Broj rada: R3-25
DOI:10.46793/CIRED2026.R3-25_270

EDITOR DISTRIBUTIVNE MREŽE SA PODRŠKOM ZA VIZUALIZACIJU CIM MODELA UZ OSLONAC NA LLM I SPARQL

EDITOR OF DISTRIBUTION NETWORK WITH SUPPORT FOR CIM MODEL VISUALIZATION LEVERAGING LLM AND SPARQL

Vladana ALEKSIĆ, Fakultet tehničkih nauka, Univerzitet u Novom Sadu, Novi Sad, Srbija
Milan GAVRIĆ*, Fakultet tehničkih nauka, Univerzitet u Novom Sadu, Novi Sad, Srbija
Goran ŠVENDA, Fakultet tehničkih nauka, Univerzitet u Novom Sadu, Novi Sad, Srbija

KRATAK SADRŽAJ

Elektroenergetski sistemi (EES) postaju sve složeniji, a time i zahtevniji za modelovanje. Njihov razvoj zahteva standardizovanu razmenu podataka koja podržava blagovremenu i pouzdanu izmenu modela. U tu svrhu koristi se CIM (*Common Information Model*) standard, podaci o izgrađenosti mreže se najčešće razmenjuju u CIMXML formatu. Pretraga podataka o mreži i detaljna analiza vrši se SPARQL upitima koji se izvršavaju nad RDF skladištem. Primena SPARQL-a zahteva poznavanje sintakse jezika upita i semantike modela, što ograničava jednostavnu interakciju krajnjih korisnika sa modelom mreže. U ovom radu je prikazan editor distributivne mreže (EDM) koji omogućava vizualizaciju CIM modela, interaktivan rad nad modelovanom mrežom, kao i dobijanje dodatnih informacija i izmene modela mreže korišćenjem prirodnog jezika (*Natural Language*) primenom SPARQL upita i LLM-a (*Large Language Model*). Razvijeni EDM kombinuje direktno izvršavanje predefinisanih SPARQL upita, koje obezbeđuje brz i pouzdan odgovor o podacima u modelovanoj mreži, sa mogućnostima lokalno pokrenutog LLM-a za obradu zahteva koji se ne oslanjaju na SPARQL upite. Podrška za proračune se oslanja na integraciju sa OpenDSS alatom. Mogućnosti praktične primene editora su testirane na primeru distributivnog izvoda sa 34 čvora - IEEE 34-bus Feeder.

Ključne reči: editor distributivne mreže, CIM, LLM, SPARQL

ABSTRACT

Power systems (PS) are becoming increasingly complex and therefore more demanding to model. Their development requires standardized data exchange that supports timely and reliable model modification. For this purpose, the CIM (*Common Information Model*) standard is used, while data on the network configuration are most commonly exchanged in the CIMXML format. Searching for network data and detailed analysis are obtained using SPARQL queries executed over an RDF store. The application of SPARQL requires knowledge of the query language syntax and the model's semantics, which limits simple interaction for end users with the network model. This paper presents a distribution network editor (EDM) that enables visualization of the CIM model, interactive work with the modeled network, and the ability to obtain additional information and modify the network model using natural language (Natural Language) through SPARQL queries and an LLM (*Large Language Model*). The developed

* Autor za korespondenciju / Corresponding author: Milan Gavrić, gavricm@uns.ac.rs

Public

EDM combines the direct execution of predefined SPARQL queries, which provides fast and reliable responses about the data in the modeled network, with the capabilities of a locally running LLM to process requests that do not rely on SPARQL queries. Support for calculations is based on the implemented integration with the OpenDSS tool. The possibilities for practical application of the editor were tested using the IEEE 34-bus Feeder as an example.

Keywords: CIM, Editor, LLM, Power distribution network, SPARQL

1 UVOD

Razvoj informacionih tehnologija je omogućio primenu novih pristupa u upravljanju i analizi distributivnih sistema. U ovom radu prikazan je pristup koji objedinjuje primenu LLM, SPARQL i RDF tehnologija u svrhu modelovanja distributivne mreže (DM).

Standardan pristup je podrazumevao primenu NoSQL tehnologija, na primer RDF za modelovanje DM i primenu SPARQL upita za dobijanje odgovarajućih informacija o modelu mreže [1]. Sve dodatne informacije su se nalazile u propratnoj dokumentaciji, koja je omogućavala pretragu, uglavnom, po ključnim rečima.

Razvoj LLM-a je dozvolio širu podršku upitima od značaja za modelovanje mreže, a njegova lokalna implementacija je omogućila jednostavnu postavku rešenja, bez dodatnih bezbednosnih rizika (kao što je pristup podacima u oblaku). Na ovaj način je omogućena jednostavnija interakcija korisnika sa modelom mreže i efikasnije izvršavanje različitih operacija prilikom izmene mreže ili traženja odgovora na pitanja od značaja za upravljanje mrežom.

Cilj rada je razvoj editora za vizualizaciju i izmene modela distributivne mreže EDM, pri čemu se korisniku omogućava rad sa modelom bez potrebe za direktnim poznavanjem detalja njegove složene unutrašnje strukture. Na ovaj način je podržana jednostavnija interakcija korisnika sa modelom mreže i omogućeno je efikasnije izvršavanje različitih operacija prilikom izmene mreže ili, na primer, traženje odgovora na pitanja od značaja za upravljanje mrežom.

Glavni doprinosi ovog rada su:

- DM modelovana u skladu sa CIM-om uz omogućene promene podataka i vizualizaciju grafa mreže,
- podrška komunikaciji zasnovanoj na prirodnom jeziku uz oslonac na lokalno postavljen LLM,
- ispitivanje mogućnosti objedinjavanja RDF, SPARQL i LLM tehnologija prilikom modelovanja DM u skladu sa CIM standardima.

Problem izmene modela DM definisan je u drugom delu rada. U trećem i četvrtom delu prikazani su osnovni alati, tehnologije, i arhitektura razvijenog EDM-a. Mogućnosti njegove primene prikazane su u petom delu. Nakon zaključka, prikazana je korišćena literatura.

2 POSTAVKA PROBLEMA

Modelovanje i upravljanje savremenim DM zahteva integrisanje razvijenih softverskih sistema za geografsko predstavljanje, upravljanje, nadzor, planiranje, analize i automatizaciju (kao što su SCADA, ADMS, OMS, GIS i dr.) koji postaju sve kompleksniji za upotrebu. Osnova svakog od ovih softverskih sistema i njihovih proračuna je model mreže, a naročito deo modela mreže od interesa za domen datog sistema. Različiti proračuni u realnom vremenu, simulacije i "*what-if*" analize zahtevaju da je domenski model mreže ažuran, i da može brzo i jednostavno da se promeni. Nažalost, čak i jednostavne izmene modela izgrađenosti mreže koji predstavlja osnov za sve druge sisteme (npr. da se doda novi element, ukloni postojeći, ili izmene njegovi podaci),

zahtevaju da se poznaju struktura CIM modela, međusobne veze između objekata i načina na koji se podaci čuvaju u RDF formatu. Pritom, svaka grupa izmena mora da se verifikuje, da li narušava konzistentnost modela i ispravnost relacija između elemenata, koje su neophodne za mnogobrojne analize i proračune. Praktično, korisnici (poput dispečera, inženjera koji rade na operativnom planiranju, ili izdavanju energetskih saglasnosti), nisu upoznati, i ne moraju da budu upoznati sa CIM standardom, RDF zapisom i načinom na koji se model ažurira na nivou implementacije. Samim tim, u postojećim rešenjima koja često u sebi sadrže zatvorene sisteme koji predstavljaju silose informacija, i najmanje izmene modela postaju komplikovane, zahtevaju podršku eksperata i vreme. To značajno otežava svakodnevne aktivnosti zaposlenih, od kojih se zahtevaju sve brže operativne odluke i sve kompleksnije analize.

Na primer, tradicionalni pristup izmeni modela, kao što je zamena transformatora, podrazumeva više uzastopnih koraka, angažovanje različitih stručnjaka i potencijalno rad u više softverskih sistema, pri čemu se postupak razlikuje u zavisnosti od toga da li se model menja za potrebe rada dispečera, ili za potrebe simulacija i analiza koje sprovode Smart Grid inženjeri. Samim tim nameće se potreba za editorom koji će omogućiti jednostavnu, brzu i pouzdanu izmenu modela DM uz mogućnost postavljanja dodatnih pitanja na maternjem jeziku. Pritom, primena tog editora ne sme da bude ograničena klijentovim znanjem o implementaciji CIM modela, načinu skladištenja podataka, mogućnostima generisanja upita nad podacima i slično.

3 OSNOVNI ALATI I TEHNOLOGIJE

Za modelovanje DM koristi se CIM standard - standardi serije IEC 61968 i IEC 61970, koji opisuju komponente, i njihove međusobne relacije u sistemu [2]. CIM klase predstavljaju apstraktni model mreže, pri čemu svaka klasa odgovara određenom fizičkom ili logičkom elementu mreže. Time je omogućeno da se stvarna infrastruktura predstavi kao objektno orijentisan model pogodan za analizu i razmenu podataka. Povezivanjem različitih elemenata, definisana je konektivnost cele mreže. Time je omogućeno da se uklanjanjem ili dodavanjem elementa minimalno utiče na ostatak sistema. Za razmenu modela, CIM koristi RDF (*Resource Description Framework*) [3], koji omogućava opisivanje klasa i njihovih međusobnih relacija na jedinstven način. Njegov osnovni cilj jeste obezbeđivanje jedinstvenog načina definisanja semantike radi lakše i tačnije razmene resursa.

SPARQL (*SPARQL Protocol and RDF Query Language*) predstavlja jezik upita specijalno dizajniran za rad sa RDF modelom podataka [4]. Sintaksom podseća na SQL upitni jezik, ali je prilagođen za rad sa RDF grafovima i RDF trojkama. Kao W3C standard usvojen je 2008.

U cilju lakše manipulacije sa podacima koriste se biblioteke Notation3 (N3.js), za parsiranje i rad sa RDF trojkama, i Comunica engine, za izvršavanje SPARQL upita nad RDF grafom. N3 proširuje RDF model uvođenjem promenljivih, lista i logičkih implikacija, čime je omogućena naprednija analiza grafova i podataka, zbog čega se često koristi u praktičnoj obradi RDF grafova [5]. Comunica podržava rad sa različitim tipovima izvora i izvršavanje upita nad više izvora istovremeno, pa se često koristi u situacijama gde podaci nisu smešteni na jednom mestu, već dolaze iz različitih sistema [6].

Large Language Model (LLM) je model veštačke inteligencije koji je obučen na velikim količinama tekstova kako bi mogao da razume i generiše prirodan jezik [7]. Zasniva se na dubokom učenju i neuronskim mrežama sa velikim brojem parametara, što mu omogućava da u tekstu prepoznaje obrasce, značenje i kontekst.

Ollama je aplikacija koja omogućava lokalno pokretanje preuzetih LLM modela [8]. Omogućava rad putem komandne linije i izbor modela koji se izvršava u pozadini. Jedan od

modela koji se može koristiti je Llama 3. Ovaj model donosi poboljšanja u procesu obučavanja i sposobnosti razumevanja konteksta teksta i predstavlja značajan napredak u primeni velikih jezičkih modela u različitim oblastima obrade prirodnog jezika [8].

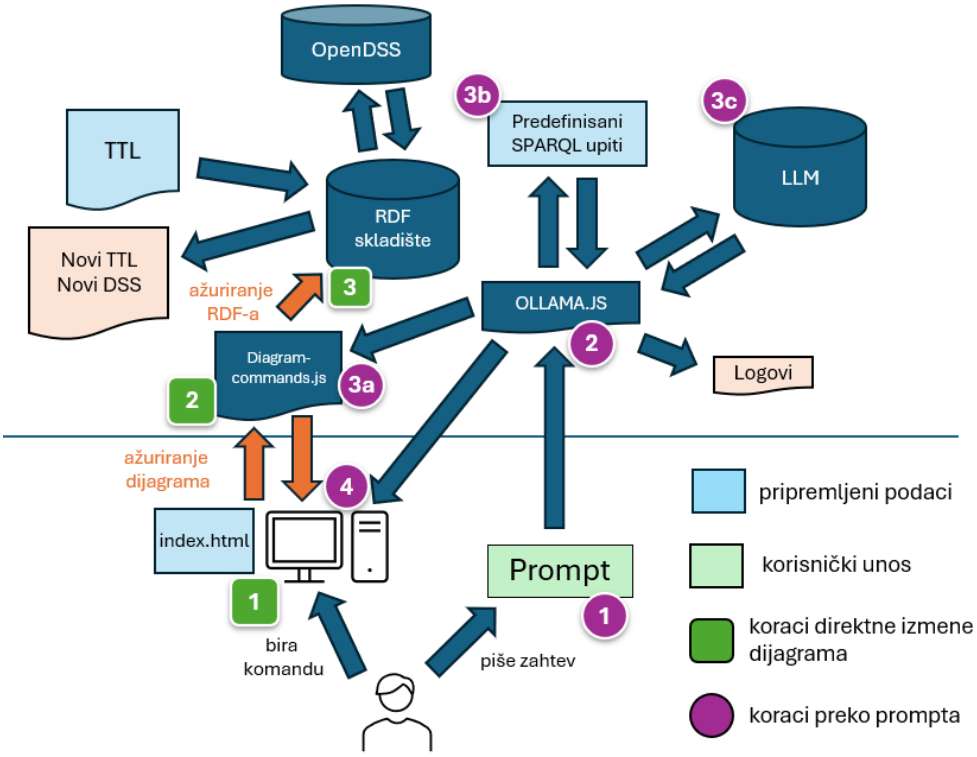
OpenDSS je alat koji omogućava izvršavanje složenih analiza nad modelom DM [9]. Omogućava simulaciju rada mreže uz uvažavanje svih izmena na modelu, kao što su, na primer, dodavanje ili uklanjanje DER elemenata. Razvijen je 1997. i od tada se uspešno koristi u industrijskim postrojenjima, istraživačkim centrima i drugim oblastima za analizu EES.

4 EDITOR DISTRIBUTIVNE MREŽE - EDM

Osnovna ideja EDM-a je objedinjavanje prethodno definisanih alata i tehnologija u jedinstven sistem (editor) koji omogućava brzu i tačnu izmenu modela DM. Pritom, cilj je da je primena tog sistema jednostavna. U nastavku je opisana arhitektura EDM-a i objašnjen način njegovog funkcionisanja, uz opis ulaznih i izlaznih tačaka.

4.1 Arhitektura EDM-a

Arhitektura EDM-a je prikazana na slici 1. Ona je organizovana tako da korisniku omogući dva osnovna načina rukovanja modelom mreže. Prvi način podrazumeva direktan izbor komandi kroz korisnički interfejs, dok se drugi zasniva na unosu zahteva prirodnim jezikom putem prompta. Bez obzira na način zadavanja zahteva, izmene se odražavaju na RDF skladište i grafički prikaz mreže, čime se održava njihova međusobna usklađenost.



Slika 1 - Prikaz arhitekture EDM-a

Direktan način rukovanja sistemom zasniva se na izboru komande u korisničkom interfejsu. Korak 1 predstavlja izbor odgovarajuće komande, odnosno operacije koju korisnik želi da izvrši nad dijagramom. Nakon izbora komande, korak 2 predstavlja njeno prosleđivanje delu sistema koji izvršava odgovarajuću operaciju nad SVG (*Scalable Vector Graphics*) dijagramom i ažurira njegov prikaz. Ovaj način je namenjen operacijama kod kojih je moguće unapred

definisati način izmene i ne zahteva obradu zahteva putem LLM modela. Korak 3 predstavlja ažuriranje RDF skladišta, odnosno modela mreže.

Prompt način rukovanja sistemom zasniva se na komunikaciji sa sistemom putem prirodnog jezika. Korisnik u okviru korisničkog interfejsa unosi prompt sa pitanjem ili zahtevom za izvršavanje određene operacije. Korak 1 predstavlja unos prompta, odnosno tekstualnog zahteva korisnika. U koraku 2 zahtev se prosleđuje fajlu *ollama.js*, koji predstavlja centralnu tačku za obradu zahteva zadatih prirodnim jezikom. Na osnovu sadržaja zahteva, *ollama.js* određuje način njegove dalje obrade i usmerava ga ka odgovarajućem načinu izvršavanja. Obrada se zatim nastavlja kroz korak 3a, 3b ili 3c.

U zavisnosti od sadržaja zahteva, obrada se nastavlja na jedan od tri načina. Korak 3a odnosi se na zahteve za direktnu izmenu dijagrama. U ovom slučaju zahtev se prosleđuje modulu *diagram-commands.js*, koji formira i izvršava odgovarajuću komandu za izmenu mreže. Korak 3b odnosi se na zahteve za koje postoji odgovarajući unapred definisani SPARQL upit. Tada se odabrani upit izvršava nad RDF skladištem, a dobijeni rezultat se vraća korisniku. Korak 3c se koristi kada zahtev nije moguće obraditi na prethodna dva načina. U tom slučaju LLM generiše odgovor na osnovu korisničkog zahteva i konteksta koji mu prosleđuje sistem, pri čemu može da koristi informacije o trenutnom stanju mreže. Korak 4 odnosi se na ažuriranje korisničkog interfejsa u vidu ažuriranog SVG dijagrama ili prikaza rezultata SPARQL upita, odnosno odgovora LLM-a.

Ovakva podela omogućava da se jednostavne i unapred definisane operacije izvršavaju direktno, bez nepotrebnog uključivanja LLM modela. Istovremeno, korišćenjem SPARQL upita za poznate upite nad modelom smanjuje se potreba za generisanjem odgovora pomoću LLM-a i omogućava se direktan pristup podacima iz RDF skladišta. LLM se tako koristi prvenstveno u situacijama u kojima je potrebna obrada prirodnog jezika ili kada zahtev nije moguće povezati sa unapred definisanom operacijom.

Primenom OpenDSS alata, omogućeni su osnovni proračuni, odnosno analiza trenutnog modela mreže – dodatno validacija modelove DM.

Nakon izmene modela, korisnik ima opciju da eksportuje model u .ttl i .dss format koji sadrži celokupnu strukturu mreže prikazanu na korisničkom interfejsu. Ova funkcionalnost omogućava da se trenutno stanje sistema sačuva, ponovo učita, ili dalje obrađuje u drugim kompatibilnim alatima.

4.2 Inicijalizacija i učitavanje podataka

Primena EDM-a započinje inicijalizacijom i učitavanjem podataka, slika 2.

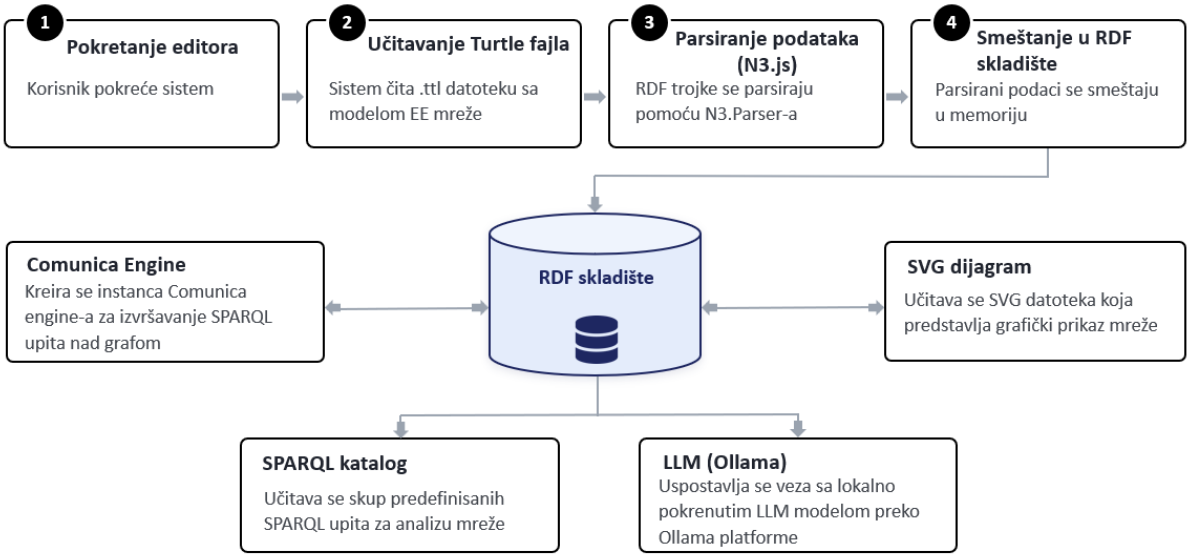
Prvi korak predstavlja pokretanje EDM-a i inicijalizaciju okruženja potrebnog za dalji rad sistema. Nakon pokretanja aplikacije pripremaju se osnovne komponente neophodne za učitavanje i obradu modela. Drugi korak predstavlja učitavanje *Turtle* (.ttl) datoteke koja sadrži model DM. Sistem pristupa datoteci i njen sadržaj prosleđuje biblioteci N3.js za dalju obradu. Treći korak predstavlja parsiranje učitanih podataka korišćenjem biblioteke N3.js. Tokom ovog procesa sadržaj *Turtle* datoteke transformiše se u niz RDF trojki pogodnih za dalju obradu. Četvrti korak predstavlja smeštanje podataka u RDF skladište. Sve RDF trojke se smeštaju u lokalno RDF skladište nad kojim se kasnije izvršavaju upiti i izmene.

Nakon inicijalizacije RDF skladišta, kreira se instanca Comunica engine-a koja omogućava izvršavanje SPARQL upita nad trenutnim stanjem modelovane mreže. U ovom slučaju N3.Store predstavlja RDF skladište u kojem se čuvaju RDF trojke tokom rada sistema. Za razliku od trajnih RDF baza podataka (*triple store*), koje se koriste za dugoročno čuvanje RDF

modela, podaci u ovom sistemu se učitavaju iz *Turtle* datoteke i čuvaju u memoriji EDM-a.

Pored učitavanja RDF modela, tokom inicijalizacije učitavaju se i ostali resursi neophodni za rad. Kreira se okruženje za izvršavanje SPARQL upita nad učitanim RDF modelom, dok se ostvaruje i komunikacija sa lokalno pokrenutim Llama 3 modelom, preko Ollama platforme. Temperatura modela predstavlja parametar koji utiče na stepen slučajnosti prilikom generisanja odgovora. Niže vrednosti temperature usmeravaju model ka konzistentnijim i predvidljivijim odgovorima, dok više vrednosti omogućavaju raznovrsnije i kreativnije generisanje teksta. U ovom sistemu temperatura je postavljena na vrednost 0.1, što predstavlja nisku vrednost i smanjuje mogućnost generisanja nepotrebnih ili netačnih informacija u smislu halucinacija. Na taj način odgovori modela su precizniji i prilagođeniji tehničkom domenu primene.

Nakon uspešno završene inicijalizacije, RDF skladište postaje dostupno svim komponentama sistema. Na taj način grafički prikaz mreže (SVG), SPARQL upiti i LLM model koriste isti RDF model, čime se obezbeđuje konzistentnost podataka tokom rada aplikacije.



Slika 2 - Inicijalizacija i učitavanje podataka

4.3 SPARQL katalog i NLP mapiranje

Katalog SPARQL upita predstavlja centralno mesto u kome su definisani svi upiti koje sistem koristi za pretraživanje CIMXML modela mreže. EDM sadrži predefinisane SPARQL upite od značaja za rad sa DM-om, ali korisnik može da definiše svoje SPARQL upite i da ih doda u katalog. Svaki upit predstavljen je kao par ključ-vrednost (*key-value*). Ključ predstavlja jedinstveni identifikator upita, dok vrednost sadrži odgovarajući SPARQL upit. Na ovaj način identifikator se može direktno iskoristi za pronalaženje i izvršavanje odgovarajućeg upita.

Upiti u katalogu obuhvataju različite aspekte CIM modela, kao što su broj i spisak instanci pojedinih klasa (npr. *ConnectivityNode*, *ACLineSegment* i *PowerTransformer*), topologija mreže i povezanost čvorova, parametri opreme (impedansa, naponi, strujne i naponske granice i mnoge druge), kao i informacije o potrošačima i izvorima napajanja.

Proces mapiranja zasniva se na prepoznavanju ključnih reči i obrazaca u korisničkom upitu pomoću regularnih izraza (*RegEx*). Analizom korisničkog unosa sistem prepoznaje specifične pojmove i izraze, na osnovu kojih razlikuje zahteve. Nakon toga, na osnovu prepoznatih obrazaca, pronalazi odgovarajući upit iz SPARQL kataloga koji će biti izvršen nad RDF skladištem. Na ovaj način korisnik ne mora direktno da zadaje SPARQL upite, već sistem

automatski povezuje prirodni jezik sa odgovarajućim upitom.

4.4 Hibridni način pripreme odgovora – SPARQL i LLM režim

Kako bi sistem korisniku pružio brze i pouzdane odgovore, realizovan je hibridni mehanizam koji kombinuje direktno preuzimanje podataka iz CIM modela putem SPARQL upita sa generisanjem odgovora pomoću LLM modela. U zavisnosti od rezultata obrade korisničkog zahteva, sistem radi u jednom od sledeća dva režima: SPARQL ili LLM.

SPARQL režim predstavlja slučaj u kome korisnikov upit uspešno pronalazi odgovarajući SPARQL upit iz kataloga, a njegovim izvršavanjem nad RDF modelom dobija se rezultat. U tom slučaju, dobijeni podaci se direktno prikazuju korisniku, bez pozivanja LLM modela. Na ovaj način odgovor se dobija brže i zasniva se isključivo na podacima iz CIM modela.

LLM režim se aktivira u situacijama kada nije moguće dobiti direktan odgovor iz RDF modela. To se dešava kada se za korisnikov upit ne pronađe odgovarajući SPARQL upit, kada RDF skladište još nije spremno za rad, ili kada izvršavanje upita ne može da vrati rezultat. U tom slučaju, zahtev se prosleđuje LLM modelu zajedno sa dodatnim kontekstom koji, pored korisničkog pitanja, može da sadrži rezultat poslednjeg uspešno izvršenog SPARQL upita, trenutno stanje modela mreže i informacije o trenutno izabranom čvoru.

4.5 SVG vizualizacija i korisnički interfejs

Korisnik komunicira sa sistemom preko grafičkog korisničkog interfejsa koji prikazuje mrežu u vidu SVG dijagrama. Na dijagramu su prikazani svi elementi sistema i njihova međusobna povezanost. Neki od elemenata koje EDM prikazuje su: *ConnectivityNode*-ovi, *ACLineSegment*-i, regulatori, transformatori, kao i *spot loads* (žuti krugovi) i *distributed loads* (žute linije). Korisnik može da bira elemente dijagrama, pregleda njihove informacije i vrši izmene nad njima. Pored dijagrama, nalazi se deo korisničkog interfejsa preko kojeg korisnik može prirodnim jezikom da postavlja pitanja ili zadaje naredbe za izmenu mreže koje se uporedo odnose i na izmene SVG dijagrama. Sve izvršene izmene se prikazuju na dijagramu, tako da je korisniku uvek dostupan trenutni prikaz mreže.

4.6 OpenDSS

Ovaj alat omogućava uvid u osnovne analize EES, kao što su proračun tokova snaga, gubitaka, naponskih profila i drugi relevantni parametri. Trenutno se svakom novododatom elementu mreže automatski dodeljuju unapred definisane vrednosti. Dodavanja novih elemenata, i njihovo uklanjanje utiču na rezultate proračuna. Analize se pokreću pritiskom na dugme „Run OpenDSS“, nakon čega se rezultati prikazuju u vidu poruke u korisničkom interfejsu.

4.7 Ažuriranje modela mreže i eksportovanje .ttl i .dss fajlova

Prilikom svake izmene modela potrebno je ažurirati i RDF model, kako bi struktura grafa odgovarala stanju prikazanom na SVG dijagramu. Na taj način se SPARQL upiti, prikaz podataka i izvoz modela izvršavaju nad aktuelnim modelom mreže. Za svaki novi element generiše se jedinstveni IRI (*Internationalized Resource Identifier*) identifikator oblika urn:uuid, koji se upisuje u RDF graf i čuva u lokalnom kešu.

Na primer, prilikom dodavanja novog čvora, pored samog čvora, u RDF graf se dodaje i vod koji ga povezuje sa prethodno izabranim čvorom. Za vod se kreira instanca klase *ACLineSegment*, kojoj se dodeljuju naziv u formatu l_ui_<broj čvora> i dužina, npr. 100 m. Uz

vod se kreiraju *PerLengthPhaseImpedance*, *ACLineSegmentPhase* i *PhaseImpedanceData*, sa unapred definisanim vrednostima parametara impedanse. Povezivanje voda sa čvorovima ostvaruje se preko dva *Terminal* elementa, pri čemu se IRI postojećeg čvora pronalazi u RDF grafu, a IRI novog čvora preuzima iz lokalnog keša.

Uz novi čvor može da se kreira *Load* ili *Solar* element. Veza sa čvorom se ostvaruje preko *Terminal* elementa. Za *Load* se instanci klase *EnergyConsumer* dodeljuje naziv load_<broj čvora>, dok se svakoj fazi dodeljuju aktivna i reaktivna snaga, npr. p = 100 kW i q = 50 kVar. Za *Solar* element instanci klase *PowerElectronicsConnection* se dodeljuju ratedS = 50 kVA i ratedU = 24,9 kV, dok *PhotovoltaicUnit* dobija maxP = 50 kW. Za odgovarajuću fazu definiše se aktivna snaga, npr. p = 50 kW.

Po završetku rada, korisnik može da izveze trenutno stanje RDF modela u *Turtle* (.ttl) ili .dss format. Na taj način korisnik može da sačuva ažurirani RDF model koji sadrži sve izmene izvršene tokom rada EDM-a, uključujući dodavanje, brisanje i preimenovanje čvorova, kao i dodavanje i brisanje njegovih elemenata (npr. *Load* i *Solar*).

5 PRIMER PRIMENE

Mogućnosti EDM-a prikazane su na IEEE test mreži sa 34 čvora [10]. Iako su podaci o mreži javno dostupni, odgovarajući zapis modela nije postojao, pa je za dobijanje početnog modela korišćen OpenDSS i njegova mogućnost eksportovanja modela u CIM. Testovi su realizovani na standardnom prenosnom računaru: procesor (CPU) 12th Gen Intel(R) Core(TM) i5-12450H, radna memorija (RAM) 16 GB LPDDR5, grafička kartica (GPU) Intel(R) UHD Graphics, disk SSD NVMe 512 GB, operativni sistem Windows 11 Pro (verzija 25H2). LLM se izvršava isključivo na procesoru (CPU), bez GPU ubrzanja, što direktno utiče na vreme generisanja odgovora. U nastavku su prikazana dva primera. Primer 5.1 – dva režima rada pri izvršavanju upita vezanih za model mreže, i primer 5.2 – kako korisnik može da modifikuje mrežu.

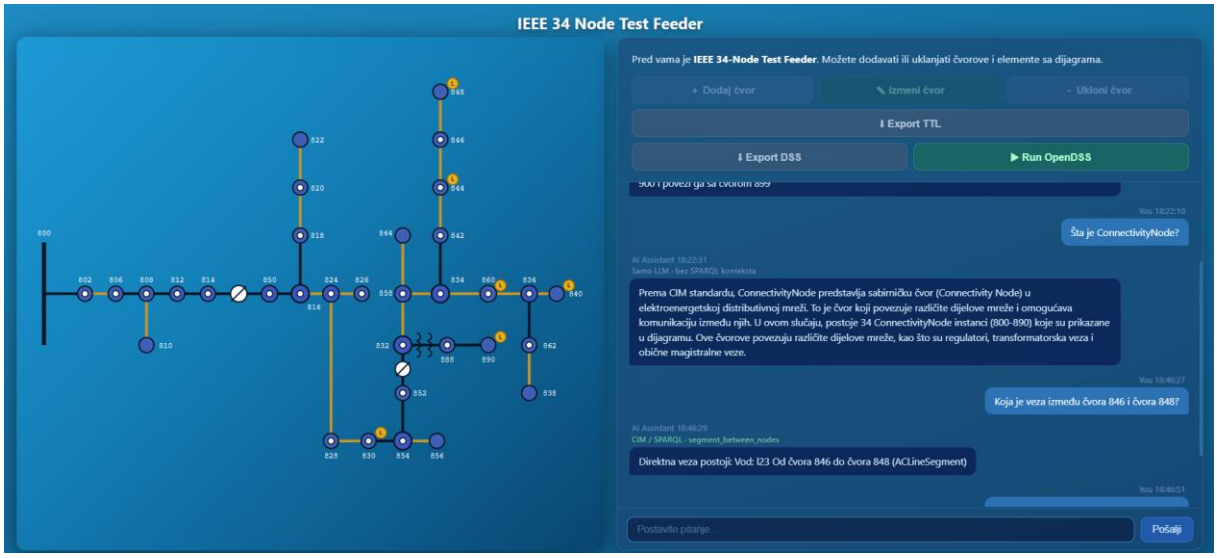
5.1 SPARQL režim i LLM režim

Na slici 3, prikazana su dva primera rada sistema nad modelom IEEE 34 test mreže. Prvi primer predstavlja upit "*Šta je ConnectivityNode*?", na koji sistem odgovara korišćenjem LLM režima. Pošto za ovakav zahtev nije pronađen odgovarajući upit u katalogu SPARQL upita, zahtev se prosleđuje LLM modelu, koji generiše odgovor na osnovu prosleđenog konteksta. Generisanje odgovora započelo je nakon 2,3 s, dok je ukupno vreme odgovora iznosilo 21,2 s.

Drugi primer predstavlja upit "*Koja je veza između čvora 846 i čvora 848*?". Ovaj zahtev sistem prepoznaje kao upit za pronalaženje direktne veze između dva konkretna čvora i izvršava ga u SPARQL režimu. Pre slanja zahteva LLM modelu, sistem proverava da li tekst odgovara nekom od unapred definisanih obrazaca za direktno izvršavanje SPARQL upita. U ovom slučaju prepoznaju se brojevi čvorova 846 i 848, kao i izrazi "veza" i "između", koji ukazuje na zahtev za pronalaženje veze između dva čvora. Na osnovu toga bira se SPARQL upit *segment_between_nodes*, pri čemu se brojevi čvorova prosleđuju kao njegovi parametri. Upit se zatim izvršava nad RDF modelom i rezultat se direktno vraća korisniku, bez pozivanja LLM modela. Vreme izvršavanja ovog upita iznosilo je 2,75 s.

Ova dva primera pokazuju osnovnu razliku između LLM i SPARQL režima rada. Dok LLM režim omogućava odgovaranje na zahteve koji nisu obuhvaćeni unapred definisanim SPARQL upitima, SPARQL režim koristi direktno izvršavanje SPARQL upita kada je zahtev moguće prepoznati. U oba slučaja odgovori odgovaraju podacima iz modela, ali je vreme izvršavanja značajno različito, što pokazuje prednost direktnog izvršavanja SPARQL upita za zahteve koji

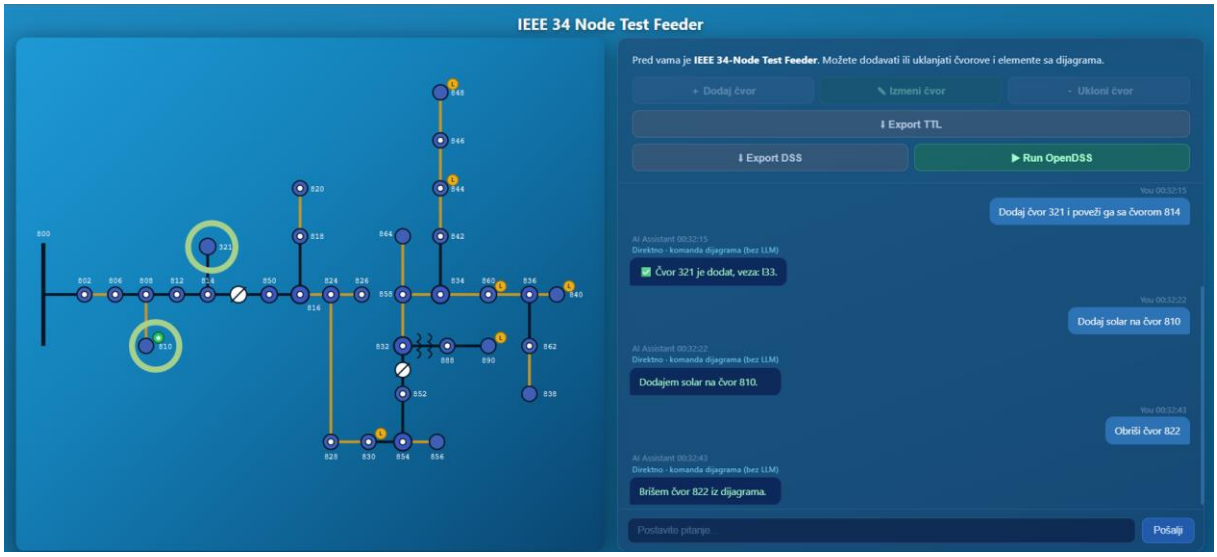
se mogu unapred definisati.



Slika 3 – LLM i SPARQL režimi

5.2 Izvršavanje operacija nad dijagramom

Na slici 4 prikazane su komande za izmenu dijagrama: "*Dodaj čvor 321 i poveži ga sa čvorom 814*", "*Dodaj solar na čvor 810*" i "*Obriši čvor 822*". Prilikom dodavanja novog čvora, sistem automatski kreira vod između novog i prethodno navedenog čvora, pri čemu se koriste unapred definisani parametri voda, poglavlje 4.7. Isto tako, prilikom dodavanja solarnog elementa koriste se unapred definisani parametri, poglavlje 4.7. Svaki novododati element upisuje se u RDF skladište zajedno sa ostalim neophodnim elementima, čime se održava konzistentnost modela. Nakon komande, izmena modela se izvršava i rezultat je vidljiv na dijagramu.



Slika 4 - Prikaz izvršenih komandi

Na ovaj način skraćuje se vreme obrade zahteva i izbegavaju se problemi koji mogu nastati usled različitih formata odgovora koje generiše LLM model. Samo u slučaju da korisnički unos ne odgovara nijednom lokalno prepoznatom obrascu, obrada se nastavlja standardnim tokom, koji podrazumeva prepoznavanje SPARQL upita i, po potrebi, pozivanje LLM modela.

Opisane izmene moguće je realizovati i ručnim putem, korišćenjem dostupnih opcija u korisničkom interfejsu. Za izvršavanje ovih operacija potrebno je prethodno izabrati odgovarajući čvor na dijagramu nad kojim se vrši izmena ili na osnovu kog se dodaje novi element. Klikom na dugme "*Dodaj čvor*" otvara se pomoćni panel u okviru kojeg korisnik unosi broj novog čvora, pri čemu sistem vrši proveru da li takav čvor već postoji u modelu. Nakon unosa podataka moguće je definisati da li novi čvor sadrži *Load* i/ili *Solar* elemente. Dugme "*Ukloni čvor*" omogućava direktno uklanjanje prethodno izabranog čvora, zajedno sa elementima koji su sa njim povezani, dok dugme "*Izmeni*" otvara poseban panel za izmenu podataka postojećeg čvora (promeniti broj čvora, dodati *Load* i/ili *Solar*).

6 ZAKLJUČAK

U ovom radu je opisan Editor Distributivne Mreže - EDM koji omogućava prikaz i editovanje CIM modela mreže, rad sa elementima kroz grafički interfejs i postavljanje pitanja prirodnim jezikom. U tu svrhu napravljen je hibridni pristup, koji povezuje različite tehnologije, pre svega SPARQL i LLM. Primenom SPARQL upita, ili LLM-a, omogućeno je da korisnik jednostavno i brzo dobije željene informacije o mreži, ili promeni postojeći model DM (npr. da doda novi, ili obriše postojeći element, ili promeni njegove podatke). Dodatno je implementirano lokalno prepoznavanje najčešćih komandi za izmene dijagrama, čime je izbegnuto nepotrebno pozivanje LLM-a za jednostavne operacije, odnosno dodatno ubrzan rad editora. Pritom, svaka promena na dijagramu se automatski prenosi u RDF skladište, tako da se svi naredni upiti izvršavaju nad trenutnim stanjem mreže. Sve izmene se čuvaju, što omogućava da su grafički prikaz i RDF model uvek usklađeni. Izmenjeno RDF skladište može se izvesti u .ttl i .dss format, kako bi korisnik mogao mrežu da nezavisno koristi i u nekim drugim sistemima.

Mogućnosti editora su prikazane na primeru DM, ali je primena moguća i na sve slične sisteme, na primer na telekomunikacionu mrežu, gasnu infrastrukturu, vodovod, itd.

LITERATURA

[1] T. McDermott, E. Stephan, T. D. Gibson, "Alternative Database Designs for the Distribution Common Information Model", *2018 IEEE/PES T&D Conference and Exposition*, Denver, CO, USA, 2018, pp. 1-5, doi: 10.1109/TDC.2018.8440470.
[2] IEC 61970–301: "Common information model (CIM) base", IEC, Geneva, Switzerland.
[3] W3C Recommendation, "Resource Description Framework", 1999: http://www.w3.org/TR/REC-rdf-syntax/; pristupljeno 2026-06-10.
[4] Open Data Support, European Commission, "Introduction to RDF & SPARQL": https://data.europa.eu/sites/default/files/d2.1.2_training_module_1.3_introduction_to_rdf_sparql_en_edp.pdf; pristupljeno 2026-03-12.
[5] Notation3, "Notation3: Knowledge Graph Language for the Semantic Web": https://notation3.org/; pristupljeno 2026-03-12.
[6] R. Taelman, J. V. Herwegen, M. Van Sande and R. Verborgh, "Comunica: a Modular SPARQL Query Engine for the Web", 2018: https://comunica.github.io/Article-ISWC2018-Resource/; pristupljeno: 2026-03-12.
[7] Microsoft, "What are LLMs?": https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-are-large-language-models-llms; pristupljeno 2026-06-10.
[8] C. Bronsdon: "Exploring Llama 3 Models: A Deep Dive", 2025: https://galileo.ai/blog/exploring-llama-3-models-a-deep-dive; pristupljeno 2026-06-10.
[9] EPRI, "OpenDSS": https://www.epri.com/pages/sa/opendss; pristupljeno 2026-03-12.
[10] IEEE, "IEEE 34 Node Test Feeder", https://cmte.ieee.org/pes-testfeeders/wp-content/uploads/sites/167/2017/08/feeder34.zip, pristupljeno 2026-02-12.