FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY
UNIVERZITA KOMENSKÉHO

Záverečná správa

CupCalculator

zimný semester 2015/2016
Dávid Jámbor
Dávid Koszeghy
Jakub Bičian

Obsah:

1 Špecifikácia požiadaviek

1.0.1 Predmet špecifikácie

Táto špecifikácia požiadaviek na softvér (ďalej ŠPS) popisuje používateľské, funkčné a parametrické požiadavky prvej verzie web aplikácie CupCalculator. ŠPS je určená pre tím, ktorý bude výsledný softvér implementovať. Špecifikácia je súčasťou zmluvy medzi zadávateľom projektu a dodávateľom. Bude slúžiť ako východisko pre vyhodnocovanie správnosti softvéru.

1.0.2 Rozsah projektu

Web aplikácia bude slúžiť na generovanie výsledkových listín orietančných behov na základe užívateľom definovaných bodovacích podmienok. Výsledková listina bude generovaná vo vopred dohodnutom formáte a bude spĺnať požiadavky zadávateľa projektu.

1.1 Všeobecný popis

V nasledujúcej kapitole sa nachádzajú bližšie informácie k funkciám a rozhraniam webovej aplikácie CupCalulator. Opisujú rôzne obmedzenia a závislosti, ktoré sú kladené na samotnú aplikáciu.

1.1.1 Perspektíva produktu

Cieľom aplikácie je zjednodušiť zbieranie údajov z orientačných behov organizovaných po celom Slovensku. Výsledky z orientačných behov sú zverejnené na internetových stránkach v XML formáte. Web aplikácia CupCalculator, dané údaje pozbiera a spracuje ich na základe požiadaviek uživateľa.

Výsvetlivky:

CSV - Comma-separated Values
HTML - HyperText Markup Language

1.1.2 Funkcie produktu

Základný cieľ produktu je umožniť usporiadateľovi orientačných behov spracovať výsledky a nastaviť bodové ohodnotenie účastníkov podľa výsledkov jednotlivých etáp v orientačných behoch.

Aplikácia bude vedieť spracovať viacero vstupných dát vo forme definovanej zadávateľom a tieto dáta spracovať a následne uložiť do databázy. Po spracovaní, vyhodnotí dáta a vypluje vysledkovú listinu zo všetkých behov nahratých do aplikácie

Web stránka CupCalculator-a obsahuje grafické rozhranie pre uživateľa a spôsob nahrávanie údajov.
Po uspešnom nahraní údajov, si uživateľ zvolí typ bodovania orientačných behov. výstupný format pre aplikáciu je CSV a HTML dokument. (definované zadávateľom)

1.1.3 Užívateľské parametre a charakteristika

Používateľom webovej aplikácie CupCalulator bude iba vyhodnocovateľ orientačných behov. Tento užívateľ bude mať právo využívať plnú funkcionalitu webovej aplikácie a nebude prichádzať k žiadnym obmedzeniam k prístupu zo strany aplikácie.

1.3 Špecifické požiadavky

1.3.1 Funkcionálne požiadavky

1.3.1.1 Grafické rozhranie

1. jednoduché/minimalistické

1.3.1.2 Nahrávanie údajov

1. cez formulár (upload tlačítko)
2. nahrávanie súboru, cez django formulárd
3. možnosť spracovania viacerých údajov (výsledky z celej sezóny)
4. konfigurácie pretekov (pridávanie/odstránenie)

1.3.1.3 Výber bodovacieho systému

1. Vyhodnocovanie podľa pomeru ubehnutého času víťaza a sútažiaceho a to podľa nasledujúcich možných ohodnotení
    a. (čas víťaza / čas sútažiaceho)*[hodnota definovaná vyhodnocovateľom]
    b. maximum(0, (2 - čas sútažiaceho / čas víťaza))*[hodnota definovaná vyhodnocovateľom]
    c. maximum(0, [hodnota definovaná vyhodnocovateľom] + [hodnota definovaná vyhodnocovateľom]*(priemerný čas sútažiacich - čas sútažiaceho/priemerný čas sútažiacich) kde sa dá nastaviť percentuálny počet sútažiacich určených na výpočet priemerneho času sútažiacich
2. Pevne stanovený interval bodového ohodnotenia, kde bude možné vyrátať body pre výhercu behu buť bodla počtu účastníkov behu alebo podľa počtu zúčastnených tímov * maximálny počet sútažiacých za tím.
    a. Bodový skok medzi jednotlivými poradiami: [hodnota definovaná vyhodnocovateľom]
    b. Počet bodov pridelených poslednému sútažiacemu: [hodnota definovaná vyhodnocovateľom]
    c. Maximálny počet sútažiacich v tímoch: [hodnota definovaná vyhodnocovateľom]
    d. Možnosť zvoliť zarátavanie bodov diskvalifikovaných sútažiacich ostatným účastnikom behu

3. Bodová tabuľka, v ktorej je možnosť napevno zadať bodove ohodnotenie jednotlivých poradí a to tak že 1. miesto dostane počet bodov uvedených v 1. riadku tabuľky. V prípade, že je viacej sútažiacich než je bodových hodnot v tabuľke bude sa opakovať posledné bodové ohodnotenie v tabuľke
4. Časové bodovanie, je precentualne ohodnotenie získaneho času a to tak, že čas vítaza určuje 100% času. V tabuľke percentualneho ohodnotenia je možné pridávať jednotlivé percentá času a ich bodové ohodnotenie.

1.3.1.4 Obodovanie súťažiacich

1. Aplikácia ohodnoti nahrané výsledky z xml súborov podľa bodového nastavenia vyhodnocovateľom
2. Následne vytvorí výsledkovú listinu sútažiacich, ktorá bude rozdelená na dve časti:
    a. Celková výsledková listina
    b. Výsledkova listina jednotlivých kategorií.

1.3.1.5. Spracovanie dokumentov

1. parsovanie XML dokumentov podla špecifikácie IOF XML format (http://orienteering.org/resources/it/data-standard-3-0/)
2. generovanie údajov
3. bežci budú mať pridelené unikátné identifické čísla v databáze
4. v prípade nejasnosti pri priarodvaní výsledkového času súťažiacemu, bude ponuknutá možnosť výsledkový čas priradiť manuálne

1.3.1.6. Export výstupných dát

1. Bude umožnený export na stiahnutie výsledkovej listiny vo forme .csv a.html

1.3.2 Požiadavky na programátorov

1.3.2.1 Vytvoriť čistý a prehľadný kód

1. uľahčenie práce
2. prehľadnosť kódu

1.3.2.2 Písanie dokumentácie

1. na základe softverových knižníc

1.3.2.3 Dodržanie dohodnutých termínov

1. nevyvárať zbytočný stres na zadávateľa a programátorov

1.3.3 Požiadavky na zadávateľa

1.3.3.1 Server pre aplikáciu

1. rozhranie, na ktorom bude aplikácia bežať
2. konto, pod ktorým bude umožnené aplikáciu vyvíjať

1.3.3.2 Dokumenty

1. testovacie údaje z orientačných behov

1.4 Obmedzenia aplikácie

Aplikácia bude spustiteľná len na serveroch s podporov mod_wsgi.

Pomalšie uživateľské rozhranie, bude zapričinené, že GUI aplikácie beži v prehliadači.

Spracovanie dát, môže trvať dlhšiu dobu, záleží podľa veľkosťi vstupných dát a výkonnosti hardvéru, na ktorom aplikácia bude prevádzkovaná.

2 Konceptuálna analýza

2.1 Užívateľské rozhranie

V nasledovnej kapitole bude popísaný prvotný návrh užívateľského rozhrania. Prvý navrhnutý koncept je na obr.2

2.2.1 Rozloženie nástrojov

Program pracuje ako webová aplikácia. Grafické rozhranie pozostáva z dvoch blokov:
nahrávanie súborov, ovládací panel. Grafické spracovanie cez Javascript knižnicu JQuery. Využitie v animáciach elementov na stránke. Cieľom je dynamické GUI.

2.2.2 Nahrávanie súborov

Slúži na načítavanie údajov, umožňuje pridelovanie sezóny.

2.2.2.1 Úprava vstupných súborov

Pridávanie a mazanie údajov. Dodatočné úpravy XML vstupov. V prípade nejasností, alebo zle uvedených údajov vo vstupe. Užívateľovi ponúkneme manuálnu úpravu vstupov, cez admin zónu.

2.2.3 Hlavné menu

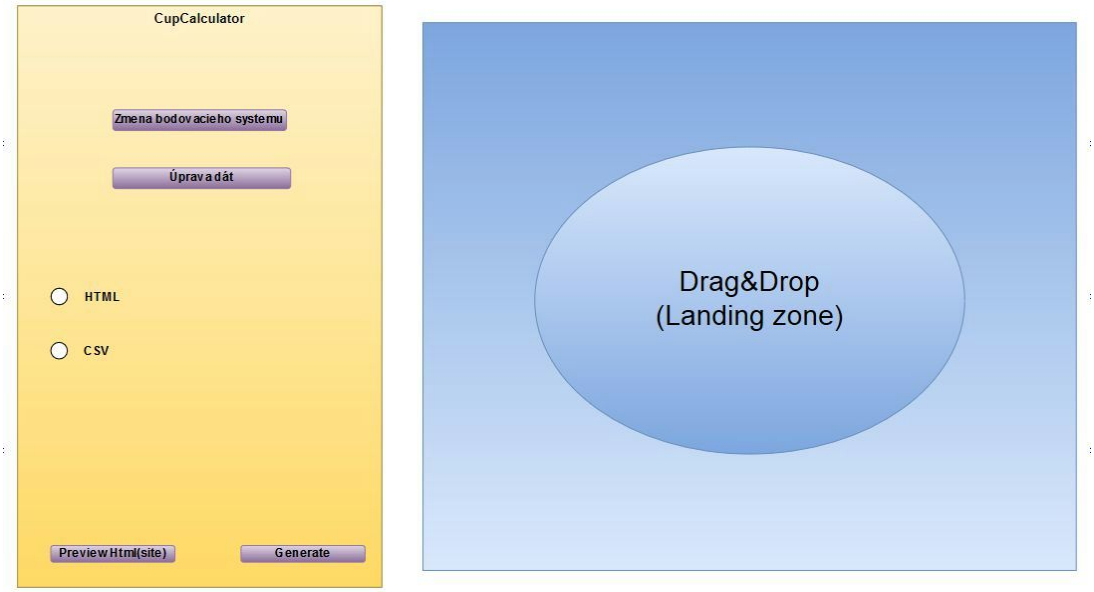
Umiestnenie: v ľavej sekcií stránky
Obsahuje tlačidlá s funkcionalitou pre spracovanie dát. Preteky sa dajú bodovať rôznymi spôsobmi. Úprava bodového systému bude možná cez hlavné menu, tak ako aj volenie výstupného formátu (výber z dvoch formátov HTML, CSV).

2.2.3.1 Generovanie výstupov

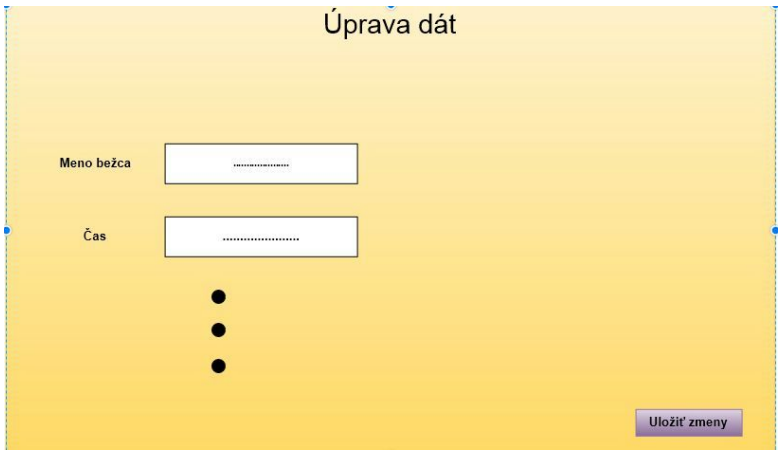
CSV aj HTML formát bude priamo stiahnuteľný. Pre HTML formát podpora nahliadnutia dokumentu. Ukážku dokumentu zobrazíme v prehliadači.

2.2.3.2 Výber bodovacieho systému

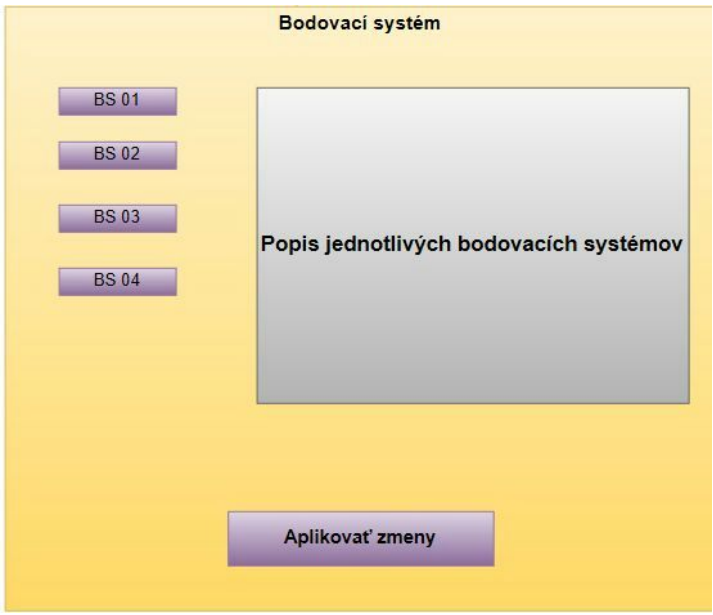
1. Vyhodnocovanie podľa pomeru ubehnutého času víťaza a súťažiaceho a to podľa nasledujúcich možných ohodnotení
    a. (čas víťaza / čas súťažiaceho)*[hodnota definovaná vyhodnocovateľom]
    b. maximum(0, (2 - čas súťažiaceho/ čas víťaza))*[hodnota definovaná vyhodnocovateľom]
    c. maximum(0, [hodnota definovaná vyhodnocovateľom] + [hodnota definovaná vyhodnocovateľom]*(priemerný čas súťažiacich - čas súťažiaceho/priemerný čas súťažiacich) kde sa dá nastaviť percentuálny počet súťažiacich určených na výpočet priemerného času súťažiacich
2. Pevne stanovený interval bodového ohodnotenia, kde bude možné vyrátať body pre výhercu behu buď bodla počtu účastníkov behu alebo podľa počtu zúčastnených tímov * maximálny počet súťažiacich za tím.
    a. Bodový skok medzi jednotlivými poradiami: [hodnota definovaná vyhodnocovateľom]
    b. Počet bodov pridelených poslednému súťažiacemu: [hodnota definovaná vyhodnocovateľom]
    c. Maximálny počet súťažiacich v tímoch: [hodnota definovaná vyhodnocovateľom]
    d. Možnosť zvoliť zarátavanie bodov diskvalifikovaných súťažiacich ostatným účastníkom behu
3. Bodová tabuľka, v ktorej je možnosť napevno zadať bodové ohodnotenie jednotlivých poradí a to tak že 1. miesto dostane počet bodov uvedených v 1. riadku tabuľky. V prípade, že je viacej súťažiacich než je bodových hodnôt v tabuľke bude sa opakovať posledné bodové ohodnotenie v tabuľke
4. Časové bodovanie, je precentuálne ohodnotenie získaného času a to tak, že čas víťaza určuje 100% času. V tabuľke percentuálneho ohodnotenia je možné pridávať jednotlivé percentá času a ich bodové ohod



obr.1: Prvý náčrt uživateľského rozhrania



obr.2: Koncept spracovania dát



obr.3: Funkcionalita bodovacieho systému

2.2 Užívatelia systému

2.2.1 Vyhodnocovateľ

Vyhodnocovateľ je jediná osoba, ktorá má prístup k aplikácii. Má plne právo aplikáciu používať. Vie nahrávať vstupné .xml súbory, nastavovať bodové ohodnotenie, riešiť kolízne situácie a exportovať výsledkové listiny.

2.3 Dáta

2.3.1 Vstupné dáta

Vstupné dáta sú všetky údaje, ktoré do aplikácie zadáva vyhodnocovateľ v súbore .xml v runs sekcií stránky. Medzi tieto údaje patrí aj úprava a mazanie bežcov v .xml.

2.3.2 Výstupné dáta

Výstupom celej aplikácie je samotné hodnotenie vo formáte HTML alebo CSV (na základe výberu vyhodnocovateľa), ktoré je potom možné zdieľať.

2.3.3 Vnútorné dáta

Pod vnútorné dáta patria všetky dočasné dáta, ktoré vznikajú a zanikajú počas behu aplikácie. Ide predovšetkým o uchovávanie si dočasných informácii potrebných na beh programu, a databázy obsahujúcej bežcov.

2.4 Procesy

2.4.1 Načítanie údajov

V rámci tohto procesu sa do aplikácie nahrajú .xml súbory. Aplikácia spracuje údaje podľa zvolených kritérií v bodovacom systéme.

2.4.2 Hodnotenie výsledkov

Údaje spracuje pythonovský skript, využijeme vstavané pythonovské knižnice. Výsledok hodnotenia uložíme do databázy aplikácie, následne daný údaj bude možné generovať v CSV a HTML formáte.

2.4.2.1 Nastavenia bodového ohodnotenia

Obsahuje výber bodovacieho systému. Podrobný popis systému je v sekcií 1.Užívateľské rozhranie -> 1.3.2 Výber bodovacieho systému.
Zmeny v systému, evidujeme stlačením potvrdzovacieho tlačítka.

2.4.2.2 Nahratie ďalších vstupných dát

Aplikáciu si ukladá rôzne stavy akcií. Nahrané údaje bude možné pridať respektíve odobrať. Následné doplnenie ďalších údajov si vyžaduje overenie správnosti dát, či nenastanú nejaké problémy s menami bežcov. Po finalizácií úprav, treba spustiť nové generovanie údajov.

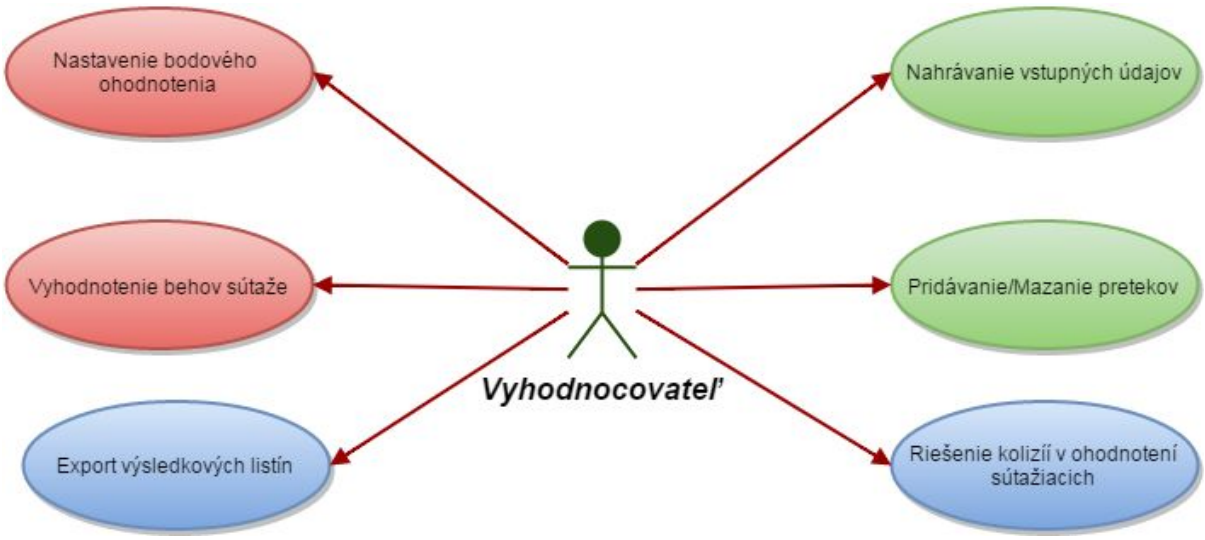
2.4.3 Export výsledkovej listiny

Výsledková listina bude obsahovať prehľad dát. Predtým než sa vyžiada uloženie výsledku, ponúkneme nadhľad k spracovaným údajom. Výber je z dvoch možností formátovania, v ktorej aplikácia vyexportuje výsledok, na základe výberu sa listina uloží ako HMTL respektíve CSV dokument.

2.5 Diagramy

2.5.1 Use-case diagram

Use-case diagram(obrázok č. 2) znázorňuje možnosti použitia - Vyhodnocovateľa. Ako jediný používateľ aplikácie, nie je nijako obmedzený, a môže využívať jej plnú funkcionalitu.

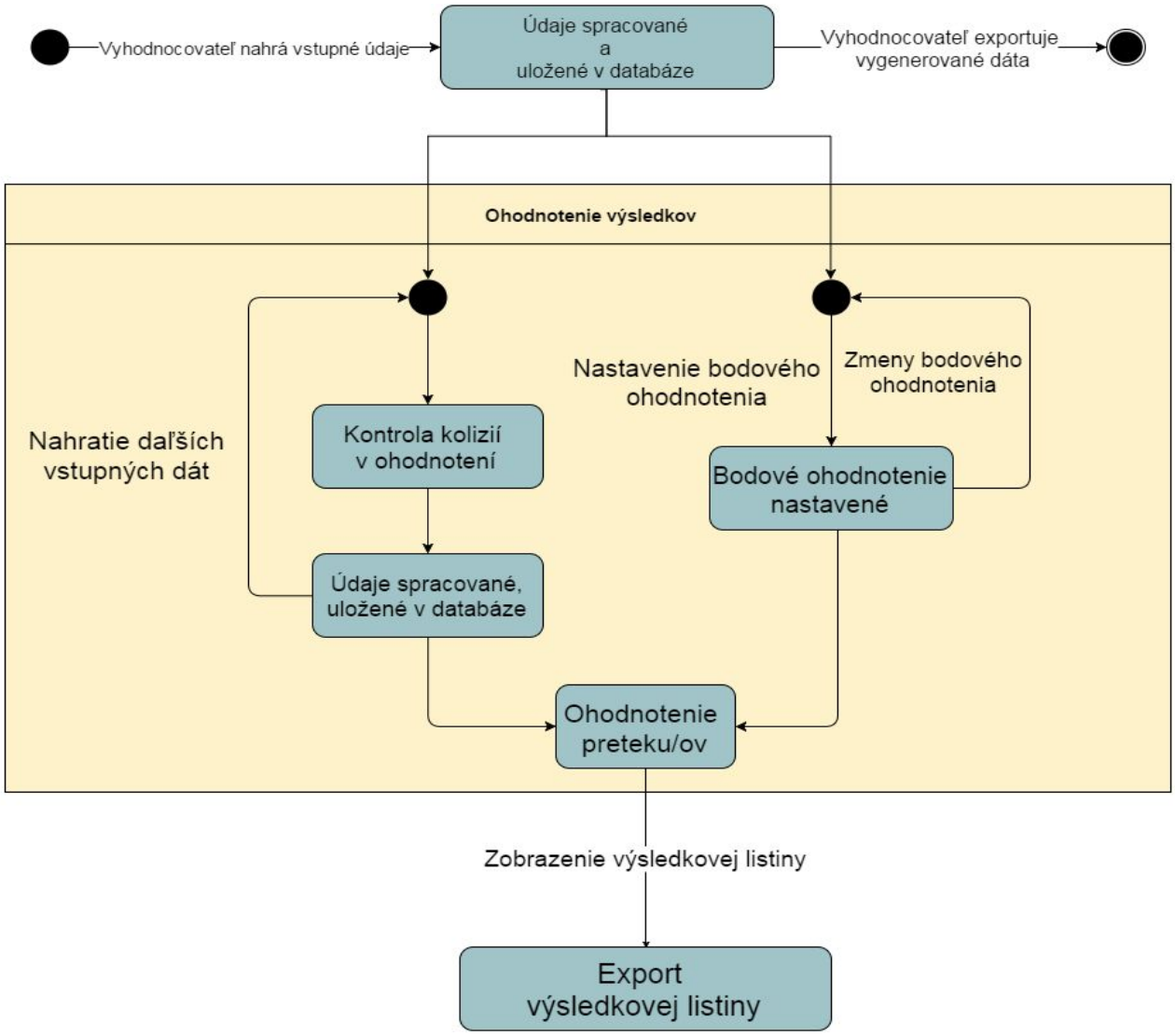


Obrázok 4: Use-case diagram

2.5.2 Stavové diagramy

Použitie webovej aplikácie CupCalculator

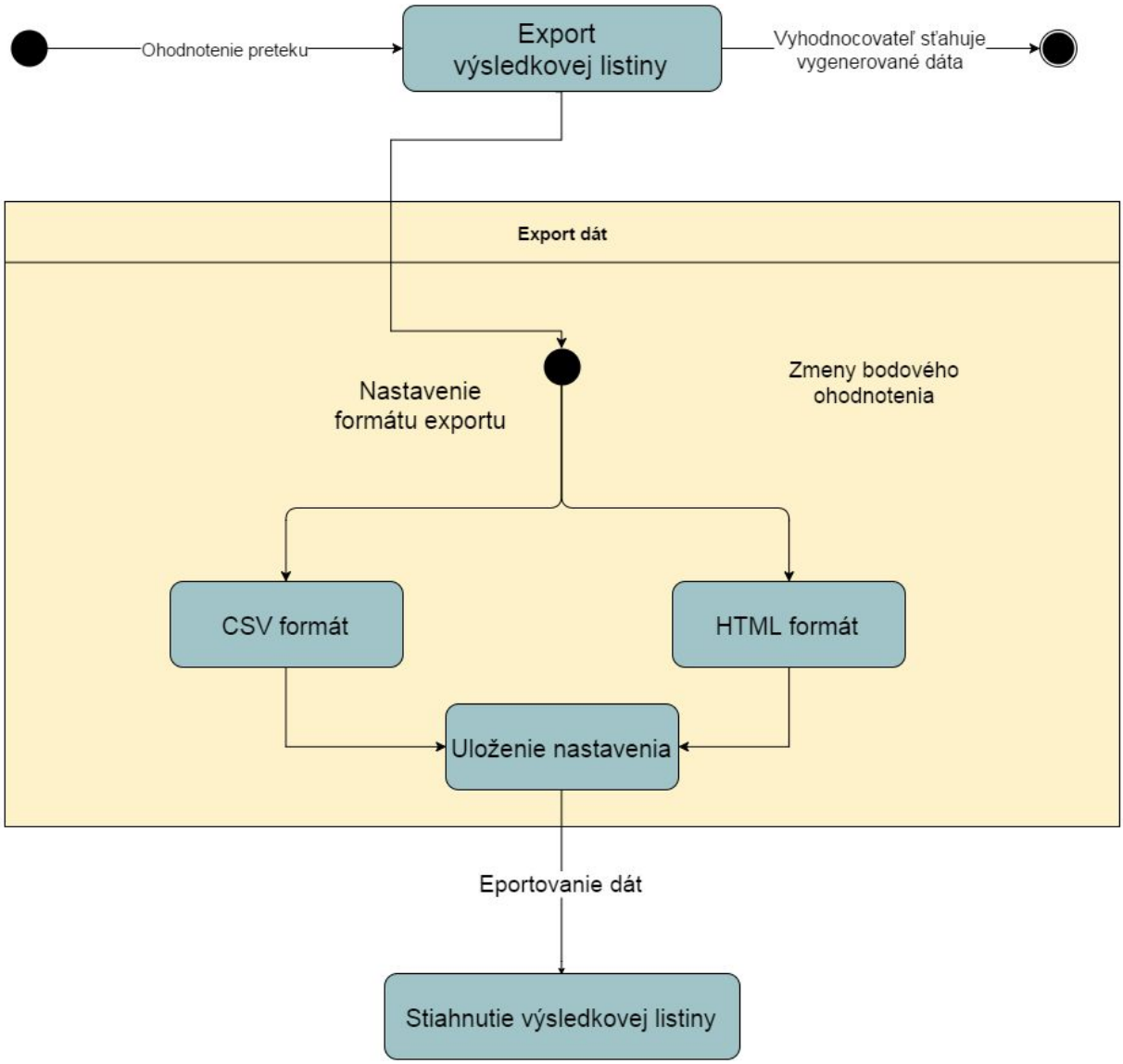
Diagram (obr č. 3) vyjadruje postupnosť stavov pri procese ohodnotenia sútaže.



Obr 3. Stavový diagram ohodnotenia sútaže.

Export výsledkových listín

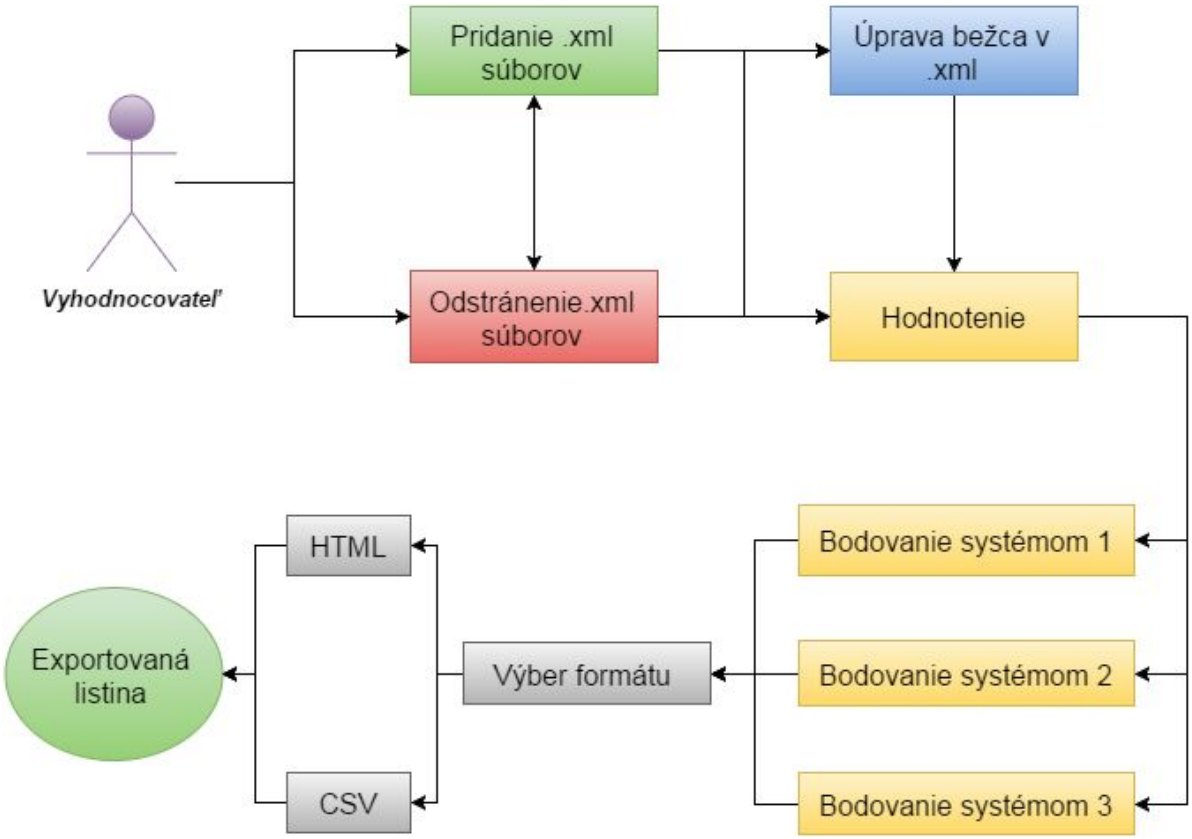
Diagram (obr č. 4) vyjadruje postupnosť stavov pri procese exportovania výsledkových listín pretekov.



obr. č. 4 Stavový diagram importovania a exportovania údajov o zamestnancoch.

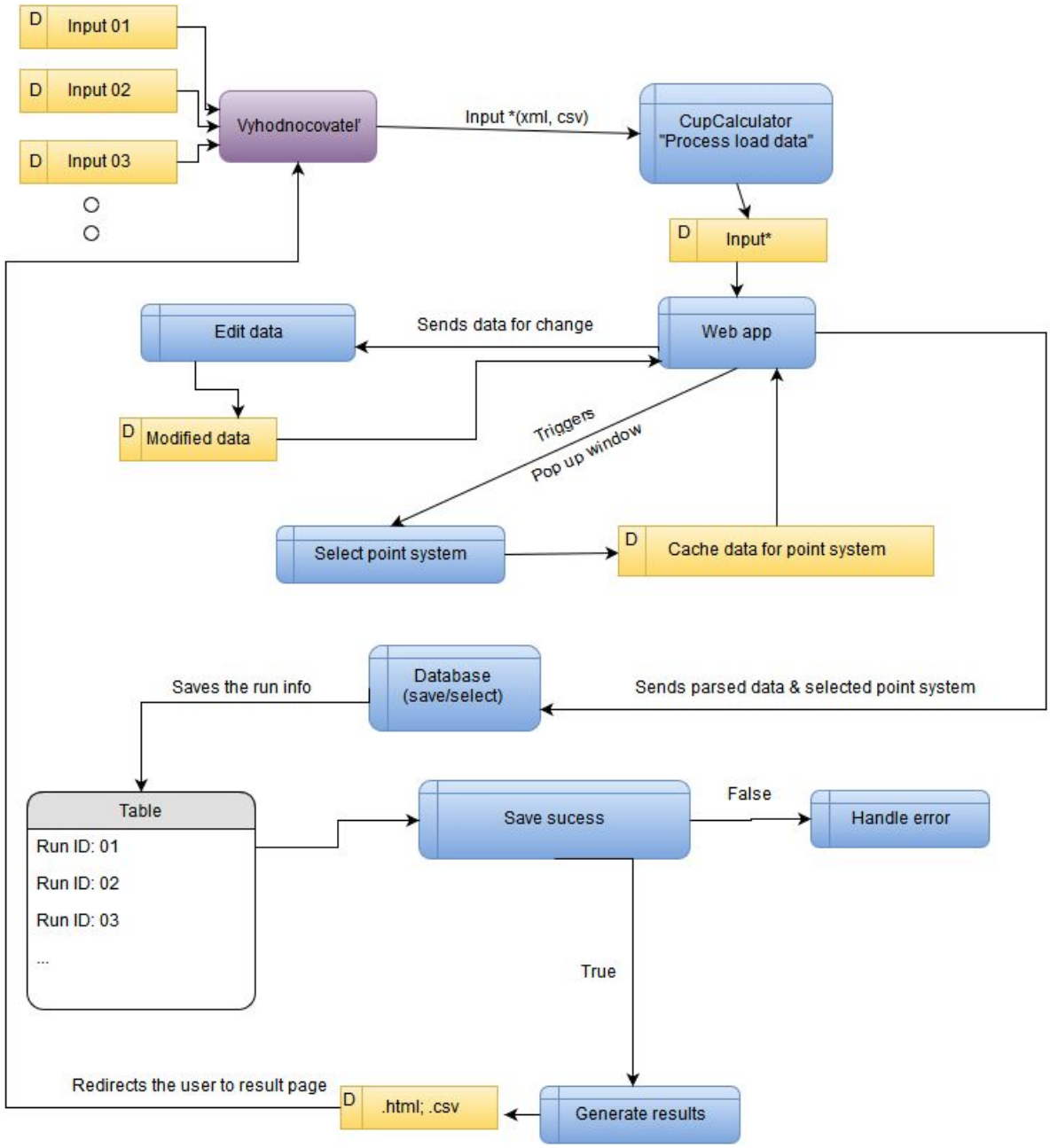
2.5.3 Entitno-relačný diagram

Diagram (obrázok č. 7) znázorňuje jednotlivé relácie medzi komponentami aplikácie
.



obr. č. 5 Entitno-relačný diagram

2.5.4 Data-flow diagram



3 Analýza technológii

3.1 Použité technológie a postupy

Táto kapitola popisuje technológie, ktoré boli použité na vývoj.

3.1.1 Prístupy

Stránka beží pod python-ovskou knižnicou Django, ktorá rozširuje python o možnosť vytvárania web stránok pomocou python skriptov. Jednoduchosť, prehľadnosť a vysoká podpora knižníc pre python, nám umožní využiť silný potenciál systému. Web stránka bude interaktívna a responzívna, čo nám umožni Bootstrap technológia.O spracovanie dát sa postará python skript, spúšťaný priamo na servery cez apachovský modul mod_wsgi.

3.1.2 Použité technológie

3.1.2.1 Django

Open-source knižnica pre python. Je to modul na vytváranie web stránok pomocou pythonu. Syntaktická jednoduchosť a znalosť pythonu, nám umožní využiť skrytý potenciál danej knižnice. Samozrejme, predchádzajúce znalosti so systémom a skúsenosti z praxe, nám uľahčia prácu.

3.1.2.1.1 Python

Programovací jazyk založený na vysokej čitateľnosti kódu, umožňujúci programátorovi zápis kódu do menej riadkov; ako napríklad v JAVA alebo C++. Jadro nášho projektu CupCalculator.

3.1.2.2 HTML

HyperText Markup Language - skrátene HTML; je štandardný značkovací jazyk na vytváranie internetových stránok. Internetový prehľiadač dokáže čítať HTML dokumenty a následne ich renderovať do viditeľnej alebo počuteľnej stránky.

3.1.2.3 CSS

Cascading Style Sheets - skrátene CSS; sa používa na popis prezentácie dokumentu, napísany v HTML dokumenete. Umožňuje grafickú modularitu a úpravú dát do interaktívnej podoby. CSS sa dá využiť aj na spravocanie dát ako XHTML, XML, SVG a XUL.

3.1.2.3.1 Bootstrap

Front-end framework, ktorý rozširuje CSS jazyk o jednoduchú implementáciu responzívného webu. Takisto umožňuje moderný/minimalistický dizajn. Záťaž na systém je minimálny, keďže je to lightweight framework, čo nám umožní rýchlu interakciu s dokumentami.

3.1.2.4 PostgreSQL

Objektovo orientovaný databázový systém, s open-source prístupom. Aktívne vyvíjaný pätnásť rokov. V súčasnosti sa aktívne rozširuje klientela systému, keďže veľa programátorov, preferuje PostgreSQL, oproti alternatíve MySQL.

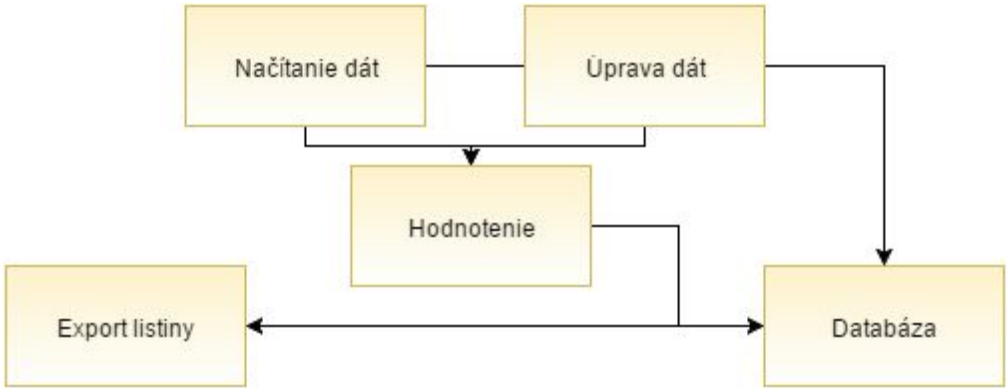
3.1.2.5 Apache

Server beží na Apache - najpoužívanejší web server softvér; s dvadsať ročnou históriou. O pravidelnú údržbu sa stará otvorená komunita, pod dohodou s Apache Software Foundation. Softvér je kompatibilný s Unix, Windows a ďalšími serverovo založenými operačnými systémami.

3.1.2.5.1 WSGI (mod_wsgi)

Na spustenie python skriptov na serveri potrebujeme modul, ktorý dokáže spustiť naše skripty. Často používané CGI skripty, boli nahradené modernou verziou WSGI skriptami. Apache natívne nepodporuje wsgi modul, preto ho treba manuálne doinštalovať a nakonfigurovať.

3.2 Dekompozícia



obrázok 1: komponenty aplikácie

3.2.1 Načítanie dát

Užívateľ nahrá dokumenty do runs sekcií stránky (prídelí aj sezónu). Statická sekcia stránky sa nemení (GUI), zobrazia sa nahrané údaje, ktoré bude možné spracovať, s možnosťami na pridanie, odobratie a modifikácie XML súborov

3.2.2 Úprava dát

Pri načítaní súborov bude môcť užívateľ v prípade nejasností alebo zle uvedených údajov vo vstupe pridávať alebo mazať údaje, keďže môžu nastať problémy pri čítaní údajov bežcov, najmä pri zhode mien.

3.2.3 Hodnotenie

Tento komponent bude slúžiť na výber bodovacieho systému z troch možností, bližšie popísaných v koceptuálnej analýze. Na základe tohoto výberu sa údaje zoradia.

3.2.4 Export listiny

Užívateľ vyberie výstupný formát, v ktorom sa následne bude môcť exportovať už utriedená listina bežcov, na základe zvoleného hodnotenia. Na výber bude výstup vo formáte CSV alebo HTML.

3.2.5 Databáza

Tento komponent bude slúžiť na uchovanie vstupných dát a prácu s nimi. Je to komponent, s ktorým budú všetky ostatné komponenty komunikovať.

4 Databáza a dátový model

Táto kapitola popisuje návrh tabuliek databázy a ich funkciu. Každá tabuľka je zaznačená v dátovom modeli (Obr.2)

4.1 Tabuľky databázy

I. Tabuľka Person

Tabuľka uchováva informácie o bežcoch v danej sezóne. Ich identifikačné číslo, meno a priezvisko, kategóriu, v ktorej súťažia a klub za ktorý bežec beží.

II. Tabuľka Result

V tabuľke sú uložené dosiahnuté časy pretekárov na konkrétnom závode. Okrem času štartu a konca obsahuje identifikačné číslo pretekára,identifikačné číslo závodu a či nebol diskvalifikovaný počas behu.Tabuľka ukladá bodové ohodnotenie pretekárov za ich výkony na jednotlivých behoch priradených hodnotiteľom pomocou bodovacieho systému CupCalculator.

III. Tabuľka Run

V tabuľke sú uložené jednotlivé behy v sezóne. Obsahuje aj názov behu, dátum konania a čas.

IV. Tabuľka Season

V tabuľke uchováva informácie o sezóne. Od kedy do kedy sezóna prebieha.

V. Tabuľka Position

Tabuľka obsahuje celkové poradie pretekárov za sezónu , umiestenie v kategórii, do ktorej bežec patrí, celkový počet bodov.

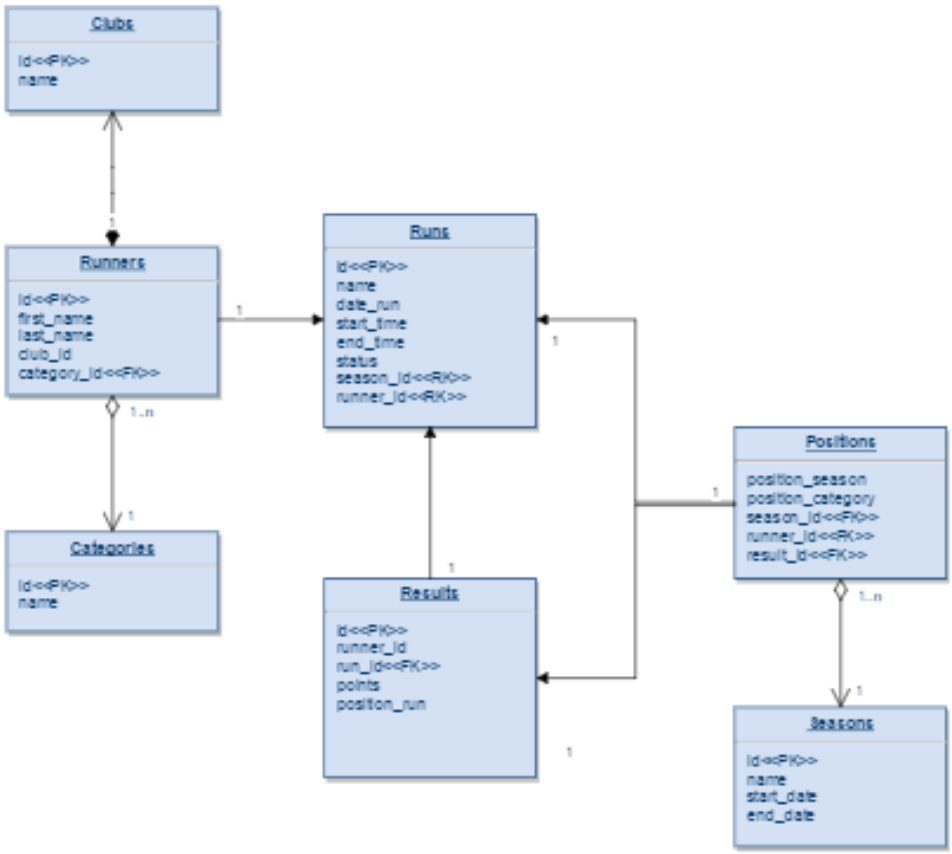
VI. Tabuľka Category

Tabulka obsahuje jednotlive kategorie bezcov.

VII. Tabuľka Club

Tabuľka obsahuje názov klubov pod ktorými bežci závodia.

4.2 Dátový model



5 Komponenty aplikácie

5.1 Komponent Parser

Hlavna trieda, pre spracovanie XML súborov. Ako vstupný parameter dostáva adresu vstupného súboru. Pre podporu rôzných vstupov, parser načíta prvotné údaje z .xml, z ktorých zistí verziu IOF. Podporujeme 2 typy IOF a to 2.0 a 3.0. Parser vráti triedu Parser_ver_2 alebo Parser_ver_3, s dátami.

Vysvetlivky:
person - preposielam premennu, cez metódy
child - preposielam vrchol stromu, aby so mohol pokračovať novým for cyklom. V každej metóde pre prehladnosť kódu sa používa vhodný ekvivalent. Ale pre zjednodušenie dokumentu, dodržim jednotný názov a to child.

Podrobný popis metód triedy Parser.

5.1.1 def __init__(self, file):

Definuje premenné pre parsovanie vstupného súboru. Metóda plní 3 funkcie. Načítanie .xml súboru do ElementTree parseru a definovanie koreňa xml stromu. Inicializácia 3 premenných (class_count, event_data, runners_data).
Následne sa spustia metódy na naplnenie premenných.

5.1.2 def collect_event_data(self):

Nájde event tag v xml súbore (pomocou get_event metódy) a zparsuje tagy: Name, EventClassificationId a StartDate. Tieto dáta sa uložia do event_data premenej.

3.1.2.1 def get_event(self):

For cyklom prejde koreň stromu, kde hľadá tag s názvom Event. Prideli ho premmenej, ktorú funkcia vracia.

3.1.2.2 def parse_startdate(self, child):

Potrebujem sa v noriť do stromu aby som načítal Date a Clock tag, ktoré obsahujú dátum a čas začatia behu. Pridá hodnoty do event_data pod klúčom data a clock.

5.1.3 def collect_runners_data(self):

Takisto ako collect_event_data prehľadáva xml strom, v ktorom hľadá ClassResult.

Ak sa daný tag nájde, zavolá sa metóda handle_classresults. A zvýšim inkrement pre class_count (obsahuje počet kategórií).

5.1.3.1 def handle_classresults(self, child):

Metóda sa vnorí hlbšie do stromu a identifikuje tagy ClassShortName a PersonResult.

ClassShortName - obsahuje skrátený názov kategórie, prideľuje sa lokálnej premennej na úpravu

PersonResult - podrobný výpis výsledkov bežca, slovník s kategóriamy (ClassShortname)

5.1.3.2. def get_personresult(self, child):

Vstupuje do PersonResult tagu, ktorý obsahuje 3 ďalšie tagy (Person, Club, Result). Na každý z nich sa volá vlastná funkcie, ktorá spracuje vrchol xml stromu.

5.1.3.3. def parse_person_tag(self, person, child):

V Person tagu, vyhľadá PersonName tag, ktorý ma 2 tagy: Family a Given. Zavolá funkciu add_name_and_surname.

5.1.3.4 def add_name_and_surname(self, person, child):

Z Family a Given tagov získa meno a priezvisko bežca a prida do slovníka, ktorý obsahuje info o bežcovi.

5.1.3.5 def parse_club_tag(self, person, child):

Spracuje ShortName tag, ktorý ma meno klubu, v ktorý bežec reprezentuje.

5.1.3.6 def parse_result_tag(self, person, child):

Najpodstatnejšie informácie o bežcovi. Obsahuje CCard, StartTime, FinishTime, Time, ResultPosition, CompetitorStatus, CourseLength, SplitTime.

5.1.3.7 def add_ccard(self, person, child):

Pridá ccard hodnotu bežcovi.

5.1.3.8 def add_starttime(self, person, child):

Pridá starttime hodnotu bežcovi.

5.1.3.9 def add_finishtime(self, person, child):

Pridá finishtime hodnotu bežcovi.

5.1.3.10 def add_splittime(self, person, child):

SplitTime tag obsahuje hodnoty etáp behu. Vytvorí dvojicu informácií ohľadom ControlCode a Time, ktorý sa pridá bežcovi ako hodnota splittime_seq_index.

5.2 Komponent databaza

5.2.1 models.py:

Balik pre narabanie s databazou, umoznuje vytvarad datovy model aplikacie a vkladat udaje do databazy. Trieda vytvori spojenie s built-in sqlite3 databazou kt. je defaultne v pythone. V pripade potreby je mozne tuto triedu adaptovat na vykonnejsi databazovy system(napr. PostgreSQL). Po uspesnom pripojeni na databazu je mozne vyuzivat metody na pridavanie udajov do databazy.

5.2.2 class Season(models.Model):

Vytvorenie tabulky pomocou:

```
name = models.CharField(max_length = 128)
start_date = models.DateField()
end_date = models.DateField()
```

Vypis jednotlivych modelov je zapezpeceny cez:

```
def __unicode__(self):
    return self.name
```

5.2.3 class Club(models.Model):

Vytvorenie tabulky pomocou:

```
name = models.CharField(max_length = 128)
shortcut = models.CharField(max_length = 3, default = '000')
```

Vypis jednotlivych modelov je zapezpeceny cez:

```
def __unicode__(self):
    return self.name
```

5.2.4 class Category(models.Model):

Vytvorenie tabulky pomocou:

```
name = models.CharField(max_length = 25)
```

Vypis jednotlivych modelov je zapezpeceny cez:

```
    def __unicode__(self):
        return self.name
```

5.2.5 class Person(models.Model):

Vytvorenie tabulky pomocou:

```
    first_name = models.CharField(max_length = 25)
    last_name = models.CharField(max_length = 25)
    club = models.ForeignKey(Club)
    category = models.ForeignKey(Category)
    ccard  = models.TextField(null = True)
    person_id = models.TextField(null = True)
    runs = models.ManyToManyField(Run, through='Result')
```

Vypis jednotlivych modelov je zapezpeceny cez:

```
    def __unicode__(self):
        return unicode( self.pk ) + ", " + self.first_name + " " + self.last_name
```

5.2.6 class Result(models.Model):

Vytvorenie tabulky pomocou:

```
    person = models.ForeignKey(Person)
    run = models.ForeignKey(Run)
    start_time = models.CharField(max_length = 25,null = True)
    finish_time = models.CharField(max_length = 25,null = True)
    result_time = models.CharField(max_length = 25, null = True)
    status = models.TextField()
    points = models.DecimalField(max_digits=10,decimal_places=5)
    position_run = models.IntegerField(null = True)
```

Vypis jednotlivych modelov je zapezpeceny cez:

```
    def __unicode__(self):
        return unicode( self.pk ) + ", " + self.person.first_name + ", " + self.run.name
```

5.2.7 class Position(models.Model):

Vytvorenie tabulky pomocou:

```
    position_season = models.IntegerField()
    position_category = models.IntegerField()
    season = models.ForeignKey(Season)
    person = models.ForeignKey(Person)
    points = models.DecimalField(max_digits=10, decimal_places=5, default=0)
```

Vypis jednotlivych modelov je zapezpeceny cez:

```
    def __unicode__(self):
```

```
        return unicode( self.position_season ) + ',' + unicode( self.position_category ) +
',' + unicode( self.season ) + ',' + unicode( self.person )
```

5.2.8 class Document(models.Model):

Vytvorenie tabulky pomocou:

```
    xmlfile = models.FileField(upload_to='xml')
    season = models.ForeignKey(Season)
```

Vypis jednotlivych modelov je zapezpeceny cez:

```
    def __unicode__(self):
#        return "AbsPath:" + os.path.abspath(self.xmlfile.name) + "\tBasename: " +
os.path.basename(self.xmlfile.name) + "\tRealPath: " +
os.path.realpath(self.xmlfile.name)
        return self.xmlfile.url
```

5.2.9 class Run(models.Model):

Vytvorenie tabulky pomocou:

```
    name = models.CharField(max_length=128)
    date = models.DateField()
    season = models.ForeignKey(Season)
```

Vypis jednotlivych modelov je zapezpeceny cez:

```
    def __unicode__(self):
        return unicode(self.pk) + ", " + self.name
```

5.3 Komponent Hodnotenie

Tento komponent obsahuje štyri triedy : Evaluation1, Evaluation2, Evaluation3 a Evaluation4. Každá z týchto tried sa stará o jeden systém hodnotenia, vypísaný v špecifikácii aplikácie.

5.3.1 Evaluation1

Táto trieda má za úlohu ohodnotiť bežcov na základe pomeru času víťaza a súťažiaceho.

Ako vstupný parameter dostáva:

1. inputList - pole bežcov, kde bežec má meno (ID), zabehnutý čas, bodové ohodnotenie (na vstupe pravdepodobne 0)
2. evalType - typ hodnotenia na základe požiadavku užívateľa

3. resp 4. valueA, valueB - hodnoty užívateľa potrebné na ohodnotenie
5. percento bežcov použité na výpočet priemerného času preteku

Konštruktor vykonáva priradenie potrebných premenných a rozvetvuje program na tri vetvy na základe evalType, to znamená, spúšťa funkcie definované nižšie na základe evalType. V prípade výberu prvých dvoch spoôsobov hodnotenia zistí najrýchlejší čas a priradí ho do self.winner, v prípade treťom vypočíta priemerný čas preteku na základe zadaného percenta a priradí ho do premennej self.average. Z týchto funkcii sa výsledok priradí do premennej self.outputList, kde už sú bežci ohodnotení.

5.3.1.1 def first (self):

Vypočíta prvý spôsob hodnotenia, definovaný ako: (čas víťaza / čas bežca) * valueA
Priradí ohodnotenie každému bežcovi zo vstupného poľa a toto pole vráti.

5.3.1.2 def second (self):

Vypočíta druhý spôsob hodnotenia, definovaný ako: maximum(0, (2 - čas bežca / čas víťaza)) * valueA
Priradí ohodnotenie každému bežcovi zo vstupného poľa a toto pole vráti.

5.3.1.3 def third (self):

Vypočíta tretí spôsob hodnotenia, definovaný ako: maximum(0, valueA + valueB * (self.average - (čas bežca / self.average)))
Priradí ohodnotenie každému bežcovi zo vstupného poľa a toto pole vráti.

5.3.2 Evaluation2

Táto trieda má za úlohu ohodnotiť bežcov na základe zadaného intervalu medzi jednotlivými bodmi pre bežcov, a bodového ohodnotenia posledného bežca.

Ako vstupný parameter dostáva:
1. inputList - pole bežcov, kde bežec má meno (ID), zabehnutý čas, bodové ohodnotenie (na vstupe pravdepodobne 0)
2. interval - interval medzi jednotlivými bodmi pre bežcov
3. lastPoints - bodové ohodnotenie posledného bežca

Konštruktor vykonáva priradenie potrebných premenných. Potom vytvorí premennú self.evalTable, ktorá je výsledok funkcie listCreate. Slúži na jednoduché hodnotenie podľa vstupných parametrov. Následne spustí funkciu evall, ktorá už pracuje s premennou self.evalTable. Z tejto funkcie sa výsledok priradí do premennej self.outputList, kde už sú bežci ohodnotení.

5.3.2.1 def createList (self):

Má za úlohu vytvoriť pole zrozumiteľné pre funkciu evall. Vezme lastPoints a interval a na základe týchto dvoch údajov vygeneruje pole bodových ohodnotení tak, že prvý prvok poľa je bodové ohodnotenie víťaza, druhé druhého bežca, a tak postupne až po posledný prvok, ktorý je bodovým ohodnotením posledného bežca = lastPoints.

5.3.2.2 def evall (self):

Vypočíta hodnotenie bežcov na základe evalTable, definovaný tak, že víťaz dostane ohodnotenie prvé v poli evalTable, a potom každý ďalší bežec ďalšie ohodnotenie.
Ak je bežcov viac, ako hodnôt v poli, dostanú bežci navyše také ohodnotenie, aké je v poslednom prvku poľa.
Funkcia následne priradí ohodnotenie každému bežcovi zo vstupného poľa a toto pole vráti.

5.3.3 Evaluation3

Táto trieda má za úlohu ohodnotiť bežcov na základe bodovej tabuľky definovanej užívateľom. Ide o pole hodnôt, usporiadané podľa toho, aké body chceme priradiť bežcom, to znamená prvá hodnota poľa patrí víťazovi, druhá druhému bežcovi a tak ďalej.

Ako vstupný parameter dostáva:
1. inputList - pole bežcov, kde bežec má meno (ID), zabehnutý čas, bodové ohodnotenie (na vstupe pravdepodobne 0)
2. evalTable - tabulka hodnotenia - pole bodových ohodnotení bežcov od prvého bežca

Konštruktor vykonáva priradenie potrebných premenných. Následne spustí funkciu evall. Z tejto funkcie sa výsledok priradí do premennej self.outputList, kde už sú bežci ohodnotení.

5.3.3.1 def evall (self):

Vypočíta hodnotenie bežcov na základe evalTable, definovaný tak, že víťaz dostane ohodnotenie prvé v poli evalTable, a potom každý ďalší bežec ďalšie ohodnotenie.
Ak je bežcov viac, ako hodnôt v poli, dostanú bežci navyše také ohodnotenie, aké je v poslednom prvku poľa.

Funkcia následne priradí ohodnotenie každému bežcovi zo vstupného poľa a toto pole vráti.

5.3.4 Evaluation4

Táto trieda má za úlohu ohodnotiť bežcov na základe bodovej tabuľky definovanej užívateľom. Táto tabuľka sa skladá z dvojice percento-body, kde 100 percent je čas víťaza, a každé ďalšie percentá v tabuľke určujú bodové ohodnotenie bežca, ktorý toto percento dosiahol.

Ako vstupný parameter dostáva:
1. inputList - pole bežcov, kde bežec má meno (ID), zabehnutý čas, bodové ohodnotenie (na vstupe pravdepodobne 0)
2. evalTable - tabulka hodnotenia - pole dvojíc bodových ohodnotení a prislúchajúcich percent

Konštruktor vykonáva priradenie potrebných premenných. Taktiež zistí, ktorý bežec zabehol najlepší čas a priradí to do premennej self.winner. Následne spustí funkciu evall. Z tejto funkcie sa výsledok priradí do premennej self.outputList, kde už sú bežci ohodnotení.

5.3.4.1 def evall (self):

Bežci dostanú body podľa percenta a hodnôt v evalTable, kde percento znázorňuje, koľko času sú za víťazom. Ak je bežec pod posledným percentom v evalTable, dostane body na základe tohoto percenta.
Funkcia následne priradí ohodnotenie každému bežcovi zo vstupného poľa a toto pole vráti.

5.4 Komponent Views

5.4.1 def index(request):

Metoda zabezpeci DML operacie a vyrenderovanie Overview stranky.

5.4.2 def runs(request):

Metoda zabezpeci DML operacie a vyrenderovanie Runs stránky. Zaroven zabezpecuje nahravanie novych XML suborov s casmi pretekarov. Tato medota automaticky pusti metodu parse.

5.4.1 def parse(request, documentPk):

Metoda sparsuje XML dokument pomocou komponenta Parser, porovna udaje s databazou a nahra nove udaje, presmeruje renderovanie stranky na komponent DatabasePerson.

5.4.1 def timeSum(timeString):

Metoda vrati prepocitany cas z HH:MM:SS formatu na pocet sekund.

5.4.1 def eval1(request):

Metoda pomocou komponentu Evaluation1 zabezpeci ohodnotenie sezony podla konfiguracie hodnotitela. Pred tym ako ohodnoti jednotlivych bezcov vynuluje im skore. Po ohodnoteni odosle informacie metode vyhodnot.

5.4.1 def vyhodnot(scoredResults, scoredRunId):

Metoda zapise ohodnotene vysledky jednotlivym bezcom a nasledne upravi pozicie bezcov v databaze podla noveho hodnotenia.

5.4.1 def nullPoints():

Metoda vynuluje bodove ohodnotenie bezcov v sezone, metoda sa pouziva pri novom hodnoteni aby neostali stare udaje v databaze.

5.4.1 def eval2(request):

Metoda pomocou komponentu Evaluation2 zabezpeci ohodnotenie sezony podla konfiguracie hodnotitela. Pred tym ako ohodnoti jednotlivych bezcov vynuluje im skore. Po ohodnoteni odosle informacie metode vyhodnot.

5.4.1 def eval3(request):

Metoda pomocou komponentu Evaluation3 zabezpeci ohodnotenie sezony podla konfiguracie hodnotitela. Pred tym ako ohodnoti jednotlivych bezcov vynuluje im skore. Po ohodnoteni odosle informacie metode vyhodnot.

5.4.1 def eval4(request):

Metoda pomocou komponentu Evaluation4 zabezpeci ohodnotenie sezony podla konfiguracie hodnotitela. Pred tym ako ohodnoti jednotlivych bezcov vynuluje im skore. Po ohodnoteni odosle informacie metode vyhodnot.

5.4.1 def score(request):

Metoda zabezpeci vyrenderovanie uvodnej stranky hodnotiacej casti aplikacie.

5.4.1 def database(request):

Metoda zabezpeci DML a vyrenderovanie celej databazy pomocou django tables.

5.4.1 def databasePerson(request):

Metoda zabezpeci DML a vyrenderovanie iba tabulky Person pomocou django tables.

5.4.1 def databaseCategory(request):

Metoda zabezpeci DML a vyrenderovanie iba tabulky Category pomocou django tables.

5.4.1 def databaseClub(request):

Metoda zabezpeci DML a vyrenderovanie iba tabulky Club pomocou django tables.

5.4.1 def databaseResult(request):

Metoda zabezpeci DML a vyrenderovanie iba tabulky Result pomocou django tables.

5.4.1 def databasePosition(request):

Metoda zabezpeci DML a vyrenderovanie iba tabulky Position pomocou django tables.

5.4.1 def databaseSeason(request):

Metoda zabezpeci DML a vyrenderovanie iba tabulky Season pomocou django tables.

5.4.1 def databaseRun(request):

Metoda zabezpeci DML a vyrenderovanie iba tabulky Run pomocou django tables.

5.4.1 def export(request):

Metoda zabezpecuje vygenereovanie vysledkovej listiny.

5.4.1 def merge(request):

Metoda zabezpeci spojenie 2 rozlicnych zaznamov v databaze do jedneho a upraví jednotlivé databazy.

5.4.1 def export_csv(request):

Metoda zabezpeci export vysledkovej listiny do .csv formatu ktory poskytne na stiahnutie.

6 Testovanie

6.1 Všeobecné testovanie

6.1.1 Testovanie užívateľského rozhrania

V tejto kapitole sme testovali správne správanie jednolivých tlačidiel, prvkov aplikácie. Test spočíval v tom, že sme aplikáciu dali “do rúk” kamarátom, ktorí neboli pri jej vývoji a sledovali sme, ako ju budú ovládať a ktoré veci pre nich budú nepochopiteľné alebo nezrozumiteľné.

Doladili sme malé detaily ako napríklad velkosť písma a hrubovytlačené dôležité veci pri hodnotení. Tiež sme vylepšili výpis databázy tak, že po kliknutí na

ňu sa zobrazia kategórie, z ktorých si môže užívateľ vybrať, ktorú zobrazí. Oproti predošlej verzíi je výpis prehľadnejší a ľahko sa užívateľ zorientuje v množstve údajov.

6.1.2 Testovanie načítania vstupných súborov

V tejto časti bolo úlohou otestovať, či sa vstupné údaje správne parsujú a načítajú do databázy. Test prebiehal tak, že sme od zadávateľa získali viaceré vstupné súbory, ktoré sme následne nahrávali do aplikácie a potom porovnávali obsah v databáze, či sa zhoduje s údajmi v súbore.

Doladili sme detaily súvisejúce s ID číslami bežcov, keďže vstupné súbory obsahovali ID užívateľa označene ako ICard, ktoré sme pôvodne chceli použiť ako ID bežca, no zistilo sa, že toto ICard sa mohlo meniť a nemuselo patriť vždy tomu istému bežcovi.

6.1.3 Testovanie správneho zobrazenia databázy

Testovanie správneho výpisu databázy bolo jednoduché, porovnávali sa údaje zobrazené v tabuľke na obrazovke s tabuľkami databázy. Posudzovali sme čitateľnosť a zrozumiteľnosť údajov.

Doladili sme filtrovanie údajov podľa kategórii, čo zvýšilo prehľadnosť a ľahšiu dostupnosť údajov. Taktiež sme tabuľku krajšie zarovnali, kedže sme zistili, že pri viacerých údajoch sa tabuľka nezobrazovala úplne podľa predstáv. Poslednou úpravou bolo pridanie možnosti zvoliť počet údajov na jednu stránku, čím sa aplikácie stala pohodlnejšou aj pre zariadenia s menšou resp. väčšou obrazovkou.

6.1.4 Testovanie hodnotenia bežcov

Sada testov na otestovanie hodnotenia prebiehala tak, že z dostupných vstupných súborov sa vytvorila databáza, ktorá sa následne rôznymi spôsobmi hodnotila, s rôznymi vstupnými hodnotami pre bodovanie. Tiež sa posudzovala zrozumiteľnosť jednotlivých požiadaviek na vstupné údaje.

6.1.4.1 Test ohodnotenia 1

Pri tomto teste sa zistil problém v rozdielnych verziách Pythonu bežiacom na serveri a použitom pri vývoji hodnotenia. Aplikácia totiž nesprávne hodnotila, obodovala len bežcov, ktorých bodový zisk bol celé

číslo. Všetkým ostatným bežcom priradila 0 bodov. Zdrojový kód bol preto prispôsobený verzií Python 2.7, problém sa tým vyriešil.

6.1.4.2 Test ohodnotenia 2

Bolo otestované hodnotenie na základe pevne zadaného intervalu. Aj tu nastal problém s rozdielnou verziou Pythonu, vyriešili sme ho ako v predošlom teste.

6.1.4.3 Test ohodnotenia 3

Testovanie prebehlo úspešne, na základe zadávanie informácii do jedného stringu v aplikácii pribudlo vysvetlenie, v akom formáte treba zadávať bodové hodnoty.

6.1.4.4 Test ohodnotenia 4

Pri tomto testovaní sme prišli na podobný problém, ako v predošlom. Pre užívateľa bol spôsob zadávania hodnôt neprehľadný, preto sme doplnili do aplikácie popis formátu vstupu a tiež príklad, ako vstupné údaje o hodnotení môžu vyzerať.

6.2 Konkrétne scenáre - Parser verzia 2

Podrobný popis testovacích scenárov. Ako ukážka je priložený print screen výsledkov z oficiálnej stránky. Ďalší obrázok znázorňuje vstupné xml datá.

M -10	**1650 m**	**7 / 7**			
1.	Bukovác Filip	Farmaceut Bratislava	19:49		OK
2.	Paluska Karol	Farmaceut Bratislava	20:57	01:08	OK
3.	Šmelík Daniel	TJ Rapid Bratislava	21:58	02:09	OK
4.	Kasza Tomáš	Farmaceut Bratislava	28:27	08:38	OK
5.	Dokupil Daniel	ŠK Sandberg	29:04	09:15	OK
6.	Sýkora Tristan	Individuals/No club	01:02:41	42:52	OK
7.	Kovác Ondrej	ŠK Vazka Bratislava	01:05:43	45:54	OK

M -12	**2010 m**	**8 / 9**			
1.	Balogh Michal	ŠK Sandberg	24:11		OK
2.	Ditri Nicholas	Sokol Pezinok	25:19	01:08	OK
3.	Kotuliak Viktor	Farmaceut Bratislava	30:31	06:20	OK
4.	Mikloš Andrej	Sokol Pezinok	37:13	13:02	OK
5.	Fischer Matej	Sokol Pezinok	38:34	14:23	OK
6.	Kukurugya Šimon	ŠK Sandberg	38:39	14:28	OK
7.	Urban Michal	ŠK Sandberg	40:22	16:11	OK
8.	Dokupil Samuel	ŠK Sandberg	43:40	19:29	OK
-	Hojko Andrej	Sokol Pezinok	01:17:16	53:05	MissingPunch

M -14	**2920 m**	**9 / 10**			
1.	Šimo Matúš	AŠK Pezinok	33:52		OK
2.	Balogh Matúš	ŠK Sandberg	41:20	07:28	OK
3.	Dubovský Filip	AŠK Pezinok	41:50	07:58	OK
4.	Havlík Adam	Sokol Pezinok	48:20	14:28	OK
5.	Schenk Michal	TJ Rapid Bratislava	49:43	15:51	OK
6.	Urban Matej	ŠK Sandberg	50:06	16:14	OK
7.	Kukurugya Matej	ŠK Sandberg	50:50	16:58	OK

Výsledky behov (Dunajská Buzola).

url: http://rg-fba.cloudapp.net/dunajska_buzola/vysledky1.html

Ukážka vstupného xml súboru.

6.2.1 Kontrola kategórií

Správne: Filip Bukovác -> M-10
Nesprávne: priradený do inej skupiny, ako M-10

6.2.2 Kontrola klubu

Správne: Paluska Karol -> Farmaceut Bratislava
Nesprávne: nezhoda klubu, so spravným riešením

6.2.3 Kontrola dlžky behu

Každá kategória ma uvedenú dlžku.
Správne: M-12 -> 2010m
Nesprávne: iná pridelená hodnota ako 2010m

6.2.4 Kontrola účasti

Zobrazuje počet úspešných bežcov.
Správne: M-12 -> 8/9
Nesprávne: M-12 -> 9/9

6.2.5 Kontrola missingpunch

Úspešnosť behu, daného bežca.
Správne: Hojko Andrej -> M-12 -> missingpunch
Nesprávne: Hojko Andrej -> M-12 -> OK

6.2.6 Kontrola času

Za aký čas to bežec zvládol.
Správne: Šmelík Daniel -> 21:58
Nesprávne: iná hodnota

6.2.7 Kontrola časového rozdielu, k prvému bežcovi

Správne: Šmelík Danie -> 2:09
Nesprávne: iné hodnoty

6.2.8 Kontrola počtu kategórií

Správne: 18 kategórií
Nesprávne: iná hodnota

6.3 Konkrétne scenáre - Parser verzia 3

Ukážka xml vstupu, podľa IOF štandardov - verzia 3.0


```
− <ResultList xlmns="http://www.orienteering.org/datastandard/3.0" status="Complete" iofVersion="3.0" creator="ORIS">
  − <Event>
      <Id type="ORIS">1234567891</Id>
      <Name>YZC2016</Name>
    − <StartTime>
        <Date>2016-01-17</Date>
      </StartTime>
    </Event>
  − <ClassResult>
    − <Class sex="">
        <Id type="ORIS">68026</Id>
        <Name>M10</Name>
      </Class>
    − <PersonResult>
      − <Person>
          <Id type="CZE">FBA0601</Id>
        − <Name>
            <Given>Karol</Given>
            <Family>Paluska</Family>
          </Name>
        </Person>
      − <Result>
          <StartTime>2016-01-17T11:06:23+01:00</StartTime>
          <FinishTime>2016-01-17T11:14:04+01:00</FinishTime>
          <Time>461</Time>
          <Position>1</Position>
          <Status>OK</Status>
        − <SplitTime>
            <ControlCode>62</ControlCode>
            <Time>39</Time>
          </SplitTime>
```


Zparsované a ohodnotené dáta

M-10

Poradie	Meno	Priezvisko	Person ID	Trojkr??ov? preteky 2016	YZC2016	Celkom
1	Karol	Paluska	FBA0601	35.00000	50.00000	85.00000
2	Jakub	Brn?k	0	50.00000	0	50.00000
3	Daniel	Ondovc?k	0	45.00000	0	45.00000
4	Daniel	Šmelík	RBA	0	45.00000	45.00000
5	Jakub	Brňák	SKS	0	40.00000	40.00000
6	Daniel	Smel?k	0	40.00000	0	40.00000
7	Matúš	Mikloš	SPE	0	35.00000	35.00000
8	Tristan	Sýkora	FBA	0	30.00000	30.00000