Datum: 15.05.2018.

Dijagram stanja

Dijagram stanja opisuje ponašanje instanci klasa i pripada skupu dinamičkih UML dijagrama (kolaboracijski, sekvencijalni i dijagram aktivnosti). Dijagram stanja ima strukturu grafa i zasniva se na teoriji automata. Osnovni gradivni elementi dijagrama stanja su: stanja (states) i prelazi između stanja (transitions).

Stanje može sadržavati dijagrame i u tom slučaju se naziva ***složeno stanje*** (composite state). Složeno stanje sadrži dijagram koji ga detaljnije opisuje i kreira se na isti način kao i složena aktivnost na dijagramu aktivnosti (desni klik -> Add -> Diagram).

U prozoru osobina dijagrama stanja (Statechart diagram properties) biramo za koju klasu, slučaj upotrebe ili komponentu radimo dijagram stanja. Novi dijagram stanja nema zadanu klasifikaciju – potrebno ju je odrediti, dok pod-dijagrami nasljeđuju klasifikaciju od direktno nadređenog stanja.

Osnovni elementi dijagrama stanja

Osnovni elementi dijagrama stanja su:

- **Početno stanje (start)**
- **Stanje (state)**
- **Završno stanje (end)**
- **Prelaz iz jednog stanja u drugo stanje (transition)**
- **Čvorište u dijagramu (junction point)**

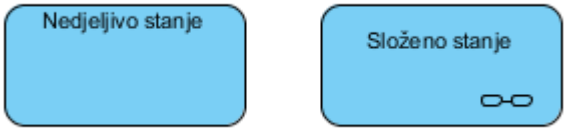
Početno stanje ili start ima isto značenje u dijagramu stanja i dijagramu aktivnosti. Svaki dijagram stanja može imati jedno i samo jedno početno stanje.

Stanje predstavlja okolnost u toku životnog vijeka objekta, slučaja upotrebe ili komponente.

Stabilnost i trajnost su osnovne karakteristike stanja. Događaji i uslovi prelaza iz jednog stanja u drugo definiraju stabilnost ili postojanost stanja. Neke aktivnosti mogu biti pridružene stanju (posebno kada objekt ulazi u stanje ili izlazi iz stanja) ili kada se događaju unutar stanja, a ne uzrokuju promjenu stanja (interni prelazi stanja).

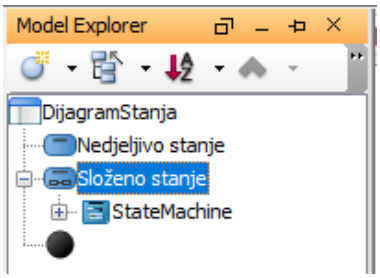
Stanje može biti nedjeljivo (Atomic state) ili složeno (Decomposed state). **Nedjeljivo stanje** ne sadrži podređena stanja, dok **složeno stanje** koristi podstanja za detaljniji opis. Složeno stanje koristi podstanja za detaljniji opis svog ponašanja. Svako stanje može sadržavati više stanja.

Simbol nedjeljivog i složenog stanja je prikazan ispod. Simbol složenog stanja sadrži ikonu dijagrama stanja u donjem desnom uglu.



Sva podstanja zajedno sa nadređenima složenim stanjem imaju zajednički prostor imenovanja (namespace).

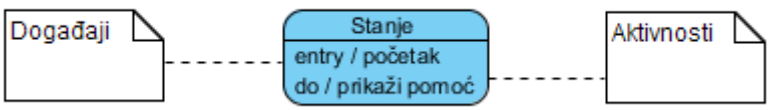
Sva podstanja jednog stanja se pojavljuju u explorer-u u hijerarhijskoj strukturi.



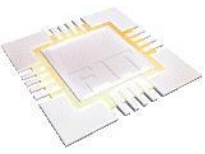
Definiranje aktivnosti u dijagramu stanja

U OOM aktivnosti se izvršavaju u momentu ulaska ili izlaska iz stanja ili kad je prelazak sa jednog stanja na drugo (transition) aktiviran.

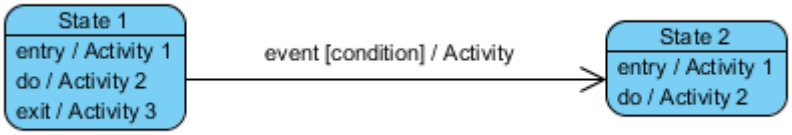
Aktivnosti predstavljaju specifikaciju izraza, događaju se u posebnim situacijama i mogu sadržavati predefinirane događaje: entry, do i exit, interne prelaze ili događaje koje kreira korisnik. Interni prelaz je onaj koji ne uzrokuje promjenu stanja, ali obavljaju aktivnosti kada ih događaji pokrenu.



Stanje može sadržavati više aktivnosti, a prelazom sa stanja na stanje može se izvršiti samo jedna aktivnost.



Na sljedećoj slici mogu se vidjeti aktivnosti stanja i aktivnosti prelaza sa stanja na stanje zajedno sa naredbama za izvršenje aktivnosti:



Osobine aktivnosti:

Osobina	**Opis**
Name	Naziv aktivnosti
Precondition	Preduslovi koji moraju biti ispunjeni da bi se aktivnost izvršila.
Postcondition	Rezultati izvršenja aktivnosti.
Description	Kratak opis aktivnosti.
Trigger event	Događaji: entry, do i exit. Također se mogu kreirati novi događaji u sklopu novih regija. Opisuje ulogu koju igraju aktivnosti u stanju ili događaje koji pokreću njihovo izvršenje.
Parameters	Parametri aktivnosti koji su analogni parametrima metoda unutar dijagram klasa.

Definiranje prelaza sa stanja na stanje (transitions)

Prelaz (transition) je usmjerena veza između stanja, koja ukazuje da jedan element može preći iz jednog stanja u drugo kada se desi određeni događaj i kada su ispunjeni postavljeni uslovi.

Prelazi mogu postojati između sljedećih elemenata dijagrama stanja:

Od\do	**Start**	**Stanje**	**Junction point**	**End**
Start	--	++	++	--
Stanje	--	++	++	++
Junction point	--	++	++	++
End	--	--	--	--

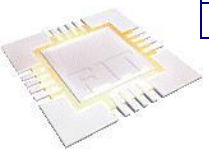
++ = dozvoljen

"--" = nije dozvoljen

Prelazi su refleksivni kada je isto početno i završno stanje prelaza.

Osobine prelaza:

Osobina	**Opis**
Name	
Description	
Source	Izvorno stanje



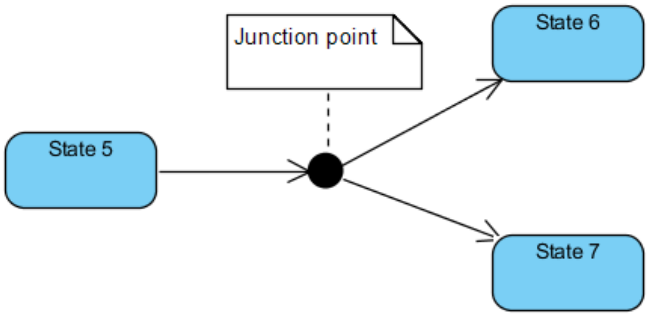
Target	Ciljno stanje
Flow Type ili Kind	Tip prelaza.
Guard	Predstavlja uslov koji može biti pridružen prelazu.
Trigger event	Read-only listbox; predstavlja sve događaje koji pokreću prelaz.
Effect	Aktivnost koja se izvršava kada se prelaz pokrene.
Operation	Read-only listbox; operacija izvornog stanja prelaza.
Operation arguments	Argumenti događaja.

Flow type	**Opis**
Success	Definira uspješan tok
Timeout	Definira timeout limit
Exception	Predstavlja izuzetak

Napomena: Prethodno su izdvojene najčešće korištene osobine aktivnosti i prelaza, s tim da su one na drugačiji način organizovane u sklopu različitih case alata.

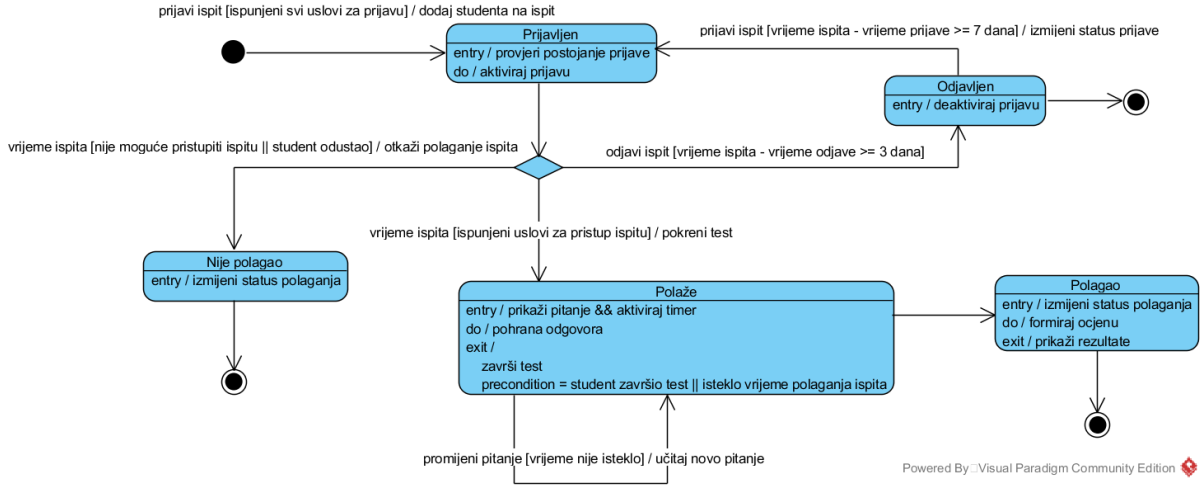
Definiranje čvorišta (junction points)

„Junction point" (tačka grananja) ima nekoliko ulaznih i izlaznih prelaza (transition) i koristi se za ujedinjenje više prelaza ili grananje jednog prelaza u više. Tačke prijelaza nije dozvoljeno kopirati.



Zadaci

Za izradu osnovnog primjera dijagrama stanja odabrana je klasa *Ispit*, te su razrađeni mogući prelazi između sljedećih stanja: Prijavljen, Polaže, Polagao, Odjavljen i Nije polagao.

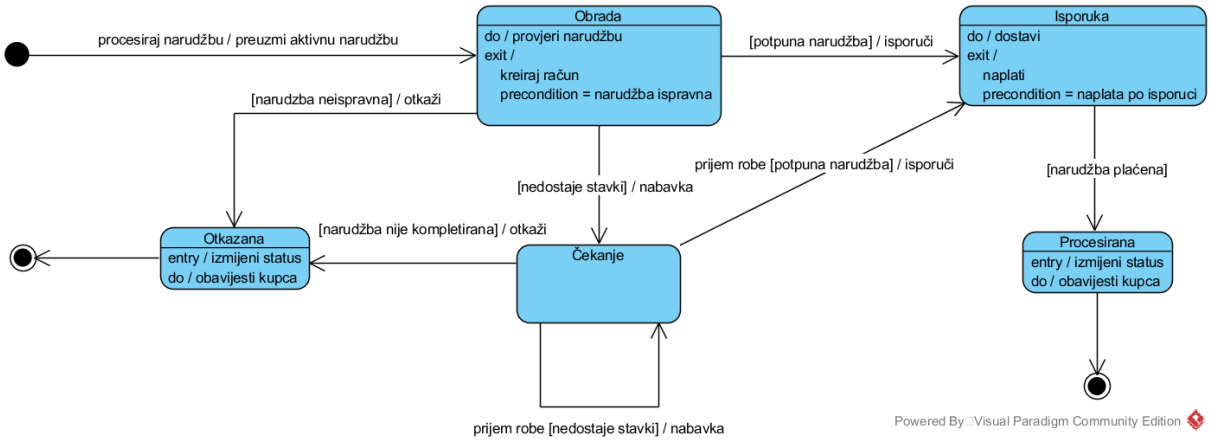


Zadatak 1:

Korištenjem dijagrama stanja modelirati postupak procesiranja narudžbe.

U toku procesiranja potrebno je osigurati da sve stavke budu dostupne prije isporuke. Ukoliko određena stavka nije pristigla, narudžba prelazi u stanje čekanja. Prije isporuke moguće je otkazati aktivnu narudžbu. Kada se prikupe sve stavke narudžbe, ona prelazi u stanje isporuke, nakon čega se označava kao procesirana, ali pod uslovom da je plaćanje uspješno obavljeno.

Prijedlog rješenja:



Zadatak 2:

Potrebno je kreirati web aplikaciju za rezervaciju kino ulaznica. Izdvojeni su sljedeći funkcionalni zahtjevi:

- Korisnici sistema mogu biti administratori (Ime, Prezime, E-mail, Telefon, Datum zaposlenja), članovi kino kluba (Ime, Prezime, E-mail, Datum registracije, Status), te posjetioci stranice koji pristupaju javnom dijelu aplikacije. Posjetioci stranice nakon registracije postaju članovi kino kluba.
- Filmovi (Naziv, Datum objavljivanja, Producent, Popularnost, Aktivan) su organizovani po žanrovima. Također je potrebno voditi evidenciju o glumcima (Ime, Prezime, Datum rođenja, Popularnost), te ih asocirati sa filmovima i obavezno pohraniti opis uloge koja im pripada. Administrator se brine za evidenciju filmova, glumaca, kao i projekcija pojedinih filmova. Svi korisnici sistema mogu pretraživati filmove po žanru i nazivu i opcionalno za odabrani film pregledati i glumačku postavu.
- Članovi kluba mogu da rezervišu kartu za odabranu projekciju filma. Raspored prikazivanja filmova je dostupan svim korisnicima sistema i on uključuje i najave novih filmova, međutim za njih nije moguće rezervisati kartu. Uz svaku rezervisanu kartu članovi biraju broj sjedišta. Pri tome nije moguće rezervisati prethodno zauzeta sjedišta. Nakon uspješne rezervacije karte, član dobiva njenu elektronsku verziju putem mail-a. Po završetku određene projekcije filma, članovi imaju mogućnost ocjene filma ukoliko je kupljena karta upotrijebljena.

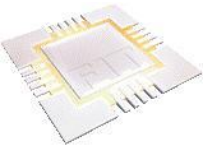
Kreirati dijagram stanja za praćenje stanja objekata klase Karta (moguća stanja: slobodna, rezervisana, upotrijebljena , otkazana).

Ispitni zadatak:

Za potrebe što efikasnije organizacije fudbalske lige postavljeni su zahtjevi za implementacijom softverskog rješenja koje treba da podrži njena osnovna pravila i procedure. Funkcionalni zahtjevi se u osnovi tiču rasporeda održavanja utakmica i bodovne tabele timova. Glavni korisnici softvera će biti administratori i menadžeri timova.

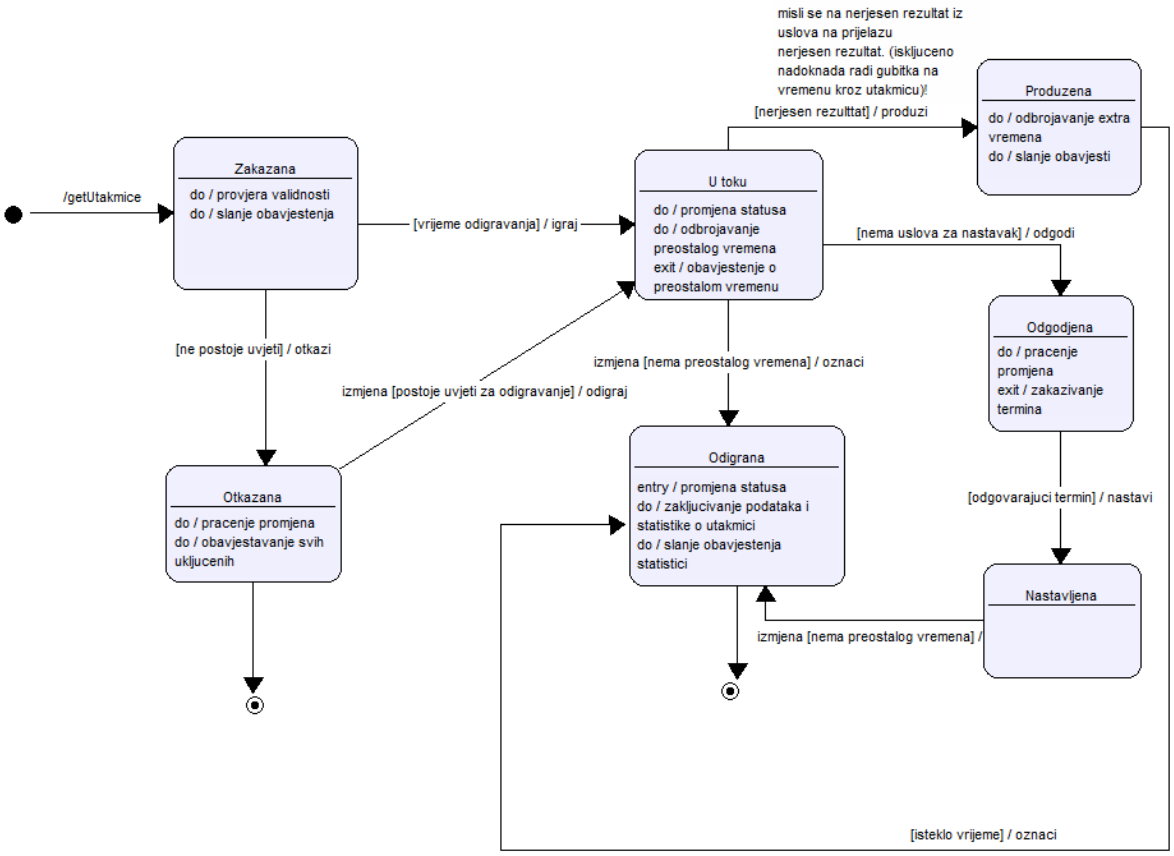
- unosu generalnih podataka o timovima (*Naziv, DatumOsnivanja, Grad, Logo*) i sezonskim rezultatima svakog tima (*BrOdigranihSusreta, BrPobjeda, BrNerijesenih, BrIzgubljenih, BrBodova*) brine se administrator sistema. On također definiše raspored održavanja utakmica timova (*Datum, Vrijeme, Rezultat, BrZutihKartona, BrCrvnihKartona*), te podatke o angažmanu sudija na pojedinim utakmicama. Po unosu rezultata utakmice sistem automatski ažurira sumarne rezultate timova (svaka pobjeda timu donosi 3 boda, dok neriješen rezultat timovima donosi po 1 bod) i administratoru prikazuje izmijenjene podatke o timovima (bodovna tabela).
- Menadžer svakog tima definiše igrače (*Ime, Prezime, DatumRodjenja, Pozicija*) tima, te prvu postavu **za svaku utakmicu**. Prvu postavu čini ukupno 11 igrača. Pored toga potrebno je definisati 5 zamjenskih igrača. Osim navedenih mogućnosti, menadžeri dobivaju notifikacije nakon svake odigrane utakmice tima, odnosno nakon svake promjene rezultata tima.

Kreirati dijagram stanja za praćenje stanja objekata klase Utakmica (zakazana, u toku, odigrana, otkazana). Po potrebi možete definisati i dodatna stanja.



Studentsko rješenje:

Ispod je prijedlog rješenja studenta koji je osvojio maksimalan broj bodova (20). Dijagram je kreiran u alatu Open ModelSphere.



Prateće video lekcije možete pogledati u sklopu YouTube kanala na sljedećim linkovima:

- Dijagram stanja (Visual Paradigm 14)
- Dijagram stanja (Open ModelSphere 3.2)

Literatura

1. Martin Fowler, UML Distilled Third Edition: A brief guide to the standard object modeling language, 2004. Download (Dostupno: 15.05.2018.).
2. Craig Larman, Applyng UML and paterns, Prentice Hall, 2004.
3. https://www.visual-paradigm.com/
4. https://www.youtube.com/user/VisualParadigm/
5. http://www.uml.org/