Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.41
DOI: https://doi.org/10.24867/09BE02Stankovic

RAZVOJ APLIKACIJE ZA SIMULACIJU RADA PROTOTIPA DISTRIBUTIVNOG ELEKTROENERGETSKOG SISTEMA U SERVICE FABRIC OKRUŽENJU

DEVELOPMENT OF AN APPLICATION FOR SIMULATION OF THE PROTOTYPE OF DISTRIBUTION POWER SYSTEM ON SERVICE FABRIC PLATFORM

Dragan Stanković, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIČKO I RAČUNARSKO INŽENJERSTVO

Kratak sadržaj – *U ovom radu predstavljena su dva programska rešenja simulacije rada distributivnog elektroenergetskog sistema – on-premise i Azure Service Fabric. Izvršena je implementacija, testiranje, analiza i poređenje performansi između dva pomenuta rešenja.*

Ključne reči: *Distributivni elektroenergetski sistem, Service Fabric, Smart Grid, Azure*

Abstract – *This paper presents two software solutions for simulating the distribution power system - on-premise and Azure Service Fabric. Implementation, testing, analysis and performance comparison between the two mentioned solutions was performed.*

Keywords: *Distributed power system, Service Fabric, Smart Grid, Azure*

1. UVOD

Kako bi se postigla odgovarajuća pouzdanost i kvalitet u isporuci električne energije krajnjim potrošačima, elektroenergetski sistem je podeljen na četiri osnovne celine: proizvodnja, prenos, distribucija i potrošnja. Na ovaj način došlo je do podele odgovornosti i razdvajanja procesa gde svaki podsistem brine o ispunjavanju zadataka iz svog delokruga poslova. U svakoj celini je došlo do razvoja odgovarajućih alata za vođenje tehničkih poslova kako bi se ostvarili što bolji rezultati. Tako sa ubrzanim razvojem tehnike i tehnologije, hardverskih i softverskih rešenja i telekomunikacione opreme dolazi do velikog napretka u domenu nadzora i upravljanja infrastrukturnim sistemima kao što je npr. distributivna mreža. Ukoliko se posmatra model i broj elemenata koji je karakterišu, distributivna mreža predstavlja najkompleksniji podsistem elektroenergetskog sistema [1].

Napredak i modernizacija su omogućili uvođenje koncepta pametnih mreža u elektroenergetski sistem. Pametna mreža (eng. *Smart Grid*) predstavlja automatizovanu mrežu koja nadgleda, štiti i optimizuje sve elemente sistema, od generatora pa sve do potrošačkih čvorova. Donosi brojne benefite poput korišćenja distribuiranih generatora, smanjenja zagađenja životne sredine, identifikacija grešaka u radu itd. [2].

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Darko Čapko, vanr. prof.

Za razliku od tradicionalnih elektroenergetskih sistema, pametne mreže sa sobom donose softverska rešenja koja za cilj imaju kontrolu rada distributivne mreže. Aplikacije omogućuju dispečerima da na jednostavan način prate, kontrolišu i upravljaju delom mreže na koju je softver primenjen. Praćenje potrošnje, lokalizacija kvara, različiti proračuni od značaja, energizacija i deenergizacija određenih područja su samo neki od benefita koje nude aplikacije iz oblasti elektroenergetike.

Prosperitet u svetu informacionih tehnologija je evidentan iz dana u dan i ide toliko daleko, da omogućava da se veliki infrastrukturni sistemi, kakva je i sama distributivna mreža, obuhvate i obrade u moćnim i udaljenim servisima koristeći *Cloud* tehnologije. Jedno od solucija za tu vrstu realizacije je *Service Fabric*, distribuirana sistemska platforma koja deli sistem na mikroservise i pojednostavljuje upravljanje aplikacijama u *Cloud* okruženju. Najveće prednosti *Service Fabric*–a su pouzdanost, skalabilnost i dostupnost [3].

Ovaj rad je organizovan u sedam poglavlja. U prvom poglavlju je dat uvod. Drugo poglavlje sadrži teorijske osnove distributivne mreže elektroenergetskog sistema i *Service Fabric* okruženja. Treće poglavlje daje uvid u predmet istraživanja. Način implementacije i arhitektura sistema se nalaze u poglavlju četiri. Testiranje i analiza rešenja su opisani u poglavlju pet. Zaključak je napisan u poglavlju šest, a korišćena literatura u poglavlju sedam. Kratka biografija se nalazi u poglavlju osam.

2. TEORIJSKE OSNOVE

2.1. Tradicionalni elektroenergetski sistemi

U dvadesetom i početkom dvadeset prvog veka su primenjivani tradicionalni koncepti i tehnike da bi se obezbedilo snabdevanje potrošača električnom energijom. Proizvodnja električne energije se odvijala u elektranama (hidro, termo, nuklearna) koje su izgrađene neposredno u blizini primarnih izvora energije, udaljene od naseljenih mesta. Smanjenje uticaja štetnih gasova na stanovništvo i određena vrsta prevencije od mogućih katastrofa su glavni razlog njihovog distanciranja u odnosu na naselja. Pored načina proizvodnje električne energije, glavna odlika tradicionalnih elektroenergetskih sistema je nizak stepen automatizacije. Otklanjanje kvarova se vršilo isključivo slanjem dežurne ekipe na teren i njihovim intervencijama. Evidentiranje utrošene električne energije je takođe bio zadatak čoveka. Obavljao se ručno, na mestu potrošnje,

očitavanjem vrednosti sa konvencionalnih električnih brojila.

2.2. Pametne mreže

Električna energija se nametnula kao vodeća stavka u svakodnevom životu ljudi i činjenica je da je život bez nje postao nezamisliv. Nemerljiv uticaj na čovečanstvo i okolinu i potražnja za njom su doveli do toga da tradicionalni elektroenergetski sistemi postaju neodrživi u takvim uslovima. Međutim, razvoj komunikacionih tehnologija, softverskih alata, nadzorno-upravljačkih sistema, mernih instrumenata i drugih naprednih sistema u oblasti elektroenergetike, doveo je do transformacije tradicionalnog elektroenergetskog sistema u pametnu mrežu.

Koncept pametnih mreža se zasniva na automatizovanom rukovođenju mrežom u realnom vremenu, uz posredstvo odgovarajuće opreme, adekvatnog nadzorno-upravljačkog sistema i softvera za upravljanje. Proizvodnja električne energije se, osim iz osnovnih tipova elektrana, dobija i iz tzv. alternativnih elektrana koje koriste obnovljive izvore kao što su: vetar, sunce i biomasa. Ova vrsta proizvodnje je uglavnom vezana za distribuirane generatore koji se najčešće priključuju na distributivnu mrežu, u blizini potrošača. Na taj način se povećava pouzdanost snabdevanja, doprinosi se očuvanju okoline i klime, a energija se proizvodi u blizini potrošnje [2].

Veliki pomak ka automatizaciji procesa je obezbeđen uvođenjem senzora, aktuatora i pametnih brojila. Tako je obezbeđen uvid u realno stanje sistema u svakom trenutku, uz mogućnost delovanja sa udaljenog mesta, posredstvom odgovarajuće aplikacije.

2.3. *Service Fabric*

Azure Service Fabric je distribuirana sistemska platforma razvijena od strane kompanije *Microsoft*. Omogućava stvaranje skalabilnih i pouzdanih aplikacija sastavljenih od mikroservisa koji rade na mnoštvu povezanih računara. Grupa računara, koji su tako povezani i pritom funkcionišu i ponašaju se kao jedan skladan sistem, naziva se klaster (eng. *Cluster)*. Ova platforma pruža sveobuhvatne mogućnosti upravljanja aplikacijama zasnovanim na arhitekturi mikroservisa. Smešta ih unutar kontejnera koji su raspoređeni širom *Service Fabric* klastera.

Mikroservisi mogu biti *stateless* ili *stateful*. *Stateless* servisi ne čuvaju lokalno promenljiva stanja. Uglavnom se potrebni podaci za rad servisa čuvaju eksterno, u bazi podataka. Instance *stateless* servisa se raspoređuju po čvorovima i u slučaju otkaza bilo koje instance ili čvora na kojoj se instanca nalazi, vrši se ponovno startovanje na ispravnom čvoru. *Stateful* servisi održavaju promenljivo stanje. Programski kod se izvršava samo na jednom čvoru koji se naziva primarni, dok su ostali čvorovi određenog servisa pasivni.

U toku rada, primarni čvor konstantno replicira podatke na sekundarne čvorove i na taj način održava konzistentno stanje sistema. Odnosno, u slučaju otkaza primarnog čvora, jedan od sekundarnih preuzima posao i nastavlja njegovo izvršavanje. Takođe, vrši se oporavak čvora koji je otkazao, pri čemu se on formira kao sekundarni čvor. Instance *stateful* servisa se nazivaju replike i njihov broj se može menjati, ali uvek je samo jedna od njih primarna [3].

3. PREDMET ISTRAŽIVANJA

Aplikacija predstavlja programsko rešenje koje služi za nadzor, kontrolu i upravljanje distributivnim delom mreže. Može da obrađuje više različitih distributivnih mreža. Predmet istraživanja su efikasnost i performanse aplikacije u lokalnom i u *Service Fabric* okruženju čiji je cilj objašnjenje da li prelazak na *Cloud* bazirano rešenje nudi bolje performanse u odnosu na *on-premise* rešenje. Kada je aplikacija pokrenuta lokalno svi njeni servisi su pokrenuti na jednom računaru i koriste njegove hardverske resurse. U *Service Fabric* okruženju dolazi do podele poslova, odnosno određeni servisi rade na virtuelnim mašinama koje pripadaju geografski udaljenom klasteru, sa različitim konfiguracijama hardvera u odnosu na lokalni računar, dok se određeni delovi programskog koda izvršavaju na lokalnoj mašini.

4. IMPLEMENTACIJA APLIKACIJE

4.1. Opis rada aplikacije

Na osnovu dnevne krive potrošnje električne energije se vrši simulacija potrošnje gde svaki potrošač koristi određenu snagu u zavisnosti od satnice, odnosno dela dana. Tako će potrošnja kada ljudi spavaju, npr. u 03:00h, biti mnogo manja nego kada se ljudi spremaju za posao, npr. u 07:00h. Simulacija dnevne potrošnje funkcioniše tako što se svakom potrošaču postavlja određena snaga koja se menja na svakih *n* sekundi. Svaka iteracija predstavlja sat vremena stvarnog života.

Potrošnja, u bilo kom delu dana, ne sme da naruši maksimalno dozvoljenu. Ne sme da dođe do strujnog preopterećenja u distributivnoj mreži. Strujno opterećenje se računa sumiranjem snaga potrošača u tzv. *korenu* (skup elemenata mreže napajanih sa istog izvora). Zatim se taj zbir deli sa naponom na izvoru kako bi se dobila jačina struje, a potom se ista poredi sa limitom sekcije koja provodi struju. Dobijena vrednost ne sme biti veća u odnosu na propisanu jačinu električne struje koju vod može da provodi.

Algoritam kružnih isključenja vodi računa o ravnomernoj restrikciji električne energije između potrošača u slučaju strujnog preopterećenja, striktno obraćajući pažnju na njihov prioritet. Npr. bolnica je potrošač najvišeg prioriteta i ne sme ostati bez napajanja, dok su obična domaćinstva najnižeg prioriteta i oni će prvi ostati bez električne energije u slučaju prekomernog korišćenja iste. Isto tako, algoritam vodi računa da vreme provedeno bez napajanja između potrošača istih prioriteta bude ujednačeno.

U svakoj iteraciji simulacije se uzimaju vrednosti potrošnje za naredni sat koji su dobijeni dnevnom krivom potrošnje. Sa tim vrednostima se vrši proračun potrošnje mreže kako bi se preventivno delovalo na sistem, da ne dođe u stanje strujnog preopterećenja. Drugim rečima, aplikacija u tekućoj iteraciji, po potrebi, vrši isključenja

potrošača kako bi mreža u narednoj iteraciji ostala u normalnom stanju.

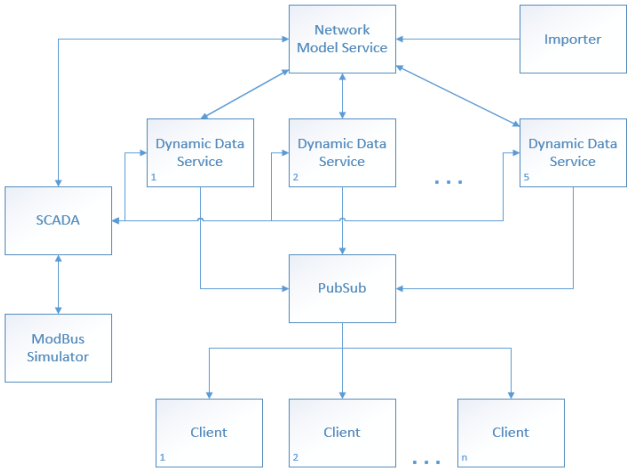
4.2. Arhitektura sistema

Arhitektura sistema *on-premise* aplikacije se razlikuje u odnosu na aplikaciju iste funkcionalnosti koja je pokrenuta u *Servic Fabric* okruženju. Implementacija u *Service Fabric* okruženju je zahtevala određene izmene zbog pravilnog rutiranja pristiglih poruka unutar klastera.

4.2.1. Arhitektura aplikacije u lokalnom okruženju

Sistem je podeljen u nekoliko komponenti. Svaka od njih ima svoja zaduženja i zadatke koje ispunjava nezavisno od ostalih komponenti sistema. Arhitektura je prikazana na slici 1, odakle se može videti da je sistem izgrađen od sledećih elemenata:

- *Importer*
- *Network Model Service* – NMS
- *Dynamic Data Service* – DDS
- *Supervisory Control And Data Acquisition* – SCADA
- *PublishSubscribe* - PubSub
- *Modbus* simulator
- *Client*



Slika 1. *Arhitektura on-premise aplikacije*

Importer - učitava CIM XML dokument u kom se nalaze svi elementi elektroenergetske mreže i informacije o njima. Na osnovu pomenutog dokumenta pakuje podatke u pogodan format i šalje ih *Network Model Service*-u.

NMS - servis u kom se čuva statika sistema. Prima podatke o elektroenergetskoj mreži od *Importer*–a na osnovu kojih kreira objekte. DDS servisu i SCADA sistemu šalje sve jedinstvene identifikatore elemenata u mreži. Ukoliko je pokrenuto više DDS instanci u sistemu, podaci se prema njima šalju po *Round-robin* principu. Servisi kojima su potrebni podaci o elementima mreže se obraćaju ovoj komponenti.

DDS - čuva se dinamika sistema. Prima listu identifikatora od NMS-a. Dobijeni podaci omogućavaju servisu kreiranje topologije mreže na nivou *korena* uz dobavljanje potrebnih informacija sa NMS-a. Simulira trenutne vrednosti potrošnje potrošača u mreži. Vrši proračun potrošnje mreže na nivou *korena* nakon promena vrednosti elemenata pristiglih sa SCADA sistema. Realizuje kružna isključenja, odnosno generiše i šalje komande SCADA sistemu. Može biti pokrenuto više instanci DDS servisa.

SCADA – komunicira sa simulatorom uz pomoć *Modbus* protokola. Prima listu identifikatora od NMS-a. Mapira digitalne i analogne ulaze/izlaze na simulator. Izvršava komande primljene od DDS komponente menjajući digitalne ili analogne vrednosti na simulatoru. Periodično proverava da li je došlo do promena vrednosti na simulatoru i promene šalje odgovarajućem DDS servisu.

PubSub – mehanizam koji služi za komunikaciju sa klijentom. Posredstvom ove komponente DDS šalje podatke klijentu o trenutnom stanju mreže.

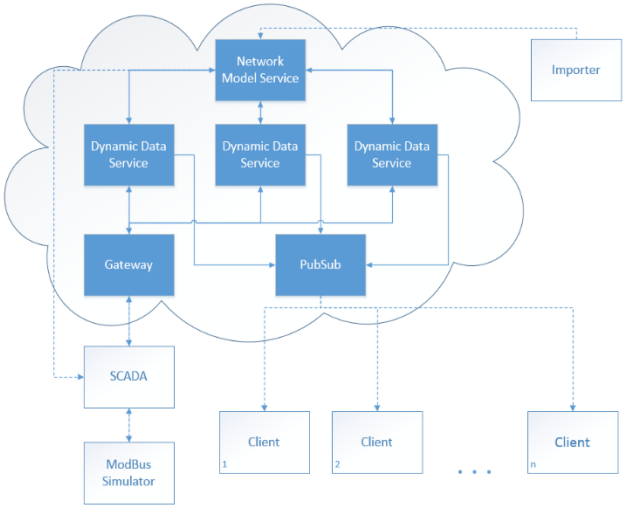
Modbus simulator – alat koji simulira stvarno stanje sistema. Sadrži analogne i digitalne ulaze/izlaze sa kojih se čita/upisuje stanje. Predstavlja stanje sistema, odnosno stanje prekidačke opreme.

Client – prati trenutno stanje mreže.

4.2.2. Arhitektura aplikacije u *Service Fabric* okruženju

Cloud arhitektura sistema je za nijansu drugačija u odnosu na lokalnu i prikazana je na slici 2. Pored komponenti koje su pomenute u prethodnom poglavlju, dodata je još komponenta:

- Gateway



Slika 2. *Arhitektura Service Fabric aplikacije*

Gateway – posrednik u komunikaciji između SCADA i DDS komponente. DDS je pokrenut na udaljenoj virtuelnoj mašini, dok je SCADA pokrenuta na lokalnom računaru. Vrši rutiranje dobijenih poruka od strane SCADA sistema ka odgovarajućoj instanci DDS sistema.

Servisi koji se pokreću u *Service Fabric* okruženju su: Gateway, NMS, DDS i PubSub. *Gateway* predstavlja *stateless* servis, dok su ostali servisi pokrenuti kao *stateful*. Ostale komponente sistema se pokreću na lokalnom računaru.

5. TESTIRANJE

Testiranje je sprovedeno u cilju poređenja performansi *Service Fabric* aplikacije sa performansama *on-premise*

aplikacije. Kao rezultat testiranja dobija se broj *korena* koje aplikacija uspeva uspešno da obradi u iterativnom periodu. Drugim rečima, rezultat testiranja je broj *korena* koje aplikacija može da obradi bez pojave strujnog preopterećenja. Promenljivi parametri prilikom testiranja su: broj DDS instanci koje su zadužene za obradu simulacije nad mrežom, trajanje pauze između dve iteracije simulacije (iterativni period) i vremenski interval na koji se vrši periodično prikupljanje podataka na SCADA komponenti.

Rezultati prikazani u tabelama 1 i 2 predstavljaju broj uspešno obrađenih *korena*. Sa *n* je označen broj pokrenutih DDS instanci, sa *sim* - trajanje pauze između dve iteracije simulacije, a sa *poll* - period za prikupljanje podataka na SCADA komponenti.

5.1. Testiranje *on-premise* aplikacije

Testiranje je izvršeno na računaru, odnosno virtuelnoj mašini, sa procesorom Intel Core i3-4130 CPU @ 3.4 GHz i memorijom od 32GB, od čega je 26GB dodeljeno virtuelnoj mašini. U tabeli 1 prikazani su dobijeni rezultati.

Tabela 1. *Rezultati testiranja on-premise aplikacije*

	Poll = 2s				Poll = 1.4s			
n \ sim	3s	5s	7s	10s	3s	5s	7s	10s
1	22	33	75	75	20	35	50	60
2	22	24	26	30	18	22	22	24
3	18	21	27	30	18	21	21	27
4	20	28	24	24	16	20	20	24
5	15	25	25	25	15	20	20	20

5.2. Testiranje *Service Fabric* aplikacije

Servisi pokrenuti u *Service Fabric* okruženju su podignuti na virtuelnim mašinama koje poseduju Intel Xeon CPU E5-2673 v3 @ 2.40 GHz, 1 Core procesor i RAM memoriju od 3.5 GB.

Ostale komponente su pokrenute na računaru sa memorijom od 6 GB i procesorom Intel Core i5-4200U CPU @ 1.60 GHz, turbo 2.30 GHz.

Rezultati dobijeni testiranjem su prikazani u tabeli 2.

Tabela 2. *Rezultati testiranja Service Fabric aplikacije*

	Poll = 2s				Poll = 1.4s			
n \ sim	3s	5s	7s	10s	3s	5s	7s	10s
1	24	25	38	40	20	28	30	30
2	20	30	36	36	16	26	28	30
3	18	21	30	36	18	27	27	30

Poređenjem rezultata iz tabela 1 i 2 se može primetiti da su rezultati u većini slučajeva približni. Uočava se da u slučaju pokrenute jedne DDS instance *on-premise* aplikacija postiže bolje rezultate, odnosno može uspešno da obrađuje više *korena*. To je naročito izraženo u slučajevima kada je trajanje pauze između dve iteracije simulacije (*sim*) 7s i 10s. Aplikacija pokrenuta u *Servic Fabric* okruženju daje za nijansu bolje rezultate kada je startovano više DDS instanci. Postoje slučajevi kada *on-premise* aplikacija postiže bolje rezultate, kada je pokrenuto više od jedne DDS instance (pokrenute dve DDS instance, pri pauzi između simulacija od 3 sekunde), ali u svim ostalim slučajevima sa više DDS instanci *Service Fabric* aplikacija je uspešnija.

6. ZAKLJUČAK

U ovom radu opisana je i testirana aplikacija koja simulira rad distributivnog dela elektroenergetskog sistema. Objašnjeno je na koji način radi i kojim se metodama služi u cilju zaštite mreže od strujnog preopterećenja. Ispitana je njena efikasnost varijacijom vrednosti parametara od značaja. Korišćenja su dva testna okruženja pri čemu su u većini slučajeva dobijeni slični rezultati.

On-premise aplikacija je bolje rešenje kada je pokrenuta jedna DDS instanca zato što bolje hardverske komponente lokalnog računara dolaze do izražaja u odnosu na iznajmljenu virtuelnu mašinu. Značajna prednost je i to što podaci cirkulišu brže u okviru lokalne mreže, na istom računaru. *Service Fabric* rešenje daje bolje rezultate sa više DDS komponenti, jer se svaka od njih pokreće na različitoj virtuelnoj mašini, za razliku od *on-premise* aplikacije gde se sve DDS instance pokreću na istom računaru. Resursi u vidu hardvera daju razliku u rezultatu. Ta razlika bi verovatno bila veća da je postojala mogućnost iznajmljivanja boljih računara, sa boljom hardverskom konfiguracijom. Probna verzija naloga na *Microsoft Azure* portalu ograničava korisnika da iznajmi virtuelne mašine sa novčanim ograničenjem (200$ u periodu testiranja), za period od mesec dana. Pritom, može se napraviti klaster sa najviše 3 čvora, s ograničenjem da skup iznajmljenih virtuelnih mašina ima na raspolaganju 4 jezgra. Drugim rečima, broj jezgara procesora iznajmljenih virtuelnih mašina u zbiru ne može biti veći od 4.

7. LITERATURA

[1] P. Vidović, "*Proračuni tokova snaga neuravnoteženih distributivnih mreža*", *doktorska disertacija*, Fakultet tehničkih nauka u Novom Sadu, Novi Sad, 2015.

[2] S. Borlase, *Smart Grids: Advanced Technologies and Solutions*, 2nd ed., CRC press, 2017.

[3] Azure Service Fabric Documentation, https://docs.microsoft.com/en-us/azure/service-fabric/ (pristupljeno u novembru 2019.)

Kratka biografija:

Dragan Stanković rođen je 03.06. 1995. godine u Somboru. Osnovnu školu je završio u Bačkom Brestovcu, a srednju ekonomsku školu u Odžacima. Fakultet Tehničkih Nauka, smer Elektroenergetski softverski inženjering, upisao je 2014. godine u Novom Sadu. Osnovne akademske studije završio je 2018. godine i upisao master akademske studije na smeru Primenjeno softversko inženjerstvo. Ispunio je sve obaveze i položio je sve ispite predviđene studijskim programom.