Univerzitet u Beogradu
Matematički fakultet

Vesna Pajić

# KONAČNI TRANSDUKTORI U NADGLEDANJU VEBA

Magistarski rad

Beograd, 2010.

Mentor: prof. dr Duško Vitas
Matematički fakultet, Beograd

Članovi komisije: prof. dr Gordana Pavlović – Lažetić
Matematički fakultet, Beograd

prof. dr Ivan Obradović
Rudarsko - geološki fakultet, Beograd

# SADRŽAJ

# 1. UVOD

Eksplozija informacionih tehnologija poslednjih godina kao svoju najznačajniju i najzanimljiviju posledicu svakako ima Internet i veb (*World Wide Web*). Veliki broj informacija koje se nalaze na globalnoj mreži od koristi su svakom pojedincu. Nove informacije i sadržaji se dodaju neprekidno, tako da neki statistički podaci govore da se svakih sto dana duplira ukupna količina informacija raspoloživih na Internetu. Ovako brz razvoj veba je doveo do toga da se za različite životne i poslovne zahteve više ne postavlja pitanje da li postoji odgovarajuća informacija na Internetu, već se pojavljuje realan problem njenog efikasnog pronalaženja.

Obrada prirodnih jezika (*Natural Language Processing*) i računarska lingvistika igraju dominantnu ulogu u pokušaju da se problem pronalaženja određene, relevantne informacije reši. U zavisnosti od strukture elektronskog teksta, osobina jezika na kome je tekst napisan i potreba korisnika, ove discipline postižu veći ili manji uspeh. Međutim, razvoj Interneta i veba je doveo do nekoliko značajnih posledica koje u velikoj meri otežavaju problem obrade pojedinačnog teksta ili korpusa, i tako određuju pravce daljih istraživanja u oblasti pretrage informacija. Neke od njih su:

- količina informacija na vebu
- različitost strana na vebu
- dinamika pojavljivanja novih i izmene postojećih stranica na vebu
- oslanjanje korisnika na Internet i veb kao izvor informacija

U okviru ovog poglavlja biće razmotrene navedene osobine veba i njihov uticaj na proces pretrage informacija.

## 1.1. Količina informacija na vebu

U prvim godinama nakon nastanka veba, kada je na Internetu bilo značajno manje strana nego danas, metoda pretrage Interneta koja je podrazumevala da korisnik sledi linkove koji se nalaze na stranama (eng. *browsing*) je bio adekvatan i dominantan metod za lociranje relevantnih dokumenata. Međutim, razvojem veba i povećanjem količine strana na njemu, ovaj način pronalaženja informacija je postao nezadovoljavajući. Pojavila se potreba za sistemima za pretragu koji bi omogućili pronalaženje odgovarajuće informacije u što kraćem vremenskom roku.

Svi današnji pretraživači (*Google*, *AltaVista* i dr.) funkcionišu na sličan način. Ovi sistemi se sastoje od nekoliko podsistema, od kojih su najznačajniji sistem za preuzimanje strana sa Interneta (eng. *crawling*), sistem za indeksiranje i sistem za rangiranje strana. *Crawling* je proces koji podrazumeva pronalaženje postojećih strana na vebu. Kada je strana pronađena, ona se indeksira, tj. deo njenog sadržaja se smešta lokalno u neku bazu podataka i dodeljuju joj se određene ključne reči pomoću kojih se olakšava pretraga.

S obzirom da se svaki sistem za pretragu suočava sa ogromnim brojem veb strana koje treba da preuzme i obradi, svaki od njih postavlja određena ograničenja [1]. Da bi *crawler* uopšte pronašao neku stranu neophodno je ili da je autor strane obavestio sistem o njenom postojanju prijavljivanjem URL strane ili da neka druga strana koju je *crawler* posetio sadrži hiper-vezu koja ukazuje na nju. U suprotnom, strana je "nevidljiva" za pretraživač, a samim tim i za korisnika. Sam proces preuzimanja strana (*crawling*) je veoma zahtevna operacija koja podrazumeva računare sa dovoljno memorije i procesne snage, kako bi mogla da prati stalno povećanje količine veb strana i informacija na Internetu. S obzirom da resursi ni jednog pretraživača nisu neograničeni, svaki *crawler* ima određena ograničenja koja zadržavaju proces *crawlinga* u granicama raspoloživih resursa. Kod nekih pretraživača se ograničava ukupan broj strana u indeksu i izbacuju stare strane kada se pojave nove, dok se kod drugih ograničava frekvencija ponovnih poseta stranama. Kad god pretraživač odluči da limitira proces preuzimanja i indeksiranja strana, deo informacija postaje nedostupan korisniku.

## 1.2. Različitost strana na vebu

Veb strane, iako po svojoj strukturi moraju da zadovolje osnovne principe HTML jezika, kako bi mogle da budu prikazane u veb čitačima, ipak se međusobno veoma razlikuju. One sadrže tekstove o najrazličitijim temama i na skoro svim jezicima sveta, ali i druge formate podataka, kakve su audio i video datoteke, slike, grafički elementi i dr. Zbog toga često postoji velika razlika u efikasnosti algoritama za rešavanje problema iz oblasti pretrage informacija. Pojedini algoritmi daju odlične rezultate kada se primene na tekstove na engleskom jeziku, ali veoma loše kada se radi o tekstovima na jeziku sa bogatom morfologijom, kakav je i srpski jezik.

Ova razlika će biti ilustrovana na primeru reči (toponima) *New York*. U većini zapadno evropskih jezika naziv ovog grada će biti napisan na isti način. Međutim, u srpskom jeziku, ime ovog grada može biti zapisano na desetinu različitih načina. Ovde će biti istaknute samo neke od varijanti [2]:

- korišćenje različitih pisama dovodi do dva različita oblika: *Njujork* i *Њујорк*. Za ćirilično i latinično pismo u upotrebi su različite kodne šeme. Takođe, u latiničnom pismu digraf "Nj" ima dve interpretacije, kao jedan karakter koji odgovara ćiriličnom њ, ili kao grupa suglasnika N+j. Korišćenjem *Unicode*-a [3], slovo Nj može biti predstavljeno kao digraf pomoću dva koda ili kao ligatura pomoću jednog koda. Dalje, slovo Nj može biti zapisano kao Nj ili NJ. Dakle, koristeći *Unicode* postoji četiri različita načina za predstavljanje slova Nj.
- toponim *New York* se u srpskom jeziku najčešće piše u transkribovanom obliku, u skladu sa fonetskim principima srpskog pravopisa, ali se u pojedinim tekstovima koristi i originalan oblik.
- promene po padežima i mogućnost izvođenja novih reči dovode do više desetina oblika toponima *New York* u srpskom jeziku.

Većina pretraživača opšte namene ima sličnu formu za postavljanje upita, u okviru koje korisnik zadaje upit zadavanjem ključnih reči. Na primer, ukoliko je potrebno pronaći tekstove na engleskom jeziku koji se odnose na *Njujork*, bilo bi dovoljno tražiti pojavljivanje toponima "New York" u tekstu. Kao rezultat ovakve pretrage bili bi pronađeni tekstovi koji sadrže sledeće izraze:

*"New York city..", "the streets of New York", "New York streets"*

Izrazi sa istim značenjem na srpskom jeziku bili bi:

*"grad Njujork", "grad New York", "ulice Njujorka", "njujorške ulice"*

Ukoliko je potrebno pronaći tekstove na srpskom jeziku koji se odnose na *Njujork*, postavljanje upita zadavanjem ključne reči nije metod koji bi doneo zadovoljavajuće rezultate. Naime, ukoliko kao ključnu reč korisnik traži toponim "New York", pretraživači o kojima je reč će pronaći samo jedan deo tekstova koji su dostupni na vebu, a odnose se na grad *Njujork*, i to one u kojima se toponim *Njujork* pojavljuje baš u traženom obliku ("*New York*"). Svi ostali dokumenti u kojima se pojavljuje neki od promenjenih oblika ove ključne reči, na primer "grad Njujork", "ulice Njujorka", "njujorške ulice" i sl., će izostati iz rezultata pretrage.

Poslednjih godina pretraživač *Google*, u nastojanjima da poboljša svoj proces pretrage i približi ga korisnicima, vraća rezultate u kojima se ključna reč pojavljuje u izmenjenim oblicima. Ovo je svakako dobar pokušaj da pretraga poboljša, ali je veliki problem što korisnik nema kontrolu nad ovim procesom.

Zbog svega navedenog postavljanje upita pomoću ključnih reči nije adekvetan metod, pa je neophodno pružiti neki drugi način kojim bi bilo moguće postavljati kompleksnije upite.

## 1.3. Dinamika pojavljivanja novih i izmene postojećih stranica na vebu

Sistem za preuzimanje strana bilo kog pretraživača može u jednom danu da preuzme i obradi ograničen broj strana sa veba. Zbog toga postoji vremenska razlika od trenutka kada je neka strana postavljena na veb do trenutka kada je *crawler* otkrije. Takođe, postoji i vremenska razlika između dve posete *crawlera* istoj strani u potrazi za novim sadržajima, što predstavlja veliki problem kod strana koje veoma često menjaju svoj sadržaj, kakve su na primer strane sa dnevnim vestima ili različiti forumi.

U većini slučajeva sistem za pretragu sam određuje interval u kome se vrši ponovno preuzimanje strane. Da bi što bolje odgovorili zahtevima korisnika, neki pretraživači, među kojima je i *Google*, ostavljaju mogućnost autoru veb sajta da smanji ovaj interval, tj. da poveća učestalost kojom *crawler* pristupa sajtu, ukoliko se radi o sajtu koji često menja sadržaj.

Još jedan od načina prevazilaženja problema dinamičnih sadržaja jeste koncept RSS ("*Really Simple Sindication*"). RSS koncept podrazumeva da autor određenog veb sajta uređuje i održava listu promena na veb sajtu. Ova lista se naziva *RSS feed*. Korisnici, zainteresovani da prate promene na sajtu, mogu da pristupe ovoj listi automatski pomoću posebnih programa, tzv. RSS agregatora.

U oba slučaja, da li će sadržaji biti vidljivi korisniku ili ne, zavisi od autora sajta. Ukoliko autor nije obezbedio *RSS feed,* niti je zahtevao smanjenje intervala posete *crawler*-a strani, korisniku ne preostaje ništa drugo nego da lično redovno proverava sadržaje određenog veb sajta u potrazi za određenom informacijom.

## 1.4. Oslanjanje korisnika na Internet kao izvor informacija

Razvoj informacionih tehnologija kao svoju logičnu posledicu ima i sve veći broj korisnika Interneta. Danas je za veliki broj stanovništva život bez Interneta nezamisliv. Bez obzira da li se koristi za obrazovanje, posao, informisanje ili zabavu, Internet i veb su postali sastavni deo naše svakodnevice.

Prema statističkim podacima sajta **www.internetworldstats.com** [4] broj korisnika Interneta u svetu je od 2000. godine do 2009. godine porastao za 1,4 milijarde. Takođe, isti izvor navodi da je u martu 2010. godine u svetu bilo 1,7 milijardi ljudi koji koriste Internet, što čini 25,6% svetske populacije. Bez obzira na to što su korisnici Interneta različitih profesija i nivoa obrazovanja, što govore različitim jezicima i imaju različite potrebe za informacijama, ipak se svi suočavaju sa istim ili sličnim formama za postavljanje upita pretraživačima. Uglavnom su to pretrage po ključnim rečima, sa eventualnim naprednim opcijama koje korisniku omogućavaju da dodatno ograniči pretragu. Korisnik je taj koji svojom veštinom izbora ključnih reči i utrošenim vremenom uspeva ili ne da pronađe traženu informaciju. Na taj način, korisnik se prilagođava pretraživaču, umesto da bude obrnuto.

# 2. DEFINICIJA PROBLEMA

Iako su savremeni sistemi za pretragu informacija na vebu relativno uspešni u pronalaženju relevantnih dokumenata, ipak veliki broj problema ostaje nerešen, pa je time skup upita koji su uspešno obrađeni ograničen. U ovom radu će problem pronalaženja određene informacije biti razmatran sa dva aspekta. Prvi je poboljšanje postavljanje upita i omogućavanje zadavanje kompleksnijih pretraga. Drugi aspekt je pristup sadržajima veb-strana u što kraćem periodu od trenutka njihovog pojavljivanja.

## 2.1. Postavljanje upita

Većina upita koje korisnici postavljaju tražeći određenu informaciju na vebu može se svrstati u jednu od tri kategorije [5]:

- informacioni upiti
- navigacioni upiti
- transakcioni upiti

Informacioni upiti uglavnom tragaju za opštom informacijom o nekoj temi. Korisnik želi da dobije određenu informaciju. Na primer, poljoprivrednik koji želi da počne da se bavi uzgojem lekovitog bilja mogao bi da postavi upit "uzgoj lekovitog bilja". Među rezultatima pretrage, korisnik bi kao relevantna dokumenta prepoznao veb-strane koje sadrže neku informaciju o temi koju je zadao, bilo da se radi o novinskim člancima, naučnim radovima ili prezentacijama preduzeća koje se bave ovim poslom. Većina današnjih pretraživača bi uspešno obavila ovu pretragu i zadovoljila potrebe korisnika.

Navigacioni upiti tragaju za tačno određenim veb-sajtom ili stranom koju korisnik ima na umu. Na primer, poljoprivrednik koji želi da se bavi uzgojem lekovitog bilja mogao bi da traži zvaničnu prezentaciju nekog preduzeća ili institucije, recimo Apotekarske ustanove Beograd, kako bi pronašao informacije za kontakt i pokušao da plasira svoj proizvod. U tom slučaju korisnik prilikom postavljanja upita "Apotekarska ustanova Beograd" očekuje samo jedan rezultat: veb-sajt ove ustanove. Uspešnost pretraživača u ovom slučaju zavisi od toga da li se konkretna veb strana nalazi u njegovom indeksu ili ne.

Transakcioni upiti su upiti koji imaju za cilj da pronađu dokumenta koja korisniku omogućavaju da izvrši neku transakciju ili operaciju, na primer naručivanje nekog proizvoda, rezervisanje hotela ili preuzimanje datoteke. U ovim slučajevima, pretraživač bi kao relevantne rezultate trebalo da vrati skup veb-strana koje pružaju ovu vrstu usluga korisniku. Pod pretpostavkom da korisnik ispravno postavi upit, većina današnjih pretraživača bi uspešno odgovorila na zahtev korisnika.

Međutim, ukoliko korisnik traži informaciju o nekom komplikovanijem događaju, savremeni pretraživači ne bi uspeli da odgovore na njegove potrebe, prvenstveno zbog toga što ne postoji mogućnost postavljanja kompleksnih upita koji nisu bazirani na ključnim rečima.

*Primer 1.*

Korisnik želi da nađe tekstove o uzgoju bilo koje vrste lekovitog bilja. Pod pretpostavkom da bi korisnik prepoznao kao relevantne dokumente koji sadrže neki od izraza "uzgoj nane", "uzgoj kamilice" ili "uzgoj majčine dušice", savremeni pretraživači koji nude opciju postavljanja upita uz upotrebu logičkih operatora bi donekle zadovoljili potrebe korisnika. Ipak, izostala bi dokumenta koja sadrže neku od pravopisnih varijacija, flektivnih ili derivacionih oblika ovih izraza.

*Primer 2.*

Korisnik želi da pronađe dokumenta koja sadrže tekstove o koaliciji između Demokratske stranke Srbije i Nove Srbije. Izrazi koji se odnose na ovu koaliciju, a koji su se tokom prve polovine 2008. godine pojavljivali u elektronskim medijima, su:

*koalicija DSS-NS*
*koalicija DSS-Nova Srbija*
*narodnjačka koalicija*
*narodnjaci*
*DSS-NS*

Pretraživači opšte namene ne omogućavaju postavljanje upita kojim bi bio opisan ovaj događaj.

Iz svega navedenog, kao glavni problem nameće se poboljšanje načina postavljanja upita od strane korisnika, koji bi omogućio definisanje kompleksnih događaja. U okviru predloženog rešenja, ovaj problem može da se prevaziđe upotrebom konačnih transduktora nad odgovarajućim leksičkim resursima.

## 2.2. Pristup sadržajima veb strana nevidljivim za pretraživače opšte namene

Sistemi za preuzimanje i indeksiranje strana pretraživača opšte namene pristupaju novopostavljenim sadržajima na vebu tek nakon izvesnog vremena. Zbog toga sadržaji na veb-stranama koje često menjaju svoj sadržaj, kao što su strane sa vestima ili forumi, ostaju nevidljivi za pretraživače.

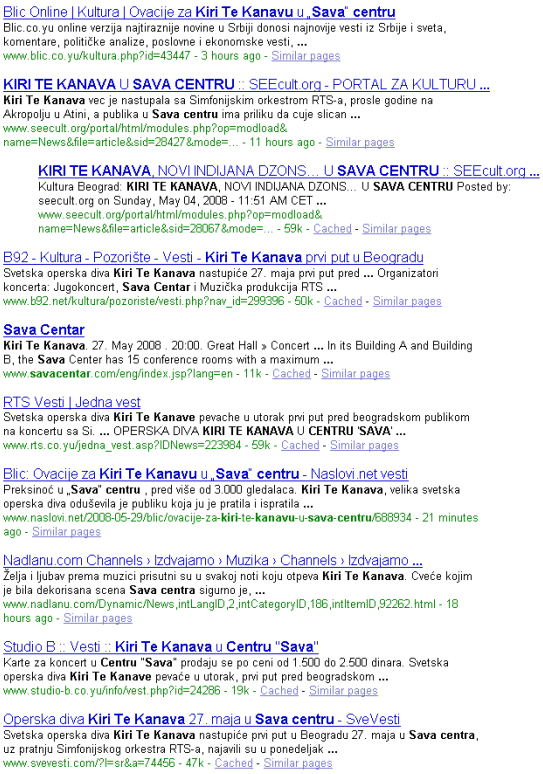
Ukoliko su korisniku neophodne informacije u što kraćem vremenskom roku od trenutka njihovog pojavljivanja na Internetu, neophodno je razviti posebne sisteme za pretragu koji bi ciljano tražili određenu informaciju po nalogu korisnika.

*Primer 3.*

Dana 27.5.2008. godine u beogradskom Sava Centru, održan je koncert operske pevačice Kiri Te Kanava. Korisnik je dva dana kasnije (29.5.2008.) želeo da pronađe bilo kakav izveštaj ili kritiku o održanom koncertu koristeći pretraživače opšte namene. U tu

svrhu je zadata pretraga sa upitom "kiri te kanava sava centar" dvoma pretraživačima, *Google* i *AltaVista*. Rezultat je bio sledeći:

- pretraživač Google je vratio oko 4000 rezultata. Među njima, informacije o održanom koncertu su se nalazile na dokumentima sa samo dva veb sajta, **www.blic.co.yu** i **www.b92.net**, dok su se ostala dokumenta odnosila ili na najave koncerta ili su sadržali tekstove u kojima se pominju Kiri Te Kanava i Sava Centar, ali u različitim kontekstima. Pa ipak, skoro sva elektronska izdanja srpskih dnevnih novina su imala vest o održanom koncertu u okviru svojih veb sajtova.


Blic Online | Kultura | Ovacije za **Kiri Te Kanavu** u „**Sava**" **centru**
Blic.co.yu online verzija najtiraznije novine u Srbiji donosi najnovije vesti iz Srbije i sveta, komentare, političke analize, poslovne i ekonomske vesti, ...
www.blic.co.yu/kultura.php?id=43447 - 3 hours ago - Similar pages

**KIRI TE KANAVA** U **SAVA CENTRU** :: SEEcult.org - PORTAL ZA KULTURU ...
**Kiri Te Kanava** vec je nastupala sa Simfonijskim orkestrom RTS-a, prosle godine na Akropolju u Atini, a publika u **Sava centru** ima priliku da cuje slican ...
www.seecult.org/portal/html/modules.php?op=modload&name=News&file=article&sid=28427&mode=... - 11 hours ago - Similar pages

**KIRI TE KANAVA**, NOVI INDIJANA DZONS... U **SAVA CENTRU** :: SEEcult.org ...
Kultura Beograd: **KIRI TE KANAVA**, NOVI INDIJANA DZONS... U **SAVA CENTRU** Posted by: seecult.org on Sunday, May 04, 2008 - 11:51 AM CET ...
www.seecult.org/portal/html/modules.php?op=modload&name=News&file=article&sid=28067&mode=... - 59k - Cached - Similar pages

B92 - Kultura - Pozorište - Vesti - **Kiri Te Kanava** prvi put u Beogradu
Svetska operska diva **Kiri Te Kanava** nastupiće 27. maja prvi put pred ... Organizatori koncerta: Jugokoncert, **Sava Centar** i Muzička produkcija RTS ...
www.b92.net/kultura/pozoriste/vesti.php?nav_id=299396 - 50k - Cached - Similar pages

**Sava Centar**
**Kiri Te Kanava**. 27. May 2008 . 20:00. Great Hall » Concert ... In its Building A and Building B, the **Sava** Center has 15 conference rooms with a maximum ...
www.**savacentar**.com/eng/index.jsp?lang=en - 11k - Cached - Similar pages

RTS Vesti | Jedna vest
Svetska operska diva **Kiri Te Kanave** pevache u utorak prvi put pred beogradskom publikom na koncertu sa Sl. ... OPERSKA DIVA **KIRI TE KANAVA** U **CENTRU 'SAVA'** ...
www.rts.co.yu/jedna_vest.asp?IDNews=223984 - 59k - Cached - Similar pages

Blic: Ovacije za **Kiri Te Kanavu** u „**Sava**" **centru** - Naslovi.net vesti
Preksinoć u „**Sava**" **centru** , pred više od 3.000 gledalaca. **Kiri Te Kanava**, velika svetska operska diva oduševila je publiku koja ju je pratila i ispratila ...
www.naslovi.net/2008-05-29/blic/ovacije-za-**kiri**-te-**kanavu**-u-**sava**-**centru**/688934 - 21 minutes ago - Similar pages

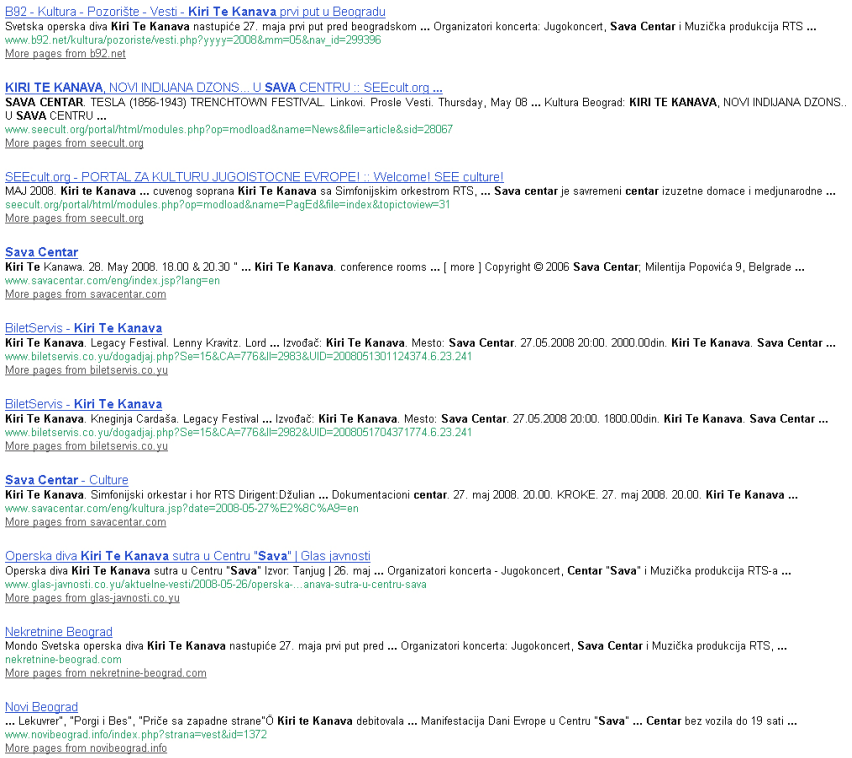
Nadlanu.com Channels › Izdvajamo › Muzika › Channels › Izdvajamo ...
Želja i ljubav prema muzici prisutni su u svakoj noti koju otpeva **Kiri Te Kanava**. Cveće kojim je bila dekorisana scena **Sava centra** sigurno je, ...
www.nadlanu.com/Dynamic/News,intLangID,2,intCategoryID,186,intItemID,92262.html - 18 hours ago - Similar pages

Studio B :: Vesti :: **Kiri Te Kanava** u **Centru** "**Sava**"
Karte za koncert u **Centru** "**Sava**" prodaju se po ceni od 1.500 do 2.500 dinara. Svetska operska diva **Kiri Te Kanave** pevaće u utorak, prvi put pred beogradskom ...
www.studio-b.co.yu/info/vest.php?id=24286 - 19k - Cached - Similar pages

Operska diva **Kiri Te Kanava** 27. maja u **Sava centru** - SveVesti
Svetska operska diva **Kiri Te Kanava** nastupiće prvi put u Beogradu 27. maja u **Sava centra**, uz pratnju Simfonijskog orkestra RTS-a, najavili su u ponedeljak ...
www.svevesti.com/?l=sr&a=74456 - 47k - Cached - Similar pages

*Sl. 1 Google rezultati za pretragu "kiri te kanava sava centar"*

- pretraživač *AltaVista* je vratio oko 1300 rezultata, među kojima ni jedan nije sadržao vesti o održanom koncertu. Prvi na listi je bio dokument sa sajta **www.b92.net**, ali koji sadrži informacije koje se odnose na najavu koncerta.


B92 - Kultura - Pozorište - Vesti - **Kiri Te Kanava** prvi put u Beogradu
Svetska operska diva **Kiri Te Kanava** nastupiće 27. maja prvi put pred beogradskom **...** Organizatori koncerta: Jugokoncert, **Sava Centar** i Muzička produkcija RTS **...**
www.b92.net/kultura/pozoriste/vesti.php?yyyy=2008&mm=05&nav_id=299396
More pages from b92.net

**KIRI TE KANAVA**, NOVI INDIJANA DZONS... U **SAVA** CENTRU :: SEEcult.org ...
**SAVA CENTAR**. TESLA (1856-1943) TRENCHTOWN FESTIVAL. Linkovi. Prosle Vesti. Thursday, May 08 **...** Kultura Beograd: **KIRI TE KANAVA**, NOVI INDIJANA DZONS.. U **SAVA** CENTRU **...**
www.seecult.org/portal/html/modules.php?op=modload&name=News&file=article&sid=28067
More pages from seecult.org

SEEcult.org - PORTAL ZA KULTURU JUGOISTOCNE EVROPE! :: Welcome! SEE culture!
MAJ 2008. **Kiri te Kanava ...** cuvenog soprana **Kiri Te Kanava** sa Simfonijskim orkestrom RTS, **... Sava centar** je savremeni **centar** izuzetne domace i medjunarodne **...**
seecult.org/portal/html/modules.php?op=modload&name=PagEd&file=index&topictoview=31
More pages from seecult.org

**Sava Centar**
**Kiri Te** Kanawa. 28. May 2008. 18.00 & 20.30 " **... Kiri Te Kanava**. conference rooms **...** [ more ] Copyright © 2006 **Sava Centar**; Milentija Popovića 9, Belgrade **...**
www.savacentar.com/eng/index.jsp?lang=en
More pages from savacentar.com

BiletServis - **Kiri Te Kanava**
**Kiri Te Kanava**. Legacy Festival. Lenny Kravitz. Lord **...** Izvođač: **Kiri Te Kanava**. Mesto: **Sava Centar**. 27.05.2008 20.00. 2000.00din. **Kiri Te Kanava**. **Sava Centar ...**
www.biletservis.co.yu/dogadjaj.php?Se=15&CA=776&II=2983&UID=20080513011243746.23.241
More pages from biletservis.co.yu

BiletServis - **Kiri Te Kanava**
**Kiri Te Kanava**. Kneginja Cardaša. Legacy Festival **...** Izvođač: **Kiri Te Kanava**. Mesto: **Sava Centar**. 27.05.2008 20:00. 1800.00din. **Kiri Te Kanava**. **Sava Centar ...**
www.biletservis.co.yu/dogadjaj.php?Se=15&CA=776&II=2982&UID=20080517043717746.23.241
More pages from biletservis.co.yu

**Sava Centar** - Culture
**Kiri Te Kanava**. Simfonijski orkestar i hor RTS Dirigent:Džulian **...** Dokumentacioni **centar**. 27. maj 2008. 20.00. KROKE. 27. maj 2008. 20.00. **Kiri Te Kanava ...**
www.savacentar.com/eng/kultura.jsp?date=2008-05-27%E2%8C%A9=en
More pages from savacentar.com

Operska diva **Kiri Te Kanava** sutra u Centru "**Sava**" | Glas javnosti
Operska diva **Kiri Te Kanava** sutra u Centru "**Sava**" Izvor: Tanjug | 26. maj **...** Organizatori koncerta - Jugokoncert, **Centar** "**Sava**" i Muzička produkcija RTS-a **...**
www.glas-javnosti.co.yu/aktuelne-vesti/2008-05-26/operska-...anava-sutra-u-centru-sava
More pages from glas-javnosti.co.yu

Nekretnine Beograd
Mondo Svetska operska diva **Kiri Te Kanava** nastupiće 27. maja prvi put pred **...** Organizatori koncerta: Jugokoncert, **Sava Centar** i Muzička produkcija RTS, **...**
nekretnine-beograd.com
More pages from nekretnine-beograd.com

Novi Beograd
**...** Lekuvrer", "Porgi i Bes", "Priče sa zapadne strane"Ó **Kiri te Kanava** debitovala **...** Manifestacija Dani Evrope u Centru "**Sava**" **... Centar** bez vozila do 19 sati **...**
www.novibeograd.info/index.php?strana=vest&id=1372
More pages from novibeograd.info

*Sl. 2 AltaVista rezultati za pretragu "kiri te kanava sava centar"*

Na osnovu ovih činjenica moglo bi se pretpostaviti da su samo autori sajtova *Blic* i *B92* prijavili svoje sajtove na *Google* pretraživač kao dinamične i zahtevali povećanu frekvenciju poseta od strane *Google*-ovog *crawlera*. Pretraživač *AltaVista* se oslanja na *Yahoo*! *Search* tehnologiju i ne pruža mogućnost smanjenja intervala posete.

Zbog svega navedenog, jedan od osnovnih problema koji će biti razmatran u ovom radu jeste omogućavanje korisniku pristup sadržajima na vebu u što kraćem vremensku intervalu. Ovaj problem može da se prevaziđe projektovanjem posebnog sistema koji podržava koncept nadgledanja veba, nad kojim korisnik ima punu kontrolu. Korisnik zadaje URL strane ili veb-sajta koji želi da nadgleda u potrazi za nekom informacijom, kao i interval preuzimanja strana.

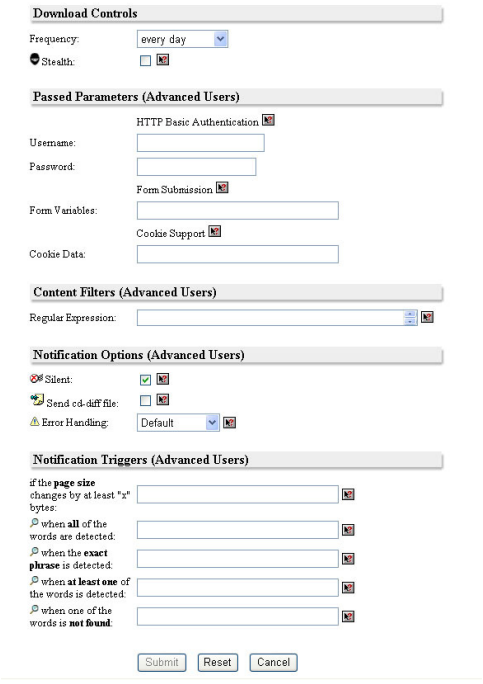
# 3. POSTOJEĆI ALATI

U nastojanjima da se navedeni problemi pretrage informacija prevaziđu, kao i da se korisnicima omogući što kvalitetniji pristup informacijama, u svetu su razvijeni manje ili više uspešni alati za pretragu i nadgledanje veba. Ni jedan od njih ne rešava definisani problem u potpunosti.

Alati specijalizovani za nadgledanje veba uglavnom pružaju slične mogućnosti korisniku, sa malim razlikama koje se uglavnom odnose na korisnički interfejs i načine alarmiranja. Svi alati za nadgledanje veba do kojih je autor ovog teksta došao na sličan način omogućavaju nadgledanje samo pojedinačnih strana, a ne celog veb sajta. Takođe, ni jedan od pomenutih alata ne omogućava postavljane drugih upita, sem upita na osnovu ključnih reči.

## 3.1. *ChangeDetect*

*ChangeDetect* je alat za nadgledanje veb strana dostupan na adresi **http://www.changedetect.com**. Nakon prijavljivanja na sistem *ChangeDetect*, korisnik ima pristup kontrolnom panelu u okviru koga zadaje nadgledanje strane na određenoj adresi. Za svaku stranu koja se nadgleda korisnik može da podesi nekoliko parametara: frekvenciju




*Sl. 3 Panel za podešavanje parametara nadgledanja veb strane alata ChangeDetect*

preuzimanja strane, regularni izraz za filtriranje sadržaja, događaje koji izazivaju alarmiranje korisnika i sl. Alarmiranje korisnika se vrši slanjem elektronske poruke, u okviru koje može biti poslato ili samo obaveštenje da je do promene došlo ili i cela strana sa obeleženim

promenama. Koristeći ovaj alat korisnik je u mogućnosti da bude obavešten o promeni veličine strane u kilobajtima ili o pojavljivanju jedne ili više reči na strani.

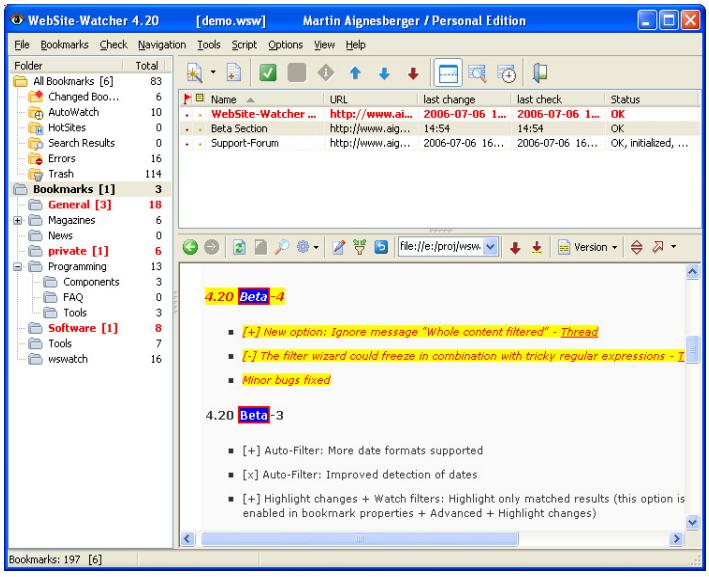
*ChangeDetect*, iako omogućava nadgledanje veb strana, ipak ima ograničenja i nedostatke kada se uzme u obzir problem pronalaženja određene informacije. Neki od nedostataka koje je autor ovog teksta uočio su:

- Najmanji interval nadgledanja strane je 12 sati. Izmene nastale na strani u roku manjem od 12 sati ne moraju da budu primećene od strane sistema.
- Sistem je dizajniran prvenstveno za obaveštavanje o promeni na određenoj strani. Ne postoji mogućnost postavljana komplikovanih upita, sem upita na osnovu ključnih reči sa upotrebom bulovskih operatora.
- Ne postoji mogućnost nadgledanja celog veb sajta. Iako u okviru kontrolnog panela korisnik može da zada više strana koje će se nadgledati, ipak u okviru svakog procesa nadgledanja sistem nadgleda samo stranu čija je URL adresa navedena.

## 3.2. *WebSite-Watcher*

*WebSite-Watcher* je softver dizajniran da radi kao klijent na korisnikovom računaru. Namenjen je za praćenje promena na neograničenom broju veb strana. Može se preuzeti sa adrese **http://www.aignes.com**.

Pomoću ovog alata moguće je nadgledanje svih formata sa tekstualnim sadržajima, pa čak i strana zaštićenih lozinkom. Takođe, ovaj alat omogućava i nadgledanje promena nad binarnim datotekama u smislu promene veličine ili datuma kada je datoteka promenjena.




*Sl. 4 Glavni prozor programa WebSite-Watcher*

Međutim, ni ovaj alat ne podržava postavljanje složenijih upita. *WebSite-Watcher* koristi regularne izraze, ali samo za ograničavanje dela sadržaja strane koji se nadgleda. Korisniku je omogućeno samo nadgledanje pojavljivanja neke od ključnih reči ili izraza.

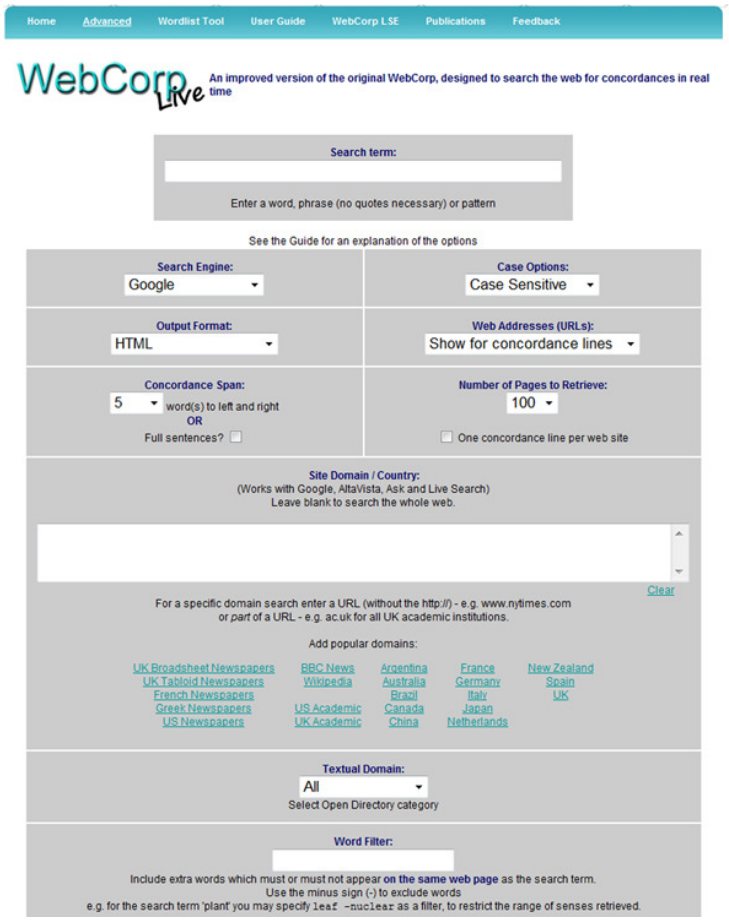
## 3.3. PageHammer

PageHammer je još jedan od alata za nadgledanje veb strana, dostupan na adresi **http://www.pagehammer.com**. Slično kao i prethodno navedeni alati, PageHammer omogućava nadgledanje pojedinačnih veb strana i alaramiranje korisnika ukoliko se pojavi neka od ključnih reči ili izraza. Najveća frekvencija provere strane je dva puta dnevno. Kao i pretodni alati, i on omogućava nadgledanje veb strana, a ne čitavih veb sajtova. Međutim, *PageHammer* omogućava korisniku da kreira tzv. kanale za nadgledanje. U okviru ovih kanala korisnik može da zada nekoliko veb strana. Parametri nadgledanja se odnose na ceo kanal, pa je time pojednostavljeno nadgledanje više strana od jednom.

## 3.4. WebCorp

*WebCorp* je dizajniran od strane Odeljenja za istraživanje i razvoj za engleski jezik na Univerzitetu u Birmingemu. Namenjen je prvenstveno lingvistima i korisnicima koji se bave jezikom u smislu istraživanja i izučavanja, ali ga je moguće koristiti i za pretragu informacija.

Razlikuje se od standardnih pretraživača po tome što kao rezultat pretrage vraća lingvističke podatke, tj. listu konkordanci sa kontekstom traženog pojma ili izraza.




*Sl. 5 Obrazac za naprednu pretragu alata WebCorp*

Korisniku je omogućeno da traži određenu reč ili izraz, ali i obrazac. Obrasci mogu biti formirani kombinacijom operatora *, koji u traženom izrazu zamenjuje bilo koju sekvencu karaktera i uglastom zagradom i ILI operatorom (|). Na taj način *WebCorp* omogućava korisniku da postavi komplikovanije upite, kao na primer "*[uzgoj|uzgajanje]*[nane|mente]*". Ovakav obrazac bi odgovarao izrazima "uzgoj nane", "uzgajanje pitome mente" i sl. Od naprednih opcija pretrage, *WebCorp* omogućava ograničenje pretrage na određeni domen i dodatno filtriranje rezultata odabirom reči koje ti dokumenti moraju ili ne smeju da sadrže. Kao polaznu tačku svoje pretrage ovaj alat koristi rezultate nekog od postojećih pretraživača koje je korisnik odabrao, a zatim ih dodatno obrađuje kako bi pronašao konkordance traženog izraza ili obrasca.

Iako ovaj alat predstavlja značajno poboljšanje u smislu mogućnosti opisivanja kompleksnijih događaja, on ipak ne rešava definisani problem, s obzirom da se oslanja na postojeće pretraživače, pa na taj način zadržava problem velikog broja nedostupnih informacija. Takođe, ovaj alat ne omogućava nadgledanje veb strane.

# 4. TEORIJSKE OSNOVE I SUŠTINA PREDLOŽENOG REŠENJA

## 4.1. Konačni automati i transduktori

### *4.1.1. Konačni automati*

*Definicija 1*

Konačni (nedeterministički) automat nad konačnom azbukom $\Sigma$ se sastoji od:

- konačnog skupa $Q$ – skupa stanja
- skupa $I \subset Q$ – skupa početnih, inicijalnih stanja
- skupa $F \subset Q$ – skupa završnih, finalnih stanja
- skupa $\Delta \subset Q \times \Sigma \times Q$ – skupa relacija prelaza

Konačni automat se zapisuje kao uređena petorka $A = (\Sigma, Q, I, F, \Delta)$. Element relacije prelaza $\Delta$ naziva se *luk*. Ako je luk $l = (p, a, q) \in \Delta$, onda je slovo $a \in \Sigma$ *etiketa* tog luka.

*Definicija 2*

Pod *izračunavanjem c* dužine *n* u automatu *A* podrazumeva se niz lukova $l_1, \ldots l_n$, gde $l_i = (p_i, a_i, q_i) \in \Delta$, tako da je $q_i = p_{i+1}$ za $i \in [1, n-1] \subset N$. Pod *etiketom* izračunavanja *c*, u oznaci $\|c\|$ se podrazumeva niska $a_1 \ldots a_n$ sastavljena od etiketa lukova izračunavanja *c*. Ako je niska $w = \|c\|$ etiketa izračunavanja *c*, onda se to zapisuje na sledeći način:

$$c : p_1 \xrightarrow{w} q_n \text{ ili } c : p_1 \longrightarrow q_n .$$

*Definicija 3*

Za izračunavanje $c : p \longrightarrow q$ se kaže da je *uspešno* ako važi da je $p \in I$ *i* $q \in F$. Za reč $\omega$ se kaže da je *prepoznata* (ili *prihvaćena*) automatom *A* ako je ta reč etiketa nekog uspešnog izračunavanja. *Jezik prepoznat (prihvaćen) automatom A* je skup svih reči prepoznatih automatom *A*:

$$L(A) = \left\{ \omega \in \Sigma^* \middle| \exists c : i \to f, i \in I, f \in F, \omega = \|c\| \right\}.$$

Pojam izračunavanja se može posmatrati i na drugi način, kao proširenje relacije prelaza $\Delta$ sa slova na reči. Tako *proširena relacija prelaza*, u oznaci $\Delta^*$, opisana je na sledeći način:

$$\Delta^* \subset Q \times \Sigma^* \times Q$$

uz uslove:

- za svako $q \in Q$, $(q, \varepsilon, q) \in \Delta^*$, gde je $\varepsilon$ prazna reč;

- ako je $\omega=a_1a_2...a_n$ $(a_i\in\Sigma,\ n\geq 1)$ i ako postoji $n+1$ stanje $q_0$, $q_1...q_n$ takvo da za svako $i\in N: 1 \leq i \leq n$, važi da je luk $(q_{i-1}, a_i, q_i) \in \Delta$, tada je $(q_0, \omega, q_n) \in \Delta^*$, a $\omega$ je *etiketa* puta u automatu koja povezuje stanje $q_0$ sa stanjem $q_n$.

  Jezik prepoznat konačnim automatom $A$ je tada

  $$L(A) = \left\{\omega \in \Sigma^* \middle| \exists i \in I, \exists f \in F, (i, \omega, f) \in \Delta^*\right\}.$$

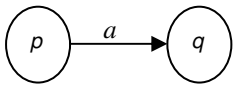
*Definicija 4*

Podskup $X \subseteq \Sigma^*$ je prepoznatljiv ako postoji automat $A$ nad azbukom $\Sigma$ takav da je $X=L(A)$. Familija svih prepoznatljivih podskupova u $\Sigma^*$ obeležava se sa $Prep(\Sigma^*)$.

*Teorema 1*

Svaki konačan podskup $X \subseteq \Sigma^*$ je prepoznatljiv jezik.

Konačni automati se najčešće predstavljaju tzv. grafom prelaza, iako mogu biti predstavljeni i na druge načine (na primer, matricom prelaza, videti u [6] ). Reprezentacija automata pomoću grafa se obično naziva *dijagram stanja*. U njoj su stanja automata predstavljena čvorovima grafa, a luk automata $(p, a, q)$ je predstavljen lukom iz čvora $p$ ka čvoru $q$ koji je obeležen etiketom $a$ (slika 6.).




*Sl. 6 Prelazak iz stanja p u stanje q nakon što je pročitan ulazni simbol a*

Izračunavanje je put u grafu, a obeležja lukova koji čine jedan put u grafu su slova etikete izračunavanja.

U grafičkom prikazu automata, početna i završna stanja (elementi skupova $I$ i $F$) se obeležavaju na poseban način. Početno stanje je obeleženo strelicom koja pokazuje na njega, dok je završno stanje dvostruko zaokruženo.

### *4.1.2. Deterministički konačni automati (DKA)*

*Definicija 5*

Stanje $q \in Q$ automata $A = (\Sigma, Q, I, F, \Delta)$ je *dostupno* ako postoji izračunavanje $c: i \to q$, gde je $i \in I$. Stanje $q \in Q$ je sudostupno ako postoji izračunavanje $c: q \to f$, gde je $f \in F$. Ako su sva stanja jednog automata dostupna, kažemo da je automat *dostupan*, a ako su sva stanja automata dostupna i sudostupna, kažemo da je automat *skresan*.

*Potkresivanje automata A* jeste postupak određivanja automata $B$ koji je skresan, pri čemu važi da je $L(A)=L(B)$. Automat $B$ se sastoji samo od onih stanja automata $A$ koja su i dostupna i sudostupna.

*Definicija 6*

Konačni automat $A = (\Sigma, Q, I, F, \Delta)$ je *deterministički* ako skup početnih stanja $I$ ima tačno jedan element i ako važi

$$(p, a, q), (p, a, r) \in \Delta \Rightarrow q=r.$$

Deterministički konačni automat se skraćeno obeležava sa DKA.

Imajući u vidu definiciju determinističkog konačnog automata, relacija prelaska se u ovom slučaju svodi na parcijalno preslikavanje $\delta: Q \times \Sigma \rightarrow Q$ koje nazivamo *funkcija prelaza* determinističkog konačnog automata. Ova funkcija se dalje proširuje u funkciju $\delta^*$ nad rečima iz $\Sigma^*$, pa se vrlo često jezik prepoznat determinističkim konačnim automatom A definiše kao:

$$L(A) = \left\{\omega \in \Sigma^* \middle| \delta^*(i, \omega) \in F\right\}$$

*Definicija 6*

Konačni automat $A = (\Sigma, Q, I, F, \Delta)$ je *potpun* (ili *kompletan*) ako za svako stanje $p \in Q$ i svako slovo $a \in \Sigma$, postoji bar jedno stanje $q \in Q$ takvo da je $(p, a, q) \in \Delta$.

*Teorema 2*

Za svaki konačan automat *A* postoji potpun deterministički konačni automat *B* takav da je *L(A)=L(B).*

Iako deterministički i nedeterministički konačni automat prepoznaju i generišu potpuno isti jezik, u izvesnim situacijama postoji značajna razlika u tome koji od ova dva automata se koristi. Prilikom upotrebe automata za generisanje jezika, svejedno je da li će biti korišćen deterministički ili nedeterministički automat. Međutim, ukoliko se automat koristi za prepoznavanje jezika, tada upotreba nedeterminističkog automata može da uzrokuje da ulazna reč ne bude prepoznata, iako pripada jeziku automata.

### *4.1.3. Konačni transduktori*

Za razliku od konačnih automata, koji definišu formalni jezik definisanjem skupa niski karaktera (reči), konačni transduktori definišu relacije između dva skupa niski karaktera. Drugim rečima, konačni automat za datu nisku $\omega$ utvrđuje da li pripada jeziku opisanom tim automatom ili ne. Konačni transduktor, sa druge strane, transformiše zadatu nisku u drugu nisku nad istom ili nekom drugom azbukom. Zbog toga se konačni transduktori često nazivaju i konačnim automatima prevodiocima.

*Definicija 7*

*Konačni transduktor* je uređena šestorka $\tau = (\Sigma_1, \Sigma_2, Q, i, F, \Delta)$, gde su

- $\Sigma_1$ i $\Sigma_2$ *ulazna* i *izlazna* konačna azbuka
- $Q$ konačan *skup stanja*
- $i \in Q$ *početno stanje*
- $F \subset Q$ skup *završnih stanja*
- $\Delta \subset Q \times \Sigma_1 \times \Sigma_2 \times Q$ relacija *prelaza*, čije elemente nazivamo *lukovima*.

Dakle, konačni transduktori mogu biti posmatrani kao mašine koje prepoznaju (čitaju) jednu nisku karaktera (reč), a generišu neku drugu.

Svakom konačnom transduktoru se može pridružiti konačni automat tako što se parovi simbola na lukovima posmatraju kao simboli konačnog automata.

*Definicija 8*

Neka je $\tau = (\Sigma_1, \Sigma_2, Q, i, F, \Delta)$ konačni transduktor. Tada je njegov pripadajući konačni automat $A = (\Sigma, Q, I, F, \Delta_1)$ definisan se:

$\Sigma = \Sigma_1 \times \Sigma_2$

$(p, (a, b), q) \in \Delta_1 \Leftrightarrow (p, a, b, q) \in \Delta$.

Alternativna definicija konačnog transduktora se sastoji u zameni skupa prelaza $\Delta$ dvema funkcijama: *funkcijom prelaza* $\delta$: $Q \times \Sigma_1 \to Q$ i *emisionom funkcijom* $\varepsilon$: $Q \times \Sigma_1 \to \Sigma_2^*$ tako da:

$$\delta(p,a) = \left\{ q \in Q \middle| \exists (p,a,b,q) \in \Delta \right\}$$

$$\varepsilon(p,a) = \left\{ b \in \Sigma_2^* \middle| \exists (p,a,b,q) \in \Delta \right\}$$

Takođe, konačni transduktor se može interpretirati i kao relacija nad niskama. Prema ovoj interpretaciji, skup lukova, kao i funkcija prelaza i emisiona funkcija, se primenjuju na niske umesto na pojedinačne simbole.

*Definicija 9*

Proširenu skup lukova $\tilde{\Delta} \subseteq Q \times \Sigma_1^* \times \Sigma_2^* \times Q$ je najmanji podskup takav da:

- za svako $q \in Q$, $(q, \varepsilon, \varepsilon, q) \in \tilde{\Delta}$
- za svako $\omega_1 \in \Sigma_1^*$, $\omega_2 \in \Sigma_2^*$, važi
  $(q1, \omega_1, \omega_2, q2) \in \tilde{\Delta}$ i $(q2, a, b, q3) \in \Delta \Rightarrow (q1, \omega_1 a, \omega_2 b, q3) \in \tilde{\Delta}$.

Ovakva definicija dopušta da se konačnom transduktoru pridruži relacija $L(\tau)$ na sledeći način:

$$L(\tau) = \left\{ (\omega_1, \omega_2) \in \Sigma_1^* \times \Sigma_2^* \middle| \exists (i, \omega_1, \omega_2, q) \in \tilde{\Delta} \wedge q \in F \right\} \subseteq \Sigma_1^* \times \Sigma_2^*$$

$L(\tau)$ opisuje *prevođenje* ili *transdukciju* koja se vrši transduktorom $\tau$.

Osnovna osobina transduktora, koja u mnogome određuje njihovu upotrebu, jeste da produkuju neki izlaz.

### *4.1.4. RTN - Recursive transition network*

U praksi, primena konačnih automata i transduktora dovodi do velikih problema zahvaljujući tome što automati i transduktori nisu modularni i mogu biti veoma složeni i teški za održavanje. Na primer, ukoliko konačnim mašinama pokušamo da opišemo sintaksu nekog jezika, odgovarajući graf bi bio veoma nepregledan, a pronalaženje svih informacija vezanih za, recimo, imeničke fraze vremenski zahtevno i nepraktično.

Zbog toga se u praksi umesto jednog ogromnog grafa konačnog automata ili transduktora često koristi kolekcija podgrafova (ili podmreža). Tako bi u primeru opisivanja sintakse jezika postojali po jedan graf za svaku kategoriju (rečenica, imenička fraza, glagolska fraza i sl.). Ovakav način grupisanja grafova u kolekcije (mreže) formalno je definisan teorijom rekurzivnih tranzicionih mreža (*Recursive transition networks – RTN*).

RTN predstavljaju proširenje konteksno slobodnih gramatika (gramatika tipa 2 po hijererhiji Čomskog [6]). Lukovi u RTN su označeni odgovarajućim gamatikama, dok su stanja označena proizvoljno. Ne postoji konekcija između čvorova RTN i gramatike kojoj je RTN ekvivalentan.

## 4.2. Značaj i primena konačnih automata i transduktora u računarskoj lingvistici

Konačne mašine se koriste u velikom broju oblasti računarske lingvistike. Njihova upotreba je opravdana i sa stanovišta lingvistike i sa stanovišta računarstva.

Sa stanovišta lingvistike, konačni automati su pogodni kao način za jednostavno opisivanje relevantnih lokalnih fenomena koji se pojavljuju u empirijskim proučavanjima jezika. Iako formalni jezik definisan automatom ne mora da bude ni u kakvoj vezi sa prirodnim jezikom (formalni jezik može biti korišćen za modeliranje stanja, npr. aparata za kafu), ipak se često koristi za modeliranje nekog dela prirodnog jezika (fonologije, morfologije, sintakse i dr.).

Sa stanovišta računarstva, korišćenje konačnih mašina je uglavnom motivisano vremenskom i prostornom efikasnošću. Vremenska efikasnost se uglavnom postiže upotrebom determinističkih automata [6] i [7]. Izlaz determinističkih mašina zavisi, uglavnom linearno, samo od veličine ulaza i zbog toga mogu biti smatrani optimalnim. Prostorna efikasnost se postiže minimizacijom determinističkih automata [11].

Primeri adekvatne reprezentacije različitih lingvističkih fenomena pomoću konačnih automata su dati u [14]. Korišćenje formalnih jezika i konačnih automata obično vodi kompaktnim reprezentacijama leksičkih pravila ili idioma i klišea.

Grafički prikaz automata omogućava vizuelizaciju i jednostavno modifikovanje automata, koje opet pomaže lingvistima u korigovanju i kompletiranju gramatike. Parsiranje kontekstno slobodnih gramatika može biti rešeno korišćenjem konačnih automata [8]. Šta više, mehanizmi na koje se oslanja većina metoda za parsiranje su usko povezani sa automatima.

Konačni transduktori se u računarskoj lingvistici koriste za morfološko parsiranje, opisivanje pravopisnih pravila, opisivanje inflekcionih pravila i sl. Detaljniji pregled teorijskih i praktičnih primena konačnih transduktora u obradi prirodnih jezika je dat u [7],[9] i [10].

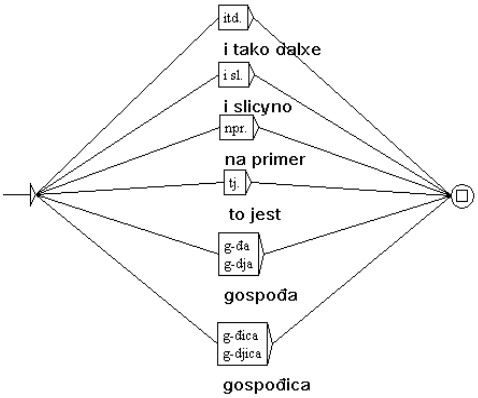
U [15] je opisan način na koji konačni transduktori pružaju prirodan i unificiran metod parsiranja rečenica u kojima se kombinuju slobodne komponente i idiomi.

Konačni transduktori su veoma adekvatni kao modeli koji mogu biti automatski proizvedeni tokom treniranja u procesu mašinskog prevođenja.U [16] je opisana tehnika kojom je moguće automatski kreirati konačne transduktore.

U [17] je prikazan način kombinovanja konačnih transduktora u serije, nazvane kaskade, pomoću kojih se transformiše neki tekst i upotreba ovih kaskada na ekstrakciju imenovanih entiteta iz novinskih članaka.

*Primer 5.*

Na slici koja sledi (slika 7) prikazana je upotreba konačnih transduktora za obradu skraćenica u srpskom jeziku. Prelaz po jednom luku transduktora prepoznaje neki od skraćenih oblika i kao izlaz produkuje izraz ili reč koja odgovara prepoznatoj skraćenici.




*Sl. 7 Konačni transduktor za prepoznavanje skraćenica*

U okviru sistema *WebMonitoring*, koji će u ovom radu biti predstavljen kao sistem za nadgledanje veba koji rešava definisani problem, konačni transduktori se koriste kako bi bilo omogućeno opisivanje kompleksnih događaja prilikom postavljanja upita, ali i za pripremu teksta koji se pretražuje.

## 4.3. Koncept nadgledanja veba

Koncept nadgledanja veba je nastao iz potrebe za automatizacijom određenih akcija koje korisnik preduzima sa ciljem da bude obavešten o promenama nastalim na određenoj veb strani ili sajtu. U praksi, česte su situacije kada korisnik posećuje veb sajt očekujući da se na toj adresi desio određeni događaj, a da pri tome nije zainteresovan za ostatak sadržaja koji se na njemu nalazi. Primeri takvih događaja su pojavljivanje rezultata nekog konkursa, pristigla elektronska poruka određenog sadržaja ili od određene osobe, pojavljivanje vesti o određenoj temi i sl. U takvim slučajevima, korisnik redovno, dinamikom za koju on smatra da je

optimalna ili koja je u datom trenutku moguća, posećuje veb sajt od interesa i pregleda njegov sadržaj u potrazi za događajem koji iščekuje. Koncept nadgledanja veba i programski sistemi koji ga primenjuju, automatizuju ovaj proces traženja informacije, simulirajući akcije koje bi sam korisnik preduzeo. Pri tome korisnik ima punu kontrolu i definiše parametre koji određuju proces nadgledanja.

Uspešan programski sistem za nadgledanje veba trebalo bi da obuhvati i integriše sledeće paradigme:

- programski interfejs koji omogućava korisniku kontrolisanje procesa nadgledanja zadavanjem osnovnih parametara, kao što su URL veb sajta ili strane koja se nadgleda, vremenska dinamika kojom se preuzimaju sadržaji sa navedene adrese, dubina (nivo) sajta do koga se vrši preuzimanje, način alarmiranja korisnika u slučaju da se određeni događaj desio i sl.
- model postavljanja upita, tj. način na koji korisnik zadaje i definiše događaj koji je od interesa
- sistem za preuzimanje strana sa Interneta; u slučaju kada je potrebno nadgledanje dela ili celog veb sajta, neophodno je da ovaj sistem bude dizajniran u skladu sa osnovnim principima *web crawling*-a [18]
- sistem za analizu preuzetih sadržaja, koji podrazumeva korišćenje adekvatnih algoritama za obradu teksta, u zavisnosti od modela postavljanja upita
- sistem za alarmiranje korisnika u slučaju da se određeni događaj desio

Rešenje predloženo od strane autora, sistem *WebMonitoring*, podržava koncept nadgledanja veba i, u odnosu na postojeća rešenja, predstavlja dve novine. Prva od njih je korišćenje konačnih transduktora, predstavljenih grafovima, za definisanje događaja koje korisnik iščekuje. Na ovaj način je omogućeno nadgledanje veoma kompleksnih događaja, koje bi inače bilo teško ili nemoguće opisati upitima na bazi ključnih reči. Druga novina je omogućavanje nadgledanje čitavog veb sajta, a ne samo pojedinačnih veb strana, integrisanjem sistema za preuzimanje veb strana (*web crawling*).

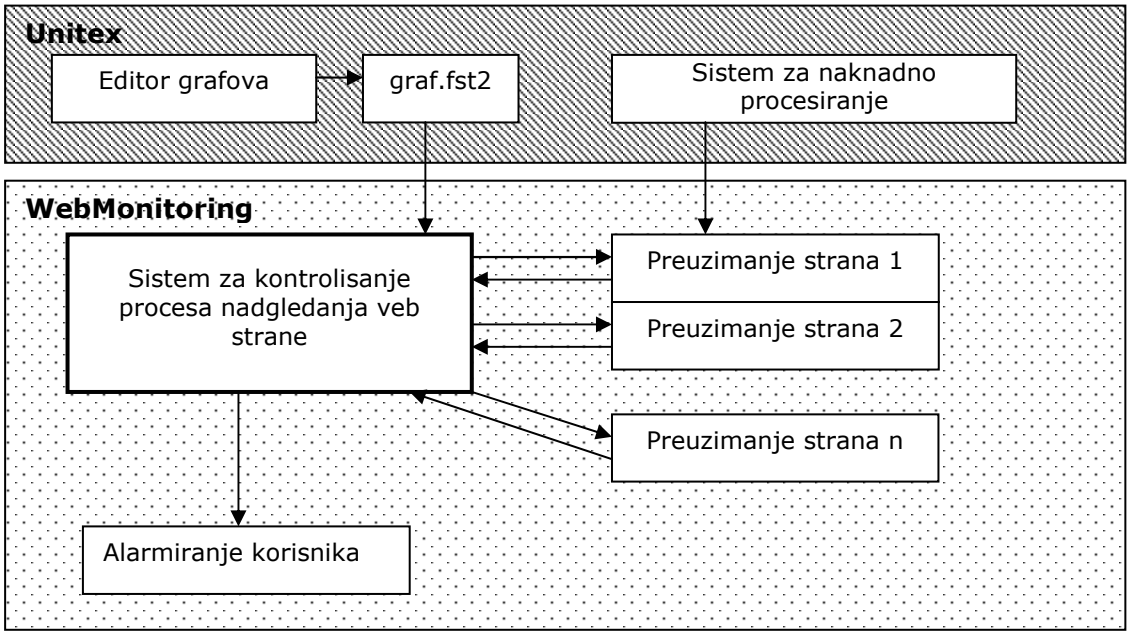
# 5. *WEBMONITORING* - OPIS PROGRAMSKOG SISTEMA ZA NADGLEDANJE VEBA

Kao rešenje prethodno definisanog problema nepostojanja adekvatnih pretraživača koji bi korisniku omogućili pristup svežim informacijama u kratkom vremenskom periodu od trenutka njihovog pojavljivanja na vebu, razvijen je programski sistem nazvan *WebMonitoring*.

Prilikom projektovanja i izrade ovog sistema uzeta je u obzir potreba korisnika da ciljano pristupa određenim sadržajima na Internetu i da pri tome ima punu kontrolu nad procesima koji se tom prilikom odvijaju. Takođe, korisniku je omogućeno da definiše složene događaje čije ga pojavljivanje interesuje, na relativno jednostavan način. Korišćenje grafova koji na intuitivan i pregledan način opisuju pojedine jezičke fenomene s jedne strane pojednostavljuje korisniku postupak postavljanja složenih upita, a sa druge strane korisniku pruža mogućnost kakvu nema prilikom korišćenja klasičnih pretraživača. Uz sve to, korisnik sam bira način na koji će biti obavešten o realizovanom događaju.

Programski sistem *WebMonitoring* je napisan u programskom jeziku Java i sastoji se od nekoliko međusobno integrisanih programskih celina:

- sistem za postavljanje upita – koristi se programski sistem *Unitex* [19]
- sistem za kontrolisanje procesa nadgledanja strana
- sistem za preuzimanje strana sa veba
- sistem za naknadno procesiranje teksta – koriste se programi sistema *Unitex*
- sistem za alarmiranje
- korisnički interfejs




*Sl. 8 Programski sistem WebMonitoring*

Korisnik, pomoću sistema *Unitex*, kreira graf koji opisuje događaj koji je od interesa. Ovaj graf se kao ulazni podatak prosleđuje sistemu *WebMonitoring*. Korisnik zadaje URL strane ili veb sajta koji se nadgleda, podešava parametre kao što su dinamika posećivanja strana, dubina do koje se vrši preuzimanje strana u odnosu na polazni URL, način alarmiranja i slično. Nakon toga sistem pokreće posebne programske niti, za svaki proces nadgledanja po jednu. Svaka programska nit preuzima strane sa Interneta, snima ih lokalno, a zatim poziva odgovarajuće programe sistema *Unitex* koji vrše obradu teksta i njegovu pretragu. Ukoliko dođe do događaja koji je korisnik opisao grafom, sistem obaveštava korisnika o tome na jedan ili više odabranih načina. Svi detalji ovog procesa, kao i samog sistema i njegovih programskih celina biće detaljno objašnjeni u narednim poglavljima.

## 5.1. Programski sistem *Unitex*

*Unitex* je kolekcija programa razvijenih za analizu teksta na prirodnim jezicima korišćenjem lingvističih resursa i alata. Resursi se sastoje od elektronskih rečnika i gramatika, posebno razvijenih za pojedine jezike. Ovaj sistem je *open source*. Dizajniran je tako da bude prenosiv, tj. da je moguće njegovo izvršavanje na različitim operativnim sistemima, kao što su *Windows*, *Linux*, *MacOS* i drugi. Programi u okviru *Unitex*-a su pisani na programskim jezicima C/C++, dok je grafički korisnički interfejs pisan u programskom jeziku JAVA. *Unitex* je projektovan tako da može da podržava različite prirodne jezike.

Sama upotreba *Unitex* sistema počinje upravo odabirom jezika koji će se koristiti. Ovaj odabir jezika utiče na dve bitne stvari u okviru dalje obrade teksta, a to su elektronski rečnici koji će se koristiti i gramatike za tokenizaciju i normalizaciju tekstova u fazi predprocesiranja.

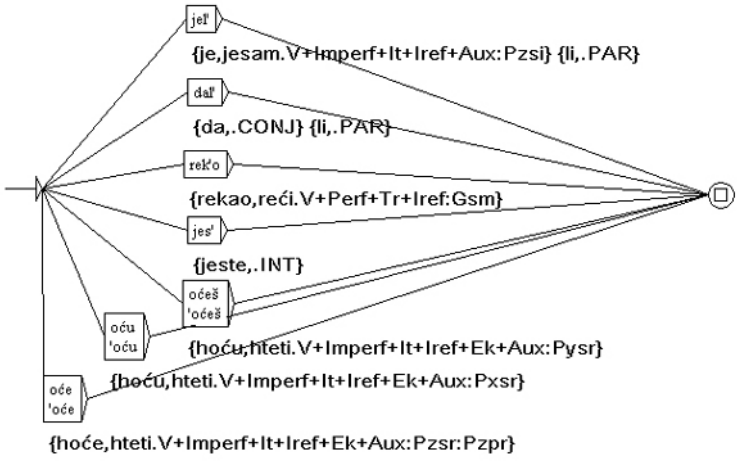
Elektronski rečnici sadrže reči i složenice nekog jezika zajedno sa njihovim lemama i skupom gramatičkih (semantičkim i flektivnim) kodovima. Upravo korišćenje ovakvih rečnika predstavlja glavnu prednost sistem *Unitex* nad drugim sistemima za pretragu i analizu teksta, s obzirom da informacije koje se nalaze u njima omogućavaju definisanje čitavih klasa reči i izraza korišćenjem veoma jednostavnih obrazaca.

Gramatike koje se koriste unutar *Unitex* sistema reprezentuju lingvitičke fenomene na osnovama rekurzivnih mreža prelaza (*Recursive Transition Networks – RTN* – [21], [22]), formalizmu koji predstavlja uopštenje teorije konačnih automata i transduktora. One su predstavljene grafovima koji se veoma jednostavno kreiraju i prilagođavaju od strane korisnika.

Posle odabira jezika, pristupa se obradi samog teksta. Unitex zahteva da tekst koji treba da se obradi bude u *Unicode Little Endian 16bit* (UTF16LE) formatu (više o ovom formatu kodiranja videti u [3]). U okviru *Unitex*-a postoje klase pomoću kojih je moguće izvršiti konverziju teksta, ukoliko format nije adekvatan.

Kada je tekst snimljen u odgovarajućem formatu, pristupa se njegovoj pripremi. Prvo se vrši normalizacija separatora teksta. Standardni separatori su razmak, tabulator i novi red. U tekstu je moguće da se pojavi nekoliko separatora jedan za drugim. S obzirom da ovakav niz separatora nije značajan za lingvističku analizu, on biva zamenjen i to novim redom ukoliko je barem jedan od separatora u nizu oznaka za novi red, ili razmakom ukoliko se u nizu

separatora ne pojavljuje oznaka za novi red. Moguće je izvršiti i normalizaciju teksta u drugom smislu, po nekom proizvoljnom pravilu. Na slici br. 6 prikazan je primer transduktora koji odgovara gramatici za normalizaciju teksta uklanjanjem elizija.

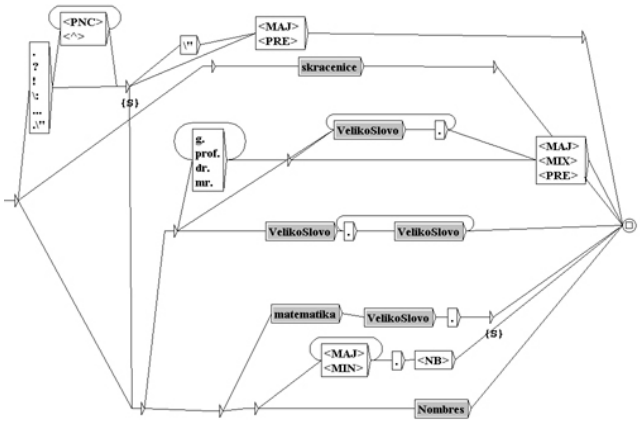



*Sl. 6 Transduktor za normalizaciju elizija u srpskom jeziku*

Zatim se pristupa tokenizaciji teksta. Tokenizacija teksta je proces u kome se tekst razdvaja na leksičke jedinice. Separatori leksičkih jedinica mogu biti različiti u različitim jezicima. Zbog toga *Unitex* vrši tokenizaciju na način specifičan pojedinom jeziku, pa se na primer za tajlandski jezik tokenizacija vrši karakter po karakter.

.

Na slici br. 7 prikazana je gramatika za razdvajanje na rečenice, koja opisuje u kom kontekstu znakovi interpunkcije treba da budu shvaćeni kao separatori rečenica.




*Sl. 7 Gramatika za razdvajanje na rečenice*

Posle faze tokenizacije kreirana je lista svih tokena koji su se pojavili u tekstu. Takođe, tekst je proširen ubacivanjem separatora rečenica. Na ovako pripremljen tekst dalje se primenjuju elektronski rečnici, tj. određuje se podskup reči iz rečnika koje se pojavljuju u tekstu.

Elektronski rečnici u *Unitex*-u su u DELA formatu i konstruisani su od strane timova lingvista za odgovarajući jezik (za elektronske rečnike srpskog jezika videti [12] i [13]). DELA sintaksa opisuje proste i složene reči pridruživanjem koda koji opisuje flektivna, sintaksna i semantička svojstva reči. Svaka reč ili složena forma je opisana u jednoj liniji elektronskog rečnika, tj. predstavlja jednu stavku u rečniku. U zavisnosti od toga da li se

čuvaju reči u izmenjenom obliku ili samo njihove leme, razlikujemo dve vrste DELA rečnika, DELAF i DELAS.

DELAF format je rečnik oblika reči, u kome jedna stavka (jedan ulaz) izgleda ovako:

`lekara,lekar.N+Hum+Ek:ms2v:ms4v:mp2v`

Navedena linija elektronskog rečnika je formirana od nekoliko delova, sa sledećim značenjem:

`lekara` – izmenjen oblik reči; ova informacija je obavezna

`lekar` – odgovarajuća lema (može da bude izostavljena)

`N+Hum+Ek` – sekvenca kodova koji određuju vrstu reči i njena gramatička svojstva, npr. **N** označava da se radi o imenici (*Noun*), **Hum** da se pojam odnosi na osobu, **Ek** da se radi o ekavskom izgovoru

`ms2v:ms4v:mp2v` – sekvenca kodova koja određuje odnos oblika reči i leme, tj. opisuju njen rod, broj, deklinaciju, konjugaciju i sl. Na primer, kod **ms2v** bi značio da se radi o imenici muškog roda (**m**-masculine), u drugom padežu jednine (**s**-singular), dok slovo **v** označava animalnost (v-živo biće).

DELAS format je veoma sličan formatu DELAF, s tom razlikom što se u ovu vrstu rečnika upisuju samo leme reči. Na primer,

`konj, N4 + Zool`

Prvi gramatički kod (**N4**) određuje koja gramatika će biti korišćena za promenu reči, dok **Zool** označava da se radi o životinji.

Nakon primene rečnika na tekst, *Unitex* sistem kreira tri pripadajuće datoteke, jednu sa prostim rečima koje se pojavljuju u tekstu, drugu sa složenim rečima i treću sa neprepoznatim rečima [23]. Ovako pripremljen tekst moguće je dalje pretraživati regularnim izrazima ili grafovima. Bez obzira na način postavljanja upita (regularni izraz ili graf), *Unitex* omogućava korišćenje tzv. leksičkih maski. Tako bi upit `<pevati.V>` odgovarao svim oblicima glagola čija je lema reč *pevati*. Sem leksičkih maski, korisniku je omogućeno i korišćenje morfoloških filtera. Na primer, filter `<<izam$>>` bi odgovarao svim rečima koje završavaju sa „izam“.

## 5.2. *Unitex* kao sistem za postavljanje upita i njegova integracija u programski sistem *WebMonitoring*

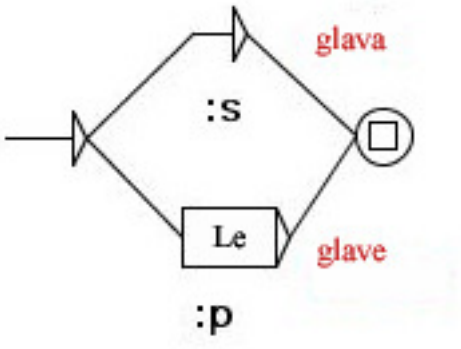
Kombinovanje leksičkih maski i morfoloških filtera, kao i primena leksičkih resursa jezika (elektronskih rečnika i gramatika) na tekst omogućava postavljanje izuzetno komplikovanih upita, i na taj način dozvoljava korisniku da vrši veoma složene pretrage teksta.

S obzirom da je u okviru programskog sistema *Unitex* razvijen grafički korisnički interfejs u okviru koga je korisniku omogućeno kreiranje grafova konačnih transduktora koji odgovaraju različitim jezičkim fenomenima, ova pogodnost je iskorišćena i u okviru sistema *WebMonitoring*.

Sistem *Unitex* koristi različite vrste grafova za različite operacije u obradi teksta. Neki od njih se koriste i u opisivanju događaja čije se pojavljivanje očekuje na nekoj veb strani.

Flektivni transduktori opisuju morfološke varijacije koje odgovaraju pojedinim klasama reči dodeljujući flektivne kodove svakoj varijanti. Putanje ovih transduktora opisuju modifikacije koje treba primeniti na kanonski oblik reči, nakon kojih se produkuje određeni izlaz koji odgovara izmenjenom obliku reči (slika 8). Putanje mogu da sadrže operatore i slova. Jedini dozvoljeni specijalan simbol jeste prazna reč `<E>`.

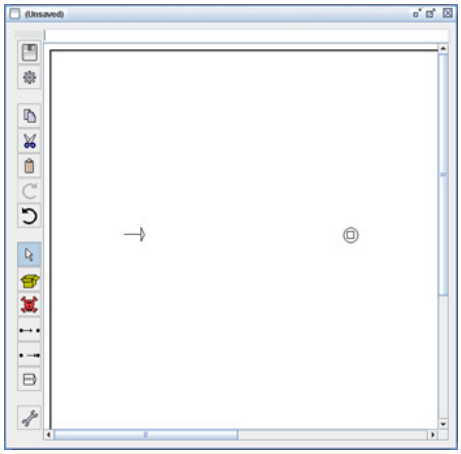
Mogući operatori su predstavljeni slovima L, R, C i D. Operator L uklanja poslednje slovo reči. Operator R ubacuje (vraća) slovo reči u izmenjeni oblik. Operator C duplira slovo u reči (kao na primer u *permitted* ili *hopped*). Operator D briše slovo iz reči i pomera sve što se nalazi desno od slova za jedno mesto. U primeru na slici 8, putanja `Le` znači da se polaznoj lemi reči uklanja poslednje slovo, a zatim dodaje slovo `e` kako bi se formirao plural imenice *glava*.




*Sl. 8 Primer flektivne gramatike*

Sintaksni grafovi, koji u stvari predstavljaju lokalne gramatike, omogućavaju opisivanje sintaksnih obrazaca koji mogu da se traže u nekom tekstu. Ova vrsta grafova ima najveću snagu u procesu pretrage teksta, s obzirom da dopušta i referisanje na informacije iz rečnika. Takođe, moguće je pozivanje podgrafova, korišćenje morfoloških filtera i korišćenje konteksta.

O bilo kojoj vrsti grafova da se radi, *Unitex* obezbeđuje prozor sa radnom površinom i alatima za njihovo kreiranje (slika 9). Više o načinu kreiranja grafova videti u [19].



*Sl. 9 GUI za kreiranje grafa u okviru Unitex-a*

Nakon kreiranja grafa, korisnik ga čuva u datoteci sa ekstenzijom *.grf*. Ova datoteka je u stvari tekstualna datoteka koja sadrži informacije o vizuelnoj prezentaciji skupa stanja i funkcije prelaza grafa, uključujući font, boju slova, boju pozadine i slično. Ovaj format podataka je odgovarajući dokle god korisnik radi na kreiranju grafa, poziva podgrafove i slično. Jednom kada je taj posao obavljen, i kada graf treba da posluži kao ulazni podatak za dalju pretragu teksta, neophodno je konvertovati ga u pogodniji format. *Unitex* za tu svrhu koristi format *.fst2*.

Datoteka u *.fst2* formatu je tekstualna datoteka koja opisuje jedan ili više grafova, ali samo njihove skupove stanja i funkcije prelaza, a ne i vizuelne elemente.

Sledi primer jedne datoteke u *.fst2* formatu.

*Primer 6.*

```
0000000002¶
-1 Replace¶
: -2 1 ¶
t ¶
f ¶
-2 skracenice¶
: 1 6 4 5 5 4 6 3 10 2 13 1 ¶
: 3 7 ¶
: 7 8 ¶
: 7 9 ¶
: 3 7 ¶
: 3 7 ¶
: 2 10 ¶
t ¶
: 11 7 12 7 ¶
: 8 7 9 7 ¶
: 3 7 ¶
f ¶
%<E>¶
%i/i slicyno¶
%sl/<E>¶
%./<E>¶
%npr/na primer¶
%tj/to jest¶
%g/gospođa¶
%-/<E>¶
%đa/<E>¶
%dja/<E>¶
%g/gospođica¶
%đica/<E>¶
%djica/<E>¶
%itd/i tako dalxe¶
f¶
```

Prva linija predstavlja broj grafova koji su kodirani u datoteci. Početak svakog od njih je označen linijom koja sadrži redni broj grafa i njegovo ime. U prethodnom primeru to su linije `-1 Replace` i `-2 skracenice`. Linije koje slede iza njih opisuju stanja grafa. Ukoliko je stanje finalno, linija počinje karakterom `t`, a ukoliko nije, karakterom `:`.

Za svako stanje, lista prelaza je predstavljena sekvencom parova celobrojnih vrednosti. Prva celobrojna vrednost predstavlja oznaku ili podgraf koji odgovara prelazu. Oznake su numerisane počevši od 0. Podgrafovi su predstavljeni negativnim vrednostima. Druga celobrojna vrednost predstavlja broj rezultujućeg stanja nakon prelaza. U svakom grafu stanja su numerisana počevši od broja 0. Po konvenciji, nulto stanje je početno stanje. Svaka linija koja predstavlja definiciju prelaza završava se razmakom. Kraj svakog grafa označen je linijom `f`.

Oznake su definisane nakon poslednjeg grafa. Ukoliko oznaka sadrži i izlaz transduktora, ulazna i izlazna sekvenca su razdvojene znakom `/` (npr. `the/DET`).

Primer koji sledi predstavlja datoteku u *.fst2* formatu koja odgovara grafu transduktora za normalizaciju elizija, a čiji je podgraf graf prikazan na slici 6.

*Primer 7.*

```
0000000002
-1 Norm
: -2 1
t
f
-2 elizije
: 1 8 3 7 4 6 6 5 7 2 8 4 10 2 11 3 13 2 14 1
: 15 2
t
: 12 2
: 9 2
: 2 2
: 2 9
: 2 2
: 2 2
: 5 2
f
%<E>
%jel/{je,jesam.V+Imperf+It+Iref+Aux:Pzsi} {li,.PAR}
%'/<E>
%dal/{da,.CONJ} {li,.PAR}
%rek/{rekao,reći.V+Perf+Tr+Iref:Gsm}
%o/<E>
%jes/{jeste,.INT}
%oće/{hoće,hteti.V+Imperf+It+Iref+Ek+Aux:Pzsr:Pzpr}
%'/{hoće,hteti.V+Imperf+It+Iref+Ek+Aux:Pzsr:Pzpr}
%oće/<E>
%oću/{hoću,hteti.V+Imperf+It+Iref+Ek+Aux:Pxsr}
%'/{hoću,hteti.V+Imperf+It+Iref+Ek+Aux:Pxsr}
%oću/<E>
%oćeš/{hoću,hteti.V+Imperf+It+Iref+Ek+Aux:Pysr}
%'/{hoću,hteti.V+Imperf+It+Iref+Ek+Aux:Pysr}
%oćeš/<E>
f
```

Dakle, u okviru rada sa *Unitex* sistemom, svaki graf je neophodno kompilirati u datoteku formata *.fst2*, kako bi bilo moguće koristiti grafove pomoću *Unitex*-ovih programa. Ovaj format zadržava strukturu sa podgrafovima unutar gramatika, te se razlikuje od konačnih transduktora. *Unitex*-ov program `Flatten` omogućava konverziju FST2 gramatike u konačni transduktor kad god je to moguće ili konstrukciju aproksimativnog transduktora, kada to nije (videti [19]). Naime, prilikom kompiliranja dolazi do zamene svakog pojedinačnog poziva nekog podgrafa samim podgrafom. Ova zamena se vrši do određene dubine. Pozivi podgrafova koji se nalaze nakon ove vrednosti bivaju zamenjeni praznom reči, i u tom slučaju rezultat kompiliranja je konačni transduktor, ali nije ekvivalentan polaznom grafu. Moguća je i opcija da se pozivi ovih grafova zadrže, kojom prilikom je zadržana ekvivalentnost, međutim rezultat kompiliranja nije konačni transduktor.

Sa stanovišta korisnika sistema *WebMonitoring*, sistem *Unitex* i njegov interfejs za kreiranje grafova predstavljaju način postavljanja upita, tj. opisivanja događaja o kojima korisnik želi da bude obavešten. Koristeći sistem Unitex, korisnik kreira graf koji opisuje događaj od interesa, a nakon toga ga i kompilira u *.fst2* format. Ovakva datoteka sadrži sve potrebne informacije o događaju koji se iščekuje i kao takva predstavlja ulazni podatak za sistem *WebMonitoring*.

Iako je *Unitex* projektovan tako da podržava mnoge svetske jezike, u eksperimentalnoj verziji sistema *WebMonitoring* podrazumevano je da se koristi srpski jezik. Zbog toga se u procesu obrade teksta i pretrage koriste elektronski rečnici srpskog jezika (videti [12] i [13]). U narednim verzijama ovog sistema biće moguće vršiti odabir i drugih jezika.

## 5.3. Sistem za kontrolisanje procesa nadgledanja veb strana

Centralno mesto sistema *WebMonitoring* zauzima sistem za kontrolisanje procesa nadgledanja veb strana, koji se sastoji od nekoliko Java klasa. Sistem je dizajniran tako da je moguće pokrenuti proizvoljno mnogo nezavisnih procesa nadgledanja. Jedan proces nadgledanja je definisan sa sledećim atributima:

*URL* – adresa veb strane od koje počinje pretraga

*graf* – lokacija datoteke *.fst2*, tj. grafa koji opisuje dogadjaj od interesa

*brojNivoa* – celobrojni podatak koji definiše broj nivoa strana do kojeg se vrši preuzimanje veb strana sa Interneta. Detaljnije o ovom atributu videti u poglavlju 5.4.

*nacinAlarmiranja* – podatak tipa *String* koji određuje na koji način će korisnik biti obavešten o događaju. Predviđene su dve mogućnosti, alarmiranje slanjem poruke na elektronsku adresu i snimanje teksta veb strane na kojoj se događaj desio lokalno na tvrdi disk računara. Ovaj atribut ima sledeći oblik *adresa;lokacija,* gde je *adresa* e-mail adresa na koju treba poslati poruku, a *lokacija* putanja do direktorijuma u kome treba snimiti veb stranu. Ukoliko su adresa ili lokacija jednake "*null*", alarmiranje korisnika navedenim načinom će izostati.

*vremenskaDinamika* – celobrojna vrednost tipa *long*, koja predstavlja vremenski period u milisekundama u kome treba ponoviti pretragu.

Svakom procesu nadgledanja odgovara po jedan objekat klase *MonitoringProcess.* Ova klasa je definisana tako da nasleđuje klasu *Thread*, tj. predstavlja izvršnu nit, pa se nadgledanje različitih strana odvija paralelno i međusobno nezavisno.

```
public class MonitoringProcess extends Thread {
  String URL;
  String graf;
  String brojNivoa;
  String email;
  String lokacija;
  long interval=0; //interval ponavljanja citavog procesa preuzimanja strana
  long pocetak=0; // vreme pocetka nadgledanja
  long brojKonkordanci=0;
  CrawlerShell cs = null;
.
.
.
}
```

Atributi klase *MonitoringProcess* omogućavaju čuvanje prethodno opisanih atributa specifičnih za pojedini proces nadgledanja, ali i drugih podataka, kao što je na primer broj konkordanci određenog događaja na strani ili stranama u okviru procesa nadgledanja (*long brojKonkordanci*), ili vezu sa odgovarajućim objektom klase *CrawlerShell*, zadužene za preuzimanaje strana sa Interneta.

Objekti klase *MonitoringProcess* bivaju nakon kreiranja smešteni u objekat *procesi* klase *Vector.* Ovaj vektor čuva sve promene izvršene od strane korisnika nad procesima nadgledanja strana, kao što su kreiranje novih procesa, brisanje postojećih, izmena parametara procesa i sl. Nakon zatvaranja aplikacije, svi procesi sadržani u ovom vektoru bivaju upisani u datoteku *MonitorData.txt*, i time trajno sačuvani.

*Primer 8.*

Linija datoteke *MonitorData.txt* koja izgleda na sledeći način

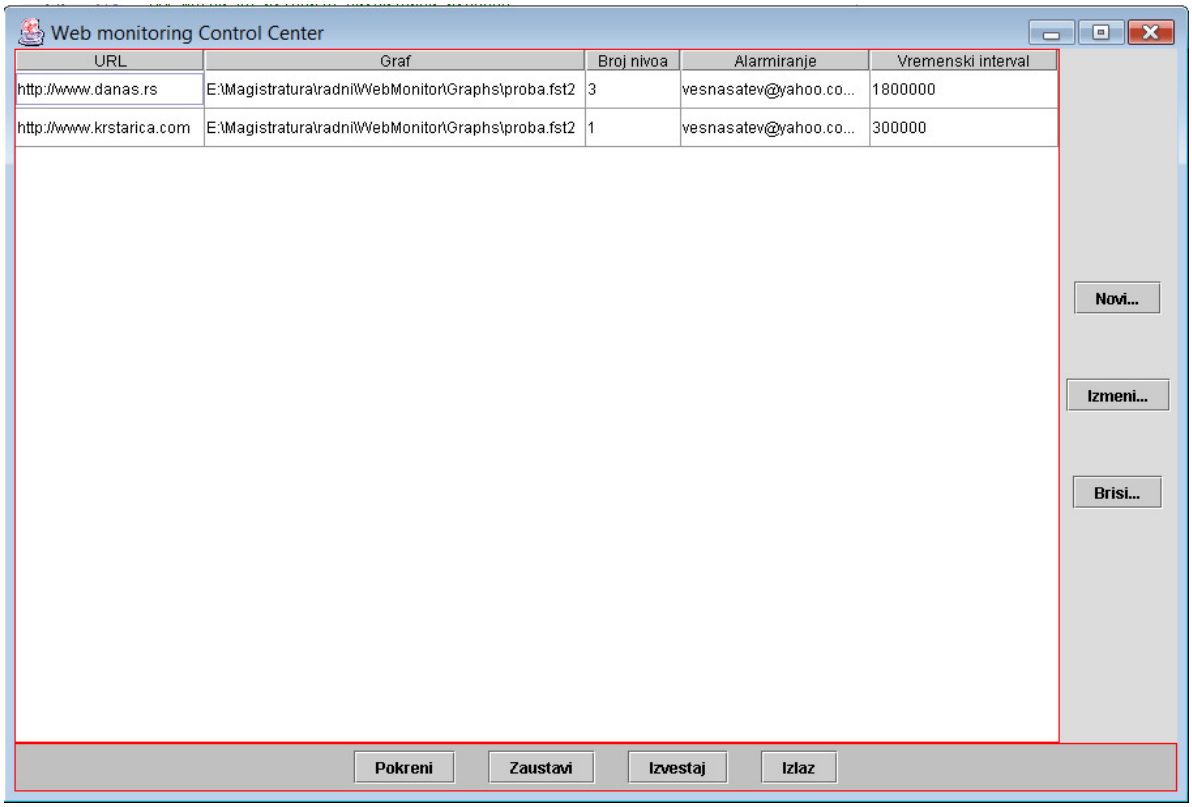
*http://www.krstarica.com E:\Graphs\dogadjaj1.fst2 1 vesnapajic@yahoo.com;null 300000*

definiše proces nadgledanja strane na adresi *http://www.krstarica.com*, pri čemu je *brojNivoa* jednak 1, što znači da se nadgleda samo ta strana, a ne i strane do kojih vode linkovi pronađeni na njoj. Događaj od interesa je opisan grafom koji se nalazi na lokaciji *E:\Graphs\dogadjaj1.fst2*. Ukoliko se desi navedeni događaj, potrebno je samo poslati poruku obaveštenja na adresu *vesnapajic@yahoo.com*. Proveru da li se događaj odigrao treba ponoviti nakon 5 minuta, tj. 300000 milisekundi.

Polazna klasa čitavog sistema je klasa *MonitorGUI*. Njenim pokretanjem na korisnikovom računaru se otvara prozor kontrolnog centra, odakle korisnik definiše i upravlja procesima nadgledanja (slika 10).

Konstruktor klase *MonitorGUI* u sebi sadrži pozive dva metoda, *preuzmiProcese()* i *jbInit()*.

```
/** Konstruktor */
  public MonitorGUI() throws HeadlessException {
    try {
      // preuzima podatke o procesima upisanim u MonitorData.dat
      preuzmiProcese();
      // inicijalno podesavanje komponenti prozora
      jbInit();
    }
    catch(Exception e) {
      e.printStackTrace();
    }
  }
```




*Sl. 10 Korisnički interfejs sistema WebMonitoring – klasa MonitorGUI*

Metod *preuzmiProcese()* otvara datoteku *MonitorData.txt*, ukoliko ona postoji. U ovoj tekstualnoj datoteci smešteni su podaci o procesima nadgledanja koji su kreirani prilikom prethodnih pokretanja aplikacije. Linije datoteke *MonitorData.txt* su oblika

```
URL \t graf \t brojNivoa \t nacinAlarmiranja \t vremenskaDinamika,
```

pri čemu svaka linija ove datoteke odgovara jednom procesu nadgledanja. Za svaki proces se kreira po jedan objekat klase *MonitoringProcess*, i svi oni bivaju smešteni u objekat *procesi* klase *Vector*. Promenljiva *procesi* je definisana kao klasna promenljiva klase *MonitorGUI.*

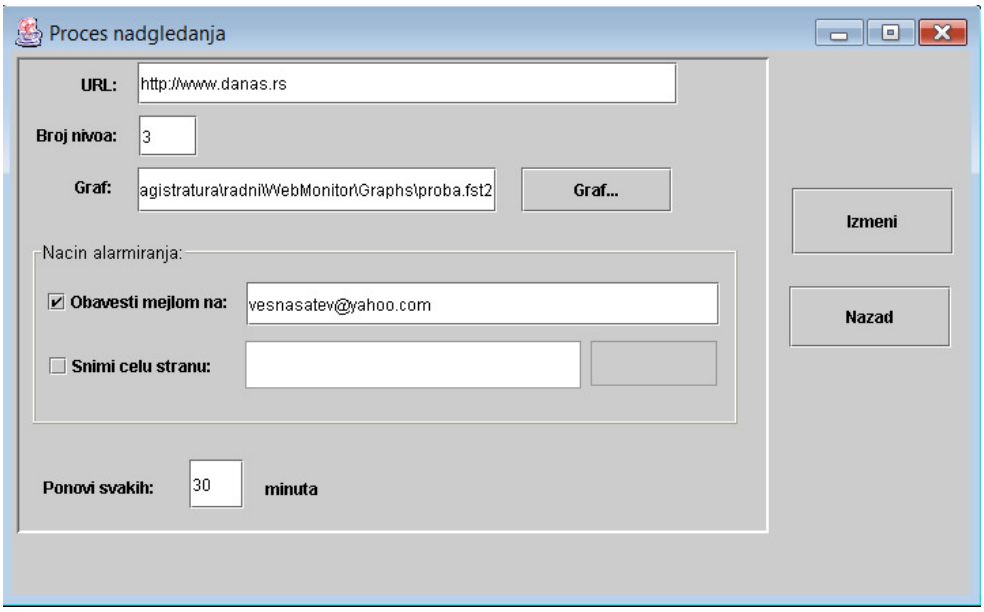
```
static Vector procesi = new Vector();
```

Prilikom pokretanja aplikacije, a nakon preuzimanja procesa iz datoteke *MonitorData.txt* i njihovog smeštanja u vektor *procesi*, vrši se inicijalizacija komponenti prozora aplikacije unutar metoda *jbInit()*. U okviru ovog metoda, osim podešavanja ostalih komponenti, kreira se i popunjava tabela koja prikazuje korisniku sve procese vektora *procesi*. Pomoću te tabele korisnik selektuje odgovarajući proces i preuzima određene akcije nad njim, pomoću dugmadi koji su mu na raspolaganju unutar prozora aplikacije. Akcije koje mogu biti preuzete su:

- kreiranje novog procesa – pritiskom na dugme "*Novi*" otvara se poseban prozor klase *ProcesGUI* (slika 11) koji omogućava unos podataka o novom procesu
- izmena postojećeg procesa - pritiskom na dugme "*Izmeni*" otvara se poseban prozor klase *ProcesGUI* (slika 11) koji omogućava izmenu podataka o postojećem procesu, i

to onom koji je u datom trenutku selektovan unutar tabele sa procesima glavnog prozora

- brisanje postojećeg proces - pritiskom na dugme "*Brisi*" podaci o selektovanom procesu se brišu iz tabele i vektora *procesi*
- pokretanje procesa nadgledanja – pritiskom na dugme "*Pokreni*" pokreće se izvršna nit u okviru koje započinje preuzimanje strana sa Interneta, počevši od strane definisane atributom URL. Pronađene strane se dalje šalje na analizu sistemu za postprocesiranje strana
- zaustavljanje procesa nadgledanja – pritiskom na dugme "*Zaustavi*" proces koji je selektovan se zaustavlja ukoliko je bio aktivan
- prikazivanje statusa procesa – pritiskom na dugme "*Izvestaj*" otvara se prozor sa porukom koja sadrži informaciju o tome da li je i kada proces pokrenut, broju strana koji je do tog trenutka obrađen, kao i broju pronađenih konkordanci




*Sl. 11 Prozor koji prikazuje podatke o jednom procesu nadgledanja*

## 5.4. Sistem za preuzimanje strana

Kada je korisnik definisao jedan proces nadgledanja, pritiskom na dugme *Pokreni* glavnog prozora aplikacije korisnik pokreće proces nadgledanja. Ovaj proces se odvija u nekoliko faza:

- prvo se sa Interneta preuzima strana koja se nalazi na adresi definisanoj atributom URL
- ova strana se šalje na analizu dvema odvojenim i nezavisnim sistemima. Jedan je sistem za preuzimanja strana koji na datoj strani pronalazi hiper veze do drugih strana i smešta ih u niz adresa sa kojih će biti preuzimane nove strane. Ovaj proces se odvija u skladu sa osnovnim principima *crawling*-a i određen je između ostalog atributom

*brojNivoa*. Ukoliko je *brojNivoa* jednak 1, ovaj postupak izostaje, tj. preuzima se samo jedna strana.

- strana se zatim šalje na analizu sistemu za naknadno procesiranje, koji vrši obradu teksta sa veb strane i njegovu pretragu u grafom koji je korisnik definisao.
- sve tri faze se ponavljaju za svaku adresu pronađenu i smeštenu u niz za dalje preuzimanje, pri čemu se *brojNivoa* smanjuje za jedan.
- jedan proces nadgledanja završava sa radom ili ako su preuzete sve strane do kojih je sistem došao (*brojNivoa* jednak je 0) ili ako je isteklo maksimalno dozvoljeno vreme za *crawling* navedenog sajta.
- ceo sistem ostaje aktivan i ponovo pokreće određeni proces nadgledanja nakon vremena definisanog od strane korisnika

Termin *crawling* je preuzet iz Engleskog jezika i označava poseban postupak pomoću koga se na efikasan način preuzimaju strane sa Interneta. Ovaj postupak podrazumeva da se polazeći od jedne ili više unapred definisanih veb adresa automatski obiđu nove strane, prateći hiper veze koje se na njima nalaze. Program koji je dizajniran da obavlja ovaj posao naziva se *crawler*. Prilikom dizajniranja *crawler*-a neophodno je voditi računa o sledećem [20]:

- efikasnost - broj hiper veza (veb adresa) koje *crawler* treba da obradi raste eksponencijalno sa povećanjem broja strana koje je obišao. Zbog te činjenice, sistem mora da bude u stanju da rukuje listom adresa na efikasan način sa stanovišta korišćenja memorije.
- duplikati – *crawler* mora da dodaje na listu adresa samo one adrese koje nije obišao, tj. da prepozna adrese strana koje su već obrađene i da ih ne dodaje na listu
- «učtivost» - *crawler* mora da vrši proveru sadržaja datoteke *robots.txt* na serveru veb strane i da ispoštuje direktive koje se nalaze u njoj. Takođe, neophodno je izbeći povlačenje prevelikog broja strana sa jednog veb sajta, kako se ne bi narušila njegova funkcionalnost.
- izbegavanje zamki – *crawler* mora da bude u stanju da prepozna i izbegne sajtove koji su kreirani sa namerom da izazovu beskonačne petlje kod *crawler*-a koji ih posećuju

Sistem za preuzimanje strana u okviru sistema *WebMonitoring* jeste *crawler* i kao takav zadovoljava osnovne principe *crawling*-a. Osnovni paket klasa je paket *crawler.driller,* koji sadrži nekoliko važnih klasa:

- klasa *_GO_PARAMS* je dizajnirana tako da čuva sve atribute važne za proces preuzimanja strana, kao što su URL početne strane, broj nivoa do koga se vrši preuzimanje, niz veb adresa od početne do tekuće strane, da li se vrši preuzimanje samo sa jednog Internet domena i sl.
- klasa *DrillingQueue* definiše dinamičku strukturu podataka za čuvanje niza URL adresa koje je potrebno obići
- klasa *PathFinder* obezbeđuje čuvanje informacije o putanji kojom je *crawler* došao do neke strane, tj. koji niz veb strana je obišao dok nije naišao na neku konkretnu stranu
- klasa *Driller* i njen metod *START()* vrše preuzimanje strana (*crawling*) od zadate strane pa sve do nivoa zadatog parametrom *brojNivoa*

Polazna klasa sistema je klasa *CrawlerShell* paket *crawler*, koja vrši sve potrebna podešavanja postupka preuzimanja strana i pokreće *crawler*. Naime, unutar ove klase postoji

veliki broj parametara pomoću kojih je moguće konfigurisati sistem za preuzimanje strana. Za potrebe sistema *WebMonitoring*, za neke od ovih parametara su uzete podrazumevane vrednosti koje korisnik ne može da menja. Tako je, na primer, predviđeno da vrednost parametra *timeout* bude 60 sekundi, što znači da će se na preuzimanje jedne strane sa Interneta čekati najviše jedan minut. Takođe, maksimalno vreme za preuzimanje strana sa jednog sajta (u jednom procesu nadgledanja), definisano parametrom *maxTimeout*, iznosi 20 minuta (1200 sekundi).

```
/** Klasa za preuzimanje strana pocevsi od odredjenog URL */
public class CrawlerShell{
.
// podrazumevane vrednosti argumenata
  static String hunt = null;
  public static boolean thisSiteOnly = false, silent = false;
  public static int maxTimeout = 1200, timeout = 60, maxlinks = 1000, maxretries = 3,
      levels = 0, sleepParamMS = 500;
  static boolean analyze = true, passedAskMe = false, flash = false,
      displayLinks = true, magicPath = false;
  static long timeStart = System.currentTimeMillis();
  static boolean logHttp = false, hideUrls = true;
  static boolean autoStatus = false; // auto status to show on the screen
```

Nakon izvršenih podešavanja parametara, pokreće se *crawler* za zadati URL. Tekst sa svake veb strane preuzete u ovom procesu biva poslat klasi *MonitoredText* na dalju analizu, tj. pretragu po odgovarajućem grafu.

## 5.5. Sistem za naknadno procesiranje teksta i alarmiranje korisnika

Kada je jedna veb strana preuzeta sa Interneta, neophodno je prvo izvršiti njenu pripremu, a zatim i pretragu za odgovarajućim događajem. Za tu svrhu u okviru sistema *WebMonitoring* postoji klasa *MonitoredText*, koja obezbeđuje metode za obradu teksta. S obzirom da centralno mesto u pronalaženju nekog događaja na strani ima *Unitex*-ov spoljašnji program *Locate*, tekst je neophodno pripremiti u skladu sa zahtevima ovog programa.

Za svaku stranu se prvo generiše ime koje će se nadalje koristiti u procesu obrade teksta za datoteke koje se tiču te strane. Kako bi ovo ime bilo jedinstveno, ono se generiše na osnovu vremena kada se vrši obrada. Zatim se kreira privremeni direktorijum pod tim imenom koji će biti korišćen za smeštanje datoteka koje nastaju u procesu obrade jedne strane.

Tekst pronađen na veb strani biva snimljen u tekstualnu datoteku smeštenu u privremeni direktorijum na tvrdom disku, i predstavlja polaznu datoteku u procesu obrade. Iako je tekst na veb stranama različito kodiran, ipak većina veb sajtova na srpskom jeziku u poslednje vreme koristi UTF8 kodiranje. UTF-8 (*8-bitni Unicode Transformation Format*) predstavlja kodiranje karaktera *Unicode* kodovima različite dužine. Pomoću njega je moguće predstaviti bilo koji karakter poštujući *Unicode* standard, a pri tom je kompatibilan i sa *ASCII* kodiranjem. Zbog te svoje osobine ovaj način kodiranja je postao dominatan za elektronske poruke i veb strane, pa se i u okviru sistema *WebMonitoring* predpostavlja je da je strana kodirana upravo koristeći UTF8. Tekst pronađen na strani snima se na tvrdi disk korisnika koristeći metode klase *fr.umlv.unitex.io.UnicodeIO*. Ova klasa je posebno dizajnirana za potrebe sistema Unitex, kako bi sve tekstualne datoteke koje nastaju u procesu obrade teksta bile kodirane i pročitane na odgovarajući način.

Posle preuzimanja strane, vrši se normalizacija teksta. Normalizacija teksta je proces u kome se prvenstveno vrši normalizacija separatora teksta, mada je moguće izvršiti i normalizaciju na osnovu nekog proizvoljnog pravila. Separatori teksta su prazno mesto, tabulator i novi red. Unutar teksta preuzetog sa neke veb strane moguće je da će se nekoliko ovih separatora nalaziti jedan pored drugog. Normalizacijom se sekvenca separatora teksta zamenjuje jednim praznim mestom. Ovaj proces se obavlja pomoću *Unitex*-ovog spoljašnjog programa *Normalize*.

Poziv programa *Normalize* ima sledeću sintaksu:

```
Normalize txt [-no_CR] [-f=norm]
```

Parametar `txt` predstavlja kompletnu putanju do tekstualnog fajla. Nakon obrade, program *Normalize* kreira nov, izmenjen tekst i smešta ga u datoteku sa ekstenzijom `.snt`. Opcioni parametar `-no_CR` zamenjuje bilo koju sekvencu separatora jednim praznim mestom, dok se parametar `-f=norm` koristi da odredi datoteku u kojoj su smeštena dodatna pravila normalizacije. U okviru sistema *WebMonitoring* prosleđuje se samo parametar `txt`.

Nakon normalizacije, klasa *MonitoredText* kreira i poseban direktorijum potreban za dalji tok procesa obreda teksta. Spoljašnji programi *Unitex*-a očekuju da ovaj direktorijum ima tačno određeno ime. Naime, ukoliko datoteka koja se obrađuje nosi naziv `VebStrana1.txt`, tada nakon procesa normalizacije bivaju kreirana datoteka `VebStrana1.snt` i direktorijum `VebStrana1_snt`.

Zatim se pristupa tokenizaciji teksta, tj. njegovom razdvajanju na leksičke jedinice. U okviru klase *MonitoredText* poziva se *Unitex*-ov spoljašnji program *Tokenize*, čiji poziv ima sledeću sintaksu:

```
Tokenize text alphabet [-char_by_char]
```

Parametar `text` predstavlja kompletnu putanju do tekstualnog fajla u kome se nalazi tekst dobijen nakon normalizacije (sa ekstenzijom `.snt`). Parametar `alphabet` predstavlja kompletnu putanju do datoteke koja sadrži definiciju alfabeta za jezik teksta. U eksperimentalnoj verziji sistema *WebMonitoring* podrazumevani jezik je srpski jezik, i korisnik nema mogućnost njegove promene. Opcioni parametar `-char_by_char` dopušta mogućnost da se tokenizacija vrši karakter po karakter, što je neophodno u slučaju nekih azijskih jezika. Za tekstove na srpkom jeziku ova opcija je isključena, pa se za jednu leksičku jedinicu smatra sekvenca slova, pri čemu su slova definisana pomoću datoteke `alphabet.txt`, zatim karakter koji nije slovo, separator rečenice i leksičke labele [19]. Lista leksičkih jedinica (tokena) biva snimljena u tekstualnu datotetu `tokens.txt`, pri čemu se svakom tokenu dodeljuje jedinstveni kod. Zatim se i ceo tekst kodira pomoću kodova tokena i biva snimljen u binarnu datoteku `text.cod`. Program `Tokenize` proizvodi i sledeće četiri datoteke:

- `tok_by_freq.txt` – tekstualna datoteka koja sadrži tokene sortirane po učestalosti pojavljivanja u tekstu
- `tok_by_alph.txt` - tekstualna datoteka koja sadrži tokene sortirane alfabetski
- `stats.n` – tekstualna datoteka koja sadrži informacije o broju separatora rečenica, broju tokena, broju reči i brojeva
- `enter.pos` – binarna datoteka koja sadrži listu pozicija novog reda u tekstu; ova datoteka se koristi u Uniitex radnom okruženju za usaglašavanje konkordanci sa originalnim tekstom

Na tekst koji je podeljen na leksičke jedinice moguće je primeniti elektornske rečnike pomoću *Unitex*-ovog spoljašnjeg programa *Dico*. U prvoj verziji programskog sistema WebMonitoring vrši se automatska primena rečnika, ukoliko on postoji. U naknadnim verzijama ovog programskog sistema korisnik će sam moći da odabere da li želi primenu rečnika ili ne. Poziv programa *Dico* ima sledeću sintaksu:

```
Dico text alphabet [-md=XXX] dic_1 [dic_2 ...]
```

Ovaj program primenjuje elektronski rečnik na tekst, pri čemu je neophodno da tekst bude podeljen na leksičke jedinice pomoću programa *Tokenize*. Parametar `text` predstavlja kompletnu putanju do datoteke u kojoj se nalazi tekst. Parametar `alphabet` predstavlja datoteku koja definiše slova za jezik na kom je tekst koji se obrađuje. Opcioni parametar `-md=XXX` se koristi da opiše koji morfološki rečnici će biti korišćeni, pri čemu `XXX` predstavlja listu rečnika u *.bin* formatu. Parametar `dic_i` predstavlja putanju i ime rečnika koji se koristi. Rečnik mora biti ili u *.bin* formatu, koji se dobija nakon primene programa *Compress* (videti [19]) ili u formatu *.fst2*, tj. u obliku grafa. Program *Dico* generiše nekoliko fajlova i snima ih u direktorijum teksta:

- `.dlf` – rečnik prostih reči iz tekstu;
- `dlc` – rečnik složenih reči iz tekstu;
- `err` – rečnik neprepoznatih reči;
- `tags.ind` – sekvence koje je potrebno umetnuti u automat teksta (videti [19])
- `stat_dic.n` – datoteka koja sadrži broj prostih, složenih i neprepoznatih reči

Sve ove datoteke bivaju snimljene u direktorijum određenog teksta, nakon čega se pristupa pretraživanju događaja opisanog grafom u tekstu. Pretraga se obavlja pomoću *Unitex*-ovog spoljašnjeg programa *Locate*, koji ima sledeću sintaksu

```
Locate text fst2 alphabet s/l/a i/m/r n [dir] [-thai] [-space] [-
md=XXX]
```

Ovaj program primenjuje određeni transduktor opisan grafom na tekst i kreira indeks pronađenih izraza koji odgovaraju zadatom grafu. Njegovi parametri su:

- `text` - kompletna putanja do teksta nad kojim se vrši pretraga (`datoteka.snt`)
- `fst2` - kompletna putanja do gramatike opisane grafom
- `alphabet` - putanja do datoteke u kojoj je opisan alfabet jezika
- `s/l/a` - parametar koji određuje da li se traži najduže (`l`), najkraće (`s`) ili sva (`a`) poklapanja
- `i/m/r` - parametar koji određuje način primene transduktora
- `n` – parametar koji određuje maksimalan broj pojavljivanja; ukoliko je njegova vrednost `all`, neophodno je pronaći sva pojavljivanja
- `dir` - opcioni parametar koji određuje direktorijum koji može da zameni podrazumevani direktorijum teksta
- `thai` – opcioni parametar, određuje da se vrši pretraga teksta na *Thai* jeziku
- `space` – opcioni parametar koji određuje da pretraga može da počne na bilo kojoj poziciji u tekstu; ovaj parametar se koristi za morfološke pretrage

- `md=XXX` – opcioni parametar koji određuje koji morfološki rečnici treba da se koriste

Nakon pretrage, program `Locate` proizvodi dve datoteke: `concord.ind` koja sadrži reference do pronađenih pojavljivanja izraza koji odgovaraju grafu i `concord.n`, koji sadrži broj pojavljivanja kao i procenat prepoznatih tokena unutar teksta. Ove dve datoteke bivaju snimljene u direktorijumu teksta. Sistem za alarmiranje ih koristi kako bi odredio da li treba ili ne da obavesti korisnika o događaju.

Unutar klase *MonitoredText* svi navedeni programi bivaju pozivani od strane dva metoda, *preProcessText()* i *search()*.

```
public File preProcessText(File f) {
    File exe = null;
    Process p = null;
    try {
        //normalizacija
     exe = new File(this.workingDir + File.separator + "App" + File.separator +
"Normalize.exe");
     p = Runtime.getRuntime().exec(exe.getAbsolutePath() + " \"" + f.getAbsolutePath()+ "\"");
     p.waitFor();

       //tokenizacija
     exe = new File(this.workingDir + File.separator + "App" + File.separator +
"Tokenize.exe");
     p = Runtime.getRuntime().exec(exe.getAbsolutePath() + " \"" + f.getAbsolutePath() +
           ".snt \"" + " \"" + alphabet.getAbsolutePath() + "\"");
     p.waitFor();
     return new File(f.getAbsolutePath() + ".snt");
    }
    catch (Exception e) {
      CrawlerLog.logException(e);
      return null;
    }
  }
```

```
public void search() {
    //pretrazuje fajl, rezultat se nalazi u fajlu concord.ind i concord.n
    File exe = null;
    Process p = null;
    try {
      //locate
      exe = new File(this.workingDir + File.separator + "App" + File.separator +
"Locate.exe");
      p = Runtime.getRuntime().exec(exe.getAbsolutePath() + " \"" + sntText.getAbsolutePath()
          + "\" \"" + graph.getAbsolutePath() + "\" \"" +  alphabet.getAbsolutePath()
          + "\" s i all");
      p.waitFor();
    }
    catch (Exception e) {
      CrawlerLog.logException(e);
    }

  }
```

Kada su kreirane datoteke `concord.ind` i `concord.n` u njima se nalazi informacija o broju pojavljivanja izraza koji odgovaraju događajima koji su od interesa. Metod *foundConcordances()* klase *MonitoredText* koristi datoteku `concord.n` i iz nje čita broj pojavljivanja. Ukoliko je ovaj broj veći od 0, tj. ukoliko su pronađeni izrazi u tekstu koji odgovaraju grafu, pomoću metoda *alert()* klase *MonitoredText* korisnik se o tome obaveštava. Način obaveštavanja zavisi od atributa *email* i *lokacija* klase *MonitoringProcess,* tj. od načina obaveštavanja koji je korisnik odabrao. U eksperimentalnoj verziji sistema *WebMonitoring* predviđene su dve mogućnosti, slanje elektronske poruke na e-mail adresu korisnika ili trajno snimanje veb strane na kojoj se neki događaj desio na lokalnoj memorijskoj jedinici.

# 6. PRIMER PRIMENE SISTEMA *WEBMONITORING* I NJEGOVA EVALUACIJA

Sistem *WebMonitoring*, iako još uvek u eksperimentalnoj fazi, ipak je potpuno funkcionalan i moguće je upotrebiti ga za nadgledanje bilo kog veb sajta ili strane. Kao što je opisano u prethodnim poglavljima, upiti koje koristi sistem *WebMonitoring* su u stvari grafovi kreirani u *Unitex*-u, koji predstavljaju konačne mašine kojima se opisuje grupa izraza nekog prirodnog jezika, tj. događaj koji korisnik iščekuje da se pojavi na nekom veb sajtu. U okviru ovog poglavlja biće opisan čitav postupak pretrage pomoću sistema *WebMonitoring*, iz ugla korisnika sistema.

## 6.1. Primer primene sistema *WebMonitoring*

Korisnika su zanimali svi tekstovi objavljeni u dnevnim listovima, koji se tiču trenutno aktuelnog predsednika Republike Srbije, Borisa Tadića. Da bi ovaj proces pretrage bio uspešno izveden, neophodno je preduzeti sledeće korake:

- definisati događaj od interesa
- opisati ga grafom koristeći sistem *Unitex*
- odabrati veb strane ili veb sajtove koji se nadgledaju
- u okviru sistema *WebMonitoring* definisati osnovne parametre nadgledanja
- pokrenuti procese nadgledanja

**Korak 1**. Događaj koji zanima korisnika jeste pojavljivanje izraza koji se odnose na predsednika Borisa Tadića u tekstovima objavljenim u dnevnim listovima. Neki od ovih izraza mogu biti:

*predsednik Tadić*

*predsednik Srbije*

*predsednik B. Tadić*

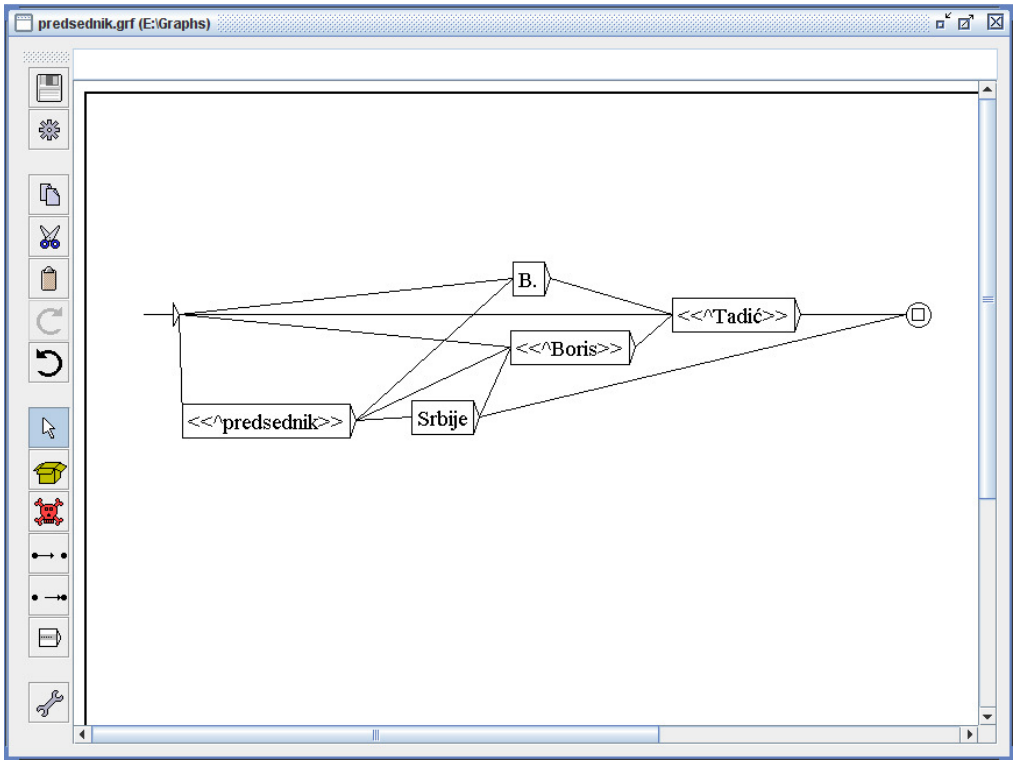
*predsednik Boris Tadić*

*Boris Tadić*

i slični izrazi, kao i njihovi promenjeni oblici (*predsednika Tadića, Borisu Tadiću* i sl.)

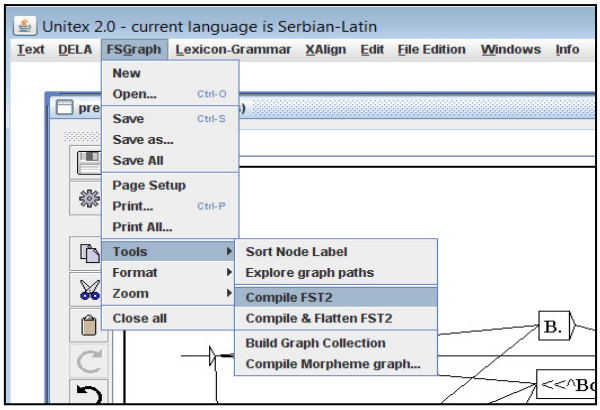
**Korak 2**. Ovako definisan događaj korisnik opisuje grafom, koristeći programski sistem *Unitex*. Postoji više načina na koji je moguće opisati navedeni događaj, u zavisnosti od veštine korisnika, kao i od resursa sa kojima raspolaže. Jedan od načina jeste korišćenje morfoloških filtera. Na primer, filter <<^*Boris*>> bi odgovarao svim rečima koje počinju sa

*«Boris»*, npr. *Boris, Borisa, Borisu* i sl. Tako bi graf koji opisuje prethodno definisani događaj mogao da izgleda kao na slici 12.




*Sl 12. Graf kojim korisnik opisuje događaj od interesa*

Nakon što je korisnik kreirao graf, on ga snima na disk pod imenom *predsednik.grf*, a zatim i kompilira u datoteku *.fst2* (slika 13).




*Sl 13. Odabir opcije Compile*

Kompiliran graf biva snimljen u istom direktorijumu kao i datoteka *.grf*.

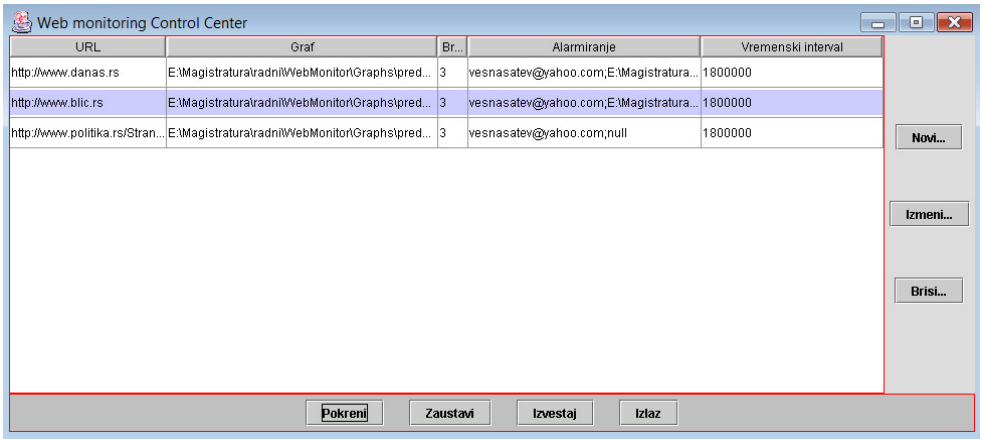
**Korak 3**. Korisnik bira veb strane ili veb sajtove koje želi da nadgleda. S obzirom da korisnika zanimaju novinski članci u kojima se govori o predsedniku Tadiću, a imajući u vidu da korisnik ne zna unapred na kojoj strani će se dati tekst pojaviti, on bira opciju da nadgleda ceo veb sajt umesto samo jedne strane. Pri tome, korisnik se odlučuje za nekoliko dnevnih listova, dostupnih na adresama **http://www.danas.rs**, **http://www.blic.rs** i **http://www.politika.rs/Stranice/33.lt.html** (latinična verzija veb sajta dnevnog lista «Politika»).

**Korak 4**. U okviru sistema *WebMonitoring*, korisnik kreira tri procesa, po jedan za svaki od navedenih sajtova (slika 14.). Prilikom kreiranja procesa, korisnik definiše sledeće parametre:


*Sl. 14 Kreiranje jednog procesa nadgledanja*

- URL – polazna adresa, veb strana koja se prva učitava i od koje kreće proces nadgledanja
- Broj nivoa – u ovom slučaju korisnik bira 3 nivoa, s obzirom da se radi o veb sajtu na kome se ceo tekst nalazi tek na dubini 3. To praktično znači da korisnik kada pristupa veb sajtu preko veb čitača, mora da otvori polaznu stranu i još dve (ukupno tri strane) da bi došao do celog teksta nekog članka. Ovaj broj korisnik bira na osnovu sopstvene procene i strukture samog veb sajta. Ukoliko korisnik želi da nadgleda samo jednu veb stranu, ovaj parametar bi trebalo da bude 1.
- Graf – korisnik unosi lokaciju na tvrdom disku na kojoj je snimljen graf *predsednik.fst2*.
- Način alarmiranja – korisnik bira opciju da bude obavešten elektronskom porukom ukoliko se pojavi izraz koji odgovara grafu, ali i da odgovarajuća strana bude snimljena na tvrdi disk
- Korisnik želi da se svakih 30 minuta ovaj proces ponovi.

Nakon što je korisnik definisao sva tri procesa nadgledanja, za svaki veb sajt po jedan proces, glavni prozor aplikacije *WebMonitoring* izgleda kao na slici 15.




*Sl. 15 Glavni prozor i kontrolni centar aplikacije*

**Korak 5**. Unutar tabele, korisnik selektuje određeni proces i pritiskom na dugme *Pokreni* pokreće procese nadgledanja. Nakon toga počinje proces preuzimanja strana i njihovog analiziranja od strane sistema *WebMonitoring*. Korisnik biva obavešten za svaku stranu na kojoj je pronađen izraz koji odgovara grafu.

## 6.2. Evaluacija sistema *WebMonitoring*

Programski sistem *WebMonitoring* je testiran na računaru sledećih karakteristika:

- *TOSHIBA* personalni računar
- procesor *Intel(R) Core(TM)2 Duo CPU @2.40 GHz*
- 2046 MB RAM memorije
- OS *Windows Vista Home Premium*
- Internet konekcija brzine 0.88 Mbps

Rezultat rada sistema *WebMonitoring*, nakon pokretanja procesa opisanih u poglavlju 6.1., dat je u tabeli 1.

*Tabela 1. Rezultat rada sistema WebMonitoring*

| URL | Broj preuzetih strana | Broj strana na kojima je pronađen izraz |
|---|---|---|
| **http://www.danas.rs** | 395 | 81 |
| **http://www.blic.rs** | 301 | 138 |
| **http://www.politika.rs/Stranice/33.lt.html** | 606 | 89 |

Program je uspešno preuzeo veb strane počevši od polazne adrese definisane parametrom URL. Sve strane su obrađene bez prijavljenih grešaka i pronađeni su izrazi koji odgovaraju grafu koji je korisnik kreirao. Korisnik je obavešten elektronskom porukom, po jednom za svaku stranu na kojoj se desio događaj. U primeru 9. prikazana je jedna takva poruka.

*Primer 9.*

```
Na strani http://www.politika.rs/rubrike/tema-dana/Iz-Tadicevog-kabineta-u-naprednjake.lt.html
pronadjen je dogadjaj opisan grafom E:\Magistratura\radni\WebMonitor\Graphs\predsednik.fst2.
Sledi lista izraza koji odgovaraju trazenom dogadjaju.

                        Tema dana : Iz      Tadićevog       kabineta u naprednjake : POLIT
ja Moja kuća Karikaturisti Tema dana Iz     Tadićevog       kabineta u naprednjake Vladimi
redsedništva, uz saopštenje pres službe     predsednika Srbije      u kome se navodi da „
a Srbije u kome se navodi da „do danas”     predsedniku Borisu Tadiću       nisu dostavlje
Ako je neko očekivao kesu ružnih reči o     predsedniku Srbije      , njegovim saradnicima
```

U sva tri procesa nadgledanja sistem je završio sa radom nakon što je prošlo 20 minuta, s obzirom da je parametar *maxTimeout*, koji u okviru klase *CrawlerShell* definiše maksimalno vreme koje sistem može da utroši na jednom veb sajtu, postavljen na 1200 sekundi. Ovo praktično znači da nisu obrađene sve veb strane do kojih je sistem došao u procesu *crawling*-a, kada je broj nivoa 3. Isti sajtovi su nadgledani i do dubine 2 i tada su uspešno obrađene sve strane.

Prilikom testiranja sistema *WebMonitoring* došlo se do sledećih procena:

- sistem veoma uspešno preuzima veb strane polazeći od jedne, inicijalne, adrese. Do sada nije bilo grešaka u radu, niti situacija da neka strana nije mogla da bude preuzeta.
- prepoznavanje izraza koji odgovaraju grafu od strane sistema WebMonitoring je potpuno, tj. svi izrazi koji se pojave na strani, a odgovaraju zadatom grafu, bivaju prepoznati od strane sistema.
- brzina rada sistema je zadovoljavajuća kada se u obzir uzme njegova namena, iako su poboljšanja u smislu povećanja brzine rada moguća. Ova poboljšanja se odnose prvenstveno na upotrebu radne memorije za čuvanje promena na tekstu između dva poziva spoljašnjih programa *Unitex*-a, umesto dosadašnjeg čuvanja na tvrdom disku. Naime, sam *Unitex* i njegovi spoljašnji programi su tako dizajnirani da sve promene na tekstu snimaju na tvrdi disk, pa je takva praksa nastavljena i u okviru sistema *WebMonitoring*. Povećanje brzine rada je moguće i uvođenjem niti unutar jednog procesa nadgledanja. U trenutnoj verziji sistema *WebMonitoring* jedan proces nadgledanja se izvršava u okviru jedne programske niti, dok se unutar njega operacije preuzimanja i obrade veb strana izvršavaju sekvencijalno. Moguće je svakoj strani dodeliti posebnu programsku nit u okviru koje bi ona bila obrađena i na taj način ubrzati program, pri čemu treba voditi računa o broju trenutno aktivnih niti. Naime, broj strana raste eksponencijalno u odnosu na broj nivoa do koga se vrši preuzimanje, pa postoji mogućnost da dođe do preopterećenja sistema.
- iz ugla korisnika, neophodno je izvršiti određena poboljšanja sistema, a u smislu omogućavanja veće kontrole nad samim procesom nadgledanja. Ovim se prvenstveno misli na mogućnost podešavanja parametara kao što su jezik teksta, upotreba rečnika, dužina konteksta, maksimalno vreme za preuzimanje strana sa jednog sajta, da li je dozvoljen odlazak na druge domene i sl.
- prilikom nadgledanja veb sajta **www.blic.rs** došlo je do pojavljivanja događaja na više od 45% preuzetih strana. Razlog tome nije veliki broj tekstova u kojima se pojavljuje izraz od interesa, već struktura samog veb sajta. Naime, unutar skoro svih strana navedenog veb sajta postoji deo u kome se nalaze naslovi trenutno aktuelnih vesti, a u okviru kojih se pojavljivao jedan od traženih izraza. Ovakvu situaciju bi trebalo izbeći u narednim verzijama sistema *WebMonitoring,* tj. nadgraditi sistem tako da bude u stanju da prepozna pojavljivanje izraza u istom kontekstu.

# 7. ZAKLJUČAK

U radu je ukazano na problem velikog i brzog razvoja informacionih tehnologija i Interneta. Svakodnevno dodavanje novih sadržaja i podataka na Internet, količina i dostupnost ovih podataka, kao i njihova ažurnost, čini da se ljudi sve više oslanjaju na Internet kao na izvor informacija. Međutim, ovakav razvoj Interneta dovodi do problema pronalaženja određene informacije. Postojeći pretraživači mogu da adekvatno odgovore na samo neke od zahteva korisnika, kako zbog ograničenja u okviru samog procesa preuzimanja strana sa Interneta, tako i zbog neadekvatnog načina postavljanja upita.

U okviru ove teze razmatrana je mogućnost poboljšanja postavljanja upita u smislu zadavanja kompleksnijih pretraga, kao i pristup sadržajima na veb stranama u što kraćem periodu od trenutka njihovog pojavljivanja.

Kao rešenje za omogućavanje kompleksnijih pretraga predložena je primena teorije konačnih transduktora, s obzirom da su konačne mašine pogodne kao način za jednostavno opisivanje relevantnih lokalnih fenomena koji se pojavljuju prilikom proučavanja prirodnih jezika, a sa druge strane su vremenski i prostorno efikasne u računarstvu. Konačni transduktori se koriste za postavljanje upita, kao i za obradu teksta, i to kroz programski sistem *Unitex*. Ovaj programski sistem ima razvijen korisnički interfejs za kreiranje i manipulaciju grafova konačnih automata i transduktora, kao i veliki broj programskih modula koji omogućavaju lingvističku obradu teksta, njegovu normalizaciju, tokenizaciju, razdvajanje na rečenice, primenu leksičkih resursa (elektronskih rečnika), kao i pretragu teksta regularnim izrazima ili grafovima. Unutar korisničkog interfejsa *Unitexa* korisnik kreira graf koji opisuje događaj koji se traži. Ovaj graf predstavlja ulazni podatak za dalje pretragu nad tekstom sa veb strane.

Pristup sadržajima na veb stranama je obezbeđen dizajniranjem i implementacijom posebnog programskog sistema, *WebMonitoring,* napisanog u programskom jeziku *Java*. Ovaj sistem podržava koncept nadgledanja veba, koji je nastao iz potrebe za automatizacijom određenih akcija koje korisnik preduzima sa ciljem da bude obavešten o promenama nastalim na određenoj veb strani ili sajtu. Korisnički interfejs programskog sistema *WebMonitoring* omogućava korisniku kontrolisanje procesa nadgledanja zadavanjem osnovnih parametara, kao što su URL veb sajta ili strane koja se nadgleda, vremenska dinamika kojom se preuzimaju sadržaji sa navedene adrese, dubina (nivo) sajta do koga se vrši preuzimanje, način alarmiranja korisnika u slučaju da se određeni događaj desio i sl. Sistem *WebMonitoring* dalje integriše sistem za preuzimanje strana, koji je dizajniran u skladu sa principima *web crawling*-a. Polazeći od jedne, inicijalne URL adrese, sistem preuzima redom veb strane na čije veze naiđe. Svaka veb strana biva analizirana sa dva aspekta. Prvo se vrši analiza strane u potrazi za hipervezama ka novim stranama koje bi sistem mogao da preuzme. Ove hiperveze bivaju smeštene u red za dalju obradu ukoliko njihova dubina («udaljenost» od polazne strane) ne prelazi zadati broj nivoa do koga se vrši preuzimanje. Zatim se tekst pronađen na strani analizira u potrazi za događajem koji se iščekuje. Primena grafa događaja na tekst vrši se pozivanjem odgovarajućih programskih modula *Unitexa*. Ukoliko je pronađen tekst koji odgovara događaju, korisnik ima mogućnost da o tome bude obavešten preko elektronske pošte. Na taj način je korisnikovo angažovanje u pretrazi svedeno samo na zadavanje parametra pretrage.

Prilikom projektovanja i izrade sistema *WebMonitoring* uzeta je u obzir potreba korisnika da ciljano pristupa određenim sadržajima na Internetu i da pri tome ima punu kontrolu nad procesima koji se tom prilikom odvijaju. Takođe, korisniku je omogućeno da definiše složene događaje čije ga pojavljivanje interesuje, na relativno jednostavan način. Korišćenje grafova koji na intuitivan i pregledan način opisuju pojedine jezičke fenomene s jedne strane pojednostavljuje korisniku postupak postavljanja složenih upita, a sa druge strane korisniku pruža mogućnost kakvu nema prilikom korišćenja klasičnih pretraživača. Uz sve to, korisnik sam bira način na koji će biti obavešten o realizovanom događaju.

Prva verzija sistema *WebMonitoring* predstavlja primer na koji način je moguće povezati leksičku obradu teksta i program *Unitex* sa konceptom kao što je nadgledanje veb strana. Iako je pomenuta verzija programa potpuno funkcionalna i pokazuje pozitivne efekte primene konačnih transduktora na proces pretrage informacije, ipak je u narednim verzijama programa neophodno izvršiti određena poboljšanja. Ta poboljšanja se pre svega odnose na otklanjanje eventualnih grešaka u kodu, zatim na povećanje performansi samog programa i njegove brzine rada. Takođe, korisnik bi trebalo da ima veću kontrolu nad samim procesom, u smislu odabira jezika, odluke da li će ili ne da primeni rečnike na tekst i slično. Sistem *WebMonitoring* je sistem opšte namene. U budućim verzijama moguće je modifikovati sistem tako da bude specijalizovan ili za pojedine vrste tekstova (na primer medicinske, tehničke i sl.) ili za posebne namene, kao što je nadgledanje elektronskog sandučeta za poruke ili pretragu za određenim proizvodom u nekoj bazi podataka dostupnoj na Internetu, prilikom čega se očekuju dodatna poboljšanja efikasnosti sistema.

U svakom slučaju, programski sistem *WebMonitoring* je od značaja, jer ukazuje na način prevazilaženja određenih problema u procesu pretrage informacija, a posebno jer koristi lingvističke alate za postavljanje upita i prevazilazi ograničenja koja imaju današnji veb pretraživači u pristupu informacijama.

# 8. LITERATURA


[1] Sherman, C., Price, G.: *The Invisible Web: Uncovering Information Sources Search Engine Can't See,* Information Today Inc., 2005.

[2] Vitas D., Pavlović - Lažetić G.: *Extraction of named entities in Serbian using INTEX*, Formaliser les langues avec l'ordinateur: De INTEX a Nooj, eds. Koeva, S, Maurel, D, Silberztein, M, Presses Universitaires de Franche-Comte, serie Archive, Bases, Corpus, n$^0$3, 2007, ISBN: 978-2-84867-189-5, ISSN: 1771-8996, 281-302.

[3] The Unicode Standard, http://www.unicode.org

[4] http://www.internetworldstats.com

[5] Manning C., Raghavan P., Schutze H.: *An Introduction to Information Retrieval,* ISBN: 0521865719, Cambridge University Press, 2008.

[6] Vitas, D.: *Prevodioci i interpretatori: Uvod u teoriju i metode kompilacije programskih jezika,* Matematički fakultet, 2006.

[7] Jurafsky, D., Martin, J. H.: *Speech and language processing,* Prentice-Hall Inc., 2000.

[8] Woods, W.A.: *Transition network grammars for natural language analysis*, Communications of the Association for the Computational Machinery*,* 13(10), 1970.

[9] Roche, E., Schabes, Y.: *Finite-state language Processing,* The MIT Press, 1997.

[10] Kornai, A.: *Extended finite state models of language,* Cambridge University Press, 1999.

[11] Aho, A.V., Hopcroft, J.E., Ullman, J.D.: *The Design and Analysis of Computer Algorithms*, Addison Wesley, Reading, MA, 1974.

[12] Krstev, C., Vitas, D.: *Corpus and Lexicon - Mutual Incompletness*, in Proceedings of the Corpus Linguistics Conference, 14-17 July 2005, Birmingham, eds. Pernilla Danielsson and Martijn Wagenmakers, ISSN 1747-9398, http://www.corpus.bham.ac.uk/PCLC/, 2005.

[13] Vitas, D., Krstev, C., Obradović, I., Popović, Lj., Pavlović-Lažetić, G.: *Processing Serbian Written Texts: An Overview of Resources and Basic Tools*, in Workshop on Balkan Language Resources and Tools, 21 Novembar 2003, Thessaloniki, Greece, eds, S. Piperidis and V. Karkaletsis, pp. 97-104, 2003.

[14] Gross, M., Perrin, D.: *Electronic Dictionaries and Automata in Computational Linguistics*, in Proceedings of LITP Spring School on Theoretical Computer Science Saint-Pierre d'Oleron, France, May 25.-29., 1987

[15] Roche, E.: *Finite state transducers: parsing free and frozen sentences,* Extended finite state models of language*,* Cambridge University Press, 108.-120. 1999.

[16] Casacuberta, F., Vidal, E., Picó, D.: *Inference of finite-state transducers from regular languages,* Pattern Recognition, Volume 38, Issue 9, Pages 1431-1443, Septembar 2005

[17] Friburger, N., Maurel, D.: *Finite-state transducer cascades to extract named entities in text,* Theoretical Computer Science, Volume 313, Issue 1, 16 February 2004, Pages 93-104

[18] Bernardini, S., Baroni, M., Evert, S..: *A Wacky Introduction*, WaCky! - Working Papers on the Web as Corpus, Dipartimento di Studi Interdisciplinari su Traduzione Lingue e Culture (SITLeC) dell'Alma Mater Studiorum – Universit`a di Bologna, 2006

[19] Paumier, S: *Users Manual*, http://www-igm.univ-mlv.fr/˜unitex/UnitexManual2.0.pdf Institut Gaspard-Monge, University of Paris-Est Marne-la-Vall´ee, France, October 2008.

[20] Baroni, M., Bernardini, S.: *BootCaT: Bootstrapping corpora and terms from the Web*. In: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC-2004), Lisbon, 2004.

[21] Sastre, J. M., Forcada M.: *Efficient parsing using recursive transition networks with output*. In Zygmunt Vetulani, editors, 3rd Language & Technology Conference (LTC'07). 5-7 October 2007. pp. 280–284. Note: (5 pp.)

[22] Sastre, J. M.: *Efficient Parsing Using Filtered-Popping Recursive Transition Networks*. Lecture Notes in Computer Science. vol. 5642. June 2009. pp. 241–244

[23] Vitas, D.: *O problemu ne(pre)poznate reči.* Zbornik Matice srpske za filologiju i lingvistiku. Matica srpska, Novi Sad, Knj. 50, str. [111]-120, 2007.