VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

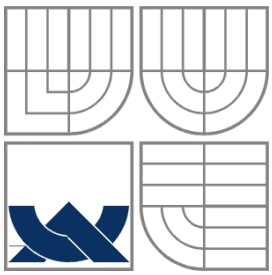
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INFORMAČNÍ SYSTÉM PRO SLEDOVÁNÍ PRÁCE UŽIVATELŮ

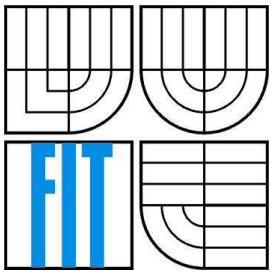
BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE MARTIN GÁBOR
AUTHOR

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INFORMAČNÍ SYSTÉM PRO SLEDOVÁNÍ PRÁCE UŽIVATELŮ

INFORMATION SYSTÉM FOR USERS' WORKLOAD MONITORING

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

MARTIN GÁBOR

VEDOUCÍ PRÁCE
SUPERVISOR

MGR. MAREK RYCHLÝ

BRNO 2008

Zadání bakalářské práce/6215/2007/xgabor00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2007/2008

Zadání bakalářské práce

Řešitel: **Gábor Martin**
Obor: Informační technologie
Téma: **Informační systém pro sledování práce uživatelů**
Kategorie: Web

Pokyny:

1. Seznamte se s technologií webových služeb a jejich aplikací. Po konzultaci s vedoucím navrhněte rozhraní webové služby umožňující příjem údajů o činnostech uživatelů na vzdálených strojích a jejich změny (např. oznámení změny činnosti, doporučení změny činnosti, registrace události, která nastane při změně činnosti, apod.).
2. Navrhněte informační systém, který bude konfigurovat síť webových služeb monitorujících činnosti skupiny uživatelů a který bude sbírat z těchto služeb informace. Systém umožní tvorbu zpráv o činnostech uživatelů (max. a min. prováděné činnosti, průměrné délky, nejčastější posloupnosti, apod.).
3. Implementujte systém jako webovou aplikaci. Pro komunikaci s monitorujícími službami musí systém poskytovat také kompatibilní rozhraní webové služby.
4. Zhodnoťte dosažené výsledky a diskutujte možnosti rozšíření.

Literatura:

- W3C. *Web Services Architecture*. Working group note, February 2004. [http://www.w3.org/TR/ws-arch/]
- Erl, T. *Service-Oriented Architecture: Concepts, Technology, and Design.* Prentice Hall PTR. 2005. ISBN 0-13-185858-0

Při obhajobě semestrální části projektu je požadováno:

- Body 1 a 2.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Rychlý Marek, Mgr.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2007
Datum odevzdání: 14. května 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Ing. Jaroslav Zendulka, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Martin Gábor**
Id studenta: 79032
Bytem: Vácka 4112, 018 41 Dubnica nad Váhom
Narozen: 29. 08. 1986, Bojnice
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
bakalářská práce

Název VŠKP: Informační systém pro sledování práce uživatelů
Vedoucí/školitel VŠKP: Rychlý Marek, Mgr.
Ústav: Ústav informačních systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:
tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☐ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

....

Nabyvatel Autor

Abstrakt

Cílem bakalářské práce je implementovat informační systém pro sledování práce uživatelů. Informační systém sbírá informace z přihlášených stanic, které poskytuje uživateli ve formě statistik. Komunikace se stanicemi probíhá prostřednictvím webových služeb. Informační systém je implementován v prostředí Java EE jako multiplatformní aplikace s webovým rozhraním.

Klíčová slova

Informační systém, sledovaní práce, webové služby, komunikace, statistiky, grafy, Java EE, JSP, XHTML, MySQL, SQL

Abstract

The goal of the bachelor thesis is an implementation of the information system for users' workload monitoring. The information system collects information from online stations, which provides to users as statistics. The communication with stations is proceeded by web services. The information system is implemented in Java EE as a multiplatform application with web interface.

Keywords

Information system, workload monitoring, web services, communication, statistics, graphs, Java EE, JSP, XHTML, MySQL, SQL

Citace

Gábor Martin: Informačný systém na sledovanie práce užívateľov. Brno, 2008, bakalárska práca, FIT VUT v Brně.

Informačný systém na sledovanie práce užívateľov

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením Mgr. Marka Rychlého.
Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

……
Martin Gábor
20.4.2008

Poďakovanie

Týmto by som sa chcel poďakovať hlavne svojmu odbornému konzultantovi Mgr. Markovi Rychlému za technickú podporu pri zostavovaní celého projektu a tejto odbornej dokumentácie.

Obsah

Úvod

V dnešnej dobe je podmienkou efektívneho fungovanie firiem osvojenie moderného manažmentu. Dobrý manažment totiž zabezpečuje prosperitu. Ten tvoria kvalifikovaný ľudia, ktorí musia mať potrebné vedomosti, skúsenosti a schopnosti pre úspešné vykonávanie manažérskych činností. Okrem toho k výkonnosti firiem významne prispieva aj efektívne využívanie všetkých informácií. Za týmto účelom sa v podnikoch využívajú systémy, ktoré produkujú kvalitné informácie, účelné pre riadenie. K snahe každej napredujúcej firmy patrí takisto dosiahnutie dobrej koordinácie manažmentu a jej zamestnancov. Pri ich veľkom množstve je to dosť komplikované. Takisto je dôležité dosiahnutie efektivity práce zamestnancov v pracovnej dobe a správne usmerňovanie ich náplne práce a obmedzenie ich nevyužitého pracovného času.

Cieľom tohto projektu je uľahčiť prácu zamestnávateľom a vedúcim a poskytnúť im komplexné riešenie na báze informačného systému, prístupného pomocou webového rozhrania kdekoľvek, ktoré by systematicky dokázalo zabezpečiť koordináciu zamestnancom a popri tom získavalo informácie o ich činnosti v reálnom čase. Výsledky by zobrazovalo v štatistikách a grafoch.

Druhá kapitola sa zaoberá analýzou technologickými prostriedkami, ktoré boli použité v tomto informačnom systéme. Nachádza sa tu stručný popis implementačných nástrojov, ktorý doplňujú obrázky pre rýchlejšie pochopenie danej problematiky.

V tretej kapitole je možné nájsť návrh celého informačného systému. V jej úvode sú popísané požiadavky na systém od zadávateľa. Nachádza sa v nej diagram prípadov užitia (*Use-Case*), ktorý popisuje užívateľov systému a ich možnosti. Tiež je tu zobrazený E-R diagram, potrebný pre správne uchovávanie informácií v databáze. Bol tiež dôležitým prostriedkom pri jej zostavení. Dôležitou podkapitolou je návrh webových služieb, potrebných pre komunikáciu s klientskou aplikáciou.

Témou piatej kapitoly je popis implementácie jednotlivých funkčných častí systému. V jej úvode je možné nájsť priebeh tvorby grafického užívateľského rozhrania. Za ním nasleduje podrobný popis práce s databázou vrátane jej samotného vytvorenia. Kapitola pokračuje popisom vytvorenia webových služieb, čomu významne dopomáhala práca s nástrojom NetBeans IDE. Ďalej sa tu nachádza implementácia štatistík, v podobe číselných údajov, ale aj grafov, ktoré významne poukazujú na výsledky rozhrania. Za tým pokračuje popis prostriedkov, ktoré zabezpečujú ovládanie a prácu so systémom. Sú tu vysvetlené pojmy stanica, náplň práce, ich prepojenie a ich miesto v systéme.

Kapitola testovanie objasňuje postup a dôvody testovania informačného systému.

Za ňou nasleduje popis možného rozšírenia, kde sú načrtnuté niektoré nedostatky a problémy, ktoré by mohli byť v niektorých prípadoch doplnené do systému.

Poslednou kapitolu je záver, ktorý obsahuje zhodnotenie dosiahnutých výsledkov vrátane mojich prínosov.

1 Analýza projektu

V tejto kapitole budú zhrnuté všetky použité technológie pri vývoji celej aplikácie.

1.1 Webové služby (WS)

Webová služba je softwarový systém navrhnutý pre podporu vzájomnej interakcie počítač – počítač prostredníctvom siete. Obsahuje rozhranie popísané väčšinou pomocou WSDL. Ostatné systémy spolupracujú s webovými službami vo vopred určenom štýle použitím SOAP správ, ktoré sú typicky prenášané použitím HTML a XML na základe princípov platiacich na webe [3].

WSDL (*Web Service Description Language*) je popis webovej služby vo formáte XML, obsahujúci všetky poskytované služby a definície ako k ním môžu klienti pristupovať. XML registre umožňujú poskytovať popisy o webových službách, vyhľadávanie iných služieb a získanie WSDL informácií popisujúcich rozhranie webovej služby [1].

SOAP (*Simple Object Access Protocol*) je protokol na výmenu správ založených na XML prostredníctvom siete, najmä pomocou *HTTP*.
Formát SOAP správ tvorí základnú vrstvu komunikácie medzi webovými službami a ponúka prostredie na tvorbu zložitejšej komunikácie. Pre komunikáciu prostredníctvom protokolov SOAP existuje niekoľko rôznych druhov šablón. Najznámejší z nich je RPC šablóna, kde jeden z účastníkov komunikácie je klient a druhý server. Ten ihneď odpovedá na požiadavky klienta. SOAP je nástupca **XML-RPC** (*XML Remote Procedure Calling*) [4].

Ďalšie špecifikácie webových služieb:

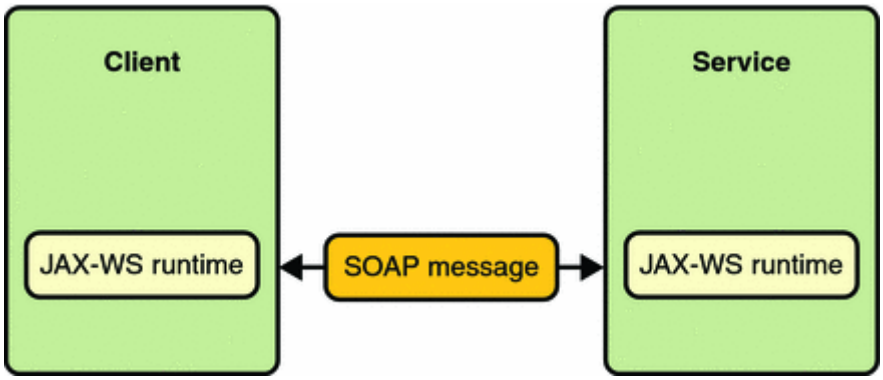
- **Bezpečnosť WS** definuje ako používať šifrovanie XML a signatúru XML v SOAP pre bezpečnú výmenu správ ako možnú alternatívu alebo rozšírenie protokolu *HTTPS*.
- **Spoľahlivosť WS** predstavuje protokol pre spoľahlivé posielanie správ medzi dvomi webovými službami.
- **Adresovanie WS** definuje spôsob popisu adresy príjemcu vo vnútri SOAP správy.
- **Transakcie vo WS** určuje spôsob manipulácie s transakciami.

Webová služba môže byť taktiež použitá na implementovanie architektúry **SOA** (*Service Oriented Architecture*), kde hlavnou zložkou komunikácie je správa, nie nejaká obsluha. Takáto služba sa označuje ako „*message-oriented*" [4].

1.1.1 JAX-WS

JAX-WS (*Java API for XML Web Services*) je technológia, ktorá umožňuje vytvoriť webové služby v prostredí Java. Jej výhodou je možnosť pristupovať k iným webovým službám, ktoré nie sú zostavené na platforme Java a naopak.

JAX-WS umožňuje vývojárom zostavovať „*message-oriented*“ ako aj „*RPC-oriented*“ webové služby. Tie sú definované ako triedy Java poskytujúce metódy. Po ich použití sa vytvoria potrebné WSDL popisy. SOAP zabezpečuje komunikáciu tak ako to je znázornené na obrázku 1.1 [5].



Obrázok 1.1: Komunikácia webovej služby s klientom prostredníctvom JAX-WS (prevzaté z [6])

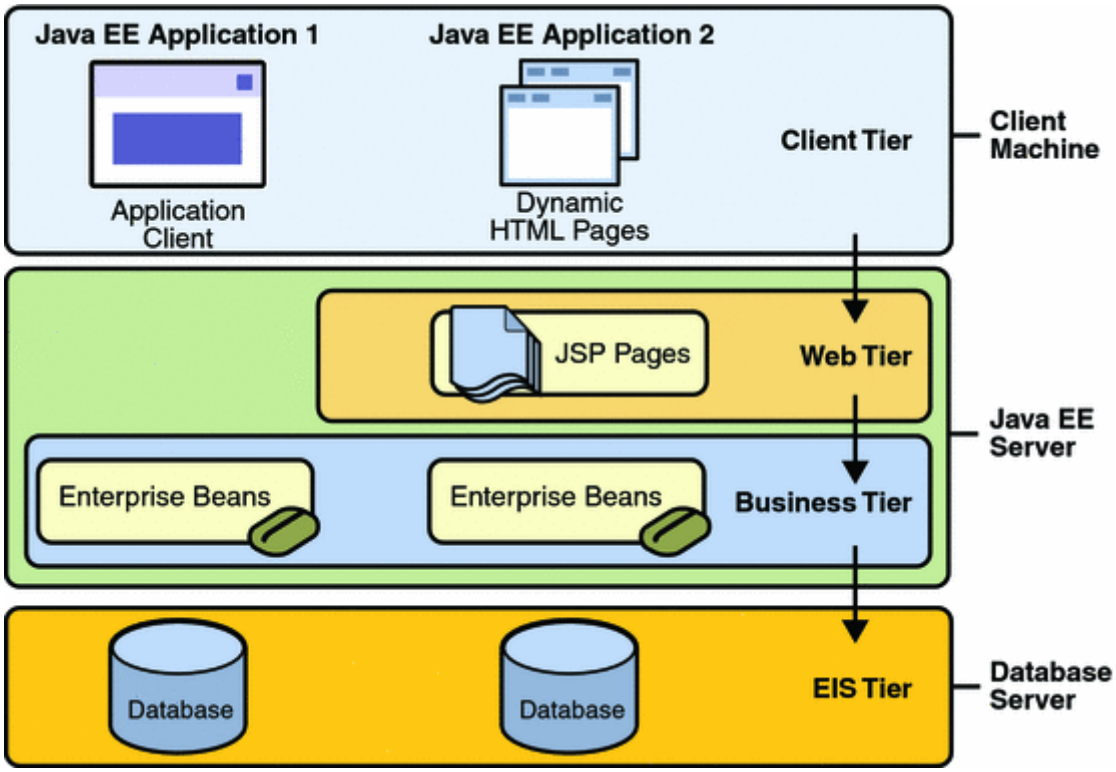
1.2 Java EE

Java Platform, Enterprise Edition (alebo **Java EE**, skôr označovaná ako *Java 2 Enterprise Edition* alebo *J2EE*) je súčasť platformy Java, určená pre vývoj a prevádzku podnikových aplikácií a informačných systémov. Základom pre platformu Java EE je platforma Java SE, nad ktorou sú definované súčasti tvoriace Java EE [4]. To umožňuje tvoriť užívateľské rozhranie so *Swing* alebo AWT (*Abstract Window Toolkit*) ako aj prostredníctvom HTML. Navyše Java EE obsahuje ďalšie API (*Application Programming Interface*) na tvorenie užívateľských rozhraní. Volá sa JSF (*JavaServer Faces*) [1].

1.2.1 Architektúra

Java EE podporuje navrhovanie aplikácií postavených na viacerých vrstvách. Tie sa bežne rozdeľujú na *prezentačnú*, *logickú* a *databázovú* pri 3-vrstvovej architektúre. To je najbežnejší spôsob pri navrhovaní v tomto prostredí. Nie je však nevyhnutný. Pri jednoduchších systémoch je možné použiť jednovrstvovú architektúru. Všetky služby poskytované systémom (*užívateľské rozhranie*, *logika*

procesov, perzistentný dátový prístup) sú spracované na jednom počítači a vytvorené do jednej aplikácie. Dôležitejšie systémy môžu využiť výhodu oddeleného databázového servera, na ktorý sa odkazujú SQL dotazmi. To umožňuje klient/server architektúra. Jej významom je oddeliť data od logiky tak, aby k nim mohli viacerí užívatelia pristupovať súbežne. Java EE umožňuje vytvoriť systém postavený aj na viacvrstvových architektúrach. Každá má svoje výhody a nevýhody, takže je dôležité podľa rozsahu projektu zvoliť tú správnu [1].



Obrázok 1.2: Viacvrstvová architektúra (prevzaté z [7])

1.2.2 Komponenty a kontajnery

V Java EE platforme sa časti aplikácie, ktoré zapúzdrujú určitú logiku podľa pravidiel Java EE, nazývajú komponenty (napr. komponenty EJB, applety). Kontajnery poskytujú podporu týmto komponentom. Tie používajú protokoly a metódy kontajnerov pre prístup iným aplikačným komponentom a službám poskytovaných serverom. Aplikačný server je vybavený kontajnerom klienta, applet kontajnerom, webovým a EJB kontajnerom [7].

1.2.3 Java Servlets

Java Servlets sú programové komponenty bežiace na strane servera napísané v Jave. Sú nezávislé na protokole a platforme. Vyžadujú však JVM (*Java Virtual Machine*). Podľa špecifikácie sú servlety nevizuálne komponenty, teda sa nezobrazujú pomocou žiadneho GUI. Však to neznamená, že

nemôžu generovať užívateľské rozhranie. V skutočnosti sa väčšinou používajú servlety pracujúce nad internetovým protokolom HTTP. Odpoveďou týchto servletov je potom v prevažnej väčšine HTML kód.

Servlety môžeme použiť na rôzne úlohy: čítanie/zapisovanie do databázy, posielanie mailov, spracovávanie formulárov, posielanie cookies atď. Jednoducho sú určené na tvorbu dynamických internet/intranet riešení, podobne ako JavaServer Pages [8].

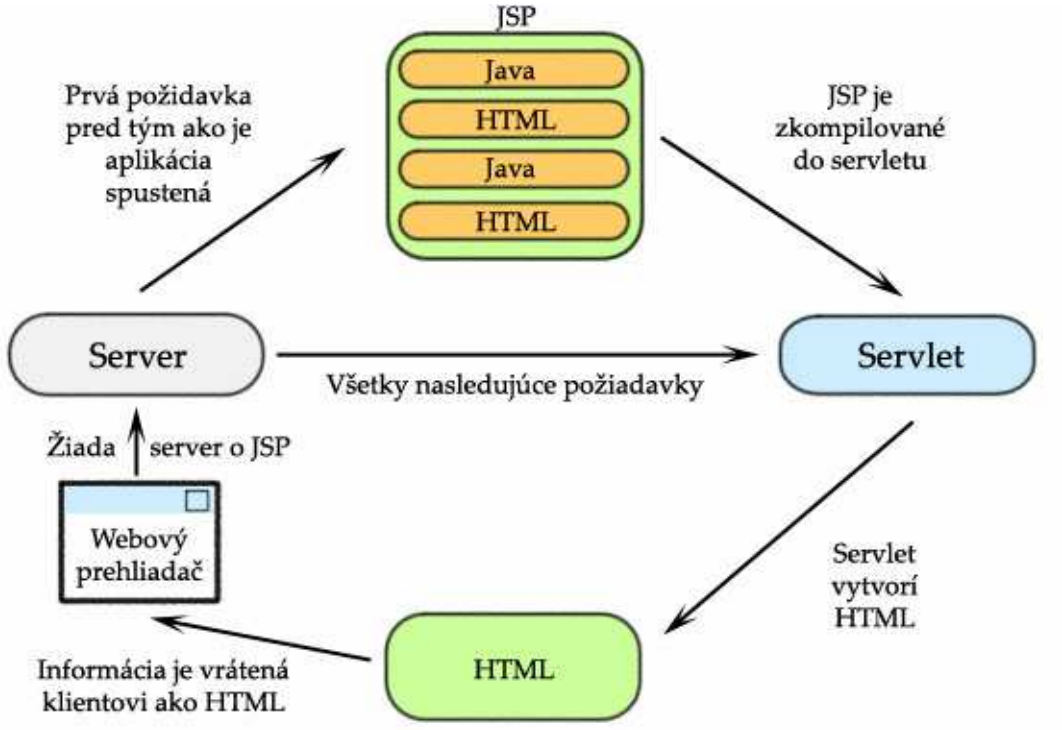
V Java EE sú implementované rozhraním *javax.servlet.Servlet*. To obsahuje metódy, ktoré sú automaticky ihneď použité pri obdržaní „*requestu*“ od servera.

1.2.4 JSP (JavaServer Pages)

JSP je technológia, ktorá bola vyvinutá spoločnosťou Sun Microsystems, za účelom vývoja aplikácií na strane servera [8].

JSP stránka je textový dokument, ktorý obsahuje dva typy textu, statické dáta, ktoré môžu byť v ľubovolnom textovom formáte (napr. *HTML, SVG, WML* a pod.) a JSP elementy, ktoré vytvárajú dynamický obsah [7].

Na rozdiel od iných technológií umožňuje JSP naplno využiť silu programovacieho jazyka Java, ktorá je veľmi výkonným univerzálnym jazykom s veľkou kolekciou rozhrania API. Rôzne balíčky s rozhraním API sú dostupné pre veľké množstvo rôznych aplikácií, vrátane databázových, grafických, bezpečnostných, spracúvajúcich transakcie a pod. Jazyk JSP má voči ostatným nesmiernu výhodu. Java nie je skriptovací jazyk, takže dokument JSP nie je pri každom požiadavku interpretovaný úplne od začiatku [2].



Obrázok 1.3: Princíp spracovania JSP stránky (podľa [1])

Spracovanie JSP stránok je zobrazené na obrázku 1.3.

1.2.5 JSTL (JSP Standard Tag Library)

JSTL je JSP knižnica, ktorá obsahuje nové tagy, použiteľné priamo v XHTML kóde. Tie sú funkcionálne podobné aplikáciám, napísaných v jazyku JSP. Napríklad, namiesto prechádzania zoznamu iteratívnym spôsobom použitím jazyka JSP, JSTL definuje štandardný tag, ktorý funguje rovnako. Takýto nástroj je možné použiť na komplikované JSP kontajnery, čím uľahčí implementáciu.

JSTL podporuje časté, štrukturálne úlohy ako spomínaná iterácia alebo podmienky, tagy pre manipuláciu s XML dokumentmi, internacionalizáciu atď. Je to výrazový jazyk, poskytujúci API pre vývojárov, ktoré uľahčuje konfiguráciu JSTL tagov a rozvoj nových, vlastných [9].

1.3 Značkovací jazyk XHTML

XHTML (Extensible HyperText Markup Language) je rodina súčasných a budúcich typov dokumentov a modulov, ktoré rozširujú HTML 4. Je založená na XML dokumente. Prvá verzia má označenie XHTML 1.0. Je to preformulovaná verzia HTML 4, založená na XML 1.0. Má niekoľko zásadných zmien a výhod, pre ktoré ju vývojári zvolia na svoje projekty. Sú to:

- XHTML sú XML kompatibilné, takže sú ľahko prehliadateľné, upravovateľné a validovateľné štandardnými nástrojmi XML.
- XHTML dokumenty môžu byť napísané tak, že ich funkčnosť je rovnaká, ak nie lepšia na užívateľských agentoch s podporou HTML 4.
- XHTML dokumenty môžu byť využité aplikáciami, ktoré spolupracujú s HTML objektovým modelom alebo XML objektovým modelom dokumentu (*DOM*).

XHTML je značkovací jazyk, ktorý obsahuje značky a atribúty značiek. Tie možné jednoducho pridať prostredníctvom XHTML modulov. Tieto umožnia použitie súčasných i nových vlastností pri vytváraní dokumentov.

XHTML má tri definície DTD (*Document Type Definition*). Sú to Strict, Transitional a Frameset. Každá z nich definuje určitú sadu elementov a ich atribút.

Jazyk je oproti HTML jednoduchší. Sú z neho odstránené atribúty prvkov, ktoré nastavovali vzhľad a rozloženie elementov. Tieto vlastnosti boli nahradené kaskádovými štýlmi [10].

1.4 Kaskádové štýly (CSS)

CSS (*Cascading Style Sheets*) boli vytvorené na formátovanie internetových dokumentov. Hlavným dôvodom bolo oddeliť štruktúru HTML alebo XHTML od vzhľadu. Tým pádom ostala logika programu v dokumente a vzhľad sa navrhuje len prostredníctvom štýlov.

CSS možno presunúť do externých súborov, čím sa zmenší dátová veľkosť a tým pádom je možné jedným súborom zmeniť štýl celej stránky. CSS zaručuje rovnaké vykresľovanie vo všetkých prehliadačoch [11].

1.5 JavaScript

JavaScript je skriptovací programovací jazyk, najčastejšie používaný s jazykom HTML, interpretovaný na strane klienta.

Servery založené na tomto jazyku odosielajú do klientských prehliadačov vopred napísané dokumenty. Okamžite keď dokument dorazí, prehliadač ho načíta a interpretuje príkazy napísane v JavaScript. To sa následne prejaví v okne prehliadača [2].

1.6 SQL a MySQL

SQL (*Structured Query Language*) je takmer univerzálne implementovaný relačný jazyk v súčasnosti používaný hlavne na správu databázy. Taktiež však podporuje procedurálne funkcie.
SQL ale nemá veľa základných prvkov väčšiny iných počítačových jazykov. Z toho dôvodu je často označovaný ako *podjazyk* dat, pretože je najčastejšie používaný v spojení s programovacími jazykmi aplikácií (napr. *C* a *Java*), čo sú jazyky, ktoré nie sú navrhnuté pre manipuláciu s dátami.

Základné syntaktické konštrukcie jazyka SQL sú nasledovné (zdroj: [12]):

- **Jazyk definície dát (DDL – Data Definition Language)** Príkazy DDL sa používajú k vytváraniu, úpravám alebo odstraňovaniu databázových objektov (napr. tabuliek, pohľadov, schém, domén, triggerov a pod.). Sú to hlavne CREATE, ALTER a DROP.
- **Jazyk pre riadenie dát (DCL – Data Control Language)** Príkazy DCL umožňujú riadiť, kto bude mať prístup k určitým objektom databáze. V jazyku DCL môžeme povoľovať alebo obmedzovať prístup pomocou príkazov GRANT a REVOKE.
- **Jazyk pre manipuláciu s dátami (DML – Data Manipulation Language)** Príkazy DML sa používajú k prehliadaniu, pridávaniu, úpravám alebo odstraňovaniu dát uložených v databáze. Hlavnými kľúčovým slovami sú SELECT, INSERT, UPDATE a DELETE.

MySQL je populárny *open-source* databázový systém. Má mnoho výhod. Je viacvláknový, podporuje viacero platforiem. MySQL je od začiatku optimalizovaný na rýchlosť aj za cenu viacerých zjednodušení [13].
Databázový systém je relačný, typu DBMS (*Database Management System*). Každá databáza v MySQL je tvorená z jednej alebo viacerých tabuliek, ktoré majú riadky a stĺpce. V riadkoch sa rozoznávajú jednotlivé záznamy, stĺpce udávajú dátový typ jednotlivých záznamov, pracuje sa s nimi ako s poľami. Práca s MySQL databázou je vykonávaná pomocou takzvaných dotazov, ktoré vychádzajú z programovacieho jazyka SQL (*Structured Query Language*) [14].

MySQL Server je klient/server systém. Komunikácia je postavená na viacerých vrstvách. Podporuje veľké množstvo rozhraní, niekoľko klientských programov a knižníc, administratívnych nástrojov a mnoho API.
MySQL Server je možné použiť aj vo forme vstavanej viacvláknovej knižnice, ktorú môžu programátori využiť vo svojich projektoch k dosiahnutiu menšieho, rýchlejšieho, ľahšie spracovateľného produktu [13].

1.6.1 MySQL Connector/J

MySQL ponúka pripojenie k databáze pre klientské aplikácie, ktoré sú vyvíjané v programovacom jazyku Java prostredníctvom JDBC (*Java Database Connectivity*) drivera, ktorý sa nazýva MySQL Connector/J.

Je to implementácia JDBC špecifikácie verzie 3.0 v jazyku Java, ktorá komunikuje priamo s MySQL serverom podľa protokolu MySQL [13].

1.7 Knižnica JFreeChart

JFreeChart je open-source Java balík, ktorý umožňuje jednoducho vytvárať hotové profesionálne grafy. Tie je možné potom exportovať do obrázkov vo formáte *PNG* a *JPEG* ako aj do vektorovej grafiky (napr. *PDF*, *EPS* a *SVG*) [14].

2 Návrh riešenia

Výsledný informačný systém musí predovšetkým splniť požiadavky zadávateľa. Preto je potrebné zhrnúť a podrobne analyzovať všetky požiadavky na systém.
Zadaný informačný systém má obsahovať niekoľko základných parametrov.

- Pomocou webových služieb komunikuje so stanicami.
- Zbiera informácie z jednotlivých staníc, ktoré mu ich posielajú a koriguje takisto pomocou správ ich činnosť.
- Systém umožňuje tvorbu správ a štatistík o činnostiach užívateľov.
- Systém musí byť implementovaný ako webové rozhranie.

2.1 Návrh rozhrania

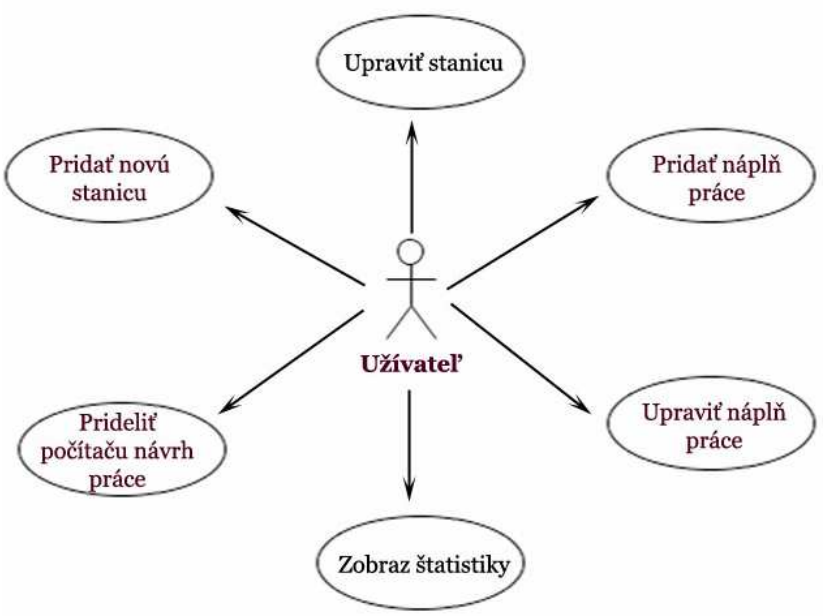
XHTML je značkový jazyk, ktorý nedokáže udržovať kontext stránok, takže musí každá stránka obsahovať všetky potrebné informácie o obsahu. Tie sa v prípade jednotného grafického návrhu rozhrania budú opakovať. To je možné vyriešiť niekoľkými spôsobmi pomocou dynamického tvorenia stránok na servery. Jednou z možností je použitie servletov a do určitej formy vkladať len obsah.

Návrh tohto informačného systému je postavený na princípe použitia kaskádových štýlov pre grafické rozmiestnenie, ktorý bude používať každá stránka ako externý odkaz na súbor. Každá stránka má rovnaké usporiadanie tagov, pričom sú do nej vkladané externé stránky pomocou JSTL. Pri formulároch sa informácie prenášajú pomocou metódy GET aj POST, pričom POST sa používa na prenos dát, ukladaných do databázy. Metóda GET pomáha v systéme k riadeniu dát, čím uľahčuje implementáciu.

2.1.1 Diagram prípadov užitia (Use-Case)

Pri tvorbe informačných systémov je dôležitou súčasťou *Use-Case* diagram. Poskytuje nám rýchlu predstavu o všetkých funkciách systému a ich účastníkoch. Diagramy prípadov užitia modelujú hranice systému, jeho účastníkov, prípady užitia a interakcie medzi nimi a účastníkmi.

V informačnom systéme na sledovanie práce užívateľov je jediným účastníkom vedúci, resp. zamestnávateľ, resp. administrátor systému. Ten má v ňom niekoľko možností. Všetky sú popísané v diagrame prípadov užitia, ktorý je znázornený na obrázku 2.1.

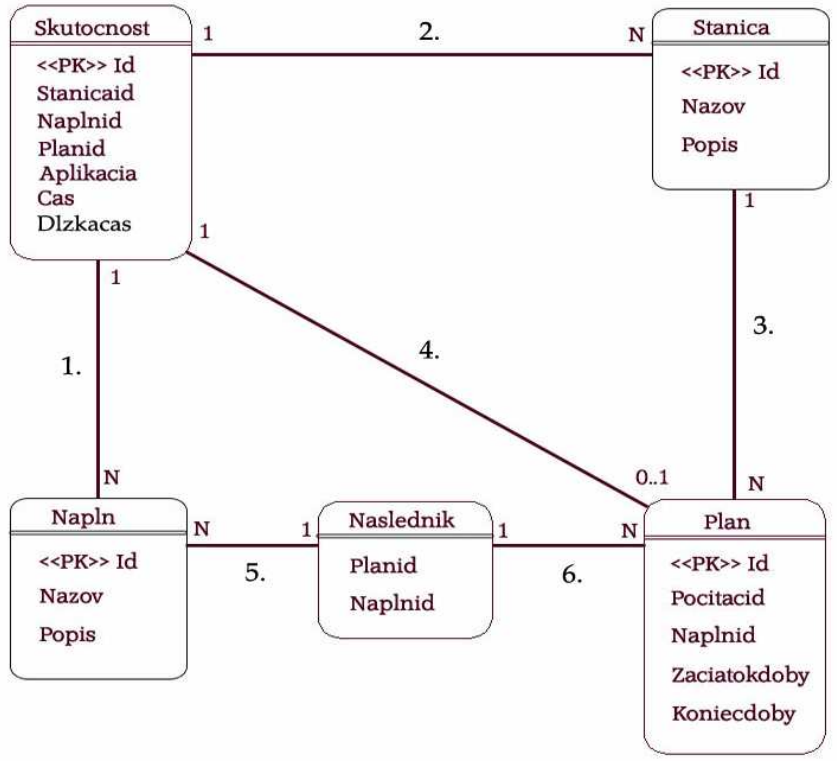


Obrázok 2.1: Diagram prípadov užitia (Use-Case)

2.1.2 Diagram tried (ER Diagram)

Jednou z najzaužívanejších metód modelovania relačných databáz je entitne-vzťahové modelovanie, ktorým výsledkom je entitne-vzťahový diagram (ER diagram). Ten potom slúži na implementáciu databázy.

E-R diagram navrhovaného informačného systému je znázornený na obrázku 2.2.



Obrázok 2.2: E-R Diagram informačného systému

E-R diagram obsahuje nasledujúcich 5 entít:

- **Stanica** predstavuje jedného účastníka v systéme, ktorého náplň práce je sledovaná. Obsahuje údaje *Nazov*, *Popis* a *Id*, ktorý jednoznačne určuje daného účastníka. Názov a popis ho identifikuje výstižnejšie.
- **Napln** určuje jednu náplň práce zadanú v informačnom systéme. Obsahuje atribúty názov a popis pre zrozumiteľný popis činnosti, na ktorej budú stanice pracovať.
- **Plan** bude obsahovať informácie o stanici, ktorá bude mať zaregistrovanú nejakú náplň práce. Sú to plánované udalosti, ktoré bude administrátor systému zadávať do systému. Obsahuje atribúty *Pocitacid* a *Naplnid*, ktoré určujú, ktorá stanica bude pracovať na danej náplni. Atribúty *Zaciatokdoby* a *Koniecdoby* zaznamenávajú časový interval zadanej činnosti.
- **Naslednik** je entita simulujúca väzbu N:N medzi entitami Napln a Plan.
- **Skutocnost** zaznamenáva skutočnosť, ktorá sa v danom čase stala na stanici. Slúží pre prípad, že sa užívateľ prepne do inej náplne, alebo sa prihlási do systému resp. odhlási zo systému. Atribút *Stanicaid* je cudzí kľúč do tabuľky *Stanica*, *Naplnid* odkazuje na *Napln*. *Planid* je takisto cudzí kľúč, ktorý obsahuje id plánu. Ten je však vyplnený len v prípade definitívneho ukončenia istej náplne práce stanicou. Atribút *Aplikacia* obsahuje stanicou používané aplikácie. V prípade, že sa tam bude nachádzať reťazec „*online*“, bude to znamenať, že užívateľ sa práve prihlásil. V prípade reťazca „*offline*“ to bude naopak predstavovať jeho odhlásenie zo stanice. Atribút *Cas* bude automaticky doplňovaný informačným systémom presným časom. *Dlzkacas* bude obsahovať dobu, po ktorú bude daná skutočnosť v platnosti.

E-R diagram obsahuje nasledujúcich 6 vzťahov:

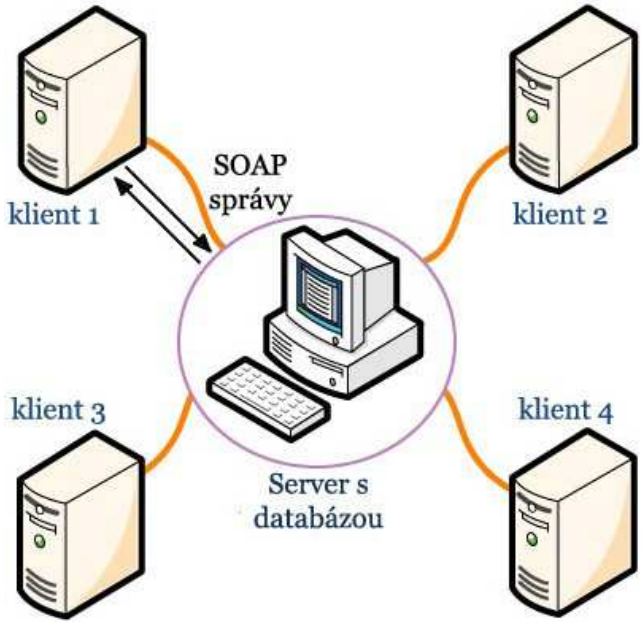
1. Väzba (**1:N**) určuje informácie o náplni, ktorá je zaznamenaná v skutočnosti. Túto náplň stanica vykonáva.
2. Väzba (**1:N**) označuje stanicu, ktorá zmenila udalosť a tým zaznamenala skutočnosť v systéme.
3. Tento vzťah (**1:N**) poukazuje na stanicu, ktorej bola pridelená určitá náplň práce v určitej dobe.
4. V prípade, že užívateľ ukončí určitú náplň práce, vytvorí sa táto väzba (**1:N**) na plán.
5. a 6. vytvárajú väzbu **N:N** spolu s pomocnou entitou *Naslednik*.

2.2 Návrh webových služieb

Webové služby v tomto informačnom systéme predstavujú jediný spôsob komunikácie medzi serverom a klientom. Server je aplikácia, na ktorej je informačný systém v prevádzke.
Výhodou webových služieb je teda hlavne ich dostupnosť v čase, kedy je server spustený. Zároveň sú činnosťou od seba nezávislé a viacvláknové. Keďže informačný systém by mal byť multiplatformný a klient je implementovaný ako aplikácia, pracujúca na platforme *Linux*, je opäť výhodné ich použiť. Komunikácia medzi serverom a klientmi teda prebieha prostredníctvom webových služieb. To znamená, že ak klient pošle požiadavku o službu, výmena informácií sa uskutoční pomocou SOAP.

V tomto informačnom systéme existujú tri služby webovej služby. Každá obsahuje parameter názov stanice, ktorý jednoznačne identifikuje klienta.

- **Zmena udalosti** je služba, ktorá informuje server o zmene udalosti u klienta. Tzn. že klient zvolí svoju náplň práce na inú. Jej použitie možno taktiež uplatniť aj pri vypnutí a zapnutí jeho klientského programu. V tom prípade bude parameter *aplikácia* naplnený ako *„online“* alebo *„offline“*. Ďalšími parametrami sú identifikačné čísla náplne a plánu. V prípade, že parameter identifikačného čísla plánu bude vyplnený, bude informačný systém považovať za ukončenú náplň práce.
- **Vyžiadanie odporúčanej náplne práce** – je služba, ktorá poskytuje klientovi v danom čase informáciu o jeho aktuálnej pracovnej činnosti zadanej v informačnom systéme.
- **Vyžiadanie všetkých náplni práce** – je služba, ktorá taktiež ponúka súhrn všetkých udalostí zadaných v systéme.



Obrázok 2.3: Komunikácia so serverom pomocou webových služieb

2.3 Voľba implementačných prostriedkov

Správna voľba implementačných prostriedkov je pri návrhu každého systému veľmi dôležitá. Trh ponúka niekoľko možností pre informačné systémy vytvorené v podobe webového rozhrania. Z tých viac známych sú to najmä PHP, ASP.NET a JSP. Sú to technológie, ktoré na strane servera dynamicky generujú obsah a posielajú ho klientovi prostredníctvom siete v podobe HTML stránok. JSP môže používať knižnice napísané v jazyku Java, preto je výhodné ho použiť. Takisto rozhranie klienta .

2.3.1 Netbeans IDE 6.0

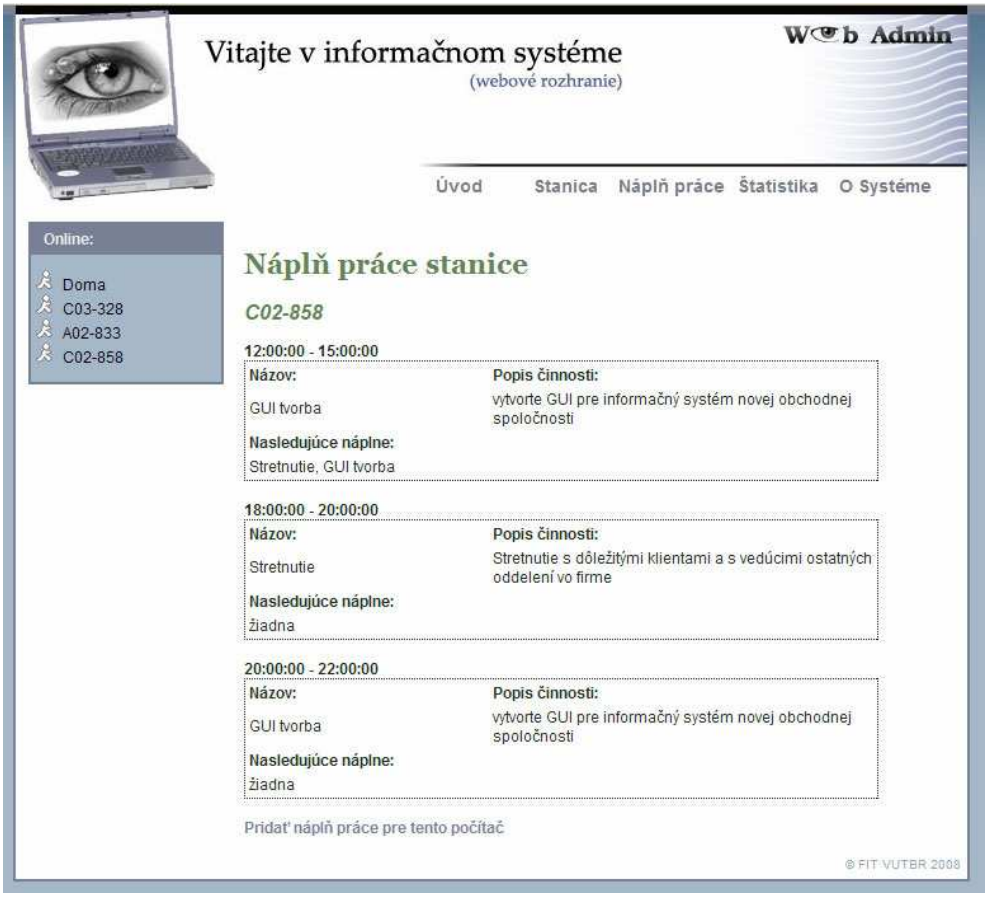
Je užívateľské rozhranie pre zhotovovanie projektov najme v technológiách Java. Práca s ním je veľmi jednoduchá a užívateľsky príjemná, preto je celý projekt zhotovený v tomto rozhraní.

3 Implementácia

V tejto kapitole je zhrnutá a podrobne popísaná implementácia jednotlivých častí systému. Sú popísané jeho funkcie, ktoré systém obsahuje. Samotná implementácia začala grafickým návrhom, ktorý bol vytvorený v podobe obrázka. Z neho bol vytvorený kaskádový štýl a základná štruktúra dokumentu. Tým bolo vytvorené grafické užívateľské rozhranie. Za tým nasledovalo vytvorenie databázy podľa návrhu E-R diagramu. Nakoniec boli postupne implementované všetky časti, ktoré zabezpečujú funkčnosť.

3.1 Grafické užívateľské rozhranie

GUI (*Graphical User Interface*), alebo v preklade grafické užívateľské rozhranie podporuje komunikáciu s užívateľom systému. Je dôležité, aby bolo esteticky namodelované a prvky rozmiestnené tak, aby bola práca pre užívateľa čo najjednoduchšia. V konečnom dôsledku si kupujúci dosť všíma GUI a niekedy sa podľa neho na úkor funkčnosti systému aj rozhoduje pri výbere, čo je dosť zarážajúce.



Obrázok 3.1: GUI informačného systému

Grafické užívateľské rozhranie informačného systému na sledovanie práce užívateľov bolo implementované pomocou kaskádového štýlu, umiestnenom v *styles/style.css*, v ktorom je určený dizajn každého elementu.

Každá stránka obsahuje iba svoju hlavnú časť s obsahom, hlavička a päta je oddelená a vložená. V hlavičke stánky (*head.jsp*) tvorí dôležitú úlohu menu, ktoré je horizontálne umiestnené na vrchnej lište. Pomocou neho je možné zobrazovať obsah jednotlivých stránok do hlavného okna, ktoré je pod vrchnou lištou.

V hlavnom okne je okrem obsahu okno s online užívateľmi, implementované tiež v oddelenom súbore *online.jsp*.

3.1.1 Kontrola užívateľom zadaných dát

Pri formulároch, ktoré sa vyskytujú v informačnom systéme je veľmi dôležité kontrolovať data, ktoré do nich užívateľ zadáva. To je možné vykonávať dvomi spôsobmi:

- **Na úrovni logickej vrstvy –** Kontrolovanie dát pred tým ako ich vložíme do databázy, do nej vstupujú už len korektné.
- **Priamo v databáze –** Vkladanie všetkých dát, pričom databáza vráti výsledok úspešnosti.

Pri projektoch sa musíme rozhodovať, komu zveríme túto úlohu. Väčšinou je to podnietené veľkosťou projektu. Pri malých môžeme použiť 2 variantu, pri tých veľkých by sme mali tú prvú, z dôvodu veľkého množstva dotazov na databázu.

Tento informačný systém nepočíta s veľkým zaťažením databáze, avšak kontrola formulárov je vykonávaná na logickej vrstve. To je zabezpečené prostredníctvom nástroja JavaScript, ktorý je vhodný pre tieto funkcie. Implementácia je v súbore *js/scripts.js*. Princíp kontroly je postavený na regulárnych výrazoch, ktoré kontrolujú či údaje vyplnené vo formulároch spĺňajú dané požiadavky.

3.2 Práca s databázou

V tejto podkapitole bude vysvetlené použitie databázy v informačnom systéme.

3.2.1 Vytvorenie

Pri vytvorení databázy som postupoval podľa E-R diagramu (obrázok 2.2). Všetky tabuľky boli vytvorené SQL skriptom, ktorý som aplikoval na MySQL servery. Všetky tabuľky okrem tabuľky *Naslednik* obsahujú primárne kľúče *Id*, ktoré sú identifikačným prvkom záznamov. Sú nastavené na *AUTO_INCREMENT*, takže pri vložení nového záznamu je id automaticky zvýšené o 1.

3.2.2 Manipulácia s dátami

Pripojenie, odpojenie a práca s databázou je sprostredkovaná balíkom WebAdmin.DB. Nachádzajú sa v ňom triedy *Station*, *Activity*, *Plan* a *Skutocnost*, ktoré popisujú jednotlivé tabuľky v databáze svojimi atribútmi. Hlavná trieda *AccessDB* zabezpečuje práve to pripojenie k databáze. Pomocou nej je možné pridávať, upravovať, zmazávať a vyberať s tabuliek uložených v databáze. To umožňujú metódy *executeQuery()* a *executeUpdate()*. Použitím týchto metód pri zložitejšom výbere z databázy sa naplní atribút *SQLResult*, z ktorého je možné pomocou metódy *getResponseDb(String Value, boolean Next)* oddeliť. Parameter *Value* je názov stĺpca, ktorý chceme na danom riadku získať. *Next* udáva, či chceme získať hodnotu z ďalšieho riadku, alebo z toho na ktorom sa nachádza ukazovateľ. Pri jednoduchom výbere z jednej tabuľky, stačí zavolať príslušnú metódu, ktorá naplní zoznam všetkých výsledkov. Napr. pri stanici sa zavolá *getStationsFromDB()*, ktorá vráti zoznam staníc.

3.2.3 Použitie knižnice JSTL

Okrem triedy *AccessDB* napísanej v jazyku Java, sú v informačnom systéme použité JSTL tagy pre vkladanie a upravovanie záznamov v databáze. Je tak urobené kvôli jednoduchosti, výsledok je však rovnaký.

3.3 Implementácia webových služieb

Webové služby je možné jednoducho vytvoriť v rozhraní NetBeans IDE. Celý priebeh zabezpečujú formuláre, ktoré následne vytvoria potrebnú šablónu. V informačnom systéme na sledovanie práce užívateľov bola vytvorená jedna webová služba, ktorá obsahuje jednotlivé operácie navrhnuté podľa návrhu (kapitola 2.4). Jej názov je *EventService* a jej funkčnosť je zabezpečená triedou *Event.java*, ktorá sa nachádza v balíku *WebAdmin.WS*.

Po spustení informačného systému na servery, je webová služba dostupná na adrese */EventService?WSDL*. Tá slúži klientom na sprístupnenie operácií, ktoré ponúka k plnému využívaniu. Táto služba je pre program nesmierne dôležitá, pretože bez nej by sledovanie nebolo možné vykonávať.

3.4 Štatistiky

Štatistiky sú dôležitou súčasťou webového rozhrania, pretože spracúvajú hodnoty v databáze, z ktorých sú vyvodené výsledky v podobe číselných údajov a grafov. Tie sú ukazovateľom pre užívateľa systému o práci, vykonávanej na jednotlivých staniciach.

Implementácia spočívala vo vytvorení triedy *Statistics*, ktorá zabezpečuje všetky logické funkcie na spracovanie dát z databázy prostredníctvom triedy *AccessDB*. Trieda sa nachádza v balíku

WebAdmin.general. Obsahuje metódy, ktoré ukladajú výsledky databázových dotazov do atribútov triedy. Tie sa potom dajú získať príslušnými metódami.

Triedu *Statistics* používajú priamo stránky napísané v JSP, ktoré priamo zobrazujú výsledky.

3.4.1 Grafy

Grafy boli implementované s použitím vloženej knižnice *JFreeChart* a vytvorených tried pre spracovanie údajov zo systému. Tie sa nazývajú *BarChart* a *PieChart* a nachádzajú sa v balíku *WebAdmin.graphs*. Prvá menovaná slúži na vytvorenie stĺpcového grafu a jeho naplnenie hodnotami. Druhá zabezpečuje obsluhu koláčového grafu.

Spomínané triedy sú využité v jsp stránke *graph_gen.jsp*, ktorá podľa druhu vygeneruje daný graf. Stránka, ktorá chce graf zobraziť použije len HTML tag pre zobrazenie obrázku s odkazom na stránku *graph_gen.jsp*, napríklad:

**

Stránka spracuje potrebné údaje z databázy prostredníctvom triedy *Statistics* a vytvorí graf, ktorý prevedie na obrázok vo formáte *PNG*.

Obrázok 3.2: Príklad grafu v sumarizačných štatistikách

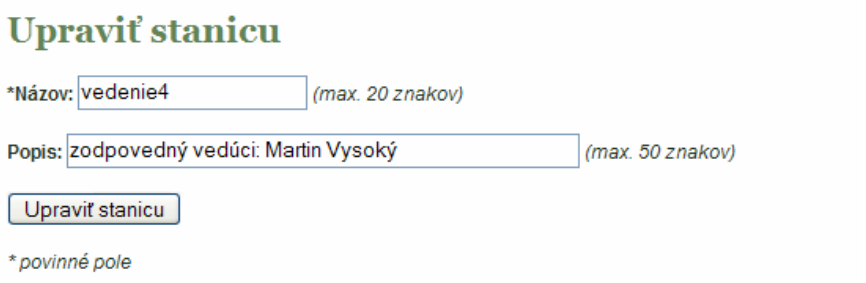
3.5 Popis činnosti systému

Informačný systém slúži na sledovanie užívateľov zo staníc, ktoré obsahujú klientskú aplikáciu a sú zaregistrovaný v databáze na servery. Aplikácia komunikuje pomocou webových služieb so serverom a informuje ho o ich činnosti, alebo odoberá správy o náplniach práce, ktoré by mal zamestnanec na stanici vykonávať. Ak budú informácie prichádzať z neznámej stanice, server nebude reagovať.

Úvodná stránka (*index.jsp*) uvíta užívateľa systému dvadsiatimi poslednými udalosťami, ktoré nastali. Každá nová udalosť sa zobrazí na vrchu tabuľky.

3.5.1 Stanica

Registrovať stanicu je možné vo formulári, ktorý sa nachádza na stránke *st_pridat.jsp*. Užívateľ musí vložiť jej názov, popis je voliteľný. Systém umožňuje tiež zmazanie a úpravy. Tie sa nachádzajú na stránke *st_upravit.jsp*. Kliknutím na prislúchajúcu stanicu sa zobrazí stránka st_*upravit2.jsp*, kde sa automaticky doplnia aktuálne dáta, tak ako je to na obrázku 3.3.



Obrázok 3.3: Upravenie stanice

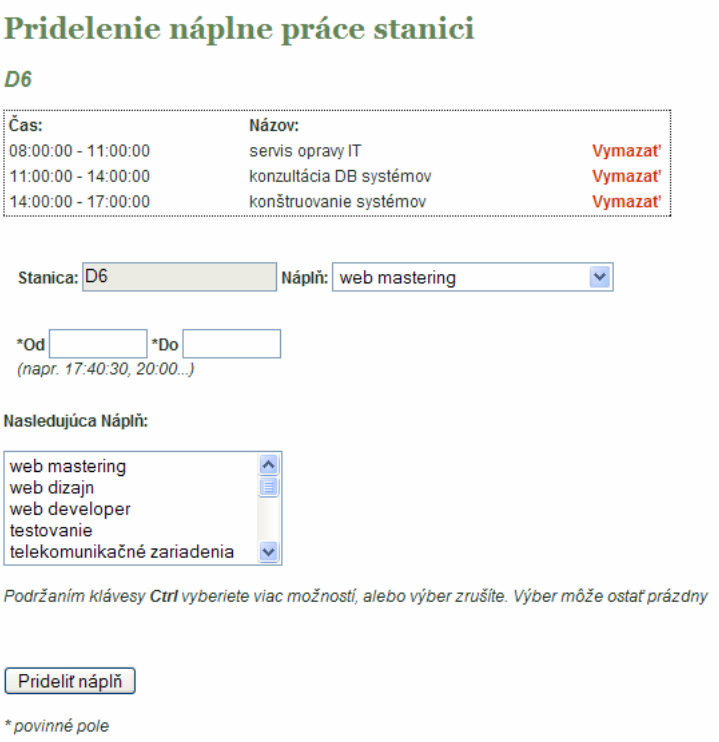
Zoznam všetkých staníc je možné potom prezerať na stránke *st_zoznam.jsp*. Kliknutím na stanicu v tabuľke sa zobrazia všetky náplne práce, ktoré má stanica zaregistrované. To zabezpečuje stránka *np_zobraznapln.jsp*.

3.5.2 Náplň práce

Práca s náplňou práce je v systéme podobná práci so stanicou. Pridávať novú je možné na stránke *np_pridat.jsp*. Upraviť ju a vymazať na stránke *np_upravit.jsp*. Celý zoznam náplne práce je dostupný na stránke *np_zoznam.jsp*.

3.5.2.1 Pridelenie náplne práce

Prideliť náplň práce môže užívateľ otvorením stránky Stanica > Prideliť náplň práce (*st_pridel.jsp*). Zobrazí sa zoznam všetkých staníc, z ktorých je potrebné vybrať jednu, ktorej plány chceme editovať. Otvorí sa stránka *st_pridel2.jsp*, znázornená na obrázku 3.4. Na jej vrchnej časti sa nachádza zoznam už zadaných náplní, ktoré nie je možné upravovať, len vymazať. Pod zoznamom je umiestnený formulár pre vloženie novej náplne práce. V ňom je potrebné zvoliť náplň, ktorú má v zadanom čase stanica vykonávať, jej začiatok a ukončenie a zo zoznamu „Nasledujúca náplň“ vybrať ľubovoľný počet náplní, ktoré by mali nasledovať.



Obrázok 3.4: Pridelenie náplne práce

Začiatok a koniec času náplne musí byť kontrolovaný, aby užívateľ nemohol zadať nesprávne údaje. To je kontrolované JavaScriptom, tak ako je uvedené v kapitole 3.1.1. Okrem toho musí systém zistiť, či uvedený časový interval sa už nenachádza, alebo neprelína s iným časom v databáze.

3.5.3 Ostatné funkcie systému

Informačný systém na práce sledovanie užívateľov obsahuje **tabuľku online užívateľov**, tak ako bolo spomenuté na začiatku kapitoly. Tá je samostatne implementovaná v súbore *online.jsp* a importovaná v každej jsp stránke s výnimkou štatistík. Zobrazenie online užívateľov funguje principiálne tak, že pri každom Id z tabuľky Stanica je vytiahnutý posledný záznam z tabuľky *Skutocnost* a pokiaľ v stĺpci *Aplikácia* nie je „*offline*“, je názov stanice v tabuľke online užívateľov zobrazený.

V systéme sa nachádza tiež trieda *DateTool* (knižnica *WebAdmin.general*), ktorá bola implementovaná na prácu s časom. Obsahuje preto metódy na získanie aktuálneho času, zistenie rozdielu dvoch časov v sekundách a pod. Jej účel spočíva hlavne pri pridávaní záznamov v tabuľke *Skutocnost*, kde sa do stĺpca *Cas* pridáva aktuálny čas.

4 Testovanie

Testovanie systému je dôležitá súčasť pri jeho vývoji, preto by sa nemala podceňovať, naopak je treba jej venovať dostatok času. Jej cieľom je zistiť chyby vzniknuté v priebehu vývoja a implementácie. To zistíme podrobením kritickými hodnotami, ideálne inou osobou.

Testovanie tohto informačné systému prebehalo celkom priaznivo, pričom bolo odhalených niekoľko chýb, ktoré boli opravené. Bola kontrolovaná najmä funkčnosť všetkých častí, ktoré sa v systéme nachádzajú. Hlavnou zložkou testovania mala byť komunikácia s klientskou aplikáciou. Tá však v čase vývoja tohto systému nebola dostupná a tak sa testovanie muselo zaobísť bez nej. V prostredí NetBeans IDE tak bolo vytvorené webové rozhranie určené pre tento účel, ktorý nahradzoval klientskú aplikáciu. Tým pádom bola otestovaná komunikácia, kde boli zistené niektoré chyby, ktoré boli tiež opravené. Nebolo však zistené, či systém spĺňa všetky požiadavky na webové služby, ktoré sú určené na komunikáciu s klientskou aplikáciou. Preto môžu byť dotiahnuté v rámci rozšírení, keď bude zhotovená.

5 Možné rozšírenia

Informačný systém bol implementovaný na základe analýzy a návrhu. V týchto fázach vývoja boli popísané všetky požiadavky zadávateľa projektu. Tiež bolo odhalených niekoľko nedostatkov, niektoré z nich sa prejavili až pri testovaní. Táto kapitola sa preto zaoberá možnými rozšíreniami informačného systému.

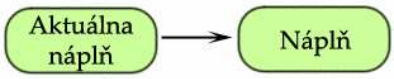
Prvým možným rozšírením by mohlo byť vloženie bezpečnostných prvkov do systému. Týka sa to či už prístupu, komunikácie s databázou, alebo komunikácie s klientom. Prístup do systému by mohol byť vyriešený užívateľským prístupovým heslom, ktorý by podľa tabuľky v databáze zabezpečoval autorizáciu.

Komunikácia s databázou je aktuálne vyriešená pomocou SQL príkazov, ktoré môžu byť napadnuté útočníkmi útokom, ktorý sa nazýva „*SQL Query Poisoning*“ a pod. To je možné vyriešiť nahradených terajších príkazov parametrizovanými príkazmi.

Komunikácia s klientskou aplikáciou je tiež z bezpečnostného hľadiska na veľmi nízkej úrovni. Je to spôsobené najmä tým, že identifikácia stanice predstavuje použitie jej názvu vo webovej službe ako parameter. To znamená, že sa hocikto môže vydávať za nejakú stanicu a zaznamenávať jej udalosti do systému. Druhá podstatná vec je tá, že informácie sa prenášajú v nezašifrovanej forme.

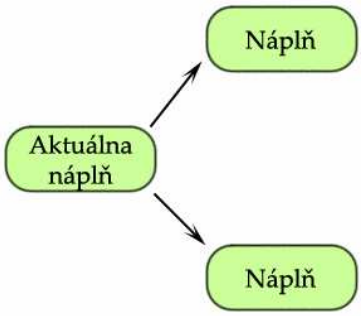
Najväčším rozšírením a prínosom do informačného systému by bola implementácia automatického riadenia a usmerňovania staníc pomocou nasledujúcich náplní. Tie sú v systéme momentálne implementované, ale slúžia len pre informáciu užívateľa. Prínosom by bolo, keby pomocou nich mohol systém oznamovať užívateľom, ktorí ich majú zaregistrované, správy o tom, že môžu začať prácu s danou náplňou. To len v prípade, že aktuálna náplň je ukončená. To však prináša niekoľko problémov:

- Náplň, ktorú užívateľ práve vykonáva ukazuje práve na jednu nasledujúcu náplň, po jej ukončení sú oboznámené všetky stanice, ktoré ju majú vykonávať. (Obrázok 5.1)



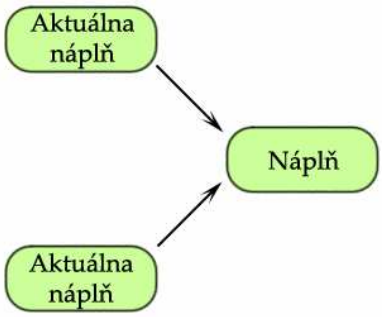
Obrázok 5.1: Aktuálna náplň odkazuje na jednu nasledujúcu náplň

- Náplň, ktorú užívateľ práve vykonáva ukazuje na viac nasledujúcich náplní, po jej ukončení sú oboznámené všetky stanice, ktoré majú jednotlivé stanice vykonávať. (Obrázok 5.2)



Obrázok 5.2: Aktuálna náplň odkazuje na viac nasledujúcich náplni

- Náplne, ktoré viacerý užívatelia práve vykonávajú ukazujú práve na jednu náplň, po ich ukončení sú oboznámené stanice, ktoré ju majú vykonávať. (Obrázok 5.3)



Obrázok 5.3: Aktuálne náplne odkazujú na jednu nasledujúcu náplň

Najväčší problém je práve v tom, že každú náplň môže mať zaregistrovanú viac užívateľov.

Ďalším možným rozšírením by mohlo byť implementovania kalendára, podľa ktorého by sa zadávali náplne pre stanice. Takže plány na každý deň by boli iné pre jednotlivé stanice. Zmenilo by to tiež oznamovanie skutočnosti stanice, ktorá nastala.

6 Záver

K úspešnej práci každého vedúceho alebo manažéra patrí prehľad o kvalite a zohranosti jeho tímu, tzn. zamestnancov, ktorí pracujú pod jeho vedením a spoločne tým dosahujú určité výsledky. Pri väčšom počte to však býva problém, preto je výhodné mať systém, ktorý by spoľahlivo zaznamenával pracovnú činnosť všetkých zamestnancov a tým by zlepšil niekoľko faktorov úspešného podnikania. Jedným z nich môže byť aj nepopierateľnosť odpracovaného času, ale aj iné, vopred spomenuté v úvode.

Náplňou informačného systému, ktorý som vyvíjal bolo sledovanie práce užívateľov prostredníctvom webových služieb, ktoré zabezpečujú komunikáciu. Aplikácia bola implementovaná ako webové rozhranie, ktoré obsahuje užívateľské rozhranie pre jednoduché ovládanie funkcií systému. Požiadavky zadané zadávateľom prešli postupne cez analýzu, návrh a nakoniec implementáciu systému. V týchto fázach sa našlo niekoľko nejasností a niekoľko optimalizácií špecifikácie zadania, ktoré boli priebežne prispôsobované zadávateľom smerom k užívateľovi systému.

Systém je navrhnutý najmä pre stredné a väčšie firmy, obzvlášť pre ich oddelenia. Jeho práca úzko súvisí s klientskou aplikáciou, ktorá v dobe vývoja systému nebola k dispozícií, takže komunikácia s ňou môže byť prípadne upresnená. Bez nej je však systém do istej miery nepoužiteľný.

V súčasnej dobe je aplikácia pripravená na používanie. Chyby, ktoré neboli odhalené sa môžu časom prejaviť. Takisto je nutné program prispôsobiť na mieru konkrétnej firme, ktorá by ho chcela používať. Kvalitný software je tiež podporovaný kvalitným servisom.

Ďalší vývoj samotného programu by mohol byť v prvom rade týkajúci sa vecí, ktoré boli spomenuté v kapitole možných rozšírení (kapitola 5). Z môjho vlastného pohľadu by pri zväčšení projektu do väčších rozmerov, tzn. pridaním viacerých častí a pod., pomohlo lepšie oddelenie prezentačnej, logickej a dátovej vrstvy. Projekt bol vytvorený bez hlbších predchádzajúcich skúsenosti a znalostí s návrhom a implementáciou podobnej aplikácie s použitím daných technológií. S tým samozrejme súvisí dĺžka času vývoja projektu, ktorá by mohla byť pri dostatočných skúsenostiach kratšia. Preto je, samozrejme, vhodné dobré načasovanie a rozdelenie všetkých častí vývoja projektu.

Prínos by mohol byť markantný v oblasti skvalitnenia práce zamestnancov v každej firme. Tu však už je zainteresovaných viac firiem, ktoré poskytujú komplexné riešenia pre túto činnosť na základe podobného projektu. Tie vytvárajú prirodzenú konkurenciu, preto by bolo vhodné rozširovať projekt a tým skvalitňovať služby, ktoré ponúka.

Literatúra

[1] Kevin Mukhar, Chris Zelenak, James L. Weaver, Jim Crume: *Beginning Java EE 5: From Novice to Professional*. New York, 2006.

[2] Barry Burd: *JSP: Java Server Pages, Podrobný prúvodce*. Praha, 2003

[3] W3C: *Web Services Architecture*. Dokument dostupný na URL: http://www.w3.org/TR/ws-arch/ , 12.4.2008

[4] Wikipedia, The Free Encyclopedia: *Web Service*. Dokument dostupný na URL: http://en.wikipedia.org/wiki/SOAP, 18.4.2008

[5] Wikipedia, The Free Encyclopedia: *Web Service*. Dokument dostupný na URL: http://en.wikipedia.org/wiki/Java_API_for_XML_Web_Services, 19.4.2008

[6] Sun Microsystems Documentation: *Creating a Simple Web Service and Client with JAX-WS*. Dokument dostupný na URL: http://docs.sun.com/app/docs/doc/8193669/bnayn?l=en&q=JAX&a=view, 19.4.2008

[7] Sun Microsystems Documentation: *Distributed Multitiered Applications*. Dokument dostupný na URL: http://docs.sun.com/app/docs/doc/819-3669/bnaay?l=en&q=JAX&a=view, 20.4.2008

[8] Interval: *Java Servlets - predstavenie technológie*. Dokument dostupný na URL: http://interval.cz/clanky/java-servlets-predstavenie-technologie/, 25.4.2008

[9] Sun Microsystems Documentation: *Java Server Pages, Standard Tag Library*. Dokument dostupný na URL: http://java.sun.com/products/jsp/jstl/reference/docs/index.html, 26.4.2008

[10] W3C: *XHTML*. Dokument dostupný na URL: http://www.w3.org/TR/xhtml1/, 28.4.2008

[11] W3C: *CSS*. Dokument dostupný na URL: http://www.w3.org/Style/CSS/, 29.4.2008

[12] Robert Scheldon: *SQL: A Beginner's Guide, Second Edition*. California, 2003

[13] Sun Microsystems: *MySQL 5*. Dokument dostupný na URL: http://dev.mysql.com/doc/refman/5.0/en/index.html, 1.5.2008

[14] JFreeChart project, Dokument dostupný na URL: http://www.jfree.org/jfreechart, 1.5.2008

Zoznam príloh

Príloha 1. CD