Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/12BE26Sekulic

AUTOMATSKO TESTIRANJE ADMS FUNKCIONALNOSTI ZA UČITAVANJE SNIMKA DINAMIČKIH PODATAKA

AUTOMATED TESTING OF ADMS LOAD SNAPSHOT FUNCTIONALITY

Dragiša Sekulić, Savo Đukić, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *U radu je izvršena analiza automatskog testiranja ADMS funkcionalnosti za učitavanje snimka dinamičkih podataka. Predstavljene su osnovne vrste testiranja softvera, kao i razlike između manuelnog i automatskog testiranja. Opisani su alati i okruženje korišćeni za pisanje automatskih testova. Na kraju rada dati su rezultati izvršavanja napisanih automatskih testova, kao i zaključci do kojih se došlo izvršenom analizom.*

Ključne riječi: *Automatski test, AMDS softver, funkcionalnost za učitavanje snimka dinamičkih podataka.*

Abstract – *In this paper the analysis of automated testing of ADMS Load Snapshot functionality is performed. The basic types of software testing, as well as the differences between manual and automated testing, are presented. Tools and environment used for writing automated tests are also described. At the end, results of automated tests are provided, as well as the conclusions derived based on performed analysis.*

Keywords: *Automated test, ADMS software, Load snapshot functionality.*

1. UVOD

Testiranje je veoma značajna aktivnost u procesu životnog ciklusa razvoja softvera. U izradi softvera, testiranje predstavlja pokušaj da se pronađu greške u softveru koji je napravljen. Greške u softveru mogu prouzrokovati ogromne novčane štete, tako da se moraju uočiti i otkloniti što je moguće ranije [1].
Kroz ovaj rad realizovano je automatsko testiranje funkcionalnosti za učitavanje snimka dinamičkih podataka (eng. Load Snapshot) koja je implementirana u naprednom softveru za menadžment distributivnih elektroenergetskih sistema (ADMS). Pošto se automatsko testiranje softvera sve više koristi radi uštede vremena, javila se ideja da se automatizuje testiranje ove funkcionalnosti.

2. TESTIRANJE SOFTVERA

Testiranje, u užem smislu, predstavlja provjeru da li određeni softver u potpunosti odgovara originalnim korisničkim zahtjevima, tj. da li je u potpunosti implementiran prema korisničkim zahtjevima. Testiranje, u širem smislu, predstavlja sistem kontrole kvaliteta, što znači da se osim provjere softvera, provjeravaju i sve njegove prateće komponente i karakteristike [1].
Testiranje softvera još možemo definisati kao proces koji se koristi da bi se utvrdila ispravnost, potpunost i kvalitet razvijenog softvera, a sastoji se od nekoliko aktivnosti: planiranje testiranja, dizajn testova, izvršavanja testova, evaluacija testova [1].

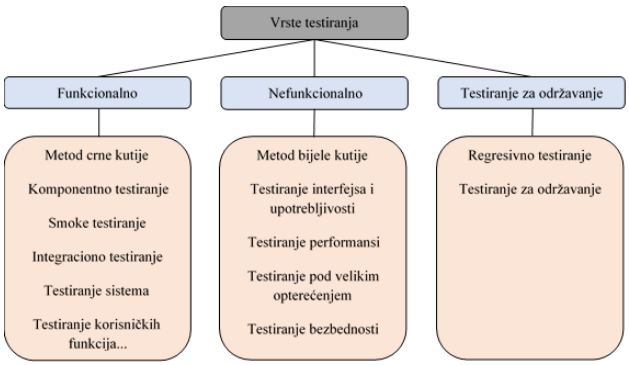
Testiranje se može vršiti manuelno i automatski. U tabeli 2.1 su date neke od razlika između manuelnog i automatskog testiranja [2].

Tabela 2.1. Razlike između manuelnog i automatskog testiranja

Parametar	Manuelno testiranje	Automatsko testiranje
Vrijeme obrade	Zahtjeva mnogo vremena i značajne ljudske resurse.	Znatno brže od manuelnog.
Pouzdanost	Manje pouzdano zbog mogućnosti ljudskih grešaka.	Pouzdanije je jer se izvodi pomoću alata i skripti.
Promjene korisničkog interfejsa	Ne utiču na manuelno testiranje.	Zahtevaju modifikacije automatskih testova.
Isplativost	Neisplativo za obimno regresivno testiranje.	Neisplativo za regresivno testiranje malog obima.
Programersko znanje	Nije potrebno.	Potrebno.
Idealan pristup	Kada je test slučaj potrebno izvršiti samo jednom.	Kada se često izvršavaju isti skupovi test slučajeva.

Testiranje softvera se najčešće dijeli na funkcionalno, nefunkcionalno i testiranje za održavanje, u zavisnosti od toga da li se kontrolišu samo krajnje funkcionalnosti ili i sam programski kod. Slika 2.1 prikazuje vrste testiranja [3].

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je dr Savo Đukić, docent.



Slika 2.1. Vrste testiranja

3. ALATI KORIŠĆENI ZA RAZVOJ AUTOMATSKIH TESTOVA

Za potrebe razvoja automatskih testova korišćeno je PyCharm okruženje, koje se koristi za programiranje u Python-u i koje rruža različite mogućnosti kao što su pomoć pri pisanju koda, grafički debager i integrisanu jedinicu tastera. PyCharm okruženje podržava rad na više različitih platformi (Windows, Linux), kao i rad sa JSON fajlovima, koji se koriste za čuvanje i razmjenu različitih tipova podataka [4].

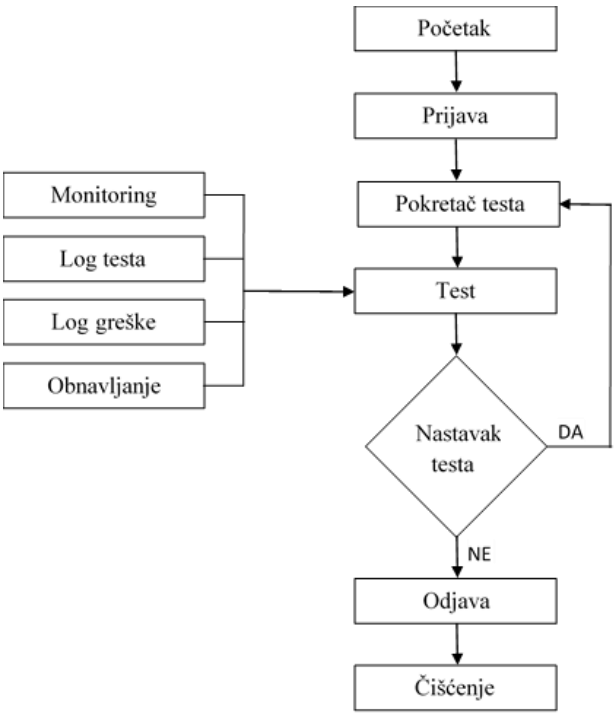
Kao osnova za razvoj automatskih testova korišćeni su Microsoft SQL Server i SQL Server Management Studio. SQL Server predstavlja integrisano riješenje za upravljanje bazama podataka. SQL Server Management Studio je alat koji pojednostavljuje upravljanje relacionom bazom podataka u sistemu SQL Server-a [5].

4. OKRUŽENJE AUTOMATSKOG TESTIRANJA

Uloga okruženja u kojem se razvijaju automatski testovi je ista kao arhitektura softvera u procesu njegovog razvoja. Okruženje automatskog testiranja definiše zajedničke funkcije, standardne testove, oslikava cjelokupnu strukturu u kojoj će se realizovati automatski testovi i definiše pravila prilikom imenovanja, dokumentovanja i upravljanja testovima. Dakle, okruženje daje mogućnost razvijanja održivih automatskih testova [6].

Zajedničke funkcije su funkcije koje svaki automatski test koristi prilikom svog razvoja i izvršavanja. Funkcije koje mogu biti sadržane u svim automatskim testovima su npr. ispisivanje loga testa, loga greške, rezultata izvršavanja, itd. Navedene funkcije se mogu pozivati iz bilo kog dijela automatskog testa, jer su realizovane kroz potprograme. Na slici 4.1 je prikazan algoritam zajedničkih funkcija [6].

Početak je funkcija koja priprema okruženje u kojem će se izvršavati automatski test. **Prijava** je funkcija koja pokreće okruženje ADMS softvera, provjerava da li je aplikacija dostupna za automatsko testiranje i provjerava validnost korisničkog imena i šifre za pokretanje aplikacije. **Pokretatač testa** je potprogram koji pokreće automatske testove. **Monitoring** je funkcija koja vrši nadgledanje izvršavanja svakog potprograma automatskog testa i provjerava stanje softvera nakon svakog izvršenog koraka. **Log testa** u posebnu datoteku upisuje rezultate izvršavanja svakog koraka i vrijeme izvršavanja. **Log greške** u posebnu datoteku upisuje rezultate nakon svakog neuspiješnog izvršavanja koraka automatskog testa. **Odjava** je obrnuta funkcija od učitavanja – vraća softver u stanje prije izvršavanja automatskog testa. **Čišćenje** je funkcija koja je slična funkciji početak, sa razlikom što ova funkcija uklanja posledice izvršavanja potprograma automatskog testa.



Slika 4.1. Algoritam zajedničkih funkcija

5. OPIS FUNKCIONALNOSTI ZA UČITAVANJE SNIMKA DINAMIČKIH PODATAKA

Funkcionalnost za učitavanje snimka dinamičkih podataka omogućava učitavanje snimljenih dinamičkih podataka (stanja prekidača, vrijednosti mjerenja, itd.) koju su upisani u bazu podataka. Ova funkcionalnost pruža inženjerima mogućnost oflajn analize radnih stanja mreže iz prošlosti, učitavanjem dinamičkih podataka i pokretanjem elektroenergetskih funkcija, u cilju pronalaženja uzroka problema koji su nastali u mreži (npr. kvar na dijelu mreže ili prekid napajanja potrošača).

Korisnici funkciju za učitavanje snimka dinamičkih podataka mogu da koriste kao alat za rekreiranje stvarnih stanja iz prošlosti i kao alat za podršku „šta ako“ analizama. Ova funkcionalnost korisniku omogućava da nakon učitavanja snimka dinamičkih podataka mijenja konfiguraciju mreže i vidi uticaj koji bi ta promijena imala u određenom trenutku.

Takođe, funkcionalnost pruža mogućnost provjere uklopnog stanja prekidača, kao i vrijednosti mjerenja, u željenom trenutku u prošlosti. Validnost učitanih dinamičkih podataka zavisi isključivo od toga da li su podaci upisani u bazu podataka; ukoliko podaci nisu upisani, korisnik neće imati validno stanje. Naravno, rad sa ovom funkcionalnošću je moguć samo u simulacionom modu ADMS-a (učitavanje nije moguće u realnom vremenu).

6. AUTOMATSKI TESTOVI ZA VERIFIKACIJU FUNKCIONALNOSTI UČITAVANJA SNIMKA DINAMIČKIH PODATAKA

U nastavku su dati primjeri realizovanih automatskih testova. Prva tri testa verifikuju uklopna stanja prekidača, a druga tri vrijednosti merenja. U nastavku je za svaki test data svrha, podaci od interesa i očekivani rezultat:

- Test 1 – Verifikacija uklopnog stanja prekidača na osnovu istorijskih podataka isčitanih iz baze podataka. Podaci od interesa su trenutak promjene stanja i vrijednost stanja prekidača. Očekivani rezultat testa je da se vrijednost stanja prekidača isčitana iz baze podataka poklapa sa vrijednošću iz kontrolnog prozora za učitani snimak iz prošlosti.
- Test 2 – Verifikacija uklopnog stanja prekidača na osnovu istorijskih podataka isčitanih iz baze podataka za željeni vremenski period. Podaci od interesa su trenutak promjene stanja i vrijednost stanja prekidača. Očekivani rezultat testa je da se vrijednost stanja prekidača isčitana iz baze podataka poklapa sa vrijednošću iz kontrolnog prozora za učitani snimak iz prošlosti.
- Test 3 – Verifikacija uklopnog stanja prekidača na osnovu istorijskih podataka isčitanih iz baze podataka. Podaci od interesa su trenutak promjene stanja i vrijednost stanja prekidača. Očekivani rezultat testa je da se vrijednost stanja prekidača isčitana iz baze podataka poklapa sa vrijednošću iz kontrolnog prozora za učitani snimak dinamičkih podataka iz prošlosti.
- Test 4 – Verifikacija vrijednosti mjerenja na osnovu istorijskih podataka isčitanih iz baze podataka. Podaci od interesa su trenutak promjene vrijednosti i vrijednost mjerenja. Očekivani rezultat testa je da se vrijednost mjerenja isčitana iz baze podataka poklapa sa vrijednošću iz kontrolnog prozora za učitani snimak dinamičkih podataka iz prošlosti.
- Test 5 – Verifikacija vrijednosti mjerenja na osnovu istorijskih podataka isčitanih iz baze podataka za željeni vremenski period. Podaci od interesa su trenutak promjene vrijednosti i vrijednost mjerenja. Očekivani rezultat testa je da se vrijednost mjerenja iščitana iz baze podataka poklapa sa vrijednošću iz kontrolnog prozora za učitani snimak dinamičkih podataka iz prošlosti.
- Test 6 – Verifikacija vrijednosti mjerenja na osnovu istorijskih podataka isčitanih iz baze podataka. Podaci od interesa su trenutak promjene vrijednosti i vrijednost mjerenja. Očekivani rezultat testa je da se vrijednost mjerenja isčitana iz baze podataka poklapa sa vrijednošću iz kontrolnog prozora za učitani snimak dinamičkih podataka iz prošlosti.

6.1. Rezultati izvršavanja automatskih testova

Testiranje predmetne funkcionalnosti je izvršeno na testnom sistemu ADMS softvera u četiri ciklusa automatskog testiranja. U tabeli 6.1.1 i 6.1.2 su prikazani rezultati izvršavanja ciklusa.

Iz tabele 6.1.1 se vidi da je Test 3 u drugom ciklusu neuspiješno izvršen. Iz logova je zaključeno da je došlo do usporenja sistema, koje je prouzrokovalo da prozor sa signalima ne bude otvoren u predviđenom vremenskom periodu (greška 031). Ovaj ciklus automatskog testiranja je pokazao da postoji problem u performansama sistema.

Kao što se vidi u tabel1 6.1.2, i u trećem ciklusu je pao jedan test, dok je jedan test preskočen. Analizom logova Testa 4 je uočeno da se vrijednost mjerenja iščitana iz baze podataka ne poklapa sa vrijednošću u kontrolnom prozoru (greška 041). Što se tiče Testa 2, u zadatom vremenskom periodu nije bilo promijena uklopnog stanja prekidača, tako da je test preskočen.

Iz tabele 6.1.2 se vidi da su dva testa neuspiješno izvršena, Test 1 i Test 6. Analizom logova je utvrđeno zašto je došlo do pada navedenih testova. Problem koji je doveo do pada Testa 1 je to što se uklopno stanje prekidača iščitano iz baze podataka nije poklopilo sa uklopnim stanjem u kontrolnom prozoru prekidača (greška 011). Analizom logova Testa 6 je zaključeno da se istorijski podaci iz baze podataka nisu mogli iščitati za odabrano mjerenje, tj. pozvana metoda je vratila praznu listu, kao da nema upisanih podataka (greška 061).

Tabela 6.1.1. Rezultati izvršavanja prvog i drugog ciklusa

Broj testa	Prvi ciklus		Drugi ciklus	
	Status izvršavanja	Broj greške	Status izvršavanja	Broj greške
Test 1	Prošao	/	Prošao	/
Test 2	Prošao	/	Prošao	/
Test 3	Prošao	/	Pao	031
Test 4	Prošao	/	Prošao	/
Test 5	Prošao	/	Prošao	/
Test 6	Prošao	/	Prošao	/

Tabela 6.1.2. Rezultati izvršavanja trećeg i četvrtog ciklusa

Broj testa	Treći ciklus		Četvrti ciklus	
	Status izvršavanja	Broj greške	Status izvršavanja	Broj greške
Test 1	Prošao	/	Pao	011
Test 2	Preskočen	/	Prošao	/
Test 3	Prošao	/	Prošao	/
Test 4	Pao	041	Prošao	/
Test 5	Prošao	/	Prošao	/
Test 6	Prošao	/	Pao	061

U tabeli 6.1.3 su prikazana vremena trajanja izvršavanja testova, manuelno i automatski. Iz tabele se vidi da su vremena izvršavanja automatskih testova mnogo manja od vremena manuelnog izvršavanja istih. Kao što se iz ove tabele vidi, ukupno vrijeme trajanja manuelnog izvršavanja je 1489 sekundi, odnosno 24 minuta i 49 sekundi, dok je vrijeme trajanja automatskog izvršavanja 484 sekunde, odnosno 8 minuta i 4 sekunde. Dakle, vrijeme trajanja automatskog izvršavanja testova je oko 3 puta manje od manuelnog. Prikazana tabela pokazuje jednu od prednosti automatskog testiranja u odnosu na manuelno, a to je ušteda vremena potrebnog za izvršavanje testova.

Tabela 6.1.3. Poređenje vremena trajanja izvršavanja testova manuelno i automatski

Broj testa	Vrijeme trajanja manuelnog izvršavanja testa [s]	Vrijeme trajanja automatskog izvršavanja testa [s]
Test 1	277	107
Test 2	235	35
Test 3	261	136
Test 4	250	33
Test 5	206	36
Test 6	260	137
Ukupno vrijeme trajanja izvršavanja testova	1489	484

6.2. Detektovane prednosti i mane automatskog testiranja

Prednosti automatskog testiranja, detektovane kroz rad, su:

- Izvršavanje automatskih testova na novijim verzijama softvera; potrebne su minimalne modifikacije samo u slučaju da je došlo do unapređenja testirane funkcionalnosti.
- Potreban je manji broj test inženjera i manje vrijemena za testiranje, čime je postignuta mnogo bolja iskorišćenost resursa.
- Izvršavanje automatskih testova je moguće u bilo kom vremenskom periodu (24/7).
- Automatskim testiranjem se postiže smanjenje vremena potrebnog za isporuku softvera.

Mane automatskog testiranja, detektovane kroz rad, su:

- Test inženjer mora proći obuku za pisanje automatskih testova, odnosno mora da poznaje programski jezik u kom se realizuju automatski testovi. Na to se utroši određeno vrijeme, koje se kasnije isplati zbog uštede na vremenu izvršavanja testova.
- Modifikacije automatskog testa ili loša implementacija mogu dovesti do neuspiješnog izvršavanja.
- Ista greška u softveru može biti registrovana kroz više automatskih testova (ista greška može dovesti do pada niza automatskih testova).
- Automatski test može biti lažno neuspiješno izvršen. Nepravilna konfiguracija ulaznih parametara i nevalidno testno okruženje mogu biti razlozi lažnog neuspiješnog izvršavanja testa, koje nije posledica greške u softveru, niti loše napisanog automatskog testa.
- Automatski test može biti lažno uspiješno izvršen. Loš dizajn automatskog testa, nepravilno provjeravanje moguće greške u softveru i nepravilno navedeni očekivani rezultati mogu biti razlozi lažnog uspiješnog izvršavanja testa.

7. ZAKLJUČAK

Tema ovog rada je automatsko testiranje funkcionalnosti za učitavanje snimka dinamičkih podataka implementirane u ADMS softver. U prvom poglavlju rada je definisan proces, vrste i metoda testiranja. Nakon toga su opisani korišćeni alati i okruženje za automatsko testiranje, kao i sama funkcionalnost koja je predmet testiranja. Na kraju rada je dat opis napisanih automatskih testova i rezultati njihovog izvršavanja.

Automatizacija testova je realizovana upotrebom programskog jezika Python u razvojnom okruženju PyCharm. Realizovani automatski testovi verifikuju uklopna stanja prekidača i vrijednosti mjerenja. Istorijski podaci iščitani iz baze podataka su poređeni sa podacima prikazanim u ADMS klijentskim aplikacijama.

Automatskim testiranjem realizovanim kroz ovaj rad je postignuta ušteda vremena i potrebnih ljudskih resursa. Takođe je postignuta mogućnost testiranja 24/7, jer automatski testovi mogu da se izvršavaju u bilo koje doba dana i noći. Zaključak je da je automatsko testiranje pouzdanije, efikasnije i ekonomičnije u odnosu na manuelno, iako obe metode testiranja imaju svoje prednosti i mane.

8. LITERATURA

[1] Popović Jovan, *Testiranje softvera u praksi*, Računarski fakultet, Beograd, 2012.

[2] https://www.guru99.com/difference-automated-vs-manual-testing.html

[3] Glenford J. Myers, *The art of software testing*, John Wiley and Sons, New Jersey, 2004.

[4] Pedro Kroger, *Modern Python Development With PyCharm*, 2015.

[5] Paul Atkinson, Robert Vieira, *Microsoft SQL Server 2012 programiranje: Od početka*, CET, Beograd, 2013.

[6] Linda G. Hayes, *Automated Testing Handbook*, Software Testing Institute, Richardson, 2004.

Kratka biografija:

Dragiša Sekulić rođen je u Bijeljini 1990. godine. Diplomski rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Energetska elektronika i električne mašine odbranio je 2016. godine.

Savo Đukić rođen je u Novom Sadu 1983. godine. Doktorsku disertaciju odbranio je 2014. godine na Fakultetu tehničkih nauka iz oblasti Elektroenergetski sistemi.