Univerzitet u Beogradu

Fakultet organizacionih nauka

Laboratorija za softversko inženjerstvo

Seminarski rad iz predmeta

Projektovanje softvera

Tema: Deo softverskog sistema za praćenje jednodnevnih takmičenja u ribolovu

|  |  |
| --- | --- |
| Profesor:  dr Siniša Vlajić | Student:  Aleksa Ristić 804/13 |
|  |  |

Beograd, 2021.

Sadržaj

[1. Korisnički zahtev 3](#_Toc536392291)

[1.1 Verbalni opis 3](#_Toc536392292)

[1.2 Slučajevi korišćenja 4](#_Toc536392293)

[SK 1: Slučaj korišćenja – Unos novog takmičara 6](#_Toc536392294)

[SK 2: Slučaj korišćenja – Pregled takmičara 8](#_Toc536392295)

[SK 3: Slučaj korišćenja – Izmena takmičara 9](#_Toc536392296)

[SK 4: Slučaj korišćenja – Brisanje takmičara 10](#_Toc536392297)

[SK 5: Slučaj korišćenja – Unos takmičenja 11](#_Toc536392298)

[SK 6: Slučaj korišćenja – Izmena takmičenja 12](#_Toc536392299)

[SK 7: Slučaj korišćenja – Brisanje takmičenja 14](#_Toc536392300)

[SK 8: Slučaj korišćenja – Pregled takmičenje 15](#_Toc536392301)

[SK 9: Slučaj korišćenja – Unos rezultata takmičenja 15](#_Toc536392302)

[2.Analiza 17](#_Toc536392303)

[**2.1. . Ponašanje softverskog sistema - Dijagram sekvenci slučaja poseta** 17](#_Toc536392304)

[DS 1: Dijagram sekvenci slučaja korišćenja – Unos takmičara 18](#_Toc536392305)

[DS 2: Dijagram sekvenci slučaja korišćenja – Pregled takmičara 20](#_Toc536392306)

[DS 3: Dijagram sekvenci slučaja korišćenja – Izmena takmičara 22](#_Toc536392307)

[DS 4: Dijagram sekvenci slučaja korišćenja – Brisanje takmičara 25](#_Toc536392308)

[DS 5: Dijagram sekvenci slučaja korišćenja – Unos takmičenja 27](#_Toc536392309)

[DS 6: Dijagram sekvenci – Izmena takmičenja 29](#_Toc536392310)

[DS 7: Dijagram sekvenci – Brisanje takmičenja 33](#_Toc536392311)

[SK 9: Slučaj korišćenja – Pregled takmičenja 35](#_Toc536392312)

[SK 10: Slučaj korišćenja – Unos rezultata takmičenja 36](#_Toc536392313)

[2.2 Ponašanje softverskog sistema – Definisanje ugovora o sistemskim operacijama 41](#_Toc536392314)

[2.3 Struktura softverskog sistema – Konceptualni (domenski) model 49](#_Toc536392315)

[Struktura softverskog sistema 50](#_Toc536392316)

[2.4 Relacioni model 51](#_Toc536392317)

[3. PROJEKTOVANJE 56](#_Toc536392318)

[3.1 Arhitektura Softverskog Sistema 56](#_Toc536392319)

[3.2. Projektovanje korisničkog interfejsa 57](#_Toc536392320)

[1.2.1. Projektovanje ekranskih formi 58](#_Toc536392321)

[SK 1: Slučaj korišćenja – Unos novog takmičara 58](#_Toc536392322)

[SK 2: Slučaj korišćenja – Pregled takmičara 62](#_Toc536392323)

[SK 3: Slučaj korišćenja – Izmena takmičara 63](#_Toc536392324)

[SK 4: Slučaj korišćenja – Brisanje takmičara 64](#_Toc536392325)

[SK 5: Slučaj korišćenja – Unos takmičenja 65](#_Toc536392326)

[SK 6: Slučaj korišćenja – Izmena takmičenja 66](#_Toc536392327)

[SK 7: Slučaj korišćenja – Brisanje takmičenja 68](#_Toc536392328)

[SK 8: Slučaj korišćenja – Pregled takmičenje 69](#_Toc536392329)

[SK 9: Slučaj korišćenja – Unos rezultata takmičenja 69](#_Toc536392330)

[3.2.3. Projektovanje kontrolera korisničkog interfejsa 71](#_Toc536392331)

[3.3. Projektovanje aplikacione logike 73](#_Toc536392332)

[3.3.1. Kontroler aplikacione logike 73](#_Toc536392333)

[3.3.2. Sistemske operacije 77](#_Toc536392334)

[3.3.3. Domenske klase 82](#_Toc536392335)

[3.3.4 Broker baze podataka 83](#_Toc536392336)

[3.3.5. Projektovanje skladišta podataka 85](#_Toc536392337)

[4. IMPLEMENTACIJA 88](#_Toc536392338)

[5. TESTIRANJE 90](#_Toc536392339)

[6. LITERATURA 91](#_Toc536392340)

# Korisnički zahtev

## 1.1 Verbalni opis

Klub ribolovaca odlučio je da radi bolje organizacije ribolovačkih takmičenja napravi aplikaciju koja ce omogućiti čuvanje i praćenje takmičenja kao i njegovih takmičara i delegata.

Aplikacija omogućava prijavljivanje korisnika (delegata) u sistem. U sistemu će se voditi evidencija o takmičarima i takmičenjima.

Aplikacija omogućava unos (kreiranje) novog takmičenja, odnosno novih takmičara, promenu podataka o njima, ukoliko postoji potreba za tim, kao i pretraživanje takmičara i takmičenja. U aplikaciji je, takođe, bilo neophodno obezbediti i brisanje i ažuriranje pomenutih podataka.

## 1.2 Slučajevi korišćenja

U ovom slučaju uočeni su sledeći slučajevi poseta i oni su prikazani na slici 1:

1. Unos novog takmičara
2. Pregled takmičara
3. Izmena takmičara
4. Brisanje takmičara
5. Unos takmičenja (složen SK)
6. Pregled takmičenja
7. Izmena takmičenja (složen SK)
8. Brisanje takmičenja
9. Unos rezultata takmičenja



Slika 1. Dijagram slučajeva korišćenja

### 

## SK 1: Slučaj korišćenja – Unos novog takmičara

**Naziv SK**

Unos *novog takmičara*

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos novog takmičara. Učitana je lista svih zemalja.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da kreira *novog takmičara*. (APSO)
2. **Sistem** kreira *novog takmičara*. (SO)
3. **Sistem** prikazuje **Korisniku** *novog takmičara* i poruku:"**Sistem** je kreirao novog *takmičara*". (IA)
4. **Korisnik** unosi podatke o *novom takmičaru*. (APUSO)
5. **Korisnik** proverava da li je korektno uneo podatke o *novom takmičaru*. (ANSO)
6. **Korisnik** poziva **sistem** da zapamti podatke o *novom takmičaru*. (APSO)
7. **Sistem** pamti podatke o *novom takmičaru.* (SO)
8. **Sistem** prikazuje **Korisniku** poruku: “**Sistem** je uspešno zapamtio *takmičara*.”. (IA)

**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da omogući kreiranje novog takmičara on prikazuje **korisniku** poruku: „**Sistem** ne može da kreira novog takmičara“. Prekida se izvršavanje scenarija (IA)

8.1. Ukoliko **sistem** ne može da izvrši unos novog takmičara on prikazuje **korisniku** poruku : „**Sistem** ne može da zapamti novog takmičara*.*“ (IA)

## SK 2: Slučaj korišćenja – Pregled takmičara

**Naziv SK**

Pregled takmičara

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičarima*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednost po kojoj pretražuje *takmičare*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičare* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičare* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** *takmičare* i poruku:"**Sistem** je našao *takmičare* po zadatoj vrednosti". (IA)
5. **Korisnik** bira *takmičara*. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o *takmičaru*. (APSO)
7. **Sistem** učitava podatke o *takmičaru*. (SO)
8. **Sistem** prikazuje **korisniku** *takmičara* i poruku:"**Sistem** je uspešno učitao podatke o *takmičaru*". (IA)

**Alternativna scenarija:**

4.1. Ukoliko **sistem** ne može da nađe *takmičare* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičare*  ”. Prekida se izvršavanje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *takmičara*, on prikazuje **korisniku** poruku:"Sistem ne može da učita *takmičara*". (IA)

## SK 3: Slučaj korišćenja – Izmena takmičara

**Naziv SK**

Izmena takmičara

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos novog takmičara. Učitana je lista svih zemalja.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednost po kojoj pretražuje *takmičare*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičare*  po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičare* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** *takmičare* i poruku:" **Sistem** je uspešno pronašao *takmičare* po zadatim vrednostima ". (IA)
5. **Korisnik** bira *takmičara* kojeg želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičaru*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičaru*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičaru* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičaru*". (IA)
9. **Korisnik** menja podatke o *takmičaru*. (APUSO)
10. **Korisnik** poziva **sistem** da zapamti podatke o *takmičaru.* (APSO)
11. **Sistem** pamti podatke o *takmičaru*. (SO)
12. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspesno izmenio *takmičara*.” (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičara*  on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičara*”. Prekida se izvršavanje scenarija. (IA)

8.1. Ukoliko sistem ne može da nađe takmičara on prikazuje korisniku poruku:"Sistem ne može da nađe izabranog takmičara". Prekida se izvršavanje scenarija. (IA)

12.1. Ukoliko **sistem** ne može da zapamti podatke o *takmičaru* on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *takmičara*”. (IA)

## SK 4: Slučaj korišćenja – Brisanje takmičara

**Naziv SK**

Brisanje takmičara

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik** i **sistem** (program)

**Preduslov: Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičarima*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednost po kojoj pretražuje *takmičare*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičare*  po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičare* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** *takmičare* i poruku:" **Sistem** je uspešno pronašao *takmičare* po zadatim vrednostima ". (IA)
5. **Korisnik** bira *takmičara* kojeg želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičaru*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičaru*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičaru* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičaru*". (IA)
9. **Korisnik** poziva **sistem** da obriše podatke o *takmičaru.* (APSO)
10. **Sistem** briše podatke o *takmičaru*. (SO)
11. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspesno obrisao *takmičara*.” (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičara*  on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičara*”. Prekida se izvršavanje scenarija. (IA)

8.1. Ukoliko sistem ne može da nađe takmičara on prikazuje korisniku poruku:"Sistem ne može da nađe izabranog takmičara". Prekida se izvršavanje scenarija. (IA)

11.1. Ukoliko **sistem** ne može da obriše podatke o *takmičaru* on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše *takmičara*”. (IA)

## SK 5: Slučaj korišćenja – Unos takmičenja

**Naziv SK**

Unos takmičenja

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos takmičenja. Učitana je lista takmičarskih staza i takmičara.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da kreira novo *takmičenje*. (APSO)
2. **Sistem** kreira novo *takmičenje*. (SO)
3. **Sistem** prikazuje **korisniku** novo *takmičenje* i poruku:"**Sistem** je uspešno kreirao novo *takmičenje*". (IA)
4. **Korisnik** unosi podatke o novoj *takmičenju.* (APUSO)
5. **Korisnik** poziva **sistem** da zapamti podatke o novoj *takmičenju*. (APSO)
6. **Sistem** pamti podatke o novoj *takmičenju*. (SO)
7. **Sistem** prikazuje korisniku poruku: „Sistem je uspesno zapamtio novo takmičenje“. (IA)

**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da kreira *novo takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da kreira *novo takmičenje* ”. Prekida se izvršenje scenarija. (IA)

7.1.Ukoliko **sistem** ne možeda zapamti podatke o *novoj takmičenju*, on prikazuje **korisnuku** poruku: “Sistem ne može da zapamti novo *takmičenje*.” (IA)

## SK 6: Slučaj korišćenja – Izmena takmičenja

**Naziv SK**

Izmena *takmičenja*

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik**  i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*. Učitana je lista takmičarskih staza i takmičara.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *takmičenja.* (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičenja* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
5. **Korisnik** bira *takmičenje* koje želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičenju*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)
9. **Korisnik** unosi (menja) podatke o *takmičenju*. (APUSO)
10. **Korisnik** poziva **sistem** da zapamti izmene podataka o *takmičenju* .(APSO)
11. **Sistem** pamti izmene o *takmičenju*. (SO)
12. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspesno izmenio *takmičenje*.“ (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. Prekida se izvršavanje scenarija. (IA)

12.1. Ukoliko **sistem** ne može da izmeni podatke o *takmičenju*  on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *takmičenje*”. (IA)

## SK 7: Slučaj korišćenja – Brisanje takmičenja

**Naziv SK**

Brisanje *takmičenja*

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik**  i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *takmičenja.* (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičenja* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
5. **Korisnik** bira *takmičenje* koje želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičenju*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)
9. **Korisnik** poziva **sistem** da obriše podatake o *takmičenju* .(APSO)
10. **Sistem** briše izmene o *takmičenju*. (SO)
11. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspesno obrisao *takmičenje*.“ (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. Prekida se izvršavanje scenarija. (IA)

11.1. Ukoliko **sistem** ne može da obriše podatke o *takmičenju*  on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše *takmičenje*”. (IA)

## SK 8: Slučaj korišćenja – Pregled takmičenje

**Naziv SK**

Pregled *takmičenje*

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik**  i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *takmičenja.* (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičenja* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
5. **Korisnik** bira *takmičenje* koje želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičenju*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. (IA)

## SK 9: Slučaj korišćenja – Unos rezultata takmičenja

**Naziv SK**

Unos rezultata *takmičenja*

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik**  i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *takmičenja.* (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičenja* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
5. **Korisnik** bira *takmičenje* koje želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičenju*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)
9. **Korisnik** unosi (menja) podatke o *takmičenju*. (APUSO)
10. **Korisnik** poziva **sistem** da zapamti izmene podataka o *takmičenju* .(APSO)
11. **Sistem** pamti izmene o *takmičenju*. (SO)
12. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspesno izmenio *takmičenje*.“ (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. Prekida se izvršavanje scenarija. (IA)

12.1. Ukoliko **sistem** ne može da izmeni podatke o *takmičenju*  on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *takmičenje*”. (IA)

# 2.Analiza

Nakon faze prikupljanja zahteva, prelazi se na sledeću fazu, fazu analize. Ova faza opisuje logičku strukturu i ponašanje sofvterskog sistema, tj. poslovnu logiku softverskog sistema.

* Ponašanje opisujemo pomoću:
* sistemskih dijagrama sekvenci
* ugovora o sistemskim operacijama
* Strukturu opisujemo pomoću:
* konceptualnog modela
* relacionog modela

**2.1. . Ponašanje softverskog sistema - Dijagram sekvenci slučaja poseta**

Ponašanje sistema se može opisati preko UML-ovih **sekvencnih dijagrama** [Larman]**,** odnosno preko **dijagrama saradnje**[JPRS]**.**

**Sistemski dijagram sekvenci** prikazuje, za izdvojeni scenario SK, događaje u određenom redosledu, koji uspostavljaju interakciju izmenu aktora i softverskog sistema.

**Događaj** koji **napravi** aktor je pobuda za poziv sistemske operacije. Preciznije rečeno, događaj koji napravi aktor prihvata primalac događaja koji nakon toga poziva sistemsku operaciju. To znači da aktor ne poziva sistemsku operaciju neposredno već to čini preko posrednika (primaoca događaja). Poziv sistemske operacije ukazuje na interakciju izmenu aktora i sistema. Za događaj koji predstavlja pobudu za poziv SO se često kaže da je to **sistemski događaj.**

### DS 1: Dijagram sekvenci slučaja korišćenja – Unos takmičara

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos novog takmičara. Učitana je lista svih zemalja i lista svih timova.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da kreira *novog takmičara*. (APSO)
2. **Sistem** prikazuje **Korisniku** *novog takmičara* i poruku:"**Sistem** je kreirao novog *takmičara*". (IA)
3. **Korisnik** poziva **sistem** da zapamti podatke o *novom takmičaru*. (APSO)
4. **Sistem** prikazuje **Korisniku** poruku: “**Sistem** je uspešno zapamtio *takmičara*.”. (IA)



**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da omogući kreiranje novog takmičara on prikazuje **korisniku** poruku: „**Sistem** ne može da kreira novog takmičara“. Prekida se izvršavanje scenarija (IA)



8.1. Ukoliko **sistem** ne može da izvrši unos novog takmičara on prikazuje **korisniku** poruku : „**Sistem** ne može da zapamti novog takmičara*.*“ (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **kreirajTakmicara**(*Takmicar*)
2. *signal* **zapamtiTakmicara**(*Takmicar)*
3. *signal* **vratiListuZemalja**(*List<Zemlja>)*

### DS 2: Dijagram sekvenci slučaja korišćenja – Pregled takmičara

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičarima*.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *takmičare* po zadatoj vrednosti. (APSO)
2. **Sistem** prikazuje **korisniku** *takmičare* i poruku:"**Sistem** je našao *takmičare* po zadatoj vrednosti". (IA)
3. **Korisnik** poziva **sistem** da učita podatke o *takmičaru*. (APSO)
4. **Sistem** prikazuje **korisniku** *takmičara* i poruku:"**Sistem** je uspešno učitao podatke o *takmičaru*". (IA)



**Alternativna scenarija:**

4.1. Ukoliko **sistem** ne može da nađe *takmičare* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičare*  ”. Prekida se izvršavanje scenarija. (IA)



8.1. Ukoliko **sistem** ne može da nađe *takmičara*, on prikazuje **korisniku** poruku:"Sistem ne može da učita *takmičara*". (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **pronadjiTakmicare**(Takmicar*,List<Takmicar>*)
2. *signal* **ucitajTakmicara**(*Takmicar)*

### DS 3: Dijagram sekvenci slučaja korišćenja – Izmena takmičara

**Preduslov: Sistem** je uključen. **Sistem** prikazuje formu za unos novog takmičara. Učitana je lista svih zemalja i lista svih timova.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *takmičare*  po zadatoj vrednosti. (APSO)
2. **Sistem** prikazuje **korisniku** *takmičare* i poruku:" **Sistem** je uspešno pronašao *takmičare* po zadatim vrednostima ". (IA)
3. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičaru*. (APSO)
4. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičaru* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičaru*". (IA)
5. **Korisnik** poziva **sistem** da zapamti podatke o *takmičaru.* (APSO)
6. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspesno izmenio *takmičara*.” (IA)



**Alternativna scenarija:**

4.1. Ukoliko **sistem** ne može da nađe *takmičara*  on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičara*”. Prekida se izvršavanje scenarija. (IA)



8.1. Ukoliko sistem ne može da nađe takmičara on prikazuje korisniku poruku:"Sistem ne može da nađe izabranog takmičara". Prekida se izvršavanje scenarija. (IA)



12.1. Ukoliko **sistem** ne može da zapamti podatke o *takmičaru* on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *takmičara*”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **vratiListuZemalja**(Zemlja*,List<Zemlja>*)
2. *signal* **pronadjiTakmicare**(Takmicar*,List<Takmicar>*)
3. *signal* **ucitajTakmicara**(*Takmicar)*
4. signal **zapamtiTakmicara**(Takmicar)

### DS 4: Dijagram sekvenci slučaja korišćenja – Brisanje takmičara

**Preduslov: Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičarima*. Učitana je lista svih zemalja.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *takmičare*  po zadatoj vrednosti. (APSO)
2. **Sistem** prikazuje **korisniku** *takmičare* i poruku:" **Sistem** je uspešno pronašao *takmičare* po zadatim vrednostima ". (IA)
3. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičaru*. (APSO)
4. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičaru* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičaru*". (IA)
5. **Korisnik** poziva **sistem** da obriše podatke o *takmičaru.* (APSO)
6. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspesno obrisao *takmičara*.” (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičara*  on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičara*”. Prekida se izvršavanje scenarija. (IA)



8.1. Ukoliko sistem ne može da nađe takmičara on prikazuje korisniku poruku:"Sistem ne može da nađe izabranog takmičara". Prekida se izvršavanje scenarija. (IA) 

11.1. Ukoliko **sistem** ne može da obriše podatke o *takmičaru* on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše *takmičara*”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **vratiListuZemalja**(Zemlja*,List<Zemlja>*)
2. *signal* **pronadjiTakmicare**(Takmicar*,List<Takmicar>*)
3. *signal* **ucitajTakmicara**(*Takmicar)*
4. signal **zapamtiTakmicara**(Takmicar)

### DS 5: Dijagram sekvenci slučaja korišćenja – Unos takmičenja

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos takmičenja. Učitana je lista takmičara i takmičarskih staza.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da kreira novo *takmičenje*. (APSO)
2. **Sistem** prikazuje **korisniku** novo *takmičenje* i poruku:"**Sistem** je uspešno kreirao novo *takmičenje*". (IA)
3. **Korisnik** poziva **sistem** da zapamti podatke o novoj *takmičenju*. (APSO)
4. **Sistem** prikazuje korisniku poruku: „Sistem je uspesno zapamtio novo takmičenje“. (IA)



**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da kreira *novo takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da kreira *novo takmičenje* ”. Prekida se izvršenje scenarija. (IA)



7.1.Ukoliko **sistem** ne možeda zapamti podatke o *novoj takmičenju*, on prikazuje **korisnuku** poruku: “Sistem ne može da zapamti novo *takmičenje*.” (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **vratiListuTakmicarskihStaza**(TakmicarskaStaza*,List<TakmicarskaStaza>*)
2. *signal* **vratiListuTakmicara**(Takmicar*,List<Takmicar>*)
3. *signal* **kreirajTakmičenje**(*Takmičenje)*
4. signal **zapamtiuTakmičenje**(Takmičenje)

### DS 6: Dijagram sekvenci – Izmena takmičenja

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*. Učitana je lista takmičara i takmicarskih staza.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
2. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
3. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
4. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)
5. **Korisnik** poziva **sistem** da zapamti izmene podataka o *takmičenju* .(APSO)
6. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspesno zapamtio *takmičenje*.“ (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)



8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. Prekida se izvršavanje scenarija. (IA)



12.1. Ukoliko **sistem** ne može da izmeni podatke o *takmičenju*  on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *takmičenje*”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

2. *signal***vratiListuTakmicarskihStaza**(TakmicarskaStaza*,List<TakmicarskaStaza>*)
3. *signal* **vratiListuTakmicara**(Takmicar*,List<Takmicar>*)
4. *signal* **pronadjiTakmičenja**(Takmičenje*,List<Takmičenje>*)
5. *signal* **ucitajTakmičenje**(*Takmičenje)*
6. signal **zapamtiTakmičenje**(Takmičenje)

## DS 7: Dijagram sekvenci – Brisanje takmičenja

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
2. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
3. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
4. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)
5. **Korisnik** poziva **sistem** da obriše podatake o *takmičenju* .(APSO)
6. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspesno obrisao *takmičenje*.“ (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)



8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. Prekida se izvršavanje scenarija. (IA)



11.1. Ukoliko **sistem** ne može da obriše podatke o *takmičenju*  on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše *takmičenje*”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **pronadjiTakmičenja**(Takmičenje*,List<Takmičenje>*)
2. *signal* **ucitajTakmičenje**(*Takmičenje)*
3. signal **obrisiTakmičenje**(Takmičenje)

## SK 9: Slučaj korišćenja – Pregled takmičenja

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjima*.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
2. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
3. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
4. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)



8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal* **pronadjiTakmičenja**(Takmičenje*,List<Takmičenje>*)
2. *signal* **ucitajTakmičenje**(*Takmičenje)*

## SK 10: Slučaj korišćenja – Unos rezultata takmičenja

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*. Učitana je lista zemalja. Učitana je lista takmičara i timova.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
2. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
3. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
4. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)
5. **Korisnik** poziva **sistem** da zapamti izmene podataka o *takmičenju* .(APSO)
6. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspesno zapamtio *takmičenje*.“ (IA)



**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)



8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. Prekida se izvršavanje scenarija. (IA)



12.1. Ukoliko **sistem** ne može da izmeni podatke o *takmičenju*  on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *takmičenje*”. (IA)



Sa navedenih sekvencnih dijagrama uočavaju se sistemske operacije koje treba projektovati:

1. *signal***vratiListuTakmicarskihStaza**(TakmicarskaStaza*,List<TakmicarskaStaza>*)
2. *signal* **vratiListuTakmicara**(Takmicar*,List<Takmicar>*)
3. *signal* **pronadjiTakmičenja**(Takmičenje*,List<Takmičenje>*)
4. *signal* **ucitajTakmičenje**(*Takmičenje)*
5. signal **zapamtiTakmičenje**(Takmičenje)

Kao rezultat analize scenarija dobijeno je ukupno 13 sistemskih operacija koje treba projektovati:

1. *signal* **vratiListuZemalja**(Zemlja*,List<Zemlja>*)
2. *signal* **vratiListuTakmicarskihStaza**(TakmicarskaStaza*,List<TakmicarskaStaza>*)
3. *signal* **pronadjiTakmicare**(Takmicar*,List<Takmicar>*)
4. *signal* **ucitajTakmicara**(*Takmicar)*
5. signal **kreirajTakmicara**(Takmicar)
6. signal **obrisiTakmicara**(Takmicar)
7. signal **zapamtiTakmicara**(Takmicar)
8. *signal* **vratiListuTakmicara**(Takmicar*,List<Takmicar>*)
9. *signal* **pronadjiTakmičenja**(Takmičenje*,List<Takmičenje>*)
10. *signal* **ucitajTakmičenje**(*Takmičenje)*
11. signal **zapamtiTakmičenje**(Takmičenje)
12. signal **obrisiTakmičenje**(Takmičenje)
13. signal **kreirajTakmičenje**(Takmičenje)

## Ponašanje softverskog sistema – Definisanje ugovora o sistemskim operacijama

**Ugovor UG1: Kreiraj Takmičara**

Operacija: kreirajTakmicara(*Takmicar*):signal;

Veza sa SK: SK1

Preduslovi: *Prosto* *vrednosno ograničenje nad objektom Takmicar mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Takmicar mora biti zadovoljeno.*

Postuslovi: *Kreiran je novi takmicar.*



**Ugovor UG2: Zapamti Takmicara**

Operacija: zapamtiTakmicara(*Takmicar*):signal;

Veza sa SK: SK1

Preduslovi: *Prosto vrednosno ograničenje nad objektom Takmicar mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Takmicar mora biti zadovoljeno.*

Postuslovi: *Podaci o takmicaru su zapamćeni.*



**Ugovor UG3: Vrati Listu Zemalja**

Operacija: vratiListuZemalja(*List<Zemlja>*):signal;

Veza sa SK: SK1, SK2, SK4, SK6, SK7

Preduslovi: -

Postuslovi: -



**Ugovor UG4: Vrati Listu Takmicarskih staza**

Operacija: vratiListuTakmicarskihStaza(*List<TakmicarskaStaza>*):signal;

Veza sa SK: SK2, SK4, SK6, SK7

Preduslovi: -

Postuslovi: -



**Ugovor UG5: Pronadji Takmicare**

Operacija: PronadjiTakmicare(*Takmicar*):signal;

Veza sa SK: SK3, SK4, SK5

Preduslovi: *-*

Postuslovi: *-*



**Ugovor UG6: Obrisi Takmicara**

Operacija: ObrisiTakmicara (Takmicar):signal;

Veza sa SK: SK5

Preduslovi: *Strukturno ograničenje nad objektom Takmicar mora biti zadovoljeno.*

Postuslovi: *Takmicar je obrisan.*



**Ugovor UG7: Ucitaj Takmicara**

Operacija: UcitajTakmicara (Takmicar):signal;

Veza sa SK: SK3,SK4,SK5

Preduslovi:

Postuslovi:



**Ugovor UG8: Vrati Listu Takmicara**

Operacija: vratiListuTakmicara(*List<Takmicar>*):signal;

Veza sa SK: SK6, SK7

Preduslovi: -

Postuslovi: -



**Ugovor UG9: Kreiraj Takmičenje**

Operacija: kreirajTakmičenje(*Takmičenje*):signal;

Veza sa SK: SK6

Preduslovi: *Prosto* *vrednosno ograničenje nad objektom Takmičenje mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Takmičenje mora biti zadovoljeno.*

Postuslovi: *Kreirana je nova takmičenje.*



**Ugovor UG10: Zapamti Takmičenje**

Operacija: zapamtiTakmičenje(*takmičenje*):signal;

Veza sa SK: SK6,SK7,SK10

Preduslovi: *Prosto vrednosno ograničenje nad objektom Takmičenje mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Takmičenje mora biti zadovoljeno.*

Postuslovi: *Podaci o takmičenju su zapamćeni.*



**Ugovor UG11: Pronadji Takmičenja**

Operacija: PronadjiTakmičenja(*Takmičenje, List<Takmičenje>*):signal;

Veza sa SK: SK7, SK8, SK9,SK10

Preduslovi: *-*

Postuslovi: *-*



**Ugovor UG12: Obrisi Takmičenje**

Operacija: ObrisiTakmičenje (Takmičenje):signal;

Veza sa SK: SK8

Preduslovi: *Strukturno ograničenje nad objektom Takmičenje mora biti zadovoljeno.*

Postuslovi: *Takmičenje je obrisana.*



**Ugovor UG13: Ucitaj Takmičenje**

Operacija: UcitajTakmičenje (Takmičenje):signal;

Veza sa SK: SK7,SK8,SK9,SK10

Preduslovi:

Postuslovi:



## Struktura softverskog sistema – Konceptualni (domenski) model



Struktura softverskog sistema

## Relacioni model

Delegat(DelegatID, Ime, Prezime, Username, Password)

Zemlja(ZemljaID, Naziv)

TakmicarskaStaza(StazaID, Naziv, Lokacija, Tip, *ZemljaID*)

Takmicar(TakmicarID, Ime, Prezime, DatumRodjenja, *ZemljaID*)

Takmičenje(TakmičenjeID, Naziv, Datum, Kategorija, *StazaID, DelegatID*)

SpisakTakmicara( TakmičenjeID, RedniBroj*,* Ulov, Rang, *TakmicarID*)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela Zemlja** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT  /  UPDATE  CASCADES  Tim, Takmicar, Takmičenje  DELETE  RESTRICTED  Tim, Takmicar, Takmičenje |
| ZemljaID | Integer | not null and > 0 |  |  |
| Naziv | String | not null |  |  |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela TakmicarskaStaza** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT  RESTRICTED Zemlja  UPDATE  RESTRICTE Zemlja  CASCADES  Takmicenje  DELETE RESTRICTED  Takmicenje |
| StazaID | Integer | not null and >0 |  |  |
| Naziv | String | not null |  |  |
| Lokacija | String | not null |  |  |
| Tip | String | not null |  |  |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela Takmicar** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT RESTRICTED Zemlja  UPDATE RESTRICTEDZemlja  CASCADES SpisakTakmicara  DELETE  RESTRICTED SpisakTakmicara |
| TakmicarID | Integer | not null and >0 |  |  |
| Ime | String | not null |  |  |
| Prezime | String | not null |  |  |
| DatumRodjenja | Date |  |  |  |
| ZemljaID | Integer | not null and >0 |  |  |
| TimID | Integer | not null and >0 |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela Takmičenje** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT RESTRICTED TakmicarskaStaza, Delegat  UPDATE RESTRICTED TakmicarskaStaza, Delegat  CASCADES  SpisakTakmicara  DELETE CASCADES SpisakTakmicara |
| TakmičenjeID | Integer | not null and >0 |  |  |
| Naziv | String | not null |  |  |
| Datum | Date | not null |  |  |
| Kategorija | String |  |  |  |
| ZemljaID | Integer | Not null and >0 |  |  |
|  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela SpisakTakmicara** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT RESTRICTED  Takmičenje, Takmicar  UPDATE RESTRICTED  Takmičenje  Takmicar  DELETE / |
| TakmičenjeID | Integer | not null  and >0 |  |  |
| RedniBroj | Integer | not null  and >0 |  |  |
| Ulov | Double | >0 |  |  |
| Rang | Integer |  |  |  |
| TakmicarID | Integer | not null and >0 |  |  |
|  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Tabela Delegat** | | **Prosto vrednosno ograničenje** | | **Složeno vrednosno ograničenje** | | **Strukturno ograničenje** |
| **Atributi** | **Ime** | **Tip atributa** | **Vrednost atributa** | **Međuzav. atributa jedne tabele** | **Međuzav. atributa više tabela** | INSERT /  UPDATE CASCADES  Takmičenje  DELETE RESTRECTED  Takmicenje |
| DelegatID | Integer | not null  and >0 |  |  |
| Ime | String | not null |  |  |
| Prezime | String | not null |  |  |
| Username | String | not null |  |  |
| Password | String | not null |  |  |

­­­

# 3. PROJEKTOVANJE

Faza projektovanja opisuje fizičku strukturu i ponašanje softverskog sistema arhitekturu softverskog sistema). Projektovanje arhitekture softverskog sistema obuhvata projektovanje korisničkog interfejsa, aplikacione logike i skladišta podataka. Projektovanje korisničkog interfejsa obuhvata projektovanje ekranskih formi i kontrolera korisničkog interfejsa. U okviru aplikacione logike se projektuju kontroler aplikacione logike, poslovna logika I broker baze podataka. Projektovanje poslovne logika obuhvata projektovanje logičke structure i ponašanja softverskog sistema.

## 3.1 Arhitektura Softverskog Sistema

U okviru faze projektovanja opisuje se fizička struktura i ponašanje softverskog sistema tj. arhitektura softverskog sistema. U ovom seminarskom radu je korišćena klasična troslojna arhitektura, koja se sastoji od:

* Korisničkog interfejsa
* Aplikacione logike
* Skladišta podataka

Korisnički interfejs

Aplikaciona logika

Skladište podataka

**I sloj**

**II sloj**

**III sloj**

**Softverski sistem**

*Slika: Troslojna arhitektura*

Skladište podataka

**Korisnički interfejs**

Ekranska forma

KontrolerKI

**Aplikaciona logika**

KontrolerAL

Poslovna logika

Database broker

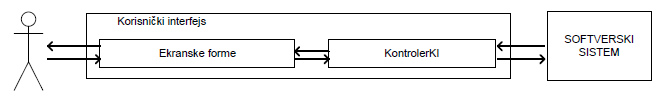
*Slika: Troslojna arhitektura – detaljniji prikaz*

## 3.2. Projektovanje korisničkog interfejsa

Kontroler korisničkog interfejsa i ekranske forme ne predstavljaju softverski sistem, već realizaciju ulaza i izlaza iz softverskog sistema. Kontroler korisničkog interfejsa ima uloga da konvertuje podatke u grafičke elemente ekranskih formi i da podatke ili pošalje u softverski sistem ili da primi podatke iz softverskog sistema i iste konvertuje u elemente ekranske forme.

Korisnički interfejs se sastoji iz:

* Ekranske forme,
* Kontrolera korisničkog interfejsa.



### Projektovanje ekranskih formi

## SK 1: Slučaj korišćenja – Unos novog takmičara

**Naziv SK**

Unos *novog takmičara*

**Aktori SK**

**Korisnik**

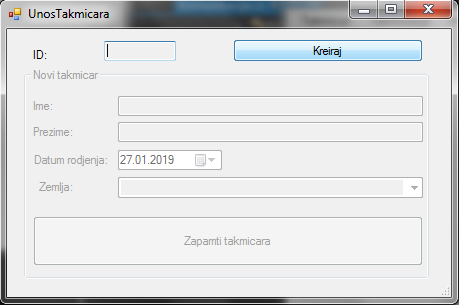
**Učesnici SK**

**Korisnik** i **sistem** (program)

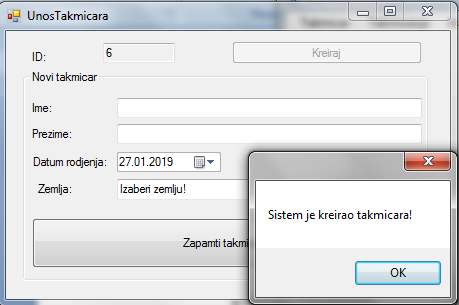
**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos novog takmičara. Učitana je lista svih zemalja.

**Osnovni scenario SK**

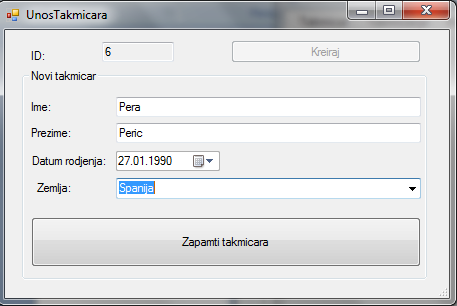
1. **Korisnik** poziva **sistem** da kreira *novog takmičara*. (APSO)



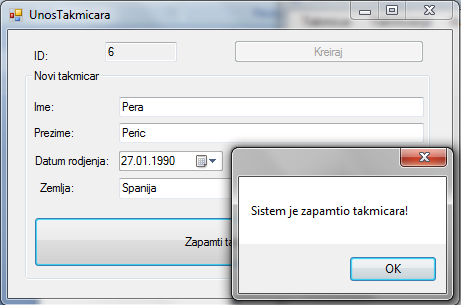
1. **Sistem** kreira *novog takmičara*. (SO)
2. **Sistem** prikazuje **Korisniku** *novog takmičara* i poruku:"**Sistem** je kreirao novog *takmičara*". (IA)



1. **Korisnik** unosi podatke o *novom takmičaru*. (APUSO)

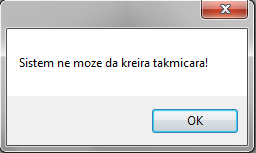


1. **Korisnik** proverava da li je korektno uneo podatke o *novom takmičaru*. (ANSO)
2. **Korisnik** poziva **sistem** da zapamti podatke o *novom takmičaru*. (APSO)
3. **Sistem** pamti podatke o *novom takmičaru.* (SO)
4. **Sistem** prikazuje **Korisniku** poruku: “**Sistem** je uspešno zapamtio *takmičara*.”. (IA)

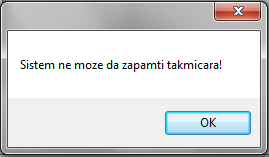


**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da omogući kreiranje novog takmičara on prikazuje **korisniku** poruku: „**Sistem** ne može da kreira novog takmičara“. Prekida se izvršavanje scenarija (IA)



8.1. Ukoliko **sistem** ne može da izvrši unos novog takmičara on prikazuje **korisniku** poruku : „**Sistem** ne može da zapamti novog takmičara*.*“ (IA)



## SK 2: Slučaj korišćenja – Pregled takmičara

**Naziv SK**

Pregled takmičara

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičarima*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednost po kojoj pretražuje *takmičare*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičare* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičare* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** *takmičare* i poruku:"**Sistem** je našao *takmičare* po zadatoj vrednosti". (IA)
5. **Korisnik** bira *takmičara*. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o *takmičaru*. (APSO)
7. **Sistem** učitava podatke o *takmičaru*. (SO)
8. **Sistem** prikazuje **korisniku** *takmičara* i poruku:"**Sistem** je uspešno učitao podatke o *takmičaru*". (IA)

**Alternativna scenarija:**

4.1. Ukoliko **sistem** ne može da nađe *takmičare* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičare*  ”. Prekida se izvršavanje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *takmičara*, on prikazuje **korisniku** poruku:"Sistem ne može da učita *takmičara*". (IA)

## SK 3: Slučaj korišćenja – Izmena takmičara

**Naziv SK**

Izmena takmičara

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos novog takmičara. Učitana je lista svih zemalja.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednost po kojoj pretražuje *takmičare*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičare*  po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičare* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** *takmičare* i poruku:" **Sistem** je uspešno pronašao *takmičare* po zadatim vrednostima ". (IA)
5. **Korisnik** bira *takmičara* kojeg želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičaru*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičaru*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičaru* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičaru*". (IA)
9. **Korisnik** menja podatke o *takmičaru*. (APUSO)
10. **Korisnik** poziva **sistem** da zapamti podatke o *takmičaru.* (APSO)
11. **Sistem** pamti podatke o *takmičaru*. (SO)
12. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspesno izmenio *takmičara*.” (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičara*  on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičara*”. Prekida se izvršavanje scenarija. (IA)

8.1. Ukoliko sistem ne može da nađe takmičara on prikazuje korisniku poruku:"Sistem ne može da nađe izabranog takmičara". Prekida se izvršavanje scenarija. (IA)

12.1. Ukoliko **sistem** ne može da zapamti podatke o *takmičaru* on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *takmičara*”. (IA)

## SK 4: Slučaj korišćenja – Brisanje takmičara

**Naziv SK**

Brisanje takmičara

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik** i **sistem** (program)

**Preduslov: Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičarima*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednost po kojoj pretražuje *takmičare*. (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičare*  po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičare* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** *takmičare* i poruku:" **Sistem** je uspešno pronašao *takmičare* po zadatim vrednostima ". (IA)
5. **Korisnik** bira *takmičara* kojeg želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičaru*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičaru*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičaru* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičaru*". (IA)
9. **Korisnik** poziva **sistem** da obriše podatke o *takmičaru.* (APSO)
10. **Sistem** briše podatke o *takmičaru*. (SO)
11. **Sistem** prikazuje **korisniku** poruku: “**Sistem** je uspesno obrisao *takmičara*.” (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičara*  on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičara*”. Prekida se izvršavanje scenarija. (IA)

8.1. Ukoliko sistem ne može da nađe takmičara on prikazuje korisniku poruku:"Sistem ne može da nađe izabranog takmičara". Prekida se izvršavanje scenarija. (IA)

11.1. Ukoliko **sistem** ne može da obriše podatke o *takmičaru* on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše *takmičara*”. (IA)

## SK 5: Slučaj korišćenja – Unos takmičenja

**Naziv SK**

Unos takmičenja

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik** i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za unos takmičenja. Učitana je lista takmičarskih staza i takmičara.

**Osnovni scenario SK**

1. **Korisnik** poziva **sistem** da kreira novo *takmičenje*. (APSO)
2. **Sistem** kreira novo *takmičenje*. (SO)
3. **Sistem** prikazuje **korisniku** novo *takmičenje* i poruku:"**Sistem** je uspešno kreirao novo *takmičenje*". (IA)
4. **Korisnik** unosi podatke o novoj *takmičenju.* (APUSO)
5. **Korisnik** poziva **sistem** da zapamti podatke o novoj *takmičenju*. (APSO)
6. **Sistem** pamti podatke o novoj *takmičenju*. (SO)
7. **Sistem** prikazuje korisniku poruku: „Sistem je uspesno zapamtio novo takmičenje“. (IA)

**Alternativna scenarija**

3.1. Ukoliko **sistem** ne može da kreira *novo takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da kreira *novo takmičenje* ”. Prekida se izvršenje scenarija. (IA)

7.1.Ukoliko **sistem** ne možeda zapamti podatke o *novoj takmičenju*, on prikazuje **korisnuku** poruku: “Sistem ne može da zapamti novo *takmičenje*.” (IA)

## SK 6: Slučaj korišćenja – Izmena takmičenja

**Naziv SK**

Izmena *takmičenja*

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik**  i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*. Učitana je lista takmičarskih staza i takmičara.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *takmičenja.* (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičenja* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
5. **Korisnik** bira *takmičenje* koje želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičenju*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)
9. **Korisnik** unosi (menja) podatke o *takmičenju*. (APUSO)
10. **Korisnik** poziva **sistem** da zapamti izmene podataka o *takmičenju* .(APSO)
11. **Sistem** pamti izmene o *takmičenju*. (SO)
12. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspesno izmenio *takmičenje*.“ (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. Prekida se izvršavanje scenarija. (IA)

12.1. Ukoliko **sistem** ne može da izmeni podatke o *takmičenju*  on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *takmičenje*”. (IA)

## SK 7: Slučaj korišćenja – Brisanje takmičenja

**Naziv SK**

Brisanje *takmičenja*

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik**  i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *takmičenja.* (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičenja* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
5. **Korisnik** bira *takmičenje* koje želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičenju*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)
9. **Korisnik** poziva **sistem** da obriše podatake o *takmičenju* .(APSO)
10. **Sistem** briše izmene o *takmičenju*. (SO)
11. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspesno obrisao *takmičenje*.“ (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. Prekida se izvršavanje scenarija. (IA)

11.1. Ukoliko **sistem** ne može da obriše podatke o *takmičenju*  on prikazuje **korisniku** poruku: “**Sistem** ne može da obriše *takmičenje*”. (IA)

## SK 8: Slučaj korišćenja – Pregled takmičenje

**Naziv SK**

Pregled *takmičenje*

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik**  i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *takmičenja.* (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičenja* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
5. **Korisnik** bira *takmičenje* koje želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičenju*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)

8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. (IA)

## SK 9: Slučaj korišćenja – Unos rezultata takmičenja

**Naziv SK**

Unos rezultata *takmičenja*

**Aktori SK**

**Korisnik**

**Učesnici SK**

**Korisnik**  i **sistem** (program)

**Preduslov:Sistem** je uključen. **Sistem** prikazuje formu za rad sa *takmičenjema*.

**Osnovni scenario SK**

1. **Korisnik** unosi vrednosti po kojima pretražuje *takmičenja.* (APUSO)
2. **Korisnik** poziva **sistem** da nađe *takmičenja* po zadatoj vrednosti. (APSO)
3. **Sistem** traži *takmičenja* po zadatoj vrednosti. (SO)
4. **Sistem** prikazuje **korisniku** nađene *takmičenja* i poruku:"**Sistem** je nasao *takmičenja* po zadatoj vrednosti". (IA)
5. **Korisnik** bira *takmičenje* koje želi da izmeni. (APUSO)
6. **Korisnik** poziva **sistem** da učita podatke o izabranom *takmičenju*. (APSO)
7. **Sistem** pronalazi podatke o izabranom *takmičenju*. (SO)
8. **Sistem** prikazuje **korisniku** podatke o izabranom *takmičenju* i poruku:"**Sistem** uspešno pronašao podatke o izabranom *takmičenju*". (IA)
9. **Korisnik** unosi (menja) podatke o *takmičenju*. (APUSO)
10. **Korisnik** poziva **sistem** da zapamti izmene podataka o *takmičenju* .(APSO)
11. **Sistem** pamti izmene o *takmičenju*. (SO)
12. **Sistem** prikazuje **korisniku** poruku: „**Sistem** je uspesno izmenio *takmičenje*.“ (IA)

**Alternativna scenarija**

4.1. Ukoliko **sistem** ne može da nađe *takmičenja* on prikazuje **korisniku** poruku: “**Sistem** ne može da nađe *takmičenja* po zadatoj vrednosti”. Prekida se izvršenje scenarija. (IA)

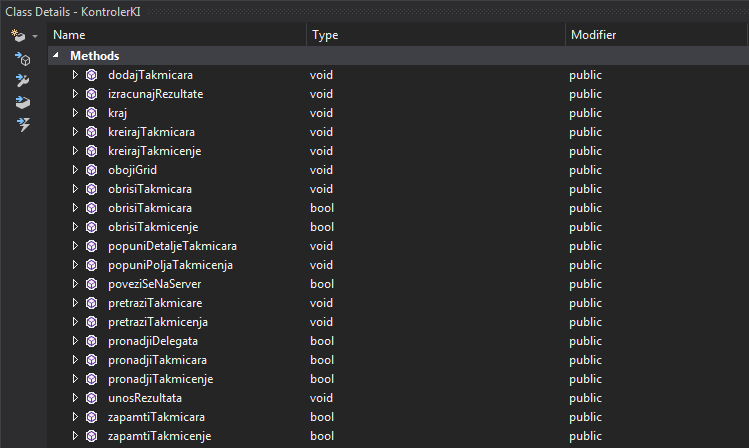
8.1. Ukoliko **sistem** ne može da nađe *takmičenje* on prikazuje **korisniku** poruku: “**Sistem** ne može da učita podatke o izabranom *takmičenju*”. Prekida se izvršavanje scenarija. (IA)

12.1. Ukoliko **sistem** ne može da izmeni podatke o *takmičenju*  on prikazuje **korisniku** poruku: “**Sistem** ne može da zapamti *takmičenje*”. (IA)

### 3.2.3. Projektovanje kontrolera korisničkog interfejsa

Aplikaciona logika služi za opisivanje strukture i ponašanja softverskog sistema i projektuje se nezavisno od korisničkog interfejsa i obrnuto. Drugim rečima, aplikaciona logika (koja predstavlja *Model* u MVC paternu) nema znanja o tome gde se nalazi korisnički interfejs (koja predstavlja *View* u MVC paternu).

Kontroler je odgovoran da prihvati zahtev za izvršenje sistemske operacije od klijenta i da ga prosledi do poslovne logike koja je odgovorna za izvršenje sistemske operacije.





Slika KKI Korisnički interfejs u kontekstu arhitekture softverskog sistema

## 3.3. Projektovanje aplikacione logike

### 3.3.1. Kontroler aplikacione logike

Kontroler aplikacione logike treba da podigne serverski soket koji će da osluškuje mrežu. Kada klijent (klijentski soket) uspostavi konekciju sa kontrolerom (serverskim soketom), tada kontroler treba da generiše nit koja će uspostaviti dvosmernu vezu sa klijentom (ulaznu i izlaznu). Slanje i primanje podataka od klijenta se ostvaruje preko soketa.

Klijent šalje zahtev za izvršenje neke od SO do odgovarajuće niti (koju smo nazvali “Obrada”), koja je povezana sa tim klijentom. “Obrada” prima zahtev i dalje ga preusmerava do klasa koje su odgovorne za izvršenje SO. Nakon izvršenja SO rezultat se vraća do aplikacione logike, odnosno do “Obrade”, koja taj rezultat šalje nazad do klijenta.

**Primer specificnog kontrolera aplikacione logike**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Net;

using System.Runtime.Serialization.Formatters.Binary;

using System.Net.Sockets;

using Biblioteka;

namespace Komunikacija

{

public class Komunikacija

{

TcpClient klijent;

BinaryFormatter formater;

NetworkStream tok;

public bool poveziSeNaServer()

{

try

{

klijent = new TcpClient("127.0.0.1", 20000);

tok = klijent.GetStream();

formater = new BinaryFormatter();

return true;

}

catch (Exception)

{

return false;

}

}

public void kraj()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.Kraj;

formater.Serialize(tok, transfer);

}

public Object pronadjiDelegata(Delegat d)

{

// slanje

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.PronadjiDelegata;

transfer.TransferObjekat = d;

formater.Serialize(tok, transfer);

// prijem

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object kreirajTakmicara()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.KreirajTakmicara;

transfer.TransferObjekat = new Takmicar();

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object zapamtiTakmicara(Takmicar t)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.ZapamtiTakmicara;

transfer.TransferObjekat = t;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object obrisiTakmicara(Takmicar t)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.ObrisiTakmicara;

transfer.TransferObjekat = t;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object pronadjiTakmicara(Takmicar t)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.PronadjiTakmicara;

transfer.TransferObjekat = t;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object pretraziTakmicare(Takmicar t)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.PretraziTakmicare;

transfer.TransferObjekat = t;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object vratiSveZemlje()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.VratiSveZemlje;

transfer.TransferObjekat = new Zemlja();

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object vratiSveTakmicare()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.VratiSveTakmicare;

transfer.TransferObjekat = new Takmicar();

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object vratiSveStaze()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.VratiSveStaze;

transfer.TransferObjekat = new TakmicarskaStaza();

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object kreirajTakmicenje()

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.KreirajTakmicenje;

transfer.TransferObjekat = new Takmicenje();

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object zapamtiTakmicenje(Takmicenje t)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.ZapamtiTakmicenje;

transfer.TransferObjekat = t;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object obrisiTakmicenje(Takmicenje t)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.ObrisiTakmicenje;

transfer.TransferObjekat = t;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object pretraziTakmicenja(Takmicenje t)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.PretraziTakmicenja;

transfer.TransferObjekat = t;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

public Object pronadjiTakmicenje(Takmicenje t)

{

TransferKlasa transfer = new TransferKlasa();

transfer.Operacija = Operacije.PronadjiTakmicenje;

transfer.TransferObjekat = t;

formater.Serialize(tok, transfer);

transfer = formater.Deserialize(tok) as TransferKlasa;

return transfer.Rezultat;

}

}

}

### 3.3.2. Sistemske operacije

U prethodnoj fazi, fazi analize, odredili smo ugovore o sistemskim operacijama, pri čemu smo rekli da jedan ugovor opisuje ponašanje jedne sistemske operacije, tako što opisuje ŠTA operacija treba da radi, ali ne i kako. Sada se u fazi projektovanja za svaki od ugovora projektuje konceptualno rešenje (realizacija) SO. To znači da ćemo za svaku klasu odgovornu za izvršenje SO definisati KAKO će se sistemska operacija izvršiti.

Na samom početku projektovanja SO izbeći ćemo aspekte realizacije koji su vezani za konekciju sa bazom, perzistentnost i transakcije.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using Biblioteka;

using Sesija;

namespace SistemskeOperacije

{

public abstract class OpstaSO

{

public Object izvrsiSO(OpstiDomenskiObjekat odo)

{

Object rezultat = null;

Broker.dajSesiju().otvoriKonekciju();

Broker.dajSesiju().zapocniTransakciju();

try

{

rezultat = Izvrsi(odo);

Broker.dajSesiju().potvrdiTransakciju();

}

catch (Exception)

{

Broker.dajSesiju().ponistiTransakciju();

}

finally

{

Broker.dajSesiju().zatvoriKonekciju();

}

return rezultat;

}

public abstract object Izvrsi(OpstiDomenskiObjekat odo);

}

}

**Ugovor UG1: Kreiraj Takmičara**

Operacija: kreirajTakmicara(*Takmicar*):signal;

Veza sa SK: SK1

Preduslovi: *Prosto* *vrednosno ograničenje nad objektom Takmicar mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Takmicar mora biti zadovoljeno.*

Postuslovi: *Kreiran je novi takmicar.*

**Ugovor UG2: Zapamti Takmicara**

Operacija: zapamtiTakmicara(*Takmicar*):signal;

Veza sa SK: SK1

Preduslovi: *Prosto vrednosno ograničenje nad objektom Takmicar mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Takmicar mora biti zadovoljeno.*

Postuslovi: *Podaci o takmicaru su zapamćeni.*

**Ugovor UG3: Vrati Listu Zemalja**

Operacija: vratiListuZemalja(*List<Zemlja>*):signal;

Veza sa SK: SK1, SK2, SK4, SK6, SK7

Preduslovi: -

Postuslovi: -

**Ugovor UG4: Vrati Listu Takmicarskih staza**

Operacija: vratiListuTakmicarskihStaza(*List<TakmicarskaStaza>*):signal;

Veza sa SK: SK2, SK4, SK6, SK7

Preduslovi: -

Postuslovi: -

**Ugovor UG5: Pronadji Takmicara**

Operacija: PronadjiTakmicara(*Takmicar*):signal;

Veza sa SK: SK3, SK4, SK5

Preduslovi: *-*

Postuslovi: *-*

**Ugovor UG6: Obrisi Takmicara**

Operacija: ObrisiTakmicara (Takmicar):signal;

Veza sa SK: SK5

Preduslovi: *Strukturno ograničenje nad objektom Takmicar mora biti zadovoljeno.*

Postuslovi: *Takmicar je obrisan.*

**Ugovor UG7: Ucitaj Takmicara**

Operacija: UcitajTakmicara (Takmicar):signal;

Veza sa SK: SK3,SK4,SK5

Preduslovi:

Postuslovi:

**Ugovor UG8: Vrati Listu Takmicara**

Operacija: vratiListuTakmicara(*List<Takmicar>*):signal;

Veza sa SK: SK6, SK7

Preduslovi: -

Postuslovi: -

**Ugovor UG9: Kreiraj Takmičenje**

Operacija: kreirajTakmičenje(*Takmičenje*):signal;

Veza sa SK: SK6

Preduslovi: *Prosto* *vrednosno ograničenje nad objektom Takmičenje mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Takmičenje mora biti zadovoljeno.*

Postuslovi: *Kreirana je nova takmičenje.*

**Ugovor UG10: Zapamti Takmičenje**

Operacija: zapamtiTakmičenje(*takmičenje*):signal;

Veza sa SK: SK6,SK7,SK10

Preduslovi: *Prosto vrednosno ograničenje nad objektom Takmičenje mora biti zadovoljeno.*

*Strukturno ograničenje nad objektom Takmičenje mora biti zadovoljeno.*

Postuslovi: *Podaci o takmičenju su zapamćeni.*

**Ugovor UG11: Pronadji Takmičenja**

Operacija: PronadjiTakmičenja(*Takmičenje, List<Takmičenje>*):signal;

Veza sa SK: SK7, SK8, SK9,SK10

Preduslovi: *-*

Postuslovi: *-*

**Ugovor UG12: Obrisi Takmičenje**

Operacija: ObrisiTakmičenje (Takmičenje):signal;

Veza sa SK: SK8

Preduslovi: *Strukturno ograničenje nad objektom Takmičenje mora biti zadovoljeno.*

Postuslovi: *Takmičenje je obrisana.*

**Ugovor UG13: Ucitaj Takmičenje**

Operacija: UcitajTakmičenje (Takmičenje):signal;

Veza sa SK: SK7,SK8,SK9,SK10

Preduslovi:

Postuslovi:



Slika OSO: Klase koje su odgovorne za SO nasleđuju klasu OpstaSO

### 3.3.3. Domenske klase

Na osnovo konceptualnih klasa prave se softverske klase strukture. Svaka klasa sadrži privatna polja atributa, getter-e, setter-e za iste, konstruktore (bilo bezparametrizovani bilo parametrizovani).

Domenske klase implementiraju interfejs *BazniObjekat* kako bi omogućili lakše implementiranje metoda *DatabaseBroker*-a. Na taj način *DatabaseBroker* prima interfejs umesto samih klasa i olakšava kreiranje generičkih upita.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data;

namespace BibliotekaKlasa

{

public interface OpstiDomenskiObjekat

{

string ime

{

get;

}

string kljuc

{

get;

}

string uslovJedan

{

get;

}

string uslovVise

{

get;

}

string vrednostiZaAzuriranje

{

get;

}

string vrednostiZaUbacivanje

{

get;

}

OpstiDomenskiObjekat napuni(DataRow red);

}

}

### 3.3.4 Broker baze podataka

Broker baze podataka je softverska klasa odgovorna za komunikacju između poslovne logike i skladišta podataka. Drugim rečima, projektuje se kako bi obezbedio perzistentni servis objektima domenskih klasa koji se čuvaju u bazi podataka. Klasa *DatabaseBroker* predstavlja perzistentni okvir koji posreduje u svim operacijama nad bazom podataka i realizuje sledeće metode:

public void OtvoriKonekciju();

public void ZatvoriKonekciju();

public void ZapocniTransakciju();

public void PonistiTransakciju();

public void PotvrdiTransakciju();

public List<OpstiDomenskiObjekat> dajSve(OpstiDomenskiObjekat odo);

public List<OpstiDomenskiObjekat> dajSveZaUslov(OpstiDomenskiObjekat odo);

public List<OpstiDomenskiObjekat> dajSveZaUslovVise(OpstiDomenskiObjekat odo);

public OpstiDomenskiObjekat dajZaUslov (OpstiDomenskiObjekat odo);

public OpstiDomenskiObjekat dajZaUslovVise(OpstiDomenskiObjekat odo);

public int azuriraj(OpstiDomenskiObjekat odo);

public int ubaci(OpstiDomenskiObjekat odo);

public int obrisi(OpstiDomenskiObjekat odo);

public int obrisiSveZaUslov(OpstiDomenskiObjekat odo);

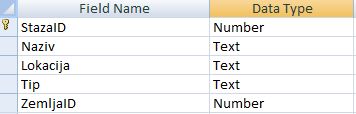
public int dajSifru(OpstiDomenskiObjekat odo);



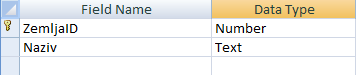
Slika DBBR: Database broker klasa se povezuje sa klasom OpstiDomenskiObjekat

### 3.3.5. Projektovanje skladišta podataka

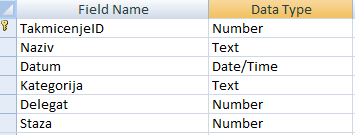
Na osnovo strukture softverskih klasa projektovane su tabele (skladišta podataka) relacionog sistema za upravljanje bazom podataka. U ovom radu je korišćen MS Access.

****

**Takmicarska Staza**

****

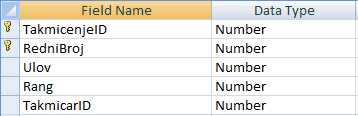
**Zemlja**

****

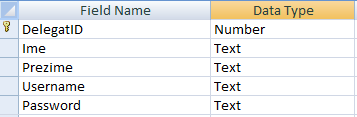
**Takmičenje**

****

**Takmičar**

****

**Spisak takmičara**

**  
Delegat**

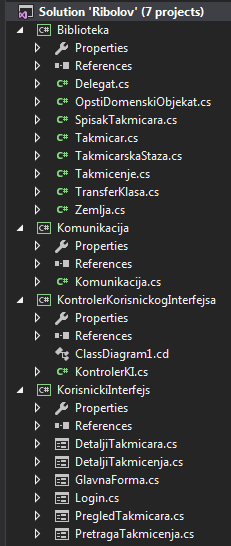
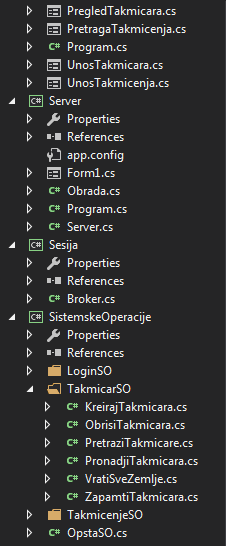


Slika ASSTBP: Arhitektura soft. sistema nakon projektovanja tabela baze podataka

# 4. IMPLEMENTACIJA

Softverski sistem, rezultat ovog rada, razvijen je u programskom jeziku C#, a izvorni kod je priložen na pratećem CD-u. Sistem je projektovan kao klijent-server. Kao razvojno okruženje korišćen je Microsoft Visual Studio 2017. Kao sistem za upravljanjem bazom podataka korišćen je MS Access.

Na osnovo arhitekture softverskog sistema dobijene su sledeće softverske klase:

# 

# 5. TESTIRANJE

Svaki od implementiranih slučajeva korišćenja je testiran. Prilikom testiranja svakog slučaja korišćenja pored unetih pravilnih podataka unošeni su i nepravilni podaci da bi se utvrdilo kakav će rezultat izvršenja biti. Na osnovo izvršenih testiranja otklonjeni su uočeni nedostaci.

# 6. LITERATURA

Projektovanje softvera – Skripta, dr Siniša Vlajić, Beograd - 2016.