Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/21BE15Stojakovic

TESTNO OKRUŽENJE ZA RUKOVANJE METEOROLOŠKIM PODACIMA POMOĆU SPARQL UPITA

TEST ENVIRONMENT FOR HANDLING METEOROLOGICAL DATA USING SPARQL QUERIES

Stefan Stojaković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj - *U radu je prikazano proširenje SPARQL Playground aplikacije koje omogućuje rad sa geoprostornim podacima, kao i njihovo iscrtavanje. Dizajniran je i implementiran WeatherService koji je testiran na primeru podataka o vremenskim prilikama, pri grafičkom prikazu opasnosti od nevremena.*

Ključne reči: *RDF, SPARQL, GeoSPARQL, WeatherService*

Abstract: *The paper presents an extension of the SPARQL Playground application that supports work with weather geospatial data. It also provides support for geospatial data drawing. WeatherService was designed, implemented and tested on the example of weather data, and hazard graphical representation.*

Keywords*: RDF, SPARQL, GeoSPARQL, WeatherService*

1. UVOD

Aplikacija *SPARQL Playground* nudi mogućnosti učenja SPARQL jezika kroz direktan rad sa SPARQL upitima. Kao takva predstavlja veoma koristan alat za studente i razvojne inženjere i predstavlja dobru bazu za dodatna proširenja koja se oslanjaju na SPARQL jezik. Jedno takvo proširenje prikazano je u ovom radu.

Originalna *SPARQL Playground* aplikacija ne podržava rad sa geoprostornim podacima, a takođe ne podržava ni rad sa podacima o vremenskim opasnostima, kao ni grafičku simulaciju istih, te je ideja rada da se podrže nedostajuće funkcionalnosti.

Implementiran je *WeatherService* za rad sa meteorološkim podacima o potencijalnim opasnostima od nevremena, koji su u TTL formatu, kao i simulator servisa, kako bi se grafički prikazao uticaj opasnosti na postojeću opremu. Obezbeđeni su podaci u TTL formatu za potrebe simulacije kretanja vremenskih nepogoda.

Novonastala *SPARQL Playground* aplikacija osim osnovnih funkcija ima i mogućnost rada sa geoprostornim podacima, te je bilo potrebno postojeći model podataka elektrodistributivnog izvoda proširiti mogućnošću prikaza geoprostornih podataka.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Milan Gavrić, docent.

2. TEORIJSKE OSNOVE

2.1 SPARQL

SPARQL Protocol and RDF Query Language (SPARQL) je standardni jezik za upite i protokol za povezane podatke na *web*-u i *Resource Description Framework* (RDF) baze podataka [1]. SPARQL omogućava korisniku da čita podatke iz baze podataka ili iz nekog drugog izvora podataka koji se može mapirati na RDF graf. Upiti SPARQL-a se sastoje od skupa trojki gde svaki element (subjekat, predikat ili objekat) može biti promenljiva. Promenljive se mapiraju na trojke u bazama podataka. SPARQL pruža i niz funkcija za konstruisanje složenih upita, za navođenje dodatnih uslova za filtriranje i za formatiranje konačnog rezultata [3].

2.2 GeoSPARQL

GeoSPARQL je definisan kao OGC (*Open Geospatial Consortium*) standard i predstavlja proširenje SPARQL jezika mogućnostima upita nad geoprostornim podacima [2]. Standard podržava predstavljanje i ispitivanje geoprostornih podataka na semantičkom *web*-u. *GeoSPARQL* definiše i rečnik za predstavljanje geoprostornih podataka u RDF-u. Pored toga *GeoSPARQL* je dizajniran za podršku sistema zasnovanih na kvalitativnom prostornom zaključivanju i sistema zasnovanih na kvantitativnim prostornim proračunima [3].

2.2.1 Klase

GeoSPARQL opisuju dve glavne klase:

geo:SpatialObject i **geo:Feature**.

Klasa **geo:SpatialObject** predstavlja sve ono što može imati prostornu predstavu, dok **geo:Feature** predstavlja opšte karakteristike objekata.

GeoSPARQL podržava određen broj topoloških prostornih relacija. Topološke relacije definisane u familiji *Simple Features* koje su korišćene u radu kao rezultat vraćaju *true* ili *false*:

1. equals – geo:sfEquals – u odnosu na jednakost dva geometrijska objekta;

2. disjoint – geo:sfDisjoint – u odnosu na razdvojenost dva geometrijska objekta;

3. intersects – geo:sfIntersects – u zavisnosti da li dva geometrijska objekta imaju presek;

4. touches – geo:sfTouches – u zavisnosti da li se dva geometrijska objekta dodiruju;

5. within – geo:sfWithin – u zavisnosti da li je jedan geometrijski objekat unutar granica drugog;

6. contains – geo:sfContains – u zavisnosti da li jedan geometrijski objekat sadrži drugi;

7. overlaps – geo:sfOverlaps – u zavisnosti da li jedan geometrijski objekat preklapa sa drugim;

8. crosses – geo:sfCrosses – u zavisnosti da li jedan geometrijski objekat preseca drugi.

2.3 RDF4J

Eclipse RDF4J je *Java* okruženje koje se koristi za rad nad *RDF* podacima. Uključuje kreiranje, parsiranje, skladištenje i analizu *RDF* upita. Nudi *API* (*Application Programming Interface*) koji je jednostavan za upotrebu i koji se može povezati sa svim vodećim rešenjima za *RDF* baze podataka i *RDF* repozitorijume. Omogućava kreiranje aplikacija koje koriste povezane podatake (*linked data*) i semantički *web*. Podržava dva skript jezika: *SPARQL* i *SeRQL* [4].

3. TEHNOLOGIJE I ALATI

3.1 Enterprise Architect i Visual Studio Code

Sparx Systems Enterprise Architect je korišćen za vizuelno modeliranje i dizajn zasnovan na UML jeziku [3, 5], dok je Visual Studio Code korišćen kao editor izvornog koda.

3.2 SPARQL Playground

SPARQL Playground je samostalna *web* aplikacija sa podrškom za rad na više platformi, koja se koristi za učenje SPARQL-a i RDF upitnog jezika [6]. Implementirana je u *Java Sprint Boot i AngularJS*. Verzija koja je korišćena se oslanja na *Sesame* 2.8.6. *framework.*

4. DIZAJN REŠENJA

4.1 Opis rešenja

Postojeća aplikacija *SPARQL Playground*, koja predstavlja osnovu ovog rada, ne ispunjava sve potrebne uslove za dostizanje cilja - rada sa geoprostornim i vremenskim podacima. Ona koristi *OpenRDF Sesame framework* kao *engine* koji ima mogućnost izvršavanja SPARQL upita, ali ne i *GeoSPARQL* upita. Tu nastaje potreba za migracijom *OpenRDF Sesame* na *RDF4J*, koji podržava rad sa geoprostornim podacima. Koristeći podatke date u RDF trojkama, kao i *RDF4J* i *Leaflet* tehnologije, predloženo rešenje implementira okruženje koje, koristeći SPARQL upitni jezik proširen sa *GeoSPARQL* upitima, prikazuje poligone koji predstavljaju mreže distributivnih izvoda, preseke linija izvoda, nadolazeće promene vremenskih uslova koje će uticati na rad opreme u polju, itd.

Rešenje koristi *Leaflet* biblioteku kako bi se prikazale pozicije na kojima su uz pomoć SPARQL upita iscrtani izvodi, tačke preseka određenih izvoda, poligoni različitih promena vremenskih uslova (oluja, grom, požar, itd.), kao i koji distributivni izvodi mogu biti pogođeni tim promenama.

4.2 Mogućnosti predloženog rešenja

Urađena je migracija Sesame na *RDF4J* kako bi aplikacija imala mogućnost rada sa *GeoSPARQL* podacima i *GeoSPARQL* upitima. Radom sa *GeoSPARQL* podacima podržani su različiti upiti u vidu iscrtavanja izvoda, izvoda koji nisu pod nepogodama, kao i izvoda koji jesu pod nepogodama, te iscrtavanje samih opasnosti u vidu pravougaonika.

Koristeći *EnterpriseArhitect* alat koji se oslanja na rad sa *UML* jezikom, proširen je postojeći *UML* dijagram sa klasama za rad sa geoprostornim podacima. Takođe je kreiran i *CIM* profil koji definiše klase i njihova svojstva koje se koriste u aplikaciji.

Kako bi se omogućio rad sa poligonima vremenskih opasnosti, kao i simulacija istih, razvijeni su *WeatherService* i *GeoSparql* simulator za *WeatherService* koji pomeraju površine koje predstavljaju poligone opasnosti za definisani vremenski interval.

Upotrebom RDF TTL serijalizacije u vidu trojki (subjekat, predikat, objekat), uz oslonac na klase definisane CIM standardom, modelovani su geoprostorni i vremenski podaci koji su korišćeni za testiranje. Takođe je podržano i učitavanje RDF fajlova u okviru *SparqlPlayground* aplikacije kako bi se testirao rad *WeatherService*-a i *GeoSparql* simulatora u samoj aplikaciji.

5. IMPLEMENTACIJA

5.1 Proširenja potrebna za rad sa geoprostornim podacima

Definisan je dodatni klasni dijagram koji bi podržao rad sa klasama geoprostornih podataka. Sam klasni dijagram sa sobom donosi klase koje će služiti prilikom predstavljanja grafičkih podataka.

To su klase *CoordinateSystem*, *Location* i klasa *PositionPoint* koja u sebi sadrži informacije o koordinatama *x*, *y*, *z*.

5.2 Proširivanje CIMXML IEEE 37 Node Test Feeder modela sa geoprostornim podacima

S obzirom da CIMXML *IEEE 37 Node Test Feeder* model koji je korišćen u rešenju ne sadrži geoprostorne podatke, isti je proširen klasama koje omogućavaju rad sa geoprostornim podacima.

Dodate su klase *Location* i *PositionPoint* koje omogućavaju rad sa geoprostornim podacima. Klasa *Location* sadrži referencu na klasu *PowerSystemResource* preko koje je povezana sa klasama modela *IEEE 37 Node Test Feeder*. Takođe klasa *Location* sadrži referencu i na novododatu klasu *PositionPoint*. Klasa *PositionPoint* sadrži *xPosition* i *yPosition* atribute koja predstavljaju koordinate x i y. Na taj način bilo koji element može biti predstavljen tačkom u koordinatnom sistemu sa odgovarajućim x i y koordinatama.

5.3 Učitavanje RDF fajlova u aplikaciji SPARQL Playground

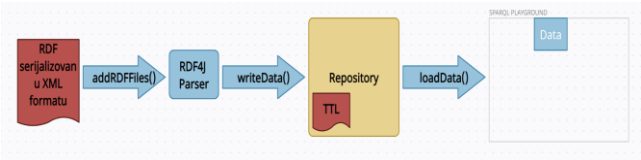
SPARQLPlayground aplikacija nudi mogućnost učitavanja podataka u vidu RDF-a, tj. neke od

serijalizacija RDF-a (konkretno TTL). Na samoj aplikaciji postoji tab sa imenom *Data* u koji se mogu smestiti podaci sa kojima se želi raditi.

Podaci se smeštaju u vidu trojki (subjekat-predikat-objekat), kako i nalaže sam RDF. Koristeći znanje iz prethodno kreiranog RDFS-a (*RDF Schema*), mogu se definisati trojke u aplikaciji *SPARQLPlayground*. Ugledajući se na podatke definisane u RDFS-u, na sličan način se mogu definisati podaci unutar *SparqlPlayground* aplikacije.

Prilikom pokretanja aplikacije iz *VisualStudioCode*-a poziva se metoda *addRDFFiles*() koja učitava podatke iz rdf-data foldera (koji su serijalizovani u XML format). Zatim se poziva i metoda *writeData*() koja na osnovu podataka koji su učitani iz rdf-data foldera, nakon parsiranja RDF podataka u TTL format uz pomoć predefinisanih *RDF4J* metoda, učitava podatke u *in-memory* repozitorijum. Proširenje u kodu predstavlja upravo *addRDFFiles*() metoda koja dozvoljava učitavanje podataka u RDF formatu.

Nakon učitavanja aplikacije *SPARQL Playground*, na predefinisanoj lokaciji, npr. http://localhost:8080 sa *frontend*-a se poziva metoda *loadData*() uz pomoć koje se učitavaju podaci u Data prozor iz *in memory* repozitorijuma. Dijagram toka učitavanja podataka prikazan je na Slici 1..



Slika 1. Dijagram toka učitavanja podataka

5.4 Izrada WeatherService simulatora

WeatherService omogućava učitavanje podataka o opasnostima koji su definisani u formatu TTL, kao i simulaciju samog pomeraja opasnosti kroz definisan interval.

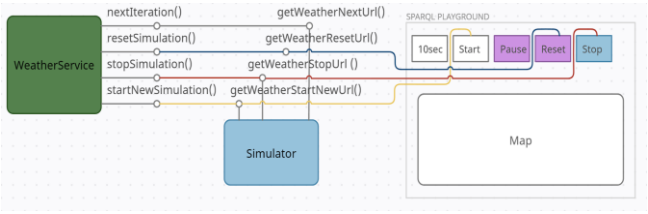
Servis ima sledeće opcije: *startNewSimulation*, gde se inicijalizuje model koji sadrži informacije o trojkama koje učestvuju u simulaciji, zatim *addNewStatements* koja služi za dodavanje novih trojki u repozitorijum kako bi *frontend* mogao da pročita nove vrednosti i, na kraju, *removeOldStatements*, gde se sa repozitorijuma brišu stare vrednosti trojki pre dodavanja novih.

Za potrebe simulacije, servis sadrži metode: *nextIteration*, gde se učitava sledeći fajl koji sadrži trojke koje su pomerene za određenu veličinu po y osi, zatim *resetSimulation*, koja služi kako bi simulacija ponovo počela i *stopSimulation*, gde se brišu svi izrazi iz repozitorijuma i radi se njegovo pražnjenje.

GeoSparql simulator koji radi merenje vremena sadrži i metode *start*, *stop*, *reset* i *pause*, koje kontrolišu rad merenja vremena i rad *WeatherService*-a. Implementirane su i metode *getWeatherStartNewUrl*, *getWeatherNextUrl*, *getWeatherStopUrl* i *getWeatherResetUrl*, koje kontrolišu pozivanje metoda sa *WeatherService*-a. Metode korišćene pri simulaciji WeatherService-a prikazane su na slici 2.

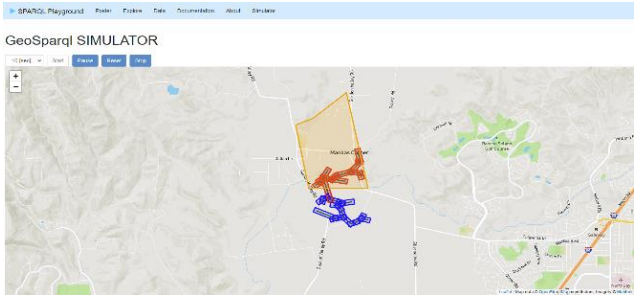
5.5 Izrada weather podataka

Do sada su definisani podaci za izvode, a definisan je i *WeatherService* koji će raditi sa podacima o opasnostima, kao i simulatori koji će omogućiti simulaciju rada sa podacima.



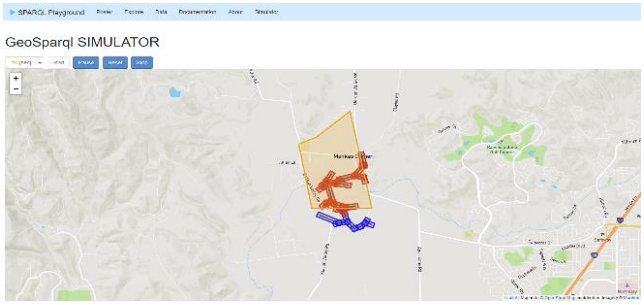
Slika 2. Metode korišćene pri simulaciji WeatherService-a

U ovom slučaju, za potrebe testiranja kreirana su 4 tipa dokumenta sa TTL podacima o opasnostima koji se kroz *WeatherService* smenjuju i tako daju utisak o pomeraju same opasnosti.



Slika 3. Slika sa početka simulacije

Nakon 10 sekundi, opasnost sa Slike 3. menja svoj položaj i na Slici 4. može se videti trenutno stanje.



Slika 4. Slika nakon 10 sekundi simulacije

5.6 Izrada alata za replikaciju CIMXML modela sa izmenjenim geoprostornim podacima

CIMXMLDuplicator je konzolna C# aplikacija koja vrši generisanje novih CIMXML datoteka na osnovu postojećih. Rad aplikacije se ogleda u nekoliko koraka. Pre svega se kopira postojeća datoteka, prođe se kroz sve koordinate i one se pomeraju za definisani *offset*. Generišu se i nove *mRID* i *Id* vrednosti, kako ne bi dolazilo do dupliranja podataka. O samom dizajnu aplikacije, kao i načinu rada, više informacija se može naći u radu [7].

5.7 Iscrtavanje elemenata zadatih u WKT formatu uz upotrebu Leaflet JavaScript biblioteke

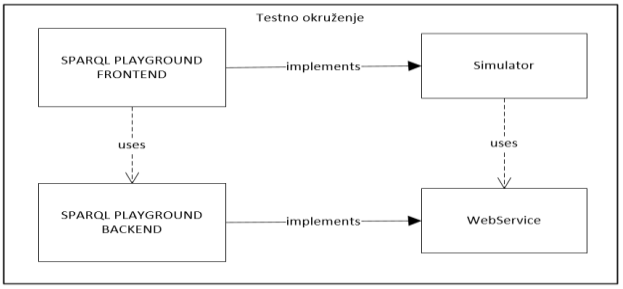
Izvode, preseke izvoda, opasnosti i sam pomeraj opasnosti potrebno je na neki način prikazati na mapama. Za to je korišćena biblioteka *Leaflet*. *Leaflet* podržava

različite funkcije mapiranja, te se podaci u WKT formatu lako prikazuju. *Well Known Text* (WKT) je OGC standard koji se koristi za predstavljanje prostornih podataka u tekstualnom formatu.

Više o samoj *Leaflet* biblioteci i njenom integrisanju u rešenje više informacija se može naći u radu [7].

6. TESTNO OKRUŽENJE

Na ***Slika 5.*** prikazan je raspored komponenti testnog okruženja u rešenju koje je implementirano.



Slika 5. Raspored komponenti testnog okruženja

SPARQL Playground aplikacija se sastoji od *SPARQL Playground frontend*-a uz pomoć kojeg je prikazan izgled *web* aplikacije koji je dostupan korisniku (klijentska strana). *SPARQL Playground frontend* sadrži raspored, grafiku, navigaciju i prikaz osnovnog sadržaja aplikacije. Između ostalog sadrži (*implements* na ***Slika 5.***) i Simulator *WeatherService*-a koji upravlja (*uses* na ***Slika 5.***) radom *WeatherService*-a. Pored toga postoji i *SPARQL Playground backend*, koji je zadužen za serversku stranu (onu stranu koja nije dostupna krajnjem klijentu). *SPARQL Playground backend* se sastoji od koda aplikacije, upotrebe različitih *framework*-a i njihove zajedničke integracije. Deo *backend*-a aplikacije je (*implements* na ***Slika 5.***) i sam *WeatherService* koji upravlja meteorološkim podacima.

SPARQL Playground frontend koristi (*uses* na ***Slika 5.***) *SPARQL Playground backend* za rad aplikacije, servisa i isporuku podataka.

7. ZAKLJUČAK

Rešenje koje je prikazano donelo je mogućnost iscrtavanja prostornih podataka, preseka distributivnih izvoda kao i vremenskih nepogoda na interaktivnoj mapi. Simulacija rada *WeatherService*-a daje osećaj pomeraja vremenskih nepogoda i njihovog uticaja na izvode u realnom vremenu.

Originalna *SPARQL Playground* aplikacija koja ne sadrži rešenja predstavljena u radu ima mogućnost rada sa SPARQL upitima, dok nova aplikacija sa razvijenim dopunama ima mogućnost rada sa *GeoSPARQL* upitima, mogućnost prikazivanja elemenata i simulaciju *WeatherService*-a.

Eventualno unapređenje aplikacije bila bi integracija postojećeg *WeatherService*-a sa *OpenWeather* API-em kako bi se dobijali podaci vremenskih nepogoda i prognoza u realnom vremenu.

Potencijalno unapređenje bi takođe moglo da se ogleda kroz modelovanje tipova opasnosti u skladu sa CIM standardom.

8. LITERATURA

[1] W3C, " SPARQL Query Language for RDF ," [Online]. Available https://www.w3.org/TR/rdf-sparql-query/

[2] Open Geospatial Consortium ,"OGC GeoSPARQL - A Geographic Query Language for RDF Data" [Online], Available: http://www.opengis.net/doc/IS/geosparql/1.0, 2012

[3] M. Gavrić, Standardi i modeliranje EES - beleške sa predavanja, Fakultet tehničkih nauka, Univerzitet u Novom Sadu, 2019.

[4] https://rdf4j.org (pristupljeno u martu 2022.)

[5] Sparx Systems, " EnterpriseArhitect," https://sparxsystems.com/products/ea/ (pristupljeno u martu 2022.).

[6] SPARQL Playground, [Online] SIB Swiss Institute of Bioinformatics, sparql-playgrund.sib.swiss/help/doc/about (pristupljeno u martu 2022.).

[7] P. Kaljević, Testno okruženje za graficki prikaz geoprostornih podataka na osnovu SPARQL upita, Novi Sad, 2021.

Kratka biografija:

Stefan Stojaković rođen je 01.09.1994. godine u Sremskoj Mitrovici. Osnovnu školu „Branko Radičević" u Šidu završio je 2009. godine. Završio je Gimnaziju „Sava Šumanović" u Šidu 2013. Fakultet tehničkih nauka je upisao 2014. godine, smer Elektroenergetski softverski inženjering. Godine 2019. završio je osnovne studije i upisao master akademske studije na Fakultetu tehničkih nauka, smer Primenjeno softversko inženjerstvo. Ispunio je sve obaveze i položio sve ispite predviđene studijskim programom.