Dijagrami stanja i tabele stanja kroz primere

Seminarski rad u okviru kursa
Verifikacija softvera
Matematički fakultet

Pavle Savić, 1075/2022
pavlesavic1389@gmail.com

11. januar 2024.

Sažetak

Cena neispravnog softvera često je materijalno ogromna, a nekada može imati i fatalne posledice. Briga o kvalitetu softvera (eng. *Quality Assurance*) i testiranje kao njen značajan deo stoga zauzimaju važno mesto u pogledu vremena, novca, napora i ljudskih resursa u procesu razvoja industrijskog softvera. Ove aktivnosti u današnje vreme, sa povećanjem konkurencije na tržištu i smanjene tolerancije korisnika na greške, dodatno dobijaju na važnosti. U testiranju softvera prisutne su različite vrste, načini i tehnike kako bi se povećala sigurnost u ispravnost i kvalitet aplikacija. U ovom radu biće ukratko predstavljeni dijagrami stanja (eng. *State-Transition Diagram*) i tabele stanja (eng. *State-Transition Table*) i njihova primena u testiranju softvera, kao i modelovanje nekoliko primera sistema ovim tehnikama.

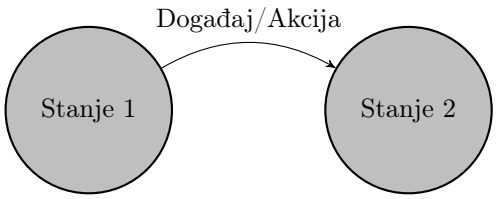
Sadržaj

1 Uvod

Prikazivanje različitih stanja u kojima sistem funkcioniše i događaja koji utiču na promenu tog stanja jeste moćna tehnika za analizu, dizajn i testiranje softverskih sistema u kontekstu razvoja softvera.

Dijagram stanja (eng. *State-Transition Diagram*) je dinamički dijagram koji vizuelno predstavlja stanja sistema i način njegove interakcije sa spoljašnjim svetom kroz događaje na koje sistem reaguje promenom stanja [2]. Dijagram stanja je slična tehnika UML dijagramu stanja (eng. *UML State Machine Diagram*) ali sa različitom notacijom. Osnovnu strukturu čine 4 tipa elemenata (slika 1):

- **Stanje** - vidljiv način ponašanja sistema, u svakom momentu sistem je samo u jednom od mogućih stanja
- **Prelaz** - dozvoljena promena iz jednog stanja sistema u drugo
- **Događaj** - interni (npr. istek tajmera) ili eksterni, prouzrokuje prelaz
- **Akcija** - operacija sistema izazvana prelazom



Slika 1: Osnovni elementi dijagrama stanja

Kod složenijih sistema poželjno je da se nacrta više dijagrama stanja za različite objekte sistema. Dijagrami stanja su veoma korisni za opisivanje ponašanja pojedinačnih objekata u celom skupu slučajeva upotrebe koji utiču na te objekte a ne treba ih koristiti za opisivanje saradnje između objekata koja uzrokuje prelaze [3]. Dijagram stanja koji prolazi kroz definisani životni ciklus objekta može da ima jedno ili više konačnih stanja. Ova stanja na dijagramu nemaju izlazne, već samo ulazne strelice. Početno stanje objekta se često posebno označava na dijagramu.

Tabela stanja (eng. *State-Transition Table*) sistematično prikazuje sve moguće prelaze između stanja u obliku matrice. Poslovni analitičar može da koristi tabele stanja da osigura da su svi prelazi identifikovani analizom svake ćelije u matrici. Sva moguća stanja se pojavljuju u prvoj koloni sleva i u prvoj vrsti odozgo. Ćelije pokazuju da li je prelaz iz kolone u vrstu validan i ako jeste prikazuje događaj koji izaziva taj prelaz i eventualnu akciju sistema. Format dijagrama pomaže zainteresovanim stranama da vizualizuju moguće sekvence prelaza, a format tabele pomaže da se osigura da sistem u svim situacijama ima predefinisano ponašanje. [5]

1.1 Testiranje zasnovano na dijagramu i tabeli stanja

Testiranje promene stanja (eng. *State Transition Testing*) je tehnika testiranja crne kutije (eng. *Black Box Testing*) koja pomaže da se utvrdi kako promene ulaznih parametara izazivaju promene stanja ili promene

izlaza u aplikaciji koja se testira. Ova tehnika testiranja pomaže da se analizira ponašanje aplikacije za različite ulazne parametre. Testeri mogu da obezbede pozitivne i negativne ulazne vrednosti testa i zabeleže ponašanje sistema. Svaki sistem koji reaguje na različit način za isti ulaz, u zavisnosti od toga šta se ranije dešava (u kom se stanju trenutno nalazi) može se modelovati konačnim automatom (eng. *Finite-state machine*). Osnovno pravilo konzistentnosti je da se sistem u proizvoljnom stanju ponaša isto bez obzira kojim putem se do tog stanja došlo.

Ova tehnika testiranja je korisna kada je potrebno da se testira sistem u odnosu na konačni skup ulaznih vrednosti. Mogu se testirati različiti sistemski prelazi za sekvencu događaja, npr. unosi ulaznih parametara preko interfejsa koji čine celovitu proceduru. Često se koristi za testiranje aplikacija u realnom vremenu. Sa druge strane nije pogodna kada je potrebno testirati različite funkcionalnosti aplikacije u kontekstu istraživačkog testiranja (eng. *Exploratory Testing*), ako sistem nema konačan broj ulaza i izlaza, kao ni za sisteme koji imaju eksponencijalni rast ulaznih parametara. Takođe ova tehnika se ne koristi kada nije potrebno ispitati ponašanje sistema za sekvencu ulaza. Tester uvek treba da ima jasnu viziju o ulazima i očekivanim izlazima iz sistema. [2] [1]

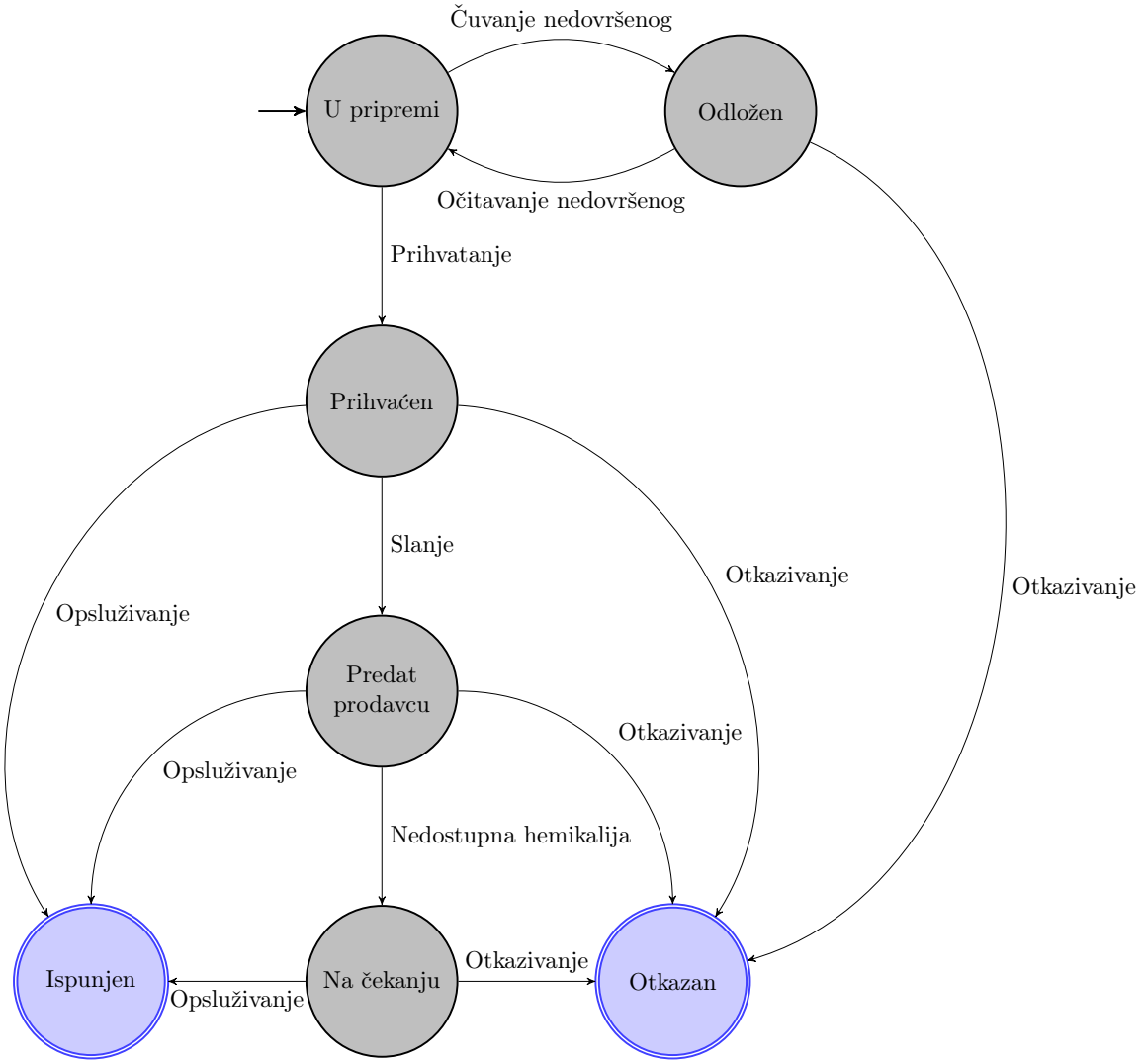
Dijagrami stanja olakšavaju rano testiranje jer testeri mogu izvesti test slučajeve njegovim obilaskom i pokriti sve dozvoljene prelazne putanje i verifikovati da se nedozvoljene putanje ne mogu desiti. Temeljnost našeg testiranja oslanja se na kompletan model. Stoga, propuštena stanja ili prelazi mogu dovesti do toga da deo sistema ostane nepokriven testovima, ostavljajući potencijalne nedostatke neotkrivenim. Pri pravljenju skupa test slučajeva mogu se zahtevati različiti nivoi pokrivenosti. Za kompleksne sisteme lako dolazi do eksplozije broja stanja pa je važno napraviti kompromis između pokrivenosti i količine testova [4]. Primer dobrog kompomisa je skup testova koji omogućava da se svaki prelaz ispita bar jednom. Takođe, može se zahtevati da se svako stanje ili svaka putanja kroz dijagram obiđu bar jednom. Preglednost tabela stanja može nam pomoći da osiguramo da u skupu test slučajeva ispitujeme sve kombinacije stanja i događaja koji iz tog stanja izazivaju validan prelaz [5].

2 Primeri

U nastavku rada biće predstavljeno nekoliko primera modelovanja jednostavnih sistema dijagramima i tabelama stanja.

2.1 Sistem nabavke [5]

Kao ilustracija, u ovom primeru biće razmotren sistem koji upravlja zalihama hemikalija u istraživačkim laboratorijama velike kompanije. Ovaj sistem omogućava naučnicima da traže nove hemikalije, prate lokaciju i status pojedinačnih hemijskih kontejnera, pruža zdravstvene i bezbednosne informacije i tako dalje. Kada korisnik pošalje zahtev za novu hemikaliju, sistem može ispuniti zahtev ili iz inventara skladišta hemikalija ili slanjem porudžbine komercijalnom prodavcu hemikalija. Prodavac takođe može staviti porudžbinu na čekanje u slučaju da trenutno nema hemikaliju na raspolaganju i o tome obavestiti kupca. Svaki zahtev će proći kroz niz stanja između vremena kada je kreiran i trenutka kada je ispunjen ili otkazan – dva završna stanja. Dijagram i tabela stanja zahteva prikazani su na slici 2 i tabeli 1.



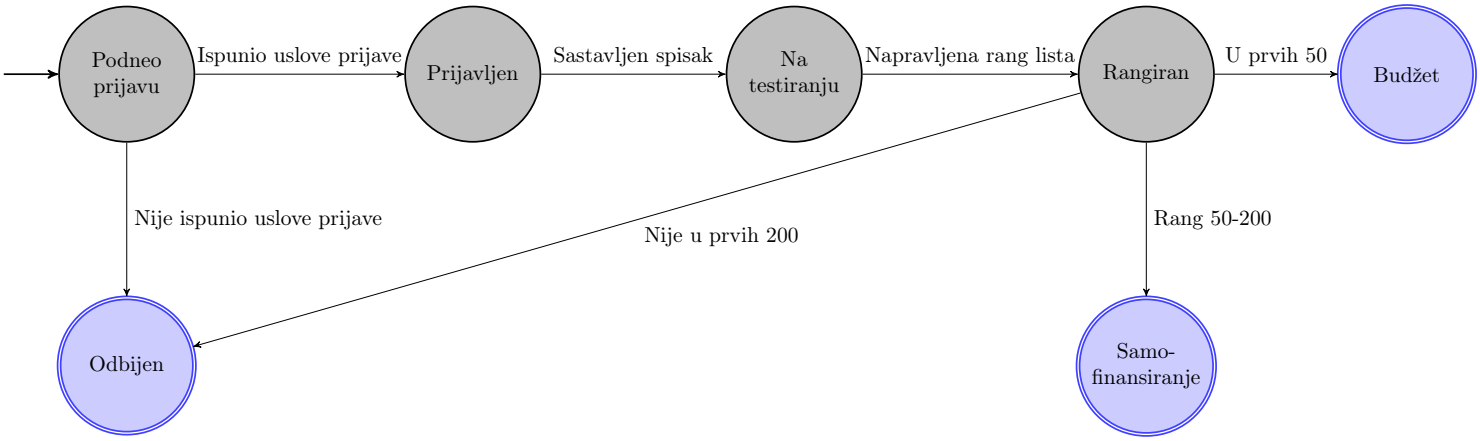
Slika 2: Dijagram stanja zahteva za 2.1

Iz stanja \ U stanje	**U pripremi**	**Odložen**	**Prihvaćen**	**Predat**	**Na čekanju**	**Ispunjen**	**Otkazan**
U pripremi	/	Čuvanje nedovršenog	Prihvatanje	X	X	X	X
Odložen	Očitavanje nedovršenog	/	X	X	X	X	Otkazivanje
Prihvaćen	X	X	/	Slanje	X	Opsluživanje	Otkazivanje
Predat	X	X	X	/	Nedostupna hemikalija	Opsluživanje	Otkazivanje
Na čekanju	X	X	X	X	/	Opsluživanje	Otkazivanje
Ispunjen	X	X	X	X	X	/	X
Otkazan	X	X	X	X	X	X	/

Tabela 1: Tabela stanja zahteva za 2.1

2.2 Upis kandidata na fakultet

U ovom primeru prikazana su stanja kandidata u sistemu kroz proceduru upisa na fakultet. Da bi kandidat bio prijavljen mora ispuniti određene uslove vezane za prethodno školovanje i uz prijavu priložiti traženu dokumentaciju. Nakon toga u zavisnosti od rezultata testiranja može završiti u jednom od 3 završna stanja - odbijen, upisan kao budžetski, upisan kao samofinansirajući student. Dijagram i tabela stanja kandidata prikazani su na slici 3 i tabeli 2.



Slika 3: Dijagram stanja kandidata za 2.2

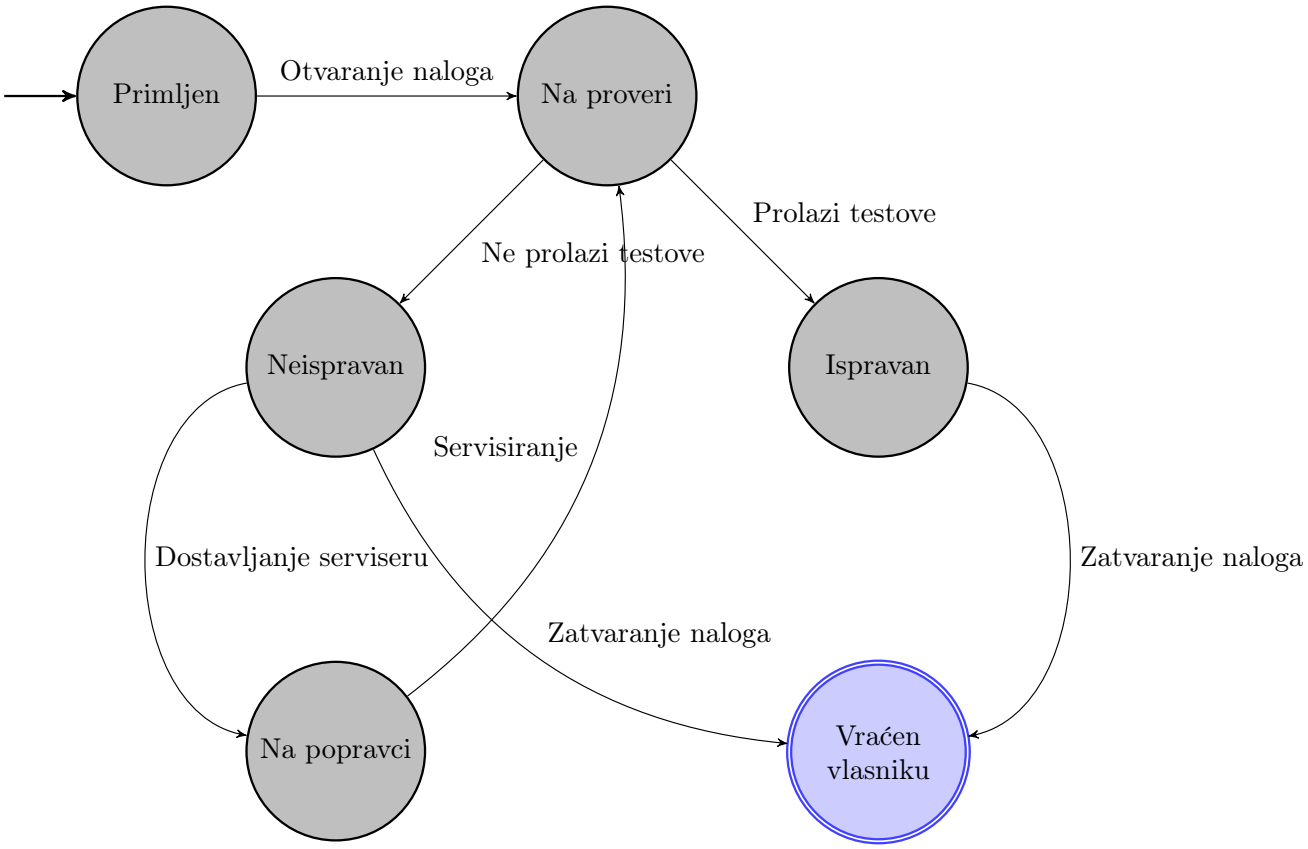
Iz stanja \ U stanje	**Podneo prijavu**	**Prijavljen**	**Na testiranju**	**Rangiran**	**Budžet**	**Samofinansiranje**	**Odbijen**
Podneo prijavu	/	Ispunjeni uslovi	X	X	X	X	Neispunjeni uslovi
Prijavljen	X	/	Sastavljen spisak	X	X	X	X
Na testiranju	X	X	/	Napravljena rang lista	X	X	X
Rangiran	X	X	X	/	U prvih 50	Rang 50-200	Nije u prvih 200
Budžet	X	X	X	X	/	X	X
Samofinansiranje	X	X	X	X	X	/	X
Odbijen	X	X	X	X	X	X	/

Tabela 2: Tabela stanja kandidata za 2.2

2.3 Servis računara

U ovom primeru prikazana su stanja računara u sistemu servisa računara. Nakon što se otvori radni nalog dijagnostifikuju se potencijalne neispravnosti računara. Potom se defekti uklanjaju sve dok se ne utvrdi da računar ispravno radi ili da servis ne može otkloniti problem. U oba slučaja radni nalog se zatvara i računar je u završnom stanju - vraćen vlasniku. Dijagram i tabela stanja računara prikazani su na slici 4 i tabeli 3 [1].

[1] X oznaka u ćeliji tabela stanja predstavlja nevalidan prelaz



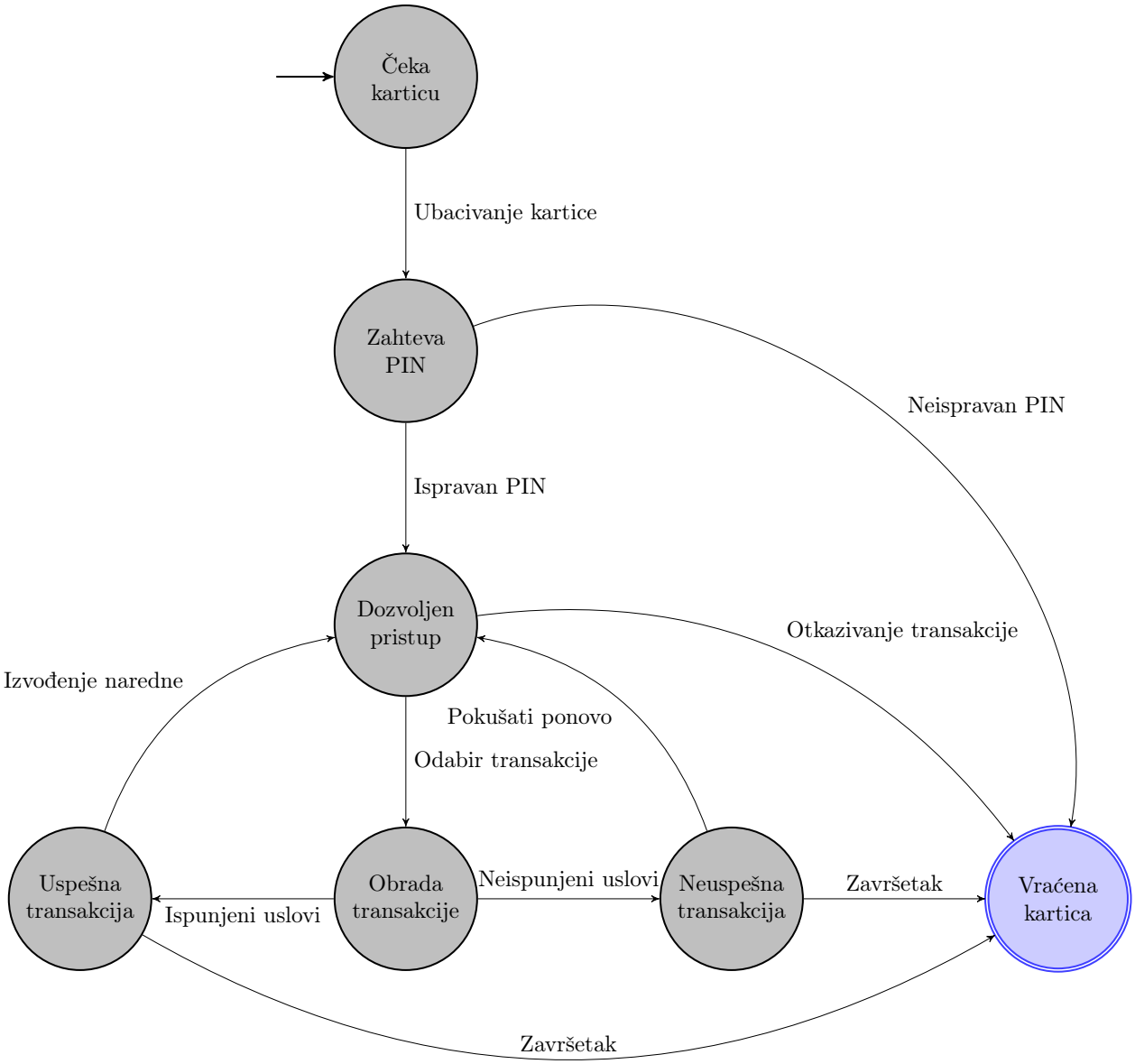
Slika 4: Dijagram stanja računara za 2.3

Iz stanja \ U stanje	**Primljen**	**Na proveri**	**Neispravan**	**Ispravan**	**Na popravci**	**Vraćen vlasniku**
Primljen	/	Otvaranje naloga	X	X	X	X
Na proveri	X	/	Ne prolazi testove	Prolazi testove	X	X
Neispravan	X	X	/	X	Dostavljanje serviseru	Zatvaranje naloga
Ispravan	X	X	X	/	X	Zatvaranje naloga
Na popravci	X	Servisiranje	X	X	/	X
Vraćen vlasniku	X	X	X	X	X	/

Tabela 3: Tabela stanja računara za 2.3

2.4 Prikaz rada bankomata [4]

U ovom primeru predstavljen je princip rada i stanja kroz koja bankomat prolazi kada klijent obavlja transakciju. Nakon što klijent ubaci debitnu ili kreditnu karticu u čitač kartica bankomata i kartica se uspešno pročita, bankomat traži unos PIN-a. U slučaju unosa ispravnog PIN-a klijentu se omogućava da izvršava željene transakcije. Klijent ne može da izvrši transakciju za koju ne ispunjava uslove (npr. podizanje gotovine u iznosu većem od tekućeg stanja na računu). Sistem ima jedno završno stanje - kartica se vraća klijentu. Dijagram i tabela stanja bankomata prikazani su na slici 5 i tabeli 4.



Slika 5: Dijagram stanja bankomata za 2.4

Iz stanja \ U stanje	**Čeka karticu**	**Zahteva PIN**	**Dozvoljen pristup**	**Obrada**	**Uspešna**	**Neuspešna**	**Vraćena kartica**
Čeka karticu	/	Ubačena kartica	X	X	X	X	X
Zahteva PIN	X	/	Ispravan PIN	X	X	X	Neispravan PIN
Dozvoljen pristup	X	X	/	Odabir transakcije	X	X	Otkazivanje transakcije
Obrada	X	X	X	/	Ispunjeni uslovi	Neispunjeni uslovi	X
Uspešna	X	X	Izvođenje naredne	X	/	X	Završetak
Neuspešna	X	X	Pokušati ponovo	X	X	/	Završetak
Vraćena kartica	X	X	X	X	X	X	/

Tabela 4: Tabela stanja bankomata za 2.4

3 Zaključak

Testiranje zasnovano na dijagramu stanja je značajna tehnika testiranja crne kutije (eng. *black box testing*), posebno za sisteme gde redosled događaja ili unosa ima veliki uticaj na to kako se program ponaša. Pomaže nam da razumevanjem stanja, događaja i akcija sistema izvedemo smislene test slučajeve. Pažljiva integracija ove tehnike u proces testiranja može značajno doprineti u proizvodnji visokokvalitetnih softverskih sistema koji su funkcionalni, pouzdani i stabilni.

Ipak, treba da budemo svesni njegovih ograničenja i kako bismo obezbedili da naša strategija testiranja bude sveobuhvatna i efikasna često je poželjno kombinovati ovu tehniku sa drugim tehnikama testiranja crne kutije (eng. *Black Box Testing*), kao što su testiranje pomoću klasa ekvivalencije (eng. *Equivalence Class Testing*), njemu srodno testiranje graničnih vrednosti (eng. *Boundary Value Testing*) ili testiranje pomoću tabele odlučivanja (eng. *Decision Table Testing*). Ovo ne umanjuje značaj dijagrama stanja u procesu verifikacije softvera i poznavanje ove tehnike važna je veština kojom bi jedan tester trebalo da ovlada.

Literatura

[1] State transition testing technique and state transition diagram with examples, 2023.

[2] Thomas Hamilton. State transition testing – diagram technique (example), 2023.

[3] Abdul Salam Kalaji, Robert Mark Hierons, and Stephen Swift. Generating feasible transition paths for testing from an extended finite state machine (efsm). In *2009 International Conference on Software Testing Verification and Validation*, pages 230–239, 2009.

[4] Rahmat Ullah Orakzai. Software testing: State transition, 2023.

[5] Karl Wiegers. Modeling system states: State-transition diagrams and state tables, 2023.