Univerzita Karlova v Praze

Filozofická fakulta

Ústav informačních studií a knihovnictví

Bakalářská práce

2007 **Štěpán Petrů**

Univerzita Karlova v Praze

Filozofická fakulta

Ústav informačních studií a knihovnictví

Studijní program: informační studia a knihovnictví

Studijní obor: informační studia a knihovnictví

Štěpán Petrů

Analýza, návrh a implementace databázového systému pro reálnou agendu s využitím relační databázové technologie

Bakalářská práce

Praha 2007-05-23

Vedoucí bakalářské práce: PhDr. Helena Kučerová

Oponent bakalářské práce:

Datum obhajoby:

Hodnocení:

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a že jsem uvedl všechny použité informační zdroje.

V Praze, 23. května 2007

……..

podpis studenta

Identifikační záznam

Petrů, Štěpán. *Analýza, návrh a implementace databázového systému pro reálnou agendu s využitím relační databázové technologie [Analysis, concept and system implementation for the key point of the agenda with the utilization of relation database technology]*. Praha, 2007-05-22. 42 s. Bakalářská práce. Univerzita Karlova v Praze, Filozofická fakulta, Ústav informačních studií a knihovnictví. Vedoucí diplomové práce Helena Kučerová.

Abstrakt

Bakalářská práce se snaží popsat již existující rezervační systém Bizzy firmy Smarcoms a dokumentovat klíčové části pomocí nástrojů jazyka UML. Úvodní kapitola obsahuje obecnou charakteristiku vývojového prostředí pro rezervační a informační systémy a zároveň rekapituluje historii aplikace od původního návrhu až po současný stav. Druhá kapitola zahrnuje obecný popis datového modelu, nejdůležitější objekty pro vykreslování rozvrhů služeb a použití tří vrstev vzoru Model-View-Controller. Třetí kapitola se podrobněji zabývá jednotlivými případy užití a popisuje jejich konkrétní implementaci v rámci celého systému. V závěru se tato práce snaží vyjmenovat hlavní chyby, kterých se vývojáři v rámci celé doby fungování systému dopustili a zároveň poskytnout doporučení pro další vývoj a směřování systému.

Klíčová slova

aplikace, systém, návrh systému, UML, the Unified Modeling Language, objektově orientované programování, datové modelování, O/R mapping, reverzní inženýrství

Obsah

Předmluva

Tuto práci jsem si vybral, protože problematika návrhu a vývojářská dokumentace softwaru je mi blízká - v současnosti pracuji jako softwarový vývojář a dokumentace je jeden z nejdůležitějších zdrojů informací. Během vývoje se vývojář setkává s dokumentací od začátku návrhu aplikace až po vytváření uživatelské dokumentace pro uživatele.

Fakt, že pracuji jako programátor ve firmě Smarcoms, která Bizzy vlastní, mě, spolu s neexistencí vývojářské dokumentace, vedla k vybrání si právě tohoto systému jako subjektu práce.

Touto prací jsem se snažil nejenom splnit zadaný úkol, vytvořit dokumentaci pro systém Bizzy, ale i získat nové znalosti, které doufám dokáži využít i ve svém zaměstnání. Jazyk UML je totiž v dnešní době nejvyužívanější nástroj pro návrh aplikací, a tak se s ním budu v budoucnu určitě setkávat a tak každá další zkušenost se mi jistě bude hodit.

Pro vypracování byly použity interní zdroje firmy Smacoms, které bohužel nejsou plně k dispozici veřejně. Bohužel právě informace z nich čerpané byly jedny z nejdůležitějších, protože poskytují nejbližší pohled na systém, předpoklady jeho vzniku a na jeho současný stav.

Na tomto místě bych chtěl poděkovat mé vedoucí práce, která mi byla neocenitelnou oporou a jen díky její trpělivosti jsem byl schopen práci dokončit. Musím také poděkovat pracovníkům firmy Smarcoms, kteří mi poskytli podporu a možnost přístupu k interním informacím. Dále je na místě poděkovat i zákazníkům této firmy, kteří denně používají systém Bizzy, protože bez nich bych neměl možnost toto téma zpracovávat.

Seznam použité literatury, který obsahuje všechny mnou použité citace v práci, je řazen abecedně. Všechny bibliografické záznamy jsou v souladu s normami ČSN ISO 690 a ČSN ISO 690-2.

1 Úvod

Návrhová dokumentace při vývoji softwaru patří mezi velmi důležité součásti jeho tvorby. Dokumentace by měla odhalit úskalí systému hned na začátku a zjednodušit komunikaci se zákazníkem, protože při použití standardních nástrojů je čitelný i pro ty zákazníky, kteří nemají technické znalosti.

Bohužel návrhová dokumentace nebyla během vývoje systému Bizzy vytvářena, vytvářet jí zpětně není zrovna jednoduché, protože tvůrce musí projít celým systémem a zpětně získávat o něm informace. Jde sice použít nástroje pro reverzní inženýrství, ale ne vždy jsou optimální cestou, protože žádný takovýto program není dostatečně univerzální, aby nedošlo k zásadní ztrátě informace.

Jedním z nejčastěji využívaným nástrojem současnosti pro návrh softwarových systémů je jazyk UML (the Unified Modeling Language). UML je systém notace, který se skládá ze skupiny diagramů a představuje standard, poskytující návrháři nástroj pro srozumitelnou komunikaci s klientem, programátory i všemi ostatními, kdo jsou na vývoji zainteresováni . [SCHMULLER 2001] Veškeré diagramy vyjadřují, co má systém dělat a ne, jak to má dělat.

UML obsahuje 8 základních diagramů, na tomto místě si popíšeme diagramy, které jsou použity v této práci:

- **Diagram případů užití,** který slouží k vyjádření funkcí systému a jejich vazeb s uživateli. Dále specifikuje i vztahy mezi jednotlivými činnostmi. Tento diagram se většinou využije na začátku vývoje, kdy je konzultován s budoucím uživatelem a kdy jsou stanoveny funkce aplikace.

- **Diagram tříd** vyjadřuje vztahy mezi jednotlivými třídami modelu aplikace. Tento diagram je užíván programátory a může již mít vazbu na konkrétní kód.

- **Diagram interakcí** zachycuje spolupráci mezi jednotlivými částmi systému v kontextu času.

- **Diagram činností** je zobrazením činnosti, ke které dochází v rámci některého případu užití nebo typické chování některého objektu. Častokrát poskytují podklad pro tvorbu algoritmu.

2 Popis systému Bizzy

2.1 Systém Bizzy

Bizzy je samoobslužný systém určený pro sportovní centra k zajištění rezervačních služeb a komunikaci se zákazníky, vytvářený a prodávaný firmou Smarcoms. Je navržen tak, aby obsluze sportoviště zjednodušil a zpřehlednil práci s rezervacemi a zákazníkům umožnil jednoduché rezervovaní služeb.

Ilustrace 1: Logo aplikace Bizzy

2.1.1 Samoobslužný systém

Systém Bizzy je schopen podle nastavení sám měnit stavy rezervací a dle toho měnit uživatele, jeho status, počítat jeho útratu atd. Dále je schopen automaticky upozorňovat na blížící se rezervace a to jak zákazníky (email, SMS), tak i obsluhu (email, SMS, pop-up okno).

2.1.2 Definice sportovního centra

Sportovní centrum je zařízení, které jako službu poskytuje nějaké sportoviště, většinou se jedná o sportovní kluby (typicky squash, tenis, bowling). Častokrát je tato činnost spojena i s drobným prodejem potravin (občerstvení) nebo sportovních potřeb. Ač je Bizzy primárně určen pro sportovní centra, je použitelný i pro jiné provozovny, které poskytují služby podobného charakteru (pronájem nějakého prostředku/zdroje, například pronájem místností, salónků).

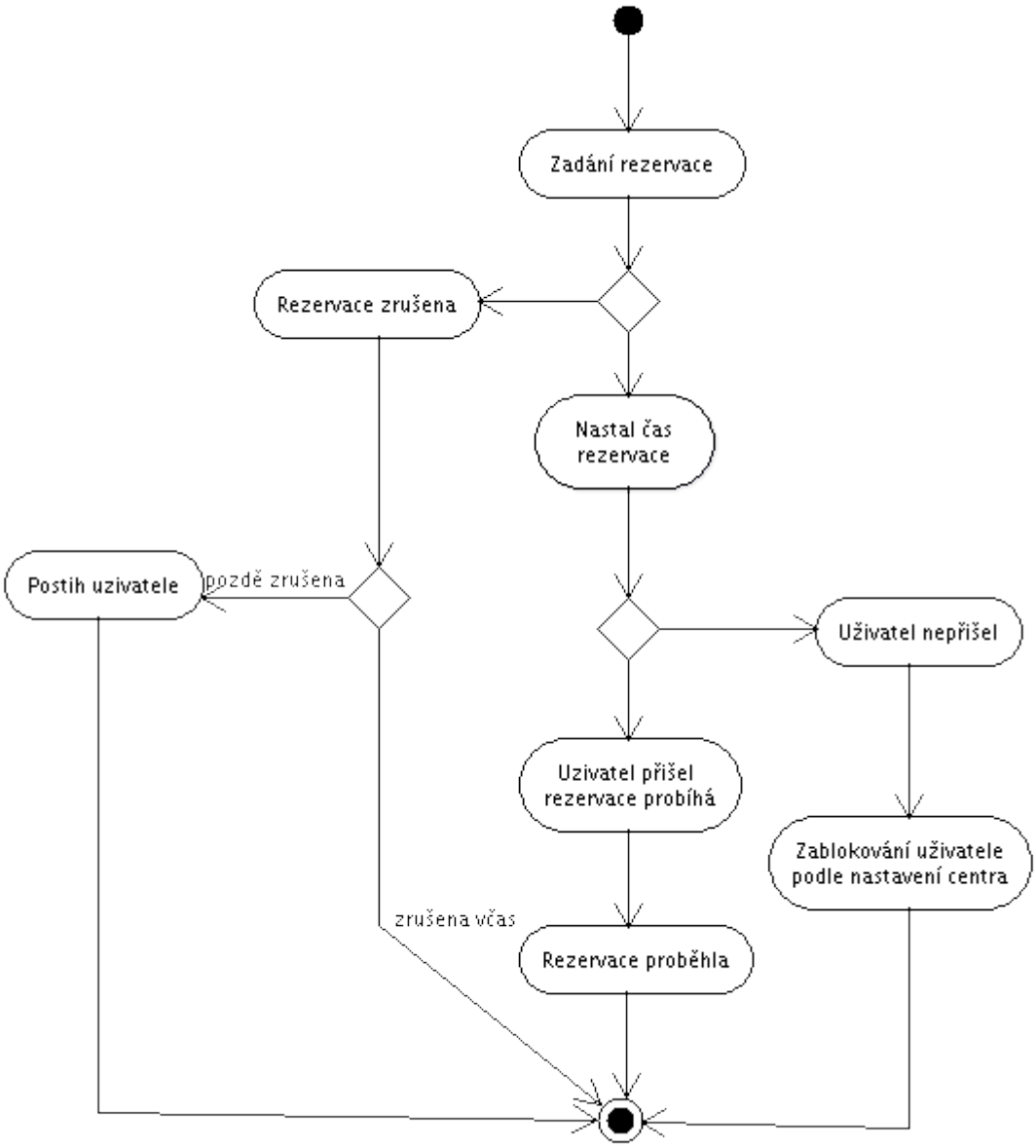
Vlastní centrum (občas i vlastník centra) bývá interně označován jako klient.

2.1.3 Definice rolí uživatelů

V centru se pohybují dva druhy uživatelů: zákazník, který využívá služby poskytované centrem, dále je to obsluha centra. Obsluhu lze dále rozdělit na operátory (běžná obsluha centra - zaměstnanci) a administrátory (dále nazýváni jako super-operátoři, kteří odpovídají vedoucím pracovníkům centra).

2.1.4 Definice rezervace

Rezervací je myšleno zamluvení si určité služby na danou dobu v určitém sportovním centru. Rezervace je závazná a pokud uživatel nesplní podmínky a nedostaví se, je možné z ní vyvodit důsledky (zablokování uživatelského účtu, propadnutí kreditu nebo bodů věrnostního programu). V případě proběhnutí rezervace může na základě nastavení centra být uživatel odměněn body do věrnostního programu.



Ilustrace 2: Základní princip fungování rezervování

2.1.5 Způsob distribuce

Systém je klientům poskytován buď jako služba, kdy je celý servis zajišťován

prodejcem, nebo jako lokální instalace provedená na počítači klienta. V případě lokální instalace si klient může vybrat, zda si chce systém spravovat sám, nebo má možnost podepsat servisní smlouvu. V případě distribuce jako služby klient nemá jiné náklady než klientský počítač vybavený prohlížečem a Internetovým připojením – celá aplikace běží na serveru firmy Smarcoms a je tak dosažitelná z kteréhokoliv počítače připojeného do sítě Internet.

2.2 Použité technologie

Systém Bizzy je třívrstvá webová aplikace typu MVC (model-view-controller) vyvinutá pro platformu Java. Jako základ je použito aplikačního rámce Jakarta Struts verze 1. Tento framework poskytuje základní funkčnost pro vrstvy controller a view. Pro vrstvu modelu je v současné době použito rámce Apache ObjectRelationalBridge, který vytváří vrstvu mezi relacemi a objekty užívaných interně v aplikaci (toto se nazývá Object – Relation mapping). Oba aplikační rámce jsou šířeny The Apache Software Foundation (dále ASF) pod svobodnou licencí APL (Apache License).

Aplikace běží za použití Sun JDK 1.5 a vyšší na JSP/servlet kontejneru poskytujícím implementaci specifikace Servlet API 2.4+ a JSP API 2.0+ jako je Apache Tomcat 5.5 (také vyvíjený ASF).

Jako datové úložiště je použit databázový systém MySQL 4, ale díky použitým technologiím není problém využít i jiného, omezení je pouze možnostmi O/R mapovacího rámce – veškerá komunikace probíhá právě přes něj.

Na jedné aplikaci lze teoreticky provozovat neomezený počet center.

2.3 Historie systému Bizzy

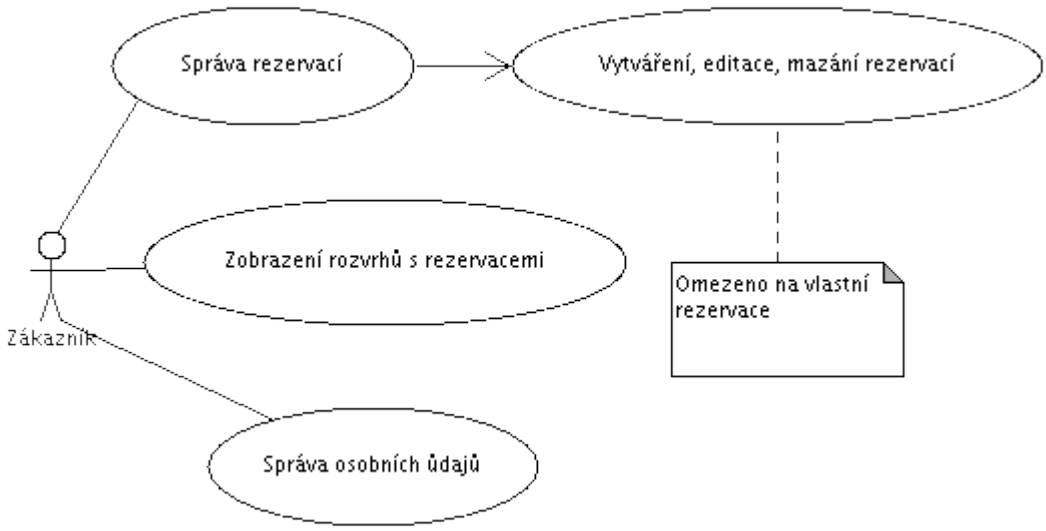
Systém Bizzy vychází z návrhu aplikace Raye Gibsona určené pro řízení firemních zdrojů a vyvíjené pro firmu T-mobile, která tento systém nakonec nekoupila. Firma Smarcoms v roce 2004 tento návrh a demoverzi zakoupila se záměrem stávající kód upravit na rezervační systém pro sportovní centra.

V době koupě původního kódu neexistovala v tomto segmentu trhu v České republice žádná konkurence, a tak se počítalo s jednoduchým systémem se základními funkcemi a rychlým vývojem. První předprodukční verze vznikla na podzim roku 2004

ve spolupráci s asociací ASARC (Asociace majitelů a provozovatelů squash a ricochetových center). Touto asociací byl zprvu systém Bizzy propagován a doporučován. Funkční systém přišel na trh v době, kdy nebyl velký zájem o služby dostupné online. Tato situace se během roku 2005 rychle změnila - došlo k nárůstu poptávky po službách dostupných v rámci sítě Internet a z ní vyplývající i nárůst požadavků sportovišť na online rezervační systém. Na přelomu roku 2006 a 2007 prošel systém rozsáhlou optimalizací výkonu a úpravou uživatelského rozhraní. V současnosti aplikace Bizzy dominuje na českém trhu a firma získává klienty v zahraničí (Slovensko a Německo).

2.4 Současné funkce systému

V dnešní době systém Bizzy poskytuje široké možnosti pro rezervování sportovišť a jiných zdrojů (zároveň se v současnosti pracuje na verzi pro správu firemních zdrojů). U sportovišť může klient umožnit rezervace jak pro zdroje jako jsou tenisové kurty, tak pro cvičení (např. dnes velmi oblíbený spinning). Bizzy ovšem není jenom rezervační systém, ale plní i funkce marketingového nástroje Customer relationship management (CRM), protože poskytuje funkce pro komunikaci se zákazníky.



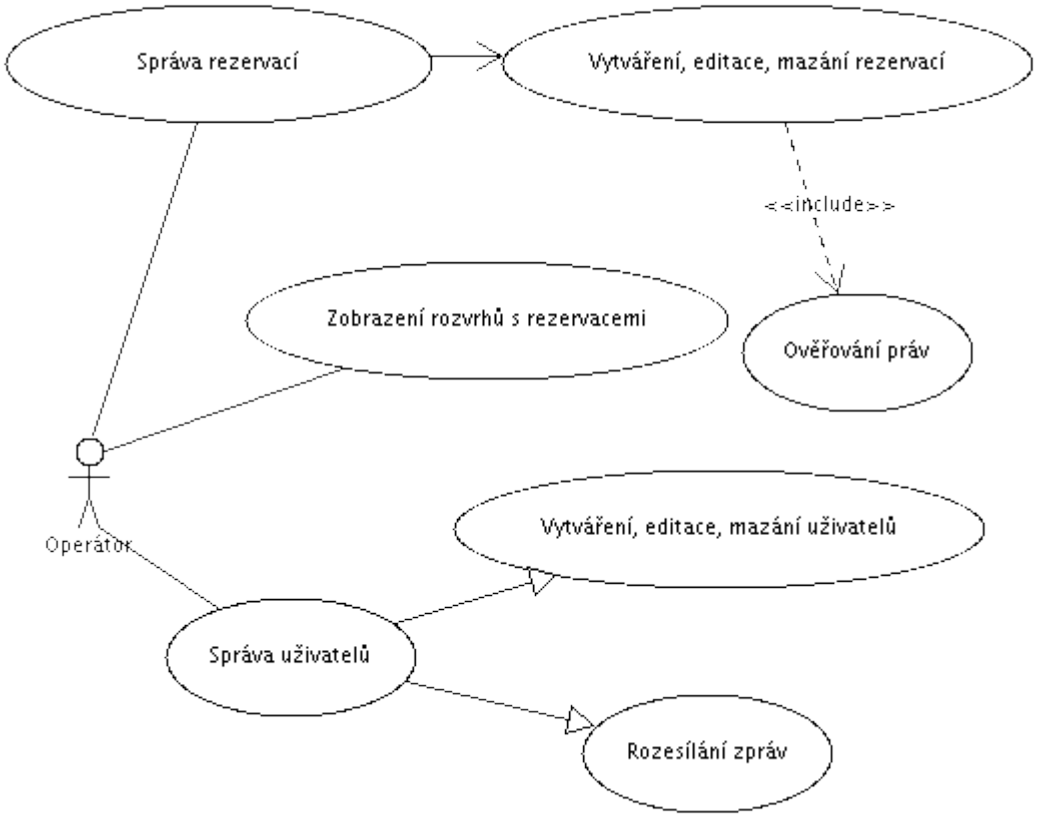
Ilustrace 3: Základní funkce uživatele - zákazník

Systém je v dnešní době lokalizovaný v češtině, slovenštině, angličtině a němčině. Jednotlivé jazykové mutace jsou použity na základě nastavení Internetového prohlížeče (program si jazyk sám detekuje). Přidání dalších jazyků je na přání zákazníka možné a díky plné interní podpoře kódování UNICODE je možné přidat i jazyky používající jiné abecedy než je latinka.

2.4.1 Základní funkce

Funkce pro práci s rezervacemi: systém umožňuje dostatečně pružné zacházení s rezervacemi. Rezervace mohou být na registrované uživatele, ale také na nově příchozí, kterým rezervaci vyřizuje obsluha. Je možné zadávat opakující se rezervace i rezervace na více zdrojů zároveň. Cena může být počítána podle uživatelovy cenové hladiny (cenová hladina může být přiřazena i přímo v rezervaci). Uživatelé mohou být na rezervaci upozorněni předem prostřednictvím e-mailu nebo SMS.

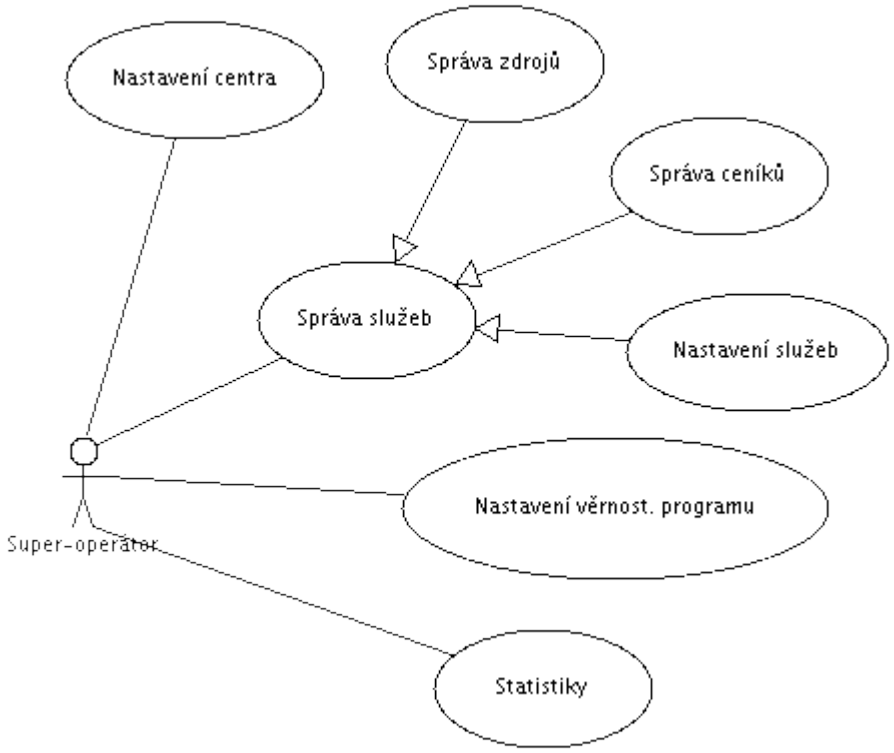
Práce s uživateli: systém dovoluje základní práci s uživateli včetně zasílání novinek e-mailem, nebo skrze SMS bránu (momentálně na mobilní telefony v České republice, Slovenské republice a Německu). Dále je možné uživatelům přiřazovat role a statusy a řadit je do různých cenových kategorií či skupin pro vyhledávání.



Ilustrace 4: Základní funkce uživatele - operátor

Vlastní nastavení centra: klient si podle verze (viz. níže) může vytvořit určitý počet služeb, které poskytuje. Každá služba má poté určitý počet zdrojů, které fyzicky odpovídají sportovištím (jednotlivé kurty, tělocvičny). Zdroje mohou mít různé otevírací doby a to i v jednotlivých dnech v týdnu. Služby mohou být před uživateli volitelně skryty a používat je čistě vnitropodnikově (někteří klienti takto interně řídí stoly v restauraci).

Cenové nastavení: každé službě lze nastavit ceníky pro jednotlivé dny v týdnu a to i



Ilustrace 5: Základní funkce uživatele - super operátor

v několika pásmech během jednotlivých dnech. Je také možné využít funkci „dodatečné náklady“, umožňující připočítávat do ceny rezervace jednotnou částku a to buď za rezervaci nebo za návštěvníka. Platby za rezervace je možné provádět buď v hotovosti, z uživatelova kreditního konta nebo předem zakoupenou permanentkou. Je možné regulovat uživatelské rezervace podle stavu kreditu nebo permanentky. Veškeré pohyby na těchto účtech jsou zaznamenávány, a proto je možné jejich vyhledání.

Statistika: systém poskytuje základní statistické funkce jako je využití zdrojů podle času a peněz, obsazení během dne či souhrny rezervací jednotlivých uživatelů.

Věrnostní systém: Bizzy poskytuje možnost odměňovat zákazníky za jejich navštěvování podniku a to podle utracených peněz nebo počtu intervalů, které v centru strávili. Odměna je nastavena jako sleva z rezervace v procentech. Další možností je takzvaný bodový systém, kdy je uživatel odměňován body za návštěvy, nákupy, přivedení nového zákazníka atd., které může v centru utrácet. Systém je nastaven tak, aby bylo čistě na centru, jakým způsobem bude tato útrata probíhat.

2.4.2 Nadstandardní funkce:

Zálohovací software: systém je dodáván s jednoduchým zálohovacím nástrojem,

který v určitých časových intervalech stahuje do počítače ve středisku rozvrh pro aktuální týden pro případ výpadku Internetového připojení. Tyto informace v případě poruchy umožňují obsluze zjistit aktuální stav rezervací a obsazenost centra.

Integrace s pokladním a skladovým systémem FOOD6 a Prudent: v dnešní době Bizzy obsahuje integraci (použit je Simple Object Access Protocol) se systémem FOOD6 firmy ALTO, poskytující široké možnosti pro vedení pokladen a skladového managementu včetně poskytování podkladů pro účetnictví. Systém FOOD6 může být v případě využití integrace použit pro ovládání rezervačního systému, ovšem hlavním úkolem integrace je sjednocení kreditního systému. Druhým pokladním programem, se kterým Bizzy spolupracuje, je Prudent firmy Prudent company. Tato integrace by měla být nasazena během března/dubna roku 2007. Systém Prudent je jednodušší než FOOD6, a proto je jeho cílová skupina spíše v menších centrech.

2.5 Budoucí funkce a návrhy na funkčnost

Práce budou také pokračovat na zefektivnění kódu a jeho optimalizaci. V nejbližší době bude muset dojít k úpravě (možná i nahrazení) vrstvy modelu, protože současně použitý aplikační rámec má problémy s výkonem.

Možnost platit platební kartou za kredit a permanentky a s tím související zvýšení bezpečnosti komunikace pomocí šifrované komunikace mezi serverem a prohlížečem.

Funkce provozoven, která by oddělila jednotlivé služby do skupin podle umístění, či jiných kritérií dle přání klienta (např. společnost ABC provozuje posilovnu a zároveň halu na tennis, tato pracoviště ovšem mají jistou část autonomie a obsluha jednoho by neměla být schopná ovládat druhý provoz, avšak zákaznické účty platí v obou).

Další plánovanou funkcí na žádost klienta je rezervování pomocí SMS. Tato možnost však má nižší prioritu, protože naráží na větší množství překážek. Dosud není rozhodnuto o obecném principu a formátu zpráv.

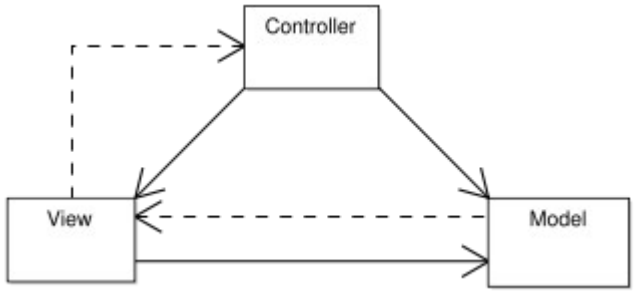
Integrace s další aplikací firmy Smarcoms, která slouží pro ovládání osvětlovací techniky sportovišť (nebo na spouštění solárií). Tato aplikace bude schopna komunikovat se serverem a na základě zaslaných dat automaticky spouštět a vypínat tuto techniku.

Rozšíření statistik o možnost tvoření reportů ve formátech PDF, Microsoft Office a některých jednoduchých strojově čitelných formátech (například csv).

2.6 MVC

Vzor Model-view-controller (dále MVC), někdy také označovaná jako Model-2 (Model-1 pouze slučuje vrstvy View a Controller), je softwarová architektura, která rozděluje datový model aplikace, uživatelské rozhraní a řídicí logiku do tří nezávislých komponent tak, že modifikace některé z nich má minimální vliv na ostatní. [Wikipedia 2007]

MVC je často chápán jako návrhový vzor, nicméně se týká architektury aplikací mnohem více než klasický návrhový vzor. Tudíž může být užitečný pojem architektonický vzor nebo možná agregační návrhový vzor.



Ilustrace 6: Základní vztahy v MVC.

Model: zodpovídá za správu vnitřní struktury a data.

View: zodpovídá za vnější prezentaci vnitřní struktury a dat.

Controller: zodpovídá za kontrolu průběhu a stavu uživatelského vstupu.

Koncept MVC se užívá především jako základní architektura webových aplikací. U složitějších aplikací je to způsob, jak zajistit jejich flexibilitu a spolehlivost i při častých změnách a jejich rychlém vývoji. Tento vzor bývá často pochopen nesprávně a mnoho aplikací tvrdí, že jej využívají, ačkoliv fakticky nesplňují základní požadavek, kterým je oddělení aplikační a prezentační logiky (což odpovídá spíše použití Modelu 1). V současnosti existuje mnoho aplikačních rámců a knihoven, které poskytují základní implementaci MVC a to jak pro webové aplikace, tak pro aplikace desktopové.

2.7 Struts

Struts je otevřený aplikační rámec určený pro vytváření webových aplikací na platformě Java vyvíjený nadací Apache, který je v aplikaci Bizzy využíván jako základ implementace MVC. Struts poskytuje komponenty View a Controller. [CAVANESS

2003]

Rámce jako je Struts umožňují programátorům zaměřit se na implementaci vlastní logiky aplikace a odstraňují nutnost implementovat infrastrukturu MVC.

2.8 Definice modelu v aplikaci Bizzy

Model je sada objektů plněná a ukládaná do databáze. Tyto objekty jsou dále zasílány pohledové vrstvě, která je interpretuje do uživatelského rozhraní. Tomuto druhu objektů se říká Data Transfer Object (DTO) nebo Value Object.

V aplikaci Bizzy lze v základu najít 4 základní třídy, které vyjadřují 4 základní entity systému. Tyto třídy jsou používání jako model pro persistenci pomocí rámce OJB, které překládá vazby mezi objekty na relace v databázi.

JavaBean je třída napsaná v jazyce platformy Java původně vyvinuta jako komponentní model pro vývoj grafických aplikací. [Sun Microsystems 2007] V dnešní době jsou používány jako obecný kontejner pro data a stavy aplikace. Každá JavaBean by měla být znovu použitelná a musí řídit několika konvencemi:

- Musí obsahovat bezparametrický konstruktor
- členské proměnné musí být dostupné pomocí get* (nebo is* u typu boolean) a set* metod
- objekt musí být serializovatelný (je možné bez problému uložit na disk jeho aktuální stav)

2.8.1 ObjectRelationalBridge – OJB

OJB je aplikační rámec pro mapování objektů na databázové relace (dále ORM), umožňující transparentní ukládání objektů do databáze. Rámec je vyvíjený nadací Apache a poskytovaný pod otevřenou licencí. Bohužel v současnosti byl vývoj zastaven. [The Apache Software Foundation 2005]

ORM je technika, která umožňuje pracovat s běžnou relační databází jako s objektovou. Nejčastěji se využívá v programech napsaných pro objektově orientované prostředí (kterým je například Java). Díky ní je zjednodušen přístup k datům v databázi,

protože se s nimi pracuje přímo v programovacím jazyku bez nutnosti ručního převodu a psaní SQL.

Mezi hlavní vlastnosti OJB patří:

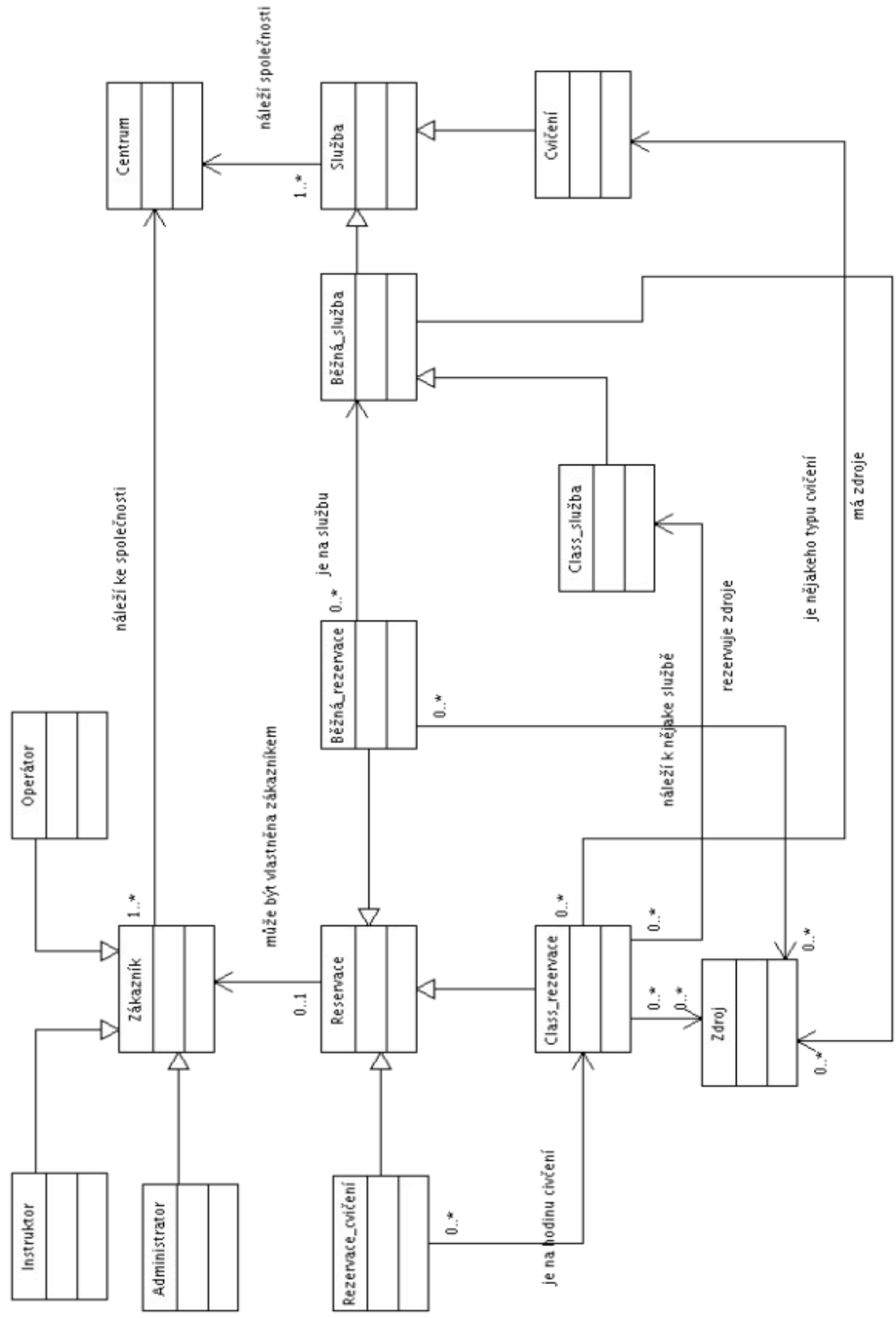
- transparentní persistence – není potřeba používat konkrétní třídy, pouze je nutné dodržovat vzor JavaBeans
- podpora všech druhů asociací (1:1, 1:n, m:n)
- podpora standardu ODMG 3.0 API a objektového dotazovacího jazyka OQL (bohužel oba standardy nejsou zcela implementovány) pro persistenci objektů
- podpora Lazy inicializace objektů pomocí proxy objektů (asociace objektu jsou z databáze získávány až ve chvíli, kdy jsou opravdu potřebné)
- použití XML pro konfiguraci mapování objektů na relace, s možností změny konfigurace za běhu aplikace.

S rámcem Bizzy komunikuje pomocí dvou tříd, které implementují 2 rozhraní definující funkce potřebné v aplikaci. V celé aplikaci se používá volání přes rozhraní kvůli zjednodušení případné změny rámce/implementace (není třeba měnit stávající kód volání databázových služeb, pouze implementaci rozhraní).

- **Rozhraní IBizzyService**: definuje metody používané pro vkládání, upravování a získávání objektů souvisejících s běžným provozem (například rezervace, uživatelé) z databáze.
- **Rozhraní ISetupService:** definuje metody určené pro nastavení systému => vztahuje se na objekty definující služby centra.

Tento přístup by v budoucnu měl být nahrazen návrhovým vzorem Data Access Objects, kdy pro každý persistentní objekt existuje jedna třída pro přístup k databázi. [Sun Microsystems [2007]] Důvodem pro přechod je zvýšení přehlednosti kódu, protože v současnosti jsou obě rozhraní opravdu veliká (současná implementace rozhraní IBizzyService pro OJB má přes 7 tisíc řádků).

2.8.2 Základní třídy modelu



Ilustrace 7: Abstraktní datový model

2.8.3 Třída Centrum

Základní jednotkou systému je centrum, které odpovídá jednotlivému sportovnímu centru. Tato jednotka obsahuje všechna nastavení společná pro centrum a dále návaznost na služby a uživatele. Obsahuje obecné informace o podniku jako jsou adresa a kontakt na centrum. Dále obsahuje volby jako jsou využívání kreditu/permanentek, možnost náhradníků pro cvičení.

2.8.4 Třída Služba

Služba je vyjádřením činnosti (prostoru, služby) poskytované podnikem zákazníkům.

Každá služba obsahuje nastavení ceny. Pro službu lze mít několik ceníků, které odpovídají časovému období v roce, zahrnující nastavení pro jednotlivé cenové hladiny uživatelů. Ceny se mohou lišit i v průběhu dne.

Služby mají zdroje, které odpovídají konkrétnímu prostoru/prostředku, který je nabízen. Každému zdroji lze nastavit čas, po který je k dispozici a to pro každý den v týdnu zvlášť.

Služby jsou trojího druhu:

- Běžná služba: umožňuje na jeden časový úsek na jednom zdroji rezervovat pouze jednoho zákazníka. Tento druh odpovídá službám jako je pronajímání tenisových kurtů a obecně prostor. Příklad: squash.

- Class-služba: umožňuje tvořit vícenásobné rezervace, tyto rezervace odpovídají cvičení nebo kurzu, který si zákazníci rezervují, zákazník není omezen časem, ale počtem volných míst. Příklad: aerobic.

- Cvičení: vlastní druh cvičení, nemá fyzické zdroje, slouží pouze pro vyjádření bližšího popisu hodiny cvičení. Zároveň u rezervací na cvičení určuje cenu rezervace.

Mezi Class-službou a cvičením je vztah, který vyplývá z jejich funkce. Class-služba vyjadřuje nadřazenou skupinu. V případě centra poskytujícího službu Aerobic je rozložení mezi class-službu a cvičení následující:

- Aerobic je class-služba, která má vazbu na zdroje, ale nedefinuje ceny.

- Step-aerobic je cvičení, které má vazbu na Aerobic, ale nemá vlastní zdroje, protože tato služba je poskytována v prostorách určených pro Aerobic (stejně jako ostatní cvičení). Cvičení definuje cenu, účtovací periodu.

2.8.5 Třída Uživatel

Systém potřebuje několik úrovní/rolí uživatelů. Obecně lze uživatele rozdělit do tří skupin:

- Zákazníci mohou být dále děleni na registrované a neregistrované. Neregistrovaní ovšem nemají na systém vliv, protože se systémem přímo nepracují, jejich vliv na systém je realizován skrze obsluhu. Registrovaní uživatelé už systém ovlivňují přímo. Možnosti zákazníků v systému jsou však omezeny na práci s vlastními rezervacemi. I množství informací jim poskytovaných je omezeno na základě nastavení centra, v základu vidí pouze své rezervace a hodiny cvičení. Registrovaný uživatel může mít zvýšena práva v systému, kdy se ho netýkají některá omezení (například omezení počtu rezervací na týden, možnost ručního výběru zdroje služby). Tento uživatel je označován jako super-uživatel.

- Obsluhu lze rozdělit na dvě skupiny. První je instruktor cvičení, který má stejná práva jako zákazník, jediným rozdílem je množství dat, které může vidět – instruktor vidí více podrobností v hodinách cvičení a to včetně všech rezervací. Další úrovní je operátor, který většinou odpovídá obsluze centra na recepci. Uživatel s touto rolí má přístup k databázi uživatelů, vidí všechny rezervace a může s nimi otevřeně pracovat. V případě práce s uživateli je možné jeho práva omezit o činnosti s penězi (zadávání permanentek, prodávání kreditu).

- Administrátor podniku spravuje celé nastavení. Vytváří a konfiguruje služby a jejich zdroje. Udává délky intervalů rezervací a ceny za využívání služeb. Pouze on má přístup ke statistickým informacím o využití služeb a celkovým tržbám.

Třída uživatel dále obsahuje informace o stavu zákazníka k systému, tyto jsou:

- Aktivní: uživatelský účet je aktivní a je oprávněn k využívání služeb centra.

- Suspendovaný: účet uživatele byl pozastaven z důvodů porušení pravidel (do tohoto stavu se uživatel dostane buď přímým zásahem obsluhy nebo při konkrétním nastavení systému automaticky).

- Registrace: uživatel se zaregistroval, ale jeho účet ještě nebyl aktivován obsluhou (systém je možné nastavit, aby k aktivaci účtu došlo automaticky po potvrzení emailem nebo SMS kódem).

2.8.6 Třída Rezervace

Rezervace je centrálním stavebním kamenem, protože právě ona je podstatou celého systému. Rezervace lze v první řadě dělit podle jejich verze:

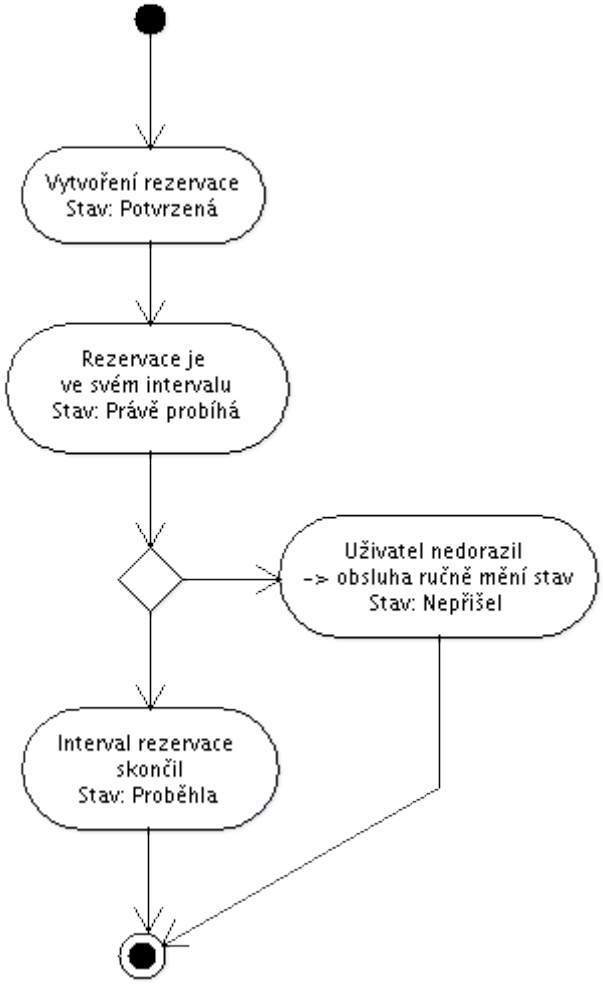
- Rezervace běžná: určená pro zaznamenání obsazení zdroje některým ze zákazníků (nezáleží, zda zákazníkem registrovaným či neregistrovaným). Obsahuje informaci o službě, kterou rezervuje, času, na který rezervuje, a uživateli, pro kterého rezervuje. Dále zahrnuje informace o ceně a způsobu platby.

- Rezervace vícenásobná: tato rezervace je vyjádřením hodiny cvičení, na které se zákazníci mohou rezervovat. Tuto verzi smí zadávat pouze obsluha. Tato verze obsahuje navíc oproti běžné rezervaci i informace o celkové kapacitě a aktuálnímu stavu obsazení.

- Rezervace na cvičení: tato rezervace se napojuje na vícenásobnou rezervaci. Má dva rozdíly oproti běžné rezervaci a to, že neobsahuje vazbu na službu ani na čas, protože toto přebírá z vícenásobné rezervace, na kterou má vazbu.

Mezi vícenásobnou rezervací a rezervací na cvičení je úzký vztah -> vícenásobná rezervace se provádí na class-službu a je prováděna obsluhou centra – tato rezervace má znaky klasické rezervace, ale s tím, že není zpoplatněna, protože uživatelem je zaměstnanec centra. Rezervace na cvičení je fakticky vázána pouze na cvičení a je spjata s koncovým zákazníkem, ta je vázána na prostor (tzn. na class-službu) zprostředkovaně přes vícenásobnou rezervaci.

Každá rezervace si drží informaci o svém stavu, který určuje její životní cyklus. Jsou to tyto:



Ilustrace 8: Pozitivní nastavení měnění stavů rezervace

- Požadavek: zákazník zadal rezervaci a rezervace čeká na potvrzení obsluhou (systém dovoluje nastavit automatické potvrzování zákaznických rezervací)
- Zamítnutá: požadavek na rezervaci byl obsluhou zamítnut.
- Potvrzená: rezervace byla obsluhou potvrzena a čeká na její čas.
- Právě probíhá: rezervace právě probíhá, do toho stavu systém v závislosti na nastavení sám přepíná rezervace, když je čas odpovídající jejich intervalu.
- Proběhla: rezervace v pořádku proběhla a její cena byla zahrnuta do tržeb a podle nastavení byl zákazník odměněn ve věrnostním programu.
- Zrušená: rezervace byla zrušena před přechodem do Právě probíhá a to buď zákazníkem v lhůtě před jejím začátkem nastaveném centrem nebo obsluhou kdykoliv před začátkem. Pokud rezervaci uživatel nestihne zrušit, musí situaci řešit s obsluhou centra.
- Nepřišel: uživatel se na rezervaci nedostavil a z tohoto důvodu byly vyvozeny důsledky dle nastavení systému.

V zásadě existují dva základní způsoby nastavení chování systému – pozitivní a negativní.

Pozitivní předpokládá, že uživatelé dodržují své rezervace a automaticky přepíná stavy rezervací v tomto pořadí:

Potvrzená -> Právě probíhá -> Proběhla

V případě, že se uživatel nedostaví, obsluha změní stav rezervace na Nepřišel ručně.

Negativní je přesně opačně, kdy rezervace je automaticky přepnuta do stavu Nepřišel, pokud se uživatel nedostaví a rezervace není obsluhou změněna na Právě probíhá.

2.9 Definice Controlleru v aplikaci Bizzy

Controller je komponenta MVC vzoru, která reaguje na události (většinou pocházející od uživatele) a podle nich vykonávající změny v komponentách Model a View - poskytující spojení mezi komponentami Modelem a View.

Controller ve webové aplikaci zastává tyto funkce:

- přijímá http požadavky z klientského prohlížeče
- překládá požadavky na konkrétní podobu prováděné operace
- vyvolává danou metodu, nebo ji předává správci
- volí pohled, který má být uživateli zaslán
- vrací pohled uživateli.

Systém využívá funkcionality aplikačního rámce Struts, který poskytuje rozhraní a implementaci controlleru ve třídě ActionServlet.

Objekt ActionServletu při své inicializaci nahraje mapování URL zasílaných klientským prohlížečem na potomky třídy Action z XML konfiguračního souboru. Objekty třídy Action obsahují vlastní aplikační logiku.

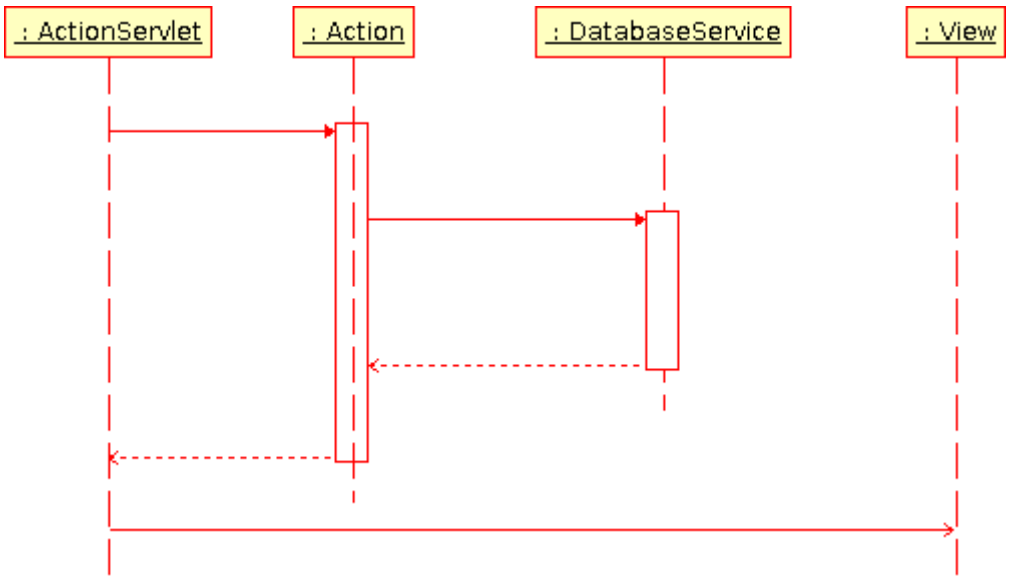
V systému je třída ActionServlet rozšířena o přístup k databázovým zdrojům a modelu, dále o inicializaci vláken využívaných k automatické práci s rezervacemi a pro odesílání upozornění.

Reálně komunikace v controlleru probíhá následně:

1. Klient zavolá adresu na serveru, ActionServlet podle konfigurace rozhodne, jakou akci má vykonat, inicializuje ji a zavolá metodu execute().

2. Action ve své metodě execute vykoná potřebný kód (případná data uloží do objektu požadavku nebo session) a vrátí ActionServletu informaci o přesměrování na view.

3. ActionServlet podle konfigurace vybere JSP soubor, který následně vykreslí data.



Ilustrace 9: Sekvence zpracování požadavku klienta

2.10 Definice View v aplikaci Bizzy

Komponenta view je v aplikaci Bizzy řešena pomocí JSP stránek.

Java Server Pages (dále JSP) je technologie společnosti Sun určená pro dynamické generování HTML, XML a jiných dokumentů pro webové klienty. Tato technologie vznikla pro usnadnění práce webových designerů a pro přehlednější oddělení aplikační logiky od vzhledu.

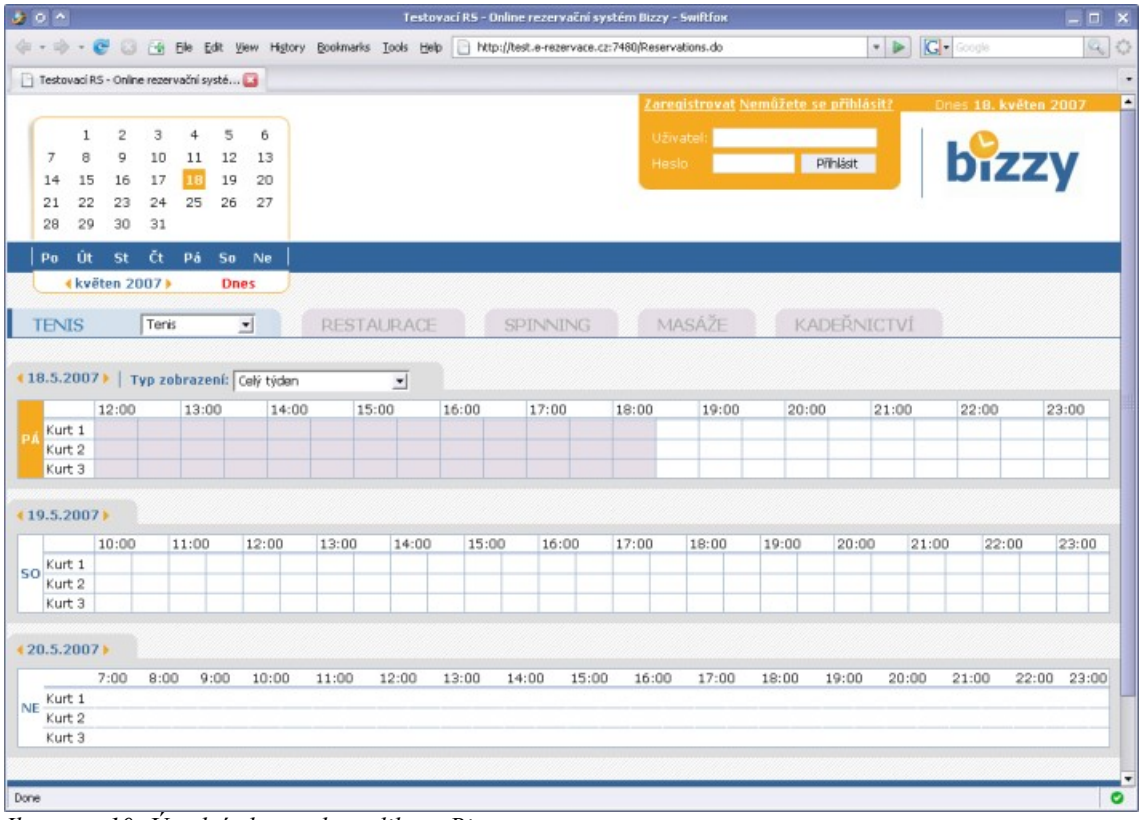
Vlastní JSP je textový dokument, v kterém jsou využity JSP tagy. Tento soubor je ve chvíli spuštění přeložen do podoby Java kódu, který je následně spuštěn.

Základní sadu tagů JSP lze rozšiřovat pomocí uživatelských tagů – třídy v jazyce Java dědící rozhraní ze základní třídy tagů. Skupiny takto vytvořených tagů se nazývají knihovny, které lze následně importovat do kterékoliv aplikace.

V aplikaci Bizzy jsou využívány tyto knihovny tagů třetích stran:

- JSTL: knihovna poskytující základní logické a iterační tagy. (autor: Sun)
- Struts taglibrary: poskytující tagy pro zjednodušení práce (komunikace) s aplikačním rámcem Struts. (autor: Apache)
- Display tag library: umožňující tvořit HTML tabulky s možností třídění dat a exportu dat do různých formátů. (autor: the Displaytag team)
- Tiles tags library: knihovna tagu určená pro definování šablon rámce Tiles

Pro organizaci a skládání jednotlivých JSP stránek do kompletního vzhledu je použito šablonovacího rámce Tiles.



Ilustrace 10: Úvodní obrazovka aplikace Bizzy

Šablona je JSP stránka, která užívá JSP uživatelské tagy k popisu rozložení stránky. Šablona funguje jako definice toho, jak budou stránky aplikace vypadat, aniž by určovala jejich obsah. Obsah je do této stránky vkládán až za běhu. Stejnou šablonu může využívat více stránek. Smyslem šablony je soudržný vzhled aplikace bez nutnosti programování každé stránky zvlášť.

Tiles je šablonovací systém umožňující skládání obsahu z jednotlivých komponent. Pomocí jeho funkcionality je možné vytvořit rozložení a za běhu aplikace do něj vkládat obsah. [CAVANESS 2003] Hlavní rysy rámce jsou tyto:

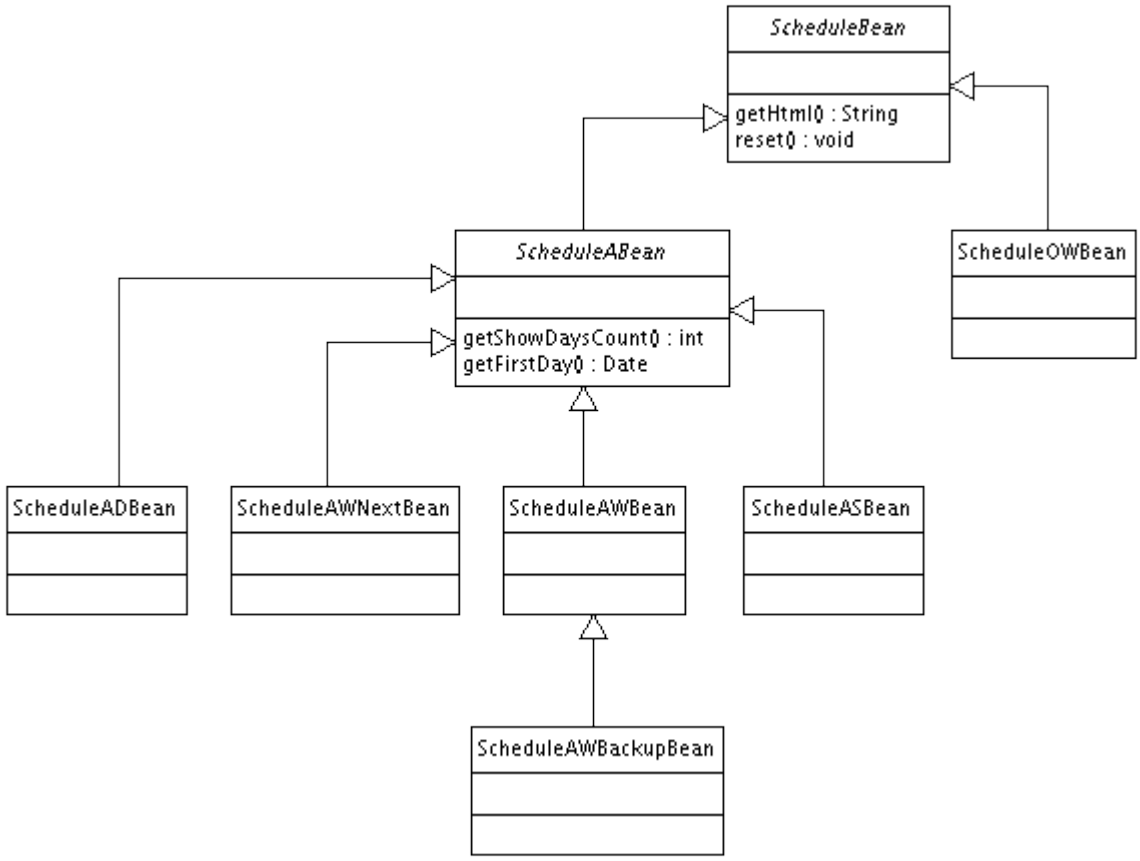
- Dynamické vytváření a nahrávání stránek
- Definice obrazovky
- Podpora a znovupoužití dlaždic (jednotlivé celky v rozvržení celé obrazovky)
- Podpora dědičnosti rozložení
- Podpora internacionalizace

2.10.1 Zobrazení rozvrhů

Pro zobrazovaní rozvrhů služeb bylo nakonec upuštěno od použití čistě JSP stránek a kód pro jejich generování byl přenesen do speciálních tříd. Důvodem pro toto rozhodnutí je problematická optimalizace HTML výstupu – vlastní rozvrhy jsou celkem rozsáhlé tabulky a JSP překladač bohužel často vkládá mnoho bílých znaků, které u výsledného HTML neúměrně zvyšují velikost.

Základem celého vykreslování je abstraktní třída **ScheduleBean**, která získá data z databáze pomocí datové služby a upraví je potřeby zobrazení.

Z třídy ScheduleBean dědí 2 třídy přímo, jsou to ScheduleABean a ScheduleOWBean.



Ilustrace 11: Třídy pro generování rozvrhů

ScheduleABean je abstraktní třída rozšiřující ScheduleBean, určená pro zobrazování jednoho zdroje na jeden den v jedné tabulce. Její potomci potom pouze definují první den, který má být zobrazen a počet dnů, které mají být zobrazeny. Zobrazení používající funkčnost této třídy jsou:

- **Jeden den**: je zobrazen jeden rozvrh se všemi zdroji jedné služby na jeden den. Třída: ScheduleADBean.

- **Týden**: je zobrazeno 7 rozvrhů na jeden den (možné nastavení, kdy se zobrazují pouze dny do konce týdne) se všemi zdroji jedné služby. Třída: ScheduleAWBean.

- **Dnes + 7 dní**: který je podobný týdennímu, pouze s tím rozdílem, že je zobrazen jeden den navíc (vhodné zobrazení, pokud uživatelé rezervují rezervace v týdenních periodách). Třída: ScheduleAWNextBean.

- **Jeden den všechny služby**: zobrazí denní rozvrh pro každou běžnou a class službu. Třída: ScheduleASBean.

- **Zálohovací pohled**: je velmi podobný jako „Jeden den všechny služby“ pouze je ochuzen o všechny aktivní prvky (přepínání mezi pohledy atd.), protože je získáván a ukládán zálohovacím softwarem na pevný disk centra pro případ výpadku Internetu. Třída: ScheduleAWBackupBean.

ScheduleOWBean naopak přímo implementuje plnou funkčnost. Jejím výsledkem je **detailní pohled -** jeden zdroj jedné služby je zobrazen v jedné tabulce na celý týden. Toto zobrazení je nejčastěji využíváno pro Class-služby, které nemívají více zdrojů.

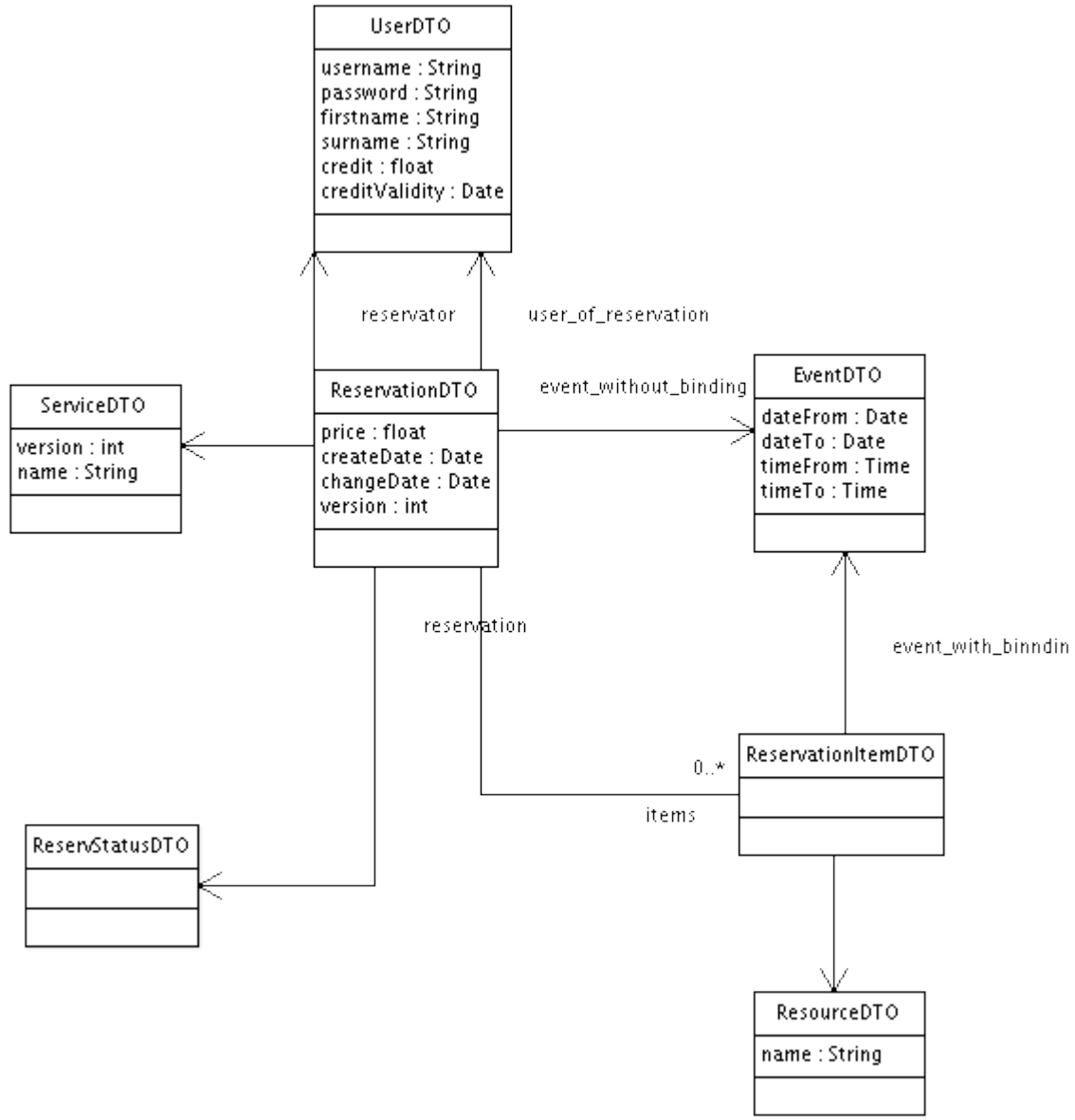
Všechna tato zobrazení využívají uspořádání, kdy je časová osa umístěna horizontálně, na žádost klientů je vyvíjeno vertikální zobrazení, které je používáno ve většině plánovacích systémech (většina PIM aplikací jako je například Microsoft Outlook).

3 Popis jednotlivých případů užití

3.1 Zobrazení rozvrhů s rezervacemi

Zobrazení rozvrhů s rezervacemi je základní funkce rezervačního systému. Rozvrh je první část aplikace, které si uživatel všimne, proto je velmi důležitý jeho vzhled a intuitivní funkcionalita.

Základní dekompozice jednotlivých zobrazení je popsána výše, v této části je popsán obecný princip generování rozvrhů. Pro pochopení vytváření rozvrhů je třeba vysvětlit skutečnou strukturu rezervace.



Ilustrace 12: Třída ReservationDTO a jeho okolí

Rezervace (objekt ReservationDTO) je kvůli jednoduššímu výběru z databáze

rozdělena na několik objektů. Informace o čase je oddělena do samostatného objektu EventDTO. Vazba na zdroj je v objektu ReservationItemDTO.

Do potomka třídy jsou načteny příslušné objekty typu EventDTO podle objektu ReservationItemDTO. Následně je vygenerováno pole objektů ScheduleIntervalItem, které obsahuje jak čistě tyto objekty, které odpovídají základnímu časovému intervalu definovanému službou, tak jejich potomky OneResIntervalItem, které odpovídají časovým úsekům jednotlivých rezervací. Následně skrze metodu getHtml() třídy ScheduleBean jsou hierarchicky volány metody pro generování HTML kódu jednotlivých položek.

Rezervace lze zobrazovat i v seznamu, který lze filtrovat dle uživatele, času a stavu. Dále je možné ze seznamu rezervacím měnit status a to jak jedné rezervaci, tak i skupinám rezervací. Takto filtrovat a upravovat smí pouze obsluha, zákazník v seznamu vidí pouze své rezervace a to pouze aktivní – rezervace, které ještě nebyly ukončeny (stav Požadavek, Potvrzená, Náhradník).

3.2 Správa rezervací

Správa a obecně práce s rezervacemi je další základní funkce aplikace Bizzy. Uživatelé mohou vytvářet, upravovat, mazat rezervace. Nyní si vysvětlíme jednotlivé činnosti.

3.2.1 Vytváření rezervací

Rezervace jsou oprávněni vytvářet všichni registrovaní uživatelé. Rozdíl je v tom, že uživatel má několik omezení:

- Doba, na jak dlouho dopředu smí uživatel zadávat rezervace, tato volba je nastavena na úrovni služby.
- Minimální lhůta před začátkem rezervace, kdy je ještě možné rezervaci vytvořit. Nastaveno na úrovni služby.
- Množství rezervací, které může uživatel udělat za 24 hodin, tato volba je nastavena na úrovni služby.
- Služba je dostupná uživatelům, nastaveno na úrovni služby. Takto

neoznačená služba nemá pro zákazníky viditelný ani rozvrh. Často používáno například pro stoly v restauraci, které jsou sledovány pouze interně.

- Omezení rezervování pouze na uživatele s kreditem nebo permanentkou. Nastaveno obecně na úrovni podniku.

Dále je uživatel omezen i v tom, jak může vytvářet rezervace, opět tato omezení neplatí pro obsluhu.

- Minimální a maximální délka rezervace, kterou může uživatel zadat. Nastavováno na úrovni služby.

- U běžné rezervace, zda může uživatel rezervovat více zdrojů, a u rezervace cvičení, kolik míst si může uživatel maximálně rezervovat. Nastavováno na úrovni služby.

Vytváření rezervace je rozděleno do tří kroků: inicializace objektu rezervace a jejích asociací, validace uživatelských dat, uložení do databáze.

1. Ve fázi inicializace dojde k naplnění objektu rezervace přednastavenými hodnotami. V případě, že rezervaci zadává zákazník, doplní se do rezervace jeho údaje. Dále je vypočítána cena.

2. Ve fázi validace systém přijme a zjistí správnost dat. Pokud jsou data v pořádku, dojde k následující fázi. Pokud jsou data nesprávná, je vrácen editační formulář s konkrétní informací o problematických údajích.

3. Fáze uložení dat do databáze předá objekt rezervace se všemi asociacemi databázové službě, která ho pomocí rámce OJB uloží do databáze.

3.2.2 Úprava rezervací

Úprava rezervace probíhá podobně jako vkládání, pouze místo fáze inicializace dojde k načtení rezervace z databáze. I všechna omezení pro uživatele jsou stejná.

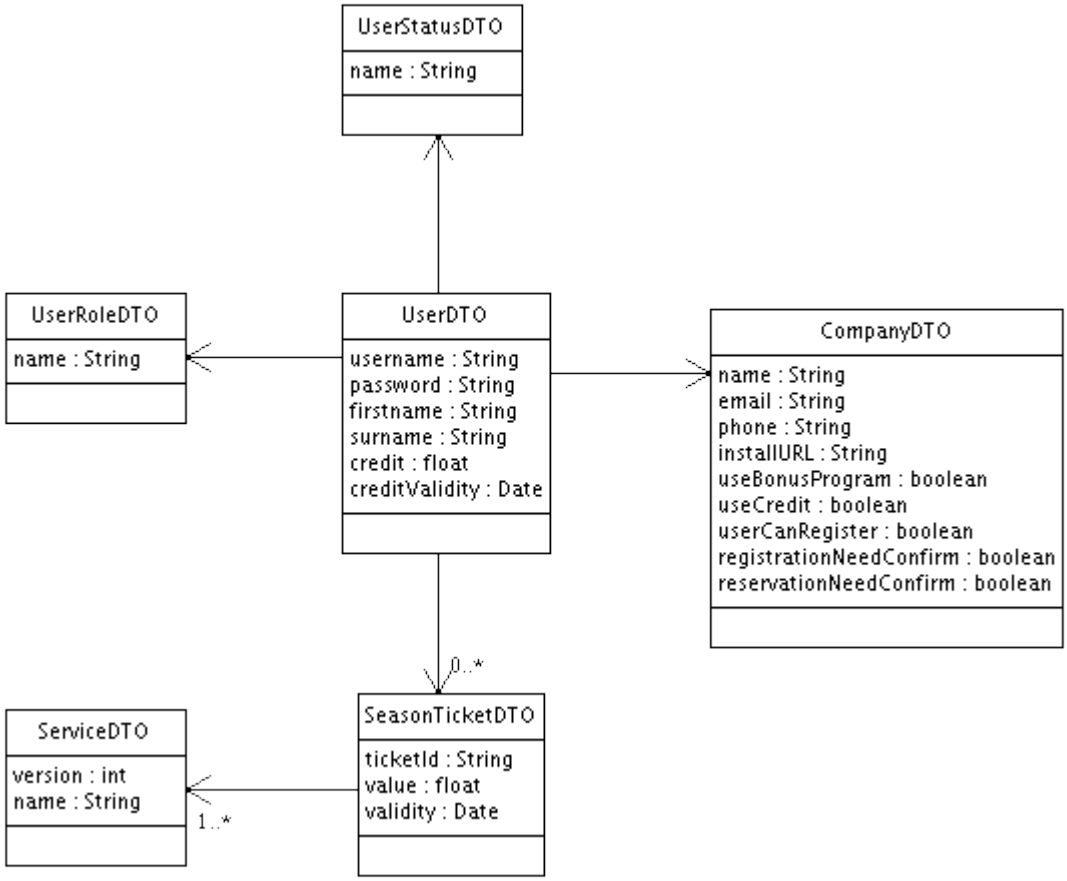
3.2.3 Mazání rezervací

Rezervace nejsou fyzicky mazány z databáze, pouze jim je přiřazen status Zrušené.

Tato akce se liší podle toho, zda ji provádí zákazník nebo obsluha. Zákazník je omezen dobou do začátku rezervace, po kterou může sám rezervaci zrušit, po této době se musí obrátit na obsluhu, která jeho rezervaci zruší. Pozdě zrušená rezervace je zaznamenána do uživatelského účtu.

!V současné době se vlastní rezervace nijak neoznačuje, ale je rozpracován nový status rezervace, který bude vyjadřovat nedostavení se s omluvou.

3.3 Správa uživatelů / osobních údajů



Ilustrace 13: Třída UserDTO a jeho okolí

3.3.1 Seznamy uživatelů

Obsluha má možnost vypsat si seznam uživatelů a ten dále filtrovat podle role, statusu, cenové hladiny a využívání služeb.

V tomto seznamu lze i vyhledávat a to podle jména, telefonu, emailu a interního čísla

centra. Toto hledání je pravostranné – uživatel nemusí zadávat celý hledaný text, ale pouze začátek.

Ze seznamu uživatelů lze i měnit statusy uživatelů a to jak pro jednotlivce, tak pro vybranou skupinu. Podobně jako změna statusu se dají vybírat uživatelé, kterým má být zaslán email nebo SMS.

3.3.2 Vytváření uživatelů

Uživatelské účty se vytváří dvěma způsoby:

Uživatel sám vyplní své osobní údaje a odešle je na server, kde se vytvoří účet a ten je deaktivován, dokud není povolen obsluhou centra a zároveň je mu odeslán potvrzovací email (který je v případě nastavení doplněn o odkaz, po jehož odkliknutí je účtu potvrzena emailová adresa (potvrzovat údaje lze i pomocí mobilního telefonu, kdy je formou SMS odeslán kontrolní kód, který je nutno do systému zadat do hodiny od jeho vygenerování). Tato funkce je volitelná a centrum ji může zakázat a povolit pouze druhý způsob.

Uživatel je registrován obsluhou, kdy jsou všechna data ověřena přímo zaměstnanci centra a účet je automaticky potvrzen.

3.3.3 Úprava uživatele

Uživatelský účet lze samozřejmě upravovat a to jak obsluhou, tak zákazník smí upravovat některá svá data (osobní údaje, kontaktní informace, přihlašovací údaje). Operátor zde může sledovat informace o útratě, množství návštěv.

3.3.4 Permanentky a kredit

Z formuláře editace uživatele jsou dostupné kredit uživatele a permanentky. Kredit uživatele slouží k úhradě služeb čerpaných uživatelem a k nákupu doplňkového zboží. Kredit je vyjádřen v peněžních jednotkách. Permanentka slouží pouze k úhradě služeb a obsahuje množství časových jednotek. Permanentku zákazník může mít více než jednu a to z toho důvodu, že permanentka může být omezena pouze na konkrétní službu či služby. Jak kreditu, tak permanentce je možné nastavit datum platnosti, po ukončení

této lhůty případný zůstatek propadne.

Kredit je evidován ve třídě UserDTO. Permanentka z důvodu vztahu k uživateli 1:n musí být v samostatné třídě SeasonTicketDTO.

3.3.5 Rušení uživatele

Uživatelský účet lze zrušit, ale nedojde k jeho fyzickému odebrání z databáze, ale pouze je mu změněn status na Zrušený. Toto chování vyplývá z potřeby existence vazeb v minulých rezervacích. Takovýto účet není zobrazen v seznamu uživatelů, ani není umožněno se přihlásit.

3.3.6 Rozesílání zpráv

Do správy uživatelů patří i možnost s nimi komunikovat – systém Bizzy obsahuje jednoduchý formulář pro vytváření HTML emailů s možností vyhledávat emailové adresy v uživatelích systému.

Další možností jak komunikovat s uživateli je skrze SMS bránu, která umožňuje zasílat zprávy do všech sítí v České republice, Slovenské republice a do Německa. SMS zprávy jsou samozřejmě zpoplatněny.

3.4 Nastavení centra

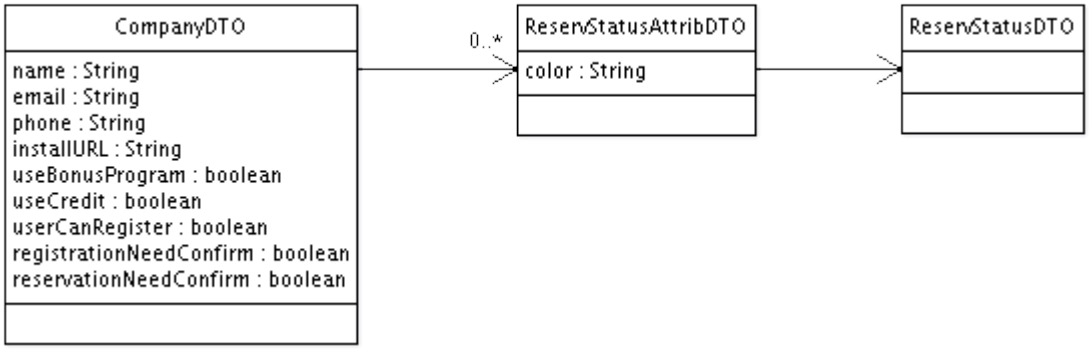
Centrum nastavuje pouze super-operátor. V této části se ukládají data přímo se týkající centra:

- Kontaktní informace centra: adresa, email, telefon.
- Možnosti zobrazení rozvrhů: barvy jednotlivých statusů rezervací (uložené v třídě ReservStatusDTO), zobrazení hlavičky systému (výběr pořadí loga a kalendáře), odlišné zobrazení rezervací vytvořených zákazníky, zobrazování rezervací zákazníkům v minulosti, zobrazení nepřihlášeným uživatelům (zda nepřihlášeným uživatelům zobrazovat obsazenost služeb).
- Nastavení pravidel registrování zákazníků – způsob potvrzování kontaktních informací.

- Nastavení pravidel rezervování – potřeba permanentky nebo kreditu, způsob přiřazování zdrojů (zákazník si nemůže ručně vybrat zdroj – je mu automaticky vybrán), povolení operátorům vytvářet rezervace v minulosti.

- Nastavení loga centra.

Nastavení centra se nalézá v třídě CompanyDTO (krom nastavení barev rezervací).



Ilustrace 14: Třída Centrum a jeho okolí

3.5 Správa služeb

Správu služeb je umožněna pouze uživatelům s rolí super-operátor. Jedná se o nejdůležitější část nastavení - obsahuje práci se službami, jejich zdroji a práci s ceníky.

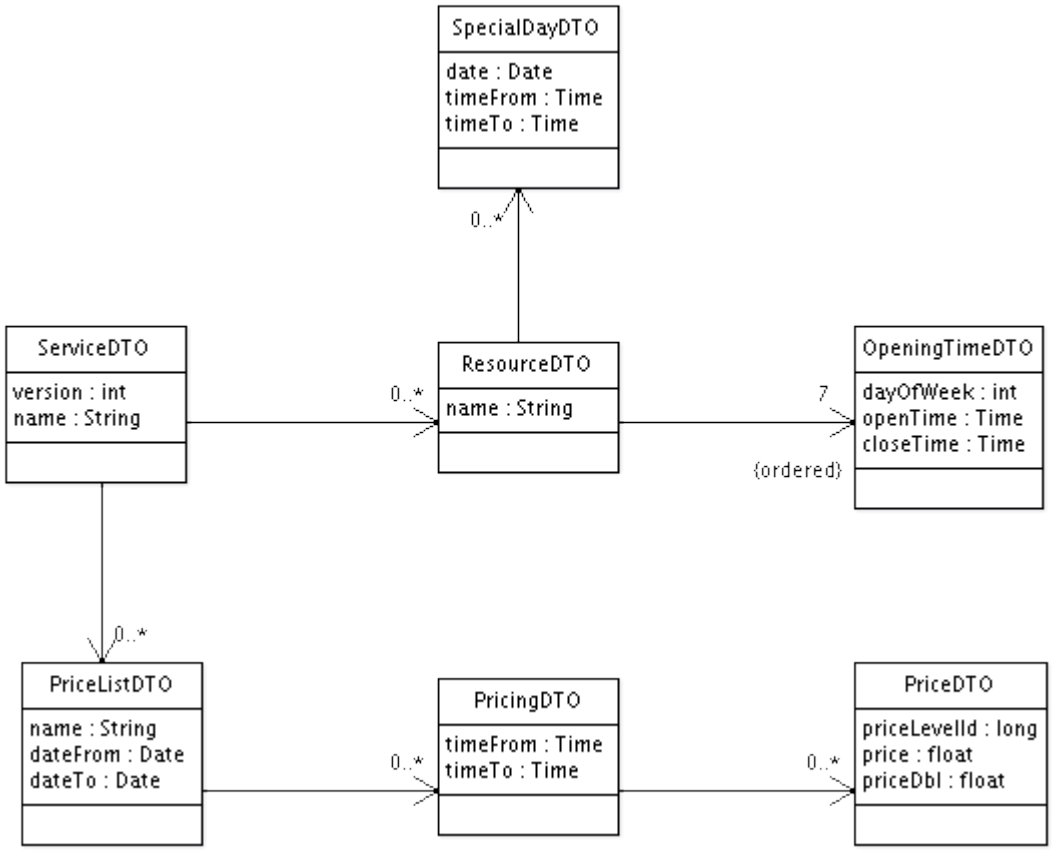
3.5.1 Nastavení služeb

V této části se vytvářejí, upravují a ruší objekty typu Služba. V případě rušení se objekt pouze označuje jako Zrušený a nedojde k fyzickému smazání z databáze – toto opatření bylo vytvořeno po několika omylem zrušených službách, kdy se problematicky obnovovala původní data.

Objekt ServiceDTO obsahuje i pravidla zobrazování služby v rozvrhu, délku intervalu a především atribut version, který určuje o jakou službu jde – jestli o běžnou, class-službu nebo cvičení.

3.5.2 Správa zdrojů

Služby mají své zdroje, které odpovídají fyzickým prostorám určeným pro provozování služby. Tyto zdroje nesou informaci o otevírací a zavírací době pro jednotlivé dny v týdnu, navíc lze nastavit konkrétní den v roce, kdy je otevírací doba odlišná, typicky používané pro svátky a volné dny.



Ilustrace 15: Třída ServiceDTO a její okolí

Zdroje lze vytvářet, upravovat a rušit, opět se neruší fyzicky z databáze, ale pouze se označují za smazané.

3.5.3 Správa ceníků

Služby mají návaznost na tvorbu ceny. Způsob oceňování rezervací je velmi důležitou částí rezervačního systému, a tak je i celkem složitý. Cena se musí měnit podle části roku i podle části dne, dále musí jít různým zákazníkům přiřazovat různé ceny.

Speciální možností v cenících jsou tzv. dodatečné náklady, které se zobrazují v rezervačním formuláři jako další dobrovolné navýšení ceny. Většinou se jedná o

dodatečné služby (v případě bowlingového centra se jedná např. o zapůjčení speciální bowlingové obuvi). Tyto náklady je možné zadávat jako pevnou částku na rezervaci, nebo na osobu, s kterou je v rezervaci počítáno. Tyto náklady jsou navázány na službu, takže je možné mít pro každou službu jinou.

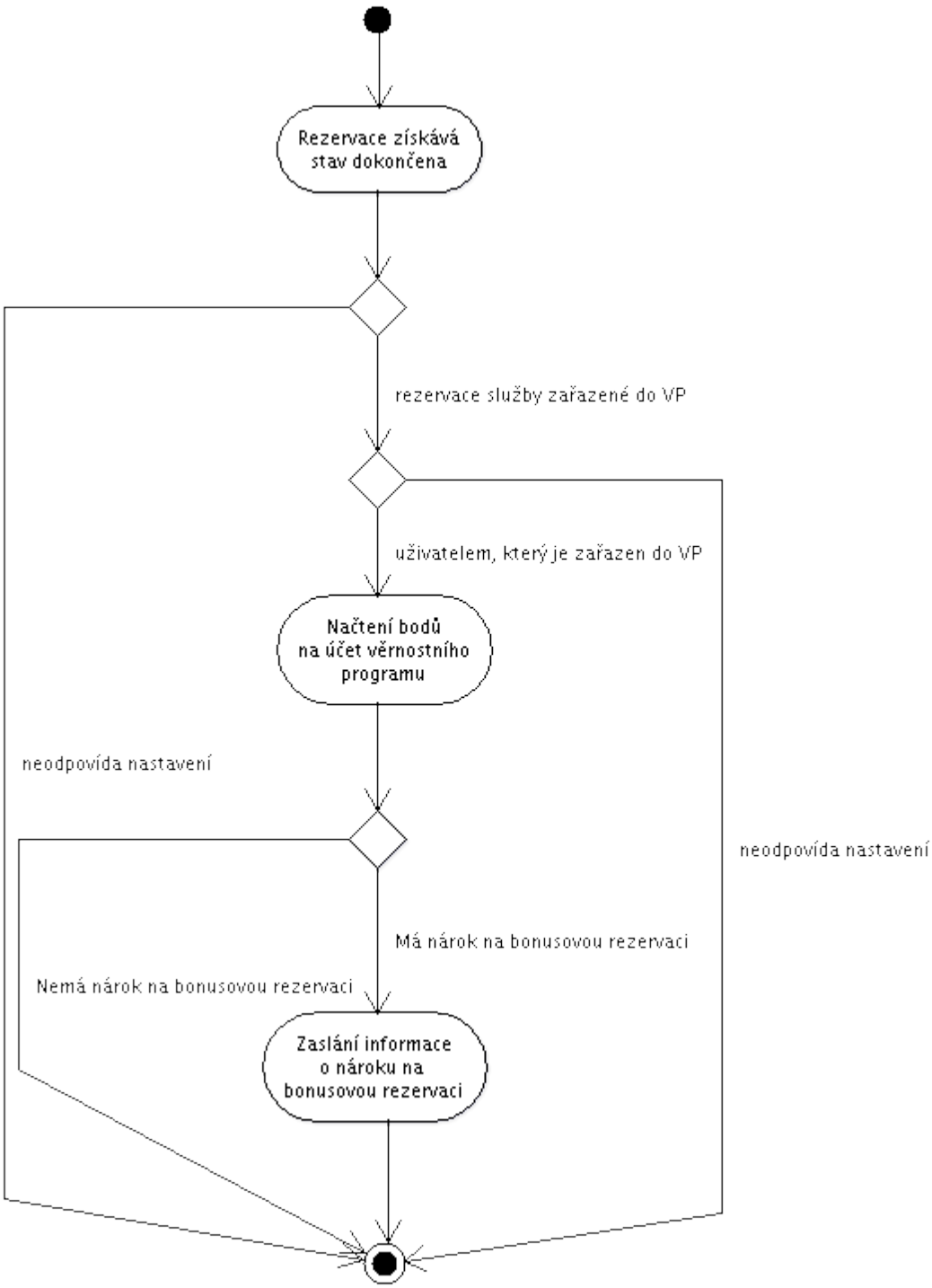
3.6 Nastavení věrnostního programu

Hlavním cílem věrnostního programu je odměňováním motivovat zákazníky k častějším návštěvám. Návštěvy zákazníka v centru jsou počítány a pokud dojde k dosažení určitého počtu návštěv, dojde k získání slevy z další rezervace (bonusová rezervace).

V případě, že je věrnostní program aktivní, započítávají se do něj všechny dokončené rezervace a to podle nastavení, buď jako peníze nebo jako využité intervaly. Dále se nastavuje procentní sleva z bonusové rezervace, jestli se započítávají pouze rezervace zadané zákazníkem a samozřejmě i podmínky, za kterých se zákazníkův věrnostní účet nuluje (body mohou být vymazány, pokud se uživatel nedostaví na rezervaci, nebo když požádá pozdě o zrušení rezervace).

Veškeré nastavení věrnostního programu se nalézá ve třídě CompanyDTO společně s ostatním nastavením centra. Body věrnostního programu jsou uchovávány v třídě UserDTO.

V současnosti se pracuje na bodovém věrnostním programu, kde odměnou nebude sleva na rezervaci, ale možnost získané body utratit za služby a zboží navolené centrem.



Ilustrace 16: Postup načítání bodů do VP při dokončení rezervace

3.7 Statistiky

Statistika se v současné době skládá ze dvou částí:

- Statistika rezervací: využití zdrojů a služeb, souhrn rezervací dle času, souhrn rezervací. Využití zdrojů a služeb vypíše tabulku, která obsahuje celkové využití zdrojů v hodinách a penězích. Souhrn rezervací dle času vyjadřuje využití zdrojů

v jednotlivých hodinách dne za daný časový úsek. Souhrn rezervací zobrazuje počty jednotlivých statusů rezervací za dané období.

- Statistika SMS zobrazující počty odeslaných zpráv v jednotlivých měsících (při každém odeslání SMS je načten bod).

V současnosti není možné statistiky exportovat do strojem čitelného formátu, ale tato možnost je do budoucna plánována. Dále je zvažována možnost generovat statistická data do grafické podoby grafu.

4 Závěr - definování nedostatků a návrhy řešení

Jak bylo uvedeno výše, systém Bizzy se vyvíjí již od roku 2004 a od té doby prochází živelným vývojem, který přerostl původní koncepci – množství funkcí aplikace se velmi zvýšilo (původně byl určen čistě pro evidenci rezervací) a došlo i k drobným změnám v zaměření (v dnešní době už má systém obecnější zaměření než na svém začátku – již není určen pouze pro čistě sportovní centra). Takovýto vývoj samozřejmě má i vedlejší nepříznivé efekty jako je nekoncepční řešení nových funkcí.

Z toho důvodu bude nutné vytvořit novou verzi systému, která bude mít alespoň základní kompatibilitu s datovým modelem předešlé verze. Cíle této nové verze jsou dva: modulárnost systému a zvýšení výkonu.

4.1 Modulárnost systému

Systém v současné době nepodporuje žádný způsob rozšiřitelnosti bez přímého a poměrně velmi složitého vstupu do kódu aplikace. [KRAVAL 2006]

Téměř každý klient má vlastní specializované požadavky a někteří jsou i ochotni si zaplatit novou funkci v systému. Problém v současnosti je, že tato funkce je dostupná všem centrům využívající systém Bizzy.

Řešením je rozdělit aplikaci na logické oddíly, pro které je nutné vytvořit modulární rámec, který dovolí aktivovat a zakazovat jednotlivé části. Je třeba navrhnout dostatečně pružné rozhraní, které dovolí povolovat jednotlivé moduly na základě nastavení klienta.

Rozdělení aplikace do jednotlivých modulů s určitou nezávislostí dále zpřehlední systém a umožní jednodušší vývoj ve skupině, protože jednotlivé moduly mohou komunikovat přes jasně dané rozhraní, díky kterému nemusí každý vývojář znát přesnou implementaci všech částí.

Toto rozdělení ale může způsobit značné problémy, protože způsobí změnu datového modelu, a tak bude třeba ho provést velmi citlivě, aby nedošlo k poškození současných dat. Existují tak pouze dvě možnosti a to buď zachovat datový model a přizpůsobit aplikaci, nebo pomocí konverzního scriptu převést původní data. Nejschůdnější cestou se jeví částečné využití obou metod.

4.2 Zvýšení výkonu systému

S nekoordinovaným vývojem souvisí i snížení výkonu a to způsobené jak špatnou optimalizací kódu, tak ne zrovna nejvýhodnější práci s databází a s databázovými transakcemi (v současnosti je tvořeno velké množství transakcí, protože systém vytváří transakci na každý dotaz).

4.2.1 Optimalizace komunikace s databází

Používaný O/R mapping rámec není v současnosti dále vyvíjen a ani opravován, a tak bude nutné využít nějaký nový. Jako nejlepší řešení se jeví rámec Hibernate (otevřený projekt zaštítěný společností RedHat), který je v dnešní době nejpoužívanější a nejspíš i v budoucnosti bude. Velkou výhodou je i rozsáhlá dokumentace a obecně lepší podpora ze strany vývojářů.

4.2.2 Změna způsobu práce s http dotazem

Rámec Struts interně pracuje s dotazem ne zrovna nejoptimálnějším způsobem, kdy dochází při předávání dotazu mezi jednotlivými akcemi k opakovanému volání správce http dotazu, který celou akci poměrně dost brzdí. Řešením by bylo změnit ho, ale to by již znamenalo předělat většinu aplikační logiky.

4.2.3 Využití testovacího rámce

Rámec Struts má pouze omezené možnosti testování, protože vlastní aplikační logika je silně svázána s API servletového kontejneru, z čehož vyplývá nutnost simulovat při testu toto prostředí. Proto je velmi složité psát automatické testy. I tento problém by bylo nutné řešit přechodem na jiný MVC rámec.

Jako testovací rámec je možné využít JUnit, který je v současnosti velmi dobře integrován do vývojových prostředí a stal se takříkajíc standardem pro automatické testy. [HYNAR 2004]

4.2.4 Více komentovaného kódu

V současnosti je množství komentářů v kódu v aplikaci Bizzy absolutně nedostačující. V případě přepisování rozsáhlejších celků je nutno více dbát na

programátorskou dokumentaci a řídit se metodikou nástroje JavaDoc, která slouží pro generování manuálu z komentářů přímo v kódu do podoby HTML.

4.3 Osobní závěr

Velmi mě překvapila rozsáhlost a složitost systému, která není při běžné práci tak jasně patrná, jako když jsem se snažil vytvořit zpětně dokumentaci. Problém je v tom, že programátor při plnění konkrétního úkolu nepracuje se systémem jako celkem, ale věnuje pozornost pouze konkrétní části, kterou upravuje. Sám jsem byl velmi překvapen některými detaily tohoto systému, které jsem zatím takto do detailu nestudoval.

Dále jsem si uvědomil důležitost vypracovávání použitelné dokumentace průběžně během tvorby systému – její zpětné vytváření je velmi obtížné.

Seznam použité literatury

ARLOW, Jim a NEUSTADT, Ila. *UML a unifikovaný proces vývoje aplikací : Průvodce analýzou a návrhem objektově orientovaného softwaru*. Brno : Computer Press, 2003. 387 s. ISBN 80-7226-947-X

BAUER, Christian, KING, Gavin. Get started with Hibernate : introducing and configuring Hibernate. *JavaWorld* [online]. 2004 [cit. 2007-03-20]. Dostupný z WWW: <http://www.javaworld.com/javaworld/jw-10-2004/jw-1018-hibernate.html>.

BUCHALCEVOVÁ, Alena a PAVLÍČKOVÁ, Jarmila. *Základy softwarového inženýrství – objektově orientovaný přístup*. 1. vyd. Praha: Vysoká škola ekonomická, 2002. 216 s. ISBN 80-245-0268-2

CAVANESS, Chuck. *Programujeme Jakarta Struts : Tvorba webových aplikací se servlety a stránkami JSP*. Přeložil Slavoj Písek. 1. vyd. Praha : Grada, 2003. 446 s. ISBN 80-247-0667-9.

FOWLER, Martin. *Refaktoring : zlepšení existujícího kódu*. Přeložil Vladimír Lahoda. 1. vyd. Praha : Grada, 2003. 396 s. ISBN 80-247-0299-1.

GAMMA, Erich, HELM, Richard, JOHNSON, Ralph. *Návrh programů pomocí vzorů : stavební kameny objektově orientovaných programů.* Přeložil Pavel Makovec. 1. vyd. Praha : Grada Publishing a.s., 2003. 388 s. Moderní programování. ISBN 80-247-0302-5.

HYNAR, Martin. *Java : nástroje.* 1. vyd. Praha : Neocortex, 2004. 325 s. ISBN 80-86330-16-8.

KANISOVÁ, Hana, MÜLLER, Miroslav. *UML srozumitelně*. 2. aktualiz. vyd. Praha : Computer Press, 2006. 176 s. ISBN 8025110834.

KRAVAL, Ilja. Jedna z velmi častých a závažných chyb při návrhu IS aneb jak vznikají tzv. "molochální systémy" : část 1. *0bjects.cz* [online]. 2006 [cit. 2007-04-22]. Dostupný z WWW: <http://www.objects.cz/clanky/clanek19/chyba_molochy1.pdf>.

KRAVAL, Ilja. Jedna z velmi častých a závažných chyb při návrhu IS aneb jak vznikají tzv. "molochální systémy": část 2. *0bjects.cz* [online]. 2006 [cit. 2007-04-22]. Dostupný z WWW: <http://www.objects.cz/clanky/clanek20/chyba_molochyII.pdf>.

KRAVAL, Ilja. *Návrh IS pomocí OOP, UML a vzorů* [online]. [2006] , 2006-03-05 [cit. 2006-12-26]. Dostupný z WWW: <http://www.objects.cz/clanky/clanky_IS/NavrhIS.pdf>.

KRAVAL, Ilja. Nejčastější chyby při modelování I.. *Databázový svět* [online]. 2004 [cit. 2007-04-20]. Dostupný z WWW: <http://www.dbsvet.cz/view.php?cisloclanku=2004080401>.

KRAVAL, Ilja. Nejčastější chyby při modelování II.. *Databázový svět* [online]. 2004 [cit. 2007-04-20]. Dostupný z WWW: <http://www.dbsvet.cz/view.php?cisloclanku=2004081201>.

Model-view-controller. *Wikipedia* [online]. 2007 [cit. 2007-04-20]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Model-view-controller>.

PAGE-JONES, Meilir. *Základy objektově orientovaného návrhu v UML.* Přeložil Karel Voráček. 1. vyd. Praha : Grada, 2001. 367 s. Moderní programování. ISBN 80-247-0210-X.

PALETA, Petr. *Co programátory ve škole neučí : Softwarové inženýrství v reálné praxi.* 1. vyd. Praha : Computer Press, 2003. 331 s. ISBN 80-2

PICHLÍK, Roman. Web frameworky v Jave. *Dagblog : blog nejen pro kodery* [online]. 2006 [cit. 2007-04-10]. Dostupný z WWW: <http://www.sweb.cz/pichlik/archive/2006_12_10_archive.html#116578886450903306>.

POLÁK, Jiří, MERUNKA, Vojtěch a CARDA, Antonín. *Umění systémového návrhu : Objektově orientovaná tvorba informačních systémů pomocí původní metody BORM.* Praha : Grada, 2003. 195 s. ISBN 80-247-0424-2

RIORDMAN, Rebecca. *Vytváříme relační databázové aplikace.* 1. vyd. Praha : Computer Press, 2000. 280 s. ISBN 80-7226-360-9.

SCHMULLER, Joseph. *Myslíme v jazyku UML.* Praha: Grada, 2001. 359 s. ISBN 80-247-0029-8

STEIN, René. Návrh aplikací v jazyce UML - Unified Modeling Language. *Interval.cz* [online]. 2003 [cit. 2007-04-20]. Dostupný z WWW: <http://interval.cz/clanky/navrh-aplikaci-v-jazyce-uml-unified-modeling-language/>. ISSN 1212-8651.

Sun Microsystems. Core J2EE Patterns - Data Access Object. Core J2EE Pattern Catalog [online]. 2001 [cit. 2007-05-10]. Dostupný z WWW: <http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html>.

Sun Microsystems. JavaBeans [online]. c1994-2007 [cit. 2007-04-20]. Dostupný z WWW: <http://java.sun.com/products/javabeans/>.

ŠEŠERA, Ľubor, MIČOVSKÝ, Aleš a ČERVEŇ, Juraj. *Datové modelování v příkladech*. Praha : Grada, 2001. 152 s. ISBN 80-247-0049-2

The Apache Software Foundation. *Apache ObJectRelationalBridge* [online]. c2006 [cit. 2007-04-10]. Dostupný z WWW: <http://db.apache.org/ojb/>.

The Apache Software Foundation. *Apache Tomcat* [online]. c1999-2007 [cit. 2007-04-20]. Dostupný z WWW: <http://tomcat.apache.org/>.

The Apache Software Foundation. *Struts* [online]. c2000-2007 [cit. 2007-04-20]. Dostupný z WWW: <http://struts.apache.org/>.

TheServerSide : your enterprise java community [online]. c2007 [cit. 2007-04-20]. Dostupný z WWW: <http://www.theserverside.com/>.

VRANA, Ivan a RICHTA, Karel. *Zásady a postupy zavádění podnikových informačních systémů : Praktická příručka pro podnikové manažery*. 1. vyd. Praha : Grada, 2005. 187 s. ISBN 80-247-1103-6

Evidence výpůjček

Prohlášení:
Dávám svolení k půjčování této bakalářské práce. Uživatel potvrzuje svým podpisem, že bude tuto práci řádně citovat v seznamu použité literatury.

V Praze, 23. 5. 2007

Štěpán Petrů

Jméno	Katedra / Pracoviště	Datum	Podpis