Specifikacija softverskih sistema

Predavanje br. 8 – Dijagram prelaza stanja (konačnih automata)

Gordana Milosavljević

Katedra za informatiku, FTN, Novi Sad

2022.

Dijagram prelaza stanja (konačnih automata)

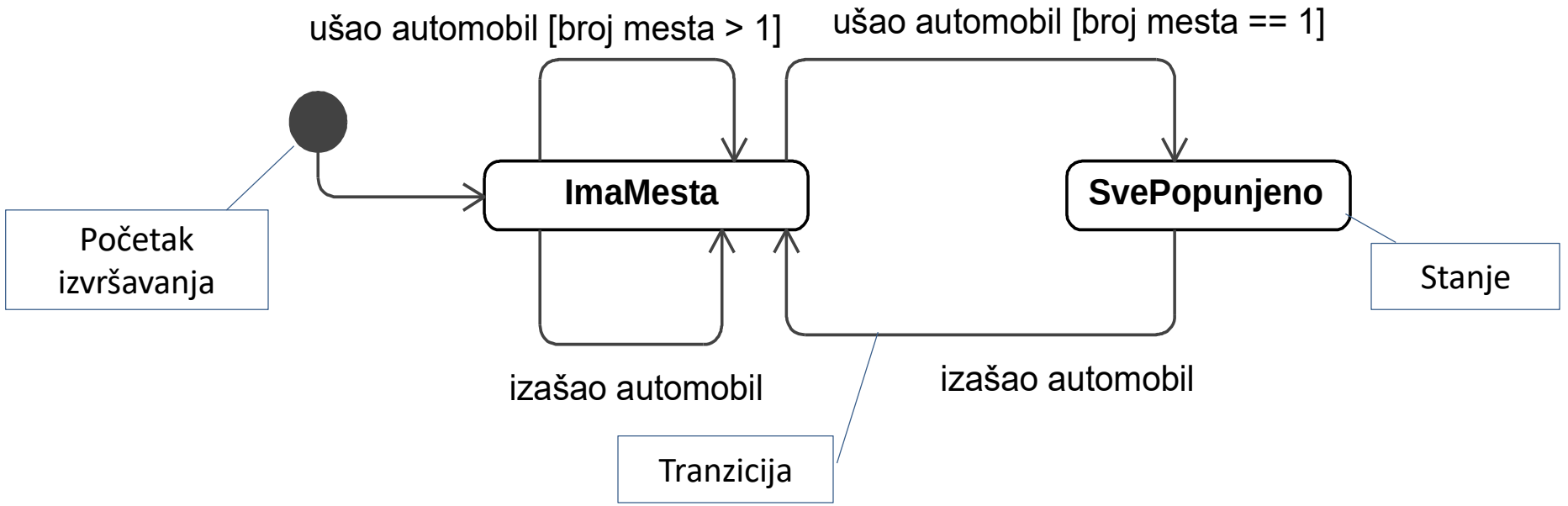
- koristi se za projektovanje softverskih ili hardverskih sistema za čije ponašanje je karakteristično da se mogu nalaziti u konačnom skupu stanja i da je prelazak iz jednog stanja u drugo uzrokovan događajima.

Primer 1- kontroler za upravljanje parkingom

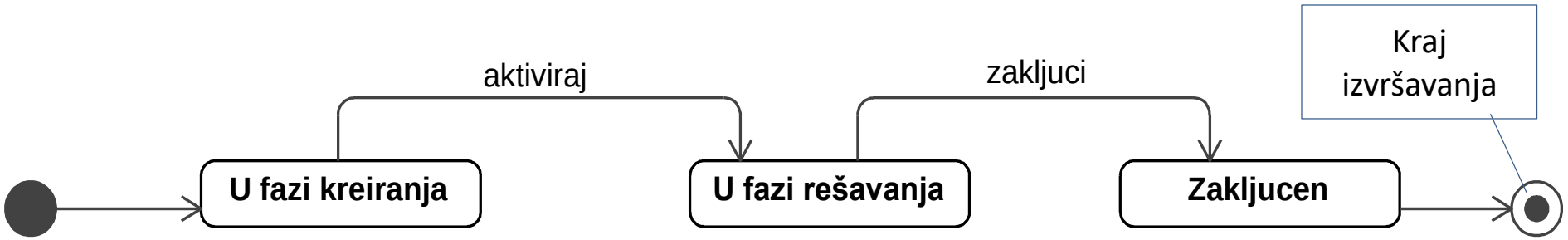
Zadatak kontrolera je:

- da prima signale za ulazak i izlazak automobila i ažurira broj slobodnih mesta koji prikazuje na ekranu postavljenim kod ulaza na parking,
- da na semaforu koji se nalazi ispred ulaza na parking drži upaljeno zeleno svetlo ako ima slobodnih mesta, odnosno da uključi crveno svetlo kada se zauzme poslednje slobodno mesto.

Primer 1 – početni dijagram prelaza stanja za kontroler parkinga



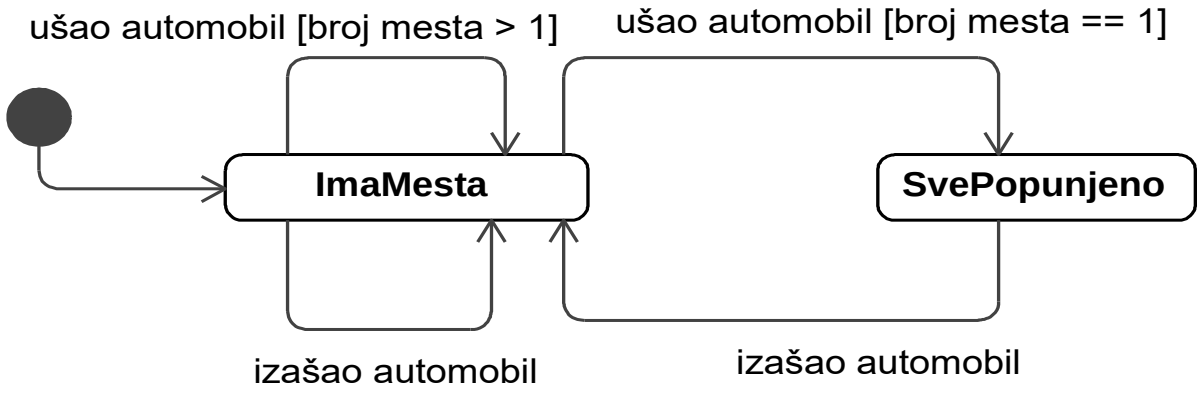
Primer 2 – Dijagram prelaza stanja za test iz sistema elektronskog ocenjivanja

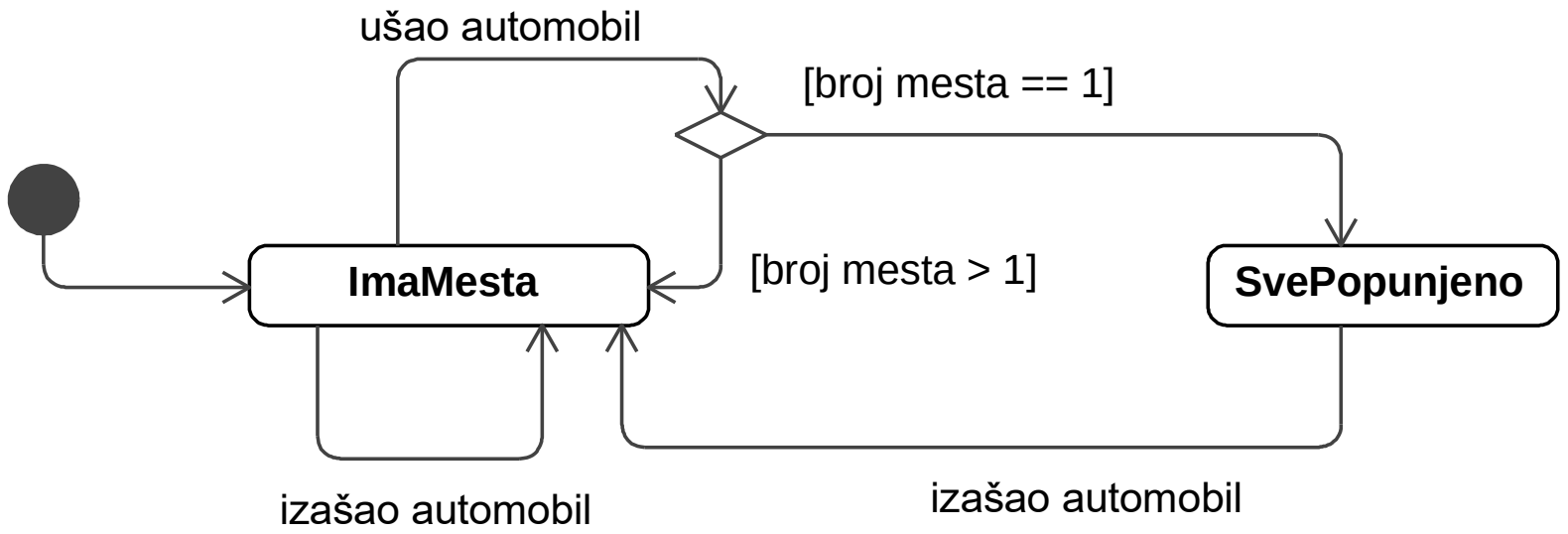


Pseudo-stanja

- U dijagramima prelaza stanja se mogu koristiti simboli za početak, kraj, uslovno izvršavanje, razdelnik i spoj iz dijagrama aktivnosti.
- Oni se tretiraju kao pseudo-stanja – nisu stanja, ali se mogu povezivati tranzicijama sa drugim stanjima.

Primer 3 – početni dijagram prelaza stanja za kontroler parkinga preglednije nacrtan



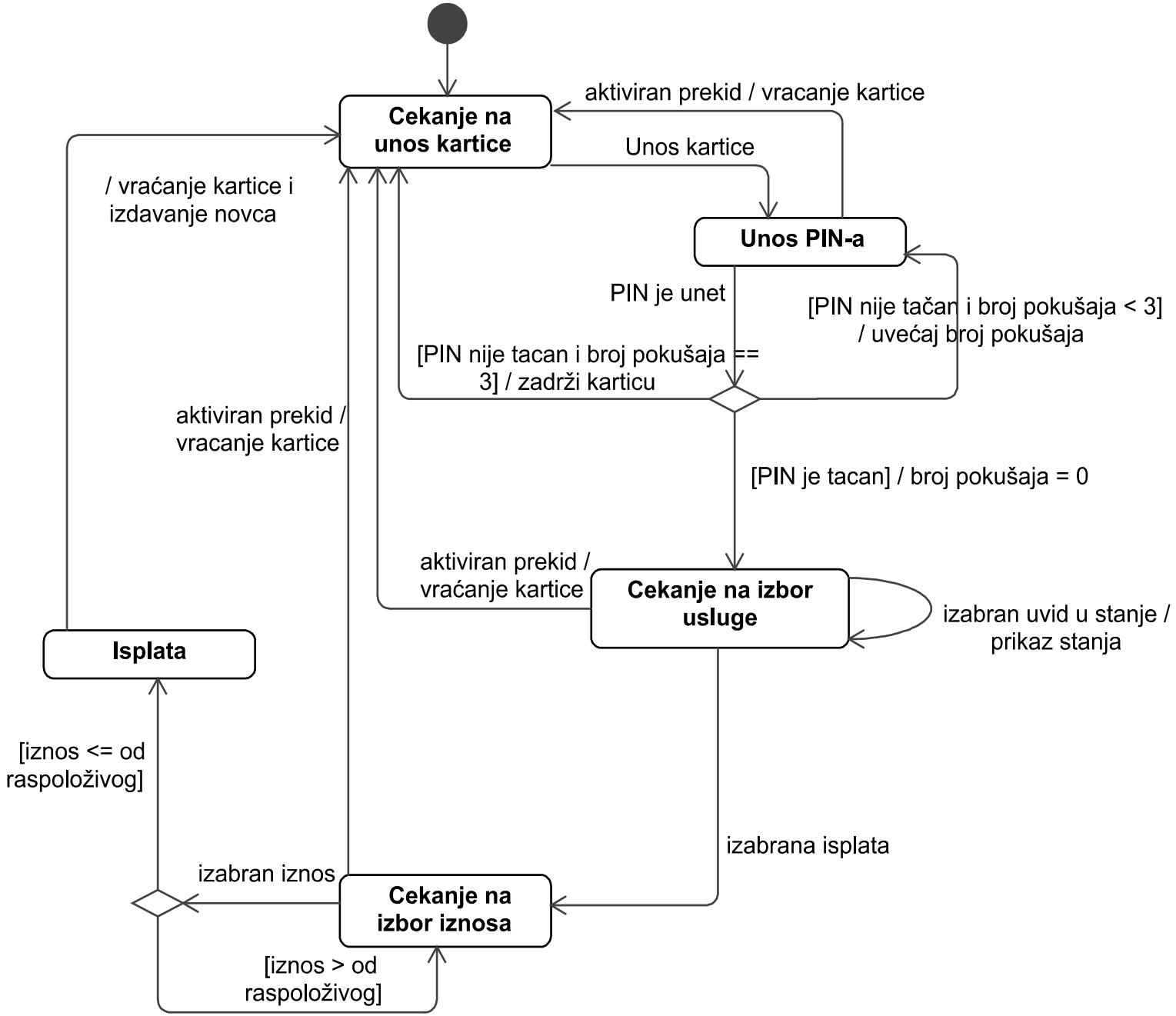


Tranzicija

- Modeluje reakciju na događaj, koja može izazvati prelazak iz jednog stanja u drugo ili povratak u isto stanje, uz izvršenje pridruženih akcija.
- Format:

  događaj [uslov] / akcija
- Uslovom se specificira da do tranzicije ne dolazi uvek, već samo ako je dati uslov zadovoljen.

Primer 4 – početni dijagram koji opisuje ponašanje bankomata

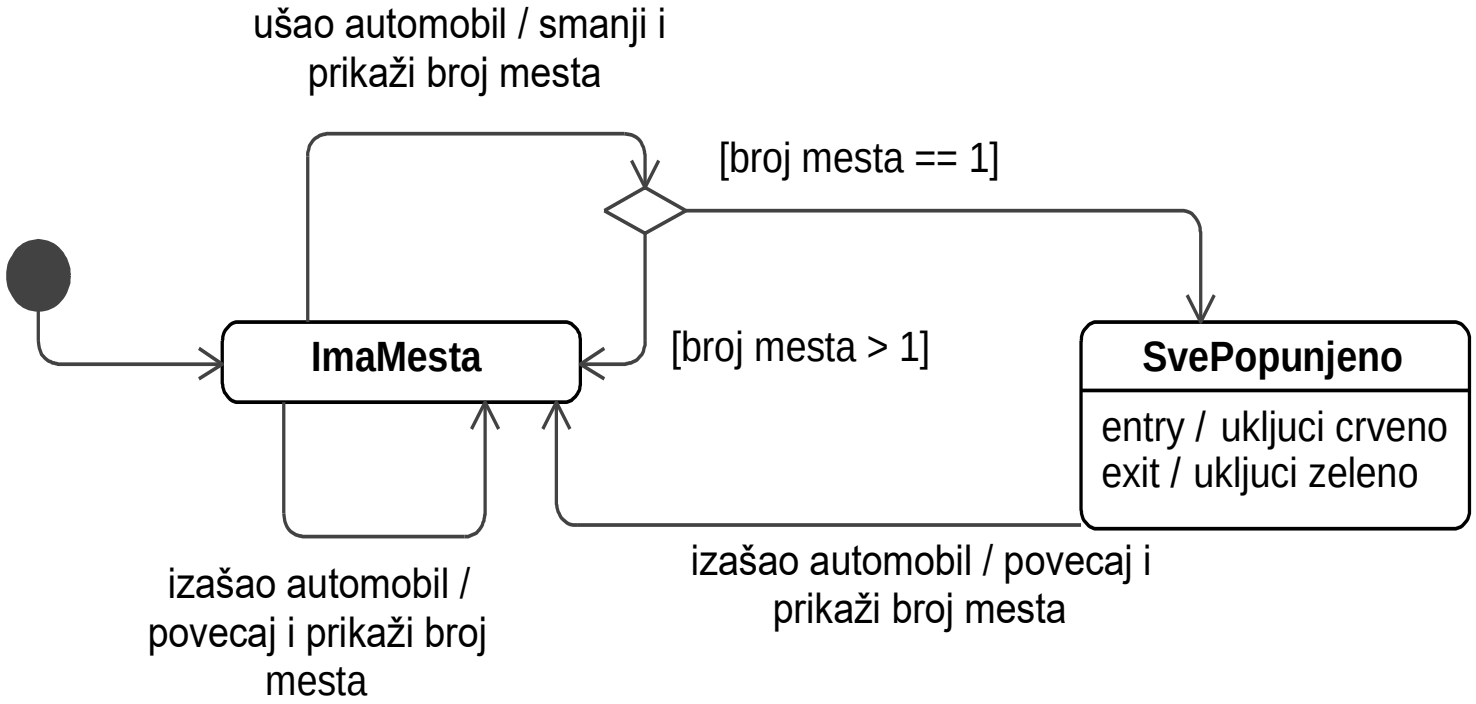


Stanje

- Naziv stanja
- Akcije:
  - u trenutku ulaska u stanje (entry),
  - u trenutku izlaska iz stanja (exit),
  - tokom boravka u stanju (do)
  - u okviru internih tranzicija

Primer 4 – dijagram prelaza stanja za kontroler sa dodatim akcijama

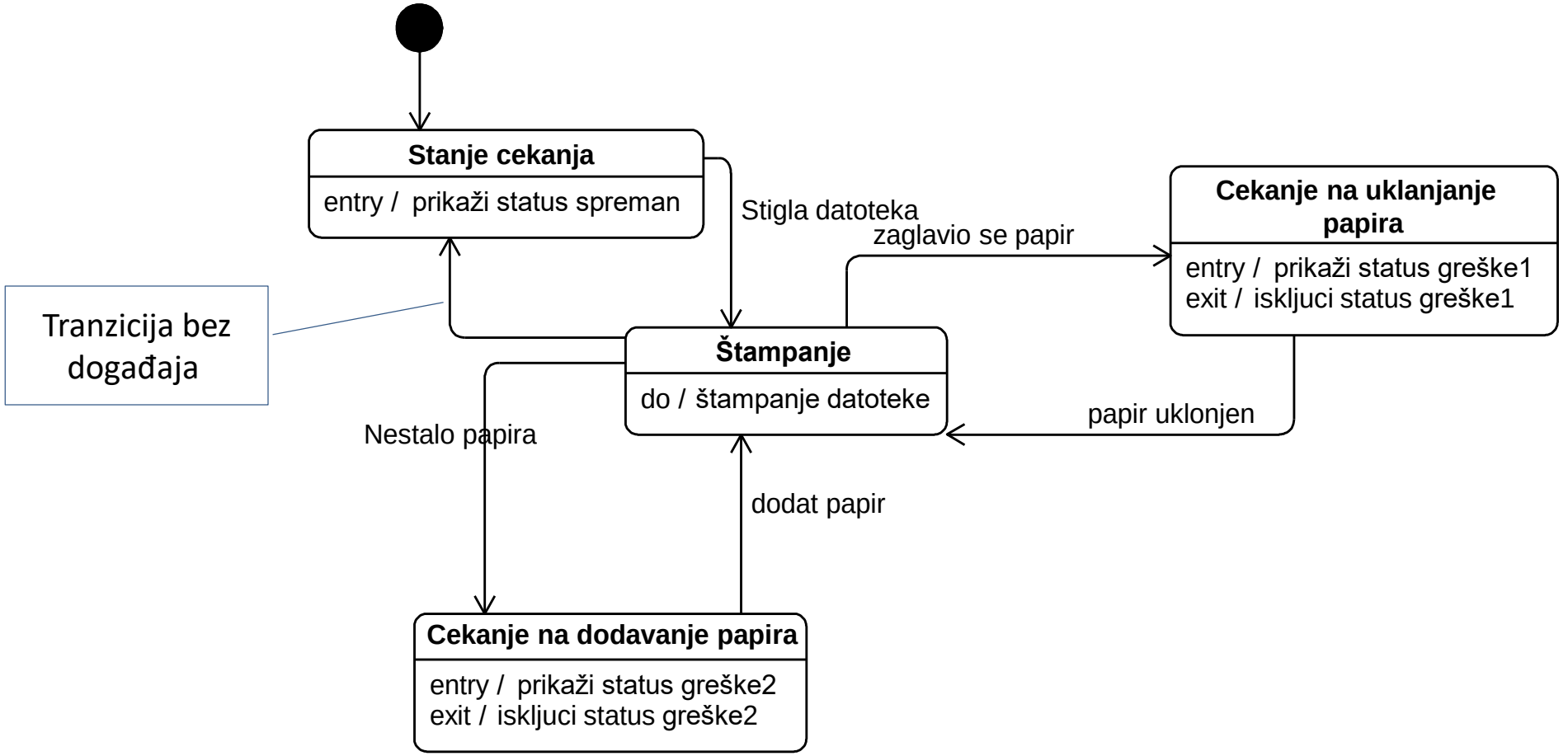
Tranzicija: događaj [uslov] / akcija



Primer 5 – rad jednostavnog štampača

- Kada se uključi, štampač se nalazi u stanju čekanja na datoteku koju treba da štampa. U tom trenutku štampač treba da prikaže status koji označava da je spreman. Kada datoteka stigne, počinje sa štampom, a status se menja u „zauzet“. Po završetku štampe, štampač se vraća u stanje čekanja na novi dokument i ponovo prikazuje status „spreman“.
- Ako se prilikom štampe zaglavi papir, štampač treba da prikaže odgovarajući status greške i da čeka da neko ukloni papir. Po uklanjanju papira, nastavlja sa štampom, a status menja na „zauzet“.
- Ako prilikom štampe nestane papira, štampač prikazuje status greške i čeka na dopunu papira. Kada se dopuna obavi, nastavlja sa štampom i prikazuje status „zauzet“.
- Radi jednostavnosti, pretpostavimo da štampač može da prima nove datoteke samo dok je u stanju čekanja na datoteku.

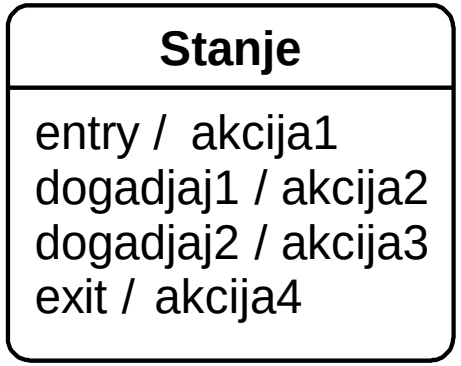
Dijagram prelaza stanja za štampač – jedno od mogućih rešenja



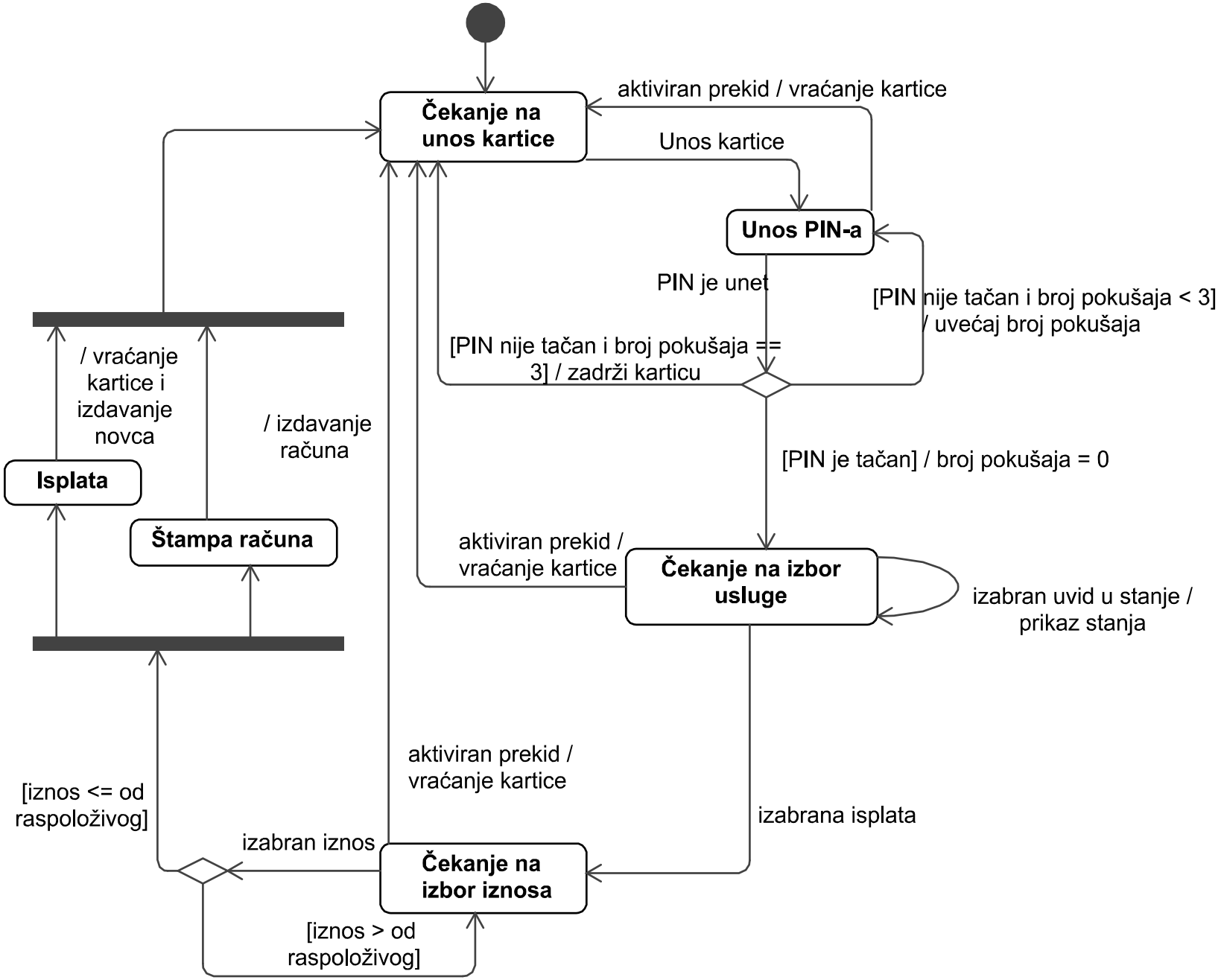
Tranzicija ne mora da ima događaj koji je aktivira *jedino* ako izlazi iz stanja koje ima **do** akciju, što znači da se stanje napušta u trenutku kada je posao koji obavlja završen!

Interne tranzicije

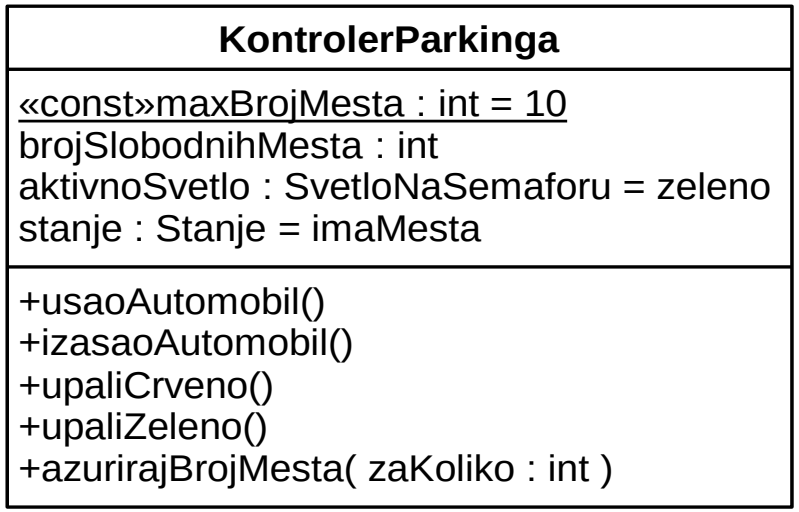
- Kada nam je potrebna tranzicija koja izvršava akciju kao odgovor na neki događaj i zatim se vraća u isto stanje, a pri tome ne želimo da se aktiviraju **entry**, **exit** i **do** akcije datog stanja, možemo koristiti interne tranzicije

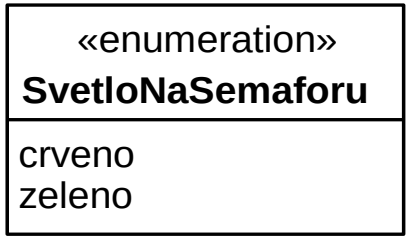


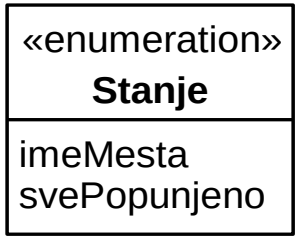
Paralelno izvršavanje



Preslikavanje na programski kod



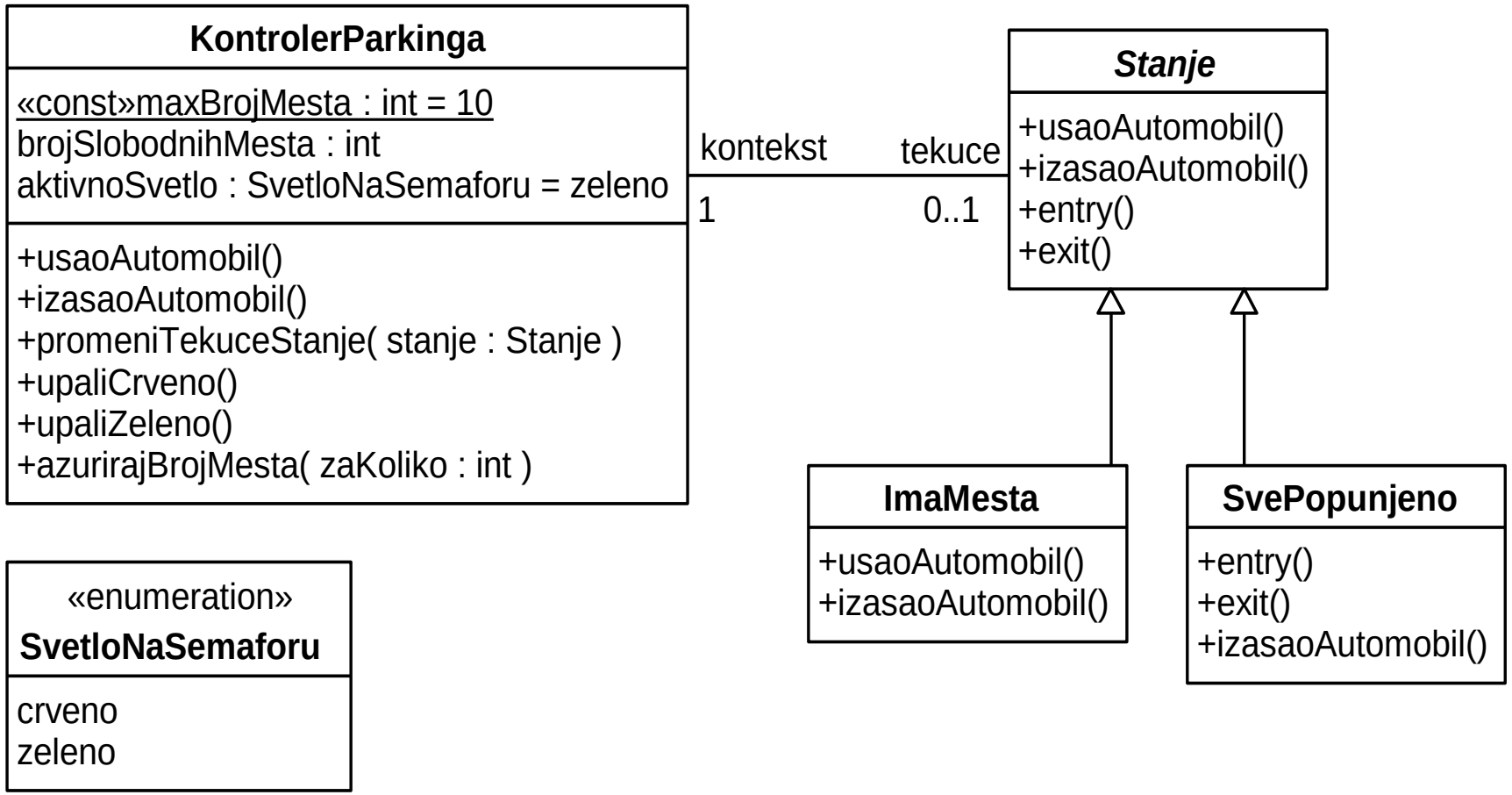




Ne preporučuje se!

```
public Boolean izasaoAutomobil() {
  switch(stanje) {
     case IMA_MESTA:
     //          ...
     break;
     case SVE_POPUNJENO:
     //          ...
     break;
  }
  ...
}
```

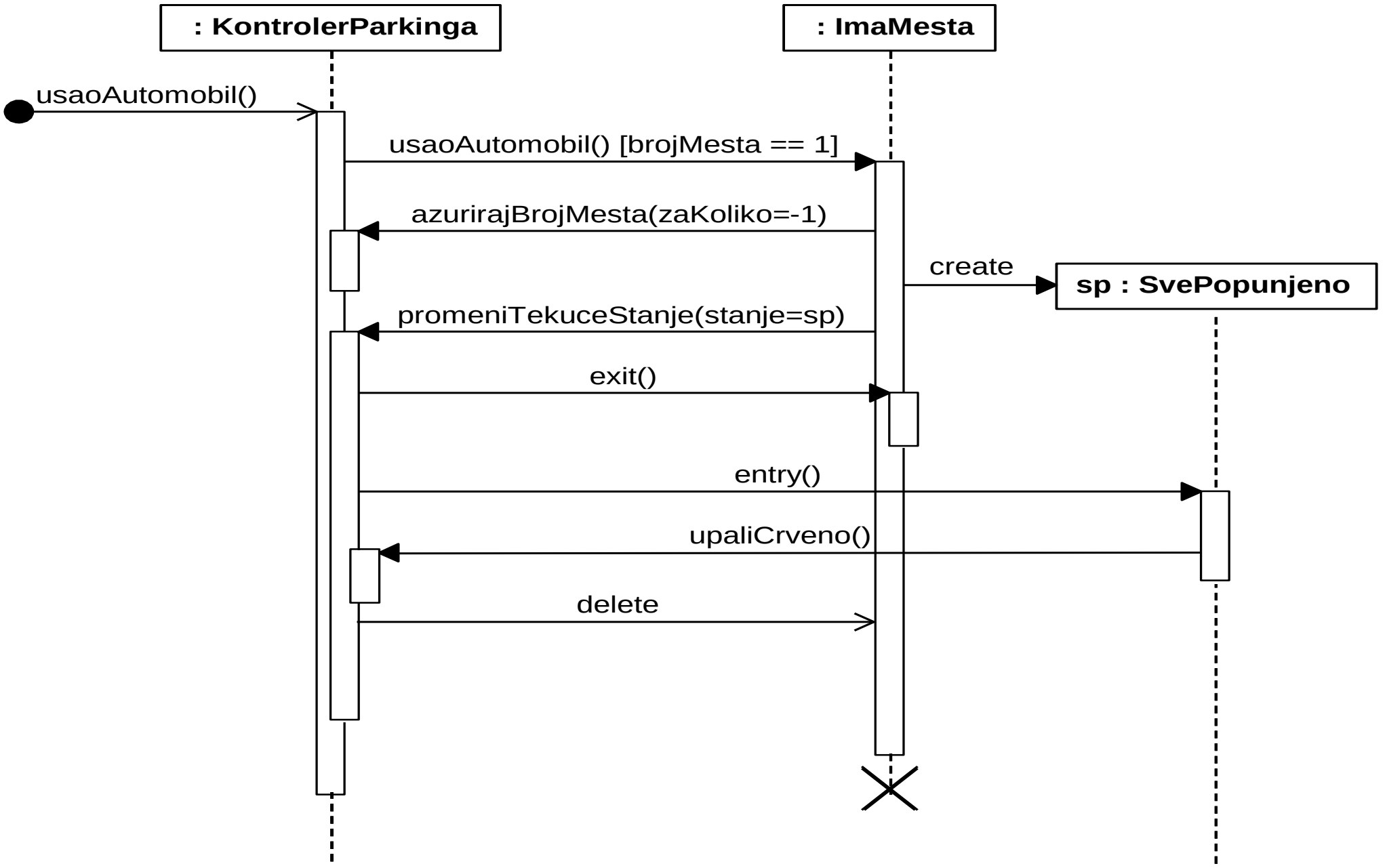
State šablon



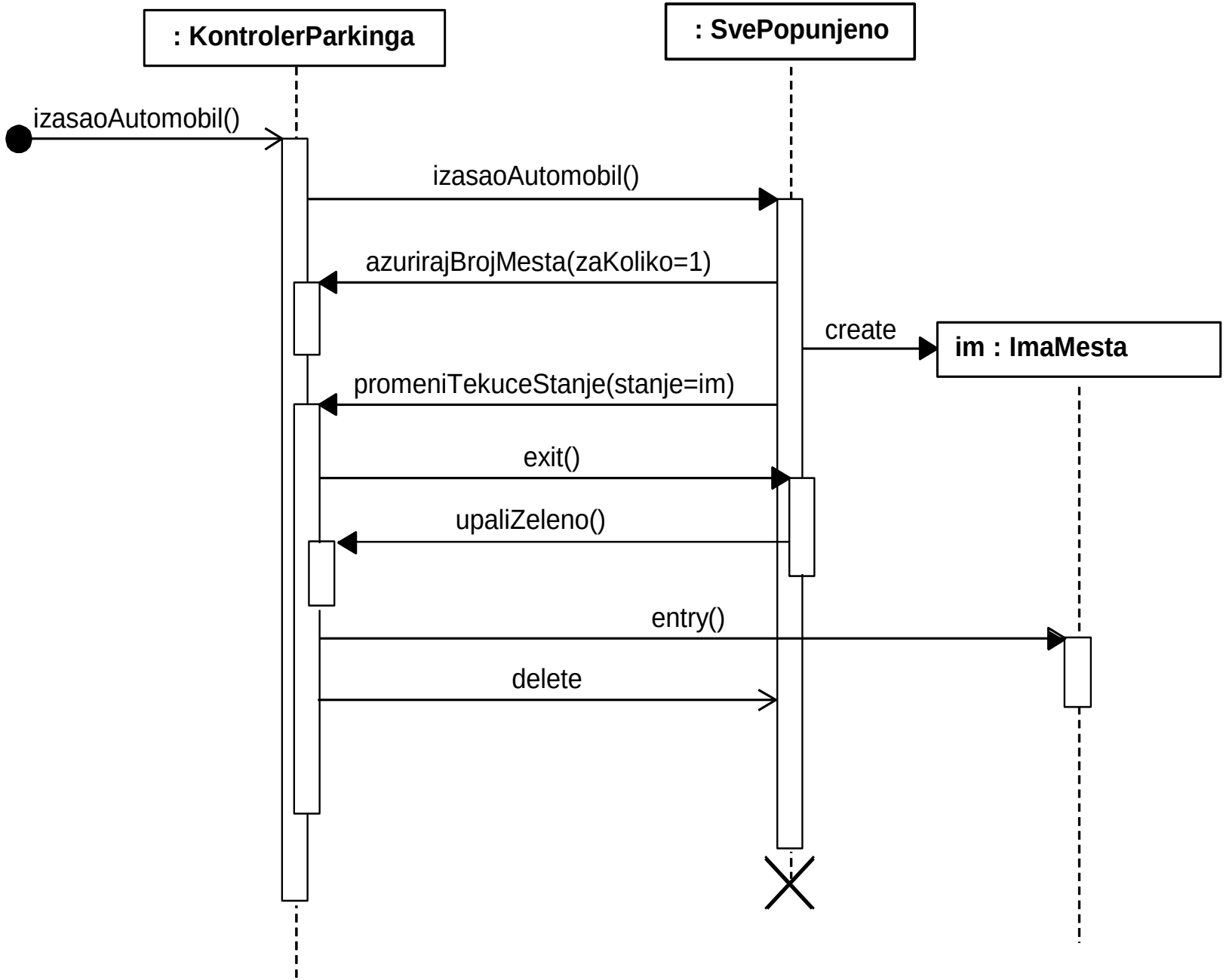
Zadatak

- Pogledati ostavljeni projekat UpravljanjeParkingom i proći debugger-om kroz metode koje implementiraju događaje za ulazak i izlazak automobila u okviru klase `KontrolerParkinga` sa ulaskom u metode koje se pozivaju.

Tranzicija u stanje „SvePopunjeno“ kada se zauzme poslednje slobodno mesto



Tranzicija u stanje „ImaMesta“ kada prvi automobil izađe sa popunjenog parkinga



Literatura

1. James Rumbaugh, Ivar Jacobson, Grady Booch, The Unified Modeling Language Reference Manual, Second Edition, Addison-Wesley, 2004
2. Scott W. Ambler, The Object Primer: Agile Model-Driven Development with UML 2.0, Cambridge University Press, 2004
3. M. Fowler, UML Distilled - A Brief Guide to the Standard Object Modeling Language, Third Edition, Addison Wesley, Boston, 2004.