo}_n)$	$\mathrm{fo}_1, \ldots, \mathrm{fo}_n \in \mathrm{U_p}$	SENTENCE[AND[fo_1 … fo_n] NOT[PREDICATE_BRANCH] OBJECT_BRANCH]
$PR^{O}_{\mathrm{P}}(\mathrm{fo}_1, \ldots, \mathrm{fo}_n)$	$\mathrm{fo}_1, \ldots, \mathrm{fo}_n \in \mathrm{U_p}$	SENTENCE[SUBJECT_BRANCH NOT[PREDICATE_BRANCH] AND[fo_1 … fo_n]]

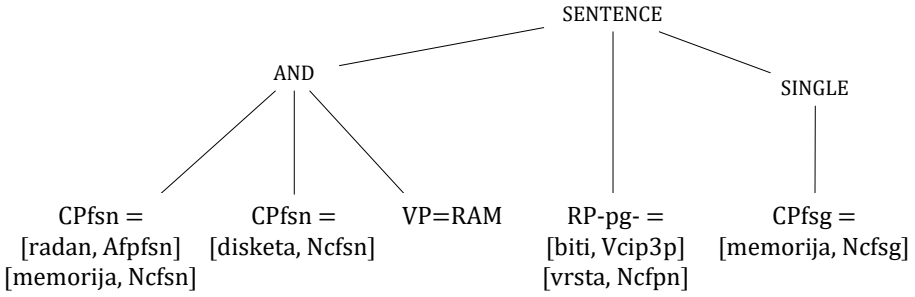
Ulazni parametri predloška rečenice su fraze odgovora fo prepoznatog učenikovog odgovora U. Za predloške rečenica s točnim i djelomično točnim frazama, fraze odgovora su sve fraze skupa točnih fraza odgovora $\mathrm{U_t}$ i skupa djelomično točnih fraza odgovora $\mathrm{U_{dt}}$. Fraze odgovora predloška rečenica s pogrešnim odgovorima su iz skupa pogrešnih fraza odgovora $\mathrm{U_p}$. Ovisno

o tome je li predloţak rečenice namijenjen subjektnom, odnosno objektnom pitanju, onda će subjektna, odnosno objektna grana SENTENCE elementa sadrţavati fraze odgovora povezane elementom AND. Nadalje, kod predloţaka rečenica s pogrešnim frazama odgovora postavlja se element NOT u predikatnoj grani. Slika 4.47 grafički prikazuje sintaksu predloška rečenice $PR^{S}_{\mathrm{T,DT}}$ za rečenicu iz primjera sa slike 4.42.



Slika 4.47. Primjer predloška rečenice subjektnog pitanja s točnim i djelomično točnim frazama odgovora

Neka su točne i djelomično točne fraze odgovora za primjer sa slike 4.47 dane u tablici 4.62. S obzirom na ove fraze odgovora, generirana rečenica za predloţak rečenice je prikazana na slici 4.48.



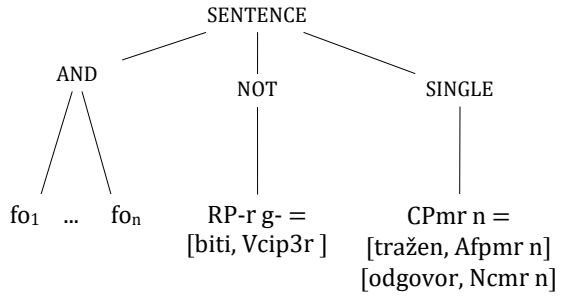
Slika 4.48. Generirana rečenica na osnovu predloška rečenice

Generiranje rečenica na osnovu predloška rečenice se sastoji od postavljanja fraza odgovora u predloţak rečenice. Pri tome se postavlja parametrizirana fraza, ako je fraza odgovora ima, inače se postavlja oblik fraze, kao što je u primjeru sa slike 4.48 postavljen oblik fraze RAM.

Konstantni predlošci rečenice se razlikuju od generiranih po tome što subjektna, predikatna i objektna grana SENTENCE elementa ne ovise o vrsti pitanja i unaprijed su definirane ili sadrţe fraze odgovora. Postoje dvije vrste konstantnih predloţaka rečenice:

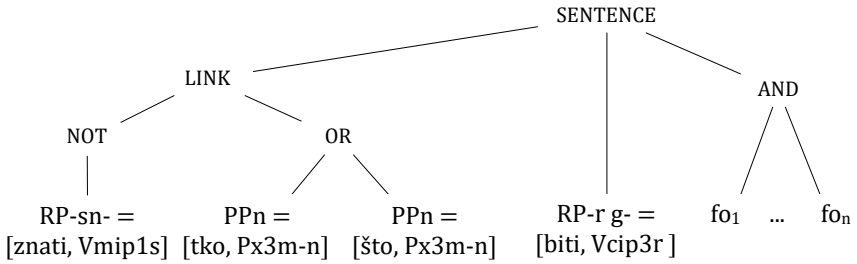
- predloţak rečenice s djelomično točnim frazama odgovora (PR_{dt}).
- predloţak rečenice s nepoznatim frazama odgovora (PR_{np}) i

Slika 4.49 prikazuje sintaksu predloška rečenice s djelomično točnim frazama odgovora. Generirana rečenica, na osnovu ovog predloška, iskazuje da fraze djelomično točnih odgovora nisu traţeni odgovor.



Slika 4.49. Sintaksa predloška rečenice s djelomično točnim frazama odgovora

Nepoznate fraze odgovora zajedno s predloškom rečenice PR_{np}, čija je sintaksa prikazana na slici 4.50, sluţe za generiranje rečenice kojom se iskazuje nepoznavanje fraza odgovora.

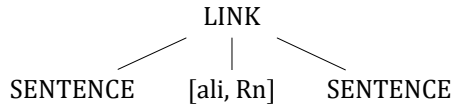


Slika 4.50. Sintaksa predloška rečenice s nepoznatim frazama odgovora

Navedene vrste generiranih i konstantnih predloţaka rečenica se mogu povezivati veznikom "ali", što ovisi o stanju odgovora. Ako se dva predloška pitanja povezuju veznikom "ali", to će se pisati

$$\text{LINK[Predloţak1 [ali, Rn] Predloţak2]} \tag{4.117}$$

, a sintaksa povezanih rečenica je prikazana na slici 4.51 u kojoj lijevi SENTENCE element pripada predlošku 1 dok desni pripada predlošku 2.



Slika 4.51. Sintaksa predloţaka pitanja povezanih veznikom "ali"

Tablica 4.64 prikazuje moguće nizove rečenica za subjektno pitanje nastalih od predloţaka rečenica ovisno o stanju odgovora. Nizovi rečenica za objektno pitanje se razlikuje od nizova rečenica subjektnog pitanja po tome što se umjesto generativnih predloţaka subjektnog pitanja koriste generativni predlošci objektnog pitanja. Konstantni predlošci rečenica se ne zamjenjuju.

Tablica 4.64. Niz rečenica povratne informacije za subjektno pitanje

stanje odgovora				sintaksa niza rečenica
$U_t \neq \emptyset$	$U_{dt} \neq \emptyset$	$U_p \neq \emptyset$	$U_{np} \neq \emptyset$	
da	da	da	da	LINK[$PR^{S}_{t,dt}(U_t \cup U_{dt})$ [ali, Rn] $PR_{dt}(U_{dt})$] $PR^{s}_{p}(U_P)$ $PR_{np}(U_{np})$
da	da	da	ne	LINK[$PR^{S}_{t,dt}(U_t \cup U_{dt})$ [ali, Rn] $PR_{dt}(U_{dt})$] $PR^{s}_{p}(U_P)$
da	da	ne	da	LINK[$PR^{S}_{t,dt}(U_t \cup U_{dt})$ [ali, Rn] $PR_{dt}(U_{dt})$] $PR_{np}(U_{np})$
da	da	ne	ne	LINK[$PR^{S}_{t,dt}(U_t \cup U_{dt})$ [ali, Rn] $PR_{dt}(U_{dt})$]
da	ne	da	da	LINK[$PR^{S}_{t,dt}(U_t)$ [ali, Rn] $PR^{S}_{p}(U_p)$] $PR_{np}(U_{np})$
da	ne	da	ne	LINK[$PR^{S}_{t,dt}(U_t)$ [ali, Rn] $PR^{S}_{p}(U_p)$]
da	ne	ne	da	[točno, Rnc] $PR^{S}_{t,dt}(U_t)$ $PR_{np}(U_{np})$
da	ne	ne	ne	[točno, Rnc] $PR^{S}_{t,dt}(U_t)$
ne	da	da	da	LINK[$PR^{S}_{t,dt}(U_{dt})$ [ali, Rn] $PR_{dt}(U_{dt})$] $PR^{s}_{p}(U_P)$ $PR_{np}(U_{np})$
ne	da	da	ne	LINK[$PR^{S}_{t,dt}(U_{dt})$ [ali, Rn] $PR_{dt}(U_{dt})$] $PR^{s}_{p}(U_P)$
ne	da	ne	da	LINK[$PR^{S}_{t,dt}(U_{dt})$ [ali, Rn] $PR_{dt}(U_{dt})$] $PR_{np}(U_{np})$
ne	da	ne	ne	LINK[$PR^{S}_{t,dt}(U_{dt})$ [ali, Rn] $PR_{dt}(U_{dt})$] [pogrešno, Rnc]
ne	ne	da	da	$PR^{S}_{p}(U_p)$ $PR_{np}(U_{np})$ [pogrešno, Rnc]
ne	ne	da	ne	$PR^{S}_{p}(U_p)$
ne	ne	ne	da	$PR_{np}(U_{np})$

S obzirom na stanje odgovora iz tablice 4.62 na subjektno pitanje sa slike 4.42, povratna informacija će uključivati sljedeći niz rečenica

Radna memorija, disketa i RAM su vrste memorije, ali disketa i RAM nisu traţeni odgovori.
Monitor nije vrsta memorije.
Ne znam što je dioda. (4.118)

Predikatna pitanja nemaju djelomično točne fraze odgovora. Neka RP predstavlja rečenicu predikatnog pitanja. Ako tutorski odgovor na pitanje uključuje oblik fraze "ne", odnosno ako je "ne" točan odgovor, tada se u predikatnoj grani od RP dodaje element NOT. S obzirom na RP, u tablici 4.65 je prikazana sintaksa niza rečenica povratne informacije predikatnog pitanja koja ovisi o stanju odgovora.

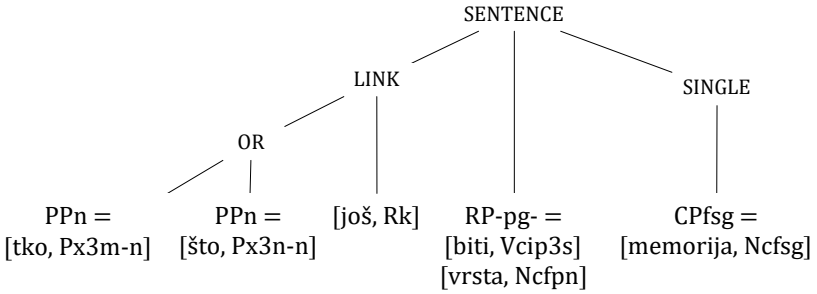
Tablica 4.65. Niz rečenica povratne informacije za predikatno pitanje

stanje odgovora			sintaksa niza rečenica
$U_t \neq \emptyset$	$U_p \neq \emptyset$	$U_{np} \neq \emptyset$	
da	da	da	[pogrešno, Rn] RP $PR_{np}(U_{np})$
da	da	ne	[pogrešno, Rn] RP
da	ne	da	[točno, Rn] RP $PR_{np}(U_{np})$
da	ne	ne	[točno, Rn] RP
ne	da	da	[pogrešno, Rn] RP $PR_{np}(U_{np})$
ne	da	ne	[pogrešno, Rn] RP
ne	ne	da	$PR_{np}(U_{np})$

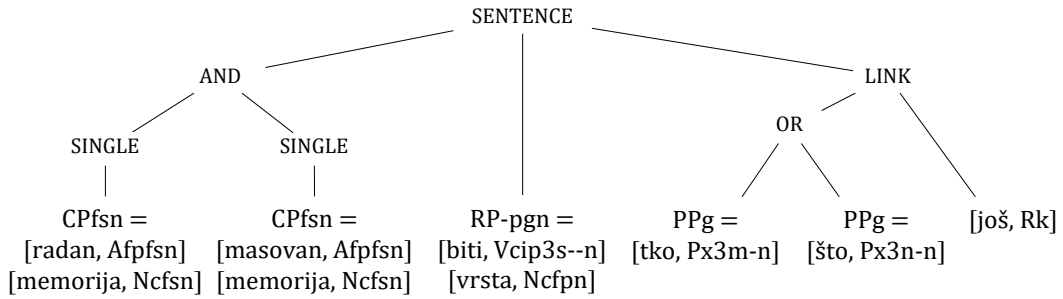
Povratna informacija predikatnog pitanja skoro uvijek započinje prikazom riječi "pogrešno" ili "točno" nakon koje slijedi rečenica pitanja RP. Ako stanje odgovora ima neprepoznatih fraza odgovora, oni će se naći u rečenici nastaloj po predlošku rečenice PR_{np}.

4.7.3.8 Planiranje pitanja podrške

Cilj planiranja pitanja podrške je odreĎivanje niza pitanja podrške koja će se postaviti učeniku, ako nije u potpunosti točno odgovorio na postavljeno pitanje. Pitanje podrške moţe biti subjektno pitanje, objektno pitanje ili predikatno pitanje. Subjektno i objektno pitanje podrške mogu imati prilog "još" i takva pitanja podrške se zovu pitanja nadopune. Pitanja nadopune se tvore tako što se zamjenične fraze "tko" i "što" poveţu elementom LINK s prilogom "još" kao što je prikazano na primjerima subjektnog i objektnog pitanja sa slika 4.52 i 4.53.



Slika 4.52. Primjer subjektnog pitanja nadopune



Slika 4.53. Primjer objektnog pitanja nadopune

Potisni stog stroja za testiranje pamti pitanja podrške koja će se generirati. Planiranjem pitanja podrške se na osnovu postavljenog pitanja, prepoznatog učenikovog odgovora i tutorskog odgovora odreĎuju pitanja podrške koja će se postaviti na potisni stog. U planiranju sudjeluje mehanizam zaključivanja temeljen na mrežnom prikazu znanja, opisan u pododjeljku 4.7.4.2, kojim se pronalazi putanja meĎu konceptima čiji identifikatori pripadaju frazama odgovora. Elementi dobivene putanje su trojke grafa područnog znanja, a svaka trojka služi za oblikovanje pitanje podrške.

Planiranje pomoćnih pitanja se neće vršiti ako je postavljeno pitanje predikatno ili ako je broj točnih učenikovih odgovora jednak broju traženih tutorskih odgovora $|U_t|=|T|$, odnosno ako nema nedostajućih tutorskih odgovora $|T_{nd}|=0$.

Ako ima nedostajućih tutorskih odgovora, tada se, obzirom na broj točnih učenikovih odgovora $|U_t|$, djelomično točnih učenikovih odgovora $|U_{dt}|$ i pogrešnih učenikovih odgovora $|U_p|$, primjenjuje jedan ili više algoritama za odreĎivanje pomoćnih pitanja. Algoritmi za odreĎivanje pomoćnih pitanja temelje se na

- pronalaženju putanja grafa područnog znanja izmeĎu koncepata odgovora i koncepata pitanja,
- pronalaženju putanja grafa područnog znanja unutar okoline koncepata pitanja i
- generiranju pitanja nadopune.

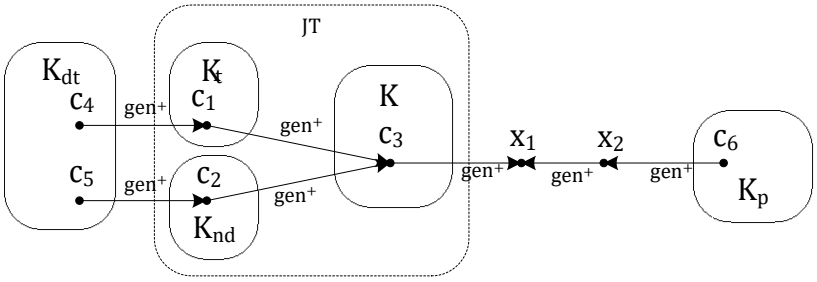
Ulazni argumenti algoritama za odreĎivanje pitanja podrške su koncepti područnog znanja koje je potrebno izdvojiti iz glavnog pitanja, prepoznatog učenikovog odgovora i traženog tutorskog odgovora. Na osnovu jezično označenih trojki *JT* i vrste pitanja se iz glavnog pitanja odreĎuju koncepti koji će imati ulogu u odreĎivanju pitanja podrške. Uvodi se *skup koncepata pitanja* K koji se definira preko funkcije izdvajanja koncepata opisane u pododjeljku 4.7.3.4. Ako je pitanje temeljeno na jezično označenim trojkama *JT*, tada se za subjektno pitanje skup K definira preko slike izdvajanja funkcije objektnih koncepata $ik_o(JT)$, dok je za objektno pitanje $K=ik_s(JT)$. Za primjer subjektnog pitanja sa slike 4.42 čiji je skup *JT* naveden u (4.103), skup koncepata pitanja je $K=\{c_3\}$.

Kao što je u pododjeljku 4.7.3.5 definirano, prepoznati učenikov odgovor je skup fraza odgovora gdje je svaka fraza ureĎena trojka čiji su elementi identifikator fraze, parametrizirana fraza i oblik fraze. Identifikator fraze je jedinstvena znakovna oznaka koncepta područnog znanja *PZ* i na osnovu inverza pridruživanja znakovne oznake zo^{-1} se odreĎuje koncept područnog znanja kojemu je pridružen identifikator fraze. S obzirom na statuse koje mogu imati fraze učenikovog odgovora, uvode se

- skup točnih koncepata K_t,
- skup djelomično točnih koncepata K_{dt} i
- skup pogrešnih koncepata K_p.

Dodatno se za tutorski odgovor uvodi skup nedostajućih koncepata K_{nd}. Na osnovu stanja odgovora primjera iz tablice 4.62 skupovi koncepata su $K_t=\{c_1\}$ i $K_{nd}=\{c_2\}$. Ako je, uz

pretpostavku, $zo(c_4)$=#disketa, $zo(c_5)$=#RAM i $zo(c_6)$=#monitor, gdje je *zo* znakovno označavanje u područnom znanju, tada je $K_{dt}=\{c_4, c_5\}$ i $K_p=\{c_6\}$. Na slici 4.54 su prikazani skupovi koncepata i trojke grafa područnog znanja koje sadrţe navedene koncepte iz primjera pitanja i primjera stanja odgovora.



Slika 4.54. Primjer skupova koncepata i trojki grafa područnog znanja

Ovisno o broju elemenata skupova koncepata, koji je jednak broju fraza odgovora, odabrat će se jedan od algoritama za odreĎivanje pitanja podrške. Tablica 4.66 prikazuje uvjete i algoritme koji će se pokrenuti ovisno o uvjetima za subjektno pitanje. Kod objektnog pitanja i kod odreĎivanja usmjerene putanje, ulazni argumenti zamijene mjesta, odnosno umjesto usmjerena_putanja(Kdt, Kt ∪ Knd, K) se izvršava usmjerena_putanja(K, Kt ∪ Knd, Kdt).

Tablica 4.66. Uvjeti, skupovi putanja i algoritmi za određivanje pitanja podrške kod subjektnog pitanja

K_t	K_{dt}	K_p	K_{nd}	**skup putanja i algoritam**
=0	=0	=0	>0	SP=okolina(K K_{nd})
=0	=0	>0	>0	SP=neusmjerena_putanja(K_p, K)
=0	>0	=0	>0	SP=usmjerena_putanja(K_{dt}, K_t K_{nd}, K)
=0	>0	>0	>0	SP=usmjerena_putanja(K_{dt}, K_t K_{nd}, K) neusmjerena_putanja(K_p, K)
>0	=0	=0	>0	nadopuna
>0	=0	>0	>0	SP=neusmjerena_putanja(K_p, K)
>0	>0	=0	>0	SP=usmjerena_putanja(K_{dt}, K_t K_{nd}, K)
>0	>0	>0	>0	SP=usmjerena_putanja(K_{dt}, K_t K_{nd}, K) neusmjerena_putanja(K_p, K)

Algoritam odreĎivanja putanje moţe odrediti usmjerene i neusmjerene putanje izmeĎu dva skupa koncepata. OdreĎivanje pojedinačne putanje izmeĎu dva koncepta je opisano u pododjeljku 4.7.4.2. Za dva skupa koncepata K_1 i K_2 odreĎuje se skup putanja SP tako što se za svaki koncept iz K_1 i svaki koncept iz K_2 postavlja upit mehanizmu zaključivanja temeljenom na grafu područnog znanja koji rezultira putanjom. Jedino kod odreĎivanja usmjerene putanje se za tri skupa koncepata K_1, K_2 i K_3 prvo odredi skup usmjerenih putanja SP_1 izmeĎu skupa K_1 i K_2, zatim skup usmjerenih putanja SP_2 izmeĎu skupova K_2 i K_3, te se na kraju putanje iz dobivenih skupova SP_1 i SP_2 nadoveţu.

Nadovezivanje dvije putanje p i p' je moguće ako i samo ako niz trojki putanje p završava trojkom čiji je objektni koncept jednak subjektnom konceptu trojke kojom započinje putanja p'. Za takve putanje kaţemo da su *nadovezive*. Na primjer, neka su $p=(x_1, y_1, z_1)\ldots(x_n, y_n, z_n)$ i $p'=(x_1', y_1', z_1')\ldots(x_m', y_m', z_m')$ nadovezive putanje, odnosno $z_n=x_1'$. Tada se definira operator nadovezivanja "+" na sljedeći način

$$p + p' = (x_1, y_1, z_1)\ldots(x_n, y_n, z_n)\ (x_1', y_1', z_1')\ldots(x_m', y_m', z_m') \qquad (4.119)$$

Algoritam za odreĎivanje putanje moţe traţiti skup usmjerenih ili neusmjerenih putanja. Na osnovu primjera sa slike 4.54, vrijedi

$$\begin{aligned}&\text{usmjerena\_putanja}(K_{dt}, K_t \quad K_{nd}, K) = \{(c_4, gen^+, c_1)(c_1, gen^+, c_3),\\ &(c_5, gen^+, c_2)(c_2, gen^+, c_3)\}\\ &\text{neusmjerena\_putanja}(K_p, K) = \{(c_6, gen^+, x_2)(x_2, gen^+, x_1)(c_3, gen^+, x_1)\}\end{aligned} \quad (4.120)$$

Dobiveni skup putanja se dodatno optimizira jer neke putanje mogu sadrţavati druge putanje ili se preklapati. Za putanju p=(x_1, y_1, z_1)…(x_n, y_n, z_n) kaţemo da *sadrži* putanju p'=(x_1', y_1', z_1')…(x_m', y_m', z_m') i pišemo $p \geq p'$ ako je $n \geq m$ i ako postoji $i \leq n-m$ takav da je $(x_i, y_i, z_i) = (x_1', y_1', z_1')$, $(x_{i+1}, y_{i+1}, z_{i+1}) = (x_2', y_2', z_2')$, …, $(x_{i+m-1}, y_{i+m-1}, z_{i+m-1}) = (x_m', y_m', z_m')$. Dvije putanje p i p' se *preklapaju* i pišemo $p \succcurlyeq p'$ ako postoji podniz $p'' = (x_i', y_i', z_i') \ldots (x_j', y_j', z_j')$ od p' tako da vrijedi $p \geq p''$ i $(x_j', y_j', z_j') = (x_n, y_n, z_n)$. Za nizove koji se meĎusobno sadrţavaju ili preklapaju definira se operacija zbrajanja $\oplus$ za koju vrijedi

$$\begin{aligned}&p \oplus p' = p \text{ ako je } p \geq p'\\ &p \oplus p' = (x_1, y_{1i}, z_1) \ldots (x_n, y_n, z_n)(x_{j+1}', y_{j+1}', z_{j+1}') \ldots (x_m', y_m', z_m') \text{ ako je } p \succcurlyeq p' \text{ i}\\ &(x_j', y_j', z_j') = (x_n, y_n, z_n)\end{aligned} \quad (4.121)$$

Neka je $p_1 = (k_1, r_1, k_2)(k_2, r_1, k_3)(k_3, r_2, k_4)$ i $p_2 = (k_2, r_1, k_3)(k_3, r_2, k_4)$. Očito vrijedi $p_1 \geq p_2$, pa je $p_1 \oplus p_2 = p_1$. Ako je $p_3 = (k_2, r_1', k_3')(k_3, r_2', k_4')(k_4, r_3', k_5')$, odnosno $p_1 \succcurlyeq p_3$, tada je $p_1 \oplus p_2 = (k_1, r_1', k_2')(k_2, r_1', k_3')(k_3, r_2', k_4')(k_4, r_3', k_5')$.

Izlazni skup putanja iz algoritma za odreĎivanje putanje se dodatno optimizira tako što se zbrajaju one putanje koje se meĎusobno sadrţavaju ili preklapaju. Neka skup putanja SP sadrţi putanje $\{p_1, p_2, p_3\}$, nakon optimizacije skupa putanja dobiva se skup $\{p_1 \oplus p_3\}$.

Optimizirani skup putanja sluţi za definiranje pitanja podrške koja se postavljaju na potisni stog stroja za testiranje. Pitanje podrške je definirano kao ureĎeni par (*t*, *vp*) gdje je

- *t*=(*x*, *y*, *z*) – trojka grafa područnog znanja i
- *vp* – vrsta pitanja, s – subjektno, p – predikatno, o – objektno pitanje.

Nakon optimizacije se skup putanja preslikava u skup nizova pitanja podrške. Način preslikavanja ovisi o vrsti postavljenog pitanja i o tome je li putanja usmjerena ili neusmjerena. U tablici 4.67 je prikazano preslikavanje putanje (x_1, y_1, z_1)…(x_n, y_n, z_n) u niz pitanja podrške.

Tablica 4.67. Preslikavanje putanje grafa područnog znanja u niz pitanja podrške vrsta

pitanja	**vrsta putanje**	**niz pitanja podrške**	**smjer dodavanja u potisni stog**
subjektno	usmjerena	((x_1, y_1, z_1), o)… ((x_{n-1}, y_{n-1}, z_{n-1}), o) ((x_n, y_n, z_n),p)	od zadnjeg elementa
subjektno	neusmjerena	((x_1, y_1, z_1), o)… ((x_n, y_n, z_n), o)	od zadnjeg elementa
objektno	usmjerena	((x_1, y_1, z_1), p) ((x_2, y_2, z_2), s)… ((x_n, y_n, z_n), s)	od prvog elementa
objektno	neusmjerena	((x_1, y_1, z_1), s)… ((x_n, y_n, z_n), s)	od prvog elementa

Tablica 4.67 opisuje i smjer dodavanja elemenata niza pitanja podrške u potisni stog stroja za učenje. Jedino usmjerene putanje će imati pitanje podrške predikatnog tipa koje je kod subjektnog pitanja na kraju niza, a kod objektnog pitanja na početku. Za usmjerene putanje iz primjera (4.120), niz pitanja podrške koji će biti postavljeni na stog je ((c_4, gen^+, c_1), o)((c_1, gen^+, c_3), p) i ((c_5, gen^+, c_2), o)((c_2, gen^+, c_3), o), odnosno učenika će se slijedom kasnije pitati

RAM je vrsta koga ili čega?
Je li radna memorija vrsta memorije?
Disketa je vrsta koga ili čega?
Je li masovna memorija vrsta memorije? (4.122)

S postavljenim pitanjima podrške na potisni stog se završava proces planiranja pitanja podrške gdje se odreĎivanje pitanja podrške temeljilo na pronalaţenju putanje.
Kod učenikovih odgovora koji nemaju točnih, djelomično točnih i pogrešnih fraza odgovora, planiranju pitanja podrške se pristupa na osnovu konceptne restrikcije grafa područnog znanja koju čine okoline koncepata pitanja K i koncepata nedostajućih odgovora K_{nd}, odnosno $GPZ\ \varepsilon(K \cup K_{nd})\ _K$. Tada se za svaka dva koncepta restrikcije traţi usmjerena putanja, i na kraju se skup dobivenih neusmjerenih putanja optimizira. Nadalje se na osnovu tablice 4.67 vrši preslikavanje i dodavanje pitanja podrške na potisni stog stroja za učenje.
Za pitanja nadopune se na stog postavlja prazno pitanje podrške (ε, ε), što stroju za učenje signalizira kako će se sljedeće pitanje temeljiti na već postavljenom pitanju s razlikom što će se pitanju dodati prilog "još" kao što je opisano u primjerima pitanja nadopune sa slika 4.52 i 4.53.
U planiranju pitanja podrške vaţnu ulogu ima mehanizam zaključivanja temeljen na grafu područnog znanja. Mehanizmom zaključivanja se, kod planiranja pitanja podrške, odreĎuju putanje izmeĎu dva koncepta. Osim toga, u poglavlju 4.7.3.6 je navedeno kako mehanizam zaključivanja ima vaţnu ulogu u odreĎivanju točnosti učenikovog odgovora.

4.7.4 Mehanizam zaključivanja

Modul stručnjaka CoLaB Tutor-a posjeduje mehanizam zaključivanja temeljen na pravilima i temeljen na mreţnom prikazu znanja, odnosno grafu područnog znanja. Ako se nad mehanizmom zaključivanja zahtijeva stanje pripadnosti činjenice, gdje je činjenica opisana s dva koncepta i relacijom područnog znanja, onda se koristi mehanizam zaključivanja temeljen na pravilima. Izvršavanjem pravila memorija mehanizma zaključivanja će sadrţavati sve činjenice i zahtijevana činjenica će pripadati područnom znanju ako se nalazi u memoriji.
Mehanizam zaključivanja temeljen na mreţnom prikazu znanja sluţi za pronalaţenje putanje izmeĎu dva koncepta područnog znanja. U ovom slučaju, graf područnog znanja je mreţni prikaz znanja mehanizma zaključivanja.
Komponenta modula stručnjaka CoLaB Tutor-a sadrţi sučelje kojim se mogu postavljati upiti mehanizmu zaključivanja. Upitima se provode zahtjevi nad mehanizmom zaključivanja. S obzirom na temeljenost mehanizma zaključivanja postoje dvije vrste upita. Upitom pripadnosti se odreĎuje je li činjenica upita unutar područnog znanja, dok se upitom za odreĎivanje putanje dobiva niz činjenica koje povezuju koncepte iz upita. Slijede opisi zaključivanja i upita mehanizma zaključivanja temeljenog na pravilima i na mreţnom prikazu znanja.

4.7.4.1 Mehanizam zaključivanja temeljen na pravilima

Područno znanje opisuje činjenice mehanizma zaključivanja temeljenog na pravilima koje predstavljaju eksplicitno znanje. Na osnovu pravila se, temeljem područnog znanja, formiraju nove činjenice koje su za to područno znanje istinite i koje predstavljaju implicitno znanje.

Činjenice, predlošci činjenica, pravila i akcije su notirani LISP sintaksom funkcionalnog programskog jezika C Language Integrated Production System (CLIPS) alata za izgradnju ekspertnih sustava [GIAR1998] [RILExxxx].
Dvije vrste činjenica se nalaze u radnoj memoriji. *Povezanost koncepata* je vrsta činjenice koja opisuje povezanost koncepta x s konceptom z preko relacije y koja moţe biti negirana. Na osnovu jezično označene trojke grafa područnog znanja *l*[*sx, ny, oy*] se tvori činjenica

$$(\text{link } x\ n\ y\ z) \tag{4.123}$$

koja kaţe da je koncept *x* povezan s konceptom *z* preko relacije *y* koja ima oznaku negiranosti *n*. Negiranosti relacije *y* moţe biti $\top$ ili $\bot$ ovisno o tome je li jezična oznaka negiranosti ⌝ ili ⌟. Činjenica o povezanosti koncepta ne uključuje jezične oznake kvantifikatora *s* i *o* niti oznaku veznika *l* jer one ne utječu na izvoĎenje zaključaka. Njihova funkcija je odreĎivanje gramatičke kategorije broja fraze kontroliranog jezika, kao što je opisano u pododjeljku 4.7.2.2.
Tranzitivnost relacije je vrsta činjenice koja govori je li relacija *y* tranzitivna.
Činjenica tranzitivnosti relacije se notira

$$(\text{trans } r\ n) \tag{4.124}$$

gdje je r individualna ili podatkovna uloga područnog znanja, a $n \in \{\top, \bot\}$. Ako je za relaciju *r* atribut $n = \top$ tada je ona tranzitivna, inače nije.
Tablica 4.68 prikazuje primjer područnog znanja i činjenica u tom područnom znanju. Činjenice su adresirane oznakama f1 do f5 radi kasnijeg jednostavnijeg referenciranja.

Tablica 4.68. Primjer područnog znanja i odgovarajućih činjenica

područno znanje	činjenice
$\wedge [\exists a, \lrcorner r^+, \exists b]$	$\text{f1} = (\text{link } a\ \bot\ r^+ b)$
$\wedge [\exists b, \lrcorner r^+, \exists c]$	$\text{f2} = (\text{link } b\ \bot\ r^+ c)$
$\wedge [\exists a, \lrcorner p, = 3]$	$\text{f3} = (\text{link } a\ \bot\ p\ 3)$
$r^+ \in E_{IU}^+$	$\text{f4} = (\text{trans } r^+ \top)$
$p \in E_{PU}$	$\text{f5} = (\text{trans } p\ \bot)$

Činjenice f4 i f5 se odreĎuju na osnovu pripadnosti skupu tranzitivnih individualnih uloga E_{IU}^+.
Pravila modula stručnjaka su oblika U1 … Un => A gdje su U1 … Un uvjeti, a A akcija pravila koja se izvršava ako su uvjeti zadovoljeni. Uvjet pravila je predloţak činjenice, koji se razlikuje od činjenice po tome što koristi varijable. Činjenica je opisana vrijednostima podataka, dok je predloţak činjenice opisan vrijednostima podataka i varijablama. Na primjer, predloţak činjenice (link ?x ?n *r c*) sadrţi varijable ?x i ?n, dok su *r* i *c* vrijednosti podataka. Ovaj primjer predloška činjenice opisuje sve jezično označene trojke grafa područnog znanja koje imaju predikat *r* i objekt *c*. Jedina akcija koja se koristi u pravilima je dodavanje novih činjenica "assert" u memoriju.
Skup pravila modula stručnjaka sadrţi sljedeće vrste pravila:

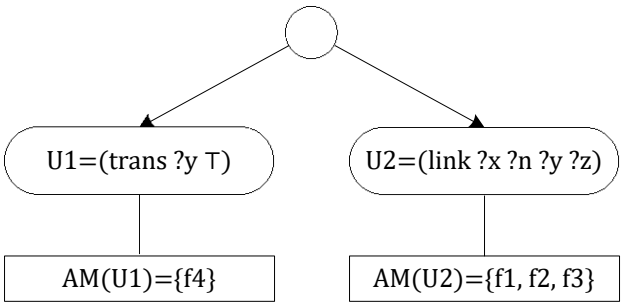
- pravilo tranzitivne povezanosti i klasifikacije
- pravilo pripadnosti individue
- pravila naslje Ďivanja svojstva

Pravilo tranzitivne povezanosti i klasifikacije kaţe ako su koncepti *a* i *b* povezani tranzitivnom relacijom *r* i ako su *b* i *c* povezani istom relacijom, tada se dodaje nova činjenica

koja povezuje koncepte *a* i *c* tranzitivnom relacijom *r*. U LISP notaciji ovo pravilo je adresirano s p1 i glasi

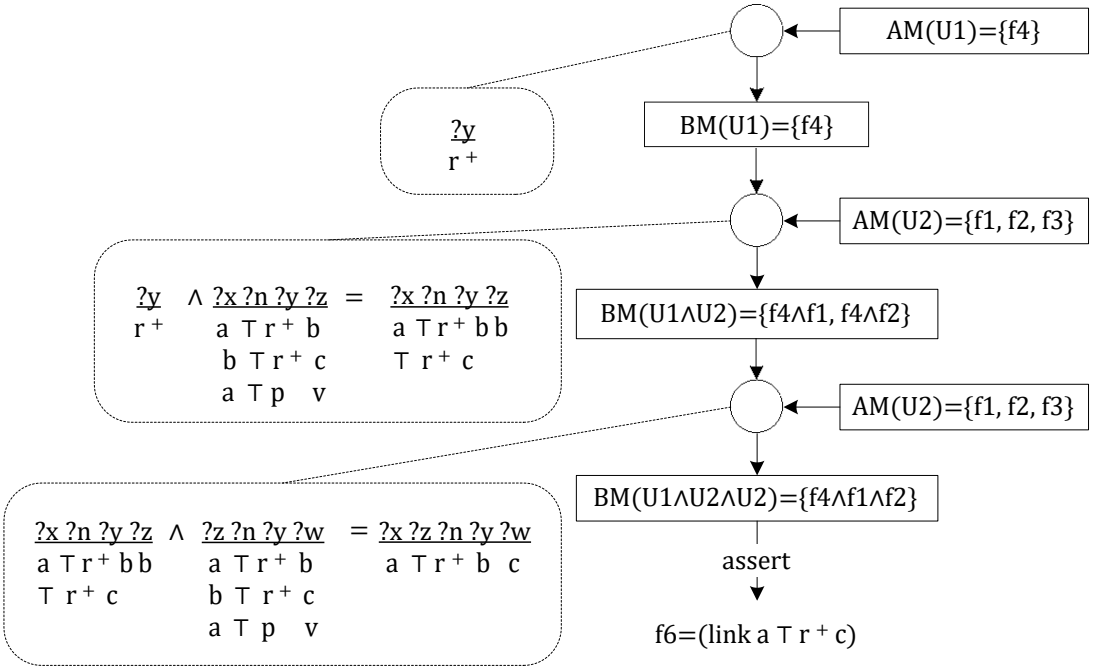
```
(defrule p1
  (trans ?y ⊤)(link ?x ?n ?y ?z)(link ?z ?n ?y ?w)
  =>
  (assert (link ?x ?n ?y ?w)))
```
(4.125)

Zaključivanje se na osnovu pravila i činjenica vrši ulančavanjem unaprijed korištenjem Rete algoritma [FORG1974] [FORG1979] [FORG1982]. Pravilo p1 iz primjera se moţe prikazati u obliku U1 U2 U2 => A, a alfa mreţa Rete algoritma za ovo pravilo i za činjenice iz tablice 4.68 je prikazana na slici 4.55.



Slika 4.55. Alfa mreţa pravila p1

Za uvjet U1, alfa memorija AM od U1 sadrţi činjenicu f4, a AM od U2 sadrţi činjenice f1, f2 i f3. Beta mreţa pravila p1 vrši presjek činjenica iz uvjeta, na osnovu predloţaka činjenica, kao što je prikazano na slici 4.56.



Slika 4.56. Beta mreţa pravila p1

U beta mreţi se vrši konjunkcija uvjeta pravila p1 na osnovu sadrţaja alfa memorije. Beta memorija BM od U1 je jednaka AM(U1) pošto se nije izvršila konjunkcija uvjeta. Iscrtanim likovima i linijama sa slike 4.56 se naglašava operacija konjunkcije činjenica po varijablama

uvjeta. Beta memorija uvjeta U1∧U2 se dobiva konjunkcijom činjenica od BM(U1) i AM(U2). Kao rezultat se dobivaju nove činjenice f4∧f1 i f4∧f2. List beta mreţe je akcija kojom se dodaje nova činjenica f6=f4∧f1∧f2 u radnu memoriju. Činjenica f6 se zatim propagira alfa mreţom do alfa memorije uvjeta U2.
Pravilom pripadnosti individue se odreĎuje pripadnost individue odreĎenoj klasi. Ovo pravilo je definirano s

```
(defrule p2
  (link ?x ⊥ ind+ ?z)(link ?z ⊥ gen+ ?w)
  =>
  (assert (link ?x ⊥ ind+ ?w)))
```
(4.126)

Za individuu *a* kaţemo da će pripadati klasi *b* ako postoji podklasa *c* klase *b* čiji je primjerak individua *a*.
Pravila naslješĎivanja svojstva kaţe da će klasa *a* naslijediti svojstvo *r* ako postoji nadklasa *b* od *a* koja posjeduje svojstvo *r*. NaslješĎivanje svojstava je definirana i za individue, odnosno individua će imati svojstvo *r* ako individua pripada klasi koja je naslijedila svojstvo *r*. Pravilo adresirano s p3 opisuje produkciju za naslješĎivanja svojstava kod klase, dok pravilo p4 vrijedi za individue.

```
(defrule p3
  (link ?x 0 gen+ ?z)(link ?z ?n ?r ?w)
  =>
  (assert (link ?x ?n ?r ?w)))

(defrule p4
  (link ?x 0 ind+ ?z)(link ?z ?n ?r ?w)
  =>
  (assert (link ?x ?n ?r ?w)))
```
(4.127)

Upit koji se postavljaju mehanizmu zaključivanja temeljenom na pravilima je oblika

pripadnost(x, n, y, z) (4.128)

Rezultat upita će biti istina ako alfa memorija Rete mreţe sadrţi činjenicu (link x n y z).

4.7.4.2 Mehanizam zaključivanja temeljen na grafu područnog znanja

Graf područnog znanja je skup trojki (x, y, z) za koje vrijedi definicija **4.5**. Mehanizam zaključivanja temeljen na grafu područnog znanja sluţi za otkrivanje putanje izmeĎu dva zadana koncepta. Putanja grafa područnog znanja GPZ_E je definirana u odjeljku 4.3.2. Za potrebe zaključivanja se uvode dvije vrste putanja. *Usmjerena putanja grafa područnog znanja* GPZ_E je ekvivalentna putanji grafa područnog znanja, dok je *neusmjerena putanja grafa područnog znanja* niz trojki $x_1, y_1, z_1 \;\; x_2, y_2, z_2 \;\; \dots \;\; x_n, y_n, z_n$ za koje vrijedi $x_{i+1}, z_{i+1} \cap x_i, z_i \neq \emptyset$ gdje je $i \in 1,2,\dots,n-1$. Za koncepte x i z kaţemo da su *usmjereno povezani* ako postoji usmjerena putanja grafa $x_1, y_1, z_1 \;\; x_2, y_2, z_2 \;\; \dots \;\; x_n, y_n, z_n$ gdje su $x = x_1$ i $z = z_n$. Analogno kaţemo da su x i z *neusmjereno povezani* ako postoji neusmjerena putanja grafa $(x_1, y_1, z_1)(x_2, y_2, z_2) \dots (x_n, y_n, z_n)$ gdje je $x \in \{x_1, z_1\}$ i $z \in \{x_n, z_n\}$.

Mehanizam zaključivanja je u stanju odrediti usmjerenu i neusmjerenu putanju između dva koncepta. Određivanje putanje se temelji na algoritmu pretraţivanja grafa po dubini koji ima svojstvo praćenja unatrag. Obilazak grafa područnog znanja se vrši po dubini i ako se ne moţe ići "dublje", onda dolazi do vraćanja unatrag. Algoritam 4.4 se odnosi na pronalaţenje usmjerene putanje grafa područnog znanja *GPZ* između zadanih koncepata k_1 i k_2.

Algoritam 4.4. Pronalaţenje usmjerene putanje grafa područnog znanja

```
1   function PronađiUsmjerenuPutanju(k1, k2, GPZ)
2         S = prazni stog
3         P = prazni stog
4         V = prazni skup
5         G = GPZ ε↓ k1 ∪{k1} K
6         for each (x,y,z) ∈ G
7               S.push((x,y,z))
8         end for
9         while S nije prazan
10              x,y,z = S.pop()
11              if z = k2 then
12                    P.push((x,y,z))
13                    return P
14              end if
15              V = V ∪{x,z}
16              ε = ε↓ z \V ∪{x}
17              G = GPZ ε K
18              if G ≠ ∅ then
19                    P.push( x,y,z )
20                    for each (x',y',z') ∈ G
21                          S.push((x',y',z'))
22                    end for
23              else
24                    if S nije prazan i P nije prazan then
25                          a,b,c = S.peek()
26                          do
27                                a',b',c' = P.pop()
28                          while a ≠ a' i P nije prazan
29                    end if
30              end if
31        end while
32        return P
33  end function
```

Kod pronalaţenja putanje koriste se dva stoga i to *S* za koncepte i *P* za trojke grafa područnog znanja. Skup *V* sluţi za pamćenje koncepata koji su se obišli tijekom pronalaţenja putanje. U liniji 5 se skup *G* izjednačava s konceptnom restrikcijom grafa područnog znanja s obzirom na donju okolinu početnog koncepta k_1. Okoline i restrikcije grafa područnog znanja su opisani u odjeljku 4.3.3. Zatim se sve trojke iz skupa *G* prebacuju na stog *S*. U liniji 9 započinje uvjetna petlja koja će se ponavljati sve dok se ne isprazni stog *S*. U petlji se prvo uzme trojka sa stoga *S* i provjerava se je li treći element trojke jednak traţenom konceptu k_2. Ako je jednak, onda je putanja pronađena, pa se zadnja trojka dodaje na stog putanje *P* i stog *P* se vraća kao rezultat funkcije. U suprotnom se skup posječenih koncepata *V* proširuje prvim i trećim elementom trojke, odnosno konceptima trojke. Zatim se u liniji 16 uzima sljedeća donja okolina koncepta iz koje se izbacuju posječeni koncepti. Na osnovu ove okoline se

dobiva konceptna restrikcija grafa područnog znanja koja se sprema u skup *G*. Ako *G* nije prazan, onda se prethodna trojka grafa postavlja u stog putanje *P*, a sve trojke iz skupa *G* se postavljaju na stog *S* i postupak se ponavlja sve dok se *S* ne isprazni ili se ne doĎe do koncepta k_2. Od linije 25 do linije 28 je opisan postupak praţnjenja stoga putanje *P* u slučaju kada trenutni koncept nema okolinu, odnosno kada je konceptna restrikcija grafa područnog znanja na okolini prazna. Tada se, bez izbacivanja sa stoga, uzme jezična trojka (*a*, *b*, *c*) iz *S*, a jezične trojke (*a′*, *b′*, *c′*) sa stoga putanje *P* se izbacuju sve dok se ne zadovolji uvjet *a* = *a*'. Ovime se postiglo vraćanje unatrag do onog elementa putanje od kojeg će se nastaviti pronalaţenje krajnjeg koncepta.
Za neusmjereni graf područnog znanja, algoritam pronalaţenja putanje je sličan algoritmu za usmjerenu putanju. Razlikuje se u tome što

- u liniji 5 se ne koristi samo donja okolina koncepta već i gornja okolina,
- u liniji 11 je uvjet $z = k_2$ ili $x = k_2$,
- u liniji 16 se okolina pronalazi po uvjetu $\varepsilon = \varepsilon\ z\ \cup \varepsilon\ x\ \ \backslash V$,
- u liniji 28 je uvjet prekidanja petlje $a \neq a'$ i $b \neq b'$ i $a \neq b'$ i $b \neq a'$.

Upit koji se postavljaju mehanizmu zaključivanja temeljenom na grafu područnog znanja mogu biti oblika

usmjerena_povezanost(x, y)
neusmjerena_povezanost(x, y) (4.129)

ovisno o tome ţelimo li dobiti usmjerenu ili neusmjerenu putanja grafa područnog znanja za zadane koncepte x i y.

5 Prikaz arhitekture i vrednovanje prototipa sustava CoLaB Tutor

Zamisao i razvoj modela sustava CoLaB Tutor su obuhvatili oblikovanje i implementaciju prototipa kao i primjerenu metodologiju vrednovanja. S tim u vezi u ovom poglavlju se prikazuju odvojeno arhitektura prototipa implementiranog CoLaB Tutor-a i način vrednovanja s odgovarajućim eksperimentalnim skupinama.
Arhitektura prototipa je na najvišoj razini opisana čvorovima tehničke podrške i programskim komponentama, a zatim se struktura temeljnih komponenti prikazuje preko klasa i objekata. Tijekom implementacije razvijalo se područno znanje "Računalo kao sustav" na kojemu su se testirale pojedine funkcionalnosti modela. U prototipnoj verziji nije implementirana podrška za valencijski leksikon glagola, jer oblikovano područno znanje sadrţi samo jednu relacijsku frazu za koju je bilo poţeljno odrediti valenciju glagola. Prilikom oblikovanja područnog znanja, iskoristilo se notacijsko svojstvo elemenata ontologije te su pojedini koncepti dodatno opisani slikovnim zapisima.
Implementirana prototipna verzija je podvrgnuta eksperimentu kojim se istraţuje osjećaj "zadovoljstva" u radu s CoLaB Tutor-om. Za provedbu eksperimenta odabrano je akcijsko istraţivanje [KEMM1988] [LOUI2007]. U akcijskoj fazi akcijskog istraţivanja [MCNI2002] ispitanici su koristili prototip CoLaB Tutor-a. Promatranje u istraţivanju je odraĐeno metodom ankete. Tijek istraţivanja, okruţenje i dobiveni rezultati su opisani u podpoglavlju 5.2.

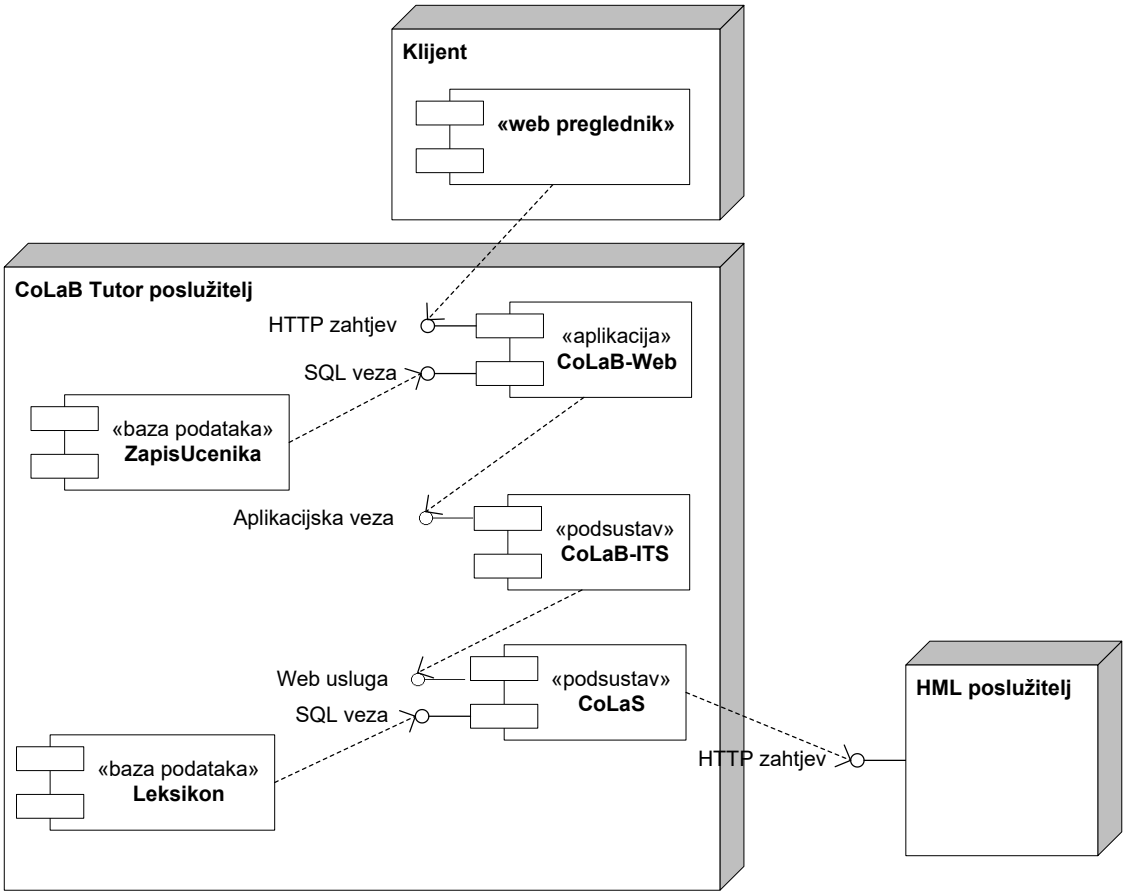
5.1 Arhitektura prototipa sustava CoLaB Tutor

Prototip CoLaB Tutor-a je razvijen u skladu s modelom opisanom u poglavlju 4. Realizacija prototipne verzije sustava je ostvarena Microsoft-ovom .NET 3.5 tehnologijom korištenjem koje su implementirane komponente, komponentna sučelja i skupovi podataka. Izvorni kod je pisan u objektno-orijentiranom programskom jeziku C#, a za formiranje SQL upita, strukturiranje i pretraţivanje XML datoteka i memorijskih kolekcija, korišten je funkcionalni programski jezik LINQ (Language Integrated Query) [MARG2008] [RATT2007]. Sintaksa LINQ upita i struktura podataka je temeljena na C# jeziku. Pomoću LINQ jezika je poprilično pojednostavljenja sintaksa algoritama, a korištenjem anonimnih i generičkih tipova podataka je ujedno olakšano i oblikovanje struktura podataka.
Prototip CoLaB Tutor-a uključuje relacijske baze podataka implementirane korištenjem Microsoft SQL 2008 platforme. Veza izmeĐu aplikacijskih komponenti CoLaB Tutor-a i Microsoft SQL 2008 posluţitelja je ostvarena integriranom komponentom .NET platforme. Web usluga CoLaS podsustava je takoĐer temeljena na integriranim komponentama .NET platforme. Korisničko sučelje, temeljeno na dinamičkim Web stranicama, je ostvareno ASP.NET tehnologijom, a korišteni Web posluţitelj je IIS (Internet Information Services). Kako bi se izbjeglo osvjeţavanje cijele Web stranice, upotrijebila se AJAX (Asynchronous Javascript + Xml) funkcionalnost implementirana kao proširenje ASP.NET tehnologije [EVJE2009].
Web aplikacija, Web usluga i relacijske baze podataka prototipne verzije CoLaB Tutor-a su postavljene na Microsoft Windows 2003 Server posluţitelju.
Arhitektura prototipa je u ovom podpoglavlju pojašnjena pomoću UML dijagrama. Dijagramima se opisuje statička struktura prototipa, odnosno struktura komponenata i klasa.

Za modeliranje relacija baza podataka koristi se E-R dijagram. Prvo se prikazuju osnovne komponente sustava, a zatim se svaka od njih razraĎuje do razine klasa i objekata.

5.1.1 Komponente sustava CoLaB Tutor

Tehnička podrška na kojoj se zasniva prototip CoLaB Tutor-a sastoji se od posluţiteljskog računala CoLaB Tutor-a, posluţiteljskog računala HML-a i klijentskog računala. Komunikacija meĎu računalima se zasniva na HTTP zahtjevima. Klijentsko računalo pomoću Web preglednika pristupa uslugama CoLaB Tutor-a. HTTP zahtjev prema HML posluţitelju, kao i njegov odgovor, se zasniva na XML podacima. Primjer XML zahtjeva i odgovora HML posluţitelja kojim se traţe osnovni oblici riječi od "računalni" i "sustav" je dan u Prilogu B. Organizacija tehničke podrške i povezanost njihovih komponenti su prikazani na slici 5.1.



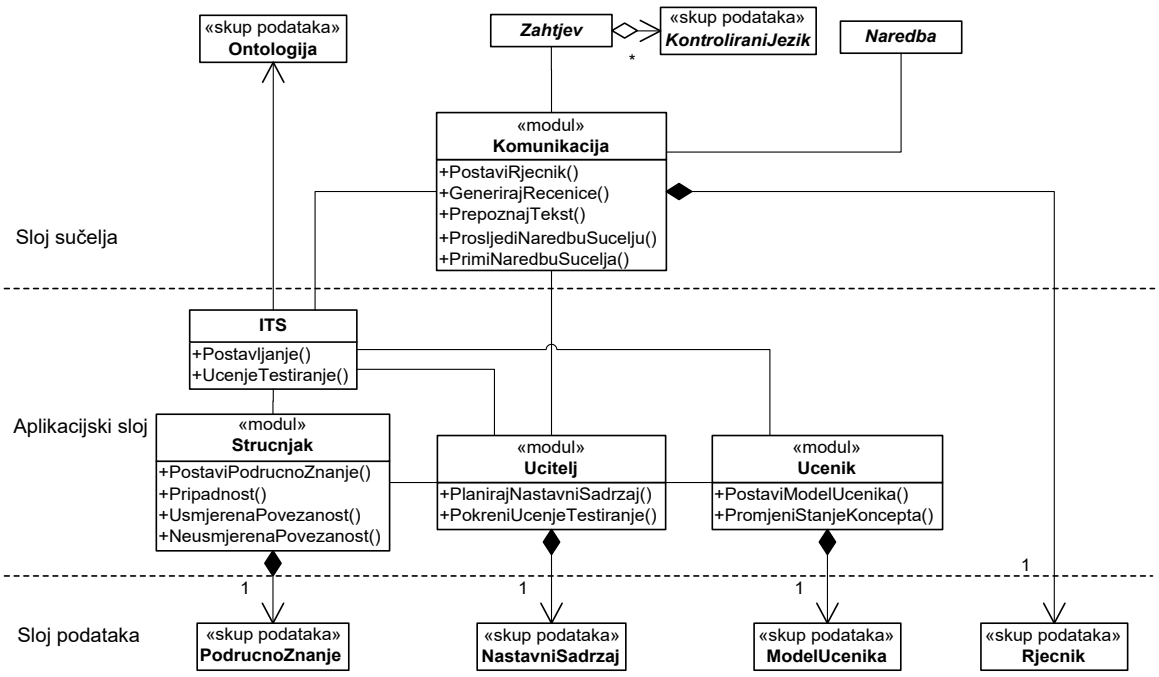
Slika 5.1. Dijagram komponenti prototipa CoLaB Tutor-a

Baza podataka, podsustav i aplikacija su stereotipovi komponenata postavljenih na CoLaB Tutor posluţitelju. CoLaB-ITS je podsustav koji implementira inteligentni tutorski sustav CoLaB Tutor-a. Pri tome CoLaB-ITS preko Web usluge pristupa CoLaS podsustavu kako bi proslijedio zahtjev za obradom kontroliranog jezika. Na nivou obrade riječi kontroliranog jezika, CoLaS moţe pristupiti bazi podataka Leksikon ili HML posluţitelju. Leksikon je baza podataka koja sluţi kao lokalni spremnik riječi hrvatskog standardnog jezika i na taj način se smanjuje opterećenost HML posluţitelja. Osim toga, u bazi podataka Leksikon se nalaze oblici riječi koje još nisu postavljene u HML-u, a odreĎeno područno znanje koristi te riječi radi imenovanja koncepata ili relacija.

CoLaB-ITS je podsustav bez klasa koje implementiraju sučelje prema korisniku. U CoLaB-Web aplikaciji je implementirano korisničko sučelje. Ujedno, preko ove aplikacije se u relacijskoj bazi podataka ZapisUcenika trajno pohranjuju podaci o prijavljenim učenicima, povijest njihovog učenja i testiranja i modeli učenika.
Podsustavi i aplikacija prototipa CoLaB Tutor-a su graĎeni u trorednoj arhitekturi. Njihova arhitektura je opisana dijagramima klasa. Prvo se daje pogled na arhitekturu podsustava CoLaB-ITS pošto su u njemu implementirani osnovni modeli podataka i strojevi s konačnim brojem stanja za voĎenje procesa učenja i testiranja znanja. Strukturno se opisuje podsustav CoLaS i baza podataka Leksikon. TakoĎer se prikazuje struktura aplikacije CoLas-Web i njezine relacijske baze podataka ZapisUcenika.

5.1.2 Podsustav CoLaB-ITS

CoLaB-ITS u aplikacijskom sloju sadrţi module stručnjaka, učitelja i učenika. Sloj podataka sadrţi klase kojima se opisuju skupovi podataka, kao što su područno znanje, nastavni sadrţaj, model učenika i rječnik kontroliranog jezika. U sloju sučelja se nalazi komunikacijski modul koji zahtjevima (klasa Zahtjev) za obradu kontroliranog jezika komunicira s CoLaS podsustavom, a preko naredbi (klasa Naredba) se ostvaruje veza s aplikacijom CoLaB-Web. Nadalje, sloj sučelja sadrţi ontologiju (klasa Ontologija) koja je ulazni skup podataka u fazi postavljanja CoLaB Tutor-a.
Slika 5.2 prikazuje klase podsustava CoLaB-ITS po slojevima troredne arhitekture.



Slika 5.2. Dijagram klasa podsustava CoLaB-ITS

Inicijalizacijom CoLaB-ITS-a se prvo stvara primjerak klase ITS, a zatim primjerci klasa koji predstavljaju module inteligentnog tutorskog sustava. TakoĎer započinje faza postavljanja, u kojoj se poziva metoda Postavljanje() klase ITS kojom se ontologija prenosi do modula koji provode njenu transformaciju u područno znanje (klasa PodrucnoZnanje), nastavni sadrţaj (klasa NastavniSadrzaj), model učenika (klasa ModelUcenika) i rječnik kontroliranog jezika

(klasa Rjecnik). Navedeni skupovi podataka su organizirani po paketima čiji su odnosi opisani u sljedećem pododjeljku.
Faza učenja i testiranja započinje pokretanjem metode UcenjeTestiranje() klase ITS, koja zatim poziva metodu PokreniUcenjeTestiranje() modula učitelja. Tada se po potrebi aktiviraju strojevi s konačnim brojem stanja koji vode proces učenja i testiranja učenika. Opća struktura stroja je u pododjeljku 5.1.2.2 opisana klasama i njihovim asocijacijama.

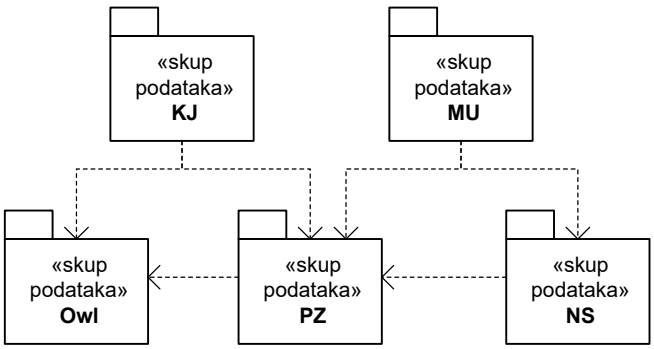
5.1.2.1 Organizacija skupova podataka

Modeli skupova podataka, opisani u podpoglavljima 4.3 i 4.4, su implementirani i organizirani po paketima. Svaki paket sadrţi klase koje modeliraju odreĎeni skup podataka. Nazivi paketa i njihov sadrţaj je opisan u tablici 5.1. Ujedno svaki od paketa je popraćen dijagramom klasa, a prikazani su u Prilogu D.

Tablica 5.1. Paketi skupova podataka

Paket	Sadrţaj	Dijagram klasa
Owl	klase modela OWL ontologije	Prilog D.1
PZ	klase modela područnog znanja	Prilog D.2
NS	klase modela nastavnog sadrţaja	Prilog D.3
MU	klase modela učenika	Prilog D.4
KJ	klase modela kontroliranog jezika	Prilog D.5

Pojedini paketi se referenciraju na klase iz drugih paketa, čime se opisuje njihova meĎusobna zavisnost (slika 5.3).

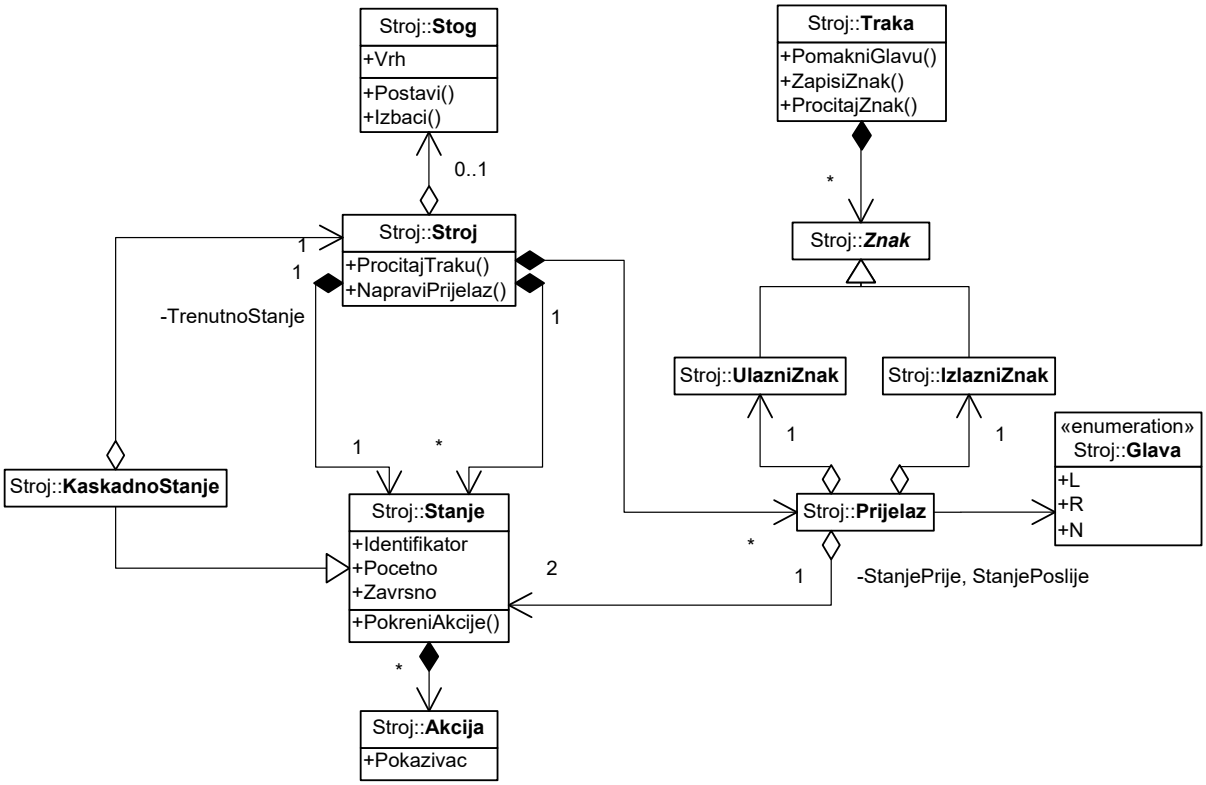


Slika 5.3. Dijagram klasa paketa podataka

Klase paketa Owl oblikuju strukturu ontologije, a na ovoj strukturi se temelji struktura područnog znanja. Rječnik kontroliranog jezika sadrţi fraze kojima su se imenovali elementi ontologije, stoga je paket KJ povezan s paketom Owl. Osim toga, klase koje opisuju rečenice kontroliranog jezika temeljene su na strukturi područnog znanja što uzrokuje zavisnost paketa KJ o paketu PZ. Nadalje, objekti nastavnog sadrţaja se referenciraju na koncepte područnog znanja, a klase modela učenika iz paketa MU se povezuju s klasom objekta nastavnog sadrţaja paketa NS i klasom koncepta iz paketa PZ.

5.1.2.2 Strojevi modula učitelja

Strojevi s konačnim brojem stanja sluţe za voĎenje učenja i testiranja učenika. Oni su dio modela učitelja, a klase koje modeliraju strojeve su prikazane na slici 5.4.

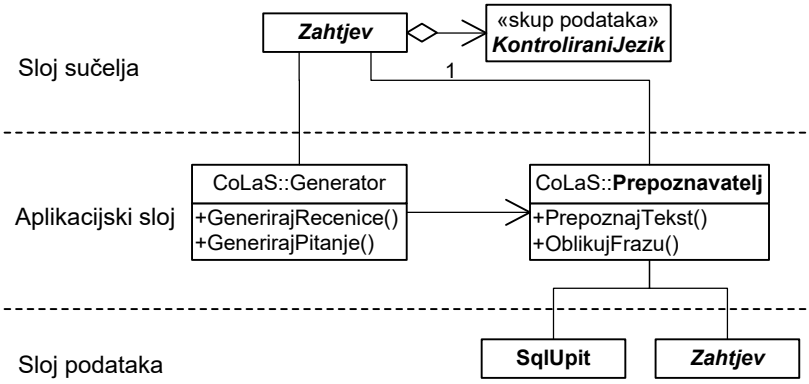


Slika 5.4. Dijagram klasa stroja s konačnim brojem stanja, stogom i glavom za čitanje i pisanje

Stroj je opisan stanjima i prijelazima, a ulazna i izlazna traka stroja sadrţi znakove. Stanje stroja moţe biti početno i završno, a takoĎer moţe uključivati niz akcija. Akcija stanja sadrţi pokazivač na metodu odreĎenog modula CoLaB-ITS-a koja se izvršava kada stroj doĎe u to stanje. Kaskadno stanje je posebna vrsta stanja koje sluţi za prijelaz iz jednog stroja u drugi stroj. Prijelazi stroja su opisani stanjem prije i poslije prijelaza, ulaznim i izlaznim znakom i pomakom glave za čitanje i pisanje. Stroj moţe sadrţavati i stog na kojeg se spremaju znakovi.
Ovim klasama se modeliraju objekti stroja za učenje i testiranje (SUT), stroja za učenje (SU) i stroja za testiranje (ST). Dijagrami objekata pojedinih strojeva su dani u Prilogu E.

5.1.3 Podsustav CoLaS i baza podataka Leksikon

Podsustav CoLaS je namijenjen obradi kontroliranog jezika. U sloju sučelja je implementirana Web usluga kojom se prenose zahtjevi podsustava CoLaB-ITS i vraćaju rezultati obrade. Arhitektura sustava CoLaS je troredna i opisana je dijagramom klasa na slici 5.5.



Slika 5.5. Dijagram klasa podsustava CoLaS

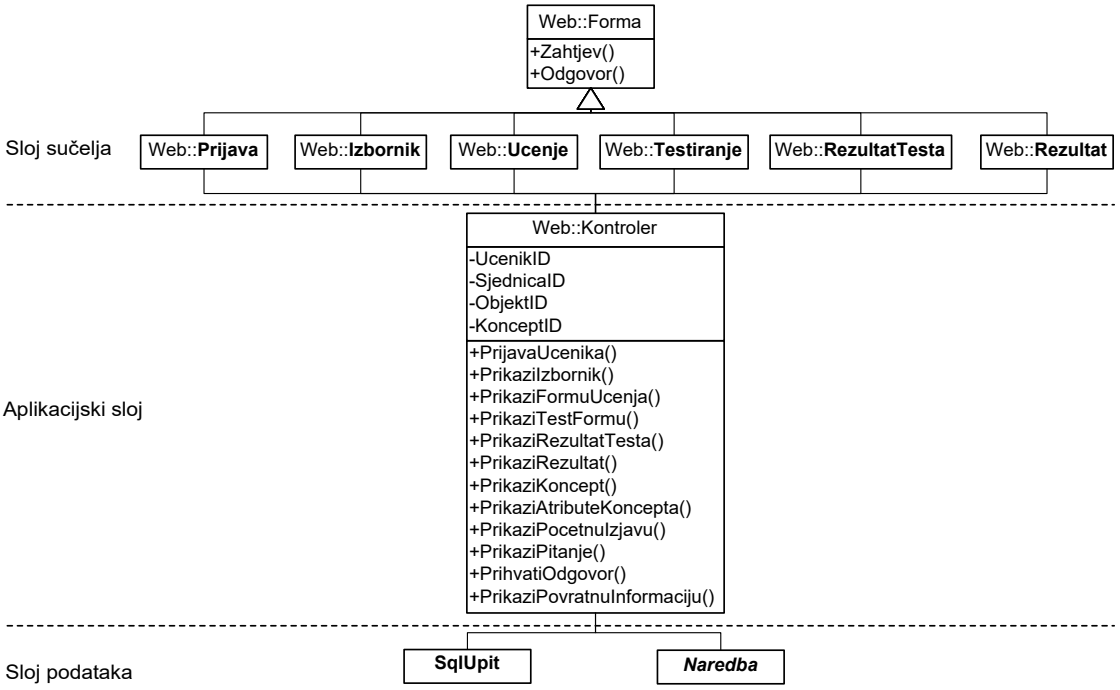
Osnovne dvije klase su Generator koja generira rečenice kontroliranog jezika i Prepoznavatelj koja prepoznaje učenikov tekstualni unos. Prilikom oblikovanja fraza kontroliranog jezika, klasa Prepoznavatelj moţe traţiti oblike riječi u bazi podataka Leksikon preko SQL upita ili moţe proslijediti zahtjev za oblikom riječi HML posluţitelju. Relacijska baza podataka Leksikon sadrţi samo jednu tablicu u kojoj su riječi opisane lemom, oblikom i morfosintaktičkom specifikacijom, kao što je prikazano na slici 5.6.

Slika 5.6. E-R dijagram baze podataka Leksikon

Web usluga u sloju sučelja sluţi za prenošenje zahtjeva za obradom kontroliranog jezika, kao i za vraćanje rezultata obrade. Svaki zahtjev (klasa Zahtjev) je opisan primjerkom apstraktne klase KontroliraniJezik, koji moţe biti riječ, fraza ili rečenica kontroliranog jezika. Primjer zahtjeva kojim se traţi prepoznavanje oblika fraze "računalni sustav" je dan u Prilogu C.

5.1.4 Aplikacija CoLaB-Web i baza podataka ZapisUcenika

Korisničko sučelje prema učeniku je ostvareno CoLaB-Web aplikacijom. Sloj sučelja sadrţi klase kojima su implementirane dinamičke Web stranice. Dinamička Web stranica funkcionira na principu zahtjeva i odgovora. Za interpretaciju zahtjeva i vraćanje odgovora je odgovorna kontrolna klasa u aplikacijskom sloju. Ova klasa je upravljana naredbama CoLaS-ITS podsustava. Za upravljanje učenja i testiranja su odgovorni strojevi modula učitelja čija struktura je opisana u pododjeljku 5.1.2.2. Dijagram klasa aplikacije CoLaB-Web je prikazan na slici 5.7.



Slika 5.7. Dijagram klasa aplikacije CoLaB-Web

Sloj sučelja sadrţi šest klasa kojima se implementiraju pojedine dinamičke Web stranice. U tablici 5.2 je opisan sadrţaj dinamičkih Web stranica.

Tablica 5.2. Dinamičke Web stranice aplikacije CoLaB-Web

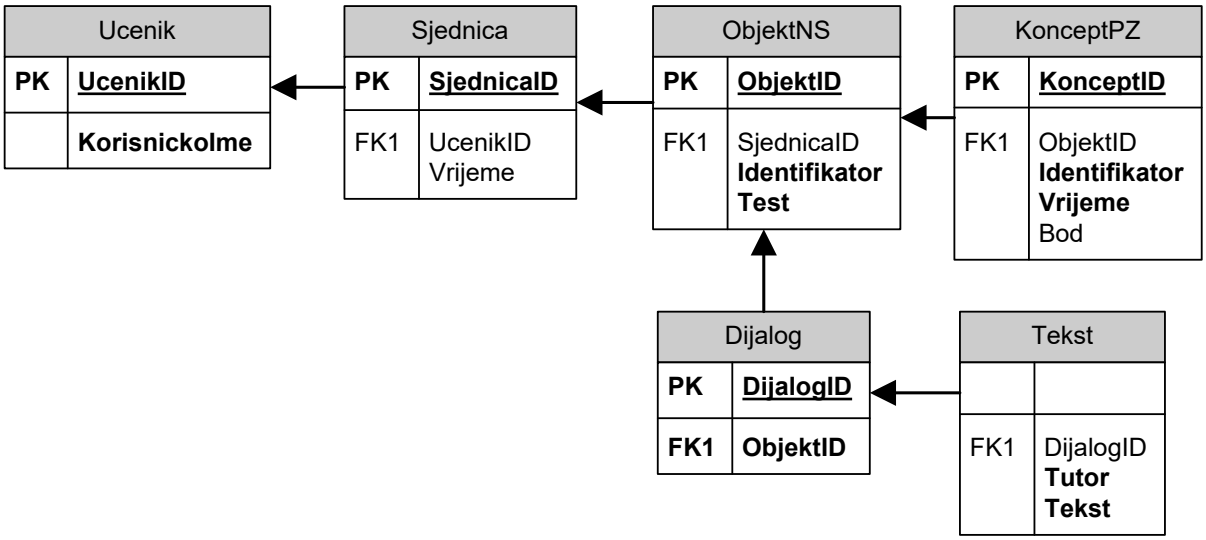
Klasa	Sadrţaj	Izgled stranice
Prijava	unos korisničkog imena, prijava korisnika	slika 5.9
Izbornik	prikaz opcije prelaska na sjedeći objekt nastavnog sadrţaja	slika 5.10 i 5.12
Ucenje	prikaz rečenica kontroliranog jezika za odabrani koncept, prikaz slikovnih zapisa vezanih za odabrani koncept, izbornik sljedećeg i prethodnog koncepta	slika 5.11
Testiranje	prikaz dijaloga na kontroliranom jeziku, unos odgovora	slika 5.13
RezultatTesta	prikaz rezultata testiranja putem dijaloga	slika 5.14
Rezultat	prikaz ocjene cjelokupnog učenja i testiranja	slika 5.15

Temeljna klasa u sloju sučelja je Forma koja sadrţi osnovne metode za ostvarivanje HTTP zahtjeva i HTTP odgovora prema Web posluţitelju.
Za izmjenu dinamičkih Web stranica i za prikazivanje sadrţaja unutar svake stranice je odgovorna Kontroler klasa. Ova klasa prima naredbe od modula učitelja CoLaB-ITS podsustava i na osnovu njih vrši izmjenu i prikaz sadrţaja stranica.
Kako bi se sačuvali rezultati učenja i testiranja prijavljenih učenika, CoLaB-Web aplikacija komunicira s relacijskom bazom podataka ZapisUcenik. U ovoj bazi se unutar šest tablica pohranjuju podaci o učeniku, kao što su:

- korisničko ime učenika,
- vrijeme pristupa sustavu,
- vrijeme pristupa objektu nastavnog sadrţaja,
- vrijeme i broj posjeta koncepta objekta nastavnog sadrţaja namijenjenog učenju,

- vrijeme, broj posjeta i ostvareni bodovi za svaki koncept objekta nastavnog sadrţaja namijenjenog testiranju,
- tekst dijaloga izmeĎu CoLaB-Tutor-a i učenika.

Slika 5.8 prikazuje relacijsku strukturu tablica baze podataka ZapisUcenika.



Slika 5.8. E-R dijagram baze podataka ZapisUcenika

Podaci zapisani u tablicama baze podataka ZapisUcenika su se iskoristili u sljedećem poglavlju kako bi se izvršila analiza rezultata testiranja znanja putem dijaloga s rezultatima tradicionalnog kolokvijalnog ispita. Osim toga, sljedeće poglavlje opisuje eksperimentalne skupine, okruţenje za provedbu eksperimenta, kao i rezultate dobivene provedenim istraţivanjem.

5.2 Vrednovanje prototipa sustava CoLaB Tutor

Prototipna testiranja su obavljena u akcijskom istraţivanju tijekom završne faze izrade doktorske disertacije s temeljnim ciljem istraţivanja i vrednovanja "zadovoljstva" sudionika (studenata) u radu sa sustavom CoLaB Tutor. Osim toga, ţeljelo se kvalitativno prikazati rezultat modeliranja učenika iskazan ocjenom nakon provedenog testiranja. U funkciji tako postavljenog cilja oblikovani su i obraĎeni posebni anketni upitnici te obavljeni razgovori sa sudionicima koji su učestvovali u realizaciji eksperimenata. Ocjene znanja studenata postignute na testiranju se takoĎer prikazuju i interpretiraju. Testiranja su provedena na dvije eksperimentalne skupine sa studentima na Prirodoslovno matematičkom fakultetu i Filozofskom fakultetu Sveučilišta u Splitu. Posebno se prikazuje tijek eksperimenta i analiziraju postignuti rezultati po eksperimentalnim skupinama studenata.

5.2.1 Okruţenje za rad na prototipu sustava CoLaB Tutor

U fazi postavljanja prototipa CoLaB Tutor-a je postavljena ontologija "Računalo kao sustav". Područno znanje nad ovom ontologijom uključuje 88 koncepata, 16 relacija i 39 slikovnih zapisa. Količina pojedinih vrsta koncepata i relacija prikazani su u tablici 5.3.

Tablica 5.3. Količina pojedinih vrsta koncepata i relacija područnog znanja "Računalo kao sustav"

vrsta koncepta	
klasa	51
individua	28
vrijednost podatka	9
vrsta relacije	
individualna uloga	11+2
podatkovna uloga	3

Nad ovakvim područnim znanjem se izvršilo planiranje i oblikovanje nastavnog sadrţaja. Dobiveni nastavni sadrţaj sastoji se od 10 objekata nastavnog sadrţaju i to od 5 objekata namijenjenih učenju i njima odgovarajućih 5 objekata nastavnog sadrţaja namijenjenih testiranju. Tablica 5.4 prikazuje broj koncepata u objektima nastavnog sadrţaja namijenjenih učenju.

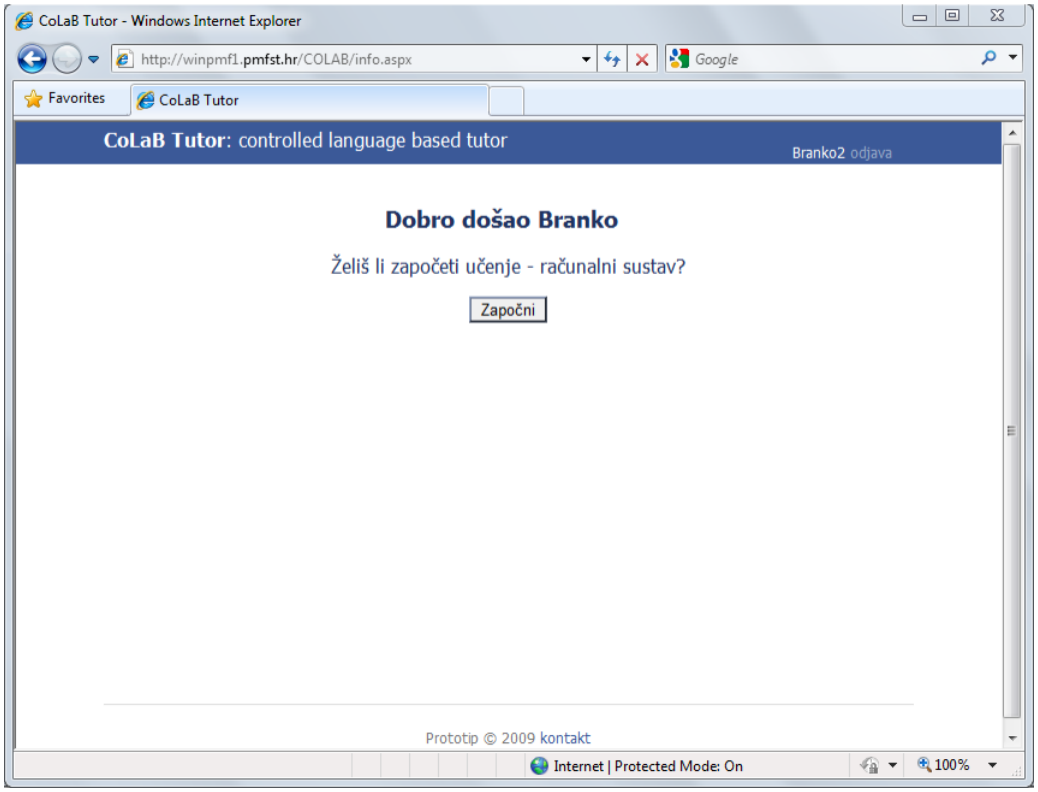
Tablica 5.4. Količina koncepata u objektima nastavnog sadrţaja oblikovanih po područnom znanju "Računalo kao sustav"

objekt nastavnog sadrţaja	
računalni sustav	29
programski jezik	3
model računalnog sustava	17
logički sklop	7
brojevni sustav	5

U fazi učenja i testiranja, učenik se učio i testirao nad ovako oblikovanim nastavnim sadrţajem. Svaki učenik prije početka rada na sustavu CoLaB Tutor je unio korisničko ime radi legalizacije rada na sustavu. Za pristup sustavu nije potrebno unositi korisničku zaporku. Slika 5.9 prikazuje formu za prijavu na sustav.

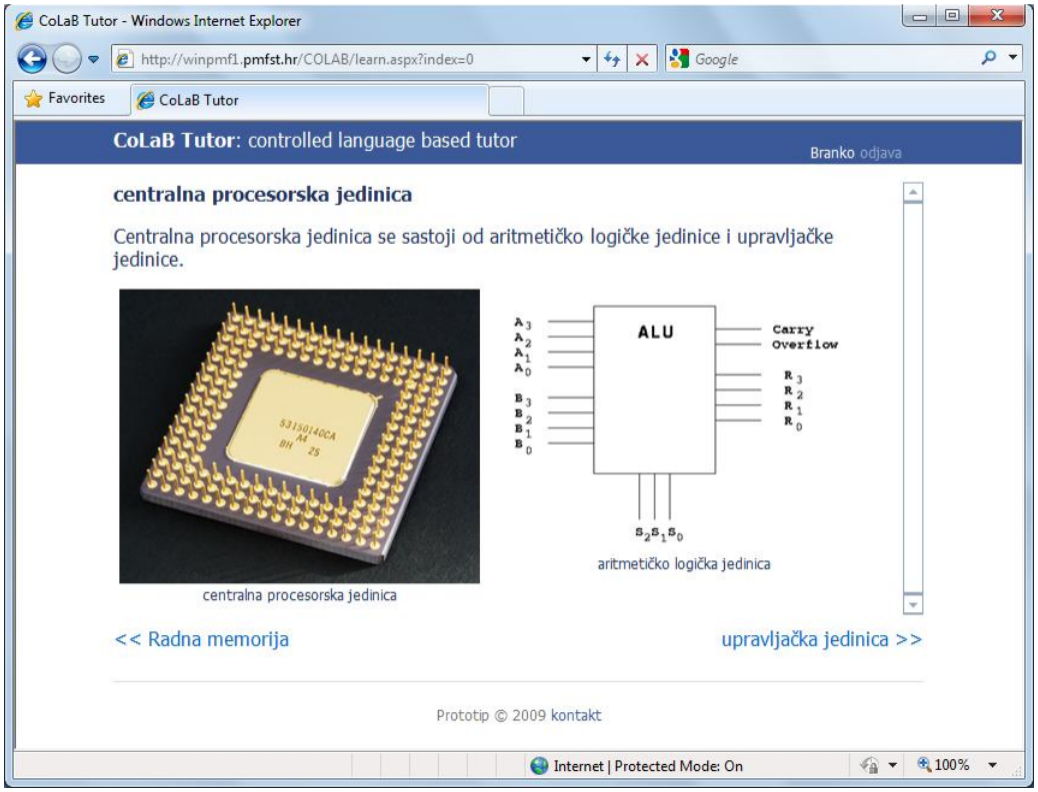
Slika 5.9. Forma za prijavu na sustav

Nakon prijave, učeniku se prikazuje pozdravna poruka i izbornik. Ako se učenik prvi put prijavio, onda se u izborniku prikazuje opcija započinjanja učenja (slika 5.10). Ujedno sustav prikazuje naziv objekta nastavnog sadrţaja kojeg će prezentirati učeniku.



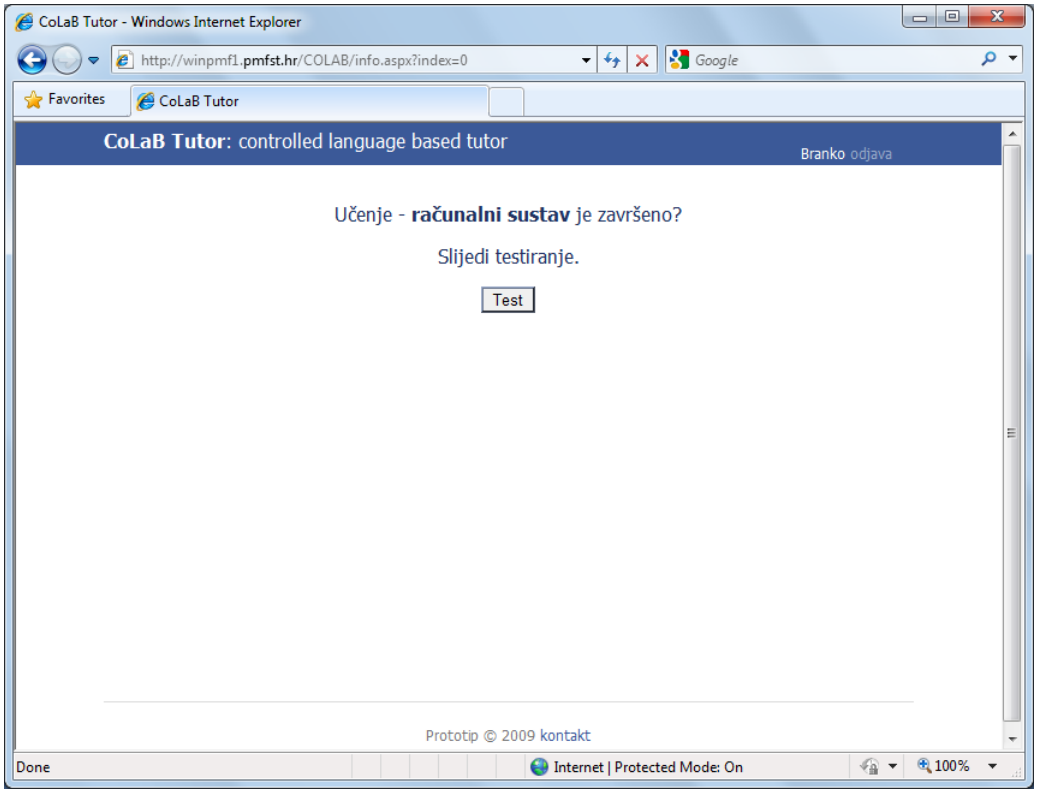
Slika 5.10. Forma izbornika (učenje)

Kada učenik odabere učenje, prvo mu se prezentira rečenica kontroliranog jezika koja se generirala na osnovu prvog koncepta u objektu nastavnog sadrţaja. U generiranoj rečenici se spominju svi koncepti s kojima je aktualni koncept povezan. Za sve ove koncepte se ujedno prikazuju slikovni zapisi, ako ih posjeduju. Nadalje, učeniku se nudi opcija prelaska na sljedeći koncept ili vraćanja na prethodni koncept. Na slici 5.11 aktualni koncept je centralna procesorska jedinica. U strukturi ovog koncepta su aritmetičko-logička jedinica i upravljačka jedinica. Koncepti centralna procesorska jedinica i aritmetičko-logička jedinica imaju slikovne zapise (slika 5.11), koji se takoĎer prikazuju. Pri dnu forme za učenje prikazuje se opcija vraćanja na prethodni koncept (radna memorija) ili opcija prijelaza na sljedeći koncept (upravljačka jedinica) objekta nastavnog sadrţaja namijenjenog učenju.



Slika 5.11. Forma za učenje

Kada učenik preĎe sve koncepte objekta nastavnog sadrţaja namijenjenog učenju, tada mu se nudi opcija prelaska na objekt nastavnog sadrţaja namijenjenog testiranju (slika 5.12).

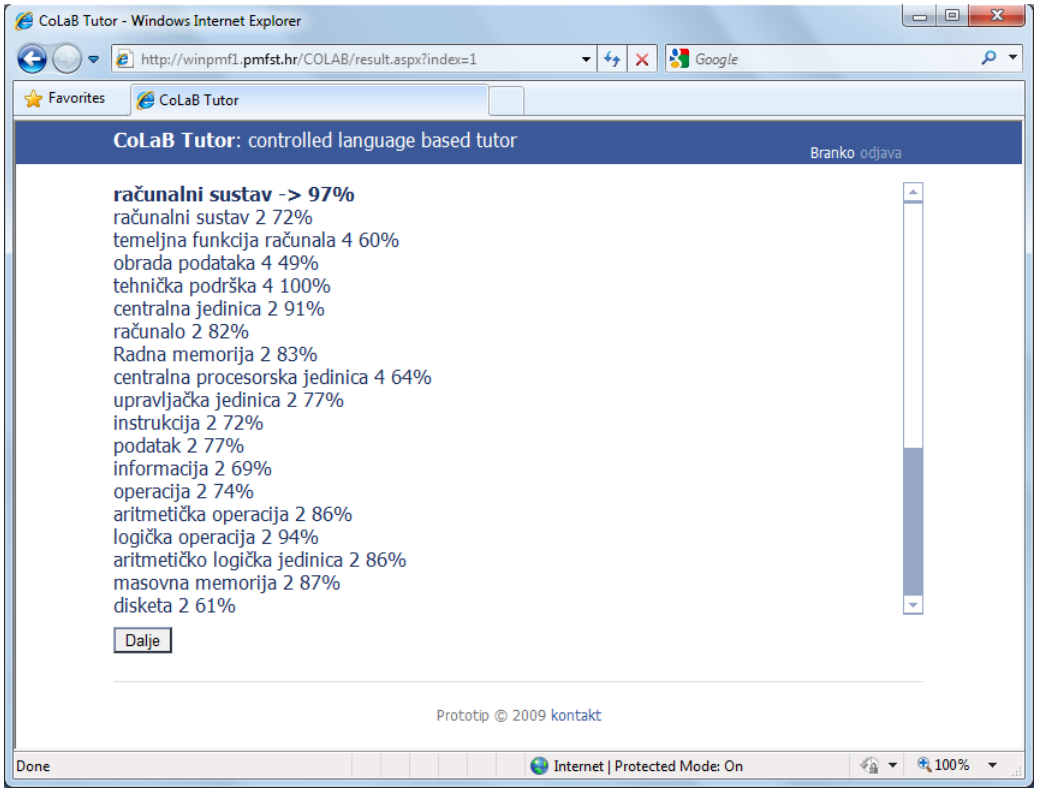


Slika 5.12. Forma izbornika (testiranje)

Odabirom ove opcije započinje dijalog u kojemu CoLaB Tutor postavlja pitanja i čeka na učenikov odgovor. TakoĎer CoLaB Tutor analizira učenikov odgovor i daje povratnu informaciju o točnosti. Slika 5.13 prikazuje formu za testiranje u kojoj se vidi slijed pitanja i odgovora izmeĎu učenika i CoLaB Tutor-a.

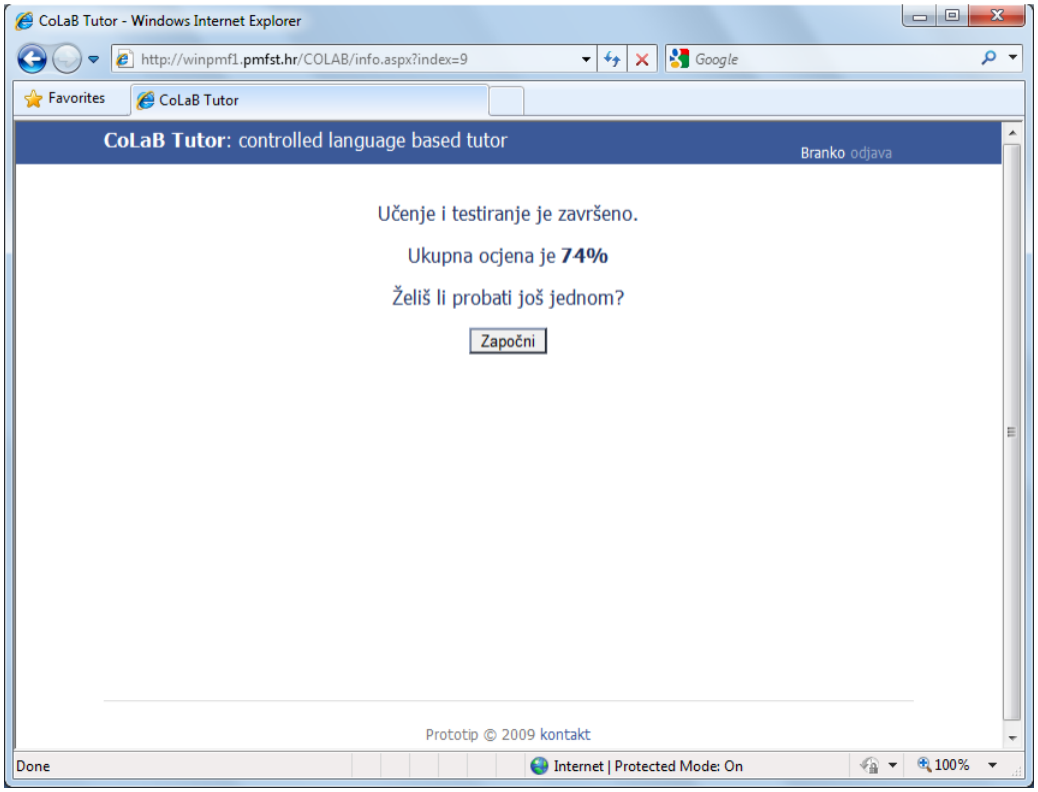
Slika 5.13. Forma za testiranje

Nakon učenikovog odgovora na posljednje pitanje prikazuju se rezultati testa (slika 5.14). Rezultat testiranja je iskazan ukupnom ocjenom za objekt nastavnog sadrţaja namijenjenog testiranju, kao i ocjenama pripadnih koncepata.



Slika 5.14. Forma za prikaz rezultata testiranja

Odabirom opcije "dalje" učeniku se moţe prikazati forma izbornika sljedećeg objekta nastavnog sadrţaja, ako je nastavni sadrţaj ima. U suprotnom se prikazuje ocjena cjelokupnog testiranja formirana na osnovu ocjena svih objekata u nastavnom sadrţaju koji su namijenjeni testiranju (slika 5.15).



Slika 5.15. Forma za prikaz ocjene

U nastavku rada učenik moţe ponovno započeti učenje i testiranje na istom nastavnom sadrţaju ili se odjaviti sa sustava. Rezultati i povijest učenja i testiranja su zapisani u relacijskoj bazi podataka ZapisUcenika.

5.2.2 Prvi eksperiment

Eksperiment je obavljen sa sedamnaest studenata *Prve godine* prediplomskog studija studijske grupe informatike na kolegiju *Uvod u računarstvo* na Prirodoslovno matematičkom fakultetu Sveučilišta u Splitu. Kolegij se odvijao u zimskom semestru akademske 2009/2010. godine. Eksperiment je proveden u okviru redovite nastave u terminu vjeţbi i u trajanju od dva nastavna sata na dan 9.11.2009., a anketa o "zadovoljstvu" u radu sa sustavom CoLaB Tutor (Prilog F.1) je obavljena 16.11.2009. godine.

Studenti su u okviru od jednog nastavnog sata upoznati sa zamisli e-učenja i sustava e-učenja i posebice procesom učenja, poučavanja i testiranja znanja pomoću sustava CoLaB Tutor. U drugom nastavnom satu studenti su se poučavali i testirali svoje znanje. Područno znanje je obuhvatilo osnovne koncepte i raščlambu znanja računalo kao sustav. Popis svih koncepata područnog znanja se nalaze u ontološkom opisu iz Priloga A.
Studenti obuhvaćeni ovim istraţivanjem su se nalazili u tjednu u kojem je, u okviru kolegija Uvod u računarstvo, bio predviĎen Prvi kolokvij u redovitom praćenju studenata tijekom odvijanja nastavnog procesa. Smatralo se da je ovo neposredna i dobra provjera u pripremi studenata za provoĎenje kolokvija predviĎenog u okviru realizacije nastavnog plana i programa odnosnog kolegija. Studenti su ovakvu provjeru prihvatili i s dosta entuzijazma pristupili radu sa sustavom CoLaB Tutor.

5.2.2.1 Prikaz i analiza rezultata testiranja znanja

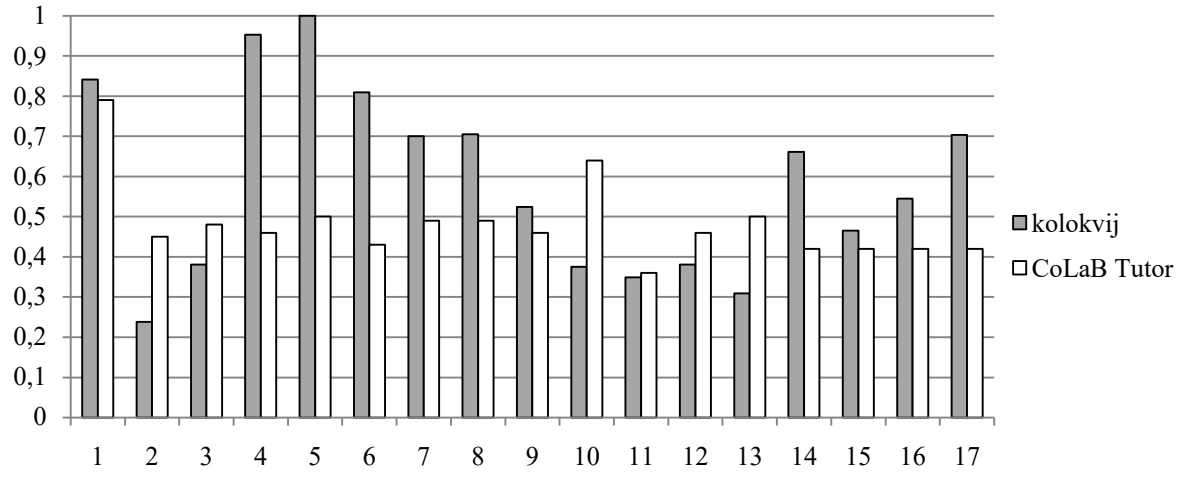
Rezultati testiranja znanja prve eksperimentalne skupine su usporeĎeni s rezultatima prvog kolokvija. Kolokvij se sastojao od 17 pitanja i to 11 pitanja kojima se provjeravalo deklarativno znanje i 6 pitanja za provjeru proceduralnog znanja. U pitanjima proceduralnog tipa se traţila izrada logičkog sklopa i konverzija broja iz jednog brojevnog sustava u drugi. MeĎutim, kako bi student mogao na proceduralan način riješiti zadani problem, morao je poznavati činjenice nad kojima se primjenjuju procedure. Na primjer, prilikom izgradnje logičkog sklopa kojim se interpretira odreĎeni logički izraz, student je morao poznavati svojstva pojedinih logičkih sklopova i logičke funkcije koje oni provode. Nadalje, za konverziju broja iz jednog brojevnog sustava u drugi, student mora poznavati bazu i znamenke pojedinog brojevnog sustava. Što se tiče pitanja kolokvija koja su deklarativne prirode, ona su gotovo ekvivalentna pitanjima koje postavlja CoLaB Tutor za vrijeme dijaloga. Neka pitanja su jednostrukog ili višestrukog odabira, dok na druga pitanja učenik odgovara kratkim esejom.
Zbog navedenih osobina kolokvija, moguće je izvršiti usporedbu rezultata testiranja znanja kolokvija s rezultatima dobivenih testiranjem na CoLaB Tutor-u. Usporedba se temelji na odnosu rezultata svakog pitanja kolokvija s rezultatima koncepata koji se spominju u pitanju. Za pitanje iz kolokvija "*Što je sistemska programska podrška*", skup odgovarajućih koncepata je "*Sistemska programska podrška*", "*Operacijski sustav*", "*Jezični prevoditelj*" i "*Programski jezik visoke razine*". Rezultat pitanja iz kolokvija se oblikovao na osnovu osvojenog boda za to pitanje koji se zatim distribuirao po segmentu [0, 1]. Na primjer, ako je učenik dobio 3

boda na pitanje, a maksimalni broj bodova je 5, onda je rezultat jednak omjeru dobivenog boda i maksimalnog broja bodova, odnosno 0.6. Za skup koncepata se rezultat dobiva aritmetičkom sredinom omjera bodova i normirane teţinske vrijednosti pojedinog koncepta, kao što je za dani skup koncepata prikazan u tablici 5.5.

Tablica 5.5. Skup koncepata i primjer rezultata za pitanje "Što je sistemska programska podrška"

koncept	normirana težinska vrijednost	bod	uspoređeni bod
Sistemska programska podrška	0.82	0.82	1.00
Operacijski sustav	0.80	0.27	0.34
Jezični prevoditelj	0.80	0.50	0.63
Programski jezik visoke razine	0.91	0.34	0.37
		rezultat	0.58

Primjer rezultata za pitanje iz kolokvija iznosi 0.6, a 0.58 je primjer rezultata dobivenog iz skupa koncepata. Graf na slici 5.16 prikazuje usporedbu rezultata po pitanjima iz kolokvija s rezultatima testiranja na CoLaB Tutor-u.



Slika 5.16. Usporedba rezultata testiranja na kolokviju i CoLaB Tutor-u

Srednja vrijednost rezultata svih pitanja iz kolokvija je 0.58, a srednja vrijednost rezultata dobivenih testiranjem na CoLaB Tutor-u je 0.48. Ovakav rezultat je očekivan jer su studenti, prije nego što su se testirali, prisustvovali tradicionalnoj nastavi. To znači da su imali određeno predznanje prije nego što su se testirali na CoLaB Tutor-u. U okruţenju CoLaB Tutor-a su opet usvajali isto znanje i testirali se nad njim, a tri dana poslije se održao kolokvij. Dakle, studenti su prije testiranja znanja na kolokviju učili za vrijeme tradicionalne nastave, kao i uz pomoć CoLaB Tutor-a. U razmaku od tri dana između testiranja na CoLaB Tutor-u i kolokvija su sigurno dodatno i temeljitije učili, što je rezultiralo boljim prosječnim rezultatom na kolokviju. Vaţno je napomenuti kako je, nakon izvršenog istraţivanja i prije kolokvija, deset studenata svojevoljno pristupilo CoLaB Tutor-u kako bi se bolje pripremilo za kolokvij.

5.2.2.2 Analiza anketnog upitnika

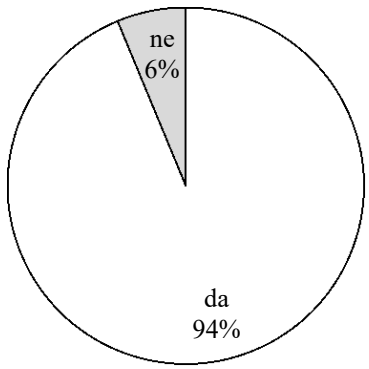
Upitnik je proveden među studentima koji su sudjelovali u eksperimentu, a sastavljen je od osam pitanja zatvorenog tipa i jednog pitanja otvorenog tipa (Prilog F.1). Pitanja anketnog upitnika se mogu podijeliti u četiri dijela. Prvim dijelom upitnika skupljeni su podaci o

korištenju računala za učenje. U drugom dijelu ispitivano je "zadovoljstvo" u korištenju CoLaB Tutor-a za vrijeme učenja i testiranja. Trećim dijelom je ispitan odnos učenja i testiranja znanja uz pomoć CoLaB Tutor-a i tradicionalne nastave. U četvrtom dijelu je ocijenjen CoLaB Tutor.

Prvi dio upitnika se sastoji od jednog pitanja. Na pitanje "Koristiš li računalo dok učiš?" su skoro svi studenti potvrdno odgovorili (tablica 5.6).

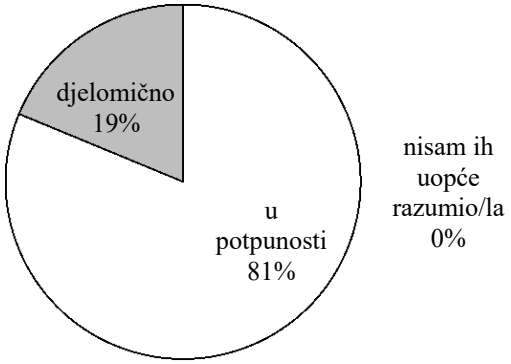
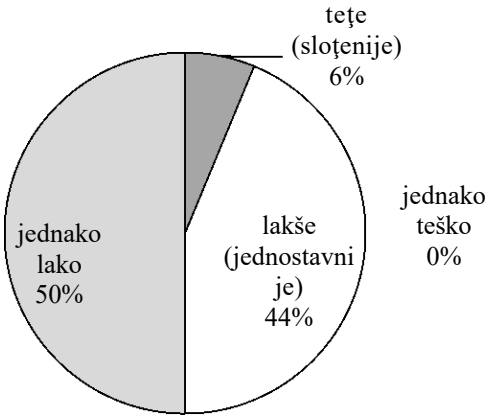
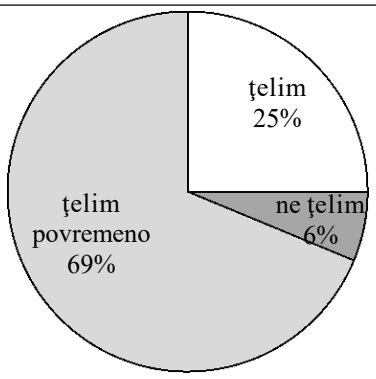
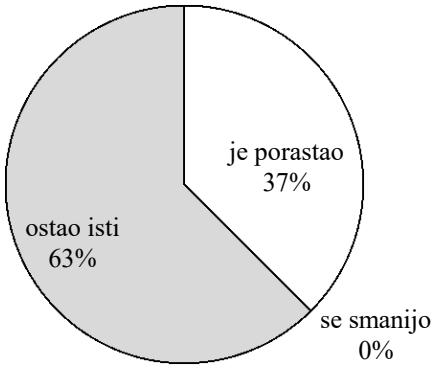
Tablica 5.6. Pitanje i odgovor prvog dijela anketnog upitnika

Pitanje 1
Koristiš li računalo dok učiš?



Većina studenata je praktično upoznata s mogućnošću učenja uz pomoć računala, pa se smatralo kako mogu dati kvalitetan kritički osvrt na učenje uz pomoć CoLaB Tutor-a.
U *drugom dijelu* anketnog upitnika je ispitano zadovoljstvo prilikom učenja i testiranja uz pomoć CoLaB Tutor-a, a rezultati su prikazani u tablici 5.7.

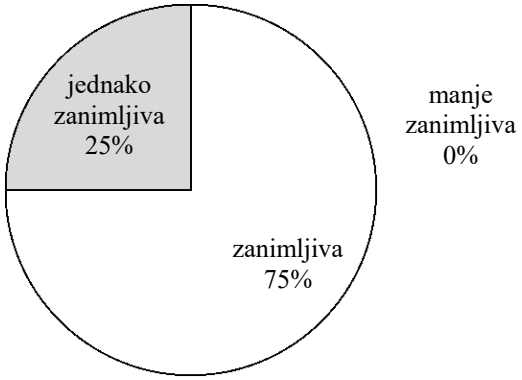
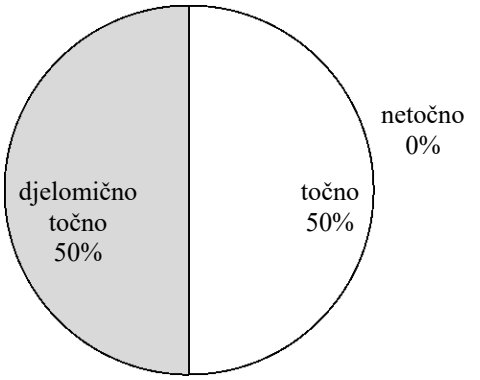
Tablica 5.7. Pitanja i odgovori drugog dijela anketnog upitnika

Pitanje 2	**Pitanje 3**
Nastavni sadrţaj računarstva obraĎen uz pomoć CoLaB Tutor-a razumio/la sam:	Kako je vrijeme odmicalo, učenje računarstva uz pomoć CoLaB Tutor-a mi je bilo:
	
Pitanje 5	**Pitanje 6**
Ţeliš li nastaviti učiti računarstvo uz pomoć CoLaB Tutor-a?	Tijekom učenja nastavnog sadrţaja računarstva uz pomoć CoLaB Tutor-a moj interes za učenjem:
	

Većina studenata je učenje nastavnog sadrţaja uz pomoć CoLaB Tutor-a razumjela u potpunosti, dok je ostatak djelomično razumio. Niti jedan student nije izjavio nerazumijevanje. Kako je vrijeme odmicalo polovici studenata je učenje bilo lakše, a polovici jednako lako. Neznatnom broju studenata je učenje bilo teţe. Skoro svi studenti su izjavili kako općenito ţele nastaviti učiti uz pomoć CoLaB Tutor-a. Njih četvrtina ţeli i dalje koristiti CoLaB Tutor za učenje dok ostali ţele povremeno. Ipak trećini učenika je porastao interes za učenjem, a ostalima je ostao isti. Na osnovi ovog dijela upitnika moţe se zaključiti kako su učenici razumjeli nastavni sadrţaj obraĎen uz pomoć CoLaB Tutor-a i općenito bi ţeljeli i dalje koristiti CoLaB Tutor u nastavi.

U *trećem djelu* je usporeĎeno učenje na klasičnoj nastavi i uz pomoć CoLaB Tutor-a. Tablica 5.8 prikazuje anketna pitanja i postotke odgovora.

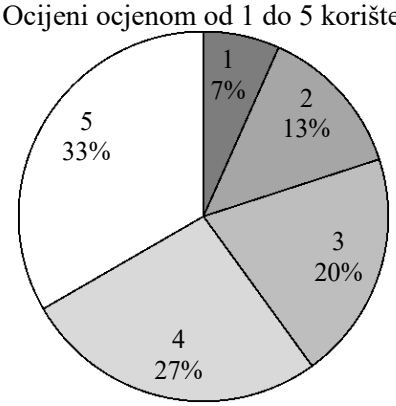
Tablica 5.8. Pitanja i odgovori trećeg dijela anketnog upitnika

Pitanje 4	**Pitanje 7**
Nastava računarstva uz pomoć CoLaB Tutor-a je u odnosu na klasičnu nastavu:	Korištenjem CoLaB Tutor-a u nastavi mogla bi se ostvariti kvalitetnija nastava?
	

Većina studenata smatra kako je učenje uz pomoć CoLaB Tutor-a u odnosu na klasičnu nastavu zanimljivije ili jednako zanimljivo. Na osnovu mišljenja studenata, smatra se kako bi se uvoĐenjem CoLaB Tutor-a korisno nadopunila klasična nastava.

Četvrtim dijelom anketnog upitnika je ocijenjen CoLaB Tutor te je analizirano općenito mišljenje o sustavu. Tablica 5.9 prikazuje ocjenu CoLaB Tutor-a.

Tablica 5.9. Pitanje i odgovor četvrtog dijela anketnog upitnika

Pitanje 8
Ocijeni ocjenom od 1 do 5 korištenje CoLaB Tutor-a!


Što se tiče mišljenja o CoLaB Tutor-u, 93% studenata je izrazilo pozitivno mišljenje, a 54% od njih je dalo zamjerke. Što se tiče pozitivnog mišljenja, studenti su izjavili kako im je pamćenje nastavnog sadrţaja bilo lakše jer su uočili povezanost meĐu konceptima područnog znanja, a pomoglo im je i to što su koncepti popraćeni slikama. Kao veliku prednost su istaknuli mogućnost povezivanja na sustav od kuće.

Jedna od bitnih zamjerki sustava je to što ne podrţava sinonime, odnosno učenici su morali odgovarati na pitanja samo onim konceptima koji su se spominjali tijekom učenjima. Na primjer, ako je traţeni odgovor na pitanje bio "centralna procesorska jedinica", učenik je morao odgovoriti upravo ovim oblikom fraze, a ne s često korištenim sinonima, kao što su "procesor" ili "mikroprocesor".

5.2.3 Drugi eksperiment

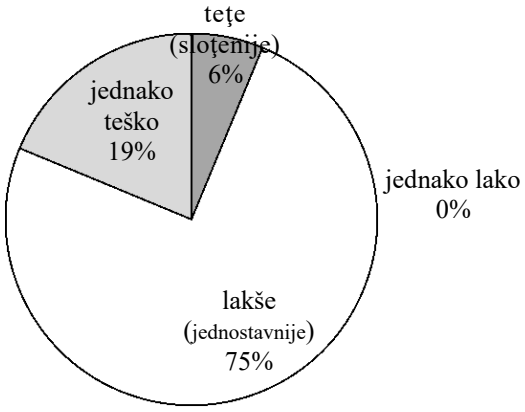
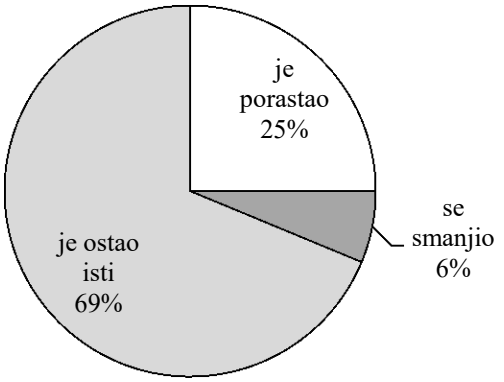
Eksperiment je obavljen sa šesnaest studenata *Pete godine* studijske grupe učitelji razredne nastave integriranog studija na kolegiju *Projektiranje sustava e-učenja* na Filozofskom fakultetu Sveučilišta u Splitu za usmjerenje *primjena informacijske i komunikacijske tehnologije (ICT) u učenju i poučavanju*. Kolegij se odvijao u zimskom semestru akademske 2009/2010. godine. Eksperiment je proveden u okviru redovite nastave u terminu vježbi i u trajanju od dva nastavna sata na dan 20.11.2009., kada je provedena i anketa o "zadovoljstvu" u radu sa sustavom CoLaB Tutor (Prilog F.2).

Studenti su u okviru od jednog nastavnog sata upoznati sa sustavom CoLaB Tutor. U drugom nastavnom satu studenti su se poučavali i testirali svoje znanje. Područno znanje je obuhvatilo osnovne koncepte i raščlambu znanja "Računalo kao sustav". Popis svih koncepata područnog znanja se nalaze u Prilogu A.
Naglašava se da su ovo studenti koji su tijekom do tada realiziranog studija pohađali i položili ispite na više kolegija iz područja primjene ICT u učenju i poučavanju. Posebno je važno kazati da su ovi studenti detaljno upoznati sa funkcionalnostima i strukturom modela TEx-Sys, a na inačici xTEx-Sys su praktično realizirali funkcionalnosti stručnjaka, učitelja i učenika. Upravo zbog ovakvih kvaliteta su i odabrani za provedbu naslovljenog eksperimenata. Smatrali smo da su njihove kompetencije na takvoj razini da mogu s kritičkog stajališta usporediti sustave CoLaB Tutor i xTEx-Sys. S tim u vezi je i priređen anketni upitnik s kojim se željelo utvrditi i vrednovati njihova stajališta.

5.2.3.1 Analiza anketnog upitnika

Upitnik je sastavljen je od šest pitanja zatvorenog tipa i jednog pitanja otvorenog tipa (Prilog F.2). Pitanja anketnog upitnika se mogu podijeliti u tri dijela. Prvim dijelom upitnika skupljeni su podaci o "zadovoljstvu" rada na CoLaB Tutor-u. U drugom dijelu ispitivano je korištenje CoLaB Tutor-a za vrijeme učenja i testiranja u odnosu na xTEx-Sys. Trećim dijelom se ocjenjuje CoLaB Tutor i daje se mišljenje o odnosu CoLaB Tutor-a i xTEx-Sys-a.
Tablica 5.10 prikazuje pitanja i odgovore *prvog dijela* upitnika.

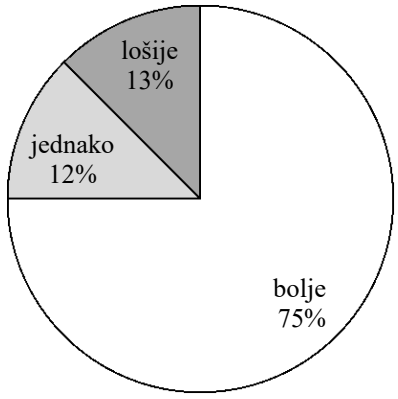
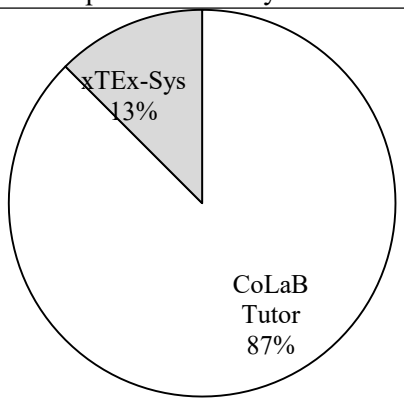
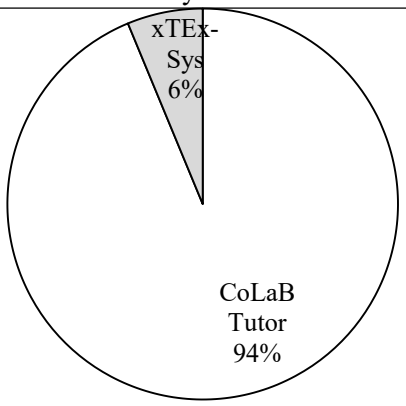
Tablica 5.10. Pitanja i odgovori prvog dijela anketnog upitnika

Pitanje 2	**Pitanje 5**
Kako je vrijeme odmicalo, učenje računarstva uz pomoć CoLaB Tutor-a mi je bilo	Tijekom učenja nastavnog sadržaja računarstva uz pomoć CoLaB Tutor-a moj interes za učenjem
	

Tri četvrtine studenata je izjavilo kako im je lakše učenje uz pomoć CoLaB Tutor-a. Većini preostalih studenata je učenje postalo jednako teško kako je vrijeme odmicalo. Ipak, nekolicini studenata je učenje postalo teţe i sloţenije. Bilo je za očekivati, ako je većini studenata bilo lakše i jednostavnije učenje tada bi im interes za učenje trebao porasti. MeĎutim, samo četvrtini studenata je porastao interes, dok je većini studenata ostao isti.

U *drugom dijelu* je izvršena usporedba CoLaB Tutor-a i xTEx-Sys-a, a u tablici 5.11 su prikazani rezultati.

Tablica 5.11. Pitanja i odgovori drugog dijela anketnog upitnika

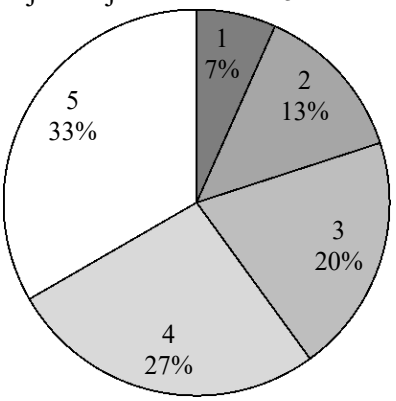
<table>
<tr><th>Pitanje 1</th><th>Pitanje 3</th></tr>
<tr><td>Nastavne sadrţaje oblikovane uz pomoć CoLaB Tutor-a _____ sam razumio/la od nastavnih sadrţaja oblikovanih u xTEx-Sys-u.
 
</td><td>Učenje uz pomoć CoLaB Tutor-a u odnosu na učenje uz pomoć xTEx-Sys je
 
</td></tr>
<tr><th>Pitanje 4</th><th>Pitanje 6</th></tr>
<tr><td>Bi li učio/la računarstvo uz pomoć CoLaB Tutor-a ili uz pomoć xTEx-Sys-a?
 
</td><td>Bi li nastava bila kvalitetnija koristeći CoLaB Tutor ili xTEx-Sys?
 
</td></tr>
</table>

Tri četvrtine studenata je izjavilo kako je nastavne sadrţaje oblikovane uz pomoć sustava CoLaB Tutor bolje razumjelo nego nastavne sadrţaje oblikovane u xTEx-Sys-u. Kod ostale četvrtine studenata, podjednaki broj studenata je jednako i lošije razumio nastavne sadrţaje u CoLaB Tutor-u u odnosu na nastavne sadrţaje u sustavu xTEx-Sys. Što se tiče učenja uz pomoć CoLaB Tutor-a u odnosu na učenje uz pomoć xTEx-Sys, znatnoj većini studenata je učenje uz pomoć CoLaB Tutor-a bilo zanimljivije. Petini studenata je bilo jednako zanimljivo kao i učenje uz pomoć xTEx-Sys-a, a manjem dijelu studenata je bilo manje zanimljivo. Ako bi se studenti morali opredijeliti za sustav koji bi koristili za učenje, skoro svi bi izabrali CoLaB Tutor. TakoĎer preko 90% studenata smatra kako bi nastava bila kvalitetnija ako bi se u njoj koristio CoLaB Tutor.

U *trećem dijelu* anketnog upitnika je ocijenjen CoLaB Tutor te je izneseno mišljenje o odnosu CoLaB Tutor-a i xTEx-Sys-a. Tablica 5.12 prikazuje ocjenu CoLaB Tutor-a.

Tablica 5.12. Pitanje i odgovor trećeg dijela anketnog upitnika

Pitanje 7
Ocijeni ocjenom od 1 do 5 korištenje CoLaB Tutor-a!



Na pitanje otvorenog tipa kojim se traţila usporedba učenja i testiranja u CoLaB Tutor-u i xTEx-Sys-u, većina studenata je bila sklonija CoLaB Tutor-u. Iako su istaknuli neke gramatičke pogreške, koje su posljedica ne implementiranosti podrške za valencijski leksikon glagola, to im nije umanjilo zadovoljstvo korištenja sustava. Prilikom učenja uz pomoć CoLaB Tutor-a istaknuli su pristupačnost i razumljivost sadrţaja te mogućnost vizualnog pamćenja jer su koncepti u generiranim rečenicama popraćeni slikama. Ipak, nekim studentima je zasmetao pravocrtni odabir koncepata, što je povećalo nesnalaţenje u organizaciji objekata nastavnog sadrţaja namijenjenog učenju. Zbog toga su neki studenti skloniji učenju u xTEx-Sys-u jer tijekom učenja slobodnim odabirom mogu odabrati sljedeći pojam iz mreţe pojmova, a i sam prikaz mreţe pojmova im je olakšavao pamćenje. Što se tiče testiranja, svi studenti su bili skloni CoLaB Tutor-u. Afirmativno su naglasili kako je testiranje jednostavnije i razumljivije. NavoĎenje do točnog odgovora, kao i povratna informacija su im poprilično pomogli. Studenti su uočili nepravilno korištenje upitnih zamjenica u generiranim pitanjima, što je posljedica nemogućnosti odreĎivanja da li se pitanje odnosi na ţivu ili neţivu stvar. Nadalje, svidjela im se mogućnost popravljanja pravopisnih pogrešaka kod unosa odgovora.

6 Zaključak

Zamisao novog modela inteligentnog tutorskog sustava se temelji na tradicionalnoj arhitekturi, ali s bitnim poboljšanjem koje je povezano s generiranjem i prepoznavanjem kontroliranog jezika. S tim u vezi, komunikacija znanjem s učenikom je ostvarena kontroliranim jezikom i voĎena od strane strojeva s konačnim brojem stanja. Temeljno znanje CoLaB Tutor-a (znanje stručnjaka) je ontološki opisano i nad njim se zasnivaju znanja učitelja i znanje učenika. U modelu CoLaB Tutor-a struktura znanja je prikazana modelima skupova podataka, a zaključivanje realizacijom funkcionalnosti koju provode komponente sustava.
Temeljni skup podataka je ontologija, odnosno područno znanje opisano ontološkim formalizmom. Nastavni sadrţaj je zasnovan na grupiranju koncepata područnog znanja u objekte nastavnog sadrţaja, što ispunjava uvjete za segmentiranje. Ovime se područno znanje dijeli na objekte nastavnog sadrţaja što je uobičajena praksa pri učenju i testiranju znanja u sustavima e-učenja.
Predstavljena je i zamisao uvjetovane povezanosti objekata nastavnog sadrţaja, pa se učeniku onemogućava nastavak učenja ako nije zadovoljio uvjete na osnovi prethodnog testiranja. U ovoj provjeri uvjeta vaţnu ulogu ima model učenika ostvaren po principu preklapanja. Zamisao preklapanja u modelu učenika se temelji na strukturi ontologije, odnosno učenik moţe pogrešno poimati koncepte područnog znanja ili ih uopće ne poznavati. Ideja komunikacije kontroliranim jezikom se takoĎer temelji na ontologiji. Prikazan je princip preobrazbe ontoloških izraza u rečenice kontroliranog jezika, kao i ideja prepoznavanja učenikovih odgovora. Izbor kontroliranog jezika kao sredstva za komunikaciju znanja je uslijedio upravo zbog formalnih granica izraţajnosti ontologije, kao i ograničenog vokabulara područnog znanja. Testiranje znanja u novom modelu inteligentnog tutorskog sustava je koncipirano dijalogom jednostrane inicijative i ostvaruje se unutar tutorskog okvira.

Inteligentni tutorski sustavi s obradom prirodnog jezika konvergiraju prema dijalogu miješane inicijative koji nije gramatički ni terminološki ograničen. Sustav najbliţi zadanom cilju je AutoTutor jer podrţava postavljanje upita od strane učenika. MeĎutim, učenikovi upiti u AutoTutor-u su ograničeni na zahtjeve za definiranjem učeniku nepoznatih pojmova iz područnog znanja. U literaturi se CIRSCIM-Tutor referencira kao prvi ITS koji koristi uvaţene tehnike obrade prirodnog jezika za realizaciju dijaloga s učenikom. Generiranje prirodnog jezika u dijalogu CIRSCIM-Tutor-a formira bogate izjave s tutorske strane. MeĎutim, učenikovi unosi ograničeni su na jednu ili više riječi koje ne čine rečenicu. Na zavidnom nivou se razumije učenikov unos u Why2-Atlas sustavu. Dijalog u Why2-Atlas sustavu započinje učenikovim unosom eseja izveden od više rečenica koje mogu sadrţavati jednostavne matematičke formule. Razumijevanje eseja se sastoji od pretvorbe u izraze predikatne logike na osnovu kojih se pronalaze učenikova nedostajuća i pogrešna poimanja. Generiranje prirodnog jezika u Why2-Atlas sustavu je realizirano istom komponentom koju koriste i DIAG-NLP verzije sustava. Ipak DIAG-NLP ne implementira razumijevanje prirodnog jezika, već pokazuje kako je samo generiranje dovoljno za povećanje učinkovitosti.

Zamisao CoLaB Tutor-a uključuje neke dobre osobine aktualnih inteligentnih tutorskih sustava s obradom prirodnog jezika. CoLaB Tutor i AutoTutor su generički sustavi, te nemaju ograničenje u upotrebi područnog znanja. Za razliku od AutoTutor-a koji primjenjuje heurističke metode u obradi prirodnog jezika, CoLaB Tutor obradu temelji na prikazu znanja. Prepoznavanje učenikova unosa u CoLaB Tutor-u je slično kao i u CIRSCIM-Tutor-u jer je

ograničeno na fraze. Kod generiranja kontroliranog jezika CoLaB Tutor primjenjuje grupiranje formalnih izjava područnog znanja kako bi se pojednostavile izlazne rečenice. Grupiranje je prisutno i u DIAG-NLP sustavu, međutim ono se zasniva isključivo na dubokoj sintaktičkoj analizi rečenica, a ne na prikazu znanja. Što se tiče dijaloga, CoLaB Tutor posjeduje dobre osobine realizacije dijaloga kroz pet koraka koji opisuju postavljanje pitanja, odgovaranje od strane učenika, davanja povratne informacije, pruţanja podrške i objašnjavanja.

Model CoLaB Tutor-a je na najvišoj razini opisan sudionicima i funkcionalnostima sustava. Osim učenika, koji uči i testira svoje znanje, stručnjak je sudionik odgovoran za opisivanje područnog znanja u skladu s ontološkim formalizmom. Istaknut je redoslijed realizacije skupa funkcionalnosti sustava koji se razlaţe na fazu oblikovanja područnog znanja, fazu postavljanja i fazu učenja i testiranja. Naravno, prva faza, u kojoj sudjeluje stručnjak, je oblikovanje područnog znanja, a ujedno su opisani principi koji pomaţu stručnjaku tijekom oblikovanja. Ulogu učitelja preuzima sam sustav koji u fazi postavljanja planira i generira nastavni sadrţaj na osnovi ontološki opisanog područnog znanja. Osim toga, u fazi postavljanja se inicijalizira model učenika, a ujedno se formira rječnik kontroliranog jezika koji sadrţi sve fraze i riječi iz područnog znanja. Konačno, CoLaB Tutor u fazi učenja i testiranja vodi proces učenja i testiranja znanja učenika.

Realizaciju funkcionalnosti provode komponente i osnovni skupovi podataka. Komponente se razlaţu na podsustave i module. Na razini podsustava se nalazi uređivač ontologije, podsustav ITS, podsustav CoLaS i podsustav HML. Uređivač ontologije koristi stručnjak prilikom ontološkog opisivanja znanja. ITS je podsustav CoLaB Tutor-a koji je odgovoran za realizaciju faze postavljanja i faze učenja i testiranja. Ovaj podsustav komunicira s CoLaS podsustavom namijenjenog obradi kontroliranog jezika. CoLaS obradu morfologije riječi temelji na uslugama HML-a, a koristi i CROVALLEX valencijski leksikon glagola koji nije modeliran kao podsustav već kao skup podataka. Moduli podsustava ITS odgovaraju tradicionalnoj arhitekturi u kojoj se ističu modul stručnjaka, modul učitelja, modul učenika i komunikacijski modul. Svaki od navedenih modula ITS-a se razlaţe na dvije vrste komponenti. Prva vrsta komponenti se koristi u fazi postavljanja kako bi izvršilo postavljanje određenog skupa podataka u ITS-u, a druga vrsta komponenti sluţi tijekom realizacije faze učenja i testiranja. Modul stručnjaka sadrţi komponentu odgovornu za preslikavanje ontološki opisanog područnog znanja u područno znanje po modelu CoLaB Tutor-a i sadrţi mehanizam zaključivanja koji se koristi tijekom učenja i testiranja učenikova znanja. Modul učitelja u fazi postavljanja koristi komponentu za planiranje nastavnog sadrţaja, a tijekom faze učenja i testiranja pokreće strojeve s konačnim brojem stanja sadrţane u komponenti za vođenje učenja i testiranja. Model učenika inicijalno postavlja komponenta modula učenika, a dijagnostička komponenta istog modula se koristi za ažuriranje modela učenika tijekom faze učenja i testiranja. Komunikacijski modul sadrţi komponentu za postavljanje rječnika kontroliranog jezika, a u fazi učenja i testiranja se koristi komponenta odgovorna za prihvat i prikaz kontroliranog jezika. Prilikom prihvata kontroliranog jezika, komunikacijski modul surađuje s modulom za prepoznavanje kontroliranog jezika koji je dio CoLaS podsustava. Osim ovog modula, CoLaS sadrţi modul za generiranje kontroliranog jezika koji pruţa usluge komunikacijskom modulu tijekom prikaza rečenica na kontroliranom jeziku. Kako bi se provela obradu kontroliranog jezika na razini riječi, moduli CoLaS podsustava surađuju s modulima za generiranje i prepoznavanje oblika riječi podsustava HML.
Modeli ulaznih i izlaznih skupova podataka komponenti CoLaB Tutor-a su formalno definirani. Prikazan je model ontološkog opisa područnog znanja čija je izraţajnost temeljena na opisnoj logici, a zapisan je OWL jezikom. Definicija modela područnog znanja je

zasnovana na skupovima uređenih trojki kojima se realizira mreţa koncepata i relacija opisana grafom područnog znanja. Uređene trojke grafa područnog znanja su proširene jezičnim oznakama koje sluţe prilikom obrade kontroliranog jezika. Model nastavnog sadrţaja se definira kao niz objekata nastavnog sadrţaja, a svaki objekt nastavnog sadrţaja je definiran nizom koncepata područnog znanja. Za definiciju modela učenika se koriste funkcije koje svakom konceptu iz objekta nastavnog sadrţaja pridruţuju teţinsku vrijednost i stanje učenja i testiranja, kao i pripadnu ocjenu. Definirana su i proširenja navedenih funkcija na objekte nastavnog sadrţaja.

Kontrolirani jezik se također promatra kao skup podataka čiji se formalni model razlaţe na modele riječi, fraza i rečenica. Riječi kontroliranog jezika su formalno opisane MULTEXT-East normom koja morfološki leksikon definira kao skup uređenih trojki čiji su članovi oblik riječi, osnovni oblik riječi i morfosintaktički opis oblika riječi. Napravljena je definicija morfološkog leksikona kojom se uvodi parametrizacija trojki i njenih članova. Parametri u morfološkom leksikonu sluţe za uvođenje promjenjivih vrijednosti na razini oblika riječi i morfosintaktičkog opisa. Na osnovu parametrizacije se definira relacija uređaja kojom se ostvaruje particija morfološkog leksikona. Ovakva definicija HML-a je omogućila formalno definiranje gramatike fraza kontroliranog jezika. Konceptne fraze su opisane produkcijama nad parametriziranim HML-om. Produkcijska pravila relacijske fraze su proširena uključivanjem specifikacije valencije glagola CROVALLEX leksikona. Model kontroliranog jezika završava opisom kontekstno neovisne gramatike rečenica.

Svaka funkcionalnost faze postavljanja je opisana koracima, po kojima se ulazni skup podataka transformira u izlazni skup podataka, kao i komponentama koje sudjeluju u transformaciji. Prvo se na OWL zapisu ontologije prikazuju koraci transformacije u područno znanje, a slijedi ga proces planiranja strukture nastavnog sadrţaja na osnovu grafa područnog znanja. Dobiveni nastavni sadrţaj nad područnom znanju sluţi za postavljanje inicijalnog modela učenika. Inicijalni model učenika je opisan normiranom teţinskom vrijednošću koncepata svih objekata nastavnog sadrţaja.

U postavljanju rječnika kontroliranog jezika sudjeluju komponente komunikacijskog modula podsustava ITS, modul za prepoznavanje fraza CoLaS podsustava, moduli za generiranje i prepoznavanje riječi HML-a i valencijski leksikon CROVALLEX. U prvom koraku se iz ontologije izdvajaju oblici fraza identificiranih koncepata i relacija, a svaki oblik fraze se rastavlja na nizove oblika riječi. Svaki oblik riječi se prepoznaje pomoću usluga HML-a, a na osnovi gramatike fraza i CROVALLEX se određuju oni prepoznati oblici riječi koji će tvoriti parametriziranu frazu kontroliranog jezika. Na kraju se identifikator koncepta ili relacije zajedno s parametriziranom frazom i oblicima riječi koje je tvore postavlja u rječnik kontroliranog jezika.

Realizacija faze učenja i testiranja opisana u podpoglavlju 4.7 se temelji na strojevima s konačnim brojem stanja. Na najvišoj razini je opisan stroj za učenje i testiranje koji je odgovoran za pravilno nizanje objekata nastavnog sadrţaja.

Kada učenik pristupi objektu nastavnog sadrţaja namijenjenog učenju, tada vođenje preuzima stroj za učenje koji prati nizanje koncepata u objektu nastavnog sadrţaja, a ujedno upravlja postavljanjem stanja učenja u modelu učenika.

Pristup učenika određenom konceptu rezultira generiranjem rečenice kontroliranog jezika koja opisuju odabrani koncept. Proces generiranja rečenica se opisuje na osnovi primjera odabranog koncepta. Prvo se izdvaja segment područnog znanja koji će sluţiti za generiranje. Dobiveni segment se zatim dodatno dijeli i niţe u skupine, a svaki niz skupina će predstavljati jednu apstraktnu rečenicu kontroliranog jezika. Listovi apstraktne rečenice sadrţe identifikatore koncepata ili relacija, na osnovi kojih se provodi pretraţivanje rječnika kontroliranog jezika kako bi se identifikatoru pridruţila parametrizirana fraza. Struktura apstraktne rečenice uvjetuje promjenu parametara parametriziranih fraza, što će na kraju

rezultirati novim oblicima fraza. Konačno na osnovi apstraktnog stabla s parametriziranim frazama se generira tekst rečenice kontroliranog jezika i prikazuje u korisničkom sučelju.

Stroj za testiranje znanja upravlja dijalogom s učenikom. Opisan je stanjima, prijelazima i akcijama koje realiziraju generiranje izjave, generiranje pitanja, prepoznavanje i analizu učenikovog odgovora, planiranje pitanja podrške i generiranje povratne informacije. Izjava se generira ako je prije postavljanja pitanja potrebno ponoviti ono što je učenik već točno odgovorio. Generiranje izjave se realizira slično kao i generiranje rečenica kontroliranog jezika. Pitanja se generiraju s predloškom preko kojeg se određuje vrsta pitanja kao i točni odgovori na pitanje. Prije nego što se provede analiza točnosti učenikova odgovora, ispravljaju se eventualne pravopisne pogreške. Ovo ispravljanje se temelji na Levenstheinovoj metrici koja mjeri udaljenost između riječi učenikovog unosa i riječi iz rječnika kontroliranog jezika. Slijedi izdvajanje fraza iz učenikovog odgovora i njihovo identificiranje s konceptima područnog znanja. Nakon prepoznavanja koncepata iz učenikovog odgovora određuje se njihov položaj u odnosu na koncepte točnog odgovora. Ovakav odnos uvjetuje točnost učenikovog odgovora te uzrokuje planiranje pitanja podrške, ako je potrebno. Točnosti učenikovog odgovora se određuje uz pomoć mehanizma zaključivanja temeljenog na pravilima. Rezultatom zaključivanja se ustvrđuje stanje poznavanja koncepata. Ovaj zaključak služi za generiranje povratne informacije kojom se navodi što je točno, djelomično točno, pogrešno, nedostajuće ili nepoznato u učenikovom odgovoru. Ako učenik nije u potpunosti točno odgovorio na pitanje onda se provodi planiranje pitanja podrške. Za potrebe planiranja se postavljaju upiti mehanizmu zaključivanja temeljenom na grafu područnog znanja. Provedeno zaključivanje rezultira odgovorom s navodom elemenata područnog znanja kojima se može ispraviti pogrešno poimanje učenika. U daljnjem planiranju se, na osnovi odgovora mehanizma zaključivanja, oblikuju pitanja podrške i postavljaju na stog stroja za testiranje. Prelaskom u sljedeće stanje stroja za testiranje se generiraju pitanja podrške i ciklus podrške se ponavlja sve dok se stog ne isprazni.

Zamisao modela CoLaB Tutor-a je omogućila implementaciju prototipne verzije sustava. U prototipu se pomoću uređivača ontologije Protégé opisalo područno znanje "Računalo kao sustav". Podsustavi CoLaB-ITS, CoLaS i aplikacija CoLaB-Web su realizirani Microsoft .NET tehnologijom, a relacijske baze podataka u Microsoft SQL poslužiteljskom okruženju. Prototip je postavljen na jedan poslužitelj koji koristi usluge morfološke obrade riječi HML poslužitelja. Korisničko sučelje za učenika je implementirano tehnologijom dinamičkih Web stranica.
Na prototipu CoLaB Tutor-a je provedeno akcijsko istraživanje na dvije skupine ispitanika. Osnovni cilj eksperimenta je bio ustanovljavanje osjećaja "zadovoljstva" u korištenju CoLaB Tutor-a.
Prvu eksperimentalnu skupinu čine studenti prve godine Prirodoslovno matematičkog fakulteta Sveučilišta u Splitu. Studenti su se testirali na CoLaB Tutor-u tri dana prije nego što su polagali kolokvijalni ispit iz kolegija "Uvod u računarstvo". Pokazalo se kako im je korištenje CoLaB Tutor-a pomoglo u pripremi za kolokvij. Štoviše većina studenata se nakon eksperimenta svojevoljno prijavila i koristila sustav. Studenti su, nakon učenja i testiranja uz pomoć sustava CoLaB Tutor, popunili anketni upitnik u kojem se tražila ocjena "zadovoljstva". Interpretacijom rezultata anketnog upitnika ustanovljeno je da su studenti razumjeli nastavni sadržaj i smatraju kako bi primjena CoLaB Tutor-a utjecala na kvalitetu tradicionalne nastave.
Drugo eksperimentalno istraživanje provedeno je nad studentima pete godine Filozofskog fakulteta Sveučilišta u Splitu. Ovi studenti su od prije bili upoznati s radom na sustavu xTEx-Sys i anketnim upitnikom se nastojalo usporediti korištenje sustava xTEx-Sys i sustava

CoLaB Tutor. Studenti su izjavili kako su bolje razumjeli nastavni sadrţaj prezentiran u CoLaB Tutor-u i općenito su mu dalji bolju ocjenu.

Eksperimenti provedeni nad dvjema skupinama studenata, osim pozitivnih ocjena, dali su zanimljive kritike na implementacijskoj razini prototipa sustava CoLaB Tutor što poticajno djeluje na nastavak istraţivanja. U tom smislu, kontrolirani jezik zasigurno mora podrţati sinonime što je izvedivo na osnovi izjednačavanja koncepata ili relacija prilikom ontološkog opisivanja područnog znanja. MeĎutim, izbor oblika upitnih zamjenica "tko" ili "što" u generiranim pitanjima nije podrţana od strane ontologije. Eventualno rješenje je da se u ontologiji uvede oznaka kojom bi se koncept označio kao ţiva ili neţiva stvar. Nadalje, ontologija dopušta višejezično nazivanje koncepata i relacija. Ako bi se provelo strojno prevoĎenje jezika uspostavom gramatika fraza i rečenica jezika u kojeg se prevodi, bilo bi moguće napraviti višejezičnu verziju sustava CoLaB Tutor.

Prilikom generiranja rečenica kontroliranog jezika provodi se grupiranje jednostavnih rečenica u sloţene. Kada učenik odabere koncept kojeg namjerava naučiti, tada se najčešće generira jedna sloţena rečenica, a ponekad dvije. Ako bi se promijenila paradigma učenja po kojoj bi učenik mogao vidjeti više rečenica koje se odnose na više koncepata, tada je potrebno uvesti zamjenice i nove veznike kako bi se eliminiralo ponavljanje riječi u rečenicama. Ovakav pristup je prisutan kod generiranja prirodnog jezika. Za razumijevanje prirodnog jezika potrebno je omogućiti otkrivanje značenja rečenica koje učenik unese, a semantika rečenice bi se trebala temeljiti na još izraţajnijoj logici kao što je predikatna logika prvog reda. Generiranje i razumijevanje prirodnog jezika su prvi koraci prema uspostavi dijaloga miješane inicijative.

U CoLaB Tutor-u se prilikom planiranja nastavnog sadrţaja generiraju objekti nastavnog sadrţaja nepromjenjivog sadrţaja i redoslijeda. Statička struktura nastavnog sadrţaja nije prilagoĎena svim modelima učenika. Kao buduće rješenje se predlaţe dinamičko planiranje, generiranje i vrednovanje nastavnog sadrţaja tijekom faze učenja i poučavanja, a ne samo u fazi postavljanja. Na osnovi dinamičke strukture nastavnog sadrţaja zasigurno bi se trebala provesti prilagodba modela učenika.

Temeljem svega iznesenog smatramo da su ovim istraţivanjem postignuti sljedeći izvorni znanstveni doprinosi:

- *Oblikovan je inteligentni tutorski sustav s komunikacijskim modulom zasnovanim na obradi kontroliranog jezika koja se provodi na ontološki formaliziranim područnim znanjem*.
  Model inteligentnog tutorskog sustava CoLaB Tutor-a zasniva svoja znanja na ontološki formaliziranom područnom znanju, a komunikacija znanjem je ostvarena kontroliranim jezikom. Područno znanje, nastavni sadrţaj i model učenika su formalno definirana znanja CoLaB Tutor-a temeljem teorije skupova i teorije grafova. Oblikovan je model komponenata CoLaB Tutor-a i opisano je zaključivanje nad navedenim znanjima. Kontrolirani jezik, kao sredstvo za komunikaciju znanjem, je modeliran uz pomoć formalne teorije jezika. Komunikacija znanja je ostvarena preko komunikacijske komponente koja, u suradnji s podsustavom CoLaS, provodi generiranje i prepoznavanje kontroliranog jezika.

- *Modeliran je stroj s konačnim brojem stanja za voĎenje procesa učenja, poučavanja i testiranja znanja*.
  Strojevi s konačnim brojem stanja (znaju što i kada napraviti) su implementirani u modulu učitelja i upravljaju ostalim modulima inteligentnog tutorskog sustava radi voĎenja procesa učenja, poučavanja i testiranja znanja. Modelirana su tri stroja s konačnim brojem stanja. Stroj za učenje i testiranje je namijenjen za voĎenje slijeda objekata nastavnog sadrţaja. Slijedom koncepata područnog znanja u objektu nastavnog sadrţaja namijenjenog učenju upravlja stroj za učenje. Stroj za testiranje upravlja dijalogom s učenikom. Strojevi "imaju pomičnu glavu" koja moţe pisati na ulaznoj traci, a prelaskom u odreĎeno stanje se moţe provesti niz akcija. Akcijom se prenose instrukcije modulima inteligentnog tutorskog sustava koji, nakon provedene obrade, vraćaju rezultate što utječe na prijelaze u sljedeća stanja stroja. Stroj za testiranje je proširen potisnim stogom kako bi se sačuvala generirana pitanja podrške tijekom dijaloga na kontroliranom jeziku.

- *Modelirana je komponenta za obradu hrvatskog standardnog jezika koja uključuje generiranje i prepoznavanje kontroliranog jezika*.
  Komponenta za obradu kontroliranog jezika je osmišljena kao nezavisan sustav. Ovime se omogućava drugim sustavima da koriste obradu kontroliranog jezika. Temeljne funkcionalnosti ove komponente su generiranje i prepoznavanje kontroliranog jezika. Ulaz u proces generiranja kontroliranog jezika su jezično označene trojke koje imaju oblik (subjekt, predikat, objekt). Subjekt i objekt mogu biti u jednini ili mnoţini, dok glagol predikata moţe biti u negiranom obliku. Prepoznavanje kontroliranog jezika ima za ulaz niz fraza, a komponenta je u stanju prepoznati je li svaka od fraza u valjanom obliku. U obradi kontroliranog jezika komponenta se oslanja na Hrvatski morfološki leksikon i CROVALLEX hrvatski valencijski leksikon glagola.

- *Oblikovana su pravila za preslikavanje ontološkog opisa područnog znanja u rečenice kontroliranog jezika prema preporukama za kodiranje morfosintaktičkih opisa*.
  Izraţajnost ontologije uvjetuje izraţajnost rečenica kontroliranog jezika. Svaka izjava iz ontologije se moţe preslikati u adekvatne rečenice kontroliranog jezika, a preslikavanje je opisano pravilima za generiranje rečenica. Pravilima se prvo oblikuje pretvorba ontološkog opisa područnog znanja u apstraktne rečenice kontroliranog jezika. Ovim pravilima je opisan redoslijed frazi koje se pojavljuju u rečenicama. Nadalje, pravilima je opisana promjena oblika frazi u rečenici, što je moguće izvesti jer su oblici riječi u frazama morfosintaktički opisani. Na kraju su dana pravila koja apstraktnu rečenicu s pravilima odreĎenim oblicima fraza preslikava u tekst rečenice.

- *Izveden je prototip modela CoLaB Tutor-a i provedeno vrednovanje primjerenom metodologijom*.
  Implementirana je prototipna verzija CoLaB Tutor-a zajedno s ontologijom "Računalo kao sustav". Eksperimentalno istraţivanje je realizirano sa dvije skupine ispitanika koji su u okruţenju eksperimenta koristili prototip CoLaB Tutor-a s postavljenim područnim znanjem "Računalo kao sustav". Kao metoda istraţivanja je odabrano akcijsko istraţivanje jer su se ţeljeli uočiti potencijalni problemi i djelovati na njih. Planiranjem istraţivanja su osmišljeni anketni upitnici i tijek istraţivanja. Akcija istraţivanja se sastojala od učenja i testiranja ispitanika uz pomoć prototipa CoLaB Tutor-a. U fazi promatranja su popunjeni i interpretirani anketni upitnici. Refleksijom istraţivanja se nagovještava budući rad na poboljšanju sustava.

7 Literatura

[ADVA2004] Advanced Distributed Learning: "*SCORM Overview*"; 2004.

[ANDE1986] J. R. Anderson, C. F. Boyle, G. Yost: *The Geometry Tutor.* The Journal of Mathematical Behavior. 1986. pp. 5-20.

[ANDE1988] J. R. Anderson: *The Expert Module.* In Polson MC, Richardson JJ, editors. Foundations of Intelligent Tutoring System. Hillsdale, New Jersey: Lawrence Erlbaum Associates; 1988. pp. 21-53.

[BAAD2003] F. Baader: *Description Logic Terminology.* In Baader F, McGuinness DL, Nardi D, Patel-Schneider PF, editors. The Description Logic Handbook: Theory, implementation, and applications.: Cambridge University Press; 2003.

[BARI2003] E. Barić, M. Lončarić, D. Malić, S. Pavešić, M. Peti, V. Zečević, et al.: "*Hrvatska gramatika. 4th ed.*" Zagreb: Školska knjiga; 2003.

[BECH2004] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuiness, P. F. Patel-Schneider, et al.: OWL Web Ontology Language Reference. ; 2004; citirano 29.6.2009.; dostupno na: http://www.w3.org/TR/2004/REC-owl-ref-20040210/.

[BECK1988] J. D. Becker: *Unicode 88.* Palo Alto, CA: Xerox Corporation; 1988.

[BECK1996] J. Beck, M. Stern, E. Haugsjaa: *Applications of AI in Education.* Special Issue on Artificial Intelligence. 1996: pp. 11-15.

[BECK2002] H. Beck, S. H. Pinto: *Overview of Approach, Methodologies, Standards, and Tools for Ontologies.* ; 2002.

[BECK2004] D. Becket: RDF/XML Syntax Specification. ; 2004; citirano 29.6.2009.; dostupno na: http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/.

[BERN1996] A. Bernaras, I. Laresgoiti, J. M. Corera: Building and Reusing Ontologies for Electrical Network Applications. In *Proceedings of the European Conference on Artificial Intelligence (ECAI'96)*; 1996; Budapest, Hungary. pp. 298-302.

[BIRO2004] P. V. Biron, K. Permanente, A. Malhotra: XML Schema Part 2: Datatypes Second Edition. ; 2004; citirano 12.7.2009.; dostupno na: http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/.

[BRAC1977] R. J. Brachman: *A Structural Paradigm for Representing Knowledge.* PhD Thesis. Cambridge: Harvard University; 1977.

[BRAC1979] R. J. Brachman: *On the Epistemological Status of Semantic Networks.* In Associative networks: representation and use of knowledge by computers. Findler ed. New York: Academic Press; 1979. pp. 3-50.

[BRAY2006] T. Bray, D. Hollander, A. Layman, R. Tobin: Namespaces in XML 1.0 (Second Edition). ; 2006; citirano 12.7.2009.; dostupno na: http://www.w3.org/TR/2006/REC-xml-names-20060816/.

[BRAY2008] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau: Extensible Markup Language (XML) 1.0 (Fifth Edition). ; 2008; citirano 14.7.2009.; dostupno na: http://www.w3.org/TR/2008/REC-xml-20081126/.

[BRES1986] J. Bresnan: "*The Mental representation of grammatical relations*" Cambridge, Massachusetts: MIT Press; 1982.

[BRIC2004] D. Brickley, R. V. Guha, B. McBride: RDF Vocabulary Description Language 1.0: RDF Schema. ; 2004; citirano 12.7.2009.; dostupno na: http://www.w3.org/TR/2004/REC-rdf-schema-20040210/.

[BROW1980] J. S. Brown, K. VanLehn: *Repair theory: A generative theory of bugs in procedural skills.* Cognitive Science. 1980. 4(4). pp. 379-426.

[BROW1982] J. S. Brown, R. R. Burton, J. De Kleer: *Pedagogical, Natural Language and Knowledge Engineering Techniques in SOPHIE I, II and III.* In Sleeman DH, Brown JS, editors. Intelligent Tutoring Systems. New York: Academic Press; 1982. pp. 227-282.

[BURN1988] H. L. Burns, C. G. Capps: *Foundations of Intelligent Tutoring Systems: An Introduction.* In Polson MC, Richardson JJ, editors. Foundations of Intelligent Tutoring Systems. Hillsdale, New Jersey: Lawrence Erlbaum Associates; 1988. pp. 1-19.

[BURT1982a] R. R. Burton, J. S. Brown: *An Investigation of Computer Coaching for Informal Learning Activities.* In Sleeman DH, Brown JS, editors. Intelligent Tutoring Systems. New York: Academic Press; 1982. pp. 79-98.

[BURT1982b] R. R. Burton: *DEBUGGY: Diagnosis of Errors in Basic Mathematical Skills.* In Sleeman DH, Brown JS, editors. Intelligent Tutoring Systems. New York: Academic Press; 1982. pp. 157-183.

[CARB1970] J. R. Carbonell: *AI in CAI: An Artificial-Intelligence Approach to Computer-Assisted Instruction.* IEEE Transactions on Man Machine Systems. 1970. 11(4). pp. 190-202.

[CARR1977] B. Carr, I. P. Goldstein: *Overlays: A Theory of Modelling for Computer Aided Instruction.* AI Memo. Cambridge: Massachisetts Institute of Technology; 1977. Report No.: 406.

[CHAR1996] E. Charniak: "*Statistical Language Learning*": MIT Press; 1996.

[CHIS2001] M. TH. Chi, S. A. Siler, H. Jeong, T. Yamauchi, R. G. Hausmann: *Learning from human tutoring.* Cognitive Science. 2001. 5 pp. 471-553.

[CHOM1956] N. Chomsky: *Three models for the description of language.* IEEE Transactions on Information. 1956: pp. 113-124.

[CLIPxxxx] CLIPS: A Tool for Building Expert Systems. ; citirano 8.10.2009.; dostupno na: http://clipsrules.sourceforge.net/.

[COLL1975] A. Collins, E. H. Warnock, J. J. Passafiume: *Analysis and Synthesis of Tutorial Dialogues.* In Bower G, editor. The Psychology of Learning and Movivation. New York: New York: Academic Press; 1975. pp. 49-87.

[COLL1982] A. Collins, A. L. Stevens: *Goals and Strategies of Inquiry Teachers.* In Glaser R, editor. Advances in Instructional Psychology. hillsdale, New York: Lavrence Elbraum Edition; 1982. pp. 65-119.

[COWA2004] J. Cowan, R. Tobin: XML Information Set (Second Edition). ; 2004; citirano 20.10.2009.; dostupno na: http://www.w3.org/TR/2004/REC-xml-infoset-20040204/.

[DAVI2008] M. Davis, K. Whistler: Unicode Character Database. ; 2008; citirano 1.9.2009.; dostupno na: http://www.unicode.org/Public/5.1.0/ucd/UCD.html.

[DEUT1962] M. Deutsch: Cooperation and trust: Some theoretical notes. In Jones MR, editor. *Nebraska symposium on motivation*; 1962; Lincon, NE: Nebraska University Press. pp. 275-320.

[DIEU2002] B. Di Eugenio, M. Glass, M. J. Trolio: The DIAG experiments: Natural Language Generation for Intelligent Tutoring Systems. In *INLG02, The Third International Natural Language Generation Conference*; 2002. pp. 120-127.

[DIEU2005a] B. Di Eugenio, D. Fossati, D. Yu, S. Haller, M. Glass: Aggregation improves learning: experiments in natural language generation for intelligent tutoring systems. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*; 2005; Ann Arbor, Michigan. pp. 50-57.

[DIEU2005b] B. Di Eugenio, D. Fossati, D. Yu, S. Haller, M. Glass: Natural Language Generation for Intelligent Tutoring Systems: a case study. In *Proceeding of the 2005 conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology*; 2005; Amsterdam, Netherlands. pp. 217-224.

[DIEU2008] B. Di Eugenio, D. Fossati, D. Yu, S. Haller, M. Glass: *Be Brief, And They Shall Learn: Generating Concise Language Feedback for a Computer Tutor.* International Journal of Artificial Intelligence in Education. 2008. 18(4)..

[EVAN2003] J. S. Evans: *In Two Minds: Dual Process Accounts of Reasoning.* Trends in Cognitive Sciences. 2003: pp. 454-459.

[EVAN2005] M. Evans, J. A. Michael: "*One-on-One Tutoring by Humans and Computers*": Psychology Press; 2005.

[EVJE2009] B. Evjen, D. Wahlin, D. Reed: "*Professional ASP.NET 3.5 AJAX* ": Wrox; 2009.

[FERN1997] M. L. Fernández, A. Gómez-Pérez, N. Juristo: METHONTOLOGY: From Ontological Art Towards Ontological Engineering. In *Proceedings of the AAAI97 Spring Symposium*; 1997; Stanford, USA. pp. 33-40.

[FERN1999] M. L. Fernández: Overview Of Methodologies For Building Ontologies. In *Proceedings of IJCAI99's Workshop on Ontologies and Problem Solving Methods: Lessons Learned and Future Trends.*; 1999; Stockholm, Sweden. pp. 4.1-4.13.

[FLIE2007] G. Fliedl, C. Kop, J. Vöhringer: *From OWL Class and Property Labels to Human Understandable Natural Language.* In LCNS Natural Language Processing and Information Systems.: Springer Berlin / Heidelberg; 2007. pp. 156-167.

[FORG1974] C. L. Forgy: *A network match routine for production systems.* ; 1974.

[FORG1979] C. L. Forgy: *On the efficient implementation of production systems.* Ph.D Thesys. Pittsburgh, PA, USA: Carnegie-Mellon University; 1979.

[FORG1982] C. L. Forgy: *Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem.* Artificial Intelligence. 1982. 19 pp. 17-37.

[FREE1996] R. Freedman: *Interaction of Discourse Planning, Instructional Planning and Dialogue in an Interactive Tutoring System.* PhD Thesis. Northwestern University; 1996.

[FREE1999] R. Freedman: Atlas: A Plan Manager for Mixed-Initiative, Multimodal Dialogue. In *AAAI-99 Workshop on Mixed-Initiative Intelligence*; 1999; Orlando, Florida. pp. 1-8.

[FREE2000] R. Freedman: Using a Reactive Planner as the Basis for a Dialog Agent. In *Proceedings of the Thirteenth Florida Artificial Intelligence Symposium (FLAIRS-2000)*; 2000; Orlando, Florida. pp. 203-208.

[FUCH2005] N. E. Fuchs, S. Höfler, K. Kaljurand, F. Rinaldi, G. Schneider: *Attempto Controlled English: A Knowledge Representation Language Readable by Humans and Machines.* In Lecture Notes in Computer Science.: Springer Berlin / Heidelberg; 2005. pp. 213-250.

[GAGN1979] M. R. Gagné, J. L. Briggs: "*Principles of Instructional Design*" New York: Holt, Rinehart & Winston; 1979.

[GENE1987] M. R. Genesereth, N. Nilsson: "*Logical Foundations of Artificial Intelligence*" San Mateo, California: Morgan Kaufmann Publishers; 1987.

[GENT1977] D. R. Gentner, D. A. Norman: *The FLOW Tutor: Schemas for Tutoring.* Report No. 7702. San Diego: Center for Human Information Processing; 1977.

[GENT1979] D. R. Gentner: *COACH: A Schema-Based Tutor.* Report No. 7903. San Diego: California University; 1979.

[GERT1998] A. S. Gertner, C. Conati, K. VanLehn: Procedural help in Andes: Generating hints using a Bayesian network student model. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence AAAI-98*; 1998; Madison, WI. pp. 106-111.

[GIAR1998] J. D. Giarratano, G. D. Riley: "*Expert Systems: Principles and Programming*": Course Technology; 1998.

[GOLD1982] I. P. Goldstein: *The genetic graph: A representation for the evolution of procedural knowledge.* In Sleeman DH, Brown JS, editors. Intelligent Tutoring Systems. New York: Academic Press; 1982. pp. 51-77.

[GÓME1996] A. Gómez-Pérez: *A Framework to Verify Knowledge Sharing Technology.* Expert Systems with Application. 1996. 11(4). pp. 519-529.

[GÓME1998] A. Gómez-Pérez: *Knowledge Sharing and Reuse.* In Liebowitz J, editor. The Handbook of Applied Expert Systems.: CRC; 1998. pp. 10.1-10.35.

[GRAE1995] A. C. Graesser, N. K. Person, J. P. Magliano: *Collaborative Dialogue Patterns in Naturalistic One-on-One Tutoring.* Applied Cognitive Psychology. 1995. 9 pp. 359-387.

[GRAE1999] A. C. Graesser, K. Wiemer-Hastings, P. Wiemer-Hastings, R. Kreuz: *Auto Tutor: A simulation of a human tutor.* Journal of Cognitive Systems Research. 1999. 1 pp. 35-51.

[GRAE2000] A. C. Graesser, P. Wiemer-Hastings, K. Wiemer-Hastings, D. Harter, T. R. Group, N. Person: *Using Latent Semantic Analysis to Evaluate the Contributions of Students in AutoTutor.* Interactive Learning Environments. 2000. 8(2). pp. 129-147.

[GRAE2001a] A. C. Graesser, N. K. Person, D. Harter, T. R. Group: *Teaching Tactics and Dialog in AutoTutor.* International Journal of Artificial Intelligence in Education. 2001. 12 pp. 257-279.

[GRAE2001b] A. C. Graesser, K. VanLehn, C. P. Rose, P. W. Jordan, D. Harter: *Intelligent tutoring systems with conversational dialogue.* AI Magazine. 2001. 22 pp. 39-51.

[GRAE2004] A. C. Graesser, S. Lu, G. T. Jackson, H. H. Mitchell, M. Ventura, A. Olney, et al.: *AutoTutor: A tutor with dialogue in natural language.* Behavioral Research Methods, Instruments, and Computers. 2004. 36 pp. 180-193.

[GRAE2005] A. C. Graesser, P. Chipman, B. C. Haynes, O. Andrew: *AutoTutor: An Intelligent Tutoring System with Mixed-initiative Dialogue.* IEEE Transactions in Education. 2005. 48(4). pp. 612-618.

[GRUB1993] T. Gruber: *A Translation Approach to Portable Ontology Specifications.* Knowledge Acquisition. 1993. 5 pp. 199-220.

[GRUB2006] A. Grubišić, S. Stankov, B. Ţitko: An Approach to Automatic Evaluation of Educational Influence. In *Proceedings of the 6th WSEAS International Conference on Distance Learning and Web Engineering*; 2006; Lisbon, Portugal. pp. 22-24.

[GRÜN1994] M. Grüninger, M. S. Fox: The Role of Competency Questions in Enterprise Engineering. In *IFIP WG 5.7 Workshop on Benchmarking. Theory and Practice.*; 1994; Trondheim, Norway.

[HALF1988] H. M. Halff: *Curriculum and Instruction in Automated Tutors.* In Polson MC, Richardson JJ, editors. Foundations of Intelligent Tutoring Systems. Hillsdale, New Jersey: Lawrence Erlbaum Associates; 1988. pp. 79-108.

[HALL1985] M. AK. Halliday: "*An Introduction to Functional Grammar*": Arnold; 1985.

[HAYE2004] P. Hayes: RDF Semantics. ; 2004; citirano 29.6.2009.; dostupno na: http://www.w3.org/TR/2004/REC-rdf-mt-20040210/.

[HEWL2005] D. Hewlett, A. Kalyanpur, V. Kolovski, C. Halaschek-Wiener: Effective Natural Language Paraphrasing of Ontologies on the Semantic Web. In *End User Semantic Web Interaction Workshop, International Semantic Web Conference (ISWC 2005)*; 2005; Galway, Ireland.

[HONK2007] T. Honkela: Philosophical Aspects of Neural, Probabilistic and Fuzzy Modeling of Language Use and Translation. In *IJCNN 2007. International Joint Conference on Neural Networks*; 2007; Orlando, Florida. pp. 2881-2886.

[HORR2003a] I. Horrocks, P. F. Patel-Schneider, F. van Harmelen: *From SHIQ and RDF to OWL: The Making of a Web Ontology Language.* Journal of Web Semantics. 2003. 1(1). pp. 7-26.

[HORR2003b] I. Horrocks, P. F. Patel-Schneider: Reducing OWL Entailment to Description Logic Satisfability. In *International Semantic Web Conference 2003*; 2003; Sanibel Island, Florida. pp. 17-29.

[JOHN1984] L. W. Jonhson, E. Soloway: PROUST: Knowledge-based Program Debugging. In *Proceedings of American Association of Artificial Intelligence Conference*; 1984; Los Altos, CA. pp. 369-380.

[JORD2006] P. W. Jordan, M. Makatchev, U. Pappuswamy, K. VanLehn, P. L. Albacete: A Natural Language Tutorial Dialogue System for Physics. In *Proceedings of International FLAIRS Conference*; 2006; Florida. pp. 521-526.

[JURM2000] D. Jurafsky, J. H. Martin: "*Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*" Russell S, Norvig P, editors. New Jersey: Prentice Hall; 2000.

[KALY2005] A. Kalyanpur, B. Parsia, E. Sirin, B. Cuenca Grau, J. Hendler: *Swoop: A Web Ontology Editing Browser.* Web Semantics: Science, Services and Agents on the World Wide Web. 2005. 4(2). pp. 144-153.

[KEMM1988] S. Kemmis, R. McTaggart: "*The Action Research Planner. 3rd ed.*" Geelong, Victoria: Deakin University Press; 1988.

[KLYN2004] G. Klyne, J. J. Caroll: Resource Description Framework (RDF): Concepts and Abstract Syntax. ; 2004; citirano 29.6.2009.; dostupno na: http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.

[KNIG1994] K. Knight, S. K. Luk: Building a large-scale knowledge base for machine translation. In *Proceedings of the twelfth national conference on Artificial intelligence.*; 1994; Seattle, Washington. pp. 773-778.

[KNUB2004a] H. Knublauch, R. W. Fergerson, N. F. Noy, M. A. Musen: The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications. In *Third International Semantic Web Conference ISWC2004*; 2004; Hiroshima, Japan. pp. 229-243.

[KNUB2004b] H. Knublauch, M. A. Musen, A. L. Rector: Editing description logics ontologies with Protégé OWL Plugin. In *Proceedings of International Workshop on Description Logics DL2004*; 2004; Whistler, BC, Canada.

[KURT2002] V. Kurt, P. W. Jordan, C. RP. Rose, D. Bhembe, M. Böttner, A. Gaydos, et al.: The Architecture of Why2-Atlas: A Coach for Qualitative Physics Essay Writing. In *Proceeding of Intelligent Tutoring Systems ITS 2002*; 2002. pp. 158-167.

[LANE2005] C. H. Lane, K. VanLehn: *Teaching program planning skills to novices with natural language tutoring.* Computer Science Education. 2005 183-201. 15(3)..

[LANG1984] P. Langley, S. Ohlsson: Automated Cognitive Modeling. In *Proceedings of American Association of Artificial Intelligence*; 1984; Los Altos, CA: Morgan Kaufman. pp. 193-197.

[LAVO1997] B. Lavoie, O. Rambow: A Fast and Portable Realizer for Text Generation Systems. In *Proceedings of the Conference on Applied Natural Language Processing (ANLP'97)*; 1997; Washington, DC. pp. 265-268.

[LOUI2007] C. Louis, L. Manion, K. Morrison: "*Metode istraživanja u obrazovanju*" Zagreb, Hrvatska: Naklada Slap ; 2007.

[MAKA2004] M. Makatchev, P. W. Jordan, K. VanLehn: Modeling Students' Reasoning About Qualitative Physics: Heuristics for Abductive Proof Search. In *Proceedings of Intelligent Tutoring Systems ITS'2004*; 2004. pp. 699-709.

[MAKA2005] M. Makatchev, B. S. Hall, P. W. Jordan, U. Pappuswamy, K. VanLehn: Mixed Language Processing in the Why2-Atlas Tutoring System. In *AIED 05 Workshop 8 on Mixed Language Explanations in Learning Environments*; 2005; Amsterdam, Netherlands. pp. 35-42.

[MARG2008] F. Marguerie, S. Eichert, J. Wooley: "*LINQ in Action*" Greenwich, CT: Manning Publications, CO; 2008.

[MCNI2002] J. McNiff, J. Whitehead: "*Action Research: Principles and Practice. 2nd ed.*" London: RoutledgeFalmer; 2002.

[MELČ1988] I. A. Mel'čuk: "*Dependency Syntax: Theory and Practice*" New York: State University of New York Press, Albany; 1988.

[MIKE2008] N. Mikelić Preradović: *Pristupi izradi strojnog tezaurusa za hrvatski jezik.* PhD Thesis. Zagreb: Filozofski fakultet u Zagrebu; 2008.

[MILL2001] B. I. Mills: Using the Atlas Planning Engine to Drive an Intelligent Tutoring System: CIRCSIM-Tutor*Version 3. In *Proceedings of the Fourteenth International Florida Artificial Intelligence Research Society Conference*; 2001; Key West, Florida. pp. 211-215.

[MINS1975] M. Minsky: *A Framework for Representing Knowledge.* In Winston P, editor. Psychology of Computer Vision.: McGraw-Hill; 1975.

[NAHK1989] K. Nakhoon, M. W. Evens, J. A. Michael, A. A. Rovick: CIRCSIM-Tutor: An Intelligent Tutoring System for Circulatory Physiology. In *Computer Assisted Learning, 2nd International Conference, ICCAL '89*; 1989; Dallas, Texas. pp. 254-266.

[NORM1973] D. A. Norman: *Memory, Knowledge and the Answering of Questions.* In Solso LR, editor. Contempoary Issues in Cognitive Psychology: The Loyola Symposium. Washington, DC: Winston & Sons; 1973. pp. 135-165.

[NOYF2000] N. F. Noy, R. W. Fergerson, M. A. Musen: The knowledge model of Protégé-2000: Combining interoperability and flexibility. In *Knowledge Engineering and Knowledge Management Methods, Models, and Tools*; 2000; Juan-les-Pins, France. pp. 69-82.

[NOYM2001] N. F. Noy, D. L. McQuinness: *Ontology Development 101: A Guide to Creating Your First Ontology.* Technical Report SMI-2001-0880. ; 2001.

[ORAI2009] I. Oraić: *Kako razvrstati glagole s elementom se u valencijskome rječniku hrvatskih glagola?* Rasprave Instituta za hrvatski jezik i jezikoslovlje. 2009 Travanj. 34(1). pp. 269-283.

[PATE2004] P. F. Patel-Schneider, P. Hayes, I. Horrocks: OWL Web Ontology Language Semantics and Abstract Syntax. ; 2004; citirano 29.6.2009.; dostupno na: http://www.w3.org/TR/2004/REC-owl-semantics-20040210/.

[PHIL2006] A. Phillips, M. Davis: *Tags for Identifying Languages.* Internet Draft. IETF Network Working Group; 2006. Report No.: RFC 4646.

[QUIL1968] M. R. Quillian: *Semantic Memory.* In Minsky MM, editor. Semantic Information Processing. Cambridge: MIT Press; 1968. pp. 227-270.

[RAMZ1994] K. A. Ramzan, M. W. Evens, J. A. Michael, A. A. Rovick: Architecture of CIRCSIM-Tutor (v.3): A Smart Cardiovascular Physiology Tutor. In *Proceedings of the 7th Annual IEEE Computer-Based Medical Systems Symposium*; 1994; Winston-Salem, NC: IEEE Computer Society Press. pp. 158-163.

[RATT2007] J. CJ. Rattz: "*Pro LINQ: Language Integrated Query in C# 2008*": APRESS; 2007.

[REEV1998] B. Reeves, C. Nass: "*The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places*" New York: CSLI Lecture Notes; 1998.

[REIS1985] B. J. Reiser, J. R. Anderson, R. G. Farrell: Dynamic Student Modelling in an Intelligent Tutor for Lisp Programming. In *Proceedings of the International Joint Conference on Artificial Intelligence*; 1985; Los Altos, California. pp. 8-14.

[REIT2000] E. Reiter, R. Dale: "*Building natural language generation systems*": Cambridge University Press; 2000.

[RICK1997] J. Rickel, L. W. Johnson: Integrating Pedagogical Agents into Virtual Environments. In *Proceedings of the first international conference on Autonomous agents*; 1997; Marina del Rey, CA. pp. 30-38.

[RICK1998] J. Rickel, L. W. Johnson: STEVE: A Pedagogical Agent for Virtual Reality. In *Proceedings of the Second International Conference on Autonomous Agents*; 1998; Minneapolis, Minnesota. pp. 332 - 333.

[ROCH1997] Roche E, Shabes Y, editors: "*Finite-state Language Processing (Language, Speech, And Communication)*": The MIT Press; 1997.

[ROSÉ2000] C. P. Rosé: A Framework for Robust Semantic Interpretation. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics*; 2000; San Francisco, CA: Morgan Kaufmann. pp. 311-318.

[ROSÉ2000a] C. P. Rosé: A Framework for Robust Semantic Interpretation. In *Proceedings of the First Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL '00)*; 2000; Seattle, Washington. pp. 311-318.

[ROSÉ2000b] C. P. Rosé: A Syntactic Framework for Semantic Interpretation. In *Proceedings of the ESSLLI Workshop on Linguistic Theory and Grammar Implementation*; 2000; Birmingham, Great Britain.

[ROSÉ2001] C. P. Rosé, P. Jordan, R. Michael, S. Stephanie, K. VanLehn, A. Weinstein: Interactive Conceptual Tutoring in Atlas-Andes. In *Artificial Intelligence in Education: AI-ED in the Wired and Wireless Future, Proceedings of AI-ED 2001*; 2001; Amsterdam, Holland. pp. 256-266.

[ROSI2000] M. Rosić: *Zasnivanje sustava obrazovanja na daljinu unutar informacijske infrastrukture.* magistarski rad. Zagreb, Hrvatska: Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu; 2000.

[ROSI2004] M. Rosić, V. Glavinić, B. Ţitko: Intelligent Authoring Shell Based on Web Services. In *IEEE International Conference on Intelligent Engineering Systems 2004 - INES 2004*; 2004; Cluj-Napoca, Romania. pp. 50-56.

[ROVI1986] A. Rovick Allen, J. A. Michael: CIRCSIM: An IBM PC computer teaching exercise on blood pressure regulation. In *Proceedings of the 30th International Union of Physiological Sciences*; 1986; Vancouver, Canada.

[SCHA1991] M. S. Schauß, G. Smolka: *Attributive concept descriptions with complements.* Artificial Intelligence. 1991. 48(1). pp. 1-26.

[SCHR1995] G. Schreiber, B. Wielinga, W. Jansweijer: The KACTUS View on the 'O' Word. In *Proceedings of the National Dutch AI Conference NAIC'95*; 1995; Rotterdam, Holland. pp. 159-168.

[SELF1974] J. A. Self: *Student models in computer-aided instruction.* International Journal of Man-Machine Studies. 1974. 6 pp. 261-276.

[SELF1990] J. Self: *Theoretical Foundations for Intelligent Tutoring Systems.* Journal of Artificial Intelligence in Education. 1990. 1(4). pp. 3-14.

[SLAT2000] D. Slater: Interactive Animated Pedagogical Agents: An introduction to an emerging field. In *International Conference on Computers and Human Interaction*; 2000; Atlanta, GA.

[SLEB1986] Sleeman DH, Brown JS, editors: "*Intelligent Tutoring Systems*": Academic Press, Inc.; 1986.

[SMIT1982] L. R. Smith, P. Walker, P. Spool: *The Recognition of Instructional Strategies in the Modelling of Student Acquisition of Problem-Solving Skills.* Report. New Brunswick: NJ: Rutgers-The State University; 1982.

[SOWA1987] J. F. Sowa: *Semantic Networks.* In Shapiro SC, editor. Encyclopedia of Artificial Intelligence. New York: Wiley and Sons; 1987.

[STAN1997] S. Stankov: *Izomorfni model sustava kao osnova računalom poduprtog poučavanja načela voĎenja.* doktorska disertacija. Split, Hrvatska: Fakultet elektrotehnike, strojarstva i brodogradnje, Sveučilište u Splitu; 1997.

[STAN2003] S. Stankov: *Web orijentirana inteligentna hipermedijska autorska ljuska.* Tehnološki projekt. Hrvatska: Ministarstvo znanosti i tehnologije; 2003. Report No.: TP-02/0177-01.

[STAN2004] S. Stankov, V. Glavinić, A. Grubišić: What is our effect size: Evaluating the Educational Influence of a Web-Based Intelligent Authoring Shell? In *IEEE International Conference on Intelligent Engineering Systems 2004 - INES 2004*; 2004; Cluj-Napoca, Romania. pp. 545-550.

[STAN2005] S. Stankov, B. Ţitko, A. Grubišić: Ontology as a Foundation for Knowledge Evaluation in Intelligent E-learning Systems. In *International Workshop on Applications of Semantic Web Technologies for E-Learning (SW-EL'05) in conjunction with 12th International Conference on Artificial Intelligence in Education (AI-ED 2005)*; 2005; Amsterdam, Netherland. pp. 81-84.

[STAN2008] S. Stankov, M. Rosić, B. Ţitko, A. Grubišić: *TEx-Sys Model for Building Intelligent Tutoring Systems.* Computers & Education. 2008. 51(3). pp. 1017-1036.

[STEV1977] A. L. Stevens, A. Collins: The Goal Structure of a Socratic Tutor. In *Proceedings of the National ACM Conference*; 1977. pp. 256-263.

[TADI2003] M. Tadić, S. Fulgosi: Building the Croatian Morphological Lexicon. In *Proceedings of the EACL2003 Workshop on Morphological Processing of Slavic Languages*; 2003; Budimpešta, MaĎarska. pp. 41-46.

[TADI2006] M. Tadić: Croatian Lemmatization Server. In *Formal Approaches to South Slavic and Balkan Languages (FASSBL-5)*; 2006; Sofia, Bulgaria. pp. 140-146.

[TOWN1997] D. M. Towne: *Approximate reasoning techniques for intelligent diagnostic instruction.* International Journal of Artificial Intelligence in Education. 1997. 8 pp. 262-283.

[UNIC2003] The Unicode Consortium: "*The Unicode Standard, Version 4.0*": Addison-Wesley Professional; 2003.

[USCH1995] M. Uschold, M. King: Towards a Methodology for Building Ontologies. In *Workshop on Basic Ontological Issues in Knowledge Sharing*; 1995; Montreal, Canada.

[VANL1982] K. VanLehn: *Bugs are not enough: Empirical studies of bugs, impasses and repairs in procedural skills.* The Journal of Mathematical Behaviour. 1982. 3(2). pp. 3-71.

[VANL1983a] K. VanLehn: Human Procedural Skill Acquisition: Theory, Model and Psychological Validation. In *AAAI 1983*; 1983; Washington, DC. pp. 420-423.

[VANL1983b] K. VanLehn: *Felicity conditions for human skill acquisition: Validating an AI-based theory.* Technical Report C-21. Palo Alto, CA: Xerox Palo Alto Research Center; 1983.

[VANL1988] K. VanLehn: *Student Modelling.* In Polson MC, Richardson JJ, editors. Foundations of Intelligent Tutoring Systems. Hillsdale, New Jersey: Lawrence Erlbaum Associates; 1988. pp. 55-78.

[VANL2007] K. VanLehn, A. C. Graesser, G. T. Jackson, P. Jordan, A. Olney, C. P. Rosé: *When are tutorial dialogues more effective than reading?* Cognitive Science. 2007. 31(1). pp. 3-62.

[WAYE1991] E. C. Way: "*Knowledge Representation and Metaphor*" Norwell, MA, USA: Kluwer Academic Publishers; 1991.

[WENG1987] E. Wenger: "*Artificial intelligence and tutoring systems: Computational approaches to the communication of knowledge*" Los Altos, CA: Morgan Kaufman; 1987.

[WESC1977] K. T. Wescourt, M. Beard, L. Gould: Knowledge-Based Adaptive Curriculum Sequencing for CAI: Application of a Network Representation. In *Annual Conference, Association for Computing Machinery* ; 1977; New York. pp. 234-240.

[WHIT1998] M. White, T. Caldwell: EXEMPLARS: A Practical Extensible Framework For Dynamic Text Generation. In *Proceedings of the Ninth International Workshop on Natural Language Generation*; 1998. pp. 266–275.

[WOOL1985] B. P. Woolf, D. D. McDonald: *Building a Computer Tutor: Design Issues.* AEDS Monitor. 1985. 23(9-10). pp. 10-18.

[WOOL1992] B. P. Woolf: *AI in Education.* In Shapiro SC, editor. Encyclopedia of Artificial Intelligence. New York: John Wiley & Sons, Inc.; 1992. pp. 434-444.

[WOOL2008] B. P. Woolf: "*Building Intelligent Interactive Tutors: Student-centered strategies for revolutionizing e-learning*": Morgan Kaufmann; 2008.

[ŢITK2001] B. Ţitko: *Pristup projektiranju računalom poduprtih inteligentnih tutorskih sustava.* diplomski rad. Split, Hrvatska: Prirodoslovno-matematički fakultet, Sveučilište u Splitu; 2001.

[ŢITK2005] B. Ţitko: *Model vrednovanja znanja u inteligentnim sustavima e-učenja.* magistarski rad. Zagreb, Hrvatska: Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu; 2005.

[ŢITK2009] B. Ţitko, S. Stankov, M. Rosić, A. Grubišić: *Dynamic Test Generation over Ontology-based Knowledge Representation in Authoring Shell.* Expert Systems with Applications. 2009. 36(4). pp. 8185-8196.

8 Prilozi

8.1 Prilog A - Ontologija "Računalo kao sustav" opisana OWL jezikom

8.1.1 Prilog A.1 - RDF/XML zapis ontologije "Računalo kao sustav"

```
<?xml version="1.0"?>

<rdf:RDF xmlns="http://www.pmfst.hr/racsus.owl#"
     xml:base="http://www.pmfst.hr/racsus.owl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:racunarstvo="http://www.pmfst.hr/racsus.owl#">
  <owl:Ontology rdf:about=""/>

  <!--
    ///////////////////////////////////////
    //
    // Object Properties
    //
    ///////////////////////////////////////
     -->
  <!-- http://www.pmfst.hr/racsus.owl#cita -->
  <owl:ObjectProperty rdf:about="#cita">
    <rdfs:label>čita</rdfs:label>
  </owl:ObjectProperty>

  <!-- http://www.pmfst.hr/racsus.owl#ima -->
  <owl:ObjectProperty rdf:about="#ima">
    <rdfs:label>ima</rdfs:label>
  </owl:ObjectProperty>

  <!-- http://www.pmfst.hr/racsus.owl#izvrsava -->
  <owl:ObjectProperty rdf:about="#izvrsava">
    <rdfs:label>izvršava</rdfs:label>
  </owl:ObjectProperty>

  <!-- http://www.pmfst.hr/racsus.owl#je_dio -->
  <owl:ObjectProperty rdf:about="#je_dio">
    <rdf:type rdf:resource="&owl;TransitiveProperty"/>
    <rdfs:label>je dio</rdfs:label>
    <owl:inverseOf rdf:resource="#se_sastoji_od"/>
  </owl:ObjectProperty>

  <!-- http://www.pmfst.hr/racsus.owl#koristi -->
  <owl:ObjectProperty rdf:about="#koristi">
    <rdfs:label>koristi</rdfs:label>
  </owl:ObjectProperty>

  <!-- http://www.pmfst.hr/racsus.owl#pise -->
  <owl:ObjectProperty rdf:about="#pise">
    <rdfs:label>piše</rdfs:label>
```

```
</owl:ObjectProperty>

<!-- http://www.pmfst.hr/racsus.owl#prevodi -->
<owl:ObjectProperty rdf:about="#prevodi">
  <rdfs:label>prevodi</rdfs:label>
</owl:ObjectProperty>

<!-- http://www.pmfst.hr/racsus.owl#se_sastoji_od -->
<owl:ObjectProperty rdf:about="#se_sastoji_od">
  <rdf:type rdf:resource="&owl;TransitiveProperty"/>
  <rdfs:label>se sastoji od</rdfs:label>
</owl:ObjectProperty>

<!-- http://www.pmfst.hr/racsus.owl#sluzi_za -->
<owl:ObjectProperty rdf:about="#sluzi_za">
  <rdfs:label>služi za</rdfs:label>
</owl:ObjectProperty>

<!-- http://www.pmfst.hr/racsus.owl#upravlja_s -->
<owl:ObjectProperty rdf:about="#upravlja_s">
  <rdfs:label>upravlja s</rdfs:label>
</owl:ObjectProperty>

<!-- http://www.pmfst.hr/racsus.owl#vrsi -->
<owl:ObjectProperty rdf:about="#vrsi">
  <rdfs:label>vrši</rdfs:label>
</owl:ObjectProperty>

<!--
  //////////////////////////////////////////
  //
  // Data properties
  //
  //////////////////////////////////////////
   -->
<!-- http://www.pmfst.hr/racsus.owl#ima_bazu -->
<owl:DatatypeProperty rdf:about="#ima_bazu">
  <rdf:type rdf:resource="&owl;FunctionalProperty"/>
  <rdfs:label>ima bazu</rdfs:label>
  <rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>

<!-- http://www.pmfst.hr/racsus.owl#ima_kapacitet -->
<owl:DatatypeProperty rdf:about="#ima_kapacitet">
  <rdf:type rdf:resource="&owl;FunctionalProperty"/>
  <rdfs:label>ima kapacitet</rdfs:label>
  <rdfs:range rdf:resource="&xsd;integer"/>
</owl:DatatypeProperty>

<!-- http://www.pmfst.hr/racsus.owl#ima_znamenke -->
<owl:DatatypeProperty rdf:about="#ima_znamenke">
  <rdfs:label>ima znamenke</rdfs:label>
  <rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<!--
  //////////////////////////////////////////
  //
  // Classes
  //
  //////////////////////////////////////////
```

```
     -->
  <!-- http://www.pmfst.hr/racsus.owl#Aplikacijska_programska_podrska -->
  <owl:Class rdf:about="#Aplikacijska_programska_podrska">
    <rdfs:label>aplikacijska programska podrška</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Programska_podrska"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Aritmeticka_operacija -->
  <owl:Class rdf:about="#Aritmeticka_operacija">
    <rdfs:label>aritmetička operacija</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Operacija"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Aritmeticko_logicka_jedinica -->
  <owl:Class rdf:about="#Aritmeticko_logicka_jedinica">
    <rdfs:label>aritmetičko logička jedinica</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Restriction>
            <owl:onProperty rdf:resource="#je_dio"/>
            <owl:someValuesFrom
rdf:resource="#Centralna_procesorska_jedinica"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#izvrsava"/>
            <owl:allValuesFrom>
              <owl:Class>
                <owl:unionOf rdf:parseType="Collection">
                  <rdf:Description rdf:about="#Aritmeticka_operacija"/>
                  <rdf:Description rdf:about="#Logicka_operacija"/>
                </owl:unionOf>
              </owl:Class>
            </owl:allValuesFrom>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>

<rdfs:seeAlso>http://www.ee.nmt.edu/~rhb/ee231/labs2000/lab05/lab05_alu.gif
</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Baza -->
  <owl:Class rdf:about="#Baza">
    <rdfs:label>baza</rdfs:label>
    <rdfs:subClassOf rdf:resource="&owl;Thing"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Brojevni_sustav -->
  <owl:Class rdf:about="#Brojevni_sustav">
    <rdfs:label>brojevni sustav</rdfs:label>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Centralna_jedinica -->
  <owl:Class rdf:about="#Centralna_jedinica">
    <rdfs:label>centralna jedinica</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <rdf:Description rdf:about="#Tehnicka_podrska"/>
```

```
          <owl:Restriction>
            <owl:onProperty rdf:resource="#se_sastoji_od"/>
            <owl:someValuesFrom rdf:resource="#Masovna_memorija"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#se_sastoji_od"/>
            <owl:someValuesFrom rdf:resource="#Racunalo"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Centralna_procesorska_jedinica -->
  <owl:Class rdf:about="#Centralna_procesorska_jedinica">
    <rdfs:label>centralna procesorska jedinica</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Restriction>
            <owl:onProperty rdf:resource="#se_sastoji_od"/>
            <owl:someValuesFrom
rdf:resource="#Aritmeticko_logicka_jedinica"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#se_sastoji_od"/>
            <owl:someValuesFrom rdf:resource="#Upravljačka_jedinica"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/e/e7/Intel_8048
6DX2_bottom.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Disketa -->
  <owl:Class rdf:about="#Disketa">
    <rdfs:label>disketa</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <rdf:Description rdf:about="#Masovna_memorija"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#ima_kapacitet"/>
            <owl:hasValue>1.44 MB</owl:hasValue>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/a/aa/Floppy_dis
k_2009_G1.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Informacija -->
  <owl:Class rdf:about="#Informacija">
    <rdfs:label>informacija</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
```

```
        <rdf:Description rdf:about="#Podatak"/>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#ima"/>
          <owl:hasValue rdf:resource="#Znacenje"/>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>

<!-- http://www.pmfst.hr/racsus.owl#Instrukcija -->
<owl:Class rdf:about="#Instrukcija">
  <rdfs:label>instrukcija</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Restriction>
          <owl:onProperty rdf:resource="#se_sastoji_od"/>
          <owl:someValuesFrom rdf:resource="#Operacija"/>
        </owl:Restriction>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#se_sastoji_od"/>
          <owl:allValuesFrom rdf:resource="#Podatak"/>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>

<!-- http://www.pmfst.hr/racsus.owl#Interpretator -->
<owl:Class rdf:about="#Interpretator">
  <rdfs:label>interpretator</rdfs:label> <rdfs:subClassOf
  rdf:resource="#Jezicni_prevoditelj"/>
</owl:Class>

<!-- http://www.pmfst.hr/racsus.owl#Izlazna_jedinica -->
<owl:Class rdf:about="#Izlazna_jedinica">
  <rdfs:label>izlazna jedinica</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <rdf:Description rdf:about="#Tehnicka_podrska"/>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#sluzi_za"/>
          <owl:hasValue rdf:resource="#Prikazivanje_podataka"/>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </rdfs:subClassOf>
</owl:Class>

<!-- http://www.pmfst.hr/racsus.owl#Jezicni_prevoditelj -->
<owl:Class rdf:about="#Jezicni_prevoditelj">
  <rdfs:label>jezični prevoditelj</rdfs:label>
  <rdfs:subClassOf>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <rdf:Description rdf:about="#Sistemska_programska_podrska"/>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#prevodi"/>
```

```
            <owl:someValuesFrom
rdf:resource="#Programski_jezik_visoke_razine"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Kompaktni_disk -->
  <owl:Class rdf:about="#Kompaktni_disk">
    <rdfs:label>kompaktni disk</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Masovna_memorija"/>
    <rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/c/ca/CD-
ROM.png</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Kompilator -->
  <owl:Class rdf:about="#Kompilator">
    <rdfs:label>kompilator</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Jezicni_prevoditelj"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Logicka_operacija -->
  <owl:Class rdf:about="#Logicka_operacija">
    <rdfs:label>Logička operacija</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Operacija"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Logicki_sklop -->
  <owl:Class rdf:about="#Logicki_sklop">
    <rdfs:label>logički sklop</rdfs:label>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#izvrsava"/>
        <owl:allValuesFrom rdf:resource="#Logicka_operacija"/>
      </owl:Restriction>
    </rdfs:subClassOf>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/2/26/7400.jpg</
rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Masovna_memorija -->
  <owl:Class rdf:about="#Masovna_memorija">
    <rdfs:label>masovna memorija</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Memorija"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Memorija -->
  <owl:Class rdf:about="#Memorija">
    <rdfs:label>memorija</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <rdf:Description rdf:about="#Tehnicka_podrska"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#sluzi_za"/>
            <owl:hasValue rdf:resource="#Pohrana_podataka"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
```

```
    </rdfs:subClassOf>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Mis -->
  <owl:Class rdf:about="#Mis">
    <rdfs:label>miš</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Ulazna_jedinica"/>
    <rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/a/aa/3-
Tastenmaus_Microsoft.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Model_racunalnog_sustava -->
  <owl:Class rdf:about="#Model_racunalnog_sustava">
    <rdfs:label>model računalnog sustava</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Restriction>
            <owl:onProperty rdf:resource="#se_sastoji_od"/>
            <owl:someValuesFrom rdf:resource="#Centralna_jedinica"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#se_sastoji_od"/>
            <owl:allValuesFrom rdf:resource="#Izlazna_jedinica"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#se_sastoji_od"/>
            <owl:allValuesFrom rdf:resource="#Ulazna_jedinica"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>

<rdfs:seeAlso>https://computing.llnl.gov/tutorials/parallel_comp/images/von
Neumann1.gif</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Modem -->
  <owl:Class rdf:about="#Modem">
    <rdfs:label>modem</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <rdf:Description rdf:about="#Uredjaj_za_komunikaciju"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#vrsi"/>
            <owl:someValuesFrom
rdf:resource="#Serijski_prijenos_podataka"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/2/26/Motorola_m
odem_28k.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Monitor -->
  <owl:Class rdf:about="#Monitor">
    <rdfs:label>monitor</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Izlazna_jedinica"/>
```

```
<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/7/7e/LG_L194WT-
SF_LCD_monitor.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Mrezna_kartica -->
  <owl:Class rdf:about="#Mrezna_kartica">
    <rdfs:label>mrežna kartica</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Uredjaj_za_komunikaciju"/>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/6/6f/DEC_EtherW
orks_LC_%28DE100%29_FCC_ID_-_A09-DE100.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Operacija -->
  <owl:Class rdf:about="#Operacija">
    <rdfs:label>operacija</rdfs:label>
    <rdfs:subClassOf rdf:resource="&owl;Thing"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Operacijski_sustav -->
  <owl:Class rdf:about="#Operacijski_sustav">
    <rdfs:label>operacijski sustav</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Sistemska_programska_podrska"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Paralelan_prijenos_podataka -->
  <owl:Class rdf:about="#Paralelan_prijenos_podataka">
     <rdfs:label>paralelan prijenos podataka</rdfs:label>
     <rdfs:subClassOf rdf:resource="#Prijenos_podataka"/>

<rdfs:seeAlso>http://static.commentcamarche.net/en.kioskea.net/pictures/pc-
images-parallele.gif</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Podatak -->
  <owl:Class rdf:about="#Podatak">
    <rdfs:label>podatak</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:complementOf>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#ima"/>
            <owl:hasValue rdf:resource="#Znacenje"/>
          </owl:Restriction>
        </owl:complementOf>
      </owl:Class>
    </rdfs:subClassOf>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Prijenos_podataka -->
  <owl:Class rdf:about="#Prijenos_podataka">
    <rdfs:label>prijenos podataka</rdfs:label>
    <rdfs:subClassOf rdf:resource="&owl;Thing"/>
    <rdfs:seeAlso>http://www.biopcrepairs.com/images/data-transfer-
01010101.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Programska_podrska -->
  <owl:Class rdf:about="#Programska_podrska">
    <rdfs:label>programska podrška</rdfs:label>
```

```
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Programski_jezik -->
  <owl:Class rdf:about="#Programski_jezik">
    <rdfs:label>programski jezik</rdfs:label>
    <rdfs:seeAlso>http://enterprisestorageforum.webopedia.com/FIG/PROG-
LAN.gif</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Programski_jezik_niske_razine -->
  <owl:Class rdf:about="#Programski_jezik_niske_razine">
    <rdfs:label>programski jezik niske razine</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Programski_jezik"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Programski_jezik_visoke_razine -->
  <owl:Class rdf:about="#Programski_jezik_visoke_razine">
    <rdfs:label>programski jezik visoke razine</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Programski_jezik"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#RAM -->
  <owl:Class rdf:about="#RAM">
    <rdfs:subClassOf rdf:resource="#Radna_memorija"/>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/c/ca/Memory_mod
ule_DDRAM_20-03-2006.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#ROM -->
  <owl:Class rdf:about="#ROM">
    <rdfs:subClassOf rdf:resource="#Radna_memorija"/>
    <rdfs:seeAlso>http://www.novopc.com/wp-
content/uploads/2008/10/rom.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Racunalni_sustav -->
  <owl:Class rdf:about="#Racunalni_sustav">
    <rdfs:label>računalni sustav</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Restriction>
            <owl:onProperty rdf:resource="#se_sastoji_od"/>
            <owl:someValuesFrom rdf:resource="#Programska_podrska"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#se_sastoji_od"/>
            <owl:someValuesFrom rdf:resource="#Tehnicka_podrska"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#izvrsava"/>
            <owl:allValuesFrom rdf:resource="#Temeljna_funkcija_racunala"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>
    <rdfs:seeAlso>http://www.desccomputers.com/images/computer-
system.gif</rdfs:seeAlso>
  </owl:Class>
```

```
  <!-- http://www.pmfst.hr/racsus.owl#Racunalo -->
  <owl:Class rdf:about="#Racunalo">
    <rdfs:label>računalo</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Restriction>
            <owl:onProperty rdf:resource="#je_dio"/>
            <owl:someValuesFrom rdf:resource="#Centralna_jedinica"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#se_sastoji_od"/>
            <owl:someValuesFrom>
              <owl:Class>
                <owl:intersectionOf rdf:parseType="Collection">
                  <rdf:Description
rdf:about="#Centralna_procesorska_jedinica"/>
                  <rdf:Description rdf:about="#Radna_memorija"/>
                </owl:intersectionOf>
              </owl:Class>
            </owl:someValuesFrom>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>

<rdfs:seeAlso>http://www.escotal.com/Images/computer/motherboard.jpg</rdfs:
seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Radna_memorija -->
  <owl:Class rdf:about="#Radna_memorija">
    <rdfs:label>radna memorija</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Memorija"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Serijski_prijenos_podataka -->
  <owl:Class rdf:about="#Serijski_prijenos_podataka">
    <rdfs:label>serijski prijenos podataka</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Prijenos_podataka"/>

<rdfs:seeAlso>http://static.commentcamarche.net/en.kioskea.net/pictures/pc-
images-serie2.gif</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Sistemska_programska_podrska -->
  <owl:Class rdf:about="#Sistemska_programska_podrska">
     <rdfs:label>sistemska programska podrška</rdfs:label>
     <rdfs:subClassOf rdf:resource="#Programska_podrska"/>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Stampac -->
  <owl:Class rdf:about="#Stampac">
    <rdfs:label>štampač</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Izlazna_jedinica"/>
    <rdfs:seeAlso>http://images.tigerdirect.com/skuimages/large/HP-P1005-
Printer1.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Strojni_jezik -->
  <owl:Class rdf:about="#Strojni_jezik">
```

```
    <rdfs:label>strojni jezik</rdfs:label>
    <rdfs:subClassOf rdf:resource="&owl;Thing"/>
    <rdfs:seeAlso>http://www.retas.de/thomas/computer/machine-
code.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Tehnicka_podrska -->
  <owl:Class rdf:about="#
cka_podrska">
    <rdfs:label>tehnička podrška</rdfs:label>

<rdfs:seeAlso>http://blaztcomputo.com.mx/images/hardware3.JPG</rdfs:seeAlso
>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Temeljna_funkcija_racunala -->
  <owl:Class rdf:about="#Temeljna_funkcija_racunala">
    <rdfs:label>temeljna funkcija računala</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:oneOf rdf:parseType="Collection">   <rdf:Description
          rdf:about="#Prikazivanje_podataka"/> <rdf:Description
          rdf:about="#Unos_podataka"/> <rdf:Description
          rdf:about="#Obrada_podataka"/>
        </owl:oneOf>
      </owl:Class>
    </rdfs:subClassOf>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Tipkovnica -->
  <owl:Class rdf:about="#Tipkovnica">
    <rdfs:label>tipkovnica</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Ulazna_jedinica"/>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/5/57/ModelM.jpg
</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Tvrdi_disk -->
  <owl:Class rdf:about="#Tvrdi_disk">
    <rdfs:label>tvrdi disk</rdfs:label>  <rdfs:subClassOf
    rdf:resource="#Masovna_memorija"/>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/5/5e/Apertura_h
ard_disk_05.jpg</rdfs:seeAlso>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Ulazna_jedinica -->
  <owl:Class rdf:about="#Ulazna_jedinica">
    <rdfs:label>ulazna jedinica</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <rdf:Description rdf:about="#Tehnicka_podrska"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#sluzi_za"/>
            <owl:hasValue rdf:resource="#Unos_podataka"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>
```

```
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Upravljačka_jedinica -->
  <owl:Class rdf:about="#Upravljačka_jedinica">
    <rdfs:label>upravljačka jedinica</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <owl:Restriction>
            <owl:onProperty rdf:resource="#je_dio"/>
            <owl:someValuesFrom
rdf:resource="#Centralna_procesorska_jedinica"/>
          </owl:Restriction>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#upravlja_s"/>
            <owl:allValuesFrom rdf:resource="#Instrukcija"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Uredjaj_za_komunikaciju -->
  <owl:Class rdf:about="#Uredjaj_za_komunikaciju">
    <rdfs:label>ure&#273;aj za komunikaciju</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class>
        <owl:intersectionOf rdf:parseType="Collection">
          <rdf:Description rdf:about="#Tehnicka_podrska"/>
          <owl:Restriction>
            <owl:onProperty rdf:resource="#sluzi_za"/>
            <owl:someValuesFrom rdf:resource="#Prijenos_podataka"/>
          </owl:Restriction>
        </owl:intersectionOf>
      </owl:Class>
    </rdfs:subClassOf>
  </owl:Class>

  <!-- http://www.pmfst.hr/racsus.owl#Usluzni_program -->
  <owl:Class rdf:about="#Usluzni_program">
    <rdfs:label>uslužni program</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Sistemska_programska_podrska"/>
  </owl:Class>

  <!-- http://www.w3.org/2002/07/owl#Thing -->
  <owl:Class rdf:about="&owl;Thing"/>

  <!--
    ///////////////////////////////////////////////////
    //
    // Individuals
    //
    ///////////////////////////////////////////////////
     -->
  <!-- http://www.pmfst.hr/racsus.owl#Asembler -->
  <owl:Thing rdf:about="#Asembler">
    <rdf:type rdf:resource="#Programski_jezik_niske_razine"/>
    <rdfs:seeAlso>http://www.linuxsoft.cz/screenshot_img/5086-
a.jpg</rdfs:seeAlso>
  </owl:Thing>
```

```
  <!-- http://www.pmfst.hr/racsus.owl#Basic -->
  <Programski_jezik_visoke_razine rdf:about="#Basic">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>Basic</rdfs:label>

<rdfs:seeAlso>http://www.petesqbsite.com/sections/introduction/qb45.png</rd
fs:seeAlso>
  </Programski_jezik_visoke_razine>

  <!-- http://www.pmfst.hr/racsus.owl#Binarni_brojevni_sustav -->
  <Brojevni_sustav rdf:about="#Binarni_brojevni_sustav">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>binarni brojevni sustav</rdfs:label>
    <ima_znamenke>0 1</ima_znamenke>
    <ima_bazu>2</ima_bazu>
  </Brojevni_sustav>

  <!-- http://www.pmfst.hr/racsus.owl#C -->
  <owl:Thing rdf:about="#C">
    <rdf:type rdf:resource="#Programski_jezik_visoke_razine"/>
    <rdfs:label>C</rdfs:label>

<rdfs:seeAlso>http://www.scielo.br/img/fbpe/bjmbr/v30n8/2750tab1.gif</rdfs:
seeAlso>
  </owl:Thing>

  <!-- http://www.pmfst.hr/racsus.owl#DOS -->
  <Operacijski_sustav rdf:about="#DOS">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:seeAlso>http://www.subsowespac.org/sh_xp/mem-
check.jpg</rdfs:seeAlso>
  </Operacijski_sustav>

  <!-- http://www.pmfst.hr/racsus.owl#Dekadski_brojevni_sustav -->
  <Brojevni_sustav rdf:about="#Dekadski_brojevni_sustav">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>dekadski brojevni sustav</rdfs:label>
    <ima_znamenke>0 1 2 3 4 5 6 7 8 9</ima_znamenke>
    <ima_bazu>2</ima_bazu>
  </Brojevni_sustav>

  <!-- http://www.pmfst.hr/racsus.owl#Disjunkcija -->
  <Logicka_operacija rdf:about="#Disjunkcija">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>disjunkcija</rdfs:label>

<rdfs:seeAlso>http://content.tutorvista.com/maths/content/us/class12maths/c
hapter02/images/img57.gif</rdfs:seeAlso>
  </Logicka_operacija>

  <!-- http://www.pmfst.hr/racsus.owl#Fortran -->
  <Programski_jezik_visoke_razine rdf:about="#Fortran">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>Fortran</rdfs:label>
    <rdfs:seeAlso>http://pcwin.com/media/images/screen/59098-
fortran_calculus_compiler.gif</rdfs:seeAlso>
  </Programski_jezik_visoke_razine>

  <!-- http://www.pmfst.hr/racsus.owl#Heksadecimalni_brojevni_sustav -->
  <Brojevni_sustav rdf:about="#Heksadecimalni_brojevni_sustav">
    <rdf:type rdf:resource="&owl;Thing"/>
```

```
    <rdfs:label>heksadecimalni brojevni sustav</rdfs:label>
    <ima_znamenke>0 1 2 3 4 5 6 7 8 9 A B C D E F</ima_znamenke>
    <ima_bazu>16</ima_bazu>
  </Brojevni_sustav>

  <!-- http://www.pmfst.hr/racsus.owl#Konjunkcija -->
  <Logicka_operacija rdf:about="#Konjunkcija">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>konjunkcija</rdfs:label>

<rdfs:seeAlso>http://content.tutorvista.com/maths/content/us/class12maths/c
hapter02/images/img52.gif</rdfs:seeAlso>
  </Logicka_operacija>

  <!-- http://www.pmfst.hr/racsus.owl#Logicki_I -->
  <Logicki_sklop rdf:about="#Logicki_I">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>I sklop</rdfs:label>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/6/64/AND_ANSI.s
vg</rdfs:seeAlso>
    <izvrsava rdf:resource="#Konjunkcija"/>
  </Logicki_sklop>

  <!-- http://www.pmfst.hr/racsus.owl#Logicki_ILI_sklop -->
  <Logicki_sklop rdf:about="#Logicki_ILI_sklop">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>ILI sklop</rdfs:label>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/b/b5/OR_ANSI.sv
g</rdfs:seeAlso>
    <izvrsava rdf:resource="#Disjunkcija"/>
  </Logicki_sklop>

  <!-- http://www.pmfst.hr/racsus.owl#Logicki_NE_sklop -->
  <Logicki_sklop rdf:about="#Logicki_NE_sklop">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>NE sklop</rdfs:label>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/b/bc/NOT_ANSI.s
vg</rdfs:seeAlso>
    <izvrsava rdf:resource="#Negacija"/>
  </Logicki_sklop>

  <!-- http://www.pmfst.hr/racsus.owl#Logicki_NILI_sklop -->
  <owl:Thing rdf:about="#Logicki_NILI_sklop">
    <rdf:type rdf:resource="#Logicki_sklop"/>
    <rdfs:label>NILI sklop</rdfs:label>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/6/6c/NOR_ANSI.s
vg</rdfs:seeAlso>
    <izvrsava rdf:resource="#Disjunkcija"/>
    <izvrsava rdf:resource="#Negacija"/>
  </owl:Thing>

  <!-- http://www.pmfst.hr/racsus.owl#Logicki_NI_sklop -->
  <owl:Thing rdf:about="#Logicki_NI_sklop">
    <rdf:type rdf:resource="#Logicki_sklop"/>
    <rdfs:label>NI sklop</rdfs:label>
```

```
<rdfs:seeAlso>http://en.wikipedia.org/wiki/File:NAND_ANSI.svg</rdfs:seeAlso
>
    <izvrsava rdf:resource="#Konjunkcija"/>
    <izvrsava rdf:resource="#Negacija"/>
  </owl:Thing>

  <!-- http://www.pmfst.hr/racsus.owl#Negacija -->
  <Logicka_operacija rdf:about="#Negacija">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>negacija</rdfs:label>

<rdfs:seeAlso>http://content.tutorvista.com/maths/content/us/class12maths/c
hapter02/images/img59.gif</rdfs:seeAlso>
  </Logicka_operacija>

  <!-- http://www.pmfst.hr/racsus.owl#Obrada_podataka -->
  <Temeljna_funkcija_racunala rdf:about="#Obrada_podataka">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>obrada podataka</rdfs:label>
    <se_sastoji_od rdf:resource="#Pohrana_podataka"/>
    <se_sastoji_od rdf:resource="#Upravljanje_podataka"/>
  </Temeljna_funkcija_racunala>

  <!-- http://www.pmfst.hr/racsus.owl#Oduzimanje -->
  <Aritmeticka_operacija rdf:about="#Oduzimanje">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>oduzimanje</rdfs:label>
  </Aritmeticka_operacija>

  <!-- http://www.pmfst.hr/racsus.owl#Oktalni_brojevni_sustav -->
  <Brojevni_sustav rdf:about="#Oktalni_brojevni_sustav">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>oktalni brojevni sustav</rdfs:label>
    <ima_znamenke>0 1 2 3 4 5 6 7</ima_znamenke>
    <ima_bazu>8</ima_bazu>
  </Brojevni_sustav>

  <!-- http://www.pmfst.hr/racsus.owl#Pascal -->
  <Programski_jezik_visoke_razine rdf:about="#Pascal">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>Pascal</rdfs:label>

<rdfs:seeAlso>http://downloadsource.net/img/c1492008ff568ed23d384727cab4aa0
1.jpg</rdfs:seeAlso>
  </Programski_jezik_visoke_razine>

  <!-- http://www.pmfst.hr/racsus.owl#Pohrana_podataka -->
  <owl:Thing rdf:about="#Pohrana_podataka">
    <rdfs:label>pohrana podataka</rdfs:label>
  </owl:Thing>

  <!-- http://www.pmfst.hr/racsus.owl#Prikazivanje_podataka -->
  <owl:Thing rdf:about="#Prikazivanje_podataka">
    <rdf:type rdf:resource="#Temeljna_funkcija_racunala"/>
    <rdfs:label>prikazivanje podataka</rdfs:label>
  </owl:Thing>

  <!-- http://www.pmfst.hr/racsus.owl#Unix -->
  <Operacijski_sustav rdf:about="#Unix">
    <rdf:type rdf:resource="&owl;Thing"/>
```

```
    <rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/commons/d/d4/X-
Window-System.png</rdfs:seeAlso>
  </Operacijski_sustav>

  <!-- http://www.pmfst.hr/racsus.owl#Unos_podataka -->
  <Temeljna_funkcija_racunala rdf:about="#Unos_podataka">
    <rdf:type rdf:resource="&owl;Thing"/>
    <rdfs:label>unos podataka</rdfs:label>
  </Temeljna_funkcija_racunala>

  <!-- http://www.pmfst.hr/racsus.owl#Upravljanje_podataka -->
  <owl:Thing rdf:about="#Upravljanje_podataka">
    <rdfs:label>upravljanje podataka</rdfs:label>
  </owl:Thing>

  <!-- http://www.pmfst.hr/racsus.owl#Windows -->
  <Operacijski_sustav rdf:about="#Windows">
    <rdf:type rdf:resource="&owl;Thing"/>

<rdfs:seeAlso>http://upload.wikimedia.org/wikipedia/en/b/bd/Windows_7.png</
rdfs:seeAlso>
  </Operacijski_sustav>

  <!-- http://www.pmfst.hr/racsus.owl#Zbrajanje -->
  <owl:Thing rdf:about="#Zbrajanje">
    <rdf:type rdf:resource="#Aritmeticka_operacija"/>
    <rdfs:label>zbrajanje</rdfs:label>
  </owl:Thing>

  <!-- http://www.pmfst.hr/racsus.owl#Znacenje -->
  <owl:Thing rdf:about="#Znacenje">
    <rdfs:label>značenje</rdfs:label>
  </owl:Thing>
</rdf:RDF>

<!-- Generated by the OWL API (version 2.2.1.1138)
http://owlapi.sourceforge.net -->
```

8.1.2 Prilog A.2 - OWL apstraktna sintaksa ontologije "Računalo kao sustav"

```
Ontology(
http://www.pmfst.hr/racsus.owl
Individual(racsus:Logicki_NE_sklop annotation(rdfs:label, "NE sklop")
type(racsus:Logicki_sklop) type(owl:Thing) value(racsus:izvrsava
racsus:Negacija))
ObjectProperty(racsus:ima annotation(rdfs:label, "ima"))
Class(racsus:Aritmeticko_logicka_jedinica Partial annotation(rdfs:label,
"aritmetičko logička jedinica") intersectionOf(restriction(racsus:je_dio
someValuesFrom(racsus:Centralna_procesorska_jedinica))
restriction(racsus:izvrsava
allValuesFrom(unionOf(racsus:Aritmeticka_operacija
racsus:Logicka_operacija)))))
Class(racsus:ROM Partial racsus:Radna_memorija)
Class(racsus:Centralna_procesorska_jedinica Partial annotation(rdfs:label,
"centralna procesorska jedinica")
intersectionOf(restriction(racsus:se_sastoji_od
someValuesFrom(racsus:Aritmeticko_logicka_jedinica))
restriction(racsus:se_sastoji_od
someValuesFrom(racsus:Upravljačka_jedinica))))
Individual(racsus:Negacija annotation(rdfs:label, "negacija")
type(racsus:Logicka_operacija) type(owl:Thing))
Class(racsus:Podatak Partial annotation(rdfs:label, "podatak")
complementOf(restriction(racsus:ima value(racsus:Znacenje))))
Class(racsus:Mis Partial annotation(rdfs:label, "miš")
racsus:Ulazna_jedinica)
Individual(racsus:Disjunkcija annotation(rdfs:label, "disjunkcija")
type(racsus:Logicka_operacija) type(owl:Thing))
Individual(racsus:Unix type(racsus:Operacijski_sustav) type(owl:Thing))
Individual(racsus:Logicki_NI_sklop annotation(rdfs:label, "NI sklop")
type(owl:Thing) type(racsus:Logicki_sklop) value(racsus:izvrsava
racsus:Negacija) value(racsus:izvrsava racsus:Konjunkcija))
Individual(racsus:DOS type(racsus:Operacijski_sustav) type(owl:Thing))
Class(racsus:Jezicni_prevoditelj Partial annotation(rdfs:label, "jezični
prevoditelj") intersectionOf(racsus:Sistemska_programska_podrska
restriction(racsus:prevodi
someValuesFrom(racsus:Programski_jezik_visoke_razine))))
Class(racsus:Centralna_jedinica Partial annotation(rdfs:label, "centralna
jedinica") intersectionOf(racsus:Tehnicka_podrska
restriction(racsus:se_sastoji_od someValuesFrom(racsus:Masovna_memorija))
restriction(racsus:se_sastoji_od someValuesFrom(racsus:Racunalo))))
Individual(racsus:Negacija annotation(rdfs:label, "negacija")
type(racsus:Logicka_operacija) type(owl:Thing))
Class(racsus:Programski_jezik_visoke_razine Partial annotation(rdfs:label,
"programski jezik visoke razine") racsus:Programski_jezik)
Class(racsus:Usluzni_program Partial annotation(rdfs:label, "uslužni
program") racsus:Sistemska_programska_podrska)
Class(racsus:Disketa Partial annotation(rdfs:label, "disketa")
intersectionOf(racsus:Masovna_memorija restriction(racsus:ima_kapacitet
value("1.44 MB""))))
ObjectProperty(racsus:koristi annotation(rdfs:label, "koristi"))
Class(racsus:Operacijski_sustav Partial annotation(rdfs:label, "operacijski
sustav") racsus:Sistemska_programska_podrska)
Class(racsus:Programski_jezik Partial annotation(rdfs:label, "programski
jezik"))
Class(racsus:Tvrdi_disk Partial annotation(rdfs:label, "tvrdi disk")
racsus:Masovna_memorija)
Individual(racsus:Basic annotation(rdfs:label, "Basic")
type(racsus:Programski_jezik_visoke_razine) type(owl:Thing))
```

```
Individual(racsus:C annotation(rdfs:label, "C") type(owl:Thing)
type(racsus:Programski_jezik_visoke_razine))
Individual(racsus:Heksadecimalni_brojevni_sustav annotation(rdfs:label,
"heksadecimalni brojevni sustav") type(racsus:Brojevni_sustav)
type(owl:Thing) value(racsus:ima_bazu "16"") value(racsus:ima_znamenke "0 1
2 3 4 5 6 7 8 9 A B C D E F""))
ObjectProperty(racsus:cita annotation(rdfs:label, "čita"))
Individual(racsus:C annotation(rdfs:label, "C") type(owl:Thing)
type(racsus:Programski_jezik_visoke_razine))
Individual(racsus:Konjunkcija annotation(rdfs:label, "konjunkcija")
type(racsus:Logicka_operacija) type(owl:Thing))
Individual(racsus:Disjunkcija annotation(rdfs:label, "disjunkcija")
type(racsus:Logicka_operacija) type(owl:Thing))
Individual(racsus:Logicki_NE_sklop annotation(rdfs:label, "NE sklop")
type(racsus:Logicki_sklop) type(owl:Thing) value(racsus:izvrsava
racsus:Negacija))
Class(racsus:Racunalo Partial annotation(rdfs:label, "računalo")
intersectionOf(restriction(racsus:je_dio
someValuesFrom(racsus:Centralna_jedinica)) restriction(racsus:se_sastoji_od
someValuesFrom(intersectionOf(racsus:Centralna_procesorska_jedinica
racsus:Radna_memorija)))))
Individual(racsus:Pascal annotation(rdfs:label, "Pascal")
type(racsus:Programski_jezik_visoke_razine) type(owl:Thing))
DatatypeProperty(racsus:ima_znamenke annotation(rdfs:label, "ima znamenke")
range(xmls:string))
Class(racsus:Operacija Partial annotation(rdfs:label, "operacija")
owl:Thing)
Class(racsus:Masovna_memorija Partial annotation(rdfs:label, "masovna
memorija") racsus:Memorija)
Individual(racsus:DOS type(racsus:Operacijski_sustav) type(owl:Thing))
Class(racsus:Sistemska_programska_podrska Partial annotation(rdfs:label,
"sistemska programska podrška") racsus:Programska_podrska)
Individual(racsus:Windows type(racsus:Operacijski_sustav) type(owl:Thing))
ObjectProperty(racsus:vrsi annotation(rdfs:label, "vrši"))
Individual(racsus:Prikazivanje_podataka annotation(rdfs:label, "prikazivanje
podataka") type(owl:Thing) type(racsus:Temeljna_funkcija_racunala))
Class(racsus:Aplikacijska_programska_podrska Partial annotation(rdfs:label,
"aplikacijska programska podrška") racsus:Programska_podrska)
Individual(racsus:Binarni_brojevni_sustav annotation(rdfs:label, "binarni
brojevni sustav") type(racsus:Brojevni_sustav) type(owl:Thing)
value(racsus:ima_bazu "2"") value(racsus:ima_znamenke "0 1""))
Individual(racsus:Unos_podataka annotation(rdfs:label, "unos podataka")
type(racsus:Temeljna_funkcija_racunala) type(owl:Thing))
Individual(racsus:Logicki_ILI_sklop annotation(rdfs:label, "ILI sklop")
type(racsus:Logicki_sklop) type(owl:Thing) value(racsus:izvrsava
racsus:Disjunkcija))
Individual(racsus:Binarni_brojevni_sustav annotation(rdfs:label, "binarni
brojevni sustav") type(racsus:Brojevni_sustav) type(owl:Thing)
value(racsus:ima_bazu "2"") value(racsus:ima_znamenke "0 1""))
DatatypeProperty(racsus:ima_bazu annotation(rdfs:label, "ima bazu")
Functional range(xmls:integer))
Class(racsus:Model_racunalnog_sustava Partial annotation(rdfs:label, "model
računalnog sustava") intersectionOf(restriction(racsus:se_sastoji_od
someValuesFrom(racsus:Centralna_jedinica)) restriction(racsus:se_sastoji_od
allValuesFrom(racsus:Izlazna_jedinica)) restriction(racsus:se_sastoji_od
allValuesFrom(racsus:Ulazna_jedinica))))
Class(racsus:Temeljna_funkcija_racunala Partial annotation(rdfs:label,
"temeljna funkcija računala") oneOf(racsus:Prikazivanje_podataka
racsus:Unos_podataka racsus:Obrada_podataka))
ObjectProperty(racsus:pise annotation(rdfs:label, "piše"))
```

```
Individual(racsus:Logicki_NI_sklop annotation(rdfs:label, "NI sklop")
type(owl:Thing) type(racsus:Logicki_sklop) value(racsus:izvrsava
racsus:Negacija) value(racsus:izvrsava racsus:Konjunkcija))
Individual(racsus:Asembler type(owl:Thing)
type(racsus:Programski_jezik_niske_razine))
ObjectProperty(racsus:sluzi_za annotation(rdfs:label, "služi za"))
Class(racsus:Logicki_sklop Partial annotation(rdfs:label, "logički sklop")
restriction(racsus:izvrsava allValuesFrom(racsus:Logicka_operacija)))
Individual(racsus:Dekadski_brojevni_sustav annotation(rdfs:label, "dekadski
brojevni sustav") type(racsus:Brojevni_sustav) type(owl:Thing)
value(racsus:ima_bazu "2"") value(racsus:ima_znamenke "0 1 2 3 4 5 6 7 8
9""))
Individual(racsus:Prikazivanje_podataka annotation(rdfs:label,
"prikazivanje podataka") type(owl:Thing)
type(racsus:Temeljna_funkcija_racunala))
Individual(racsus:Oduzimanje annotation(rdfs:label, "oduzimanje")
type(racsus:Aritmeticka_operacija) type(owl:Thing))
Class(racsus:Memorija Partial annotation(rdfs:label, "memorija")
intersectionOf(racsus:Tehnicka_podrska restriction(racsus:sluzi_za
value(racsus:Pohrana_podataka))))
Class(racsus:Baza Partial annotation(rdfs:label, "baza") owl:Thing)
Individual(racsus:Fortran annotation(rdfs:label, "Fortran")
type(racsus:Programski_jezik_visoke_razine) type(owl:Thing))
Class(racsus:Tipkovnica Partial annotation(rdfs:label, "tipkovnica")
racsus:Ulazna_jedinica)
Class(racsus:Interpretator Partial annotation(rdfs:label, "interpretator")
racsus:Jezicni_prevoditelj)
Individual(racsus:Oduzimanje annotation(rdfs:label, "oduzimanje")
type(racsus:Aritmeticka_operacija) type(owl:Thing))
Class(racsus:Programski_jezik_niske_razine Partial annotation(rdfs:label,
"programski jezik niske razine") racsus:Programski_jezik)
ObjectProperty(racsus:se_sastoji_od annotation(rdfs:label, "se sastoji od")
Transitive)
Individual(racsus:Upravljanje_podataka annotation(rdfs:label, "upravljanje
podataka") type(owl:Thing))
Class(racsus:Kompilator Partial annotation(rdfs:label, "kompilator")
racsus:Jezicni_prevoditelj)
Class(racsus:Stampac Partial annotation(rdfs:label, "štampač")
racsus:Izlazna_jedinica)
Class(racsus:Aritmeticka_operacija Partial annotation(rdfs:label,
"Aritmetička operacija") racsus:Operacija)
Class(racsus:Upravljačka_jedinica Partial annotation(rdfs:label,
"upravljačka jedinica") intersectionOf(restriction(racsus:je_dio
someValuesFrom(racsus:Centralna_procesorska_jedinica))
restriction(racsus:upravlja_s allValuesFrom(racsus:Instrukcija))))
Class(racsus:Modem Partial annotation(rdfs:label, "modem")
intersectionOf(racsus:Uredjaj_za_komunikaciju restriction(racsus:vrsi
someValuesFrom(racsus:Serijski_prijenos_podataka))))
Class(racsus:Monitor Partial annotation(rdfs:label, "monitor")
racsus:Izlazna_jedinica)
Individual(racsus:Fortran annotation(rdfs:label, "Fortran")
type(racsus:Programski_jezik_visoke_razine) type(owl:Thing))
Individual(racsus:Oktalni_brojevni_sustav annotation(rdfs:label, "oktalni
brojevni sustav") type(racsus:Brojevni_sustav) type(owl:Thing)
value(racsus:ima_bazu "8"") value(racsus:ima_znamenke "0 1 2 3 4 5 6 7""))
ObjectProperty(racsus:upravlja_s annotation(rdfs:label, "upravlja s"))
Individual(racsus:Zbrajanje annotation(rdfs:label, "zbrajanje")
type(owl:Thing) type(racsus:Aritmeticka_operacija))
Individual(racsus:Znacenje annotation(rdfs:label, "značenje")
type(owl:Thing))
```

```
Class(racsus:Instrukcija Partial annotation(rdfs:label, "instrukcija")
intersectionOf(restriction(racsus:se_sastoji_od
someValuesFrom(racsus:Operacija)) restriction(racsus:se_sastoji_od
allValuesFrom(racsus:Podatak))))
Individual(racsus:Logicki_I annotation(rdfs:label, "I sklop")
type(racsus:Logicki_sklop) type(owl:Thing) value(racsus:izvrsava
racsus:Konjunkcija))
Class(racsus:Mrezna_kartica Partial annotation(rdfs:label, "mrežna kartica")
racsus:Uredjaj_za_komunikaciju) Class(racsus:Uredjaj_za_komunikaciju Partial
annotation(rdfs:label, "uređaj za komunikaciju")
intersectionOf(racsus:Tehnicka_podrska restriction(racsus:sluzi_za
someValuesFrom(racsus:Prijenos_podataka)))) Class(racsus:Racunalni_sustav
Partial annotation(rdfs:label, "računalni sustav")
intersectionOf(restriction(racsus:se_sastoji_od
someValuesFrom(racsus:Programska_podrska)) restriction(racsus:se_sastoji_od
someValuesFrom(racsus:Tehnicka_podrska)) restriction(racsus:izvrsava
allValuesFrom(racsus:Temeljna_funkcija_racunala))))
Individual(racsus:Windows type(racsus:Operacijski_sustav) type(owl:Thing))
Class(racsus:Informacija Partial annotation(rdfs:label, "informacija")
intersectionOf(racsus:Podatak restriction(racsus:ima
value(racsus:Znacenje))))
Individual(racsus:Obrada_podataka annotation(rdfs:label, "obrada podataka")
type(racsus:Temeljna_funkcija_racunala) type(owl:Thing)
value(racsus:se_sastoji_od racsus:Upravljanje_podataka)
value(racsus:se_sastoji_od racsus:Pohrana_podataka))
Individual(racsus:Unix type(racsus:Operacijski_sustav) type(owl:Thing))
Class(racsus:Tehnicka_podrska Partial annotation(rdfs:label, "tehnička
podrška"))
Individual(racsus:Zbrajanje annotation(rdfs:label, "zbrajanje")
type(owl:Thing) type(racsus:Aritmeticka_operacija))
Class(racsus:Kompaktni_disk Partial annotation(rdfs:label, "kompaktni
disk") racsus:Masovna_memorija)
Class(racsus:Logicka_operacija Partial annotation(rdfs:label, "Logička
operacija") racsus:Operacija)
Individual(racsus:Logicki_ILI_sklop annotation(rdfs:label, "ILI sklop")
type(racsus:Logicki_sklop) type(owl:Thing) value(racsus:izvrsava
racsus:Disjunkcija))
Individual(racsus:Unos_podataka annotation(rdfs:label, "unos podataka")
type(racsus:Temeljna_funkcija_racunala) type(owl:Thing))
Class(racsus:Radna_memorija Partial annotation(rdfs:label, "radna
memorija") racsus:Memorija)
Class(racsus:Strojni_jezik Partial annotation(rdfs:label, "strojni jezik")
owl:Thing)
Class(racsus:Programska_podrska Partial annotation(rdfs:label, "programska
podrška"))
DatatypeProperty(racsus:ima_kapacitet annotation(rdfs:label, "ima
kapacitet") Functional range(xmls:integer))  ObjectProperty(racsus:prevodi
annotation(rdfs:label, "prevodi")) Individual(racsus:Logicki_NILI_sklop
annotation(rdfs:label, "NILI sklop") type(owl:Thing)
type(racsus:Logicki_sklop) value(racsus:izvrsava racsus:Negacija)
value(racsus:izvrsava racsus:Disjunkcija)) Individual(racsus:Konjunkcija
annotation(rdfs:label, "konjunkcija") type(racsus:Logicka_operacija)
type(owl:Thing)) Class(racsus:Paralelan_prijenos_podataka Partial
annotation(rdfs:label, "paralelan prijenos podataka")
racsus:Prijenos_podataka) Individual(racsus:Logicki_I
annotation(rdfs:label, "I sklop") type(racsus:Logicki_sklop)
type(owl:Thing) value(racsus:izvrsava racsus:Konjunkcija))
Individual(racsus:Heksadecimalni_brojevni_sustav annotation(rdfs:label,
"heksadecimalni brojevni sustav") type(racsus:Brojevni_sustav)
```

```
type(owl:Thing) value(racsus:ima_bazu "16"") value(racsus:ima_znamenke "0 1
2 3 4 5 6 7 8 9 A B C D E F""))
Class(racsus:Ulazna_jedinica Partial annotation(rdfs:label, "ulazna
jedinica") intersectionOf(racsus:Tehnicka_podrska
restriction(racsus:sluzi_za value(racsus:Unos_podataka))))
Class(owl:Thing Partial)
Individual(racsus:Asembler type(owl:Thing)
type(racsus:Programski_jezik_niske_razine))
Individual(racsus:Logicki_NILI_sklop annotation(rdfs:label, "NILI sklop")
type(owl:Thing) type(racsus:Logicki_sklop) value(racsus:izvrsava
racsus:Negacija) value(racsus:izvrsava racsus:Disjunkcija))
Class(racsus:Izlazna_jedinica Partial annotation(rdfs:label, "izlazna
jedinica") intersectionOf(racsus:Tehnicka_podrska
restriction(racsus:sluzi_za value(racsus:Prikazivanje_podataka))))
Individual(racsus:Pascal annotation(rdfs:label, "Pascal")
type(racsus:Programski_jezik_visoke_razine) type(owl:Thing))
Class(racsus:Brojevni_sustav Partial annotation(rdfs:label, "brojevni
sustav"))
Individual(racsus:Dekadski_brojevni_sustav annotation(rdfs:label, "dekadski
brojevni sustav") type(racsus:Brojevni_sustav) type(owl:Thing)
value(racsus:ima_bazu "2"") value(racsus:ima_znamenke "0 1 2 3 4 5 6 7 8
9""))
Individual(racsus:Pohrana_podataka annotation(rdfs:label, "pohrana
podataka") type(owl:Thing))
Class(racsus:Prijenos_podataka Partial annotation(rdfs:label, "prijenos
podataka") owl:Thing)
Individual(racsus:Oktalni_brojevni_sustav annotation(rdfs:label, "oktalni
brojevni sustav") type(racsus:Brojevni_sustav) type(owl:Thing)
value(racsus:ima_bazu "8"") value(racsus:ima_znamenke "0 1 2 3 4 5 6 7""))
ObjectProperty(racsus:izvrsava annotation(rdfs:label, "izvršava"))
Individual(racsus:Obrada_podataka annotation(rdfs:label, "obrada podataka")
type(racsus:Temeljna_funkcija_racunala) type(owl:Thing)
value(racsus:se_sastoji_od racsus:Upravljanje_podataka)
value(racsus:se_sastoji_od racsus:Pohrana_podataka))
Class(racsus:Serijski_prijenos_podataka Partial annotation(rdfs:label,
"serijski prijenos podataka") racsus:Prijenos_podataka)
ObjectProperty(racsus:je_dio annotation(rdfs:label, "je dio")
inverseOf(racsus:se_sastoji_od) Transitive)
Class(racsus:RAM Partial racsus:Radna_memorija)
Individual(racsus:Basic annotation(rdfs:label, "Basic")
type(racsus:Programski_jezik_visoke_razine) type(owl:Thing))
)
```

8.2 Prilog B - XML zapis komunikacije s HML posluţiteljem

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<req>
  <ask>
    <que lem="računalan" tok="" msd="" />
  </ask>
  <ask>
    <que lem="sustav" tok="" msd="" />
  </ask>
</req>

<?xml version="1.0" encoding="UTF-8"?>
<res>
  <tel>
    <ans lem="računalan" tok="računalni" msd="Afpmpn-"/>
    <ans lem="računalan" tok="računalni" msd="Afpmpvy"/>
    <ans lem="računalan" tok="računalni" msd="Afpmsay-n"/>
    <ans lem="računalan" tok="računalni" msd="Afpmsny"/>
    <ans lem="računalan" tok="računalni" msd="Afpmsvy"/>
  </tel>
  <tel>
    <ans lem="sustati" tok="sustav" msd="Vmps"/>
    <ans lem="sustav" tok="sustav" msd="Ncmsa--n"/>
    <ans lem="sustav" tok="sustav" msd="Ncmsn"/>
  </tel>
</res>
```

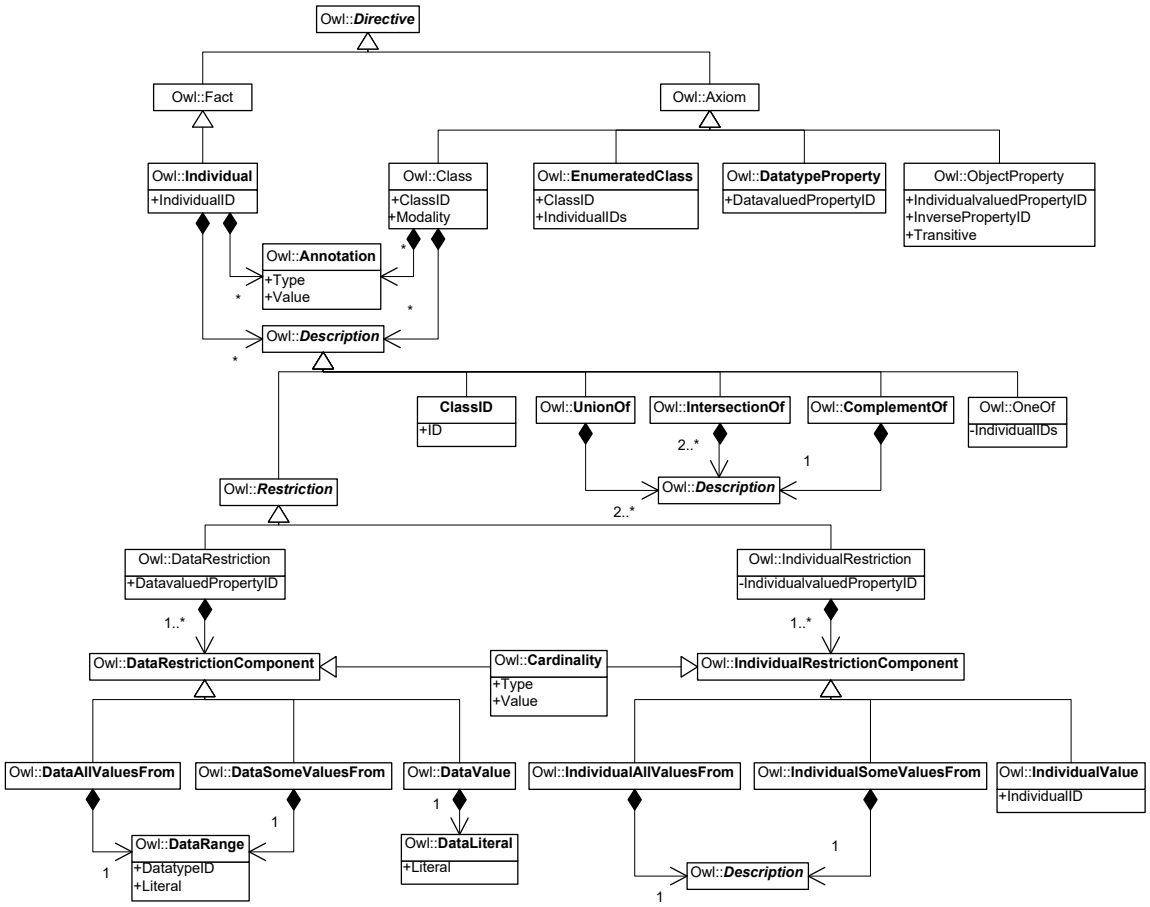
8.3 Prilog C - XML zapis zahtjeva za generiranje kontroliranog jezika

```
<Generete>
  <Triplets>
    <Triple conn="and">
      <Subject>
        <Quantifier value="single" />
        <Phrase
id="http://www.pmfst.hr/racsus.owl#Aritmeticko_logicka_jedinica" />
      </Subject >
      <Predicate>
        <Negative value="false" />
        <Phrase id="gen" />
      </Predicate>
      <Object>
        <Quantifier value="single" />
        <Phrase
id="http://www.pmfst.hr/racsus.owl#Centralna_procesorska_jedinica" />
      </Object>
    </Triple>
    <Triple conn="and">
      <Subject>
        <Quantifier value="single" />
        <Phrase
id="http://www.pmfst.hr/racsus.owl#Aritmeticko_logicka_jedinica" />
      </Subject >
      <Predicate>
        <Negative value="false" />
        <Phrase id="http://www.pmfst.hr/racsus.owl#izvrsava" />
      </Predicate>
      <Object>
        <Quantifier value="single" />
        <Phrase id="http://www.pmfst.hr/racsus.owl#Aritmeticka_operacija"
/>
      </Object>
    </Triple>
    <Triple conn="and">
      <Subject>
        <Quantifier value="single" />
        <Phrase
id="http://www.pmfst.hr/racsus.owl#Aritmeticko_logicka_jedinica" />
      </Subject >
      <Predicate>
        <Negative value="false" />
        <Phrase id="http://www.pmfst.hr/racsus.owl#izvrsava" />
      </Predicate>
      <Object>
        <Quantifier value="single" />
        <Phrase id="http://www.pmfst.hr/racsus.owl#Logicka_operacija" />
      </Object>
    </Triple>
  </Triplets>
  <Phrases>
    <Phrase
id="http://www.pmfst.hr/racsus.owl#Aritmeticko_logicka_jedinica"
value="Aritmetičko logička jedinica" />
    <Phrase id="http://www.pmfst.hr/racsus.owl#Centralna procesorska
jedinica" value="Centralna procesorska jedinica" />
    <Phrase id="http://www.pmfst.hr/racsus.owl#Aritmeticka operacija"
value="Aritmetička operacija" />
```

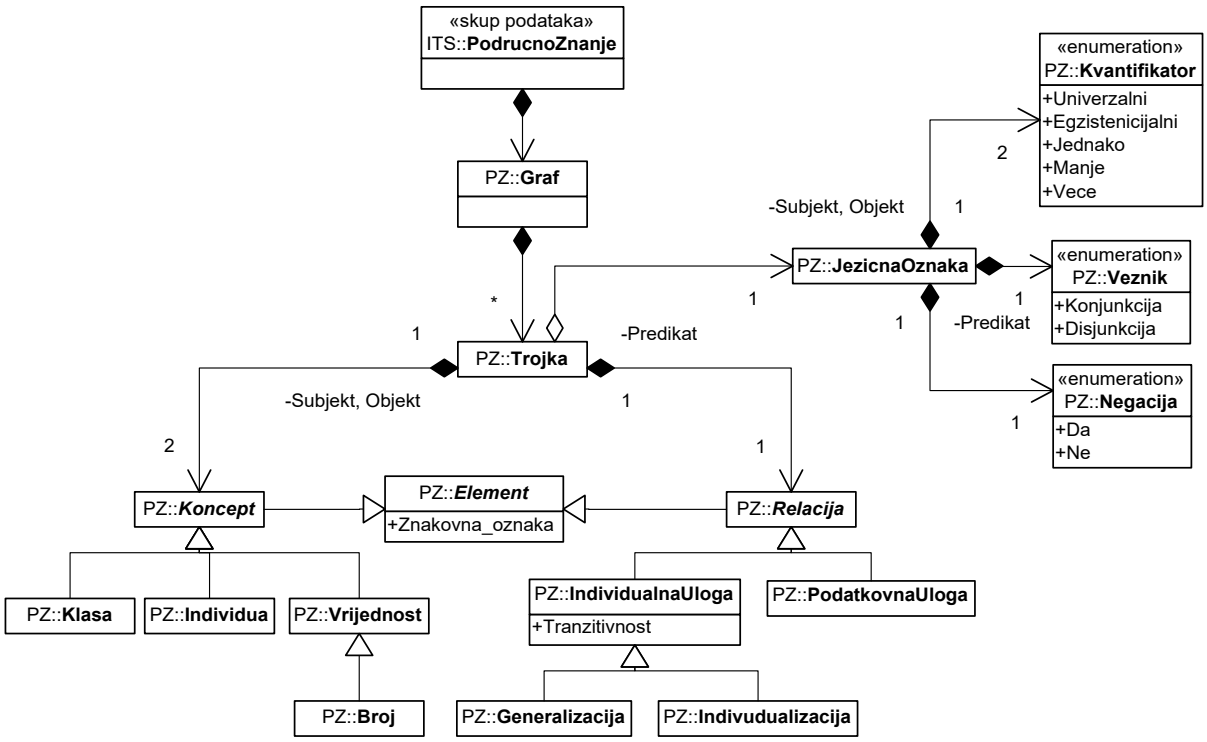
```
    <Phrase id="http://www.pmfst.hr/racsus.owl#Logicka_operacija"
value="Logička operacija" />
    <Phrase id="http://www.pmfst.hr/racsus.owl#izvrsava" value="izvršava"
/>
    <Phrase id="gen" value="je vrsta" />
  </Phrases>
</Generete>
```

8.4 Prilog D - UML dijagrami klasa skupova podataka CoLaB Tutor-a

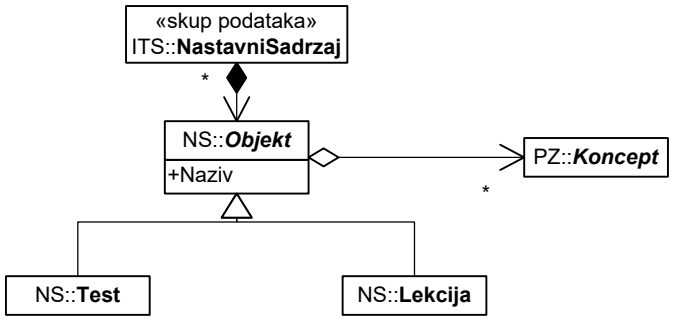
8.4.1 Prilog D.1 - UML dijagram klasa paketa Owl



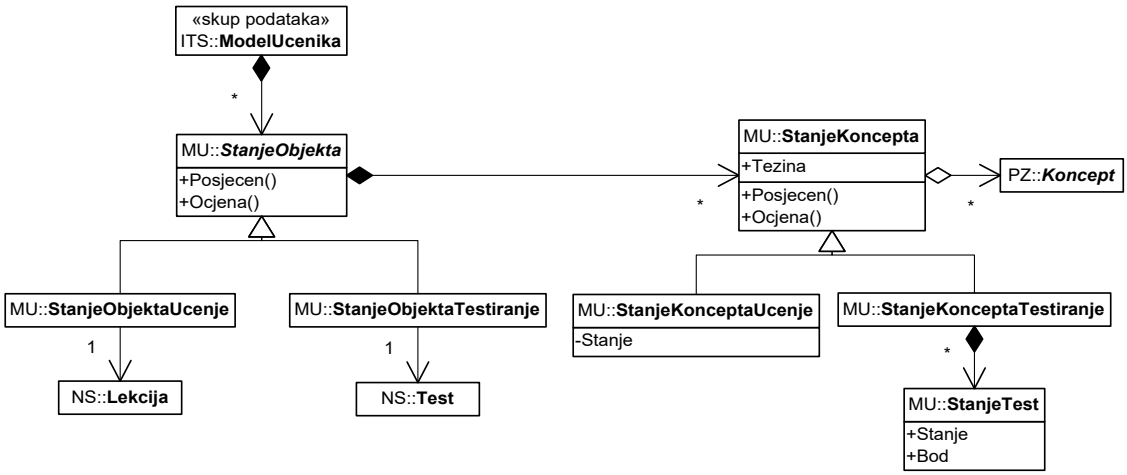
8.4.2 Prilog D.2 - UML dijagram klasa paketa PZ



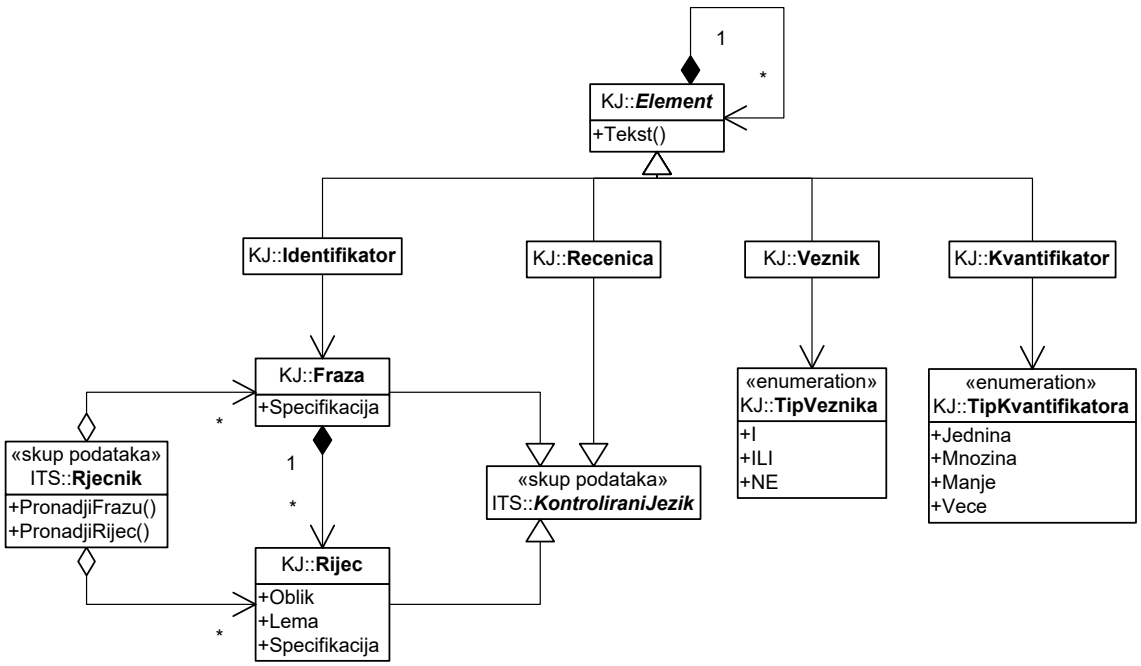
8.4.3 Prilog D.3 - UML dijagram klasa paketa NS



8.4.4 Prilog D.4 - UML dijagram klasa paketa MU

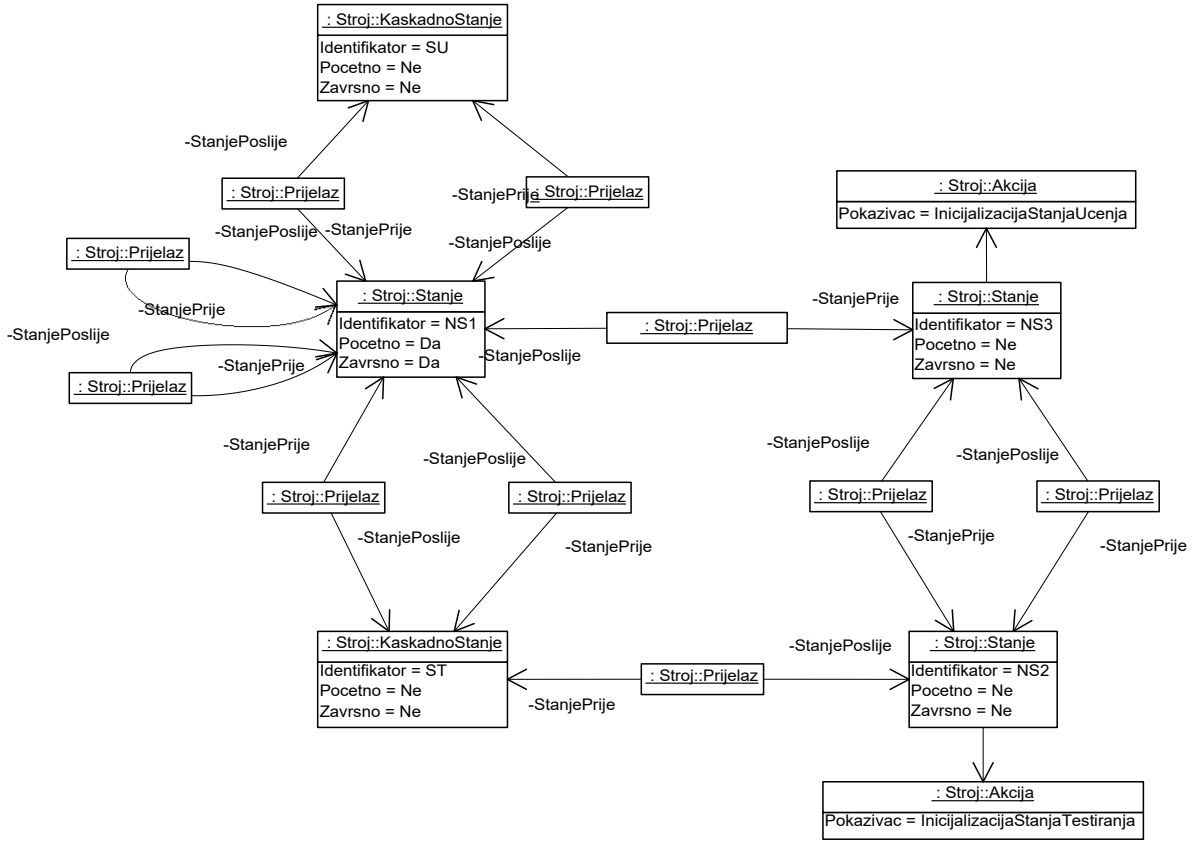


8.4.5 Prilog D.5 - UML dijagram klasa paketa KJ

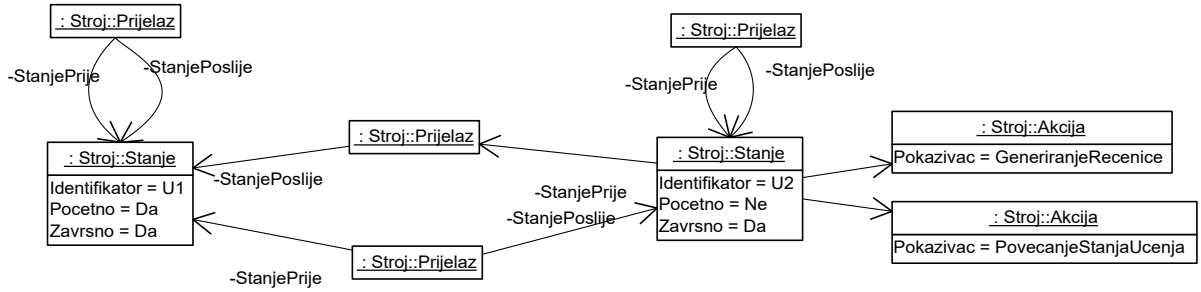


8.5 Prilog E - UML dijagrami objekata strojeva s konačnim brojem stanja

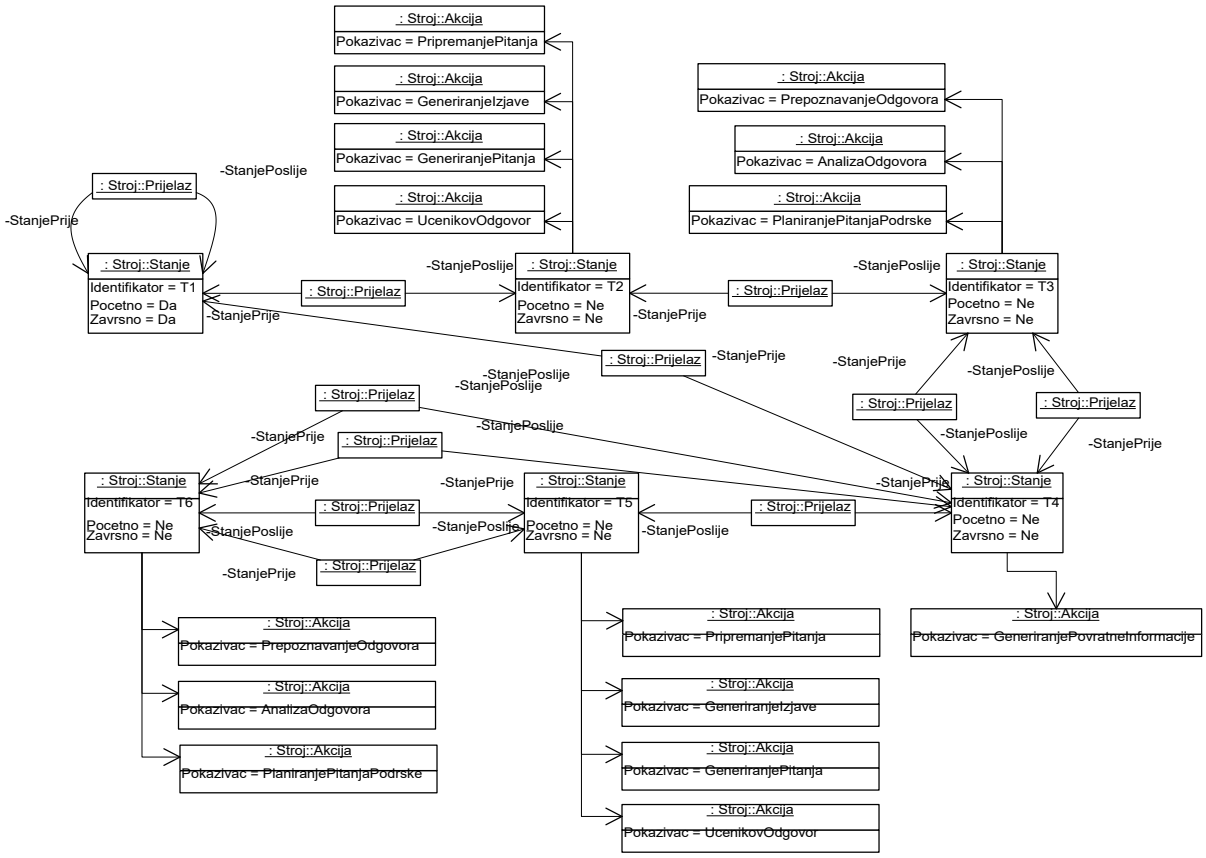
8.5.1 Prilog E.1 - UML dijagram objekata stroja za učenje i testiranje



8.5.2 Prilog E.2 - UML dijagram objekata stroja za učenje



8.5.3 Prilog E.3 - UML dijagram objekata stroja za testiranje



8.6 Prilog F - Anketni upitnici

8.6.1 Prilog F.1 - Anketni upitnik prve eksperimentalne skupine

1. Koristiš li računalo dok učiš?
   a. da
   b. ne

2. Nastavni sadrţaj računarstva obraĎen uz pomoć CoLaB Tutor-a razumio/la sam:
   a. u potpunosti
   b. djelomično
   c. nisam ih uopće razumio/la

3. Kako je vrijeme odmicalo, učenje računarstva uz pomoć CoLaB Tutor-a mi je bilo:
   a. teţe (sloţenije)
   b. lakše (jednostavnije)
   c. jednako teško
   d. jednako lako

4. Nastava računarstva uz pomoć CoLaB Tutor-a je u odnosu na klasičnu nastavu:
   a. zanimljivija
   b. manje zanimljiva
   c. jednako zanimljiva

5. Ţeliš li nastaviti učiti računarstvo uz pomoć CoLaB Tutor-a?
   a. ţelim
   b. ne ţelim
   c. ţelim povremeno

6. Tijekom učenja nastavnog sadrţaja računarstva uz pomoć CoLaB Tutor-a moj interes za učenjem:
   a. je porastao
   b. se smanjio
   c. ostao isti

7. Korištenjem CoLaB Tutor-a u nastavi mogla bi se ostvariti kvalitetnija nastava?
   a. točno
   b. netočno
   c. djelomično točno

8. Ocijeni ocjenom od 1 do 5 korištenje CoLaB Tutor-a?

   1 2 3 4 5

9. Napiši mišljenje o učenju uz pomoć CoLaB Tutor-a!

8.6.2 Prilog F.2 - Anketni upitnik druge eksperimentalne skupine

1. Zaokruţite tvrdnju koju smatrate točnom!
   a. Nastavne sadrţaje oblikovanje uz pomoć CoLaB Tutor-a bolje sam razumio/la od nastavnih sadrţaja oblikovanih u xTEx-Sys-u.
   b. Nastavne sadrţaje oblikovane uz pomoć CoLaB Tutor-a jednako sam razumio/la kao i nastavne sadrţaje oblikovane u xTEx-Sys-u.
   c. Nastavne sadrţaje oblikovane uz pomoć CoLaB Tutor-a lošije sam razumio/la od nastavnih sadrţaja oblikovanih u xTEx-Sys-u.

2. Kako je vrijeme odmicalo, učenje računarstva uz pomoć CoLaB Tutor-a mi je bilo:
   a. teţe (sloţenije)
   b. lakše (jednostavnije)
   c. jednako teško
   d. jednako lako

3. Učenje uz pomoć CoLaB Tutor-a u odnosu na učenje uz pomoć xTEx-Sys je:
   a. zanimljivije
   b. manje zanimljivo
   c. jednako zanimljivo

4. Zaokruţi: bi li učio/la računarstvo uz pomoć CoLaB Tutor-a ili uz pomoć xTEx-Sys?

   CoLaB Tutor xTEx-Sys

5. Tijekom učenja nastavnog sadrţaja računarstva uz pomoć CoLaB Tutor-a moj interes za učenjem:
   a. je porastao
   b. se smanjio
   c. ostao isti

6. Zaokruţi: bi li nastava bila kvalitetnija koristeći CoLaB Tutor ili xTEx-Sys?

   CoLaB Tutor xTEx-Sys

7. Ocijeni ocjenom od 1 do 5 korištenje CoLaB Tutor-a?

   1 2 3 4 5

8. Napiši svoje mišljenje usporeĎujući učenje uz pomoć CoLaB Tutor-a i učenje uz pomoć xTEx-Sys!

Model inteligentnog tutorskog sustava zasnovan na obradi kontroliranog jezika nad ontologijom

Saţetak:

Inteligentni tutorski sustavi su sustavi zasnovani na znanju i komunikaciji znanjem. CoLaB Tutor je inteligentni tutorski sustav koji posjeduje ontološki opisano područno znanje, a komunikaciju provodi kontroliranim jezikom.
Model CoLaB Tutor-a je temeljen na tradicionalnoj arhitekturi inteligentnih tutorskih sustava. Stručnjak je sudionik CoLaB Tutor-a zaduţen za ontološko opisivanje područnog znanja.
Sustav CoLaB Tutor-a je zaduţen da, na osnovi ontološki opisanog područnog znanja, postavi područno znanje, nastavni sadrţaj i model učenika. Učenik se uči, poučava i testira komunicirajući na kontroliranom jeziku sa sustavom.
Tijekom učenja se generiraju rečenice kontroliranog jezika, a za vrijeme testiranja provodi se tutorski dijalog voĎen strojem s konačnim brojem stanja. U tutorskom dijalogu se, osim generiranja pitanja, prepoznaje učenikov odgovor, generiraju povratne informacije i planiraju pitanja podrške.

Ključne riječi:

Inteligentni tutorski sustav, obrada kontroliranog jezika, ontologija, tutorski dijalog.

Ţivotopis

RoĎen sam 8. veljače.1975. godine u Splitu. Osnovnu školu sam pohaĎao i završio u Splitu. 1993. godine maturirao sam na Prirodoslovno - matematičkoj gimnaziji (III. Gimnazija) u Splitu. 1993. godine upisao sam se na Fakultet prirodoslovno matematičkih znanosti i odgojnih područja Sveučilišta u Splitu, smjer Matematika-Informatika.
8. veljače 2001. godine diplomirao sam na temu "Pristup projektiranju računalom poduprtih inteligentnih tutorskih sustava" i postigao visoku spremu i stručno zvanje profesor matematike i informatike.
Od 1. kolovoza 2001. godine zaposlen sam kao znanstveni novak na Fakultetu prirodoslovno - matematičkih znanosti i odgojnih područja Sveučilišta u Splitu gdje sam prijavljen na znanstveno-istraţivačkom projektu 177110 "Računalni i didaktički aspekti inteligentnih autorskih alata u obrazovanju" Ministarstva znanosti i tehnologije, glavni istraţivač prof. dr. sc. Slavomir Stankov.
Za vrijeme rada na Fakultetu prirodoslovno-matematičkih znanosti i odgojnih područja Sveučilišta u Splitu odrţavao sam vjeţbe iz kolegija "Uvod u računarstvo" i "Programiranje I", te sam za potrebe nastave napravio interne skripte za navedene kolegije. Osim toga, odrţavao sam i vjeţbe iz kolegija "Računalni praktikum I" s tematikom objektno orijentirana metodologija za projektiranje programskih sustava.
19. veljače 2001. godine stekao sam istraţivačko zvanje mlaĎeg asistenta (znanstveni novak) za znanstveno područje Tehničkih znanosti, polje Računarstva.
6. studenog 2003. godine postavljen sam na radno mjesto asistenta.
U razdoblju od 2003. do 2005. godine sudjelovao sam na tehnologijskom projektu: TP-02/0177-01 "Web orijentirana inteligentna autorska ljuska", Ministarstva znanosti i tehnologije, glavni istraţivač prof. dr. sc. Slavomir Stankov.
Od 2005. godine odrţavao sam vjeţbe iz kolegija "Ekspertni sustavi" i "Programsko inţenjerstvo".
4. studenog 2005. godine magistrirao sam na temu "Model vrednovanja znanja u inteligentnim sustavima e-učenja". Mentor magistarskog rada mi je bio mentor prof. dr. sc. Slavomir Stankov.
Od 2007. godine sudjelovao sam na znanstveno-istraţivačkom projektu 177-0361994-1996 "Oblikovanje i vrednovanje inteligentnih sustava e-učenja" Ministarstva znanosti i tehnologije, glavni istraţivač prof. dr. sc. Slavomir Stankov.
Autor sam petnaestak znanstvenih i stručnih radova.

Model of Intelligent Tutoring System Based on Processing of Controlled Language over Ontology

Abstract:

Intelligent tutoring systems are knowledge based and knowledge communication systems. CoLaB Tutor is intelligent tutoring system which contains ontologically described domain knowledge and communicates by controlled language.
Model of CoLaB Tutor is based on traditional architecture of intelligent tutoring systems. Expert is an actor of CoLaB Tutor which is responsible for ontological description of domain knowledge. System is responsible for setting up domain knowledge, course and student model based on ontological description of domain knowledge. Learner is taught and tested while communicating by controlled language with the system.
Controlled language sentences are generated during learning, and tutoring dialogue is conducted and guided by finite state machines during testing. In tutoring dialogue, besides generating questions, learner's answers are recognized, feedback is generated and supporting questions are planned.

Keywords:

Intelligent tutoring system, controlled language processing, ontology, tutor dialogue.

Biography

I was born on 8th of February 1975 in Split. I attended and finished elementary school in Split. 1993. I graduated at High School of Natural Science and Mathematics in Split. 1993. I enrolled Faculty of Natural Science, Mathematics and Education, University of Split, course of Mathematics and Informatics.
8th of February 2001 I graduated on theme "Project Approach to Computer Based Intelligent Tutoring Systems" and obtained high skill and became Professor of Mathematics and Informatics.
From 1st of August 2001 I work as scientific recruit on Faculty of Natural Science, Mathematics and Education, University of Split applied on scientific and research project "Computer and didactical aspect of intelligent authoring tools in education" Ministry of Science and Technology, project coordinator prof. dr. sc. Slavomir Stankov.
During my work on Faculty of Natural Science, Mathematics and Education at University of Split I hold exercises in courses "Introduction to Computer Science" and "Programming I.", and I made internal scripts for named courses. Besides that, I hold exercise in course “Computer workshop I.” with theme of object-oriented methodology for projecting computer systems.
19th of February 2001 my research profession is raised into young assistant (scientific recruit) on scientific area of Technical science, field Computer Science.
6th of November 2003 I was setup on assistant working place.
From 2003 till 2005 I collaborated in technological project TP-02/0177-01 "Web oriented intelligent authoring shell", Ministry of Science and Technology, project coordinator prof. dr. sc. Slavomir Stankov.
Since 2005 I hold exercise in courses "Expert systems" and "Software engineering".
4th of November 2005 I mastered on theme "Knowledge Evaluation Model in Intelligent e-Learning Systems ". Mentor of the master thesis was prof. dr. sc. Slavomir Stankov.
Since 2007 I collaborated in scientific and research project 177-0361994-1996 "Design and evaluation of intelligent e-learning systems", Ministry of Science and Technology, project coordinator prof. dr. sc. Slavomir Stankov.
I authored over fifteen scientific and expert works.

UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Branko Žitko

MODEL OF INTELLIGENT TUTORING SYSTEM BASED ON PROCESSING OF CONTROLLED LANGUAGE OVER ONTOLOGY

DOCTORAL THESIS

Zagreb, 2010