Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

Informační systém pro zpracování sportovních závodů v reálném čase

Tomáš Litera

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Katedra informačních technologií

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Tomáš Litera

Informatika

Název práce

Informační systém pro zpracování sportovních závodů v reálném čase

Název anglicky

Information system sporting events in real time

Cíle práce

Cílem práce je navrhnout, realizovat a uvést do provozu informační systém určený pro vodácký skautský závod Přes 3 pražské jezy (3jezy.skauting.cz). Zajistit účastníkům závodu rychlou a snadnou registraci a pořadatelům nástroj na zpracování zadaných dat a jejich využití v průběhu a po závodu (aktuální časy, výsledkové listiny, diplomy).

Metodika

Tato práce se zabývá problematikou informačních systémů pracujících v reálném čase, které jsou určené pro pořádání závodů. V úvodu se práce zabývá analýzou problému jako celku, následně pak dekompozicí na dílčí problémy a vlastnosti. Pomocí abstrakce jsou následně vymezeny určité charakteristiky, vlastnosti a vztahy systému s realitou. Z podstatných vlastností a vztahů bude dále realizován fungující model systému. Metodou komparace jsou vybrány nejlepší možné technologie pro běh aplikace na serveru.

Doporučený rozsah práce

30 – 40 stran

Doporučené zdroje informací

, Choi, W., Kent, A., Lea, Ch., Prasad, G., Ullman, CH. Beginning PHP 4. Wrox Press Ltd., 2000, ISBN 1861003730

Bernd Öggl, Michael Kofle. PHP 5 a MySQL 5, Průvodce webového programátora. Brno : Computer Press, a.s., 2007. ISBN 978-80-251-1813-9.

Jiří, Kosek. PHP, tvorba interaktivních internetových aplikací. Praha : Grada Publishing, spol. s r. o., 1999. 7. dotisk (2004). ISBN 80-7169-373-1.

Schneider, Robert D. MySQL, Oficiální průvodce tvorbou, správou a laděním databází. Praha : Grada Publishing, a.s., 2006. První vydání. ISBN 80-247-1516-3.

Předběžný termín obhajoby

2015/06 (červen)

Vedoucí práce

Mgr. Ing. Vladimír Očenášek

Elektronicky schváleno dne 10. 3. 2015

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 11. 3. 2015

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 11. 03. 2015

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci „Informační systém pro zpracování sportovních závodů v reálném čase“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne

Poděkování

Rád bych touto cestou poděkoval vedoucímu bakalářské práce Mgr. Ing. Vladimíru Očenáškovi za konzultace, odbornou pomoc, korekturu, dobré nápady a trpělivost. Taktéž vyjadřuji své vřelé poděkovaní RNDr. Janu Fischerovi, PhD. za poskytování podkladů pro další práci, Petru Kachlíkovi za konzultace a mentoring při budování systému a RNDr. Davidu Svobodovi, PhD. za jazykovou korekturu práce.

Informační systém pro zpracování sportovních závodů v reálném čase

---

Information system sporting events in real time

Souhrn

Tato práce se zabývá problematikou informačních systémů pracujících v reálném čase, které jsou určené pro pořádání závodů – zpracováním dat před, v průběhu a po závodu, jejich interpretací a exportem. Jsou zde popsány technologie pro tvorbu dynamických WWW stránek. Dále je uveden návrh a implementace se všemi použitými technologiemi (PHP, Nette, MySQL, JS, JQuery). Je zde také popsán způsob komunikace (SOAP) s informačním systémem založeném na jiné technologii (ASP.NET, MSSQL). Vše s ohledem na použitelnost, rozšiřitelnost a aplikovatelnost na jiné platformy (Open Source).

Summary

This thesis deals with information systems for real-time competition processing, which are designed for organizing and processing data before, during and after the race and their final interpretation and export. Technologies for development of dynamic web pages are described in this thesis. Also design and implementation of all used technologies (PHP, Nette, MySql, JS, JQuery) is mentioned here. There is also disclosed a method of communication (SOAP) for an information system based on a different technology (ASP.NET, MSSQL). Everything with regard to usability, scalability and applicability to other platforms (Open Source).

Klíčová slova: Zpracování v reálném čase, závody, SOAP, MySQL, PostgreSQL, MS SQL, Oracle, PHP, Java, ASP.NET, Nette, Symphony, Javascript, JQuery, Open Source, CMS

Keywords: Real-time processing, competition, SOAP, MySQL, PostgreSQL, MS SQL, Oracle, PHP, Java, ASP.NET, Nette, Symphony, Javascript, JQuery, Open Source, CMS

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Masivní rozšíření internetu vyneslo do popředí zájmu webové aplikace. Jedním z druhů těchto aplikací jsou informační systémy (IS). Umožnují publikovat obsah na web, ukládat, sdílet a spravovat soubory, řídit společnosti a pomáhat při rozhodování, třídit, uchovávat a zpracovávat informace. Staly se tak nedílnou součástí našeho života a dennodenně je používáme, protože nám ulehčují práci.

Právě takový informační systém je předmětem této práce. Informační systém, který má pořadatelům závodů ulehčit práci při jejich pořádání. a to již od samotného začátku, kdy je potřeba získat data o účastnících, kteří se zaregistrují do závodů, jejich zpracování a roztřídění, přes průběh samotného závodu. V průběhu závodu soutěžící procházejí různými stanovišti, kde získávají body nebo jenom čekají, a jednotlivé hodnoty je třeba zaznamenat a konečně na závěr závodu či soutěže, je potřeba výsledky maximálně rychle, ale přesto přesně vyhodnotit. V úplném závěru pak tyto výsledky a data publikovat na web nebo jenom převést do podoby vhodné k tisku.

A to vše je nutné, aby informační systém prováděl v reálném čase, tedy poskytoval uživateli maximálně aktuální možné informace z průběhu závodu či soutěže.

Hlavním smyslem textu je tedy podat informace o vzniku, vývoji, nasazení a provozu takového systému. Hned na začátku také vyvstává řada otázek, na které bude v průběhu práce odpovězeno: Na čem systém poběží? V jakém programovacím jazyce bude napsán? Co se bude dít s daty? Jak bude systém reagovat na různé stavy? Postupnou analýzou problémů bude na většinu těchto otázek odpovězeno. Tato analýza se poté následně promítne do samotného návrhu informačního systému a poté do jeho realizace. Správnost rozhodnutí a realizace bude potvrzena či vyvrácena testováním.

Realizačním záměrem je vytvořit aplikaci, která bude dostatečně obecná, aby se dala použít jak pro závody konající se na vodě nebo na suchu, tak i slalomové, běžecké či lyžařské, skautské, studentské či veřejné, závody, kterých se budou účastnit skupiny či jednotlivci, nehledě na různé věkové či výkonnostní kategorie.

I přes technologický pokrok dnešní doby stále existují sportovní akce, kde se data zpracovávají ručně (pomocí Excelu). Jedna z takových akcí je závod „Napříč Prahou přes 3

jezy“[1]. Praktická část této práce se proto zaměřuje na vývoj sportovního informačního systému právě pro tuto akci.

[1] http://3jezy.skauting.cz/

2 Cíl práce a metodika

Cílem práce je navrhnout, realizovat a uvést do provozu informační systém určený pro vodácký skautský závod Přes 3 jezy. Zajistit účastníkům závodu rychlou a snadnou registraci a pořadatelům nástroj na zpracování zadaných dat a jejich využití před, v průběhu a po závodu.

Tato práce si neklade za cíl vytvořit plně funkční informační systém se všemi jeho funkcemi a vlastnostmi. Hlavním cílem je připravit alespoň základ takové aplikace, která bude moci být v budoucnu dále rozšiřována o požadované funkce a vlastnosti bez větších problémů. Zároveň si tato práce klade za cíl uvést a popsat souvislosti a procesy, které probíhají při realizaci a vývoji informačního systému pro sportovní závody a informačních systémů obecně.

Úvod práce je zaměřen na seznámení se s vývojem technologií, které se používají pro tvorbu webových aplikací - a to jak na straně klienta (v prohlížeči), tak i na straně serveru. Dále následuje rozbor komunikace webových aplikací s uživatelem, aplikační jazyky a ukládání dat do databází za účelem vzniku informačních systémů. V neposlední řadě budou také uvedeny důvody vedoucí k využití jazyka PHP pro hlavní část této práce.

Vlastní práce se zabývá čistě vývojem samotného informačního systému. Od analýzy požadavků a vlastností systému, přes návrh běhového prostředí - použitého aplikačního jazyka, frameworku, databáze, serverové konfigurace - až k samotnému programování a tvorbě databázového a objektového modelu aplikace. V neposlední řadě také dojde na testování vytvořeného systému (a to i na jiných platformách) a jeho nasazení do reálného provozu.

Závěr práce pak zhodnocuje celkový průběh vývoje, snaží se shrnout chyby a nedostatky, které se během vývoje objevily, a obsahuje také návrh funkcionality pro další vývoj.

3 Přehled řešené problematiky

3.1 World Wide Web

V roce 1990, kdy byla služba World-Wide Web poprvé spuštěna na půdě výzkumného centra CERN, jsme si vystačili s pouhými třemi technologiemi.

3.1.1 HTML

První z nich byl jazyk HTML (HyperText Markup Language), který sloužil k zápisu webových stránek. HTML je dodnes ústřední technologií Webu[2], okolo které se vše točí. [8]

3.1.2 HTTP

Druhou nezbytnou technologií je přenosový protokol http (HyperText Transfer Protocol), který zajišťuje přenos HTML stránek z WWW serveru do prohlížeče. Původní verze http 0.9 byla velmi jednoduchá. v důsledku zvýšených požadavků na možnosti kontroly přenosu dokumentů a samotné zrychlení přenosu postupně vznikly nové verze http 1.0 a 1.1. [8]

HTTP 1.1 je dnes již standardem, který podporují všechny WWW servery a prohlížeče.[3]

3.1.3 URL

Třetí technologií nezbytnou pro implementování služby WWW jsou URL (Uniform Resource Locator). Každý objekt přístupný na Webu má svoji jedinečnou URL adresu, která slouží k vytváření odkazů na daný objekt. [8]

Z dnešního pohledu spojení těchto tří technologií nenabízí mnoho – umožňuje pouze prohlížení elektronických dokumentů, které jsou provázány systémem odkazů.

3.2 Komunikace klient/server

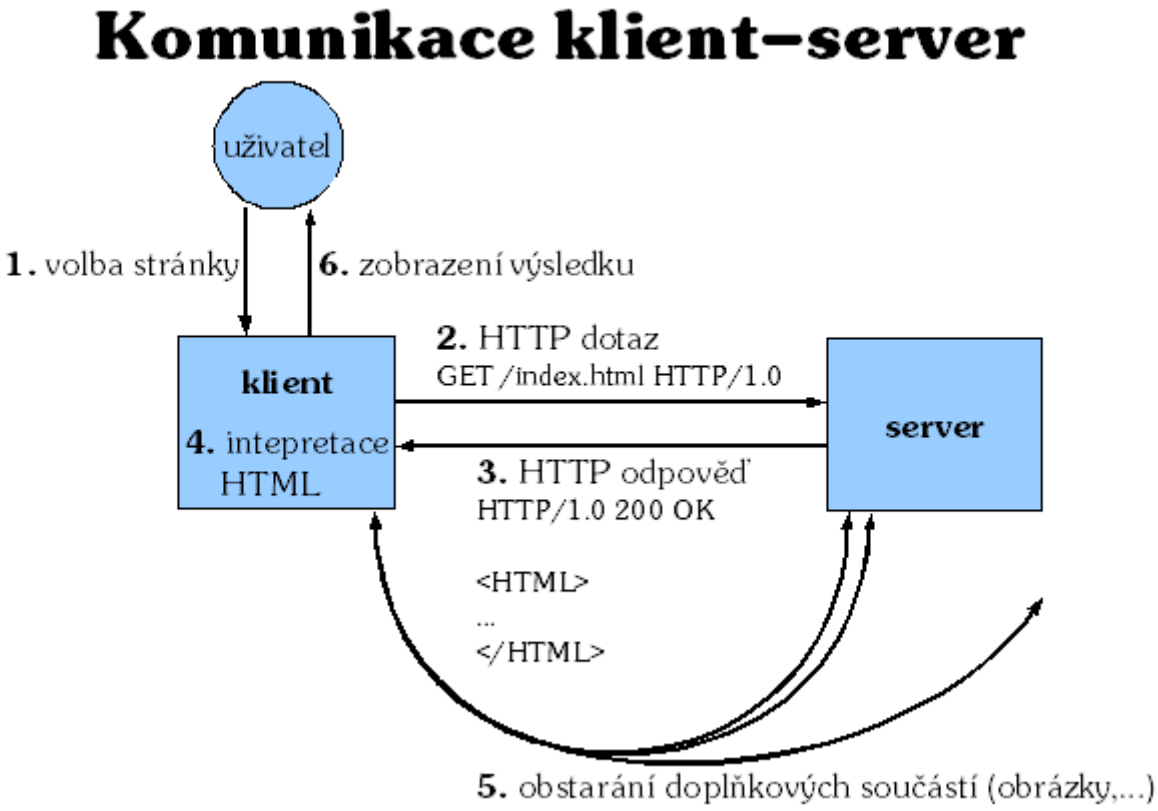
Pokaždé, když se díváme na web, abychom si prohlédli nějakou webovou stránku, vytváříme tím automaticky spojení na webový server. Proces odeslání URL je nazýván podáním požadavku

[2]Dnes již existuje jazyk HMTL ve verzi 5.0, nicméně je stále zpětně kompatibilní s původní jednoduchou verzí HTML. (http://www.w3.org/html/wg/drafts/html/CR/)

[3]http://tools.ietf.org/html/rfc2616

(request) na server. Server interpretuje URL, najde odpovídající stránku a odešle ji zpět jako část toho, co nazýváme odpovědí (response) webovému prohlížeči. Prohlížeč poté vezme kód, který obdržel od webového serveru a zkompiluje a zobrazí stránku. Prohlížeč je v této interakci označován jako klient a celá interakce jako vztah klient-server. [18]

Tento termín popisuje efektivní fungování webu na základě distribuce úkolů. Server (webový server) ukládá, interpretuje a distribuuje data a klient (prohlížeč) přistupuje k serveru, aby data získal. Pokaždé, když použijeme termín klient, odkazujeme se tak na prohlížeč. [18]



Obrázek 1: Komunikace klient - server

[24]

3.2.1 Klient

Klient je počítač (hardware) nebo počítačový program (software), který přistupuje ke vzdálené službě na jiném počítačovém systému, označovaném jako server. [19]

3.2.1.1 Prohlížeč

Webový prohlížeč (též browser) je počítačový program, který slouží k prohlížení World Wide Webu (WWW). Program umožňuje komunikaci s HTTP serverem a zpracování přijatého kódu (HTML, XHTML, XML apod.), který podle daných standardů zformátuje a zobrazí webovou stránku. Textové prohlížeče zobrazují stránky jako text, obvykle velmi jednoduše formátovaný. Grafické prohlížeče umožňují složitější formátování stránky včetně zobrazení obrázků. Pro zobrazení některých zvláštních součástí stránky, jako jsou Flash animace nebo Java applety, je třeba prohlížeč doplnit o specializované zásuvné moduly. Mezi nejznámější webové prohlížeče patří grafické (seřazeny podle počtu uživatelů)[4] Google Chrome, Internet Explorer, Mozilla Firefox, Safari, Opera, Maxthon a textové Links a Lynx. [23]

3.2.2 Server

Webový server je software, který umožňuje, aby webové stránky byly pro všechny dostupné. Dalším úkolem webového serveru je poskytovat prostor (typicky v adresáři nebo struktuře složek), ve kterém jsou uloženy a organizovány webové stránky nebo celá webová aplikace. Existuje mnoho komerčních a freeware webových serverů, avšak dva převažují a zabírají téměř 70 % trhu. Tyto dva produkty jsou webový server Apache a Internetový informační server (IIS)[5] od firmy Microsoft. [18]

3.2.3 Internetové protokoly

Internet je síť vzájemně propojených uzlů. Je určen pro přenos informací z jednoho místa na druhé. Využívá sadu síťových protokolů (známé jako TCP/IP) pro přenos informací. Síťový protokol je pak jednoduše metoda zápisu informačních paketů tak, aby mohly být zaslány do vašeho telefonu nebo přes kabel či linku z jednoho uzlu do dalšího, dokud nedorazí do svého určeného cíle.

Jednou z výhod protokolu TCP/IP je to, že může velmi rychle přesměrovat informace, pokud jsou některé konkrétní uzly nebo trasy rozbité nebo nedostupné, či pomalé. [18]

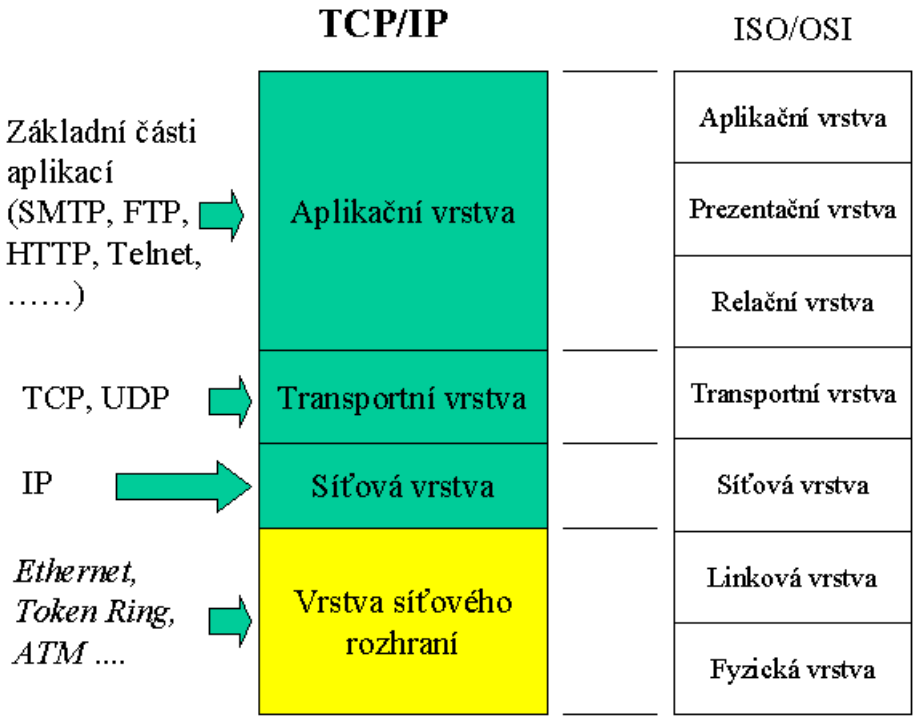
[4] http://gs.statcounter.com/#browser-ww-monthly-201205-201205-bar

[5] Internet Information Server

3.2.3.1 TCP

Když uživatel použije webový prohlížeč, aby načetl stránku, tak prohlížeč zabalí tuto instrukci pomocí protokolu TCP (Transmission Control Protocol)[6]. TCP je transportní protokol, který poskytuje spolehlivý přenosový formát pro instrukce. Zajišťuje, že celá zpráva je zabalena správně pro přenos (a také to, že je správně rozbalena a dána dohromady, poté co dosáhne cíle).

Síťový protokol TCP/IP je pak metoda popisu informačních paketů tak, aby mohly být poslány po telefonní lince, kabelem nebo jiným nosičem z uzlu do uzlu. [18]



Obrázek 2: TCP/IP vrstvy

[12]

3.3 Skriptování na straně serveru

Servery tedy obsluhovaly požadavky od klientů a vracely statické stránky. Brzy ale tato funkcionality vývojářům nestačila. První inovací byla možnost automatického generování

[6] přenosový kontrolní protokol

stránek, které obsahují informace proměnlivé v čase. HTML stránka je soubor uložený na disku WWW serveru, který má své URL. Nic však nebrání tomu, aby URL ukazovalo na nějaký spustitelný soubor (program), který vygeneruje HTML stránku. Tato stránka pak může obsahovat aktuální informace. Spustitelný soubor je vyvoláván WWW serverem a bylo proto zapotřebí rozhraní, které by definovalo způsob spuštění programu a předávání dat mezi WWW serverem a programem.

3.3.1 CGI

Rozhraní se jmenuje CGI (Common Gateway Interface). Programům, které generují HTML stránky, se proto často říká CGI skripty. [8]

Je to protokol pro propojení externích aplikací s webovým serverem. to serveru umožňuje delegovat požadavek od klienta na externí aplikaci, která dle požadavku vrátí výstup. Taková aplikace typicky zpracuje nějaký skript ve webové stránce a webovému serveru navrátí statickou stránku, která je následně poslána klientovi jako výstup jeho požadavku.

3.3.2 HTML 2.0 a CGI

Další vývoj přirozeně směřoval k tomu, aby uživatel mohl ovlivnit chování CGI skriptu. v HTML 2.0 se tedy objevily elementy, které umožňovaly na stránce definovat formulář. Údaje vyplněné uživatelem do formuláře odeslal prohlížeč serveru a ten je pomocí CGI rozhraní předal CGI skriptu k dalšímu zpracování. Tímto způsobem funguje na Internetu mnoho služeb dodnes. Různé vyhledávací servery jsou typickým příkladem. Uživatel zadá do vstupního pole formuláře klíčová slova. Ta se odešlou vyhledávacímu serveru, kde CGI skript prohledá indexy. Výsledkem běhu CGI skriptu je pak stránka HTML, která obsahuje odkazy na stránky vyhovující dotazu. [8]

Vše vypadá jednoduše, ale skutečnost bývá složitější. Psaní CGI skriptů nebylo úplně snadné. Pro jejich psaní se používaly nejčastěji různé interpretované jazyky, jako Perl nebo příkazové shelly Unixu. Nebyl však problém použít v podstatě libovolný programovací jazyk a tak existuje mnoho CGI skriptů napsaných v jazycích C a C++. [8] Jazyk Java lze pro psaní CGI skriptů rovněž použít – takovým programům se pak říká servlety. Velké databázové systémy, jako např. Oracle, umožňují psaná CGI skriptu přímo ve svém vlastním jazyce (např. PL/SQL).

Pro vytvoření CGI skriptu tedy byla nutná znalost nějakého programovacího jazyka. Kromě toho musel člověk ovládat rozhraní CGI, které nepředávalo parametry v příliš šikovném formátu. Většina CGI skriptů se proto z větší části skládala z kódu, který převáděl získané parametry do použitelné podoby.

V interpretovaných CGI skriptech navíc spoustu práce stálo dostatečné zabezpečení skriptu. Šikovný hacker totiž mohl odesláním speciálního textu v polích formuláře získat přístup k systému, na kterých běžel WWW server. [8]

CGI skripty generují svůj výstup v jazyce HTML. Jazyk HTML však nejde přímo kombinovat s jinými jazyky, a proto bylo generování HTML kódu v CGI skriptu otravná záležitost, kde se každá řádka HTML kódu zadávala jako parametr příkazu print či echo podle použitého jazyka. Správa větších aplikací byla rovněž náročná, protože aplikace je roztroušena v mnoha samostatných souborech s HTML stránkami a CGI skripty. [8]

I technologie CGI má však své meze. CGI skripty se provádějí na WWW serveru. Uživatelská odezva je tedy velmi pomalá – uživatel si stáhne stránku, poté vyplní a odešle formulář zpět na server, server spustí CGI skript a od něj získaný výstup zašle zpět do uživatelova prohlížeče. [8]

3.3.2.1 Server Side Includes

Zhruba ve stejné době jako CGI skripty se rozšířila i další technologie SSI (Server Side Includes). SSI byly jednoduché příkazy, které se zadávaly do HTML stránky jako komentář. Stránky však byly uloženy v souborech se speciální přílohou .shtml a tak WWW server věděl, že před odesláním stránky v ní má provést všechny SSI. SSI umožnily provádění jednoduchých úkonů, jako je vložení jiného souboru do stránky, nebo vypsání data poslední modifikace dokumentu. Své uplatnění nalezly především na rozsáhlých serverech, které chtěly mít na všech stránkách standardizované záhlaví a patičku – optimální úkol pro nasazení SSI. [8]

Řešení pomalé odezvy CGI skriptů spočívalo v přesunutí provádění programů na stranu klienta – do prohlížeče. Zhruba ve stejné době – během roku 1996 – byly představeny dvě různé technologie, které daný problém řeší. [8]

3.4 Programovací jazyk

3.4.1 Java

První technologií byl nový jazyk Java představený firmou Sun Microsystems. Tento jazyk umožňoval psaní Java-apletů, což byly krátké programy, které byly začleněny přímo do HTML stránky. ve stránce měly vyhrazen prostor, který byl zcela pod jejich kontrolou. Možnosti Javy jsou opravdu široké – od jednoduchých animací oživujících stránku až po zábavné hry či aplikace, které vám spočítají hladinu alkoholu v krvi. [8]

Velkou výhodou Javy byla její nezávislost na platformě – programy se po síti přenášely ve formě tzv. byte-code (bajtového kódu), který je spustitelný v libovolném operačním systému, pokud pro něj existuje interpret Javy (JVM). Dnes můžeme s odstupem času říci, že Java je výborná a perspektivní technologie. na běžných stránkách se s ní však zatím moc nepotkáváme, protože je kvůli své univerzálnosti poměrně náročná na systémové zdroje počítače. [8]

Java je plně objektově orientovaný multiplatformní programovací jazyk (neumožňuje již tvorbu programů, které by nebyly objektově orientované). Jeho univerzální využití, počínaje programy pro mobilní telefony a internetové aplikace, až po rozsáhle aplikace pro firmy a desktopové aplikace, ho vysazuje na přední příčky všech používaných jazyků. Obecně je však Java náročnější na zdroje a vyžaduje virtuální stroj – mezivrstvu, na které běží. Vzhledem ke své objektovosti vyžaduje vysokou míru abstrakce a zvládnutí vyšších technologií. [16]

3.4.2 Javascript

Druhou novou technologií roku 1996 byl JavaScript. JavaScript je jednoduchý jazyk se syntaxí vycházející z jazyku Java. s JavaScriptem přišla firma Netscape a zabudovala jej do svého legendárního prohlížeče Netscape Navigator. JavaScript se zapisoval přímo do HTML kódu stránky a uměl posloužit v mnoha situacích. Jeho nejčastější použití bylo ve spojení s formuláři. Krátké skripty v JavaScriptu mohly kontrolovat správnost údajů v polích formuláře ještě před odesláním na server. Uživatel tak získal nesrovnatelně rychlejší odezvu v porovnání s klasickým způsobem využívající pouze CGI skripty. Druhou oblastí použití JavaScriptu byla drobná vylepšení interaktivnosti stránek – celkem snadno šlo např. zařídit,

aby odkaz změnil barvu po přejetí myší. Uživatel tak hned věděl, že za odkazem se (možná) skrývá něco zajímavého. [8]

Dnes Javu i Javascript podporují všechny nejrozšířenější prohlížeče. Smutnou pravdou však je, že implementace JavaScriptu mezi prohlížeči není zcela kompatibilní. to nutí tvůrce stránek k vytváření složitějších skriptů, které se umějí přizpůsobit vlastnostem jednotlivých prohlížečů.

Úspěch JavaScriptu byl tak obrovský, že se firma Netscape rozhodla pro využití JavaScriptu na straně serveru. na serverech Netscape šlo do HTML stránek psát skripty, které se provedou přímo na serveru. Uvnitř stránky byly skripty uzavřeny mezi tagy `<server>` a `</server>` a server tak snadno rozpoznal, které části stránky má interpretovat. Výsledkem skritpů musel být HTML kód, který se doplnil do zbytku stránky, a prohlížeči se již zasílala obyčejná HTML stránka. Řešení se dříve šířilo pod názvem LiveWire, dnes je jméno výstižnější – SSJS (Server Side JavaScript). [8]

V současné době jde technologie ještě dál a existují tak i nativní Javascriptové servery. v současnosti nejvíce využívaná je NodeJS[7].

3.4.3 ASP

Aby Microsoft nezůstal pozadu, uvedl na trh ASP (Active Server Pages). ASP jsou obdobou SSJS. Jako programovací jazyk je možno využít VBScript nebo JScript, což je implementace JavaScriptu od Microsoftu. Systémy samozřejmě nejsou kompatibilní – ASP používá jiné značky k oddělení skriptu od stránky a hierarchie objektů, které zpřístupňují všechny důležité údaje, je rovněž rozdílná.

SSJS i ASP mají jednu velkou společnou nevýhodu – jsou to komerční produkty, které nejsou nikterak levné a jejich použití je navíc svázáno s použitím WWW serveru dané firmy. ASP navíc běží pouze na platformě Windows – pokud se tedy v budoucnu rozhodneme ASP aplikaci přesunout z Windows NT na výkonnější Unix, máme prostě smůlu[8]. [8]

ASP.NET má však k dispozici velmi kvalitní vývojové nástroje na vysoké úrovni, které

[7] http://nodejs.org/

[8] Pro nasazení .NET frameworku na jiné platformy sice existuje projekt MONO (Novell), ovšem nepřináší kýžené vlastnosti a chování a nerozšiřuje aplikovatelnost ASP.NET na srovnatelný počet platforem jako má třeba PHP.

umožňují jednoduše a rychle vybudovat webovou aplikaci. Odděluje funkční kód od (X)HTML a kompiluje kód, což zrychluje provádění operací a kód se stává nečitelným a je prakticky nemožné, aby ho někdo jiný zkopíroval či zneužil. [11]

3.4.4 PHP

Všechny tyto nedostatky a mnohé další odstraňuje systém PHP[9]. Princip použití PHP je obdobný jako u SSJS a ASP. na rozdíl od nich je celý produkt šířen jako freeware – to znamená bezplatně, což ho řadí do stejné oblíbenosti jako např. webový server Apache, operační systém Linux či typografický systém TEX – opět špičky ve svém oboru. [8]

Další výhodou PHP je jeho nezávislost na platformě. Dnes jsou k dispozici verze pro Unix a Windows. PHP není svázáno s žádným konkrétním serverem, může běžet na libovolném. Nejlépe si však dnes rozumí se serverem Apache. [8]

3.4.4.1 Historie

Na počátku zrodu systému stál Rasmus Lerdorf. v roce 1994 ve volném čase vytvořil v Perlu jednoduchý systém pro evidování přístupu k jeho stránkám. Jelikož neustálé spouštění interpretu Perlu velmi zatěžovalo WWW server, přepsal autor systém do jazyka C. [8]

Ačkoliv byl celý systém původně určen pro osobní Rasmusovo použití, zalíbil se i ostatním uživatelům serveru a začali ho používat. Systém se stal oblíbeným a používalo jej stále více uživatelů. Ti přicházeli s požadavky na vylepšení celého systému. Autor proto systém rozšířil, doplnil dokumentaci a uvolnil jej pod názvem Personal Home Page Tools, který se později změnil na Personal Home Page Construction Kit. v téže době autor zprovoznil elektronickou konferenci, která sloužila jako prostor pro výměnu zkušeností mezi uživateli systému. [8]

Kromě již zmíněného systému pro evidování přístupů ke stránkám vytvořil Lerdorf i nástroj, který umožňoval začlenění SQL dotazů do stránek, vytváření formulářů a zobrazování výsledků dotazů. Program, který umožnil zpřístupnění databází na Webu, se jmenoval Form Interpreter (FI)

[9]Obsah zkratky PHP nyní zcela ztrácí na svém původním významu a doporučené označení celého systému je hypertextový preprocesor PHP (PHP: Hypertext Preprocessor). [8]

Celosvětovou proslulost si získal systém PHP/FI 2.0. Tento systém vznikl spojením dvou předchozích programů autora v roce 1997. v této podobě se jednalo o jednoduchý programovací jazyk, který se zapisoval přímo do HTML stránek. PHP/FI 2.0 se rozšířilo opravdu po celém světě a pracovalo i na mnoha českých serverech. [8]

3.4.4.2 Současnost

V současné době podporuje PHP operační systémy Unix, Windows a OS počítačů Macintosh.

Samotné PHP pak může pracovat s libovolným WWW serverem, který umožňuje spouštění CGI skriptů nebo zavedení do paměti jako modu. v případě CGI pracuje PHP jako CGI skript, který zpracovává jednotlivé stránky. Výhodou tohoto řešení je kompatibilita s téměř všemi WWW servery. Konkurenční technologie Microsoftu (ASP) jsou pevně svázány s firemním WWW serverem. v případě zavedení modulem jde o zakompilované PHP, které vkládá server při svém spuštění do paměti. Interpret jazyka PHP je pak stále zaveden v paměti společně s WWW serverem a odpadají tak veškeré režijní náklady spojené s opakovaným spouštěním interpretu PHP (což je nevýhoda CGI).

V současné době běží PHP 5 jak ve 32bitovém tak i v 64bitovém prostředí. Zend Engine, jako jádro pohánějící PHP, byl přepsán do verze II (2004). PHP je v současnosti podporováno většinou webových serverů. [20]

Poslední verzí PHP je verze 5[10].

Všechny výše zmíněné technologie můžeme rozdělit do dvou skupin - podle toho, zda jsou prováděny na serveru nebo na klientovi. Nic však nebrání jejich vzájemné kombinaci, pokud je účelná – např. s klientským JavaScriptem při kontrole údajů v odesílaném formuláři.

3.5 Framework

Framework je softwarová struktura, která slouží jako podpora při programování a vývoji a organizaci jiných softwarových projektů. Může obsahovat podpůrné programy, knihovnu API, návrhové vzory nebo doporučené postupy při vývoji. [9]

[10] v době psaní této práce je poslední stabilní verze 5.5.15

Snad každá platforma pro vývoj webových aplikací poskytuje rozhraní pro oddělení práce s daty od jejich prezentace. v jazyce PHP je však situace jiná, zde neexistuje žádné MVC[11] řešení. Naštěstí jsou napsány desítky frameworků, které nám ulehčují práci při psaní aplikace a pracují s návrhovým vzorem MVC. Použitím některého z nich umožní plné využití výhod MVC architektury a díky vestavěným funkcím také zajistí méně napsaného kódu, a tím pádem rychlejší vývoj aplikací. [2]

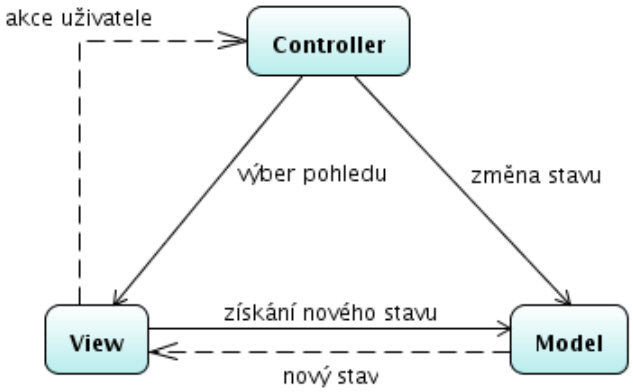
3.5.1 MVC

MVC aneb model-view-controller je návrhový vzor, tedy objektová struktura, jež odděluje data a jejich zpracování od jejich zobrazení. Model představuje datové úložiště, obstarává získání dat a práci s nimi a vrací je controlleru, který si o ně zažádal, controller je následně předá view, který je jakýmsi způsobem zobrazí.

Tento přístup jde mnohem dále za šablony, na druhou stranu je velmi těžké striktní MVC naplnit. Jen tutoriály na frameworky jsou plny porušení základních principů MVC architektury. MVC architekturou získáme přehledný kód, lehce upravitelný a navíc okamžitě přístupnější pro týmovou spolupráci. Podobným, ale téměř nepoužívaným vzorem je MPS – model-presenter-command.

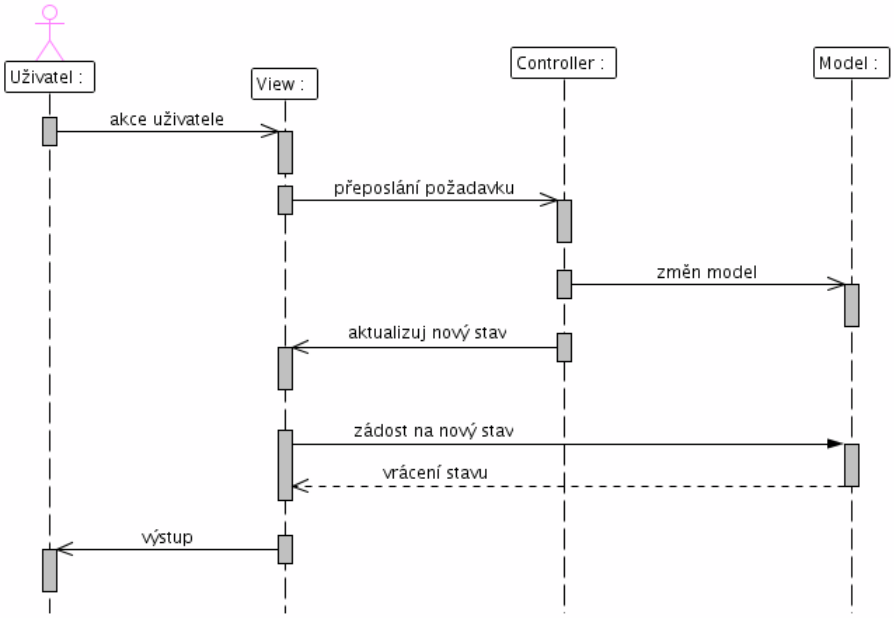
- **Model (model)**, což je doménově specifická reprezentace informací, s nimiž aplikace pracuje.
- **View (pohled)**, který převádí data reprezentovaná modelem do podoby vhodné k interaktivní prezentaci uživateli.
- **Controller (řadič)**, který reaguje na události (typicky pocházející od uživatele) a zajišťuje změny v modelu nebo v pohledu.

[11] Model View Controller



Obrázek 3: Architektura Model-View-Controller, stavový diagram

[6]

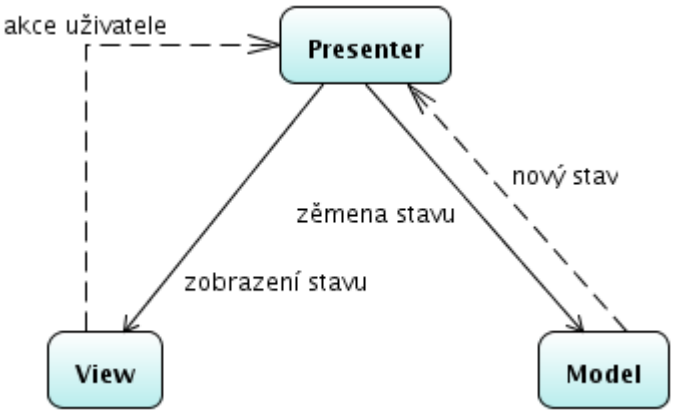


Obrázek 4: Architektura Model-View-Controller, sekvenční diagram

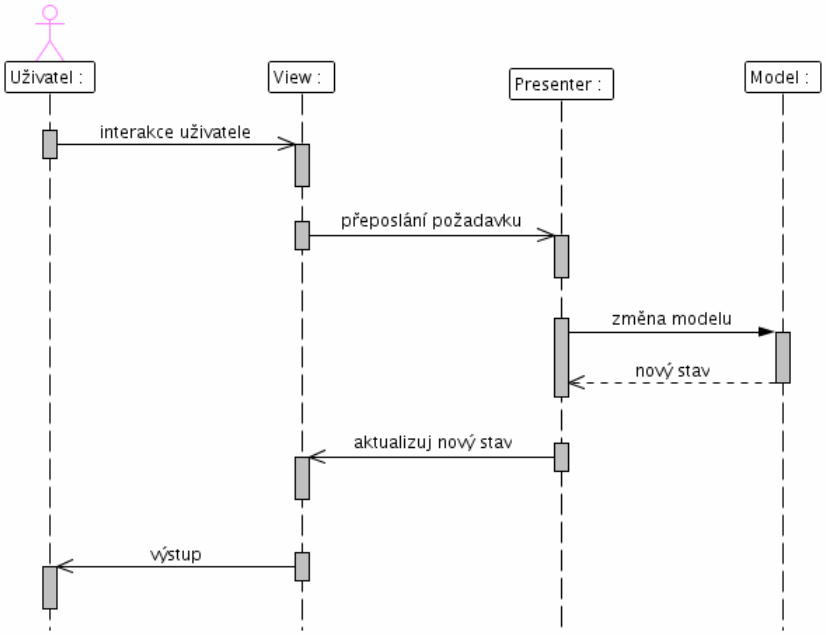
[6]

3.5.2 MVP

MVP (model-view-presenter) vychází z MVC. Rozdělení je podobné - Model reprezentuje data nebo komponenty, které s daty přímo pracují, View reprezentuje výstup informací, vzhled dat z modelu, a Presenter se stará o vyhodnocení požadavku od uživatele. Rozdíl je v pojetí obsluhy - zatímco řadič v MVC je de facto jen jednoduchou sadou pravidel a veškeré úkony jsou provedeny až modelem, u MVP je tomu jinak. Dotazy na aplikaci obsluhuje přímo Presenter, který již spouští konkrétní metody modelu. Podstatnou odlišností od MVC je také to, že pohled ani model o sobě navzájem nevědí. Data z modelu jsou pohledu předána Presenterem. [6]



Obrázek 5: Architektura Model-View-Presenter, stavový diagram [6]



Obrázek 6: Architektura Model-View-Presenter, sekvenční diagram

[6]

3.5.3 Výběr frameworků

3.5.3.1 CakePHP

CakePHP je jeden z nejrozšířenějších frameworků vůbec. Obsahuje velkou softwarovou výbavu a jeví se jako vhodný pro práci na velkých i malých projektech. Při pohledu do výsledků testů dosahuje velmi dobrých výsledků při práci s daty, navíc jeho paměťová náročnost je velmi nízká. [2] [3]

3.5.3.2 Nette

Nette Framework je neznámějším frameworkem české scény. Jeho největšími přednostmi jsou podpora jmenných prostorů, přehledné zpracování chybových hlášení a hlavně kvalitně zpracovaná dokumentace a široká základna uživatelů. Ochrání aplikaci od útoků XSS (Cross-Site Scripting), CSRF (Cross-Site Request Forgery), URL attack, control codes, invalid UTF-8, session hijacking, session fixation a dalších. Jeho časy při čtení a zápisu jsou velmi nízké a využití paměti je také na velmi nízké úrovni. Dále používá vlastní databázovou vrstvu

DIBI, která umožňuje aplikaci běžící na frameworku provozovat na různých typech databází bez velkého zásahu do kódu. [2] [5]

3.5.3.3 Zend

Zend framework je velmi populárním frameworkem a nabízí mnoho zajímavých možností a rozšíření (Zend_Acl, Zend_Mail, Zend_Pdf). Rychlosti přístupu do databáze bez použití akcelerátoru výkonu jsou však propastné. Dle zveřejněných testů pro zobrazení dat potřebuje průměrně čas kolem 1 s, a pro aktualizaci až 1,7 s. Pokud je však na serveru použit eAkcelerator nebo jiný urychlovač, jeho zpracování se několikanásobně zvyšuje stejně jako u ostatních frameworků. Zend však obsahuje spoustu funkcí, které zůstávají malými aplikacemi nevyužity. [2] [17]

3.5.3.4 Symfony

Symfony je dalším poměrně známým frameworkem. Autoři si dávají za cíl vytvoření frameworku, ve kterém by šlo snadno vytvářet robustní enterprise aplikace. Je kladen důraz na to, aby byl framework co nejvíce konfigurovatelný - aby měl vývojář plnou kontrolu nad vším. Symfony pro objektově relační mapování používá frameworků Propel nebo Doctrine. Autoři frameworku se inspirovali frameworkem Ruby on Rails, zejména pak ve způsobu routování URL. Symfony je šířen pod MIT licencí a používá architekturu MVC. [14]

3.5.3.5 Ostatní PHP frameworky

Výše zmíněné frameworky nejsou jediné pro jazyk PHP. Existuje celá řada dalších frameworků. Frameworky mají i různé specializace - framework nemusí být nutně jen sadou kódu, pomocí kterého vytváříme vlastní aplikaci od základu. Frameworky mohou být databázové (ORM)[12] nebo pro správu obsahu (CMS)[13]. Pro zajímavost uvedeme ještě několik dalších frameworků. [6]

[12] Object Relation Mapping

[13] Content Management System

Akelos	PHP 4, PHP 5	MVC
CodeIgniter	PHP 4, PHP 5	MVC
Kohana	PHP 5	MVC
Prado	PHP 5	MVC
Seagull	PHP 4, PHP 5	MVC
Drupal	PHP 4, PHP 5	CMS
Doctrine	PHP 5	ORM
Propel	PHP 5	ORM

Tabulka 1: Tabulka ostatních PHP frameworků

3.6 Databáze

Systém řízení báze dat (SŘBD či DBMS[14]) je softwarové vybavení, které zajišťuje práci s databází, tzn. tvoří rozhraní mezi aplikačními programy a uloženými daty. Občas se pojem zaměňuje s pojmem databázový systém. Databázový systém však je SŘBD dohromady s bází dat. [22]

Vzhledem k použití PHP frameworku lze realizovat projekt na více systémech řízení báze dat. Každý framework obsahuje databázovou vrstvu, která zjednodušuje a zobecňuje komunikaci s databází. Aplikaci samotnou tak již nemusí zajímat, na jakém typu databáze běží, ale pouze využívá API[15] daného frameworku. Zde si tak představíme pouze pár nejznámějších, na kterých bude aplikace testována.

3.6.1 MySQL

MySQL je multiplatformní databázový systém vyvinutý švédskou společností MySQL AB a v současné době patřící firmě Oracle. v dnešní době, s více než 100 miliony stáhnutých nebo jinak rozšířených kopií, je na vrcholu bezplatných databázových programů a je používána drtivou většinou malých a středních projektů. Z historického hlediska je MySQL zaměřena výhradně na rychlost oproti respektování standartu SQL 92. v posledních letech je však tento standart již skoro plně implementován. [15]

[14] Database Management System

[15] Application Programming Interface

Co se týče rychlosti, tak je MySQL optimalizováno pro menší databáze a v porovnání s PostgreSQL dosahuje lepších výsledků. Pokud ovšem tabulka naroste do velikosti GB, je na tom znatelně lépe konkurenční PostgreSQL. Dalším aspektem pro výběr MySQL jsou nástroje pro správu databáze, jež jsou reprezentovány aplikací phpMyAdmin umožňující kompletní přehled o datech, tabulkách a databázích. Existuje obdobný nástroj pro PostgreSQL, pod názvem phpPgAdmin, ale bohužel nedosahuje takových kvalit a neobsahuje stejné množství možností. [4]

3.6.2 PostgreSQL

PostgreSQL je výkonný multiplatformní objektově-relační databázový systém s více než 15 letou historií aktivního vývoje. Popularitu si získal především stabilitou, bezpečností, integritou dat a nulovou cenou.

PostgreSQL je již od začátku svého vývoje zaměřena na plnou implementaci standardu SQL 92. Svým zaměřením a použitím je dosti podobná databázi MySQL, avšak nedosahuje podobných rychlostí při čtení a záznamu. Kvality této databáze se projevují až při daleko větším počtu záznamů. Pro správu jsou k dispozici nástroje psql (příkazová řádka), desktopová aplikace pgAdmin nebo ww rozhraní phpPgAdmin, ovšem nejsou to plně uspokojující aplikace. [4] [13] [1]

3.6.3 MSSQL

Microsoft SQL je relační databázový systém vyvinutý společností Microsoft, využívající dotazovací jazyk SQL a T-SQL. Systém není multiplatformní a je vázán pouze na operační systém Windows. Nutný restart po aplikování aktualizací dále také snižuje garantovanou dostupnost serveru. Z hlediska zátěže procesoru a spotřeby operační paměti na tom také není v porovnání s konkurencí nejlépe. Naopak nástroj Management studio pro správu databází je na velmi vysoké úrovni a poskytuje rychlejší, komplexnější a komfortnější využití než webový phpMyAdmin.

3.6.4 SQLite

SQLite je rychlá malá databáze vhodná pro malé projekty bez nutnosti velkého databázového systému, kterým vystačí pouze aplikační jazyk PHP. SQLite je přímou součástí instalace

jazyka PHP. Ten obsahuje ve své implemetaci i derektivy pro její ovládání.

Na rozdíl od databází založených na principu klient-server, kde je databázový server spuštěn jako samostatný proces, je SQLite pouze malá knihovna, která se přilinkuje k aplikaci a pomocí jednoduchého rozhraní ji lze začít využívat. Každá databáze je uložena v samostatném souboru .dbm (Database Manager), kde se data ukládají za použití jednoduchého primárního klíče do stejně velkých bloků, a používá hašovacích technik pro rychlý přístup k datům při vyhledávání podle klíče. v SQLite je implementován téměř celý standard SQL-92. [10] [7]

3.6.5 Ostatní databázové systémy

Výše zmíněné databázové systémy nejsou jediné, které se dají použít. Existuje celá řada dalších systémů. a ne všechny vycházejí ze standardu SQL 92. Jsou vyvíjeny nové druhy databázových systémů označovaných jako NoSQL, kdy jejich úložiště je orientováno na ukládání dokumentů nebo jsou to velmi rychle klíč-hodnota databáze. Pro zajímavost je uvedeno několik dalších databází.

Oracle	SQL
Firebird	SQL
MariaDB	SQL
MongoDB	NoSQL
MS Access	SQL
CouchDB	NoSQL

Tabulka 2: Tabulka ostatních databázových systémů

3.7 Zpracování v reálném čase

Programy pracující v reálném čase musejí zajistit striktní odpověď během nějakého časového omezení. Odpověď v reálném čase je tak často chápána jako odpověď v řádu milisekund nebo i mikrosekund. a systém dokáže tuto dobu odpověď garantovat. na druhé straně systémy, které nejsou označovány za pracující v reálném, nedokáží odpověď na požadavek v určitém

krátkém časovém intervalu garantovat v žádné situaci, i když jejich obvyklá odpověď na požadavek je velmi rychlá.

Použití pojmu reálného času má tak pro systémy jiný význam než právě při simulaci, kdy je tento pojem často využíván jako pracující synchronně s reálným hodinovým signálem. Avšak v doméně přenosu dat, zpracování médií a počítačových systémů nabývá tento pojem významu „bez citelného zpoždění“. [21]

3.7.1 Kritéria zpracování v reálném čase

U systému, který pracuje v reálném čase, celková správnost výsledku nezáleží jenom na logické správnosti výsledku, ale také na čase, v jakém je výsledek doručen. Systém tak musí doručit odpověď do uplynutí určitého časového intervalu. Podle toho, jak jsou systémy schopny tento interval dodržet, je můžeme také rozdělit:

Hard – nedodržení časového omezení je úplná systémová chyba

Firm – občasné nedodržení časového omezení je tolerováno, ale může snižovat kvalitu služby (QoS), užitná hodnota výsledku je však po překročení omezení nulová

Soft – užitnost výsledku po překročení intervalu degraduje a tím i snižuje kvalitu poskytovaných služeb (QoS)

Cílem systémů pracujících v reálném čase je tak doručení všech odpovědí v určitém časovém omezení a bez významného zpoždění (latence) nebo alespoň doručení maximálního množství odpovědí s minimální latencí. [21]

3.8 Specifika závodů

Pokud má být systém dostatečně obecný, musí se umět vypořádat s různými druhy závodů a jejich specifiky. v první řadě jsou to závody skupin či jednotlivců nebo dohromady. Dále jsou to různá rozdělení v závodech do výkonnostních, věkových či jiných kategorií. Dále je potřeba také brát v úvahu různá stanoviště, kontrolní body, zastávky či jenom průběžná měření, která se mohou závod od závodu lišit a na kterých mohou být sbírána různá data. Příkladem tak může být závod ve vodním slalomu, kde stanovišti jsou jednotlivé branky. Při špatném projetí tak závodník dostává časovou penalizaci, kterou je potřeba promítnout do výsledného času. Jinak je pak potřeba zpracovávat údaje z orientačního závodu, kdy jsou

na jednotlivých stanovištích získávány body a pořadí se pak určuje jak podle bodů, tak i času. Specifika jednotlivých závodů je třeba detailně prostudovat a na základě studia natolik zobecnit, aby se dala implementovat do systému.

3.9 Mobilní zařízení

U většiny závodů se dá předpokládat, že minimálně start a cíl nejsou na stejném místě, případně závod obsahuje různá stanoviště, kde se získávají data, která jsou potřeba zadat do systému pro maximální možné urychlení závodu. Vzhledem k tomu, že systém běží na webu a internetové připojení pro přenosné počítače není všudypřítomné, tak dostávají do popředí jiná mobilní zařízení, jako jsou různé mobilní telefony a komunikátory. Otevírají se tak nové možnosti jak realizovat přenos dat do systému a to díky využití mobilního internetu. Prakticky tak lze vytvořit speciální aplikaci pro mobilní telefony, která bude zajišťovat předávání těchto dat do systému anebo celý informační systém či jeho část optimalizovat pro práci s mobilními zařízeními.

3.9.1 Využití mobilních zařízení

Komunikace v průběhu závodu bude z části prováděna také pomocí mobilních zařízení, kterými jsou hlavně mobilní telefony. Vzhledem k rozšířenosti internetu tak bude pro práci se systémem využíván tenký klient (internetový prohlížeč) mobilního zařízení. Pro tyto situace bude systém umět zařízení detekovat a mobilní zařízení tak bude využívat speciálně upravené stránky pro práci s těmito zařízeními. Jedná se hlavně o úpravy ze stran kaskádových stylů a minimalizaci přenášených dat. Je počítáno s tím, že mobilní aplikace budou využívány zejména na stanovištích a kontrolách závodů, kde bude potřeba odesílat data pouze několika málo formuláři, které budou zabezpečené autentifikací.

4 Vlastní práce

4.1 Analýza

Tato část práce je zaměřena na analýzu současného stavu a na přípravu podkladů k samotné realizaci systému. Rozložením celkového problému na podproblémy můžeme detailněji specifikovat postup vývoje systému a vyhnout se tak závažným chybám. Samotná analýza je tak hlavně zaměřena na běhové prostředí a použití programovacích jazyků a systému řízení báze dat, kdy je kladen důraz na zachování multiplatformity.

4.1.1 Požadavky na systém

4.1.1.1 Přihlašování a registrace

1. Přihlašování (vstupní strana)
   (a) přihlašovací jméno
   (b) heslo
   (c) přihlášení přes SkautIS
   (d) zaregistrovat se (veřejnost) - přihlašovat nové posádky přes internet je možné od 1. 5. do 23. 9.
   (e) přihlašování přes internet znamená slevu na startovném, pokud se budete hlásit až na startu, použijte papírovou přihlášku
   (f) odkaz na stažení papírové přihlášky
   (g) reset zapomenutého hesla
2. Registrace pro přihlašování (veřejnost)
   (a) přihlašující osoba
   (b) jméno
   (c) příjmení
   (d) přezdívka
   (e) mobil (ve formátu 123456789)

(f) adresa (ve formátu Vodácká 39, 11000 Praha 1)

(g) login

(h) heslo

(i) e-mail

(j) na uvedený e-mail bude zaslána konfirmace aktivující účet

3. Přihlášky (podané přihlášky)

(a) nová přihláška (po 23. 9. nelze již přidávat další posádky přes internet, použijte papírovou přihlášku)

(b) registrační číslo

(c) počet posádek

(d) startovné

(e) upravit

(f) smazat

(g) vygenerování variabilního symbolu

(h) žádost o vrácení startovného za posádku

4. Registrace posádek

(a) přihlašovat posádky přes internet lze až do 18:00 23. 9., pak použijte papírovou přihlášku, kterou odevzdáte na startu

(b) složení posádek lze upravovat až do 18:00 27. 9.

(c) přihláška každé posádky musí být podepsána odpovědnou osobou, pokud nebude na startu, přihlášku vytiskněte a přineste ji k registraci již podepsanou

(d) přidat neskautskou posádku

(e) přidat skautskou posádku

(f) registrační číslo

(g) kategorie

(h) popis

(i) upravit

(j) kopírovat

(k) tisk

(l) smazat

4.1.1.2 Formulář přihlášky [16]

1. Kategorie

   (a) R - dorostenci 1995 - 97 - loď P550 - třídová loď VS

   (b) F - dorostenky 1995 - 97 - loď P550 - třídová loď VS

   (c) S - starší žáci 1998 a mladší - loď P550 - třídová loď VS

   (d) G - starší žákyně 1998 a mladší - loď P550 - třídová loď VS

   (e) B - mladší žactvo 2002 a mladší - loď P550 - třídová loď VS

   (f) k - žactvo 1998 a mladší - P530 - pramice do 530 cm

   (g) D - žactvo 1998 a mladší - P650 - rychlostní pramice

   (h) C9 - dospělí - C9 - devítikánoe

   (i) C2 - dospělí - C2 - kánoe

   (j) O - dospělí - O - open - libovolná pramice

   (k) v - libovolná posádka mimo závod - Splutí - libovolné plavidlo

2. Typ lodi

   (a) P550 - třídová loď VS

   (b) P530 - pramice do 530 cm

   (c) P650 - rychlostní pramice

   (d) C9 - devítikánoe

   (e) C2 - kánoe

[16] Vychází ze současně používaného formuláře pro přihlašování na Google Dokumentech: https://docs.google.com/spreadsheet/viewform?formkey=dFc4S0dtV1diVXVfT3oxRno4anRYZFE6MQ

(f) O - open - libovolná pramice

(g) Splutí - libovolné plavidlo

4.1.1.3 Jiné údaje

1. Vysílající skautská jednotka[17]

   (a) číslo jednotky ve formátu XXX.YY.ZZZ, kde XXX je číslo obvodu, YY je číslo střediska a ZZZ číslo oddílu. Viz registr jednotek

   (b) název vysílacího přístavu (střediska)

   (c) jméno oddílu

   (d) místo působení oddílu

   (e) odpovědná osoba přítomná na závodě

   (f) mobil ve formátu 123456789

   (g) kontaktní adresa

2. Vysílající neskautský oddíl[18]

   (a) jméno oddílu

   (b) adresa oddílu

   (c) místo působení

   (d) odpovědná osoba přítomná na závodě

   (e) mobil ve formátu 123456789

3. Složení posádky

   (a) jméno

   (b) příjmení

   (c) přezdívka

   (d) datum narození ve formátu dd.mm.rrrr

[17] Příloha C

[18] Příloha D

(e) host (označení hostujícího člena)

4. Exporty

   (a) diplomy

   (b) přihlášky

   (c) seznamy lodí v dané kategorii

   (d) cílové listiny

5. Server

   (a) lebeda.skauting.cz

   (b) Red Hat Enterprise Linux (RHEL)

   (c) Apache 2.4.9

   (d) Mysql 5.6.19-67.0

   (e) rozšíření PHP: mysqli

   (f) PHP 5.4.32

4.2 Návrh běhového prostředí

Požadavkem na informační systém je použitelnost na co možná největším počtu platforem. Systém tak musí být spustitelný jak pod OS Windows, tak i OS Unix/Linux (různé distribuce) a dalšími systémy. Musíme tak dbát na výběr programovacího jazyka pomocí, kterého bude systém realizován i na výběr systému řízení báze dat. Oboje tak musí být k dispozici na co možná nejširší škále operačních serverových systémů.

4.2.1 Aplikační jazyk

PHP

Vzhledem k výše zmíněným problémům, požadavkům a možnostem objednatele byl jako aplikační jazyk zvolen jazyk PHP. v první řadě řeší problémy s multiplatformitou (je dostupný na všech operačních systémech/serverech), dále je jako jediný možný aplikační jazyk dostupný na hostingovém serveru lebeda.skauting.cz a z hodnocení výše zmíněných jazyků se pro tuto aplikaci hodí nejvíce jak vlastnostmi, tak i rychlostí.

4.2.2 Framework

Nette

Jako aplikační framework byl dle dostupných informací, porovnání a testů frameworků pro jazyk PHP zvolen právě Nette. Tento framework je nejen srovnatelný s velkými zahraničními frameworky, ale místy je i předčí. Pro naši aplikaci a použití se výborně hodí i díky české základně, bohaté dokumentaci a aktivní komunitě. Dále také díky strmé křivce učení umožní vytvořit základ aplikace velmi rychle.

4.2.3 Databázová vrstva

MySQL

Na produkčním serveru bude jako systém řízení báze dat použit MySql. Vzhledem k použitému frameworku, který již obsahuje ORM (Object Relation Mapping) nás tento systém nijak nelimituje. Díky ORM je v podstatě jedno, jestli daná databáze bude Postgre či MSSQL či jiné. Podstatnou výhodou MySql je však její rychlost zpracování transakcí, což umožní aplikaci dodávat výsledky opravdu reálném čase a bez zbytečného blokování.

4.2.4 Operační systém

Unix/Linux

Daný operační systém opět není omezením vzhledem k multiplatformitě celého projektu. i přes to, že aplikace bude vyvíjena na Windows/IIS, tak s nasazením na Linux/Apache by neměl nastat žádný vážnější problém.

4.2.5 Konfigurace serveru

Celkové závěry vedou k následujícím možným konfiguracím serveru:

4.2.5.1 Vývojové prostředí

- OS: Windows 8.1 Pro
- server: IIS 8.0
- aplikační jazyk: PHP 5.4
- framework: Nette
- databáze: MySQL 5.6

4.2.5.2 Provozní prostředí

- OS: Unix/Linux RedHat
- server: Apache 2.4.9
- aplikační jazyk: PHP 5.4
- framwork: Nette
- databáze: MySQL 5.6

4.3 Vývoj informačního systému

4.3.1 Návrh adresářové struktury

Výchozí adresářová struktura je dána požadavky frameworku a přestože lze toto nastavení změnit, většinou pro to není důvod. Aplikační část je uložena v adresáři app, veškeré knihovny v adresáři vendor a v adresářích www/css a www/js jsou uloženy soubory javascriptu a CSS nastavení. Jediným souborem v nejvyšší urovni je pak www/index.php. Veškeré aplikační soubory jsou uloženy tedy v adresáři app, společně se souborem bootstrap, kde je umístěno spuštění celé aplikace a kterou spouští zmíněný soubor index.php. Dále jsou v tomto adresáři adresáře jednotlivých modulů a v každém z nich dále adresáře pro controllery, view, modely a ukládání cache.

```
public-html/              <- kořenový adresář webu
+-- reg3jezy/
    +-- app/              <- adresář webové aplikace
    |   +-- config/       <- konfigurační soubory
    |   +-- model/        <- třídy modelové vrstvy
    |   +-- presenters/   <- třídy presenterů
    |   +-- router/       <- konfigurace URL adres
    |   +-- templates/    <- šablony
    |   +-- bootstrap.php <- spouštěcí soubor
    |
    +-- vendor/           <- knihovny pro vaší aplikaci
    |   +-- nette/        <- framework a~jiné knihovny
    |
    +-- log/              <- chybové logy
    +-- temp/             <- dočasné soubory (cache, sessiony, atd.)
    +-- test/             <- adresář pro unit testy
    +-- www/              <- místní kořenový adresář webu
```

Tabulka 3: Tabulka adresářové struktury

4.3.2 Návrh databázové struktury

Návrh databáze a její struktury vychází z analýzy požadavků na systém a entit v systému se vyskytujících. Výsledek této analýzy je pak zobrazen pomocí ER diagramu (Entity Relationship). Diagram je uveden jako příloha B této práce. Samotná struktura databáze tak bude vycházet přímo z tohoto diagramu.

4.3.3 Návrh objektového modelu

Návrh objektového modelu stejně jako návrh databázové struktury vychází z analýzy požadavků na systém a z entit v systému se vyskytujících. Samotné entity jsou následně reprezentovány interními systémovými objekty. Tyto objekty jsou vytvářeny tak, aby co nejvěrněji kopírovali chování systémových entit, jejich stavů a funkcí. Objekty spolu pak komunikují pomocí zasílání zpráv.

4.3.4 Postupu prací a milníky projektu

Na základě analýzy požadavků na informační systém byl stanoven následující postup prací a projekt tak byl rozdělen do několika milníků. Milníky umožňují dekompozici projektu do několika dílčích důležitých částí, které lze realizovat postupně a odděleně. Milníky samotné jsou pak ještě dále děleny na jednotlivé úkoly s konkrétním zadáním realizace té které části projektu.

4.3.4.1 Milníky:

1. Zaregistrování a přihlášení do systému
    - systémová registrace a přihlášení
    - přihlášení SkautIS účtem
    - odhlašování
    - potvrzení registrace a aktivace účtu
    - ztráta hesla a obnovení účtu
2. Přihláška do závodu
    - proces registrace posádky
    - revize podaných přihlášek
    - odevzdávání registrace
    - fakturace a platby

- informační maily

3. Členové týmu
    - správa členů týmu (přidávání, odebírání, editace)
    - import členů týmů z jiných systémů
4. Kontaktní údaje
    - úprava kontaktních údajů
5. Administrace
    (a) nastavení systému
    (b) nastavení závodu
    (c) kategorie
    (d) uživatelé
    (e) oprávnění
6. Výsledky
    - zobrazovaní výsledků
7. Exporty
    - výsledky
    - diplomy
    - seznamy
    - výsledné časy

Kompletní seznam i se zadáním úkolů je rozepsán na GitHubu v sekci isssues[19].

4.3.5 Zálohování a verzování

Vývoj systému je potřeba v průběhu prací postupně zálohovat a držet informace o jednotlivých verzích. k tomuto účelu se nejvíce hodí nástroj s názvem Git[20] - distribuovaný systém správy verzí vytvořený Linusem Torvaldsem.

Samotný vzdálený repositář, resp. origin, je hostován na veřejném git serveru GitHub[21].

Díky tomuto nástroji je vývoj aplikace možný prakticky odkudkoli, kde je přístup k internetu. Navíc systém umožňuje spravovat jednotlivé verze i rozdělovat vývoj do několika větví a členit tak vývoj dle důležitosti a potřeb.

[19] https://github.com/literat/3jezy-registrace/issues

[20] http://git-scm.com/

[21] https://github.com/; https://github.com/literat/3jezy-registrace

4.4 Testování

Testování je empirický technický výzkum kvality testovaného produktu či služby, v tomto případě softwaru. Cílem testů je ověřit správnou funkčnost a odhalit chyby.

Cílem testování tohoto informačního systému je ověření jeho správné funkčnosti na několika různých platformách. Druhotně pak bezchybnost prováděných operací a udržování správného stavu.

4.4.1 Konfigurace testovacího prostředí

1. varianta (vývojové prostředí)
    - OS: Windows 8.1 Pro
    - server: IIS 8.0
    - aplikační jazyk: PHP 5.4
    - framework: Nette
    - databáze: MySQL
2. varianta (testovací prostředí)
    - OS: Windows 8.1 Pro
    - server: IIS 8.0
    - aplikační jazyk: PHP 5.4
    - framework: Nette
    - databáze: MSSQL
3. varianta (testovací prostředí)
    - OS: Unix/Linux Debian
    - server: Apache 2.4.9
    - aplikační jazyk: PHP 5.4
    - framwork: Nette
    - databáze: PostgreSQL
4. varianta (preprodukční prostředí)
    - OS: Unix/Linux Debian
    - server: Apache 2.4.9
    - aplikační jazyk: PHP 5.4
    - framework: Nette
    - databáze: MySQL

4.4.1.1 Konfigurace jednotlivých testů

1. konfigurace: linux + php + mysql
2. konfigurace: linux + php + postgres
3. konfigurace: windows + php + mysql
4. konfigurace: windows + php + postgres
5. konfigurace: windows + php + mssql

4.4.2 Závěry z testování

Při testování na jednotlivých konfiguracích a platformách bylo zjištěno, že testovaný systém není závislý na platformě. Je tedy jedno, jestli bude systém v budoucnu spouštěn na platformě Windows, Linux/Unix či jiné.

Závislost informačního systému je tak pouze na aplikačním jazyku PHP, který však lze bez problémů na výše zmíněných platformách spouštět.

Rozdíl se však již projevuje při použití rozdílného webového serveru. Avšak pouze v případě konfigurace serveru pomocí externích konfiguračních souborů (.htaccess v případě Apache a web.config v případě IIS). Rozdíl je znát nejen z názvu obou souborů, ale i z použité syntaxe (XML v případě IIS). Samotné servery pak přistupují rozdílně i ke konfiguračním direktivám (př. redirect, nice url).

Dalším, a již znatelným rozdílem jsou použité databáze. v případě MySQL a PostgreSQL jsou rozdíly nepatrné a znatelné by byly až při velmi vysoké zátěži. v případě MSSQL už je posun k jiné databázi znatelnější. Příkladem může sloužit direktiva LIMIT, která omezuje počet řádků výsledku, který databáze vrací. Touto direktivou MSSQL nedisponuje. Bylo tedy potřeba upravit některé dotazy přímo v aplikaci.

Hlavním závěrem je, že informační systém může být provozován na různorodých platformách a to pouze s minimálními modifikacemi v kódu aplikace samotné.

4.4.2.1 Uživatelské testování

Testování systému uživateli má za cíl ověřit funkčnost a ovládání navrženého uživatelského rozhraní, hlavně pak jeho použitelnost. Princip spočívá v tom, že toto testování pozoruje chování samotných uživatelů, čímž může odhalit chyby, které zůstaly vývojářům skryté.

Uživatelské testování tak může určovat správný směr při vývoji samotné aplikace a předcházet tak vznikajícím problémům.

Uživatelské testování informačního systému registrace na akci Přes 3 jezy je plánováno na duben roku 2015.

4.5 Reálný provoz

V době realizace této práce není registrační systém ještě nasazen do plného provozu. Informační systém je spuštěn pouze na preprodukčním prostředí, které však 1:1 odpovídá prostředí produkčnímu. Předpokladem je zde odzkoušení chování systému v provozu a prostředí blízkém tomu reálnému.

Preprodukční prostředí, resp. registrační systém lze nalézt na adrese https://reg3jez.skauting.cz/.

Produkční prostředí, resp. samotný běžící informační systém určení uživatelům bude běžet na adrese https://reg3jezy.hkvs.cz/.

4.5.1 Nasazení aplikace do provozu

Registrační systém poběží v produkci na předpřipravené webhostingovém účtu. Přístup k tomuto účtu a k samotným souborům aplikace je však realizován pomocí FTP[22]. v současné době je tato metoda přístupu velmi oblíbená a jednoduchá, avšak pro nasazování celé aplikace nebo jejích novějších verzí nevhodná. a to nejen díky nutnosti manuálního nahrání většiny souborů, ale také kvůli množství souborů.

Tyto problémy bylo potřeba vyřešit automatizovanou správou a nasazením do produkce. Vzhledem k verzování v systému Git připadalo v úvahu automatizované nahrání do produkce na základě změny repozitáře. Toto řešení však v zásadě Git úplně jednoduše neumožňuje. Avšak existují pro něj nástroje.

4.5.1.1 Git FTP

Git FTP[23] je nástroj, který rozšiřuje klasický Git shell o příkaz ftp. na základě zadané konfigurace lze pak na server automaticky nahrát celou aplikaci (*git ftp init*) anebo její část,

[22] File Transfer Protocol - http://cs.wikipedia.org/wiki/File_Transfer_Protocol

[23] https://github.com/git-ftp/git-ftp

která je již verzována (*git ftp push*).

4.5.1.2 FTP Deployment

FTP Deployment[24] je nástroj vyvíjený předním českých PHP vývojářem Davidem Grudlem. Podobně jako Git FTP i FTP Deployment využívá informace z verzovacího systému Git. Nicméně samotný automatizace procesu nahrání souborů je v režii PHP skriptu pouštěného z příkazové řádky. Jediným parametrem skriptu je pak pouze konfigurační soubor (deployment.ini), kde jsou definovány všechna potřebná nastavení pro nahrání do produkce.

4.5.1.3 FTP Deploy služby

Existuje několik webových služeb, které se zabývají automatizací procesu nahrání souborů aplikace přes FTP z Git repozitáře na server. Nevýhodou těchto služeb je, že poskytujeme přístup k serveru třetí straně. Dále se jedná většinou buď o placené služby, nebo ve verzích zdarma omezené na jednotky projektů. Jako příklad mohu uvést FTPloy[25], dploy.io[26], Codeship[27], DeployHQ[28] a jiné.

Jako nejvhodnější řešení pro nasazení aplikace do produkce byl v tomto případě zvolen nástroj FTP Deployment, které umožňuje nejlepší konfiguraci a nejrychlejší nahrávání. Navíc v případě nahrávání dokáže korektně na danou chvíli odstavit aplikaci a vracet chybu 503 (Služba je dočasně nedostupná), případně přesměrovat na statickou informační stránku.

[24] http://phpfashion.com/ftp-deployment-nahravejte-pres-ftp-chytre; https://github.com/dg/ftp-deployment
[25] https://ftploy.com/
[26] http://dploy.io/
[27] https://codeship.com/
[28] https://www.deployhq.com/

5 Zhodnocení výsledků

Informační systém, respektive aplikace, která je stěžejním předmětem této práce, je poměrně velmi komplexním systémem. Takový systém tak nelze realizovat bez předchozích příprav a konzultací. Vlastní část této práce se tak primárně zabývala analýzou problému daného informačního systému, dekompozicí jednotlivých problémů, ujasňování vlastností systémů a konkretizací procesů. Tato přípravná práce tak zabrala velké množství času a úsilí. Výstupem však jsou kvalitní podklady a konkrétní zadání pro jednotlivé části a vlastnosti systému. Zadavatel i vývojář vědí, co a jakým způsobem má systém dělat a jak se má chovat. Díky kvalitním podkladům a upřesněnému zadání tak bylo možno vybrat vhodný aplikační jazyk a platformu, vybrat aplikační framework, navrhnout vhodnou strukturu systému řízení báze dat i objektový model, v poměrně krátkém čase.

V začátcích realizace jakékoli aplikace se tedy velmi vyplatí zamyslet se nad celkovou problematikou a rozložit si ji na menší celky a i ty následně analyzovat. v průběhu vývoje je pak k dispozici konkrétní zadání, které už se do upravuje jenom v menších detailech, a je možné se tak soustředit na celek. Takové zadání pak i podkladem pro testování.

V průběhu analytické části bylo zjištěno, že je zadavatel omezen pouze na platformu Linux s webovým serverem Apache, aplikační jazyk PHP a databázi MySQL. v rámci práce však bylo i zjišťováno, jestli by jiný aplikační jazyk či databáze nepřinesla jiné výhody a vlastnosti, které by stály za zvážení a možnost realizovat systém na jiné platformě. Pro řešení daného problému se však ukázalo, že daná konfigurace je nejlepší a nejlevnější možná. a vzhledem k multiplatformitě jak PHP, tak i MySQL není systém omezen jenom na platformu Linux, ale může být jednoduše portován i na další servery.

Pro zjednodušení a urychlení vývoje bylo uvažováno i použití aplikačního frameworku. Vzhledem k rozšířenosti jazyka PHP bylo k výběru frameworků hned několik. v potaz však byly brány jenom pouze ty s jednoduchou a jasnou strukturou, dobrou dokumentací a aktivní podporou. To vše se sešlo u frameworku Nette, u kterého byla plusem česká dokumentace i komunita.

Framework samotný pak v případě vývoje opravdu urychluje a zjednodušuje postup prací a realizaci jednotlivých komponent a stanovuje jasnou strukturu projektu. Díky dobré učící křivce se průběh vývoje systému postupně zrychluje.

Toto positivum frameworku je však také zároveň jeho negativem. V případě implementace řešení u velmi komplexních aplikací principy a komponenty frameworku již přestávají stačit. Programátor pak musí framework obcházet nebo jít přímo proti jeho principům.

Součástí vývoje informačního systému bylo i jeho uživatelské testování. Těch bylo v menším i větším měřítku naplánováno několik a byla realizována v průběhu vývoje systému. Každé z testování vždy přineslo pozitivní i negativní ohlasy na systém jako takový. Důležitá však byla zpětná vazba od uživatelů, kteří systém budou používat. Ta umožnila vždy zlepšit uživatelské rozhraní či chování systému. Navíc díky průběžnému testování byly chyby odhalovány mnohem dříve a i jednotlivé úpravy mohly být začleňovány do plánu již v průběhu vývoje a ne na konci. Tato situace tak významně přispěla ke komfortu celé realizace a ke kvalitnějšímu výslednému produktu.

Informační systém pro registraci na závod Přes 3 jezy tak v zásadě splňuje vytyčené cíle. Není však kompletně realizován se všemi výše zmíněnými vlastnostmi a funkcionalitami, avšak zadání i procesy pro jeho další vývoj a rozšiřování jsou dobře nastaveny i do budoucna.

6 Závěr

V rámci této práce realizovaný informační systém je příliš komplexní a rozsáhlý, aby bylo možné ho začít vyvíjet bez předchozích znalostí o sportovní akci nebo bez konzultací se zadavatelem. Tato práce dokazuje, že se vyplatí věnovat dostatečné množství času přípravě zadání a dokumentaci vlastností a chování systému. Kvalitní příprava poté urychluje veškeré další fáze životního cyklu projektu a vývoje systému samotného. A za použití dalších podpůrných prostředků, jako jsou třeba aplikační frameworky, lze pak informační systém stavět velmi rychle.

I přes dobrou přípravu a podpůrné prostředky nelze takto komplexní systém vyvinout ihned. V průběhu této práce byl tak připraven kvalitní základ systému, který bude v budoucnu dále vyvíjen a rozšiřován. V případě úspěšného pilotního provozu v roce 2015 by se měl systém stát integrální součástí registrace účastníků do závodu a vyhodnocování výsledků závodu. Závodníkům zpříjemní registraci a organizátorům usnadní hodně práce.

7 Seznam použitých zdrojů

Literatura

[1] Czech, C.; Group, S. P. U.: PostgreSQL. 2014-06-06, [online].
URL http://postgres.cz/wiki/PostgreSQL

[2] Daněk, P.: Velký test PHP frameworků. 2008-08-21, [online].
URL http://www.root.cz/clanky/velky-test-php-frameworku-2008/

[3] Foundation, C. S.: Cake PHP. 2014-08-18, [online].
URL http://cakephp.org/

[4] Gilfillan, I.: PostgreSQL vs MySQL: Which is better? 2003-12-16, [online].
URL http://www.databasejournal.com/features/postgresql/article.php/3288951/PostgreSQL-vs-MySQL-Which-is-better.htm

[5] Grudl, D.: Nette Framework. 2014-08-18, [online].
URL http://nette.org/

[6] Havlín, Z.: *PHP frameworky*. Diplomová práce, Česká zemědělská univerzita, 2009.

[7] Hipp, D. R.: SQLite: About. 2014-08-18, [online].
URL http://www.sqlite.org/about.html

[8] Kosek, J.: *PHP, tvorba interaktivních internetových aplikací*. Praha: Grada Publishing a.s., 7 vydání, 1997, ISBN 80-7169-373-1, 17–24 s.

[9] Škrášek, J.: PHP frameworky. 2008-02-21, [online].
URL http://programujte.com/clanek/2008022000-php-frameworky/

[10] Lebeda, M.: SQLite - ultra lehké SQL. 2003-08-04, [online].
URL http://www.root.cz/clanky/sqlite-ultra-lehke-sql/

[11] Lee, W.-M.: What Is ASP.NET. 2005-09-19, [online].
URL http://ondotnet.com/pub/a/dotnet/2005/09/19/what-is-asp-net.html

[12] Peterka, J.: Sága rodů LAN a WAN. 1197-08-01, [online].
URL http://www.earchiv.cz/a708s600/a708s684.php3

[13] PostgreSQL: Why PostgreSQL Instead of MySQL 2009. 2009, [online].
URL http://wiki.postgresql.org/wiki/Why_PostgreSQL_Instead_of_MySQL_2009

[14] Potencier, F.: About Symphony Web PHP Framework. 2014-08-18, [online].
URL http://symfony.com/about

[15] Schneider, R. D.: *MySQL, Oficiální průvodce tvorbou, správou a laděním databází*. Praha: Grada Publishing a.s., 2006, ISBN 80-247-1516-3.

[16] Sharon Zakhour, a. k.: *Java 6 Výukový kurz*. Brno: Computer Press, a.s., 2007, ISBN 978-80-251-1575-6.

[17] Technologies, Z.: Zend Framework 2: About. 2014-08-18, [online].
URL http://framework.zend.com/about/om/about

[18] W. Choi, C. L. G. P. C. U., A. Kent: *Beginning PHP 4*. New York: Wrox Press Ltd., 2000, ISBN 1861003730.

[19] Wikipedia: Klient (počítače). 2014, [online].
URL http://cs.wikipedia.org/wiki/Klient_(po%C4%8D%C3%ADta%C4%8De)

[20] Wikipedia: PHP. 2014, [online].
URL http://en.wikipedia.org/wiki/PHP

[21] Wikipedia: Real-time computing. 2014, [online].
URL http://en.wikipedia.org/wiki/Real-time_computing

[22] Wikipedia: Systém řízení báze dat. 2014, [online].
URL http://cs.wikipedia.org/wiki/Syst%C3%A9m_%C5%99%C3%ADzen%C3%AD_b%C3%A1ze_dat

[23] Wikipedia: Webový prohlížeč. 2014, [online].
URL http://cs.wikipedia.org/wiki/Webov%C3%BD_prohl%C3%AD%C5%BEe%C4%8D

[24] xsenj01: Klient - Server: Komunikace. 2015-01-27, [online].
URL http://www.gjk.cz/~xsenj01/klient%20-%20server.htm

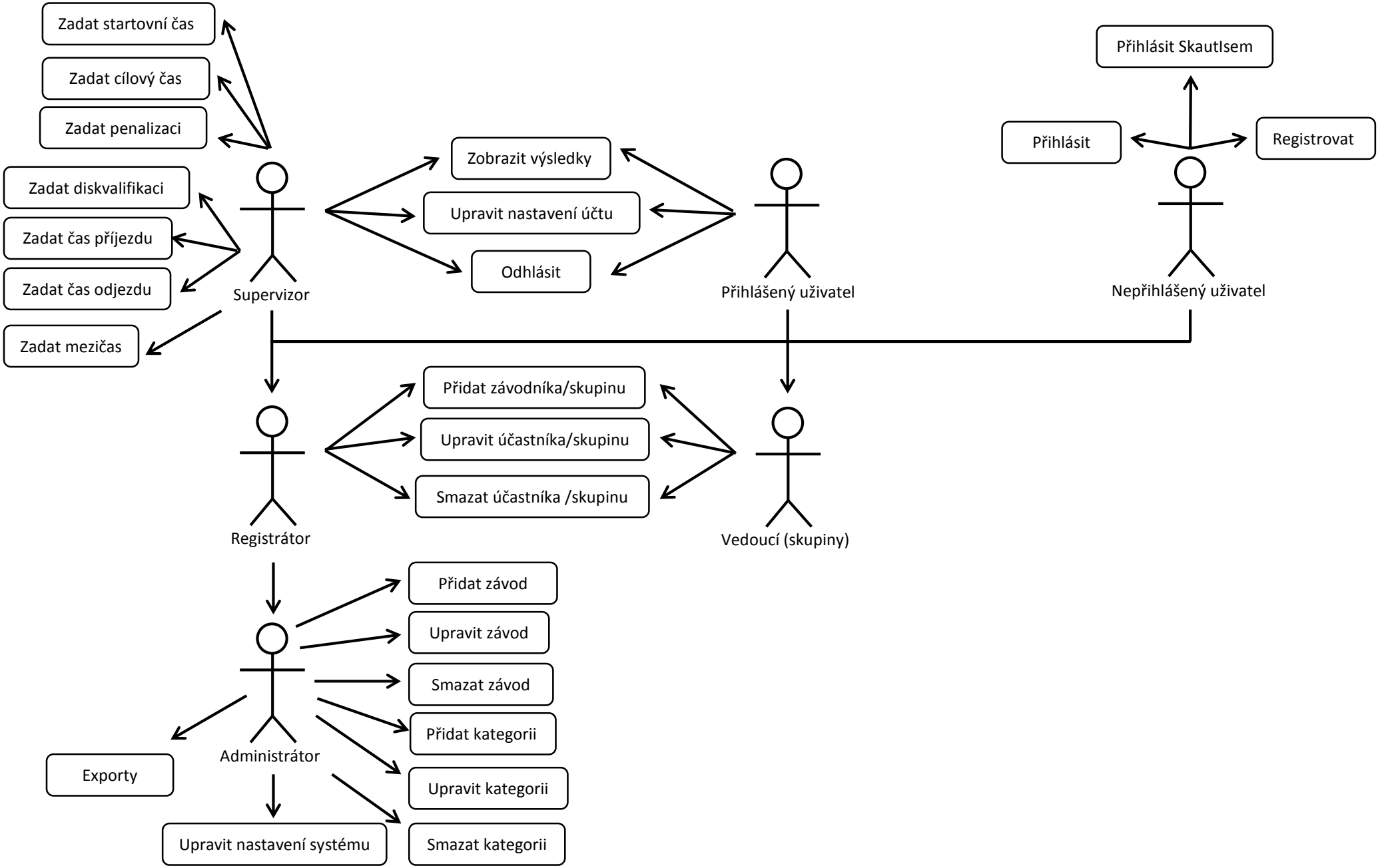
8 Přílohy

Seznam příloh:

- Příloha A: Kompletní Use Case diagram
- Příloha B: Kompletní Entity Relationship diagram
- Příloha C: Vzor papírové přihlášky na závod Přes 3 pražské jezy (skautská část)
- Příloha D: Vzor papírové přihlášky na závod Přes 3 pražské jezy (veřejná část)
- Příloha E: Testovací přístupové údaje do systému

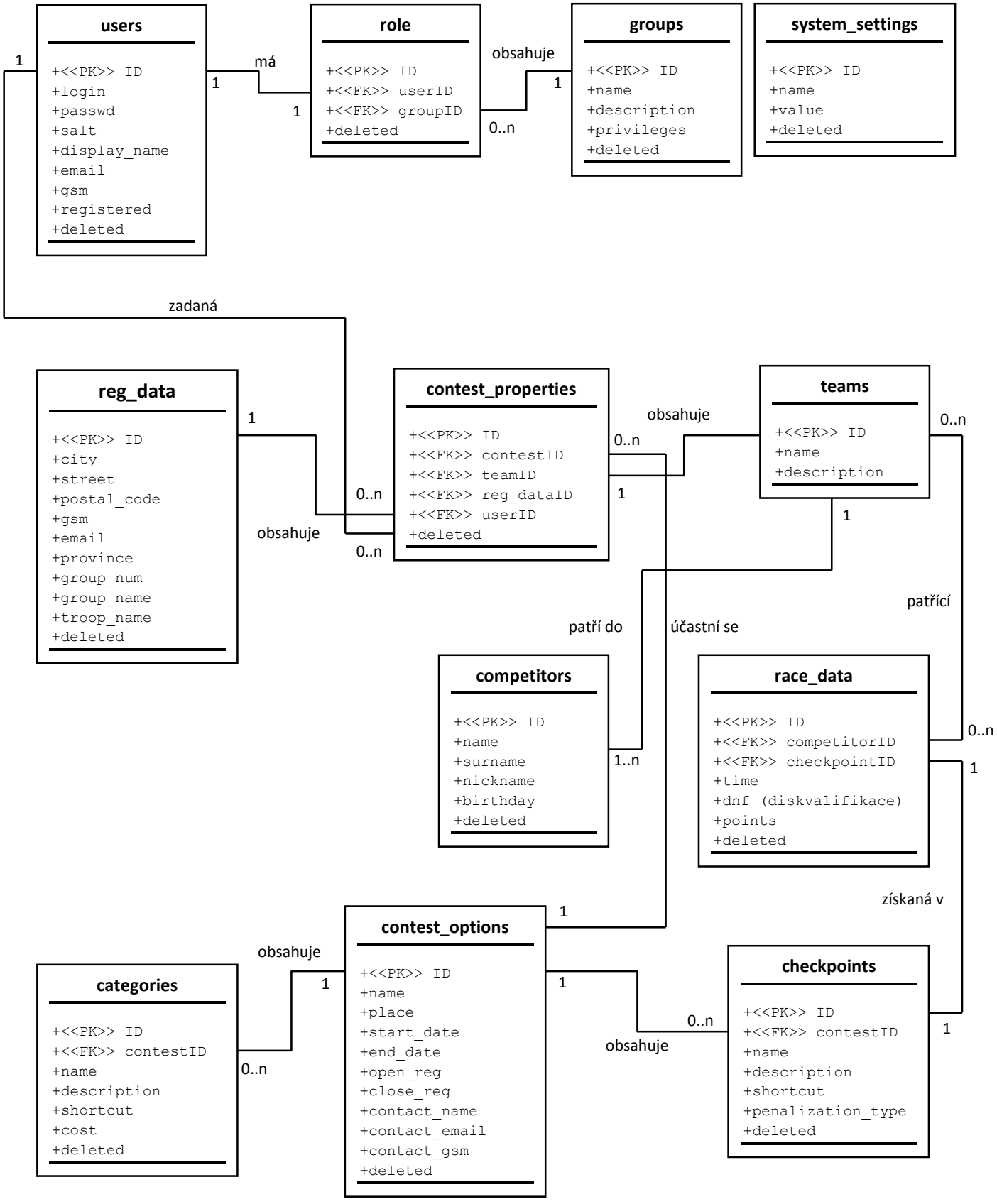
Příloha A

Kompletní Use Case diagram



Příloha B

Kompletní ER diagram



Příloha C

Přihláška do skautské části závodu

Junák - svaz skautů a skautek ČR, středisko Dvojka Praha, sídlo: Bělehradská 51, 120 00 Praha 2, IČ: 649 36 121

St. číslo	**Přihláška k závodu** **Napříč Prahou – přes tři jezy 2014** (odevzdejte při registraci)	**Pořadí**

Kategorie	R	F	S	G	B	Z	O	C	C1r	C2r	K1r	K2r	C1s	C2s	K1s	K2s
	roveři P550	rangers P550	skauti P550	skautky P550	žabičky a vlčata P550	žáci OPEN	dospělý OPEN	turistická kanoe	rychlostní singl	rychlostní debl	rychlostní kajak	rychlostní dvojkajak	sjezdový singl	sjezdový debl	sjezdový kajak	sjezdový dvojkajak
pohlaví	muži mix	ženy	muži mix	ženy												
osob	max 5					max 7		2	1	2	1	2	1	2	1	2
kormidelník	nar. 96-98		nad 18 let													
posádka nar.	1996-98		99 a ml.		2003 a ml.	99 a ml.		nad 18 let								

Loď														**Pohlaví**		
pramice						kanoe					kajak			muži	mix	ženy
do 530 cm	do 550 cm	loď P550	nad 550 cm	raft	pěti-devíti kanoe	plast	nafukovací	jiná	sjezdová	rychlostní	sjezdový	rychlostní	jiný			
11	12	13	14	21	31	41	42	43	44	45	51	52	53	1	2	3

Roky narození (bez kormidelníka)		
03 a ml.	99 a ml.	96 a ml.
4	5	6

Kontaktní adresa (neskautské oddíly zde uvedou také název, sídlo oddílu a IČO)

Přejímka *vyplní rozhodčí nebo komodor*

☐ Zajištění proti potopení ☐ Vylévačka
☐ Vyvazovací lano 5m, 8mm ☐ Záchytná oka
☐ Vesty (☐ Helmy) ☐ Vlnolam
Schválil:

Číslo přístavu (střediska): formát XXX.XX	**Jméno přístavu: (střediska)**
Číslo oddílu:	**Jméno oddílu:**
Místo působení oddílu:	**Jméno posádky:**
E-mail uvedený v předběžné přihlášce:	

Jméno a příjmení	**Přezdívka** (Napiš to, co má být uvedeno na diplomu.)	**Datum narození**	**Host**
Kormidelník:			

Prohlášení

Posádka lodi, kterou přihlašuji, je řádně poučena o všech pravidlech akce, uvedených na webových stránkách závodu http://3jezy.skauting.cz/, relevantních právních předpisech (zejména Řád plavební bezpečnosti, zákon o vnitrozemské plavbě) a bude je dodržovat stejně jako pokyny vydané při výkladu trati před závodem (vše k nahlédnutí v kanceláři registrace). Účastníci, které přihlašuji, jsou dobrými plavci, znalí plavby na vodě obtížnosti WWII, fyzicky i psychicky připraveni absolvovat závod a mají předepsané i odpovídající vybavení a záchranné pomůcky. Nezletilí účastníci mají souhlas zákonného zástupce s účastí na této akci. Účastníci (i jejich zákonní zástupci) souhlasí s pořizováním a archivováním písemností, podobizen, obrazových snímků, obrazových a zvukových záznamů týkajících se účastníků nebo jejich projevů osobní povahy, a rovněž souhlasí s jejich používáním a zveřejňováním při prezentaci a obecné propagaci činnosti Junáka a této akce, a to v souladu s § 84, § 85 a násl. zák. č. 89/2012 Sb., obč. zák. v jeho platném znění. Souhlasím, v souladu s § 5 odst. 5 a § 9 písm. a) zákona č. 101/2000 Sb. o ochraně osobních údajů v platném znění, aby výše uvedený pořadatel shromažďoval a uchovával osobní a citlivé údaje, týkající se mé osoby a osob uvedených v této přihlášce, a to především pro evidenční a informační účely, související s účastí předmětných osob na této akci.

V Praze dne 28. září 2014

.....
podpis odpovědné osoby (kormidelníka)

Vyplní pořadatel

STVRZENKA č. *3JS* ..... Dne 28. 9. 2013 Účastnický poplatek Napříč Prahou - přes tři jezy Částka: .....

Přijal: .....

Příloha D

Přihláška do veřejné části závodu

Junák - svaz skautů a skautek ČR, středisko Dvojka Praha, sídlo: Bělehradská 51, 120 00 Praha 2, IČ: 649 36 121

Přihláška do volného splutí

Přihlašující:

Jméno a příjmení Datum narození Adresa

Prohlášení přihlašujícího na Napříč Prahou – přes tři jezy 2014

Posádky lodí, které přihlašuji, jsou řádně poučeny o všech pravidlech akce, uvedených na webových stránkách závodu http://3jezy.skauting.cz/, relevantních právních předpisech (zejména Řád plavební bezpečnosti, zákon o vnitrozemské plavbě) a budou je dodržovat stejně jako pokyny vydané při výkladu trati před závodem (vše k nahlédnutí v kanceláři registrace). Účastníci, které přihlašuji, jsou dobrými plavci, znalí plavby na vodě obtížnosti WWII, fyzicky i psychicky připraveni absolvovat závod a mají předepsané i odpovídající vybavení i záchranné pomůcky. Nezletilí účastníci mají souhlas zákonného zástupce s účastí na této akci. Účastníci (i jeho zákonní zástupci) souhlasí s pořizováním a archivováním písemností, podobizen, obrazových snímků, obrazových a zvukových záznamů týkajících se účastníka nebo jejich projevů osobní povahy, a rovněž souhlasí s jejich používáním a zveřejňováním při prezentaci a obecné propagaci činnosti Junáka a této akce, a to v souladu s § 84, § 85 a násl. zák. č. 89/2012 Sb., obč. zák. v jeho platném znění. Souhlasím, v souladu s § 5 odst. 5 a § 9 písm. a) zákona č. 101/2000 Sb. o ochraně osobních údajů v platném znění, aby výše uvedený pořadatel shromažďoval a uchovával osobní a citlivé údaje, týkající se mé osoby a osob uvedených v této přihlášce, a to především pro evidenční a informační účely, související s účastí předmětných osob na této akci.

V Praze dne 28. září 2014

.....
podpis přihlašujícího

Osob do 18 let z Prahy:	
Osob do 18 let mimo Prahu:	
Osob od 19 do 26 let:	
Osob od 27 let:	
Osob celkem:	
Pramic:	
Kánoí (i nafukovacích):	
Paddleboardů:	
Raftů:	
Kajaků:	
Lodí celkem:	

Vyplní pořadatel

STVRZENKA č. *3JS*..... Dne 28. 9. 2014 Účastnický poplatek Napříč Prahou – přes tři jezy 30 Kč za osobu Částka: .....Kč

Přijal: .....

Příloha E

Testovací přístupové údaje do systému

Přihlášení do systému je možné na adrese http://reg3jez.skauting.cz/.

Testovací účty pro systém registrace

uživatelské jméno	heslo
admin@example.com	adminadmin
test@example.com	testtest

Testovací účty pro SkautIS

uživatelské jméno	heslo	oprávnění
kraj.vary	vary.Web2	Vedoucí/administrátor kraje
kraj.tgm	tgm.Web3	Vedoucí/administrátor kraje
okres.blansko	blansko.Web1	Vedoucí/administrátor okresu
stredisko.koprivnice	koprivnice.Web5	Vedoucí/administrátor střediska
snem.sneznik.kk	komise1	Člen kandidátní komise střediskového sněmu
snem.sneznik.uc	ucastnik1	Účastník střediskového sněmu